CSC2222: COMPUTER SYSTEMS II

M S HIRUNI PEIRIS
DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF RUHUNA

COMPUTER ARITHMETIC

LEARNING OBJECTIVES

At the end of this lecture you will be able to

- Explain ALU operations
- Demonstrate sign-magnitude and 2's complement numerical data representation and their features
- Explain the processes of addition, subtraction,
- Employ addition, subtraction algorithms

ARITHMETIC AND LOGIC UNIT

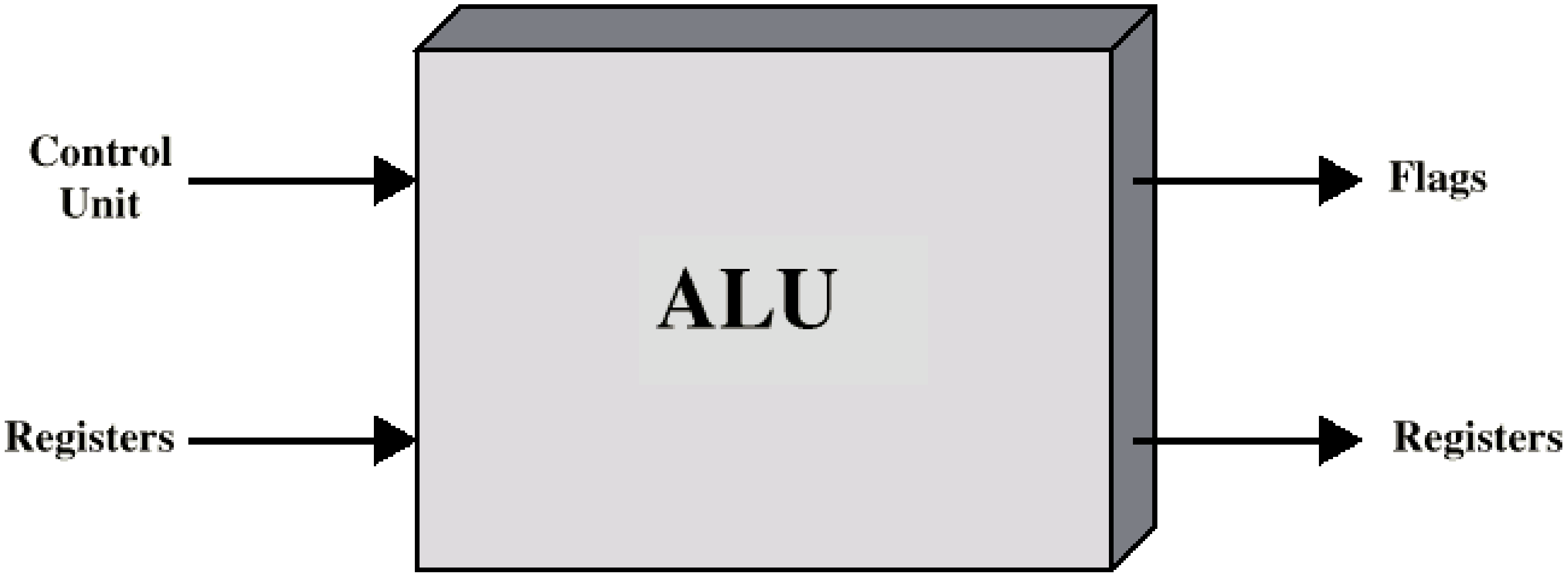
- Performs the calculations

- Everything else in the computer is there to service this unit

- Handles integers

- May handle floating point (real) numbers

ALU INPUTS AND OUTPUTS

- 

INTEGER REPRESENTATION

- Only have 0 & 1 to represent everything
- Positive numbers stored in binary
  - e.g. 41=00101001
- Sign-Magnitude
- Two's compliment

SIGN MAGNITUDE

- Left most bit is sign bit
  - 0 means positive
  - 1 means negative
- +18 = 00010010
- -18 = 10010010
- Problems
  - Need to consider both sign and magnitude in arithmetic
  - Two representations of zero (+0 and -0)

TWO'S COMPLEMENT

- Generally, there are two types of complement of Binary number:
  - 1's complement
  - 2's complement.
- To get 1's complement of a binary number, simply invert the given number.
- For example, 1's complement of binary number 110010 is 001101.

peirismsh@dcs.ruh.ac.lk

TWO'S COMPLEMENT

- There is a simple algorithm to convert a binary number into 2's complement.
- To get 2's complement of a binary number, simply invert the given number and add 1 to the least significant bit (LSB) of given result.

peirismsh@dcs.ruh.ac.lk

TWO'S COMPLEMENT EXAMPLE

- Find 2's complement of binary number 10101110.
- Simply invert each bit of given binary number, which will be 01010001. Then add 1 to the LSB of this result,
- i.e., 01010001+1=01010010 which is answer.

TWO'S COMPLEMENT IN SINGED NUMBERS

- To get the two's complement negative notation of an integer, we write out the number in binary, then invert the digits, and add one to the result.
- Suppose we're working with 8 bit quantities (for simplicity's sake) and suppose we want to find how -28 would be expressed in two's complement notation. First we write out 28 in binary form.

TWO'S COMPLEMENT EXAMPLE

- 28 → 00011100
- Then we invert the digits. 0 becomes 1, 1 becomes 0.
- 11100011
- Then we add 1.
- Answer: 11100100

TWO'S COMPLEMENT

- +3 = 00000011
- +2 = 00000010
- +1 = 00000001
- +0 = 00000000
- -1 = ?
- -2 = ?
- -3 = ?

2'S COMPLEMENT BENEFITS

- One representation of zero
- Arithmetic works easily (see later)
- Negating is fairly easy

  -3

  3 = 00000011

  Boolean complement gives 11111100

  Add 1 to LSB 11111101

NEGATION SPECIAL CASE

- 0 = 00000000
- Bitwise not 11111111
- Add 1 to LSB +1
- Result 1 00000000
- Overflow is ignored, so:

- 0 = 0 √

DECIMAL TO TWO'S COMPLEMENT (METHOD 2)

−128	64	32	16	8	4	2	1

(a) An eight-position twos complement value box

−128	64	32	16	8	4	2	1
1	0	0	0	0	0	1	1

−128 +2 +1 = −125

(b) Convert binary 10000011 to decimal

−128	64	32	16	8	4	2	1
1	0	0	0	1	0	0	0

−120 = −128 +8

(c) Convert decimal −120 to binary

RANGE OF NUMBERS

- 8 bit 2s compliment
  - +127 = 01111111 = 2^7 -1
  - -128 = 10000000 = -2^7

- 16 bit 2s compliment
  - +32767 = 011111111 11111111 = 2^{15} - 1
  - -32768 = 100000000 00000000 = -2^{15}

CONVERSION BETWEEN LENGTHS

Sign magnitude

- +18 (8 bits) = 00010010
- +18 (16 bits) = 00000000 00010010

- -18 (8 bits) = 10010010
- -18 (16 bits)= 10000000 00010010

CONVERSION BETWEEN LENGTHS

Two's Complement

- +18 (8 bits) = 00010010
- +18 (16 bits) = 00000000 00010010

- -18 (8 bits) = 11101110
- -18 (16 bits)= 11111111 11101110
-

ADDITION AND SUBTRACTION

- **Addition**: Normal binary addition, monitor sign bit for overflow

- **Subtraction**

  A← Minuend

  -B← Subtrahend

  - Take twos compliment of subtrahend and add to minuend
  - A - B = A + 2's(-B)

- So we only need addition and complement circuits

OVERFLOW AND CARRY BIT

- Carry bit: The bit beyond the end of the word
- Overflow :
  - In addition, the result may be larger than the word size
  - ALU must signal this fact and to discard the result.

Overflow Rule

If two numbers are added, and they are both positive or both negative, then overflow occurs if and only if the result has the opposite sign.

OVERFLOW RULE

1. (-7)+5
2. (-4)+4
3. 3+4
4. (-4)+(-1)
5. 5+4
6. (-7)+(-6)

ADDITION

1001 = −7 +0101 = 5 1110 = −2 (a) (−7) + (+5)	1100 = −4 +0100 = 4 10000 = 0 (b) (−4) + (+4)
0011 = 3 +0100 = 4 0111 = 7 (c) (+3) + (+4)	1100 = −4 +1111 = −1 11011 = −5 (d) (−4) + (−1)
0101 = 5 +0100 = 4 1001 = Overflow (e) (+5) + (+4)	1001 = −7 +1010 = −6 10011 = Overflow (f) (−7) + (−6)

•

SUBTRACTION

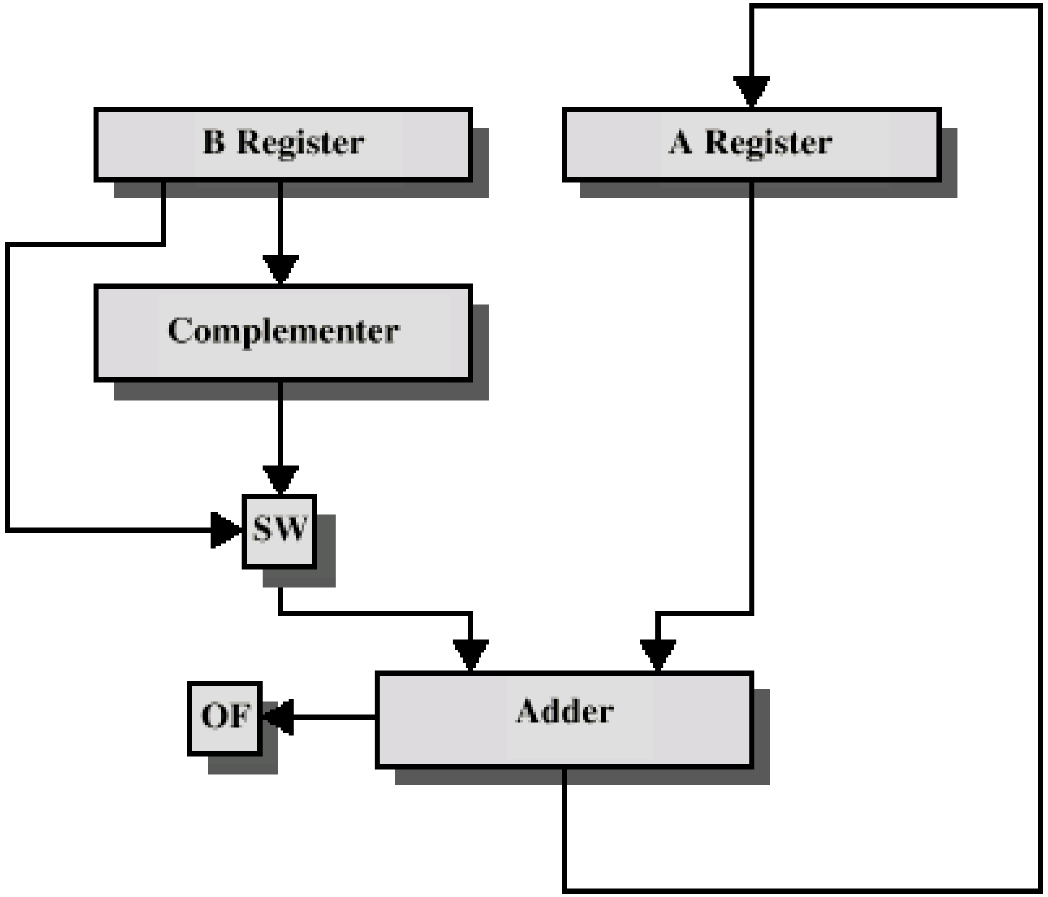
To subtract one number (subtrahend) from another (minuend), take the twos complement (negation) of the subtrahend and add it to the minuend.

1. 2-7
2. 5-2
3. -5-2
4. 5-(-2)
5. 7-(-7)
6. -6-4

SUBTRACTION

0010 = 2 +1001 = –7 1011 = –5 (a) M = 2 = 0010 S = 7 = 0111 –S = 1001	0101 = 5 +1110 = –2 10011 = 3 (b) M = 5 = 0101 S = 2 = 0010 –S = 1110
1011 = –5 +1110 = –2 11001 = –7 (c) M = –5 = 1011 S = 2 = 0010 –S = 1110	0101 = 5 +0010 = 2 0111 = 7 (d) M = 5 = 0101 S = –2 = 1110 –S = 0010
0111 = 7 +0111 = 7 1110 = Overflow (e) M = 7 = 0111 S = –7 = 1001 –S = 0111	1010 = –6 +1100 = –4 10110 = Overflow (f) M = –6 = 1010 S = 4 = 0100 –S = 1100

H/W FOR ADDITION AND SUBTRACTION



OF = overflow bit
SW = Switch (select addition or subtraction)

- Questions?