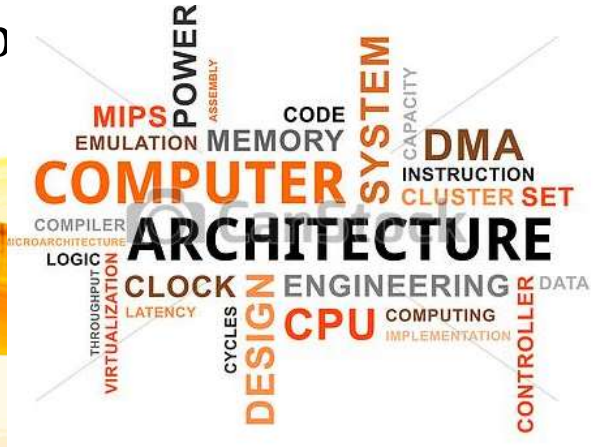
UNIT II
ARITHMETIC OPERATIONS

Addition and subtraction of signed numbers – Design of fast adders – Multiplication of positive numbers - Signed operand multiplication- fast multiplication – Integer division – Floating point numb



Recap the previous Class

Computer Arithmetic's

- **Arithmetic Instruction** Manipulate data to produce results necessary for the solution of computational problem

- Four Basic Arithmetic Operations are – **Addition, Subtraction, Multiplication and Division**

- An **arithmetic processor** is the part of a processor unit that executes arithmetic operations

- Arithmetic operations can be performed for following data types

  - Fixed point binary data in signed magnitude representation
  - Fixed point binary data in signed 2's complement representation
  - Floating Point binary data
  - Binary coded decimal data

Addition and Subtraction

Representation of both *positive* and *negative* numbers

- Following 3 representations

Signed magnitude representation
Signed 1's complement representation
Signed 2's complement representation

Example: Represent +9 and -9 in 7 bit-binary number

Only one way to represent + 9 ==> 0 001001

Three different ways to represent - 9:

In signed-magnitude: 1 001001
In signed-1's complement: 1 110110
In signed-2's complement: 1 110111

Conversion

4382	
2191	0
1095	1
547	1
273	1
136	1
68	0
34	0
17	0
8	1
4	0
2	0
1	0
0	1

4382_{ten} =
$1\ 0001\ 0001\ 1110_{two}$

Hexadecimal: base 16. Octal: base 8

$1010\ 1011\ 0011\ 1111_{two} = ab3f_{hex}$

DECIMAL	HEX	BINARY
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Sign Magnitude Number

- A sign-magnitude number Z can be represented as (A_s, A) where A_s is the sign of Z and A is the magnitude of Z.
- The leftmost position, A_s, is the sign bit.
- The sign bit is either positive = 0 or negative = 1

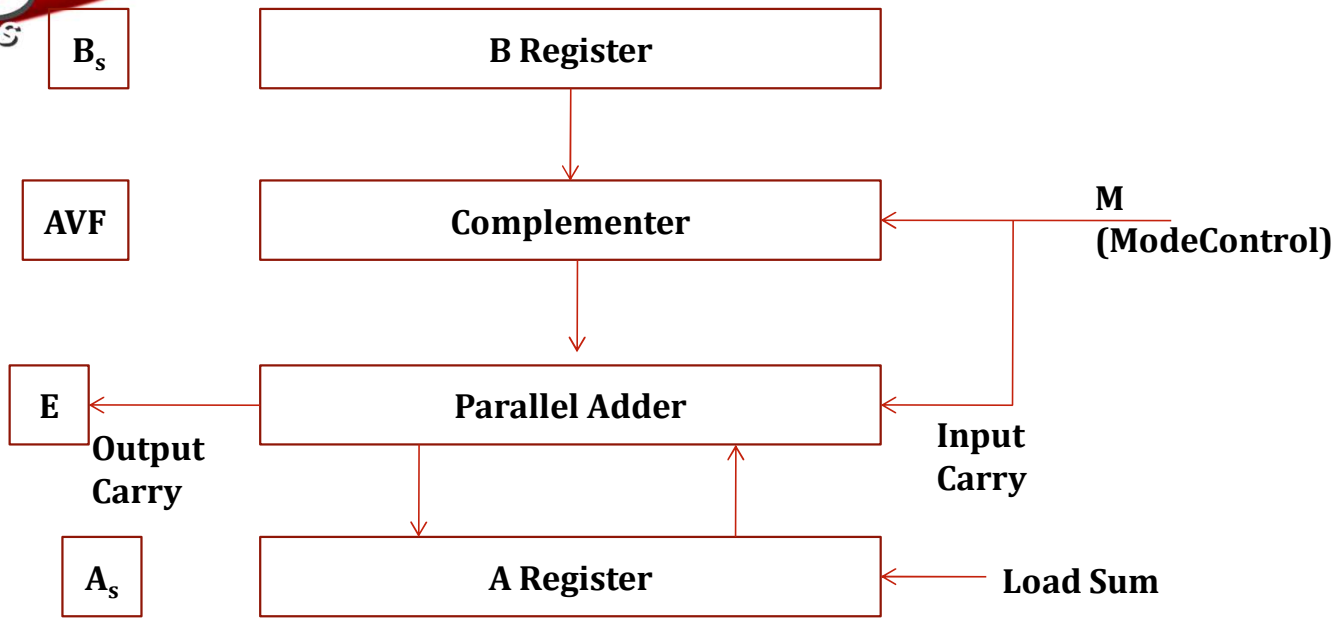
Number	Signed-Magnitude
+3	0 11
+2	0 10
+1	0 01
+0	0 00
-0	1 00
-1	1 01
-2	1 10
-3	1 11

Computer Arithmetic's

Addition (subtraction) Algorithm

- When the sign of A and B are identical (different) , add the magnitudes and attach the sign of A to the result.
- When the signs of A and B are different (identical), compare the magnitudes and subtract the smaller number from the larger.
  - Choose the sign of result to be same as A if A>B
  - or the complement of sign of A if A<B
  - if A=B subtract B from A and make the sign of result positive

Hardware Implementation



Simple procedure require magnitude comparator, an adder, two subtractor however alternative reveals that using 2's complement for operation requires only an adder and a complementor

M=0 output = **A+B** **M=1** output = A+B'+1= **A-B**

Addition and Subtraction

Operation	Add Magnitudes	Subtract Magnitudes		
		A>B	A<B	A=B
(+ A) + (+ B)	+ (A + B)			
(+ A) + (- B)		+ (A - B)	- (B - A)	+ (A - B)
(- A) + (+ B)		- (A - B)	+ (B - A)	+ (A - B)
(- A) + (- B)	- (A + B)			
(+ A) - (+ B)		+ (A - B)	- (B - A)	+ (A - B)
(+ A) - (- B)	+ (A + B)			
(- A) - (+ B)	- (A + B)			
(- A) - (- B)		- (A - B)	+ (B - A)	+ (A - B)

EXAMPLE

- Example of adding two magnitudes when the result is the sign of both operands:

```
   +3   0 011
+  +2   0 010
-------------
   +5   0 101
```

- Example of adding two magnitudes when the result is the sign of larger magnitude

```
   -3   1 011
+  +2   0 010
-------------
-( +3   0 011
    - 2) 1 010
-------------
   -(1) 1 001
```

Arithmetic Addition

For example, (+25) + (-37) = -(37 - 25) = -12 and is done by subtracting the smaller magnitude 25 from the larger magnitude 37 and using the sign of 37 for the sign of the result.

Eg 1:

+6	00000110	
+13	00001101	
+19	00010011	

Eg 2:

-6	11111010	(Signed 2's complement of -6)
+13	00001101	
+7	1]00000111	

Eg3:

+6	00000110	
-13	11110011	(Signed 2's complement of -13)
-7	11111001	(Signed 2's complement of -7)

Eg4:

-6	11111010	(Signed 2's complement of -6)

Arithmetic subtraction

Eg 1:

-6	11111010	(Signed 2's complement of -6)
-13	11110011	(Signed 2's complement of -13)
-6	11111010	
-13	00001101	(2's complement of -13)
+7	1] 00000111	

Eg 2:

+13	00001101	
+6	00000110	
+13	00001101	
+6	11111010	(2's complement of +6)
+7	1] 00000111	

Eg 3:

-6	11111010	(Signed 2's complement of -6)
+13	00001101	
-6	11111010	
+13	11110011	(2's complement of +13)
-19	1] 11101101	

- Addition and subtraction is the basic operation to be performed by computer

- Digits are added bit by bit from right to left, with carries passed to the next digit to the left.

Example

Adding 6_{10} to 7_{10} in binary

Solution

6	0110
7	0111
13	1101

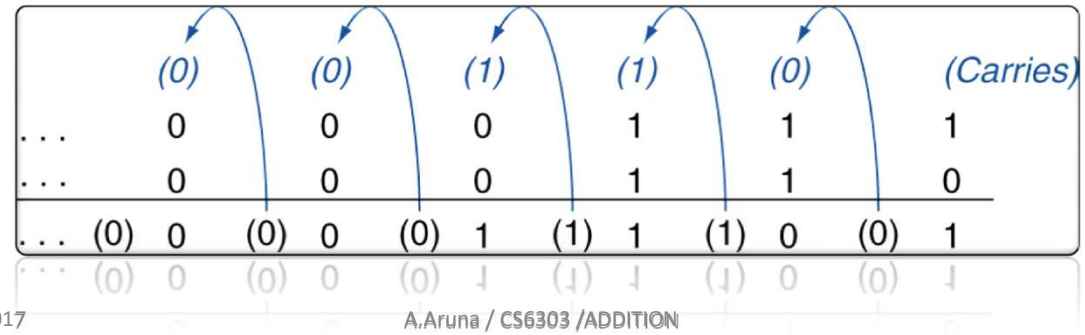
COMPUTER ADDITION

- Can be taken place in 32 bit formats

$0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0111_2 = 7_{10}$

$0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0110_2 = 6_{10}$

$0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 1101_2 = 13_{10}$



Overflow conditions for addition and subtraction

- overflow occurs when adding two positive numbers and the sum is negative, or vice versa. This means a carry out occurred into the sign bit.
- Overflow occurs in subtraction when we subtract a negative number from a positive number and get a negative result, or when we subtract a positive number from a negative number and get a positive result. This means a borrow occurred , from the sign bit.


Thank You