Basic Arithmetic Computing

Team Dosen

Telkom University

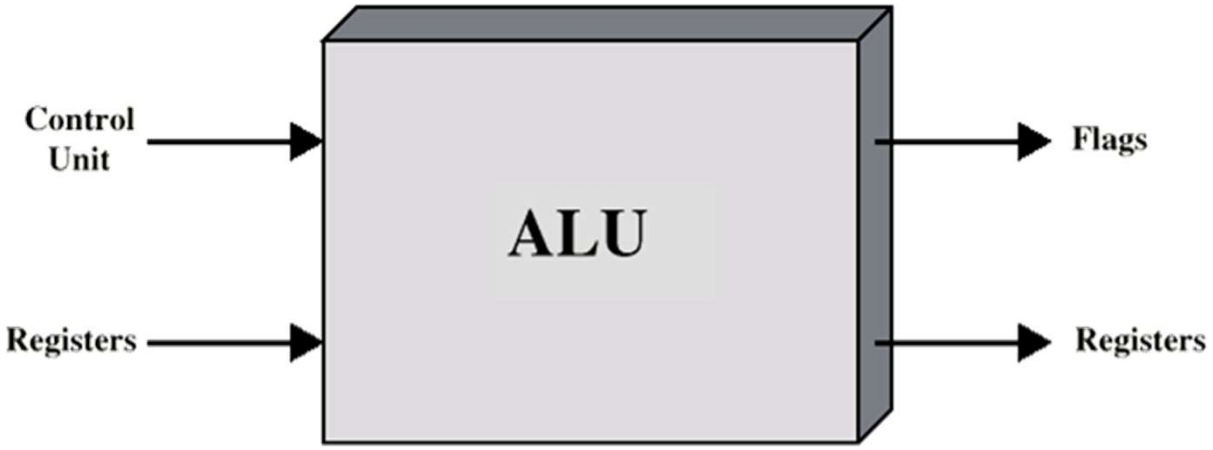
2016

UNIVERSITAS Telkom

Arithmetic & Logic Unit

- Pekerjaan : menghitung
- Menangani integer
- Bisa menangani bilangan floating point (real) dengan algortima tertentu atau menggunakan hardware ALU FP khusus
- Status dari hasil perhitungan disimpan di register Flag/Status

ALU Input dan Output



Representasi Integer

- Hanya menggunakan simbol 0 & 1 untuk menyatakan sesuatu
- Bilangan positif dinyatakan dalam biner
  - Mis: 41=00101001
- Tidak ada tanda minus
- Tidak ada koma
- Tanda + - dinyatakan dengan bit
- Bilangan negatif dinyatakan dgn 2’s complement

Tanda + -

- Bit paling kiri (MSB) menyatakan tanda
- Contoh format signed magnitude 8 bit
  - 0 adalah positif → +18 = 00010010
  - 1 adalah negatif → -18 = 10010010

- Masalah
  - Perhitungan memperhatikan bit–bit nilai dan bit tanda
  - 0 dinyatakan 2 kali (+0 → 00000000 dan -0 → 10000000)

2’s Complement

- +3 = 00000011
- +2 = 00000010
- +1 = 00000001
- +0 = 00000000
- -1 = 11111111
- -2 = 11111110
- -3 = 11111101

Bit-ke	7	6	5	4	3	2	1	0
Bobot	-128	64	32	16	8	4	2	1

Keuntungan

- Hanya satu bilangan 0
  - 00000000 → 0
  - 10000000 → -128
- Aritmetik menjadi mudah
- Pengubahan menjadi negatif:
  - 3 = 00000011
  - Komplementnya = 11111100
  - Tambah 1 ke LSB → 11111110

  Maka -3 = 11111101

UNIVERSITAS Telkom

Jangkauan Bilangan

- 8 bit 2s complement
  - +127 = 01111111 = 2^7 -1
  - -128 = 10000000 = -2^7
- 16 bit 2s complement
  - +32767 = 011111111 11111111 = 2^{15} - 1
  - -32768 = 100000000 00000000 = -2^{15}

- N bit 2s complement
  - Positif $2^{N-1} – 1$
  - Negatif -2^{N-1}

UNIVERSITAS Telkom

Konversi ke bilangan dengan jumlah bit lebih besar

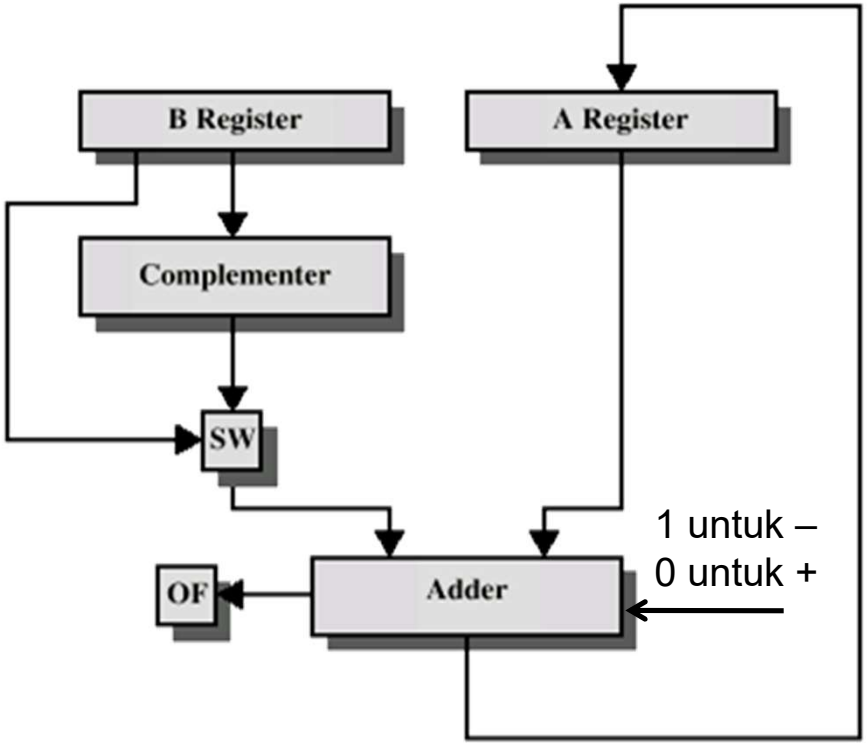
- Contoh konversi 8 bit ke 16 bit
  - Bilangan positif ditambah bilangan 0 di depan sampai penuh 16 bit
    - +18 = 00010010
    - +18 = 00000000 00010010
  - Bilangan negatif ditambah bilangan 1 di depan sampai penuh 16 bit
    - -18 = 11101110
    - -18 = 11111111 **11101110**

Penjumlahan dan Pengurangan

- Penjumlahan biner normal
- Perhatikan bit tanda untuk overflow
- a - b = a + (-b)

- Jadi hanya diperlukan rangkaian penjumlah dan komplemen

Hardware untuk Penjumlahan dan Pengurangan



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

Perkalian Biner Tanpa Tanda

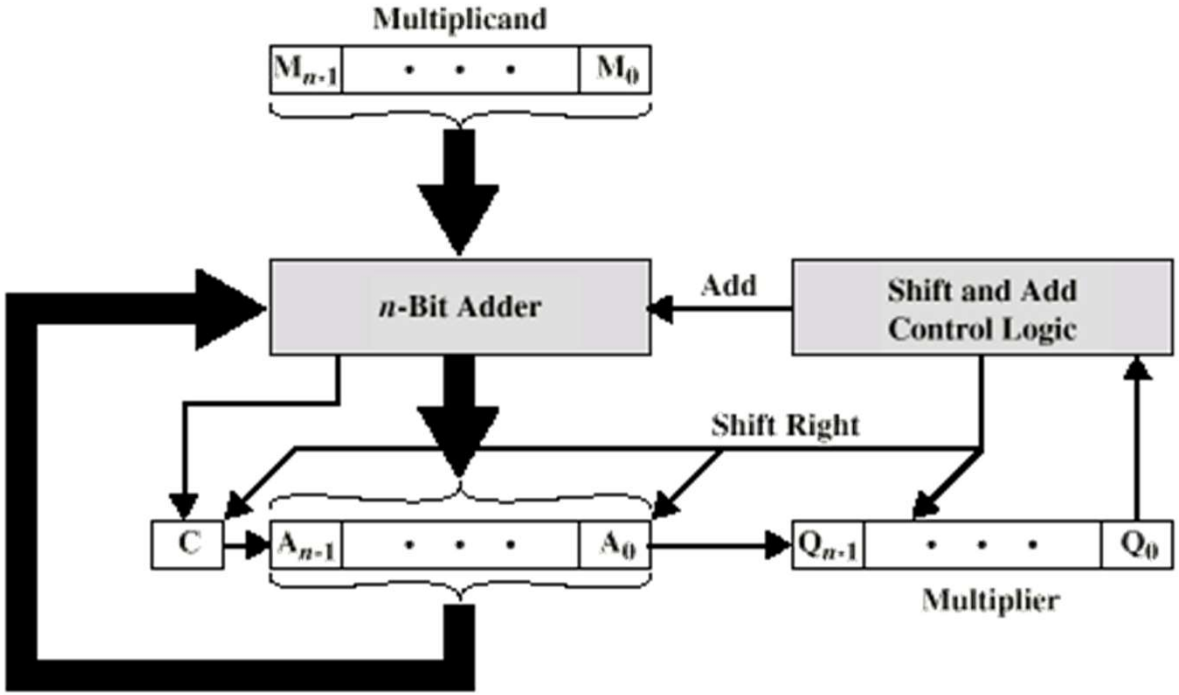
- Operasi repetitif dari operasi penjumlahan dan pergeseran

- 1011 Multiplicand (11)
- x 1101 Multiplier (13)
- 1011 Partial products
- 0000 Note: if multiplier bit is 1 copy
- 1011 multiplicand
- 1011 otherwise zero
- 10001111 Hasil (143)

Perlu tempat (register) dua kali lebih lebar atau 2 register untuk menampung hasil perkalian

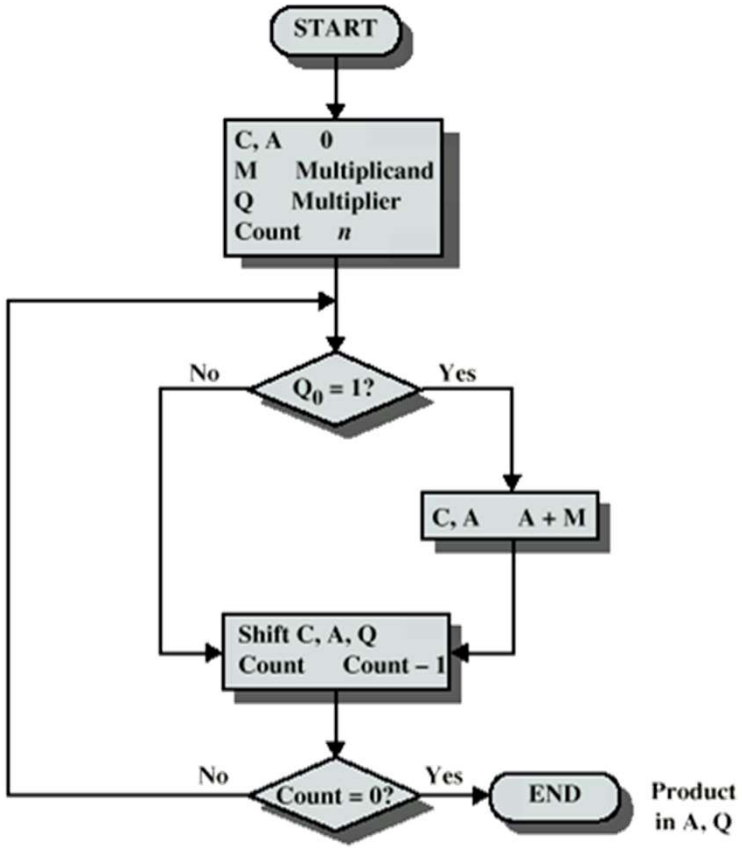
UNIVERSITAS Telkom

Diagram Perkalian Biner Tanpa Tanda



(a) Block Diagram

Flowchart untuk Perkalian Biner Tanpa Tanda



Contoh Eksekusi

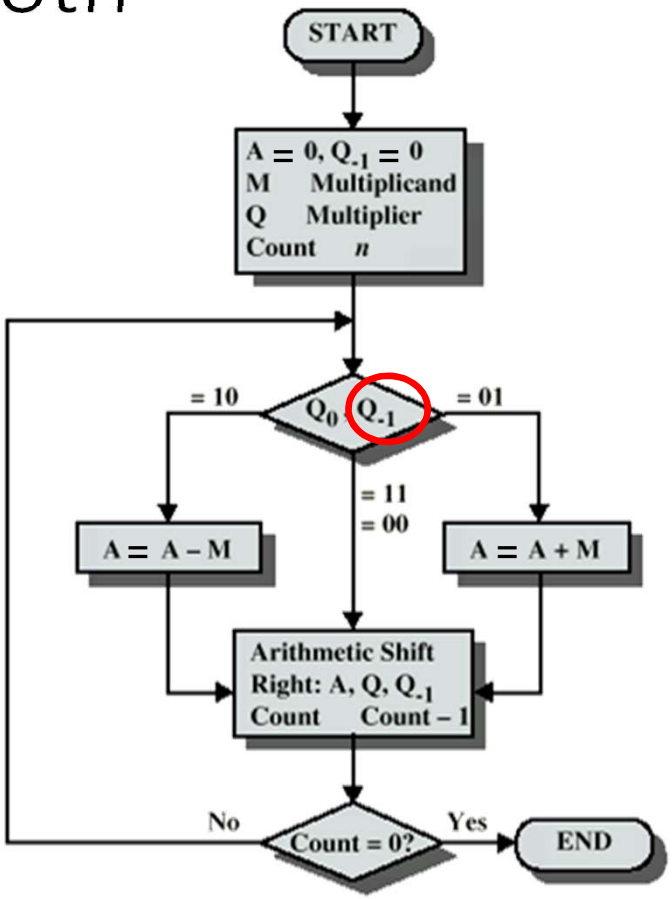
C	A	Q	M		
0	0000	1101	1011	Initial Values	
0	1011	1101	1011	Add	First Cycle
0	0101	1110	1011	Shift	
0	0010	1111	1011	Shift	Second Cycle
0	1101	1111	1011	Add	Third Cycle
0	0110	1111	1011	Shift	
1	0001	1111	1011	Add	Fourth Cycle
0	1000	1111	1011	Shift	

UNIVERSITAS Telkom

Perkalian Bilangan Negatif

- Tidak bisa dilakukan dengan cara tersebut!
- Solusi 1
  - Ubah ke positif jika memungkinkan
  - Kalikan seperti cara sebelumnya
  - Jika tanda berbeda → inverskan hasil
- Solusi 2
  - Algoritma Booth

Algoritma Booth



Contoh Algoritma Booth

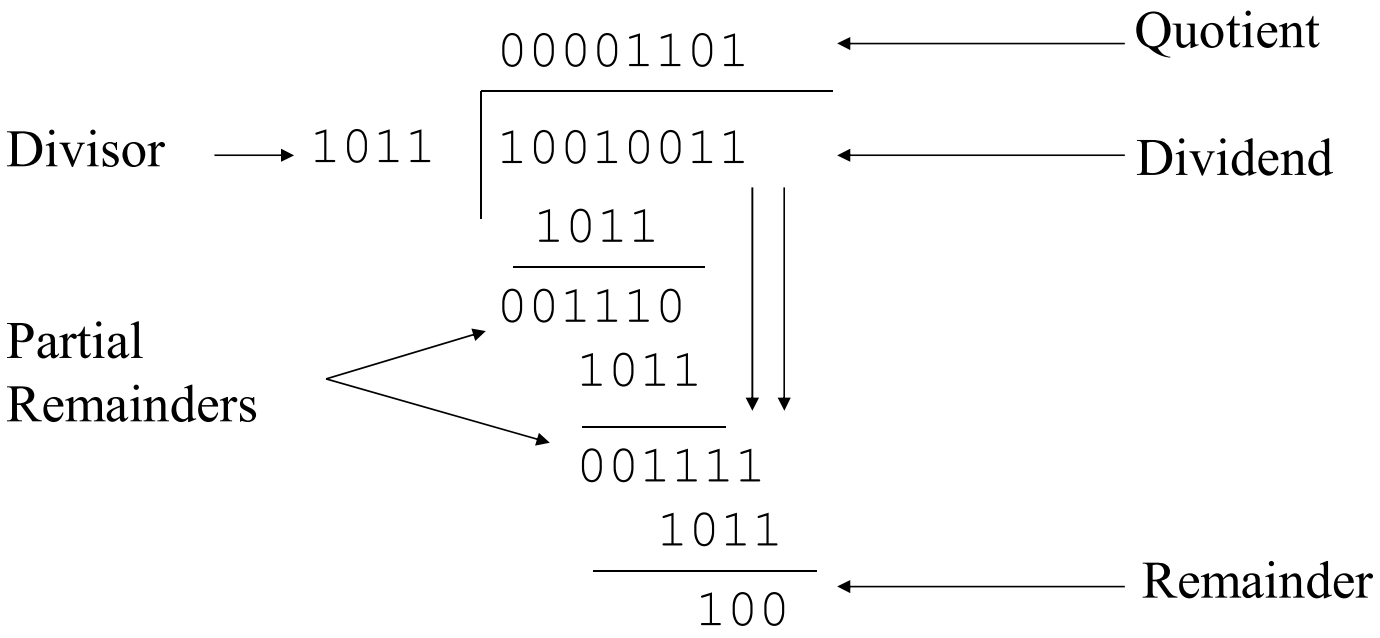
A	Q	Q_{-1}	M		
0000	0011	0	0111	Initial Values	
1001	0011	0	0111	A = A - M	First Cycle
1100	1001	1	0111	Shift	
1110	0100	1	0111	Shift	Second Cycle
0101	0100	1	0111	A = A + M	Third Cycle
0010	1010	0	0111	Shift	
0001	0101	0	0111	Shift	Fourth Cycle

UNIVERSITAS Telkom

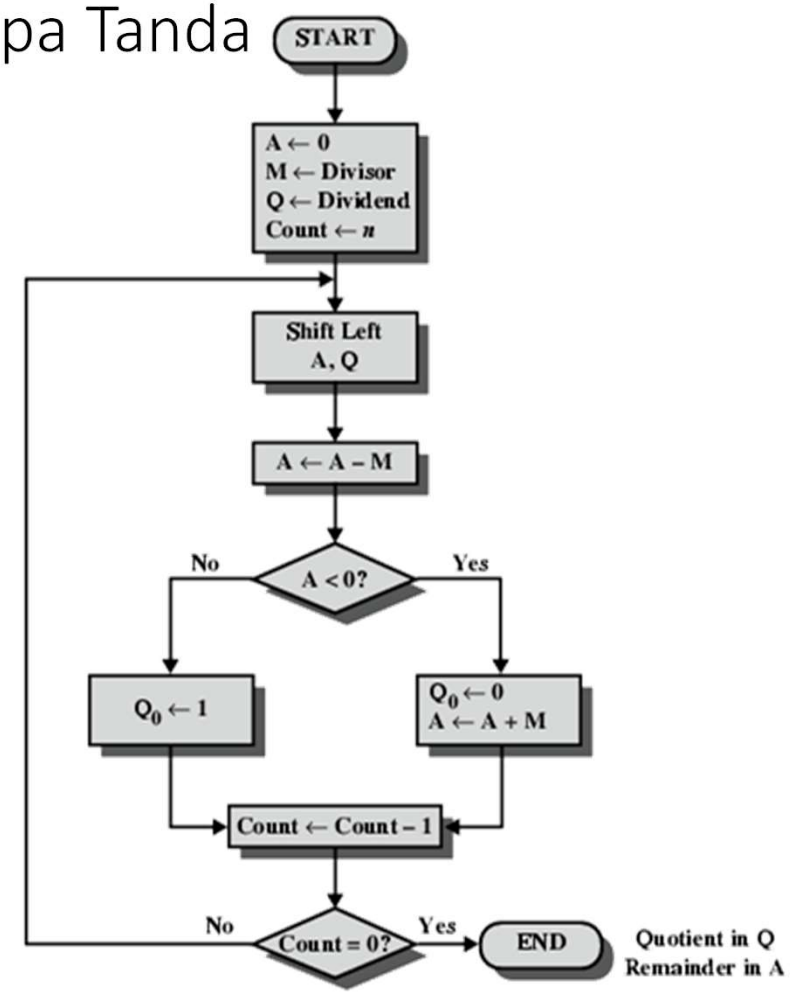
Pembagian (Division)

- Lebih rumit dari perkalian
- Berdasarkan long division

Pembagian Integer Biner Tanpa Tanda



Flowchart Pembagian Biner Tanpa Tanda



Catatan:
Walaupun flowchart ini untuk biner tanpa tanda,
A merupakan bilangan bertanda (sign-number)

UNIVERSITAS Telkom

Pembagian Biner Tanpa Tanda

A	Q	
0000	0111	Initial value
0000	1110	Shift
1101		Use twos complement of 0011 for subtraction
1101		Subtract
0000	1110	Restore, set $Q_0 = 0$
0001	1100	Shift
1101		
1110		Subtract
0001	1100	Restore, set $Q_0 = 0$
0011	1000	Shift
1101		
0000	1001	Subtract, set $Q_0 = 1$
0001	0010	Shift
1101		
1110		Subtract
0001	0010	Restore, set $Q_0 = 0$

1. M diisi dengan 2's complement dari divisor
2. Lakukan pergeseran logic 1 bit ke kiri
3. A = A – M
4. Pengecekan MSB A
   a) Jika hasil MSB dari A = 0 (positif), $Q_0 = 1$
   b) Jika hasil MSB dari A = 1 (negative), set $Q_0 = 0$, kembalikan nilai A ke nilai sebelumnya.
5. Ulangi langkah 2-4 sebanyak (jumlah bit Q)
6. Sisa bagi disimpan di A, Hasil Bagi di Q

Rumus-rumus lain

$$\exp(x) \approx \lim_{n \to \infty} \; 1 + x + x^2/2! + x^3/3! + \ldots + x^n/n!$$

Hitung $\sin(x)$ dengan pendekatan:

$$\lim_{n \to \infty} \; S_n;\; S_n = x - x^3/3! + x^5/5! - \ldots + (-1)^{n-1} x^{2n-1}/(2n-1)!\;,\; n \_ 1$$

$$\sin(x) \approx x - x^3/3! + x^5/5! - \ldots$$

suku umum dapat ditulis sebagai : $t_1 = (-1)^{i-1} x^{2i-1} / (2i-1)$

Hitung $\cos(x)$ dengan pendekatan:

$$S = 1 - x^2/2! + x^4/4! - \ldots + (-1)^n x^{2n} /(2n)!\;,\; n \_ 0$$

dan kondisi berhenti: $|T| \le \varepsilon * |S|$

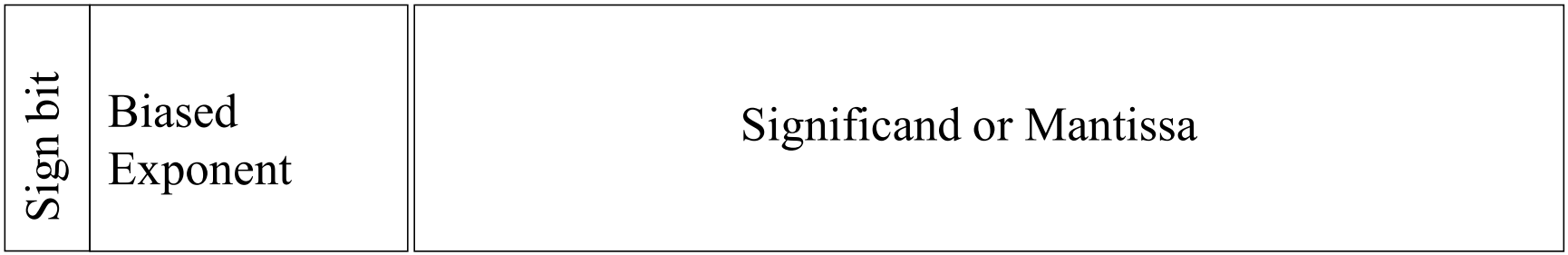
UNIVERSITAS Telkom

Floating-Point

- Bilangan dengan pecahan kadang kadang sulit dinyatakan dengan bilangan biner contoh : 0.3 → 0.01001100110011001100….
- Bilangan yang sangat besar dan bilangan yang sangat kecil, misal menyatakan 0.00000000000000000000000000000025 yang dalam bilangan desimal ditulis sebagai $2.5x10^{-32}$
- Titik(koma) di bilangan biner? Komputer hanya mengenal simbol ‘0’ dan ‘1’
- Register mempunyai jumlah bit tertentu (8 bit, 16 bit dsb)
  - Jangkauan bilangan terbatas, tidak mungkin bisa menampung bilngan yang sangat besar atau sangat kecil
- Solusi : Bilangan Floating Point

UNIVERSITAS Telkom

Format Floating-Point 32-bit



- +/- .significand x $2^{exponent}$
- Titik terletak tetap antara bit tanda (sign bit) dan mantissa
- Exponent menyatakan tempat nilai (exceed-127)

Contoh Floating Point

(a) Format

$1.1010001 \times 2^{10100} = 0\ 10010011\ 10100010000000000000000 = 1.638125 \times 2^{20}$

$-1.1010001 \times 2^{10100} = 1\ 10010011\ 10100010000000000000000 = -1.638125 \times 2^{20}$

$1.1010001 \times 2^{-10100} = 0\ 01101011\ 10100010000000000000000 = 1.638125 \times 2^{-20}$

$-1.1010001 \times 2^{-10100} = 1\ 01101011\ 10100010000000000000000 = -1.638125 \times 2^{-20}$

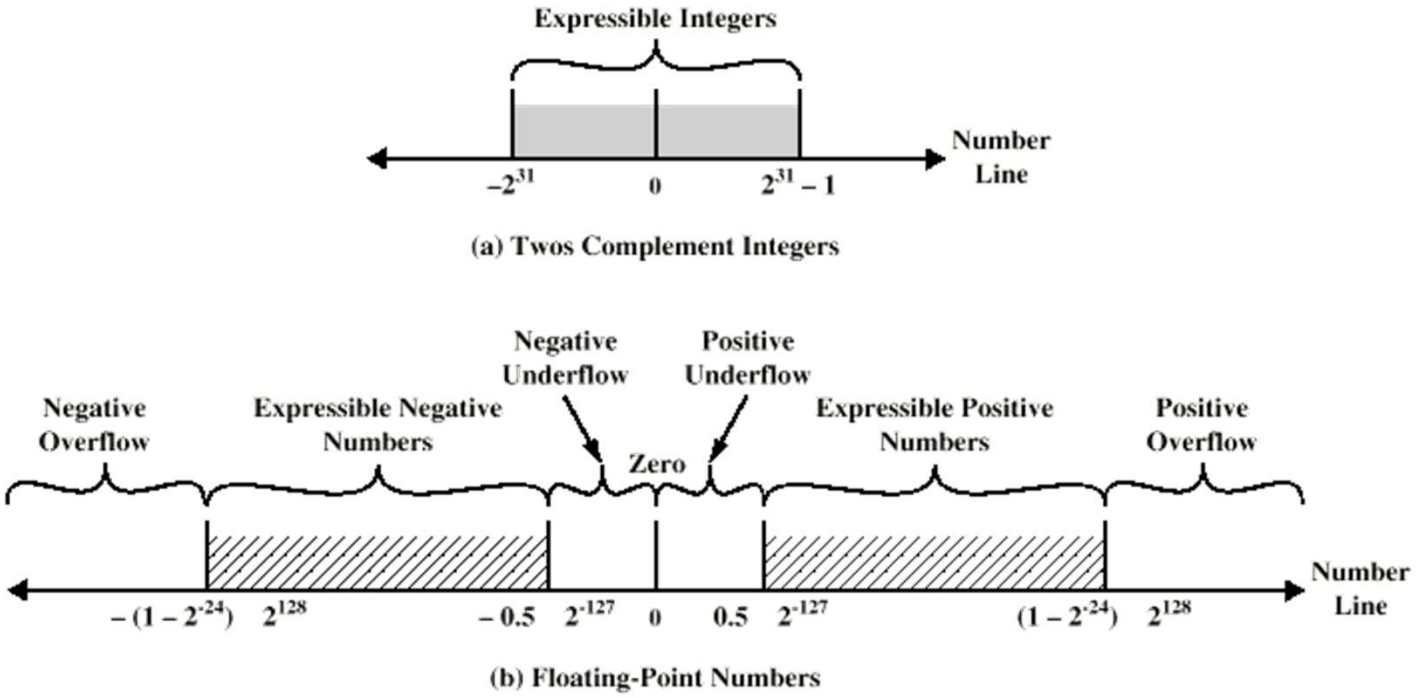
UNIVERSITAS Telkom

Jangkauan FP

- Untuk bilangan 32 bit
  - Eksponen 8 bit
  - Bilangan positif $2^{-127} \sim (2-2^{-23})*2^{128}$
  - Bilangan negatif $-(2-2^{-23})*2^{128} \sim -2^{-127}$
- Ketelitian
  - Mantisa 23 bit $2^{-23} \approx 1.2 \times 10^{-7}$
  - Sekitar 6 digit desimal

UNIVERSITAS Telkom

Bilangan 32 Bit Bisa Dinyatakan



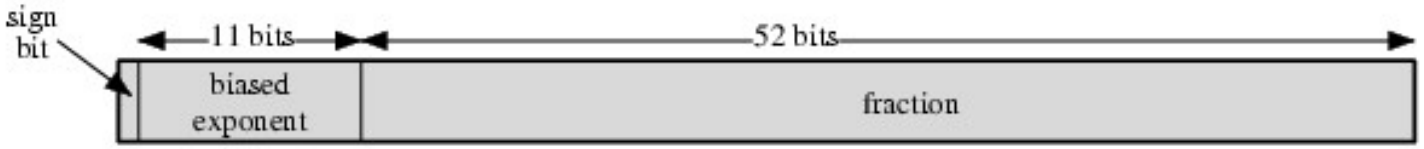
IEEE 754

- Standar untuk penyimpanan FP
- Standar 32 dan 64 bit
- Eksponen 8 (exceed 127) dan 11 bit (exceed -1023)
- Basis bilangan 2

UNIVERSITAS Telkom

Format IEEE 754

(a) Single format



(b) Double format

Format IEEE 754

Parameter	Format			
	Single	Single Extended	Double	Double Extended
Word width (bits)	32	≥43	64	≥79
Exponent width (bits)	8	≥11	11	≥15
Exponent bias	127	unspecified	1023	unspecified
Maximum exponent	127	≥1023	1023	≥16383
Minimum exponent	−126	≤−1022	−1022	≤−16382
Number range (base 10)	$10^{-38}, 10^{+38}$	unspecified	$10^{-308}, 10^{+308}$	unspecified
Significand width (bits)*	23	≥31	52	≥63
Number of exponents	254	unspecified	2046	unspecified
Number of fractions	2^{23}	unspecified	2^{52}	unspecified
Number of values	1.98×2^{31}	unspecified	1.99×2^{63}	unspecified

UNIVERSITAS Telkom