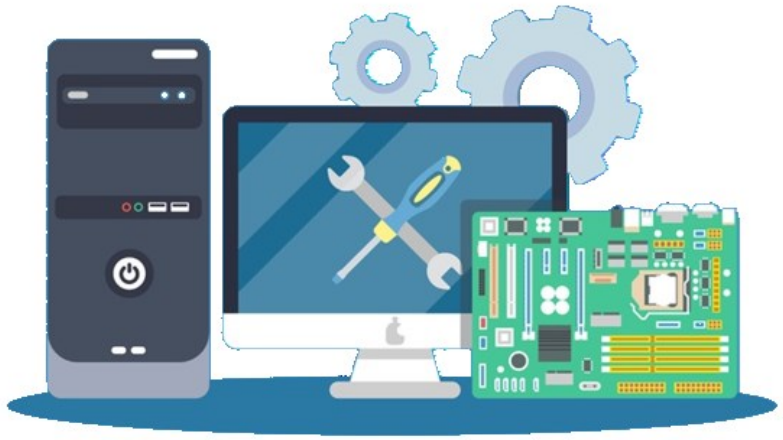
PROGRAM STUDI
TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

MATA KULIAH
ORGANISASI DAN
ARSITEKTUR KOMPUTER



Representasi Floating Point

TIM Organisasi dan Arsitektur Komputer

2020

Materi yang harus dikuasai

- Number System

Capaian Pembelajaran

- Mahasiswa mampu menjelaskan representasi floating point

Latar Belakang

- Bilangan dengan bit terlalu besar atau terlalu kecil (fractional/pecahan) memiliki keterbatasan dalam representasi. Sebagai contoh:
  - `80000000000`
  - `0.0000000008`
- User dapat melakukan kesalahpahaman penulisan bilangan tersebut.
- Representasi bilangan yang terlalu besar atau kecil, dapat menyebabkan kesalahan operasi. Ada beberapa bit yang tidak tertulis

Latar Belakang

- Untuk memudahkan representasi bilangan dengan jumlah digit terlalu besar atau kecil dapat menggunakan representasi floating-point
- Contoh:
  - 80000000000 dapat direpresentasikan 8.0×10^{10}
  - 0.0000000008 dapat direpresentasikan 8.0×10^{-10}
- Representasi bilangan dengan range terlalu besar atau terlalu kecil dapat direpresentasikan dengan beberapa digit saja

Representasi Floating Point

- Format representasi floating point pada bilangan biner sama seperti representasi pada bilangan base-10 (desimal)
- Format representasi yang digunakan adalah:

$$\pm S \times B^{\pm E}$$

- Bilangan tersebut dapat disimpan dalam *binary word* dengan tiga komponen:
  - Sign : plus atau minus
  - Significand **S**
  - Exponent **E**

Normalisasi Floating Point

- Untuk memudahkan operasi bilangan (integer) floating-point dibutuhkan normalisasi bilangan.
- Berikut aturan normalisasi floating-point:
  - Bilangan most significant bit (MSB) pada **S**ignificand harus 1 (satu), maka bentuknya adalah: $\pm 1.bbb...b \times 2^{\pm E}$
  - dimana b adalah digit biner 0 atau 1.
- Contoh:
  - $1.1010001 \times 2^{10100}$
  - $-1.1010001 \times 2^{10100}$

Floating-Point Single Precision(32-bit)

Sign Bit	Biased Exponent (E)	Mantissa (m)
1 bit	8 bits	23 bits

- **Sign:**
  - **0 untuk positive**
  - **1 untuk negative**

- **Biased exponent** merupakan **nilai tetap** dengan cara nilai dikurangkan dari eksponen untuk mendapatkan nilai pangkat sebenarnya.
  Nilai bias adalah 127 (2^7 - 1), dengan range adalah **-127** sampai **+128**
  Untuk mendapatkan biased exponent pada single precision nilai ditambahkan dengan **127**

- **mantissa,** merupakan representasi significand yaitu bagian pecahan (fractional), nilai setelah koma.

Contoh – FP Single Precision

- $1.1010001 \times 2^{10100}$

0	10010011	1010001 0000 0000 0000 0000

Sign:
Bilangan adalah positive
Jadi, Sign adalah 0

Biased Exponent:
Pangkat ditambahkan dengan 127.
$127_{10} : 01111111_2$

```
  00010100
+ 01111111
  10010011 → E
```

Mantissa:
Bilangan setelah koma (radix point):
1010001…0 → m
Note: jika bilangan kurang dari 23 digit, maka dapat ditambahkan 0 hingga jumlahnya 23 digit

latihan

- Konversi bilangan dibawah ini ke dalam format floating point single precision:
  - $-1.1010001 \times 2^{10100}$
  - $1.1010001 \times 2^{-10100}$

Representasi Floating Point IEEE standard 754

- Representasi floating point distandarkan oleh IEEE standard 754 penyamaan orientasi program numerik pada computer
- Program dapat diimplementasikan dari satu processor ke processor lain *(easy portability)*
- Standar ini telah digunakan secara luas di semua processor dan aritmatika co-processor

Representasi Floating Point IEEE standard 754

- Format normalisasi bilangan menggunakan representasi :

$$(-1)^s * (1.m) * 2^{e-127}$$

- Dimana:
  - **s** adalah tanda bilangan (sign bit)
    0 untuk positif,
    1 untuk negative
  - **m** representasi dari mantissa
  - **e** adalah pangkat (exponent)

Contoh 1

Tentukan bilangan decimal yang ekuivalen dengan bilangan floating point **40200000H?**

Contoh 1 – Penyelesaian

1. Ubah bilangan tersebut dalam biner

   0100 0000 0010 0000 0000 0000 0000 0000

2. Pada single precision pembagian bit :
   sign 1 bit, **e**xponent 11 bit, dan **m**antissa 23 bit

0.	0	1	0	…	0
2^0	2^{-1}	2^{-2}	2^{-3}	…	2^{-n}

0 100 0000 0 010 0000 0000 0000 0000 0000

sign exponent mantissa

```
s : 0
e : 1000 0000 : 128
m : 010 … 0 : 0.25
```

Hasil Akhir:

$= (-1)^s * (1.m) * 2^{e-127}$
$= (-1)^0 * (1.25) * 2^{128-127}$
$= 1.25 * 2^1$
$= 2.5_{10}$

Contoh 2

Tentukan normalisasi floating point 32-bit dari bilangan decimal berikut

- 5_{10}

Contoh 2 – Penyelesaian …(1)

Tahapan:

1. **Normalisasi bilangan floating point menggunakan format**

$$(-1)^s * (1.m) * 2^{e-127}$$

2. **Identifikasi sign bit, exponent dan mantissa**

cara menentukan exponent dan mantissa

1. Cari hasil pangkat terdekat dengan decimal. Dalam hal ini, hasil pangkat terdekat dengan 5 adalah 4 (2^2)
2. Jika hasil pangkat diketahui, maka mantissa dapat dicari, dengan cara:
   1.m = 5 / 4
   1.m = 1.25
   jadi, mantissa adalah 0.25

$$5_{10} = (-1)^s * (1.m) * 2^{e-127}$$
$$5_{10} = (-1)^0 * (1.25) * 2^2$$
$$5_{10} = 1.25 * (1.25) * 2^{129 - 127}$$

s = 0
e = 129 (10000001_2)
m = 0.25 ($0.010\ldots 0_2$)

Contoh 2 – Penyelesaian …(2)

Tahapan:

1. Normalisasi bilangan floating point menggunakan format

$$(-1)^s * (1.m) * 2^{e-127}$$

2. Identifikasi **s**ign bit, **e**xponent dan **m**antissa

3. **Konversi sign bit, exponent dan mantissa ke biner dengan jumlah digit:**

Sign Bit (s)	Biased Exponent (e)	Mantissa (m)
1 bit	8 bits	23 bits

s = 0
e = 129 (10000001_2)
m = 0.25 ($0.010\ldots 0_2$)

0 10000001 010 0000 0000 0000 0000 0000

Contoh 2 – Penyelesaian …(3)

Tahapan:

1. Normalisasi bilangan floating point menggunakan format

$$(-1)^s * (1.m) * 2^{e-127}$$

2. Identifikasi **s**ign bit, **e**xponent dan **m**antissa

3. Konversi sign bit, exponent dan mantissa ke biner dengan jumlah digit:

Sign Bit (s)	Biased Exponent (e)	Mantissa (m)
1 bit	8 bits	23 bits

4. **Bagi bit tersebut dalam 4 kelompok bit**

Hasil dalam bentuk biner

0100	0000	1010	0000	0000	0000	0000	0000

Hasil dalam bentuk hexadecimal

4	0	A	0	0	0	0	0

latihan 1

Tentukan bilangan decimal yang ekuivalen dengan bilangan floating point:

- **40400000H?**
- **40B00000H?**

Latihan 2

Tentukan normalisasi floating point 32-bit dari bilangan decimal berikut:

- 8_{10}
- 10_{10}

Referensi

- William Stallings – Computer Organization and Architecture Designing For Performance 9th Edition (2013)
- Mustafa Abd-el Bhar, Hesham El Rewini – Fundamentals of Computer Organization and Architecture 9th edition (2005)

TERIMA KASIH

ANY QUESTIONS?