

Adding Two Numbers (same sign)

Normalised form of a 32 bit (normal) floating point number.

$$A = (-1)^S \times P \times 2^{E-bias}, \quad (1 \leq P < 2, E \in \mathbf{Z}, 1 \leq E \leq 254) \quad (7.22)$$

Normalised form of a 32 bit (denormal) floating point number.

$$A = (-1)^S \times P \times 2^{-126}, \quad (0 \leq P < 1) \quad (7.23)$$

Symbol	Meaning
S	Sign bit (0(+ve), 1(-ve))
P	Significand (form: 1.xxx(normal) or 0.xxx(denormal))
M	Mantissa (fractional part of significand)
E	(exponent + 127(bias))
\mathbf{Z}	Set of integers

- Recap : Floating Point Number System

Addition

- Add : $A + B$
 - **Unpack** the E fields $\rightarrow E_A, E_B$
 - Let the E field of the **result** be $\rightarrow E_C$
- **Unpack** the **significand** (P)
 - P contains \rightarrow 1 bit before the decimal point, 23 **mantissa** bits (24 bits)
 - **Unpack** to a 25 bit number (unsigned)
 - W \rightarrow Add a leading 0 bit, 24 bits of the **signficand**

Addition - II

- With no loss of generality
 - Assume $E_A \geq E_B$
- Let significands of A and B be P_A and P_B
- Let us initially set $W \leftarrow \text{unpack}(P_B)$
- We make their exponents equal and shift W to the right by $(E_A - E_B)$ positions

- $$W = W \gg (E_A - E_B)$$
$$W = W + P_A$$

Renormalisation

- Let the **significand** represented by register, W , be P_W
 - There is a possibility that $P_W \geq 2$
 - In this case, we need to **renormalise**
 - $W \leftarrow W \gg 1$
 - $E_A \leftarrow E_A + 1$
- The final **result**
 - Sign bit (same as sign of A or B)
 - Significand (P_W), exponent field (E_A)

Example

Example: Add the numbers: $1.01_2 * 2^3 + 1.11_2 * 2^1$

Answer:

The decimal point in W is shown for enhancing readability. For simplicity, biased notation not used.

1. $A = 1.01 * 2^3$ and $B = 1.11 * 2^1$
2. $W = 01.11$ (*significand* of B)
3. $E = 3$
4. $W = 01.11 \gg (3-1) = 00.0111$
5. $W + P_A = 00.0111 + 01.0100 = 01.1011$
6. **Result:** $C = 1.011 * 2^3$

Example - II

Example: Add : $1.01_2 * 2^3 + 1.11_2 * 2^2$

Answer:

The decimal point in W is shown for enhancing readability. For simplicity, biased notation not used.

1. $A = 1.01 * 2^3$ and $B = 1.11 * 2^2$
2. $W = 01.11$ (*significand* of B)
3. $E = 3$
4. $W = 01.11 \gg (3-2) = 00.111$
5. $W + P_A = 00.111 + 01.0100 = 10.001$
6. Normalisation: $W = 10.001 \gg 1 = 1.0001$, $E = 4$
7. **Result:** $C = 1.0001 * 2^4$

Rounding

- Assume that we were allowed only two **mantissa** bits in the previous example
 - We need to perform **rounding**
- Terminology :
 - Consider the sum(W) of the significands after we have **normalised** the result
 - $W \leftarrow (P + R) * 2^{-23} (R < 1)$

Rounding - II

- P represents the significand of the **temporary result**
- R (is a **residue**)
- Aim :
 - **Modify** P to take into account the value of R
 - Then, **discard** R
 - Process of rounding : $P \rightarrow P'$

IEEE 754 Rounding Modes

Rounding Mode	Condition for incrementing the significand	
	Sign of the result (+ve)	Sign of the result (-ve)
Truncation		
Round to $+\infty$	$R > 0$	
Round to $-\infty$		$R > 0$
Round to Nearest	$(R > 0.5) \vee (R = 0.5 \wedge lsb(P) = 1)$	$(R > 0.5) \vee (R = 0.5 \wedge lsb(P) = 1)$
\wedge (logical AND), R (residue)		

Implementing Rounding

- We need three bits
 - lsb(P)
 - msb of the **residue** (R) \rightarrow r (**round bit**)
 - OR of the rest of the bits of the **residue** (R) \rightarrow s (**sticky bit**)

Condition on Residue	Implementation
$R > 0$	$r \vee s = 1$
$R = 0.5$ —	$r \wedge s = 1$
$R > 0.5$	$r \wedge s = 1$
r (round bit), s (sticky bit)	

Renormalisation after Rounding

- In rounding : we might **increment** the significand
 - We might need to **renormalise**
 - After renormalisation
 - Possible that E becomes equal to 255
 - In this, case declare an **overflow**

Addition of Numbers (Opposite Signs)

- $C = A + B$

- Same assumption $E_A \geq E_B$

- **Steps**

- **Load** W with the significand of B (P_B)

- Take the 2's **complement** of W ($W = -B$)

- $W \leftarrow W \gg (E_A - E_B)$

- $W \leftarrow W + P_A$

- If ($W < 0$) **replace** it with its 2's complement. Flip the sign of the result.

Addition of Numbers (Opposite Signs)-II

- Normalise the result
 - Possible that $W < 1$
 - In this case, keep shifting W to the left till it is in normal form. (simultaneously decrement E_A)
- Round and Renormalise

Multiplication of FP Numbers

- Steps

- $E \leftarrow E_A + E_B - \text{bias}$

- $W \leftarrow P_A * P_B$

- Normalise (shift left or shift right)

- Round

- Renormalise

Simple Division Algorithm

- **Divide** A/B to produce C
 - There is no **notion** of a **remainder** in FP division
- **Algorithm**
 - $E \leftarrow E_A - E_B + \text{bias}$
 - $W \leftarrow P_A / P_B$
 - normalise, round, renormalise
- **Complexity** : $O(n \log(n))$