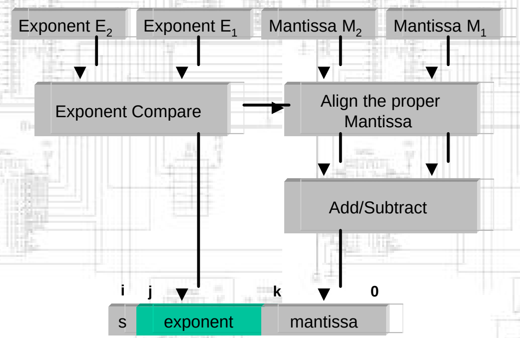
IEEE Single-precision special case

Valvue	S	Exponent	Mantissa	Example
Zero	X	0	0	+0, -0
Infinity	X	FF	0	+¥, -¥
Not-a-Number	X	FF	non-zero	0¸0, Ö-3
Max positive	0	FE	(1.)11…1	1.1…1´2^{127}=2^{128}
Min positive	0	01	(1.)00…1	1.0…1´2^{-126}=2^{-126}

Floating-point Adder



Exponent Compare & Align the proper Mantissa

X = 1.1100 ´2^4 = 28
+Y = 1.1000 ´2^2 = 6

	1.1100	´2^4
Align	+0.0110	´2^4
Add	10.0010	´2^4 = 34

Floating-point Addition
Example

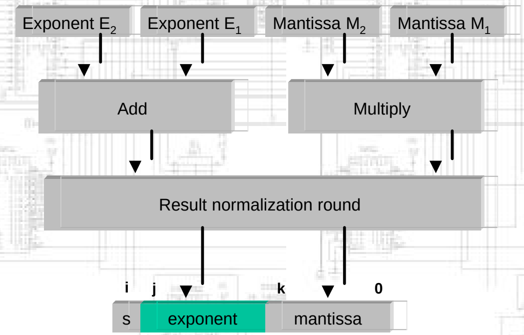
X = 1.1100 ´2^4 = 28
+Y = 1.1100 ´2^2 = 7

	1.1100	´2^4
Align	+0.0111	´2^4
Add	10.0111	´2^4
Normalise	1.00111	´2^5
Round	1.0010	´2^5
Normalize	1.0010	´2^5 = 35

Floating-point Multiplication

$X = m_X \cdot 2^{ex}$
$Y = m_Y \cdot 2^{ey}$
$Y \cdot X = (m_X \cdot m_Y) \cdot 2^{(ex+ey)}$

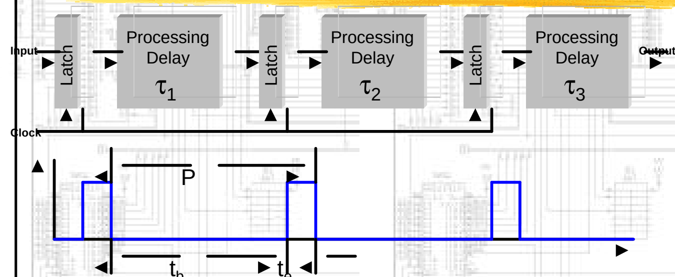
Floating-point Multiplication



Pipelining

- A pipeline is an assembly line
- A large task is decomposed into a serial sequence of:
  - Sub-tasks
  - Stages
- Each stage has an input latch to store its current subtask.

Schematically



P = Clock Period = $t_b + t_e$
t_b ³ max $\{\tau_1\ \tau_2\ \tau_3\}$= max processing delay
t_e ³ set + hold time for latches

Pipeline Performance

- Consider a pipeline with m stages
- Each with propagation delay τ_i ,i=1,m
- Which is clocked at the maximum possible rate
  - $\tau = P_{min} = \max(\tau_i) + t_e$

Piped and Non-piped

- Then we define τ_{pipe} and τ_{seq} as the time taken by piped and non-piped (sequential) to process n inputs:

$$\tau_{pipe} = m \times \tau + (n - 1) \times \tau$$

$$= (m + n - 1) \times \tau$$

$$\tau_{seq} = n \times \sum_{i=1}^{m} \tau_i$$