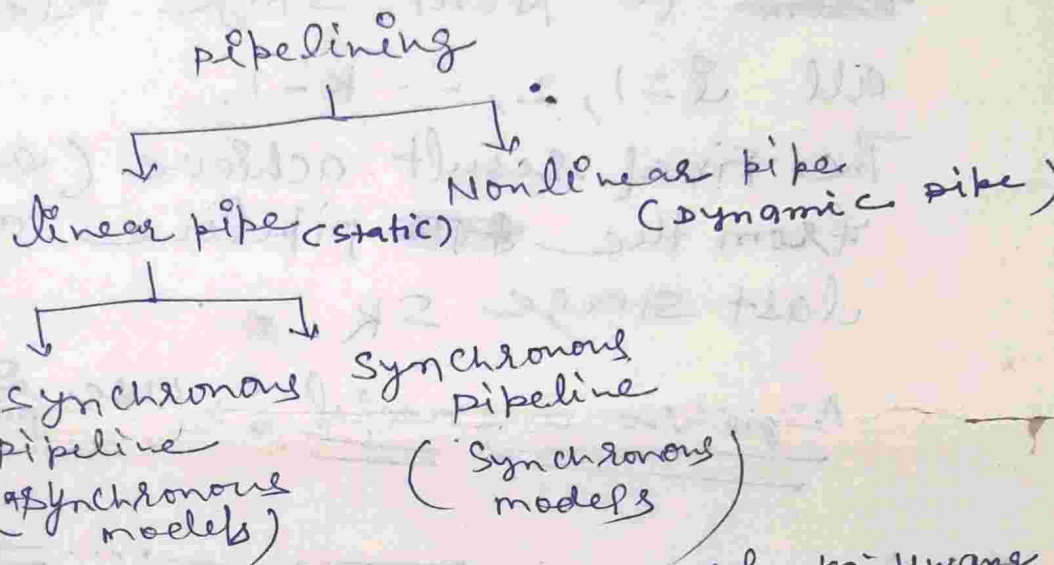
Pipelining :→ cascade of the processing stages which are linearly or nonlinearly (variable function) connected to perform the operation on the stream of data, flowing from one end to other end.



Ref. - Kai-Hwang CH. pipelining & parallel processors

Linear pipeline processors →

The Linear pipeline processor is a cascade of processing stages which are linearly connected to perform a fixed function over a stream of data flowing from one end to other end.

In modern computers linear pipeline are applied for instruction execution: Arithmetic computation and Memory Access Operations

It is divided into two types. or
Asynchronous & synchronous models

A linear pipeline processors is constructed with K processing stages. External Inputs (operands) are fed into the pipeline at the first stage. The Result of S_i stage is passed to next Stage S_{i+1} for all $i = 1, 2, \cdots k-1$.
The Final result achieve (emerges) from the pipeline at the last stage S_k.

Asynchronous model :→ (use in message passing)

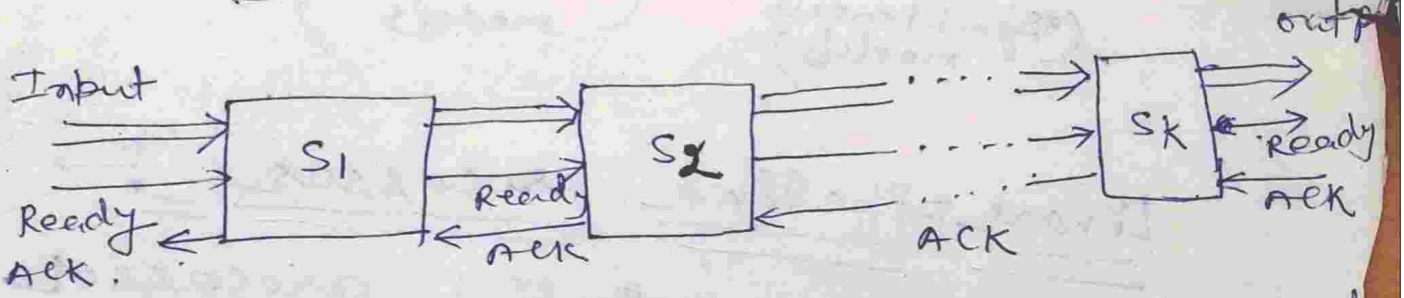


fig: An Asynchronous pipeline mod

the data flow between adjacent stages in an Asynchronous pipeline is controlled by a handshaking protocol.
when stage S_i is ready to transmit, it send a ready signal to stage S_{i+1}.
After stage S_{i+1} receives the incoming data, it return an Acknowledge signal to S_i.

Synchronous model :→

In synchronous pipeline, the latch is use to store data between adjacent stages.

In synchronous pipeline, there is approximately ~~same~~ equal delay on all stages.

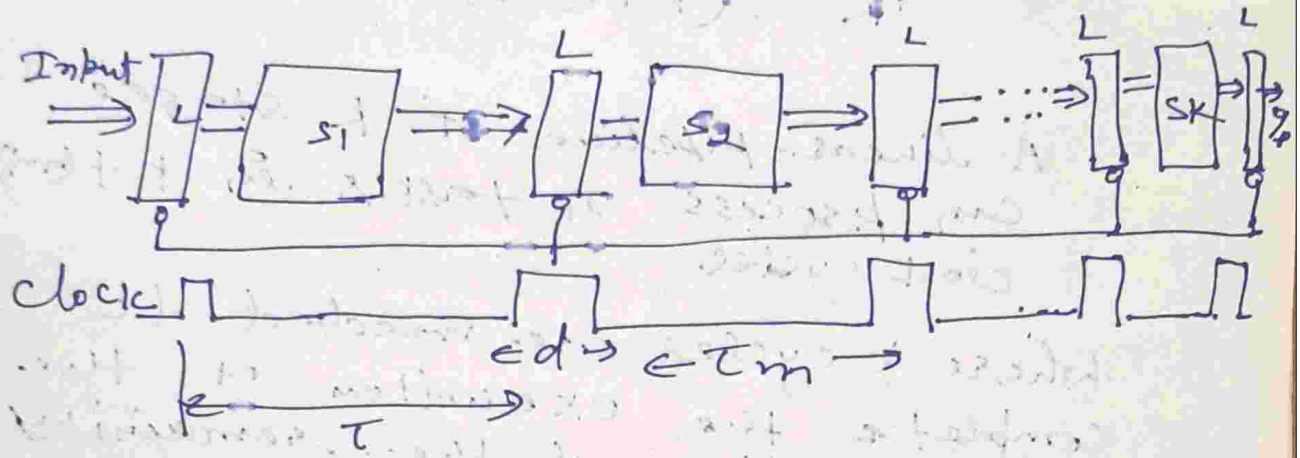


fig: A synchronous pipeline model.

S_i = stage i
L - Latch
τ - clock period or clock cycle
t_m - Maximum stage delay
d - delay on Latch.
ACK - Acknowledge signal.

So we can say that all the stages are synchronize with the help of Latch.

$$\boxed{\tau = t_m + d} = \max_i \{\tau_i\}_1^K + d$$

Here - $t_m >> d$

Pipeline frequency = $\boxed{f = \frac{1}{\tau}}$

Speed up, Efficiency and Throughput

$$\text{Speed up} = \frac{\text{Non pipeline process time}}{\text{pipeline process time}}$$

$$\frac{T_1}{T_k} = \frac{nkT}{(k+(n-1))T} = \frac{nk}{k+(n-1)}$$

A linear pipeline of k stages can process n tasks in $k+(n-1)$ clock cycles.

where k cycles are needed to complete the execution of the first task and the remaining $(n-1)$ task require $(n-1)$ cyc

so $T_k = [k+(n-1)]T$

& Non pipelined time $T_1 = nkT$

so- $$S_k = \frac{nk}{k+(n-1)}$$

$$S_k = \frac{nk}{n\left[\frac{k}{n} + (1-\frac{1}{n})\right]}$$

$$= \frac{k}{\frac{k}{n} + 1 - \frac{1}{n}}$$

if $n \to \infty$

then $S_k \to k$

The maximum speedup is $S_k \to k$ as $n \to \infty$. This maximum speedup is very difficult to achieve because of data dependences between successive tasks (instructions), interrupts etc.

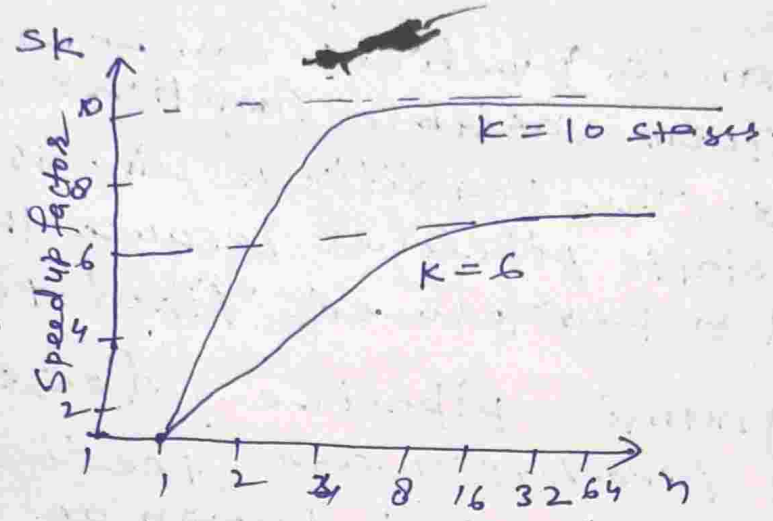


Fig: Speed up factor as a function of numbers of operations.

Efficiency & Throughput :→

The efficiency E_k of a linear k stage pipeline is defined as.

$$E_k = \frac{S_k}{k} = \frac{n}{k+(n-1)}$$

the efficiency approaches 1 when $n \to \infty$, and a lower bound on E_k is $1/k$ when $n = 1$. The pipeline throughput H_k is defined as the numbers of tasks (operations) performed per unit time.

$$H_k = \frac{n}{[k+(n-1)]\tau} = \frac{nf}{k+(n-1)} .$$

$$\boxed{H_k = \frac{nf}{k+(n-1)}}$$

The maximum throughput f occurs when $E_k \longrightarrow 1$ as $n \longrightarrow \infty$.

Nonlinear pipeline processors :→

A dynamic pipeline can be reconfigured to perform variable functions at diff t's times. The traditional linear pipelines are static pipelines because they are used to perform fixed functions

A Dynamic pipeline allows feedforward and feedbackward (feedback) connection So Dynamic pipeline is known as nonlinear pipeline

Reservation and Latency Analysis :→

A static pipeline, it is relatively easy to partition a given function into a sequence of linearly ordered subfunction. However function partitioning in a dynamic pipeline become quite involved because the pipeline stages are interconnected with loops in addition to streamline connections.

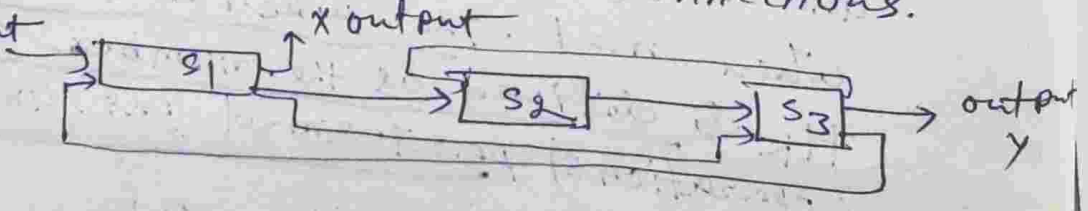


Fig: A three stage pipeline

A multifunctional dynamic pipeline is shown in fig. This pipeline has 3 stages. Besides the streamline connection from S_1 to S_2, and from S_2 to S_3,

there is a feed forward connection from S_1 to S_3 and two feedback connections from S_3 to S_2 and S_3 to S_1.

These feedforward and feedback connections make the scheduling of successive events into the pipeline a nontrivial task. With these task the output of pipeline is not necessarily from the last stage. In fact, following different dataflow patterns one can use the same pipeline to evaluate 'diff's functions.

	1	2	3	4	5	6	7	8
S_1	X					X		X
S_2		X		X				
S_3			X		X		X	

Fig: Reservation table for function X

	1	2	3	4	5	6
S_1	Y				Y	
S_2			Y			
S_3		Y		Y		Y

Fig: Reservation table for function Y

✓ The reservation table for a dynamic pipeline becomes more interesting because a nonlinear pattern is followed.

✓ Given a pipeline configuration, multiple reservation tables can be generated for the evaluation of diff's functions.

Latency cycle & latency →

Latency :→ The number of time units (clock cycles) between two initiations of a pipeline is the latency between them. The latency value must be non negative integer.

latency cycle :→ A latency cycle is a latency sequence which repeats some subsequence

Forbidden latency → latency that cause the collision, which is known as forbidden latency.

INSTRUCTION pipeline :→

A stream of instructions can be executed by a pipeline in an overlapped manner, is known as instruction pipeline.

There is diff'n - diff'n type of instruction pipeline

- 2-stage pipeline
- 5-stage pipeline
- Seven stage pipeline

Fig:— two stage pipe

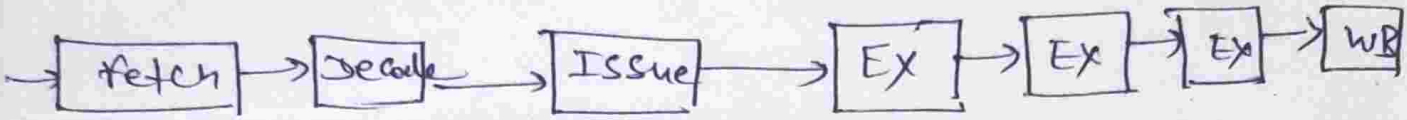


Fig: Seven stage pipeline

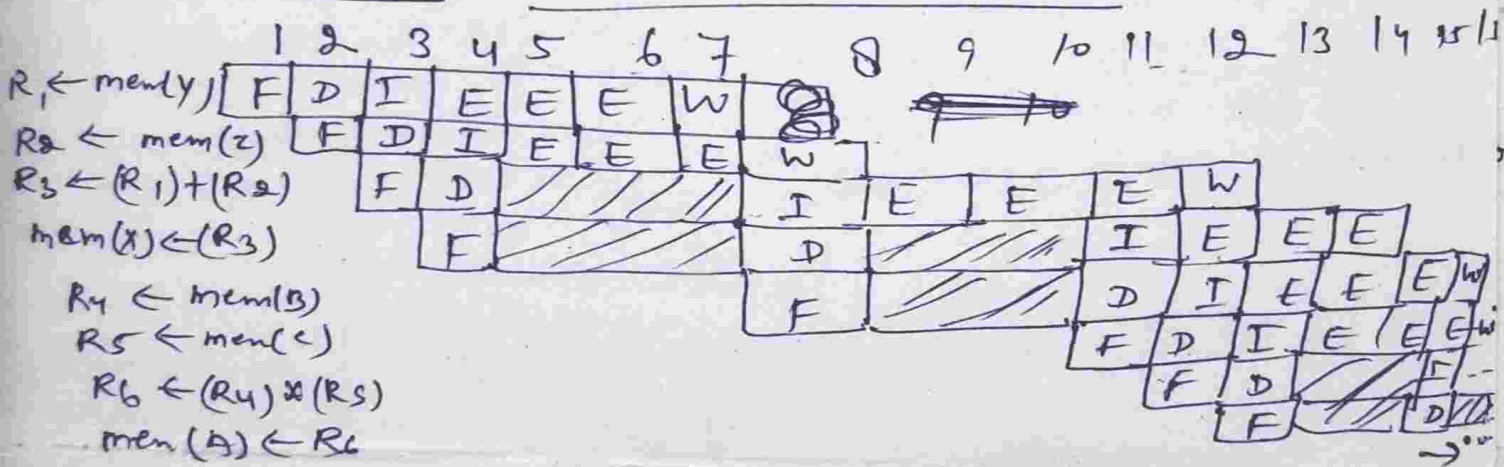


Fig: in order instruction issue.

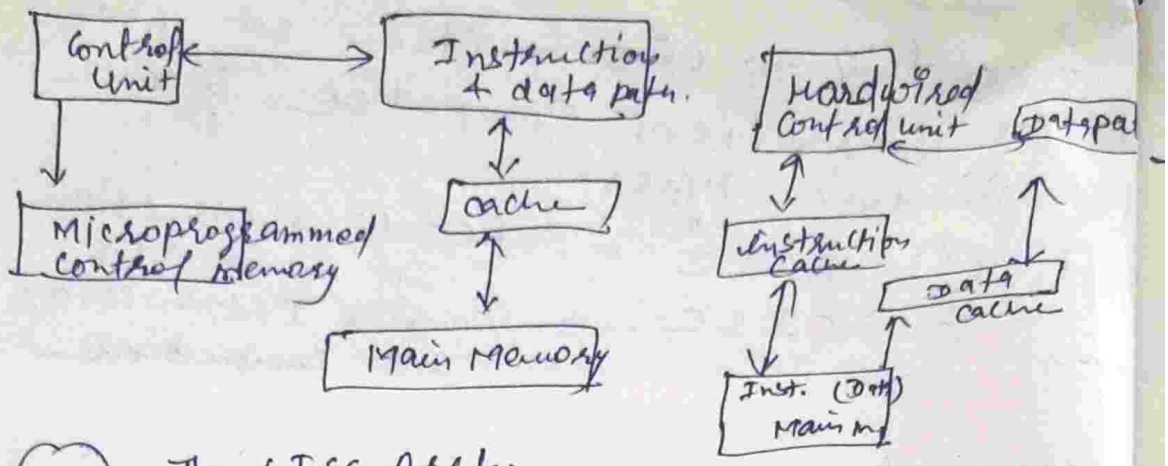
Ref: M. Morris mano / Computer system Arch.
CH. pipeline and vector processing
Page: 312

Instruction set Architecture

According to instruction set Arch. we classified Computer system in two types

1. RISC → Reduce instruction set computer
2. CISC → Complex Instruction set computer.

Architectural Characteristic	CISC	RISC
1- Instruction set size and instruction set format	Large set of instruction with variable formats (16-64 bits per instruction)	small set of instruction with fixed (32 bit) format and most Register Based Instruction
2. Addressing Modes.	12-24	Limited (3 to 5)
3 General purpose Registers & cache design	0-24 GPRs with a unified cache for instructions & data, Recent designs also use split caches	Large number of GPRS (32-192) There is split data & instruction cache.
4 Requirement	lower end. (low speed)	use at highest end (high speed)
5- CPI	CPI between 2 to 15	Less CPI or CPI < 1.5
6 CPU Control	Microprogram. Controlled (use ROM)	Hardwired controlled



(a). The CISC Arch. with Microprogrammed Control & unified cache

(b). The RISC Arch with Hard wired control & split instruction & data Cache.

fig: Distinctions between typical RISC & CISC processor Arch.

CISC → CISC stands for Complex instruction set computer.

A CISC computer has 120 to 150 instructions, which is variable length.

→ It use ~~small~~ less GPRS (8 to 24)

→ There is no split cache.

→ They ~~has~~ have ~~high~~ High CPI

RISC → RISC stands for Reduce instruction set computer.

A RISC computer has ~~a~~ less than 100 instruction. with fixed length (32 bit)

→ It use large number of GPRS (32 to 192)

→ There is split cache

→ they have less CPI. (CPI < 1.5)

Ref: Kai Hwang / RISC, CISC

Arithmetic pipeline design :->

pipeline Arithmetic units are usually found in very high speed computers. They are used to implement floating point operations, multiplication of fixed point numbers.

ex:- The ~~inputs~~ input of floating point shown in below, in Normalised floating point binary numbers.

$$X = A \times 2^{a} \longrightarrow \text{exponents}$$
$$Y = B \times 2^{b}$$
$$\text{mantissa}$$

A and B are two fractions that represent the mantissa and a and b are the exponent.

the floating point addition & subtraction can be performed in four segments.

(1) - Compare the exponents.
(2) - Align the mantissa.
(3) - Add or subtract the mantissa.
(4) - normalise the result.

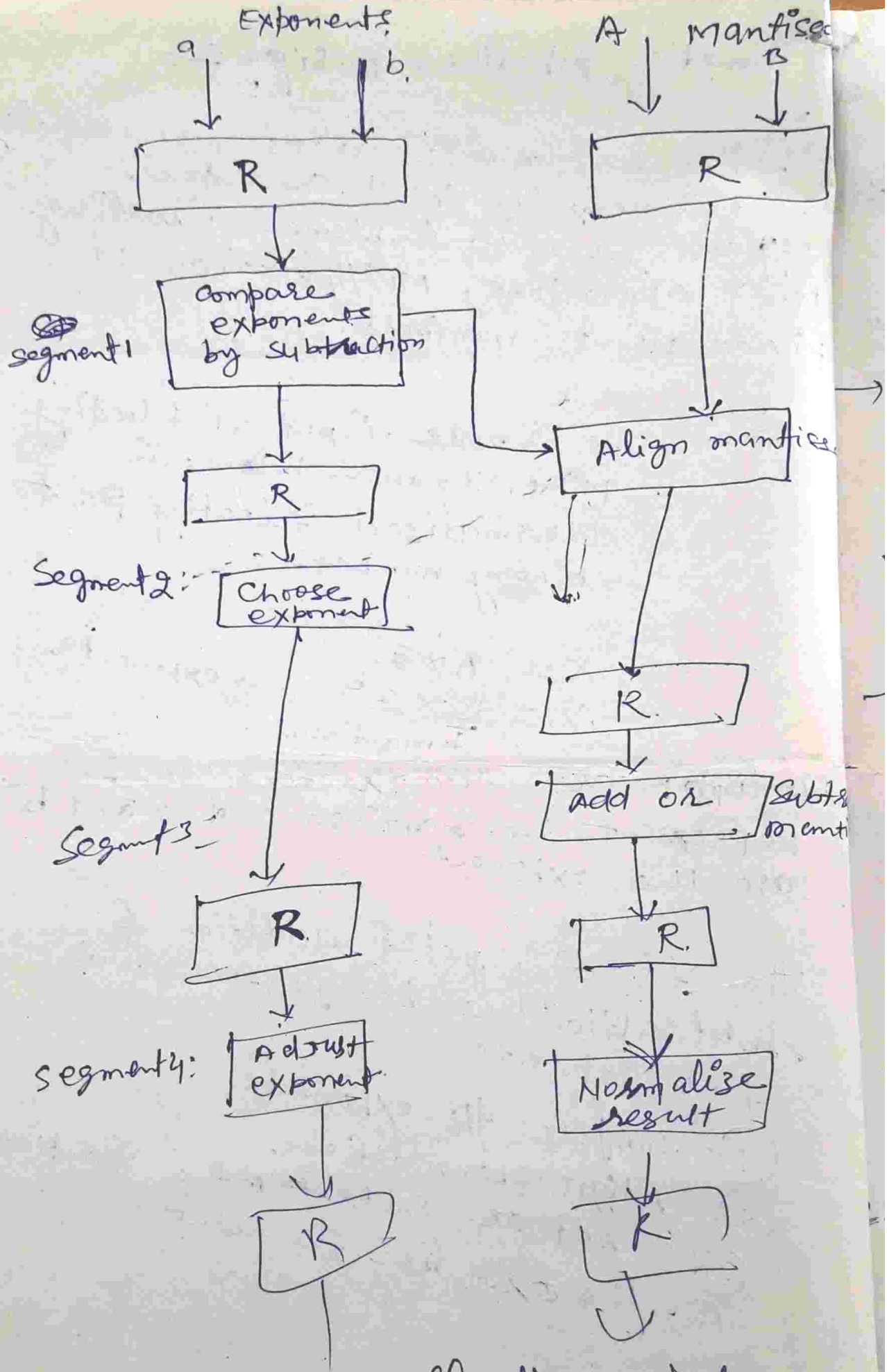


fig :- → pipeline for floating point addition & subtraction

$X = \underbrace{0.9504}_{\text{Mantissa}} \times 10^{3}$

Fraction

$Y = 0.8200 \times 10^{2}$

→ The exponents are subtracted in the first segment to obtain

$3 - 2 = 1$

The large exponent 3 is chosen as the exponent of the result (segment 2)

The next segment shift the mantissa of Y to the right to obtain

$X = 0.9504 \times 10^{3}$

$Y = 0.0820 \times 10^{3}$ ← one Right shift

Align mantissa

This align the two mantissa under the same exponents.

The addition of the two mantissas in segment 3 produces the sum.

$Z = 1.0324 \times 10^{3}$

The Sum is adjusted by normalized the result so that it has a fraction with a nonzero first digit.
This is done by shifting the mantissa once to the right and increment the exponent by one to obtain the normalize Sum

$$Z = 0.10324 \times 10^{4}$$

Ref: Computer system Arch- M. morris mano page- 309