Data Flow Computers and VLSI Computation.

Control Flow Vs Data flow computer

Data flow computers are based on the concept of data-driven computation which is different from the operation of a parallel processing machine

The fundamental difference is that instruction execution a parallel computer is under program flow control, whereas in a dataflow computer is driven by the data availibility.

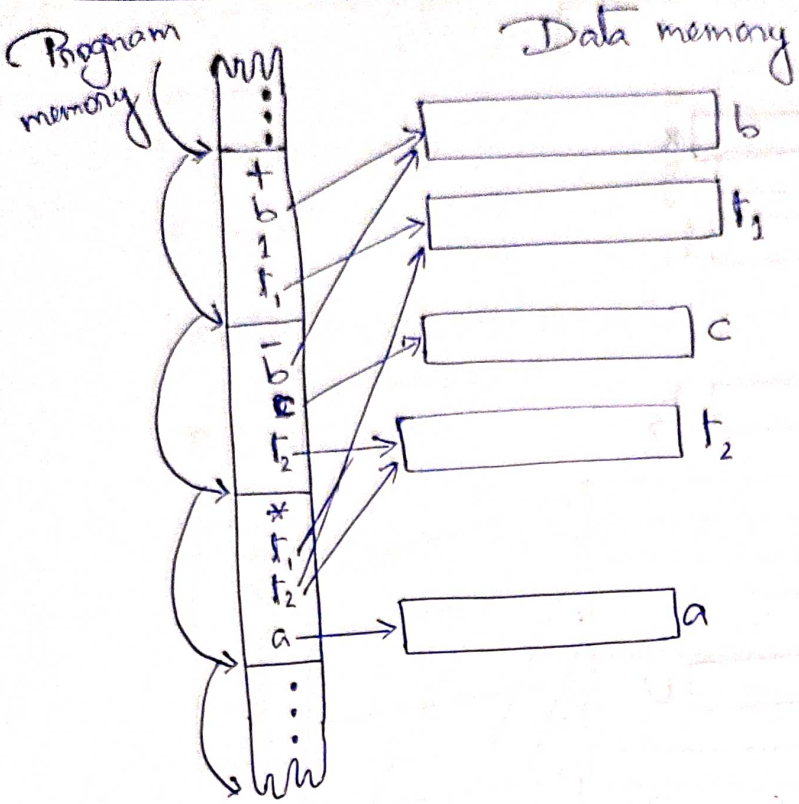
The concept of control flow & data flow computioning distinguished by the control of computation sequences in two distinct program representation

1. Sequential control flow
2. Parallel control flow

In sequential control flow a series of can be specified by a series of instruction with an explicit flow of control. Shared memory cells are the means by which data is passed between instructions. data (operands) are reffered by their memory addresses (variables). In traditional sequential control flow there is a single thread flow of control which is passed from instruction to instructions. Data is passed bet^n instructions via references to memory cells. Flow of control is implicitly sequential but special control operators can be used explicitly for parallelism. Program counters (PC) are used to sequence the execution of instruction in a centralized control example.

Date
26/12/2022

CONTROL FLOW

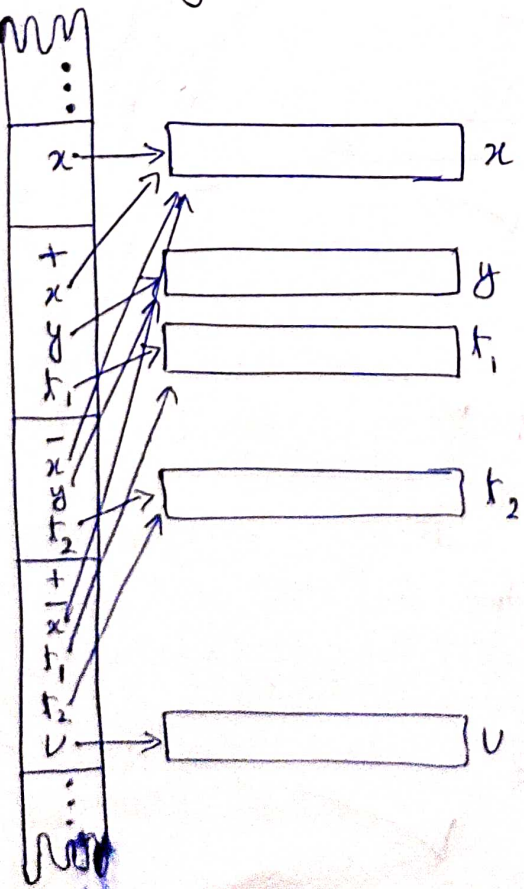


$$\left[\begin{array}{c} a = (b+1)(b-c) \\ \quad\ t_1 \quad\ t_2 \\ a = t_1 * t_2 \end{array}\right]$$

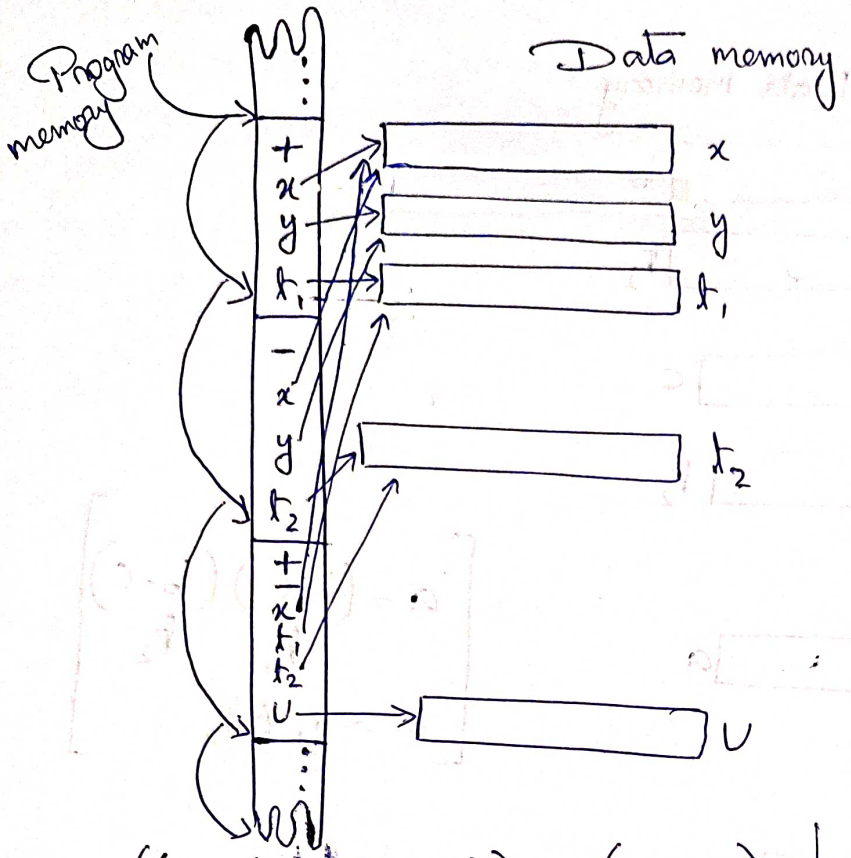
(a) Sequential Control Flow.

* Operation :- $v = (x + (x+y) - (x-y))$

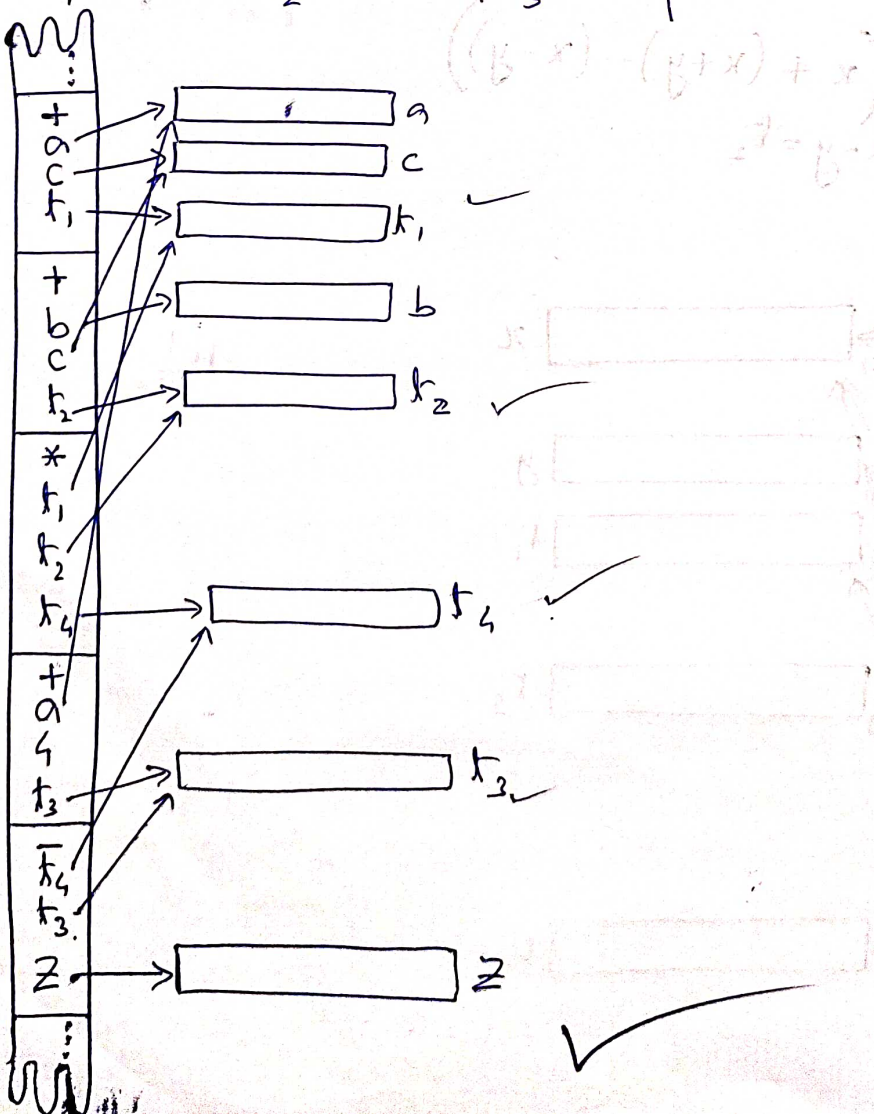
Let $x+y = t_1$, $x-y = t_2$



(1) $U = (x + (x+y) - (x-y))$
t_1 t_2



(2) $Z = ((a+c) * (b+c)) - (a+c)$ | Let $(t_1 * t_2) = t_4$
t_1 t_2 t_3



$$S = \frac{(x+y) * (W-Z) - (x+Z)}{(x+y+Z)}$$

Let $(x+y) \longrightarrow t_1$

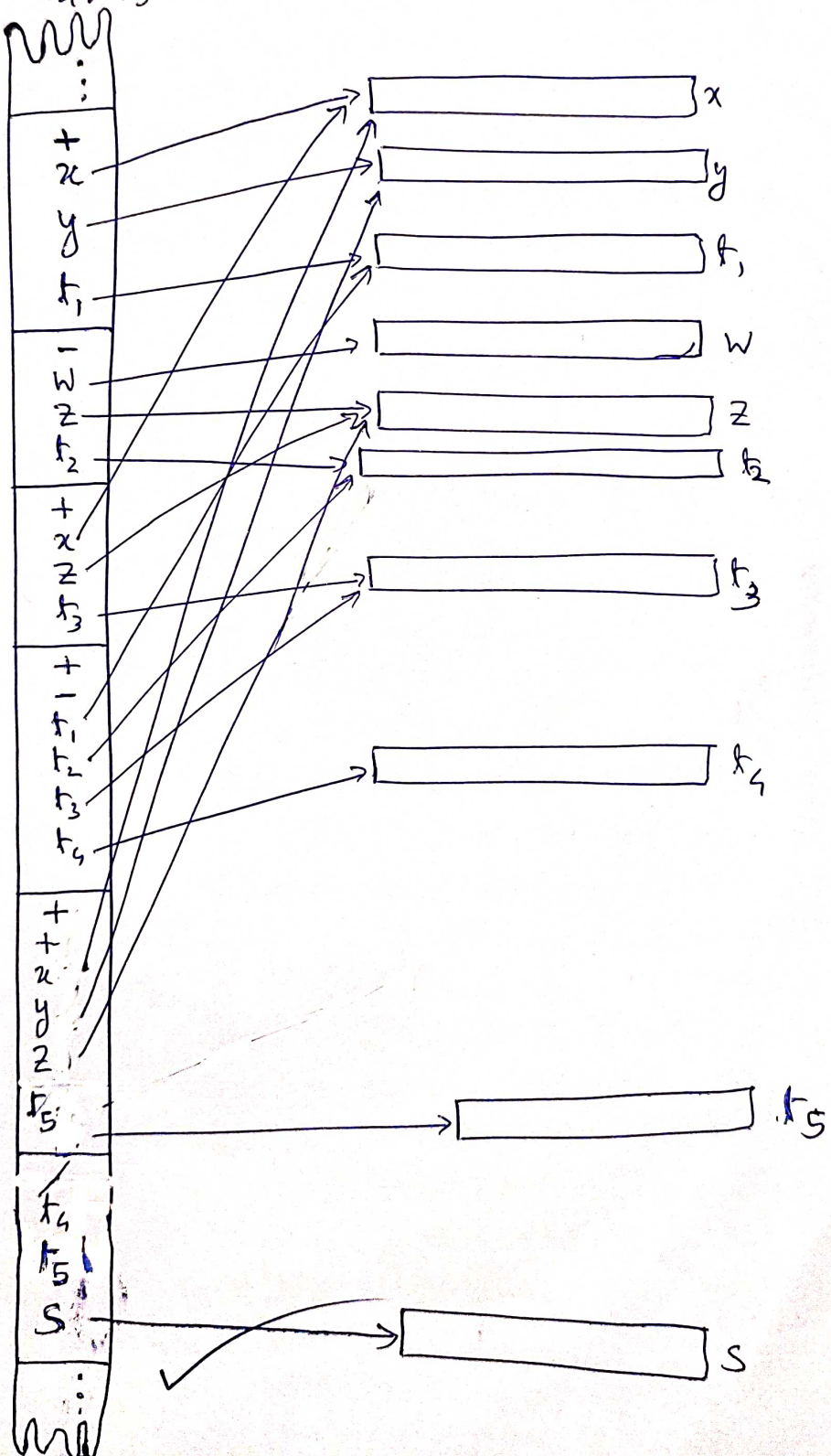
$(W-Z) \longrightarrow t_2$

$(x+Z) \longrightarrow t_3$

$t_1 + t_2 - t_3 \longrightarrow t_4$

$(x+y+z) \longrightarrow t_5$

$t_4 / t_5 \longrightarrow S$

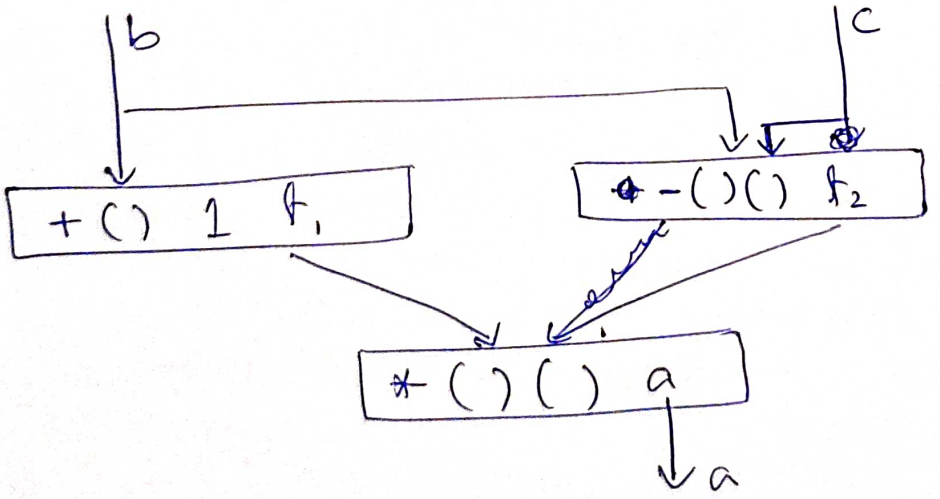


Date 12/12/2022

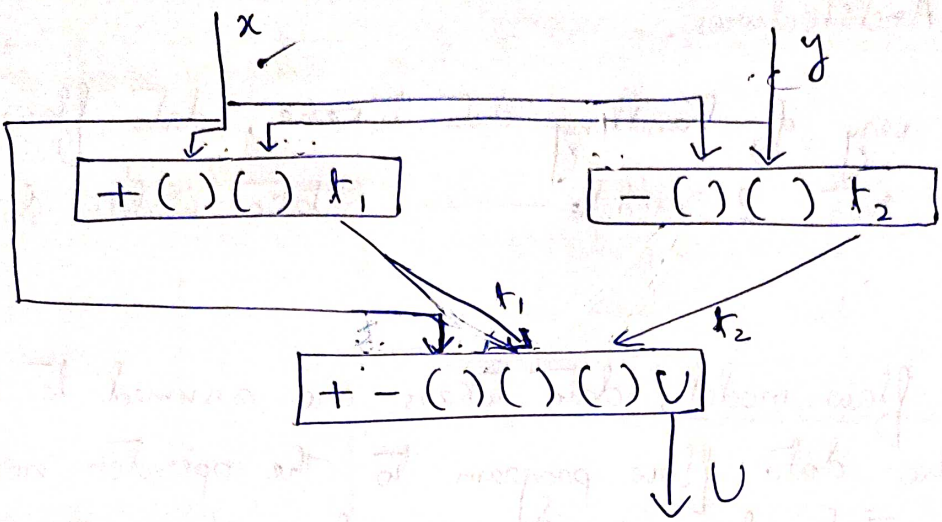
In a data flow computing, instructions are activated by the availability of the data tokens as indicated by ~~token~~ ( ). Data flow programs are represented by directed graphs which show the flow of data between instructions. Each instruction ~~is~~ is consist of an operation, 1 or 2 operand & 1 ~~or~~ or more destination to show the result. Some special features of a data flow model are as follows :—

1) Intermediate or final results are passed between the instructions.
2) There is no concept of shared data storage of a variable.
3) Program sequencing is constrained ~~by~~ only by data dependency among instructions.

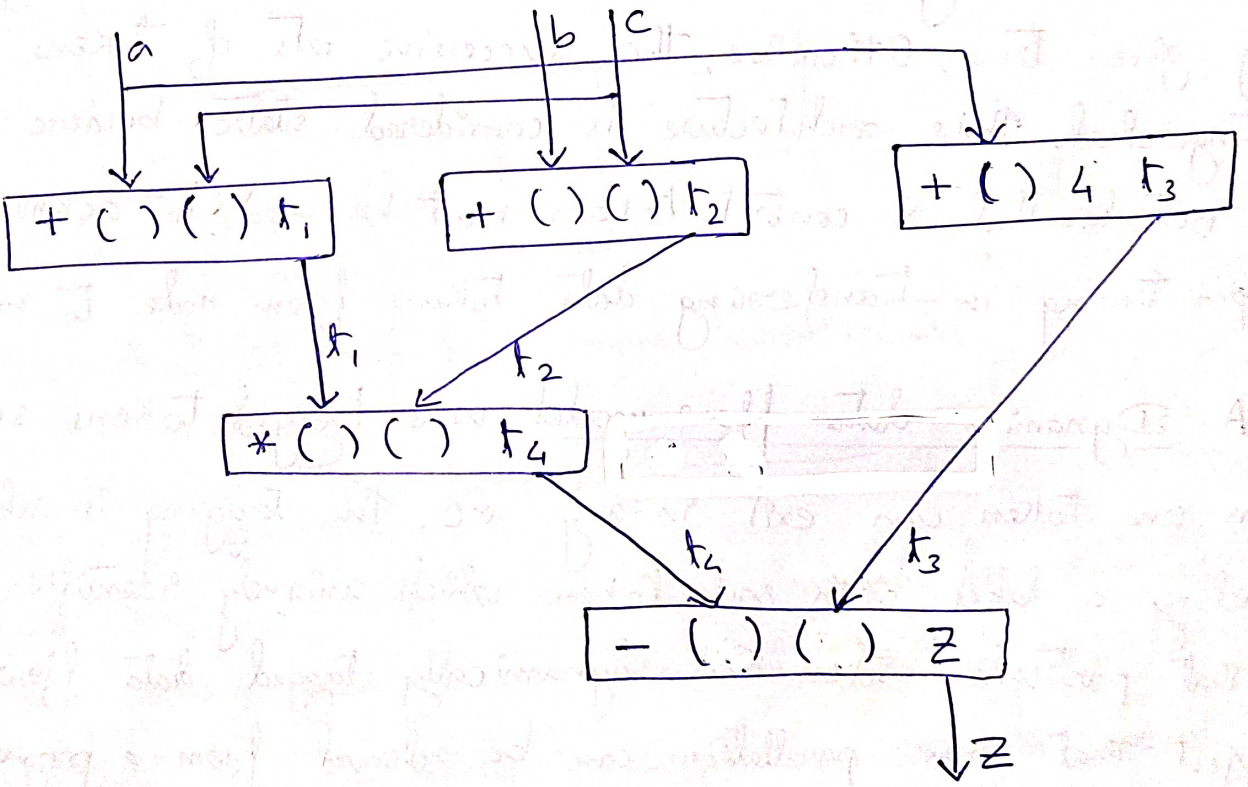
* $a = (b+1) * (b-c)$



* $V = x + (x+y) - (x-y)$



* $Z = ((a+c) * (b+c)) - (a+4)$



Date
14/12/2022

Data Flow Architectures.

Depending on the way of handling data tokens, data flow computers are divided into 2 models ——— Static models & Dynamic models.

1) In Static data flow models, data tokens are assumed to move along the arcs of the data flow program to the operation nodes. The operation gets executed when all its operand are present at the input arcs. Only one token is allowed to exist on any arc at any given time. Otherwise, the successive sets of tokens can't be distinguished. This architecture is considered static because tokens are not labelled & control tokens must be used to acknowledge the proper timing in transferring data tokens from node to node.

2) A Dynamic data flow model uses tagged tokens so that more than one token can exist in any arc. The tagging is achieved by attaching a label with each token which uniquely identifies the context of that particular token. This dynamically tagged data flow model ~~can~~ suggest that maxm parallelism can be achieved from a program.

Both static & dynamic data flow architecture have a pipelined ring structure. The ring contains 4 resources ——— memories, processors, the routing network & the I/O unit. ~~The~~

→ The memories are used to hold the instruction packets.

→ The processing unit forms the task force for parallel execution of enabled instructions.

→ The routing network is used to pass the result tokens to their destination. The I/O unit serves as an interface between the data flow computer and the outside world.

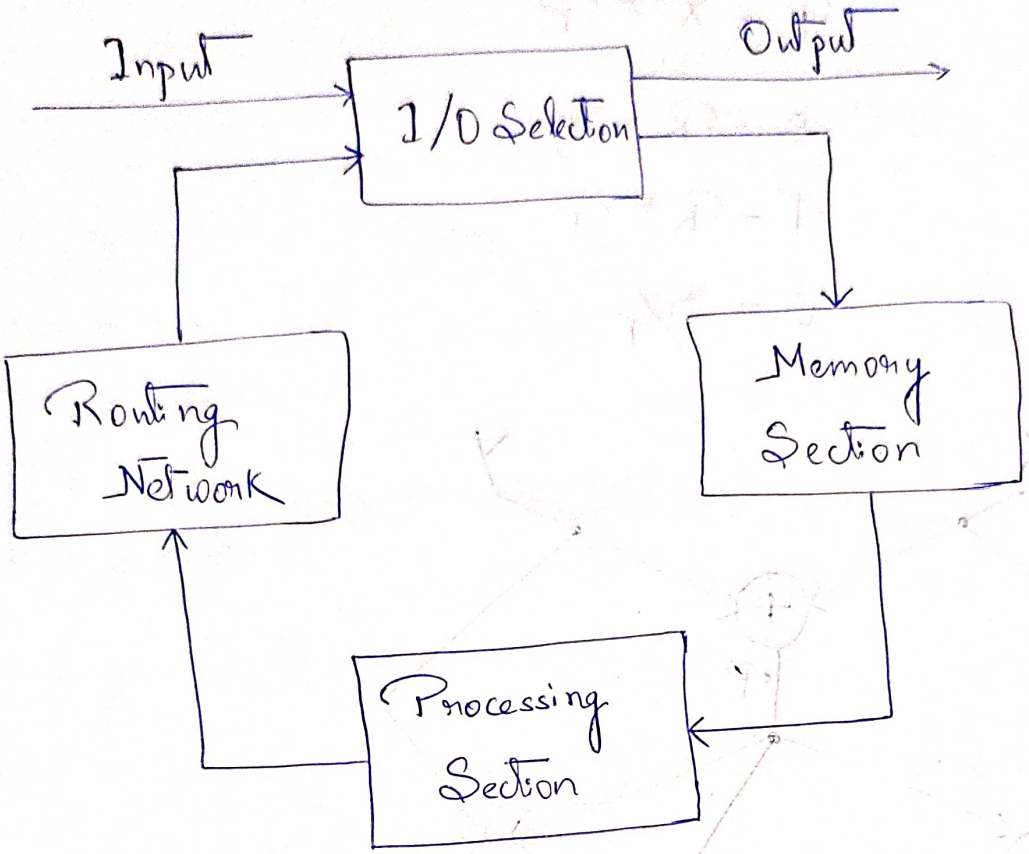


Fig:- Ring structured data flow computer architecture.

Data Flow Graphs. For Languages:

There is a need to provide a high-level language for data flow computers. The primary goal is to take advantage of implicit parallelism. Data flow computing is compatible with the use of dependence graph for compiling high-level languages. An efficient data flow language should be able to express parallelism in a program more naturally to promote programming productivity & to facilitate close interactions bet^n tokens & hardware structures.

Eg:-

We have, $P = X + Y$.

$$Q = P / Y$$

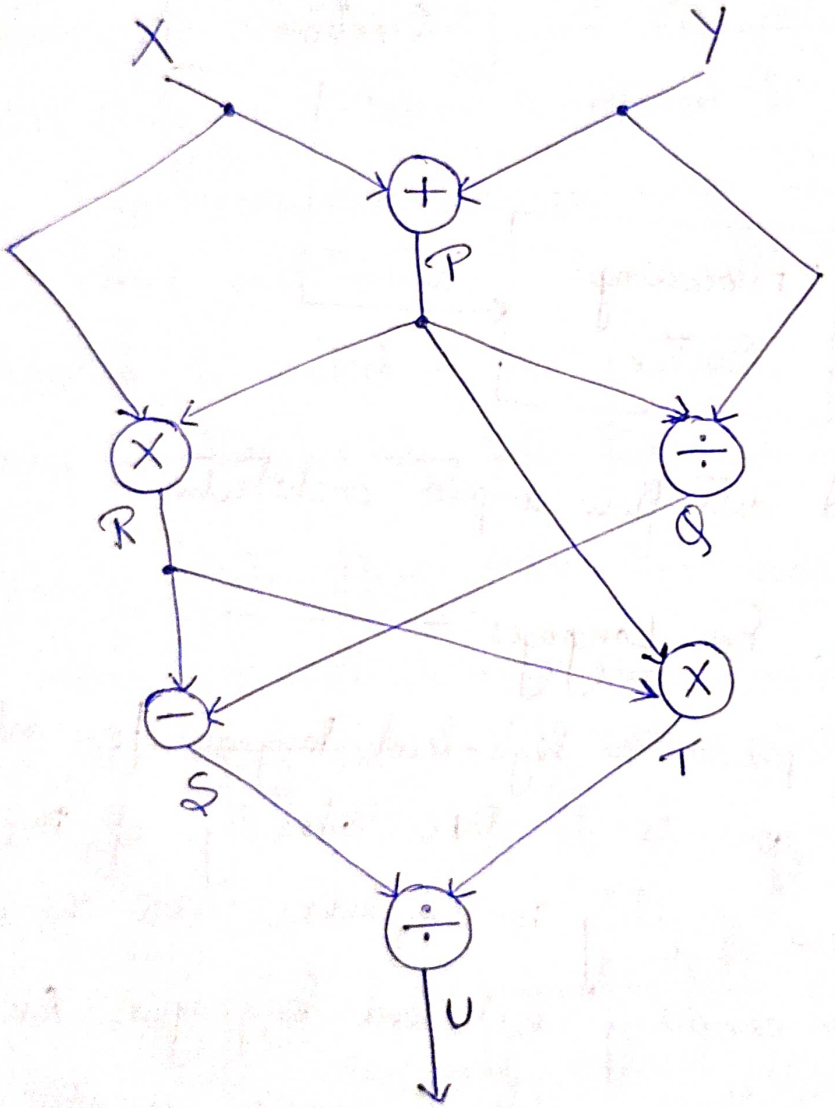
$$R = X * P$$

$$S = R - Q$$

$$T = R * P$$

$$U = S / T$$

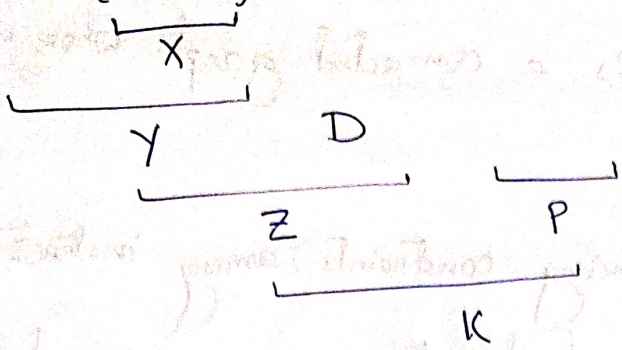
↳



$$U = (X * (X+Y) - (X+Y) \div Y) \div (X * (X+Y) * (X+Y))$$

P, R, Q, S, R, P, T, S, U

$K = (A + (B * C) + D) * (E - F)$



Let,

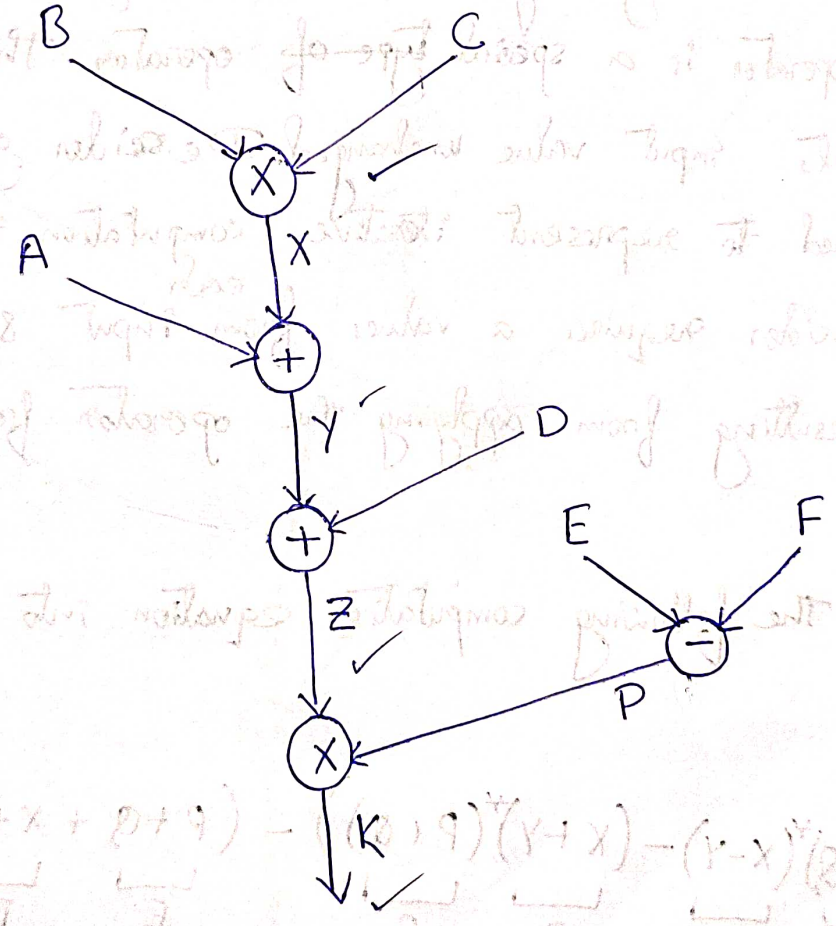
$X = B * C$

$Y = A + X$

$Z = Y + D$

$P = E - F$

$K = Z * P$



Date
15/12/2022

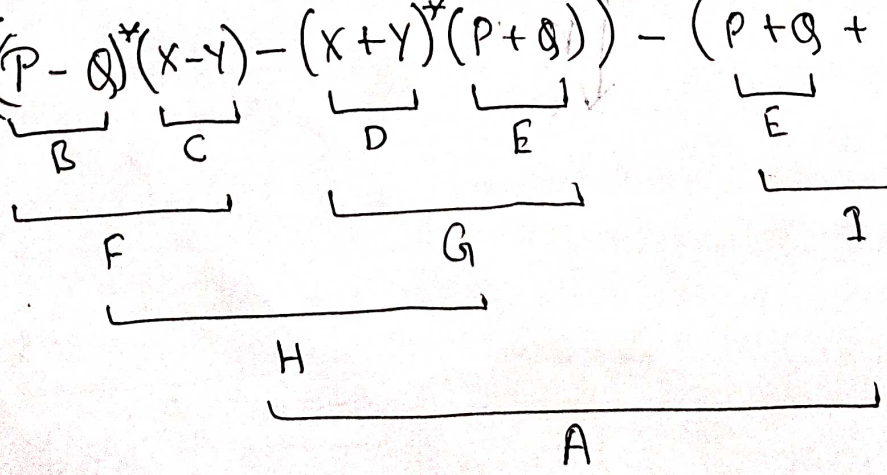
A Data flow graph is a connected graph whose nodes corresponds to operator

The graph demonstrates sequencing constraints among instructions. In data flow computer, the machine level programs are represented by data flow graphs.

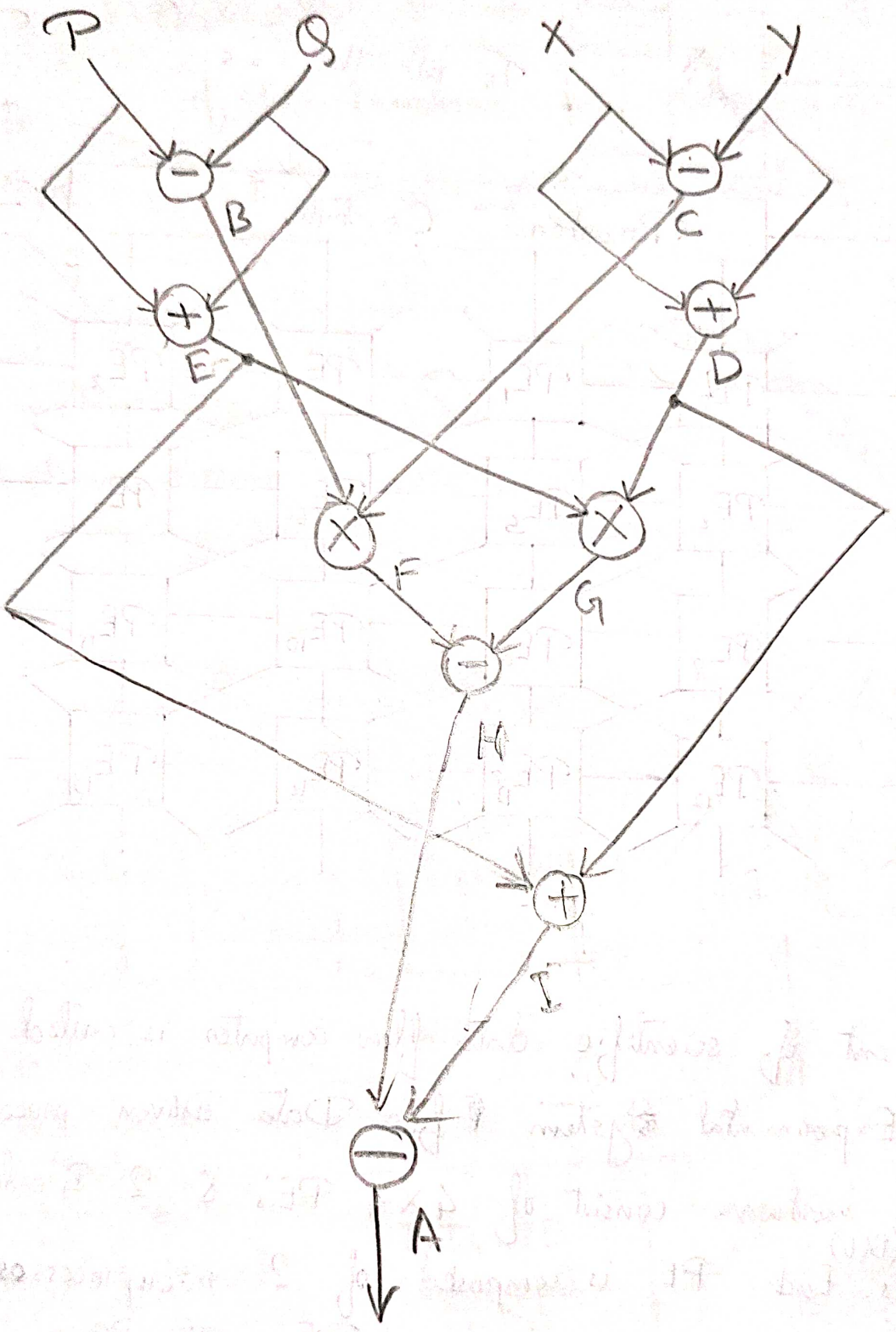
Two type of links on data flow graphs are available. The purpose is to distinguish those for numerical data from those for boolean variables. Numerical data ~~tra~~ transmits integer, real nos and boolean data carry only boolean values for control purposes. An identity operator is a special ~~type of~~ operator that has 1 input and transmits its input value unchanged. Decider gates and merge operators are used to represent iterative computation in data flow graphs. A decider requires a values from each input & produces a true value resulting from applying the operator from the values received.

Q:- Represent the following computation equation into a data flow graph.

* $A = ((P-Q)*(X-Y) - (X+Y)*(P+Q)) - (P+Q+X+Y)$

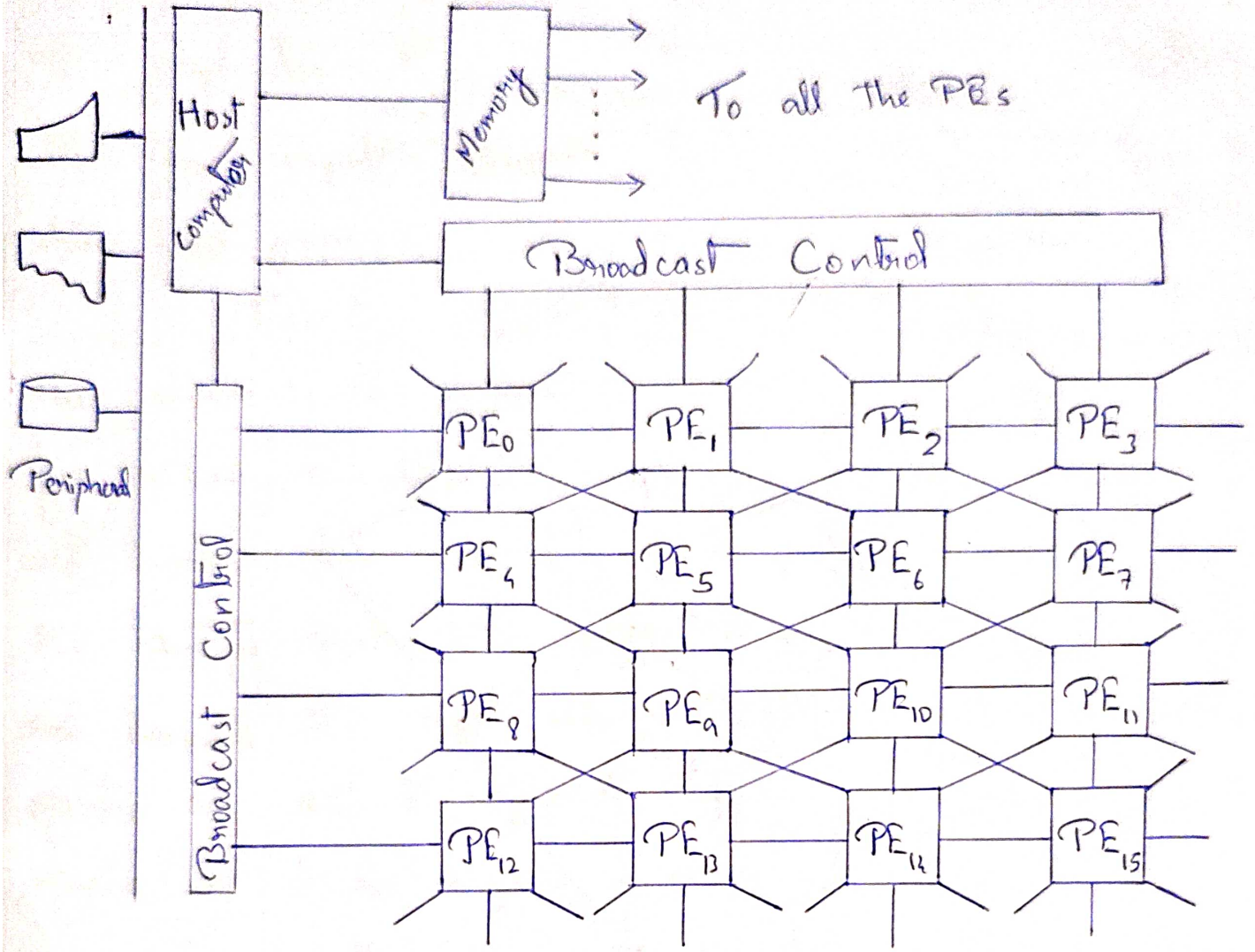


B



16/12/2022

Architecture Of EDDY:—



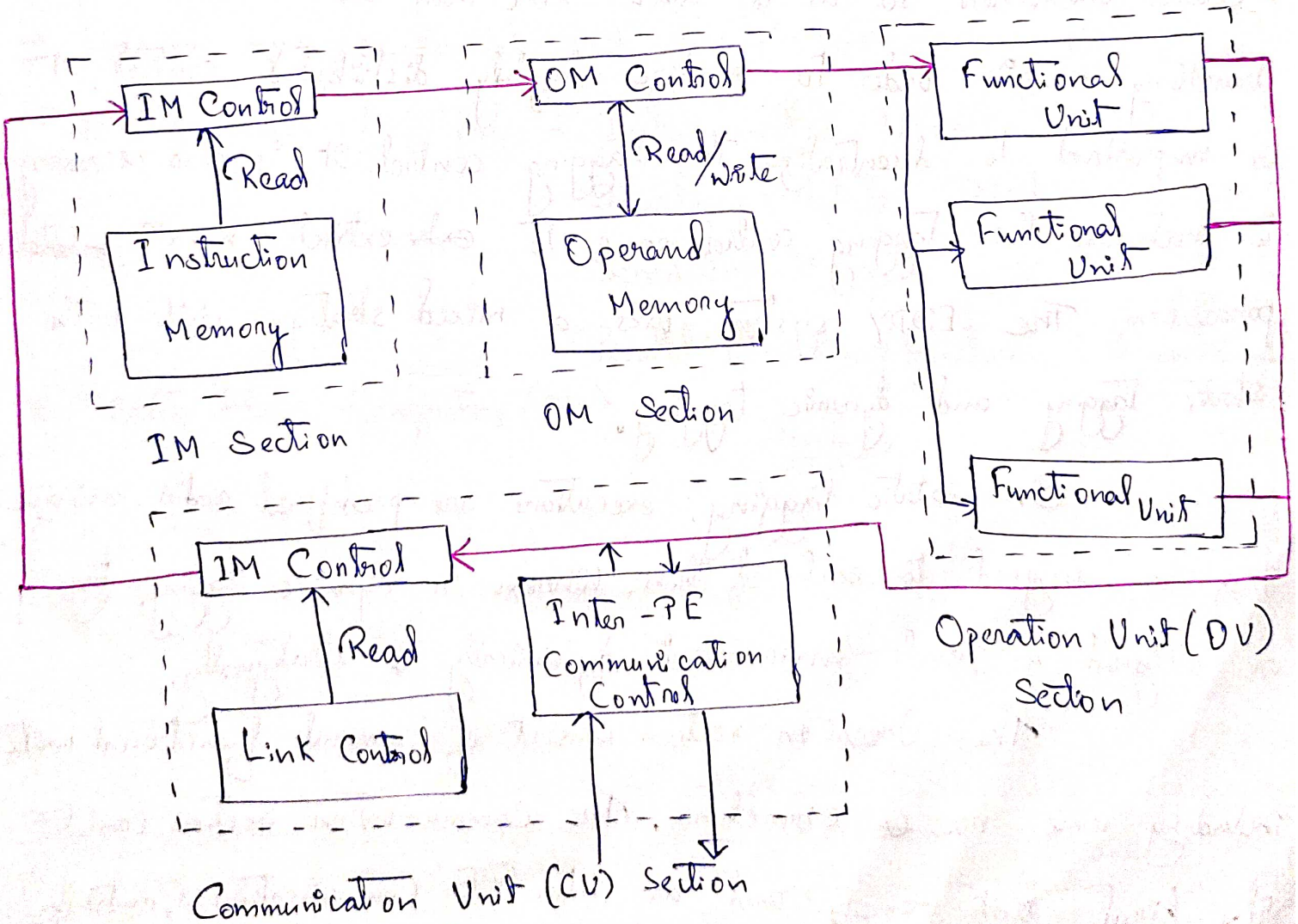
The development of scientific data flow computer is called EDDY Architecture (Experimental System for Data Driven processor array). The hardware consist of 4×4 PE's & 2 Broadcast Control Units (BCU). Each PE is composed of 2 microprocessor & connected directly with 8 neighbouring PEs. The BC can load & unload programs and data to or from all the PEs in column or row at the same time.

Date 13/12/2022

The software system implemented on each simulator simulates the circular pipeline data flow control of each PE using logical clocks. AOT generates statistical data such as operation rates of the functional unit and average queue length. The simulation results will be used to help develop the custom designed PE H/W.

The custom design PE is a circular pipeline consisting —

① Instruction Memory Section.
② Operand Memory Section
③ Operation Section
④ Communication Section.



On the arrival of each operand, the instruction memory fetches its operation node & send both the fetched instruction and operand data to the operand memory section (OM). If the arrived data is ~~an~~ data for a 2 operand operation, the OM searches for its its paired partner associatively. When the paired operand is found, an operation package is constructed and send to the Operation unit section (OU). When no paired operand is found, the arrived data token is stored in the Operand Memory with a key attached.

All data tokens are tagged which represents their execution environment so as to allow more than one token to be travelling on. In order to realize highly distributed control, it is important to decentralize the tagging control. It is also necessary to mechanise the tagging control so as to ~~exte~~ extract max^m ~~parallelism~~ parallelism. The EDDY system uses a mixed strategy with both static tagging and dynamic tagging.

In static tagging, executions are predefined and a unique tag is assigned to each of them. Whereas in dynamic tagging, tags are assigned to each environment dynamically & strategically.

The Operation section consist of several functional units including some no. ~~of~~ crunchers. The Communication Section consist of final link memory and an inter-PE Communication Controller. This controller sends the result packets to the A link memory or to

other PEs, to the memory of the EDDY system and also receives result packets from other PEs and transmits them to its own link memory.

(Speed-Up, Scale-Up) Imp.

Date
20/12/2022

Imp.

* Level Of Parallel Processing
* Parallelism in Uniprocessor
* Pipeline Computers & Array Processor
* Classification of Architectural Scheme (Flynn, Feng, Handler)
* Vector Processor (Star-100, TI ASC, Cray-1, Cyber-205) Comparisions as well * imp.
* SIMD Array Processors, SIMD Interconnection Network.
* Hazards - Types, Detection & Resolution..
* Multiprocessor Architecture - Interconnection Networks
  Deadlock (Protection, Detection, Life Cycle)
  Scheduling Strategies
  Parallel Algorithms (Synchronous & Asynchronous)
* Data Flow Computers (All). ——— CIE.