Data flow architecture :-

Data flow architecture is a series of functions in computer software where each step is automatically generated by the actions of previous function. It is also known as reactive programming. As each function is explicitly programmed, there is no need to recalculated value in one variable when another variable changes.

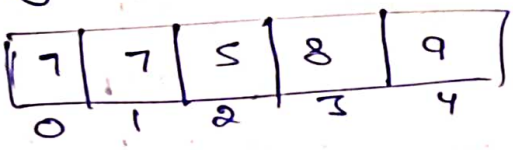
~~The concept of dataflow architecture has been compared to a factory assembly line. As with that kind of system,~~ each packet of information is optimized for top performance individually while also being integrated into the whole system. The info is arranged in a sort of pipeline, where it advances from one function to the other. One significant way in which the system differs is that each packet contains the information needed to connect it to the next link in the data flow chain. As dataflow architecture is considered to be a fairly simple form of programming. It is often used by less experienced programmers. With this method, a system can be programmed one packet

at a time. While data flow architecture is relatively easy to use, it also has some limitations. With packets dedicated to specific tasks, a system can function efficiently, but with only a certain amount of flexibility. ~~But if the system~~ Data flow architecture solves a lot of problems inherent in data flow and organization.

Introduction to Data level-Parallelism :-

Concurrent execution of the same task on each Multiple Computing Core.

Ex:- if we want to find the sum of the content of an array of size 5.



For a dual core system -

thread A - running on core 0, can sum the elements A[0] -- A[1] and

thread B, running on core 1, can sum the elements A[2] -- A[4].

So the two threads would be running in Parallel on separate Computing Cores.

In data level Parallelism we discuss about the processors.

Processors are two types :-

Data level Parallelism simply called as SIMD.

Three variations of SIMD machines:

1) vector architectures] ← 1D array
2) SIMD extensions] ← syllabus.. both are working on the array
3) Graphics Processors Units

1) Vector architecture :-

~~1) Load sets of data elements~~ with stone & nearly one by one are caude

There is a class of computational problem that are beyond the capabilities of a conventional computer. These problems require or vast no of computations that will take a conventional computer days or even weeks ~~needs~~ to complete.

- Computers will vector processing are able to handle such instructions & the. have applications in following fields.

↓

Applications.

- → Long range weather forecasting.
- → Problem exploring.
- → Medical diagnosis
- → Aerodynamics & space simulation
- → Artificial Intelligence & expert system
- → Image Processing.

} where vector parallelism are used.

These all are image forms, → graph represent ↓ vector → form of array. work.

⇒ Basically vector processing technique should be applied due to achieve the result of high performance in case of unaplicability of complex Computer. to speed up the execution time of ~~Execution.~~ instructions.

Complex instructions which operate on multiple data at the same time, require a better way of instructions execution, which was achieved by vector processor.

vector Processor, not only use instruction Pipeline. but it also pipelines the data, working on multiple data at the same time.

Defination: In Computing, a vector processor is a CPU that implements an instruction set containing instructions that operate on 1 dimensional array of data called vectors, & compared to scalar processors, whose instructions operate on single data items.

vector processors have high-level operations that work on linear arrays of numbers or vectors.

Scaler Quantity [Data]

Vector Quantity [Data | Data | Data | Data]

Vector operation:- $V = [v_1, v_2 v_3 \cdots v_n]$ → vector length.

The element v_i of vector V is written as v[I], where I is index refers is memory address or register number the number is stored.

Ex:- addition of values of 2 array:-

```
int a[10], b[10];
int c[10];
for (I=0; I<n; I++)
c[I] = a[I] + b[I];
```

C Program

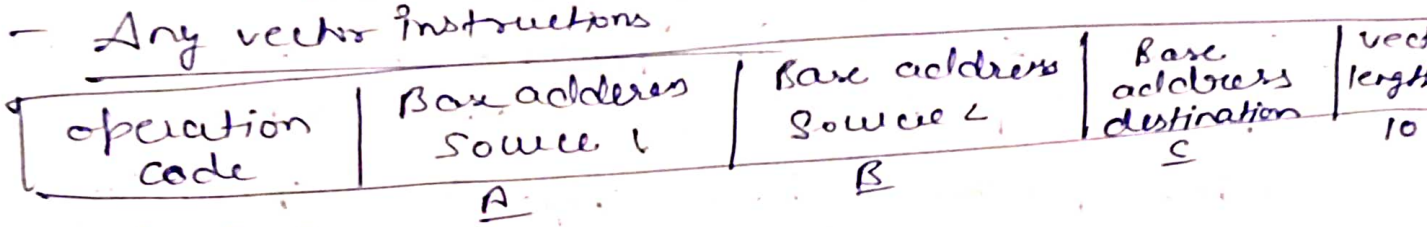
```
initialize i = 0
Read a[i], Read b[i]
Size c[i] = a[i] + b[i]
Increment I = I+1
if I<10, continue.
```

Assembly language.

Vector processors have the ability to remove overhead of Instructions: fetch & execute in loop instructions can be written as.

simple vector inst :- C(1:10) = A(1:10) + B(1:10)

- Any vector instructions



Ex- ADD

Vector architecture

1) Read set of data elements into vector register
2) Operate on vector registers.
3) Disperse the results back into memory.
4) Single instruction operate on a vector of data.
5) Vector load and store are deeply pipelined
6) High memorory latency once per load & store
7) Amortize the latency overhead size.

Intro to ~~VIII~~ V-MIPS: architecture :-

1) Vector Registers:-
   - → 8 vector registers : each register holds 64-elements
   - → 64 bits/vector elements (1VR = 512 bytes)
   - → Register file has 6 Read Ports and 8 write Ports.

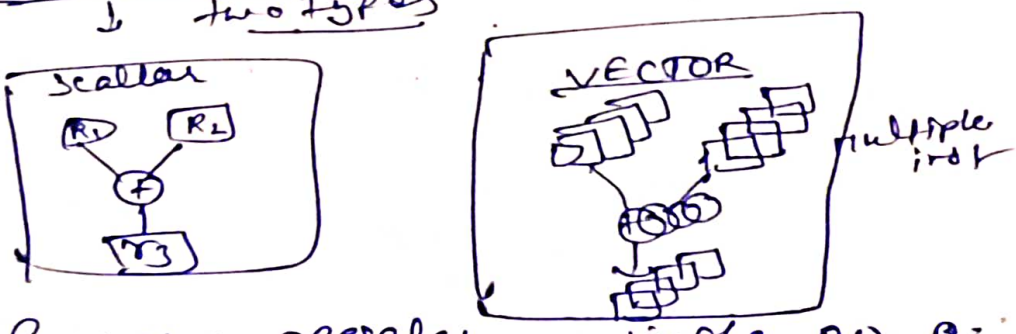
2) Scalar Registers:-
   32 GPRs ; 32 FPRs.

3) Vector functional units:-
   - → fully pipelined . can start new operation on every cycle → Capability for data and control hazards detection.

4) Vector load and ~~installation~~ store unit :-

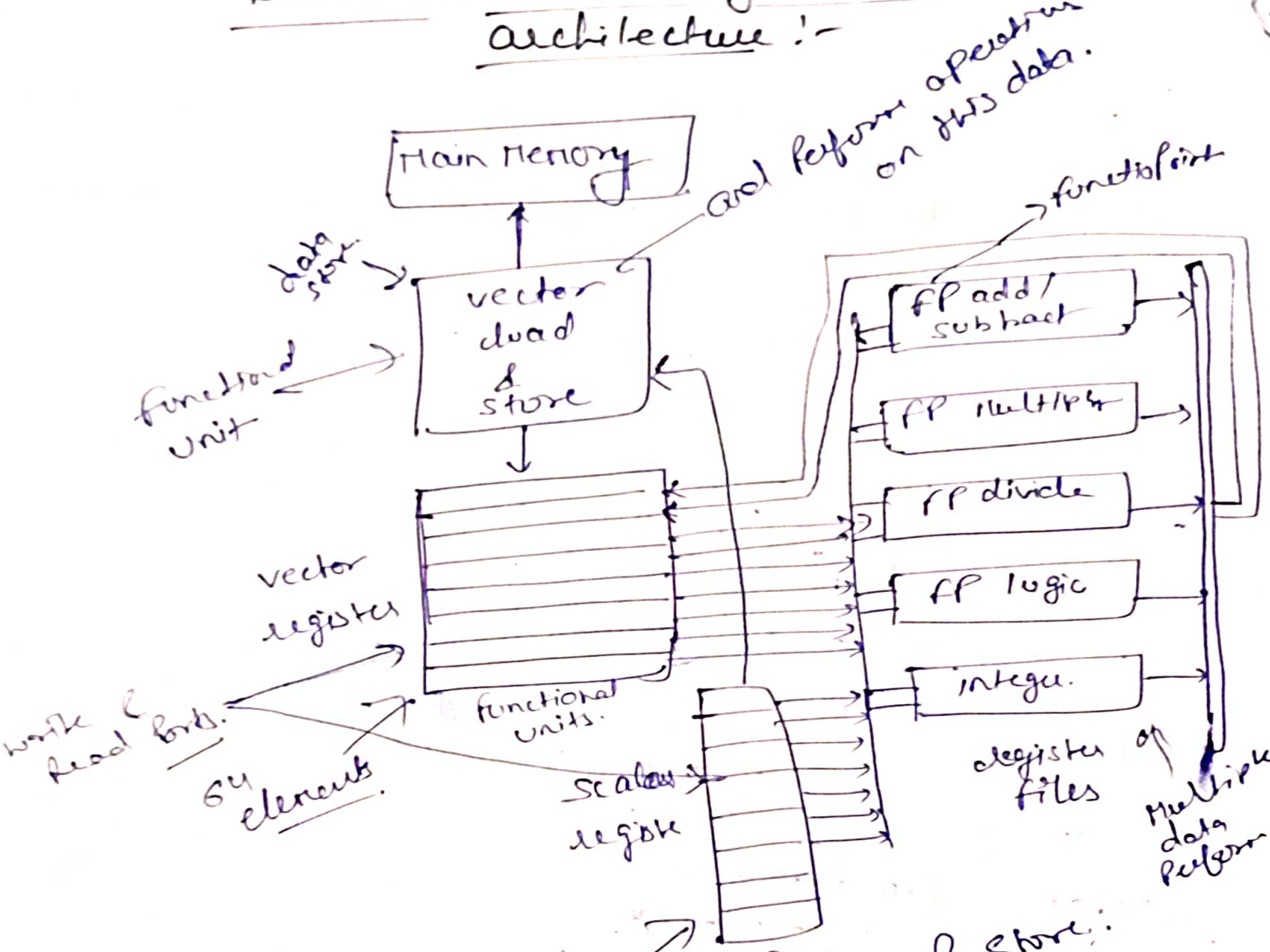
* Data Parallelism :- Parallelism arises from executing essentially the same code on the large no of object.

Vector Processors
↓ two types



→ Scalar Processor operates on single no ~~R~~.
→ Vector Processor operates on linear sequence of no.

Basic structure of Vector architecture :-



VMIP architecture

(1) Eight 64-elements vector registers.
(2) all the functional units are vector functional units.
(3) The vector and scalar registers have a significant no of read and write ports to allow multiple simultaneous vector operations.
(4) A set of crossbar switches (thick gray lines) connects these ports to the inputs and outputs of the vector functional units.

Vector Execution Time: depends on 3 factors.

1) Length of operand vectors
2) Structural hazards among the operations
3) Data dependencies.

Two forms are used for Execution Time:

1) ~~Convoy~~ Convoy: ~~set of vector instruction that could potentially execute together~~ -> A convoy of instruction must complete execution before any other instruction can begin execution.
2) chaining: Allows a vector operation to start as soon as the individual elements of its vector source operand is available.
3) chime: (unit of time) to execute on convoy.

* SIMD Processor also called data level or array processor

Single instruction multiple data.

We use SIMD → matrix-oriented scientific computing.

→ media-oriented image and sound processors.

→ SIMD is more energy efficient than MIMD because only needs to fetch one instruction per data operations processing multiple data elements.

→ makes SIMD attractive for personal mobile devices.

SIMD Computers ?

SIMD computers single inst stream and multiple data stream.

These computers are array processors. They are multiple processing elements which are supervised under same control unit. Each processing elements receives same instruction but operate on diff data from distinct module.

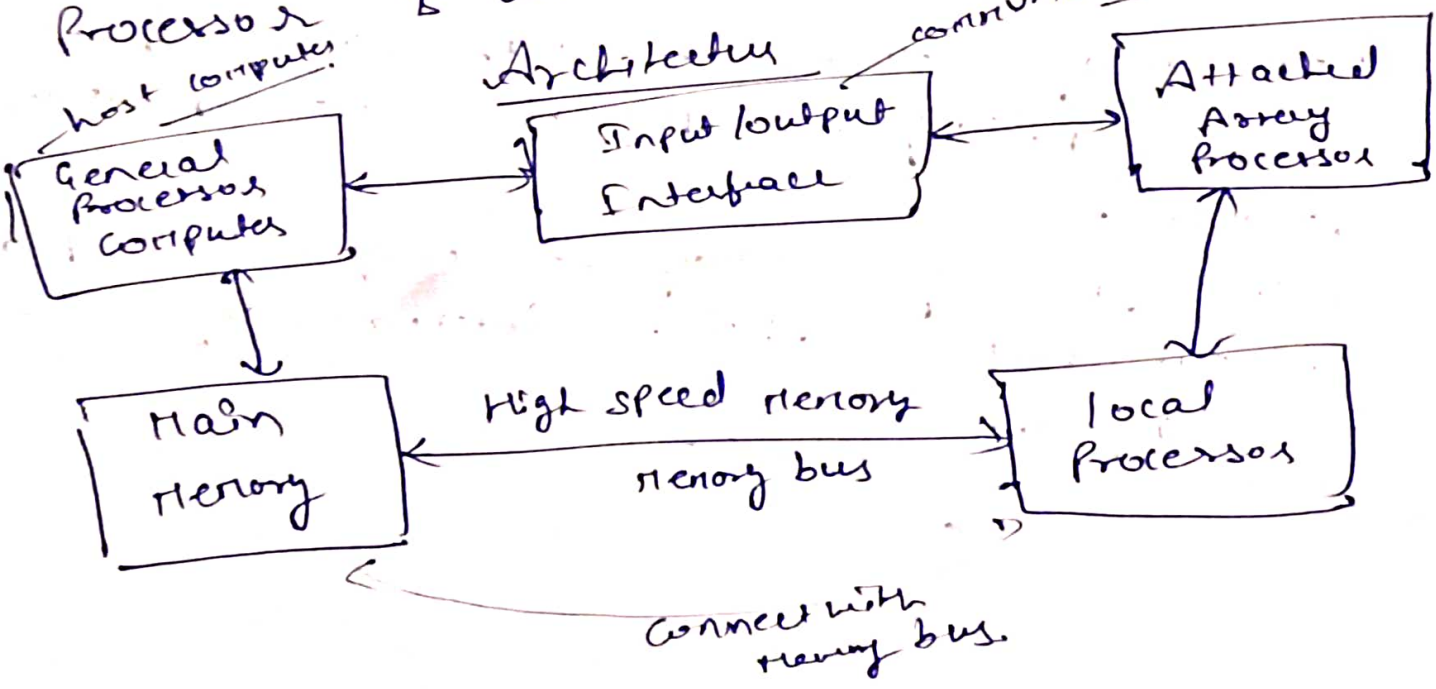
SIMD is called Array Processors (17)
It Perform computations on large array of data.

Array Processors

- Attached array Processor
- SIMD array Processors

1) Attached array Processors :- To improve the Performance of the host Computing (main computer) in specific numerical Computation tasks auxiliary Processor (attached with host computer) is attached to it.

Architecture



Attached computer has 2 interface.

1) Input/output interface to a communication processor
2) Interface with local memory.

Host computer is General Purpose Computer
Attached purpose is back-end machine driven by the host computer.

The array processor is connected through an I/O controller is the computer & the computer present -
it is an external interface.

SIMD array processor : There is computer with multiple processor unit operations parallel.
Both types of array processors, manipulates vector but their internal organization is different.