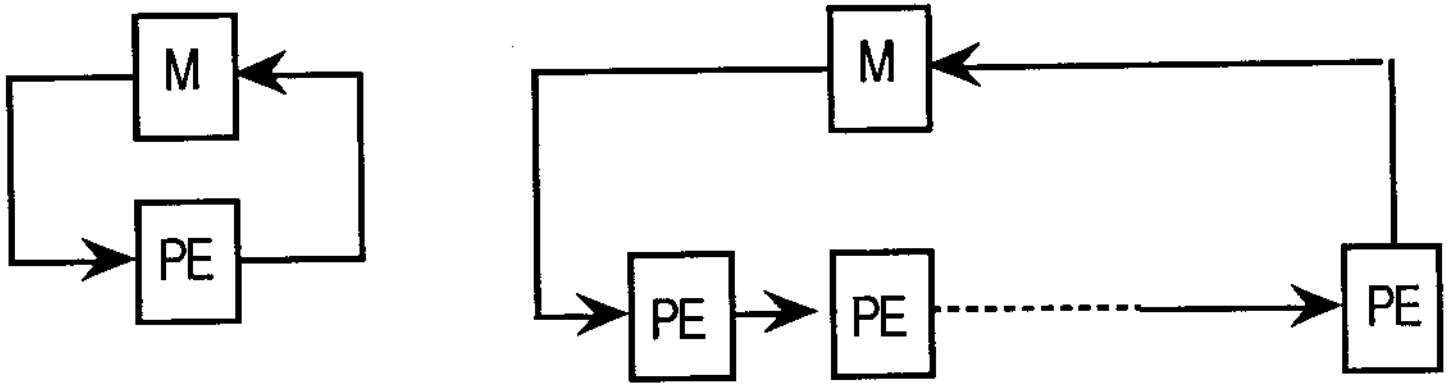
Systolic Architectures

- Basic principle: By replacing a single PE by a regular array of PEs, and by carefully orchestrating the flow of data between the PEs, high throughput can be achieved without increasing memory bandwdith requirements.



- Distinguishing features from regular pipelined computers:
  - Array structure can be non linear (e.g, hexagonal)
  - Pathways between PEs may be multidirectional
  - PEs may have local instruction and data memory, and are more complex than the stage of a pipelined computer.

Example: Convolution

- Important in its own right... plus used in filtering, pattern-matching, correlation, polynomial evaluation

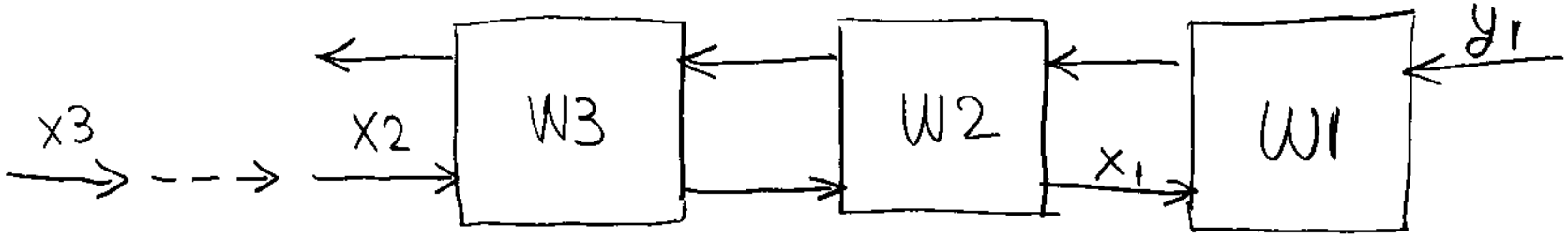
Given: sequence of weights $\{w_1\ w_2 \cdots w_K\}$

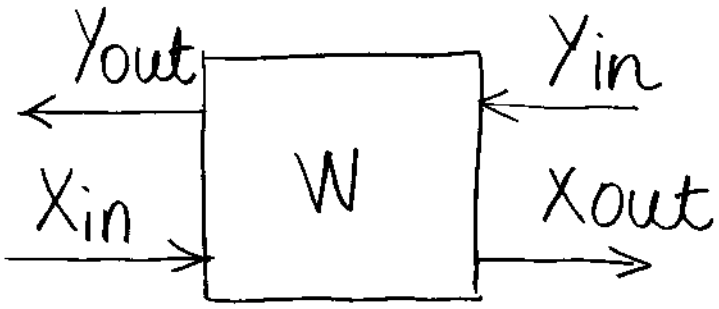
input sequence $\{x_1\ x_2 \cdots x_n\}$

Compute: $\{y_1, y_2 \cdots y_n\}$ where

$$y_i = w_1 x_i + w_2 x_{i+1} + \cdots w_K x_{i+K-1}$$

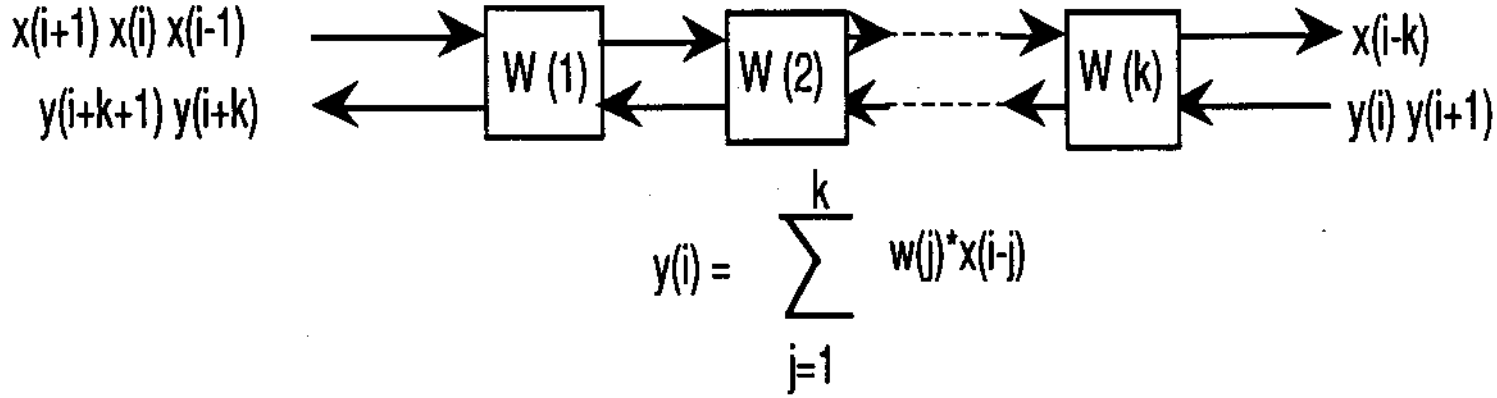
$$y_1 = w_1 x_1 + w_2 x_2 + w_3 x_3$$
$$y_2 = w_1 x_2 + w_2 x_3 + w_3 x_4$$
$$y_3 = w_1 x_3 + w_2 x_4 + w_3 x_5$$





$$Y_{out} \leftarrow Y_{in} + W * X_{in}$$
$$X_{out} \leftarrow X_{in}$$

- **Example: Systolic array for 1-D convolution**



- **Issues:**
  - **System integration: Shipping data from host array and back**
  - **Cell architecture and communication architecture (Warp, iWarp)**
  - **Software for automatically mapping computations to systolic arrays**
  - **General purpose systolic arrays**

Systolic Arrays

- H.T. Kung CMU 1978
- Specialized set of tasks only
- Key architectural issues in special-purpose systems
  1. Simple and regular
     - → Keep #unique parts small
     - → use VLSI
     - → interconnect simple and regular ways
  2. Exploit concurrency to be competitive

3. Balancing computation w/ I/O

→ since usually attached to host → comm. w/ host may be bottleneck

→ Balanced I/O: if $\frac{I/O \text{ words/sec}}{C \text{ MIPS}}$

the machine is balanced if the amount of computation required per word of I/O is: $C/I/O$

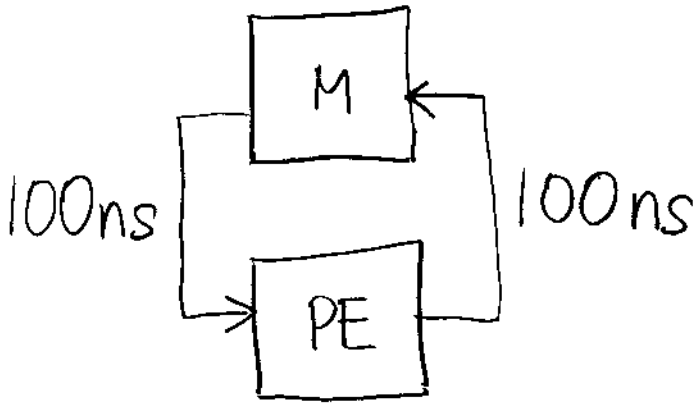
→ 2 kinds of pbms

- I/O bound: e.g. matrix addition — I/O ops $= N^2 + N^2$, C ops $= N^2$
- Compute bound: matrix mpy — I/O ops $= O(N^2)$, C ops $= O(N^3)$

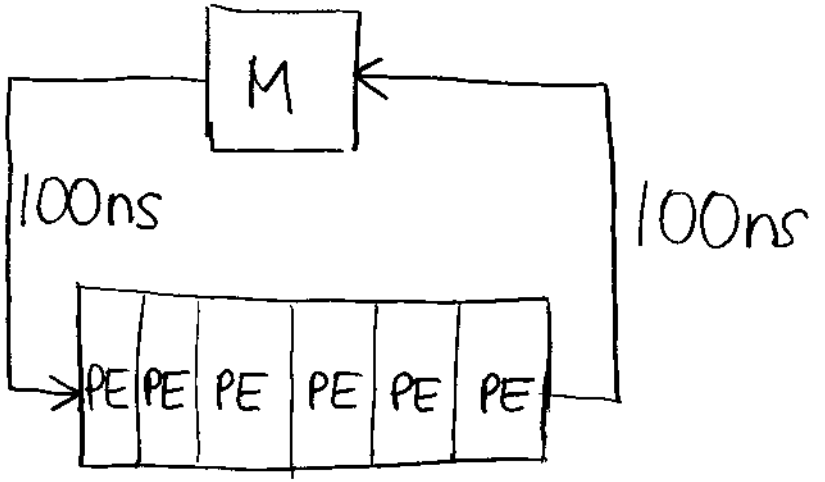
SYSTOLIC ARCH.

Central Theme

- Increase execution speed of compute-bound pbms by performing multiple computations per mem. accn



1 op / 200ns ⇒ 50 MOPS



6 ops / 200 ns ⇒ 300 MOPS

- Item stored inside PEs ⇒ I/O pbm related to
  1) I/O Bwdth 2) available mem internal to PE

Conventional Pipelines

- Usually linear
- Only partial results flow
- flow only in 1 direction

Systolic Arrays

- Not necessarily so: Grid, Hex, Triangular
- Both inputs & partial results flow
- flow in multiple directions possibly at diff speeds

Wavefront Arrays

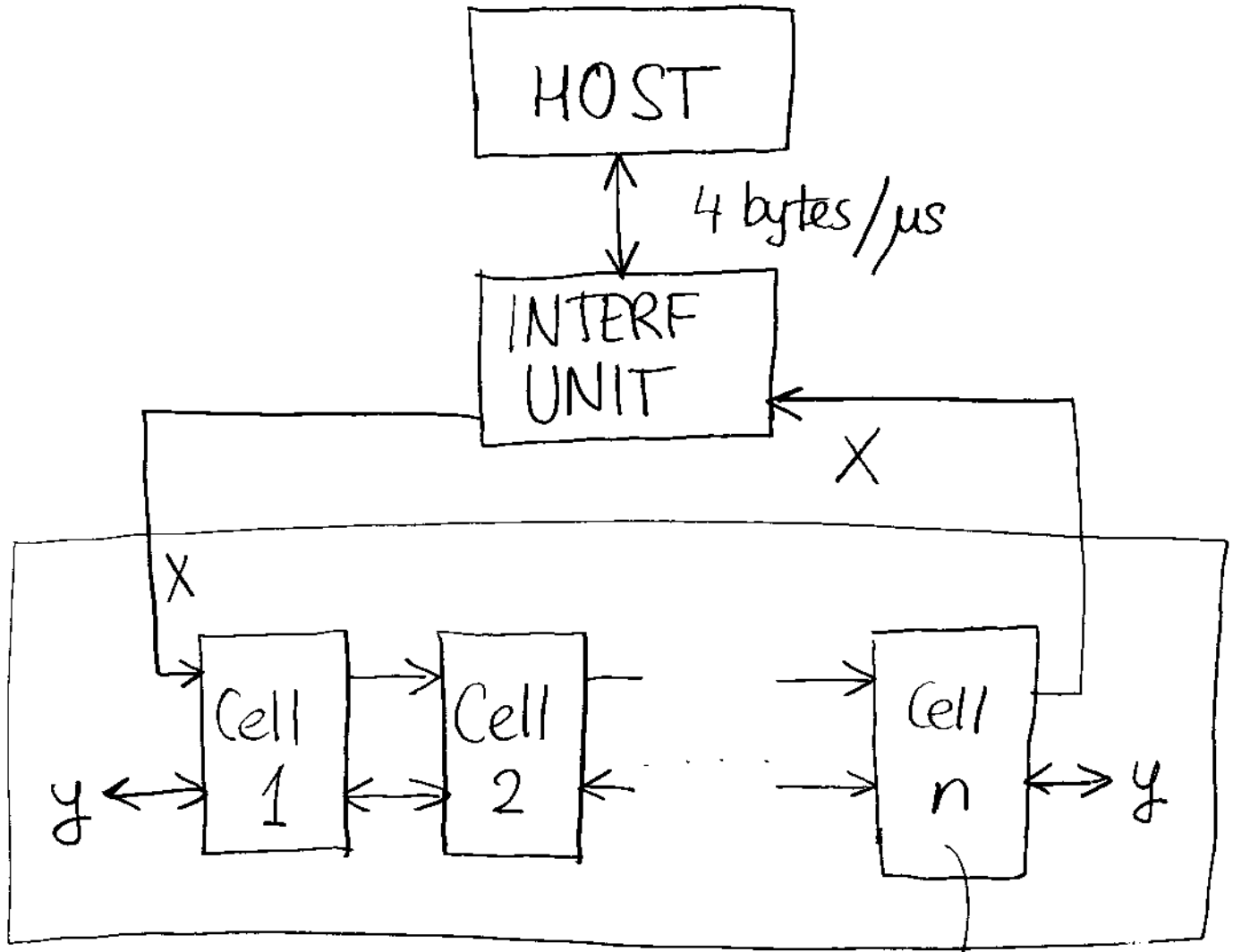
- handshaking protocol

Systolic Arrays

- syndronous in nature

The WARP Computer

- CMU, HTKung 1984-88
- Linear array of 10 cells; each is a 10 MFlop programmable processor
- Attached to host, a general UNIX machine
- High-level language W2 and optim. compiler
- Used extensively for vision and robotics tasks



Warp

8K I (272b instruction each)
32K D
512-deep queues

Why linear topology

1. Easy to use (lots of algorithms)
2. Cells are programmable → can simulate others
3. Easy to implement in HW: (Clock lines regular)
4. Demands smallest I/O BWD from host

Flow control of queues is implemented in HW

- Read from empty queue blocks
- Write to full queue blocks

Models of Parallelism Explored

- Input partitioning: input data partitioned among cells
- Output partitioning: each cell provides a portion of output, although it uses large part of input
- Pipelining: algorithm is partitioned so that each cell does one stage of processing

WARP Perf: Max 100 MFlops
Avg 30 MFlops $\simeq$ 60÷500 x faster than VAX 11/780 w/ FPU