Classification of Parallel Architecture Designs

Level of Parallelism

- Job level
  - between jobs
  - between phases of a job
- Program level
  - between parts of a program
  - within do-loops
  - between different function invocations
- Instruct stream level
- Instruction level
  - between phases of instruction execution
  - between instructions
- Arithmetic and bit level
  - within ALU units

Job Level

- **Different job phases**

  exp: CPU activities - I/O activities the overlapping may be achieved by programmer visible scheduling of resources

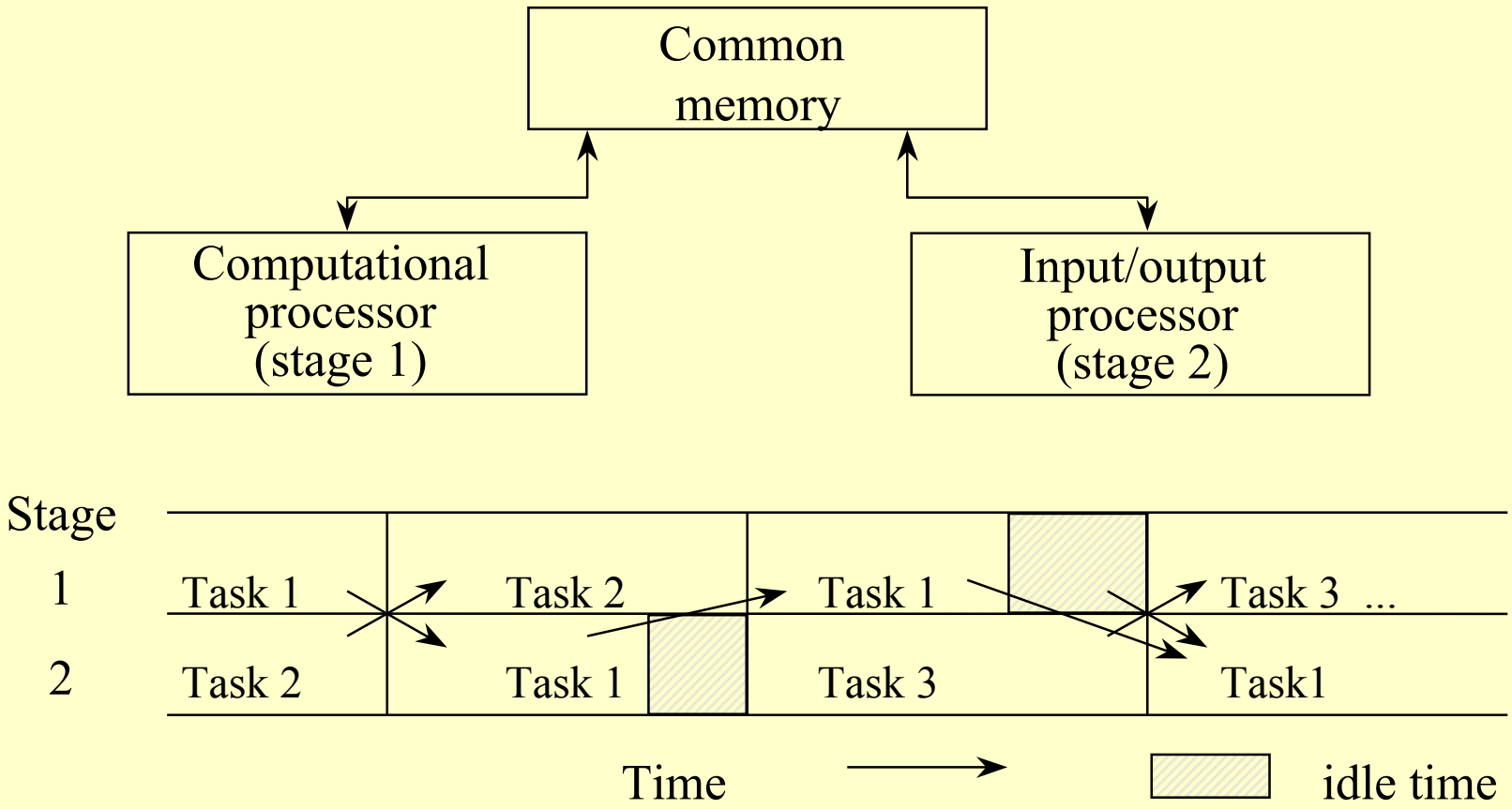
- **Different jobs**

  OS

	T1	T2
CPU	job 1	job 2
I/O	job 2	job 1

- **Architecture requirement**:

  a balanced set of replicated resources in a computer installation.



CPU/I/O Overlapping

Program Level Parallelism

- Different code sections:
  - diff. procedure/functions
  - diff. code blocks
- Different iterations for the same loop
- Data-dependencies and program partitioning

Instruction Level Parallelism (ILP)

- Between instructions
  - parallel execution of different instructions - spatial
  - key: dependency between instructions
- Between phases of instructions
  - overlapping different suboperations - pipelining
  - pipelining of a suboperations itself, e.g. ALU pipelining

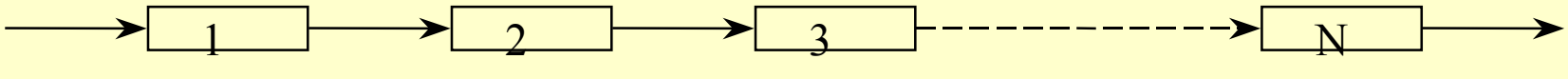
Overlay vs. Pipeline

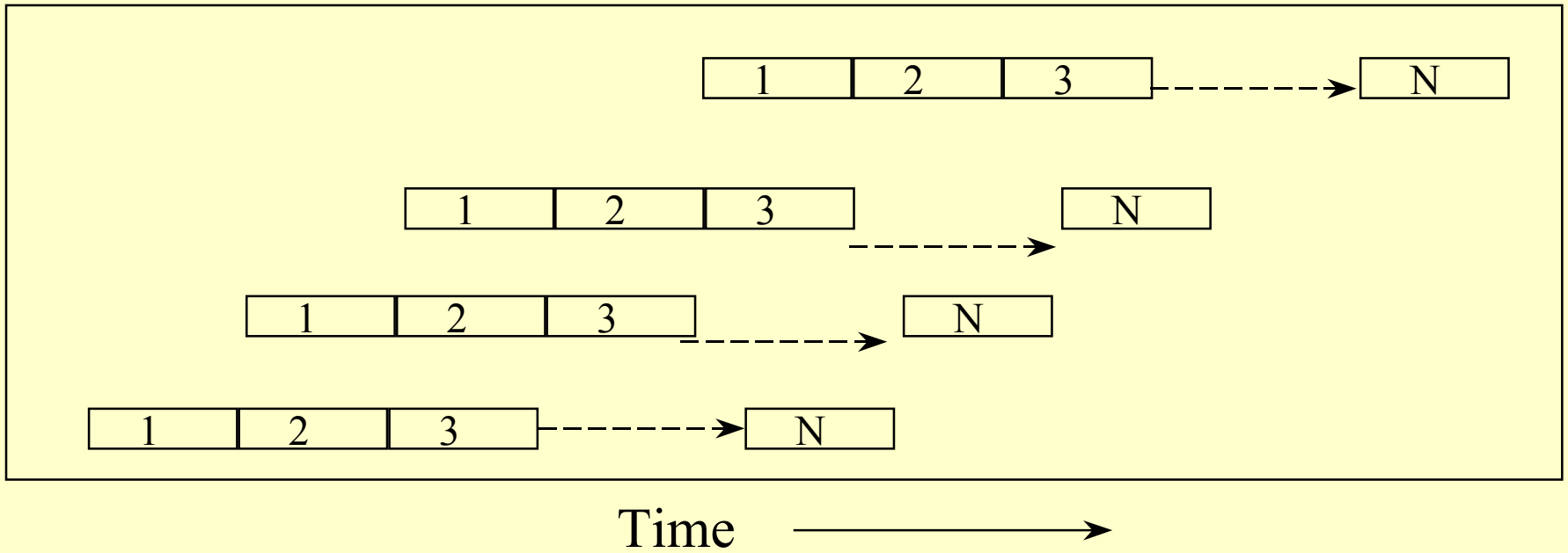
Pipeline:

- tightly coupled subfunctions
- fixed basic stage time
- independent basic function evaluation

Overlap

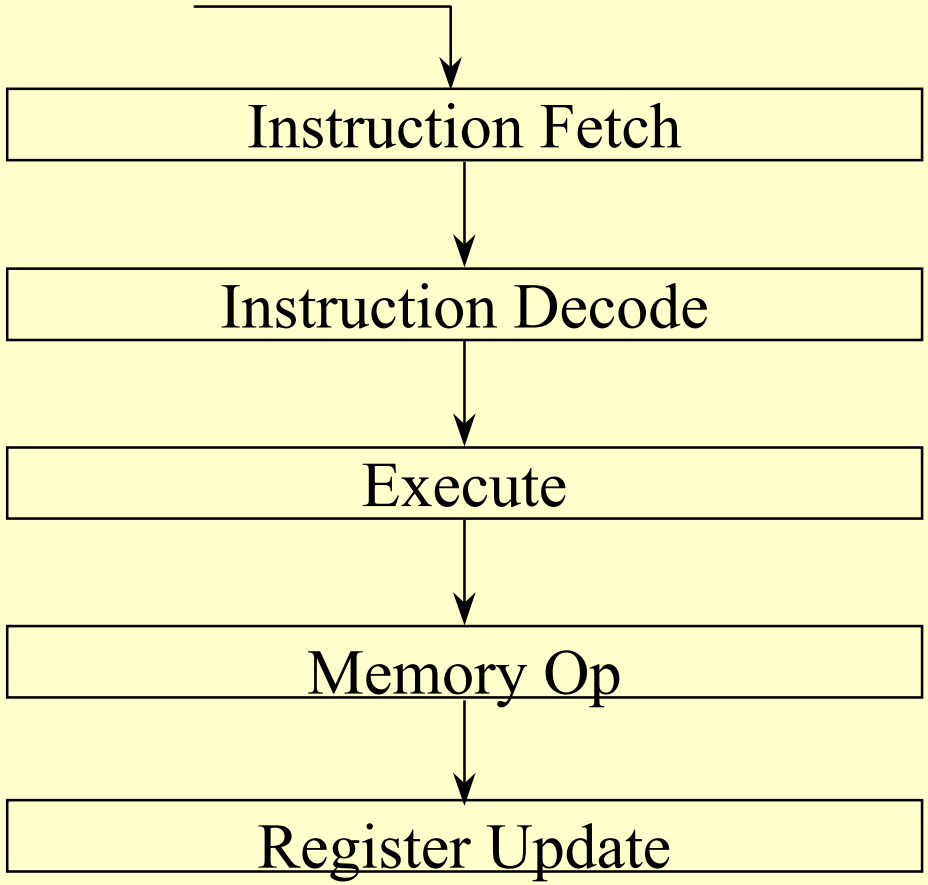
- loosely coupled subfunctions
- variable basic stage time
- dependency between function evaluation



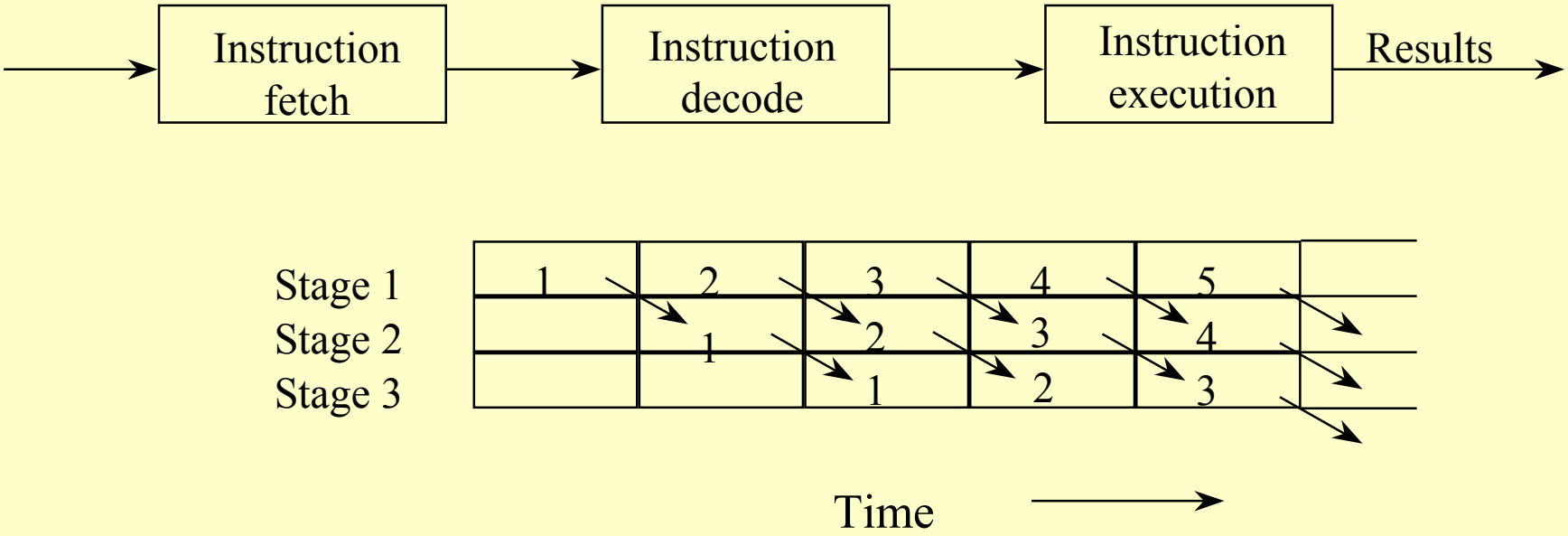


Hardware design method: RT

Principles of Pipelining



Pipelining of an Instruction Execution



number *i* corresponds
to instruction *i*

Hazards

- Any conditions within the pipelined system that disrupt, delay or prevent smooth flow of tasks through the pipelines.
- The detection and resolution of hazards constitute a major aspect of pipeline design
- Types of hazards

Flynn(72)

- Classification of parallel architecture is not based on the structure of the machine, but based on how the machine relates its instructions (streams) to the data (stream) being processed.

  A stream:
  - a sequence of items { instruction / data
  - being executed or operated on by a processor.

S I S D + I L P

S I M D + Vector

M I S D

M I M D

I L P gains increasing attention!

S I S D

- for practical purpose: only one processor is useful
- pipelining may or may not be used,

often called as serial scalar computer

SIMD

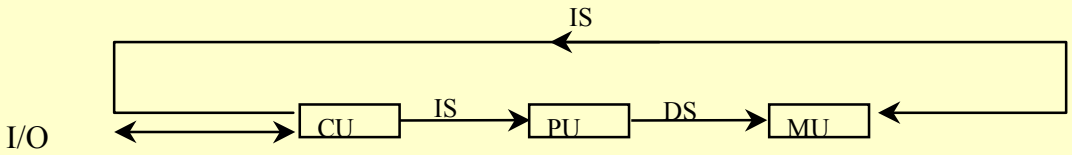
- (single inst stream/multiple data stream)
- single processor
- vector operations

  one v-op includes many ops on a data stream
- both pipelined processing or array of processors are possible
- Example:

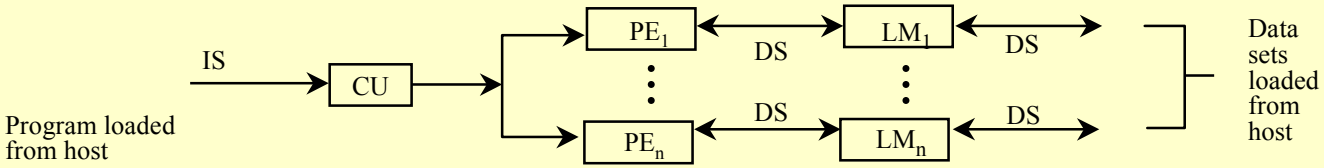
  CRAY -1

  ILLIAC-IV

  ICL DAP

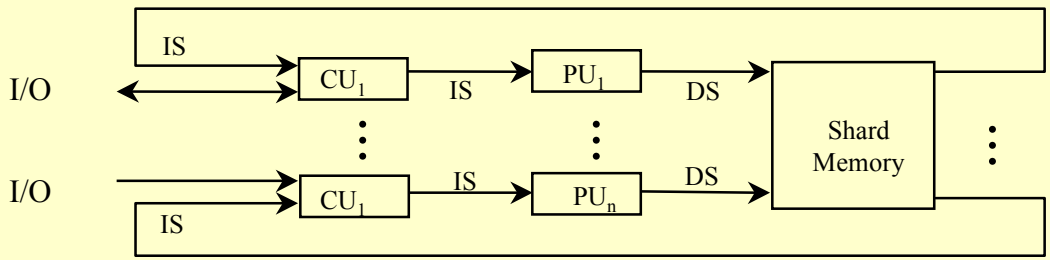


(a) SISD uniprocessor architecture



(b) SIMD architecture (with distributed memory)

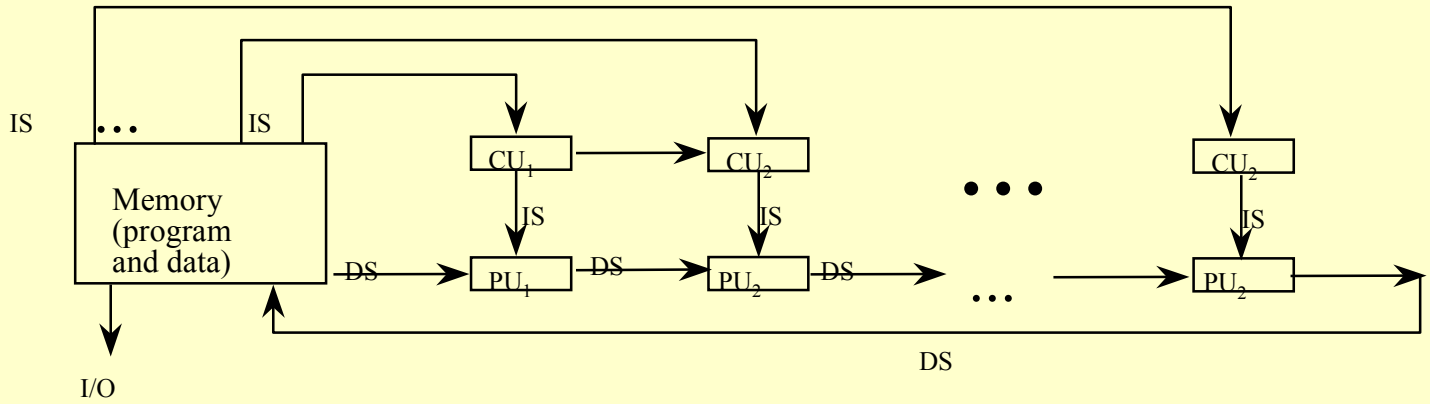
Captions:
C = Control Unit
PU = Processing Unit
MU= Memory Unit
IS = Instructin Stream
DS = Data Stream
PE = Processing Element
LM = Local Memory



(c) MIMD architecture (with shared memory)

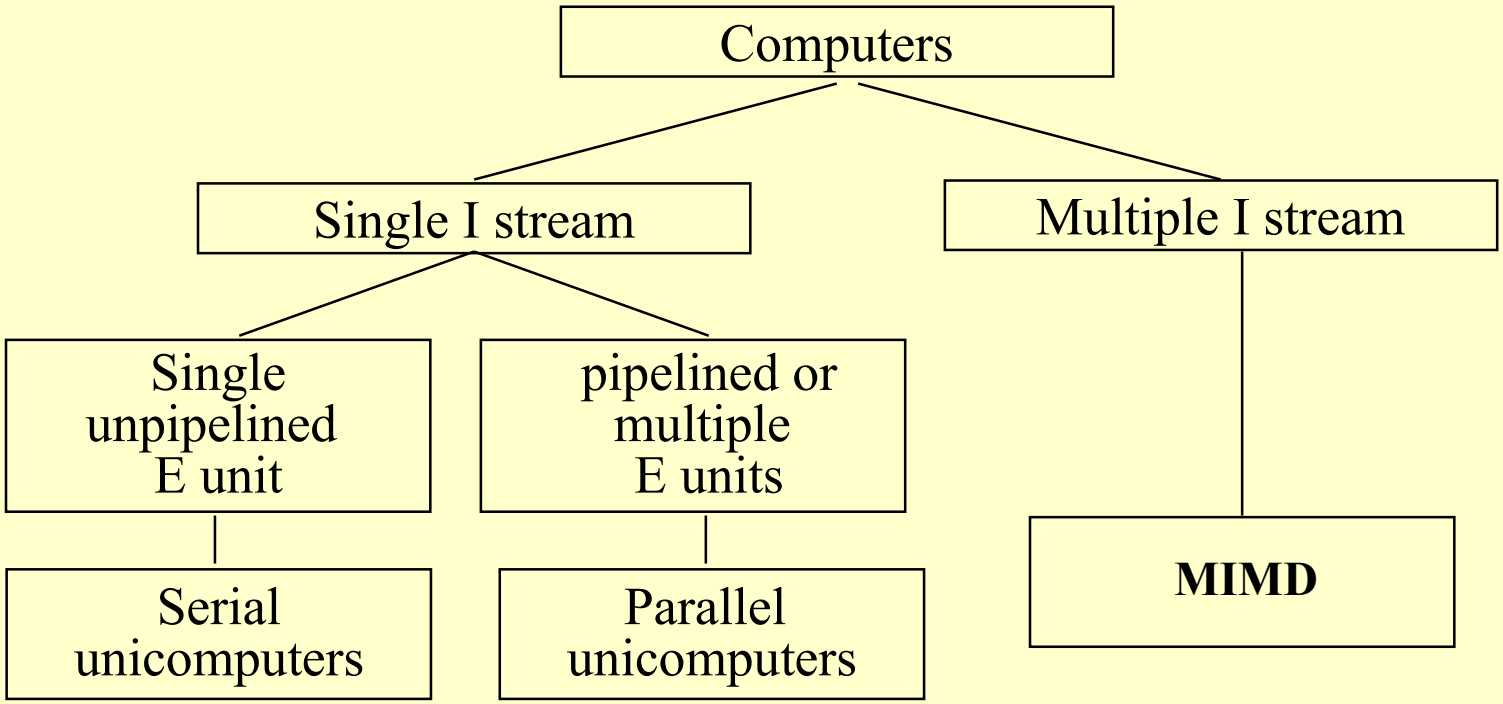
Problems of Flynn's Scheme

- too broad
- everything in SIMD: vector machines?
- MISD?
- Relation with Parallel Programming models ?

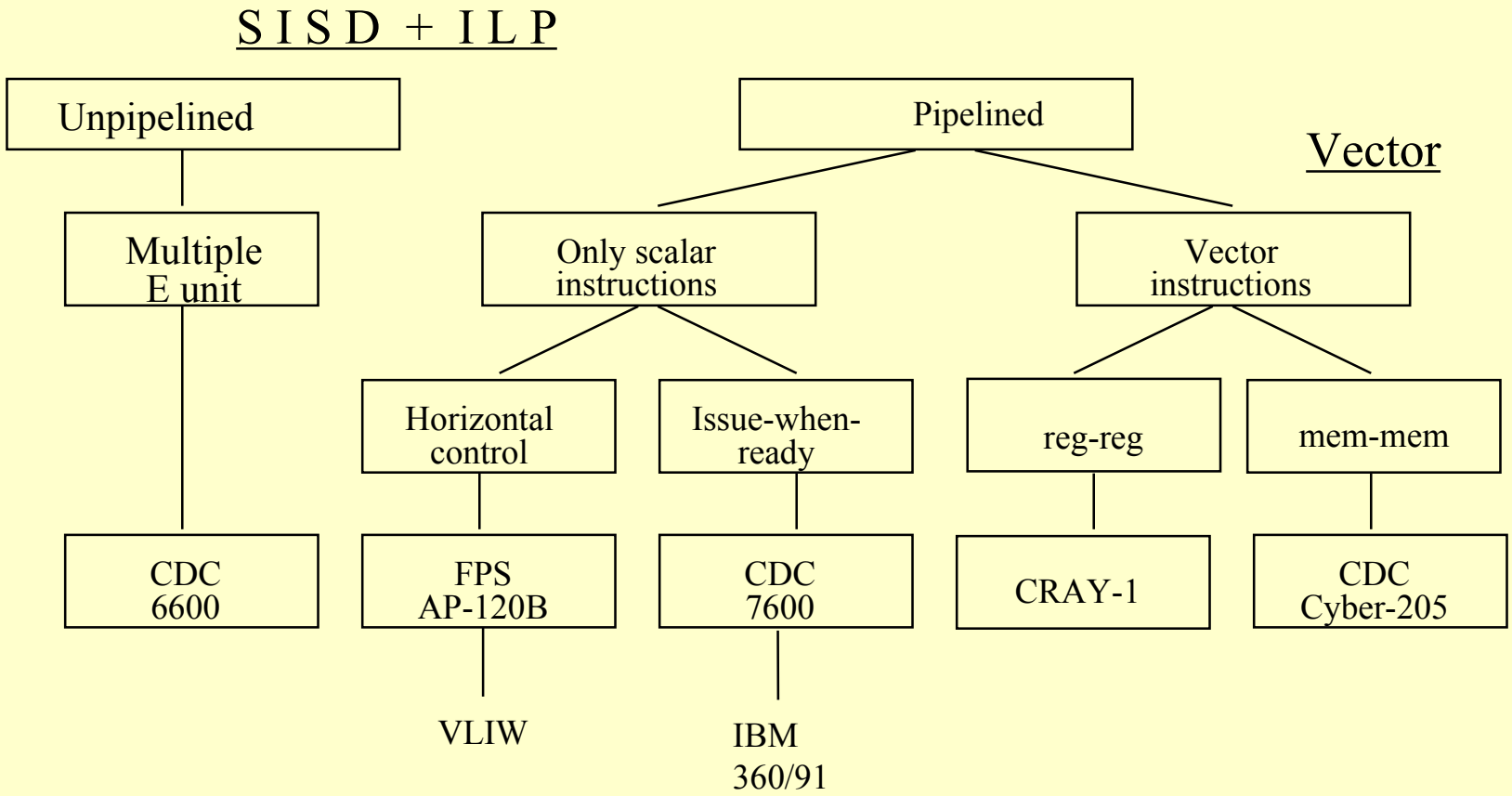


MISD architecture (the systolic array)

Captions:
CU= Control Unit
PU = Processing Unit
MU + Memory Unit
IS = Instructin Stream
DS = Data Stream
PE = Processing Element
LM = Local Memory



The broad subdivisions in computer architecture



Parallel unicomputers based on functional parallelism and pipelining

ILP Architectures

- Multiple inst /op issuing + deep-pipelining
- Superscalar
  - Multiple inst/cycle (Power PC,MIPS10000, Intel Pentium)
- VLIW
  - Multiple op in one inst/cycle
  - ESL/polycyclic or Cydra5
  - Multiflow
  - Intel iA-64 architecture ?
- Superpipelined
  - MIPS 4000/8000
  - DEC Alpha (earlier versions)
- Decoupled Arch
- Multithreaded Arch
  - EARTH/MTA
  - Multiscalar [Sohi]
  - etc.

Sources

Guang R. Gao