Topic #9

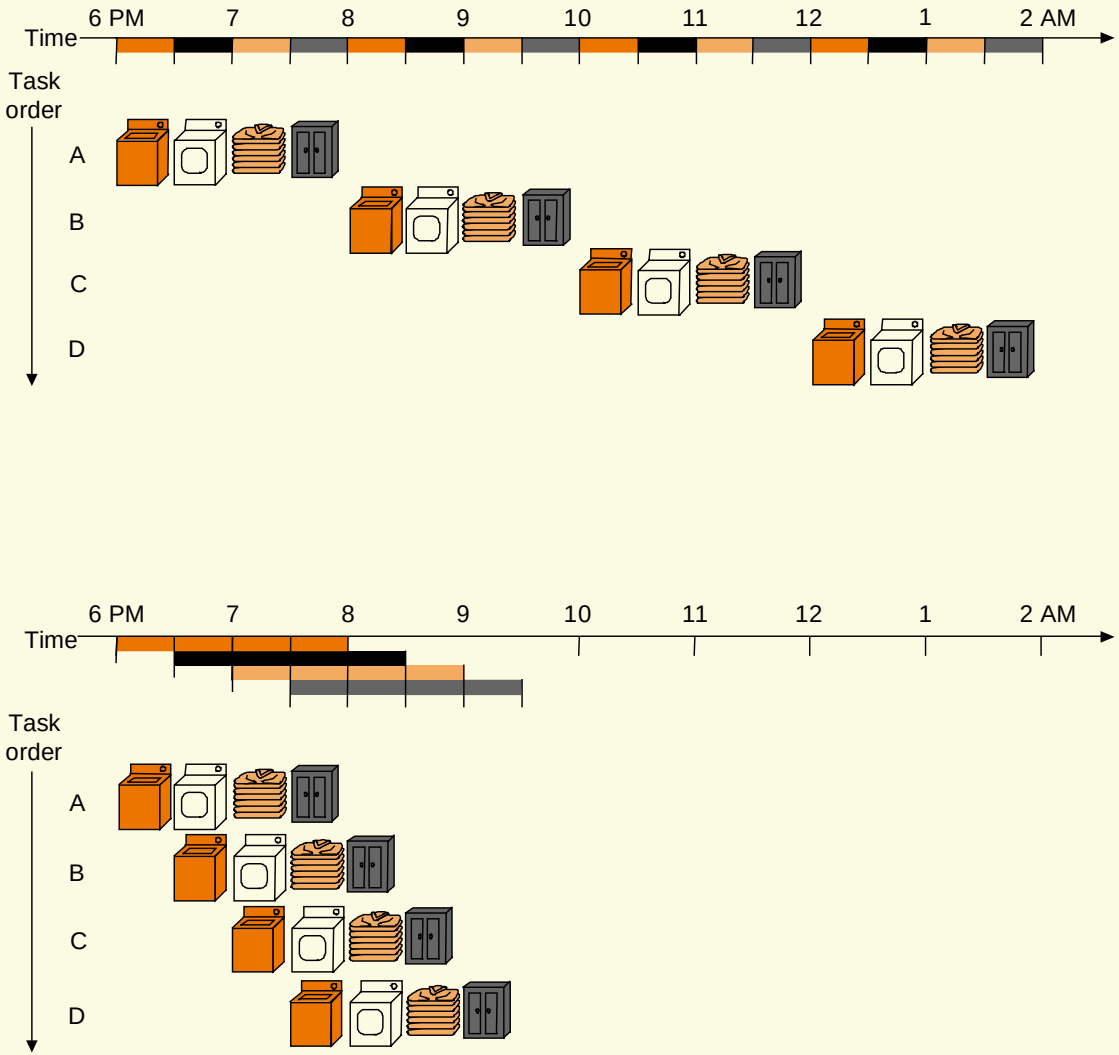
Introduction to Pipelining and Multiprocessing

High-Performance Processors

- Two techniques for designing high-performance processors:
  - **Pipelining**
  - **Multiprocessing**

- Both techniques exploit **parallelism**:
  - Pipelining: parallelism among multiple **instructions**
  - Multiprocessing: parallelism among multiple **processors**

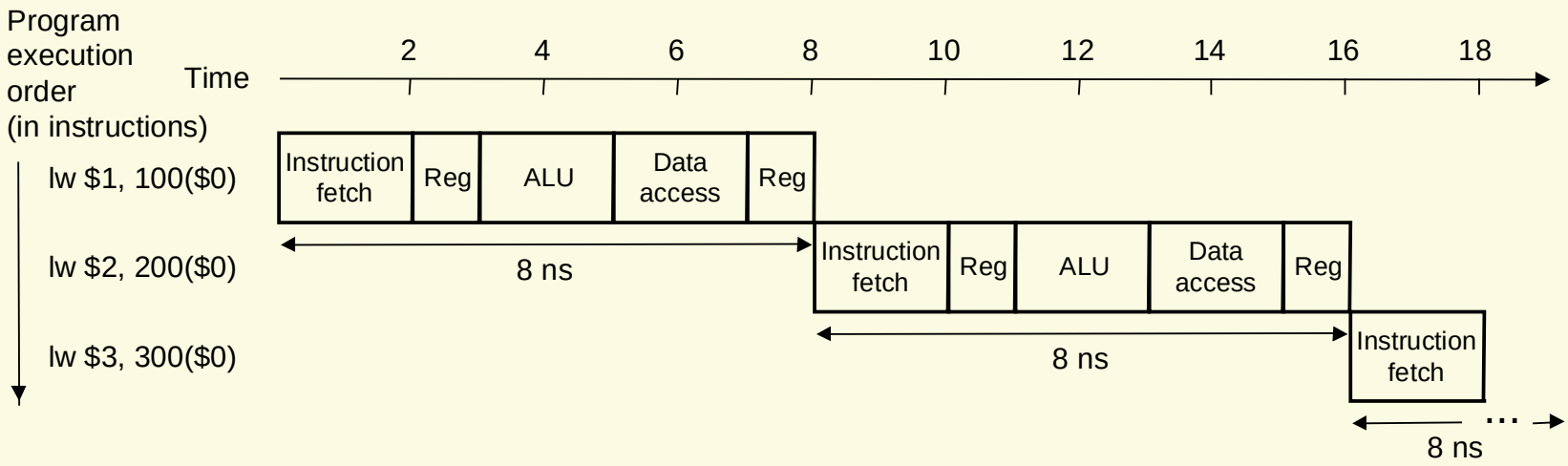
Laundry Analogy for Pipelining

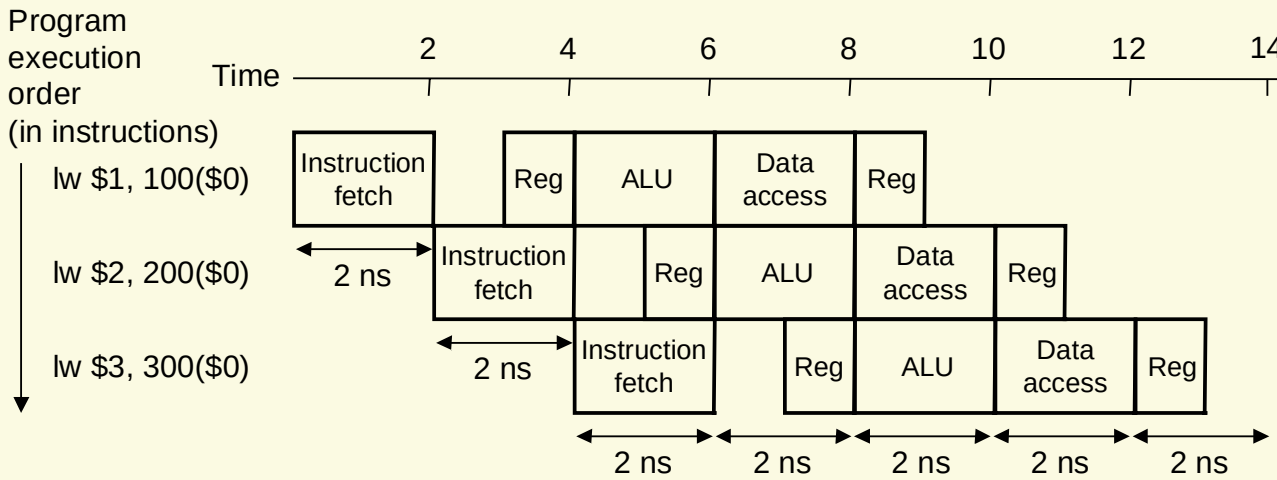


Pipelining Principles

- Pipelining does not help the latency of a single task, but it helps the **throughput** of the entire workload.
- Multiple tasks are processed simultaneously.
- The pipeline rate is limited by the slowest pipeline stage.
- **Potential speedup = number of pipeline stages**
- Unbalanced lengths of pipeline stages can reduce speedup.

Pipelined Instruction Execution





Favorable Characteristics of MIPS Instructions for Pipelining

- **All MIPS instructions are of the same length**.
  - It is easier to fetch instructions in the first pipeline stage and then decode them in the second stage.

- **MIPS has only a few instruction formats**, with the source register fields being located in the same place in each instruction.
  - The second stage can begin reading the register file when the hardware is determining the type of instruction just fetched.

Favorable Characteristics of MIPS Instructions for Pipelining

- **Memory operands only appear in load or store instructions in MIPS**.
  - We can use the execution stage to calculate the memory address and then access memory in the following stage.

- **Operands must be aligned in memory**.
  - We need not worry about a single transfer instruction requiring two data memory accesses; the requested data can be transferred between processor and memory in a single pipeline stage.

Pipeline Hazards

- **Hazards** are situations in pipelining when the next instruction cannot be executed in the following clock cycle.

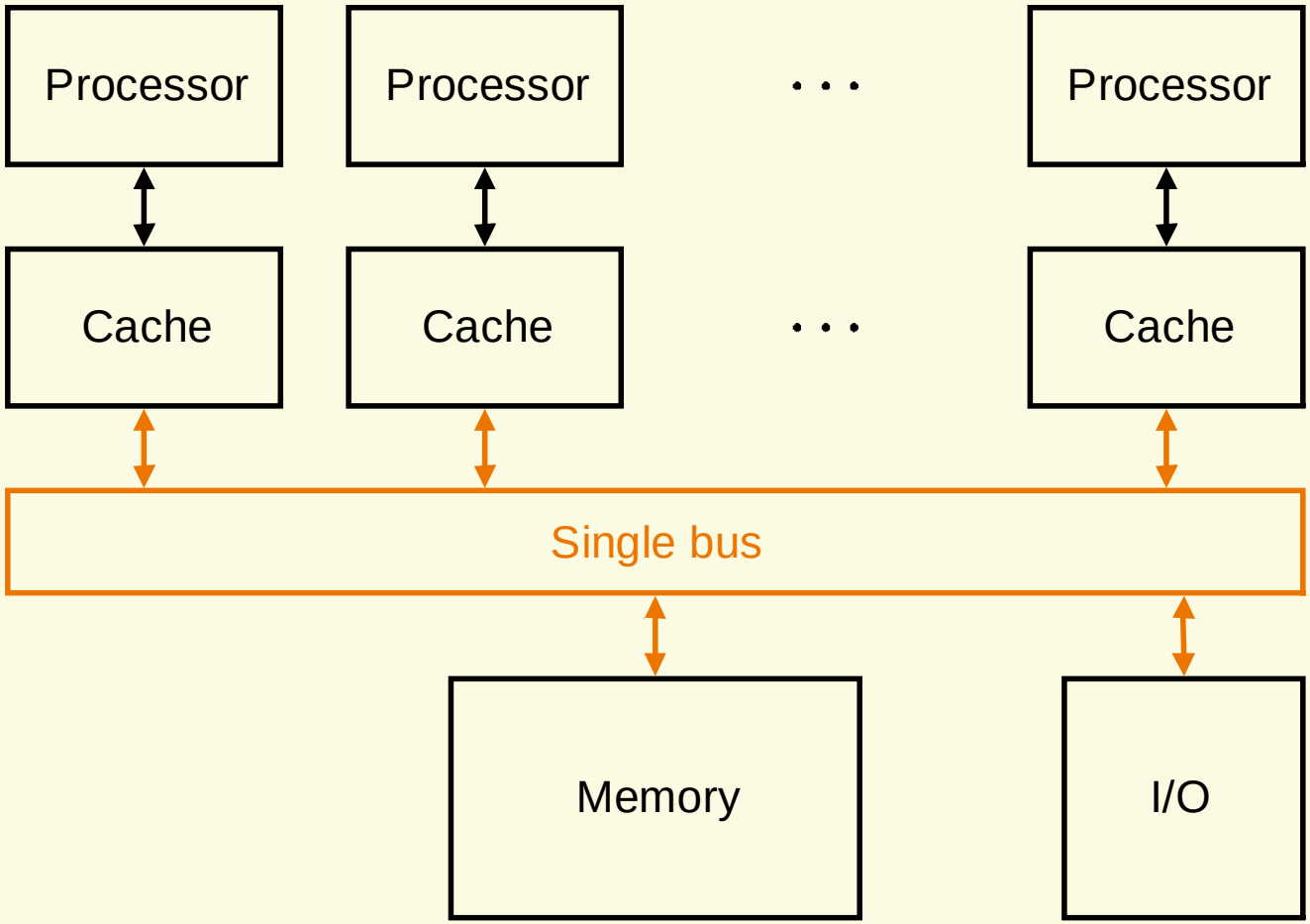
- Hazards reduce the performance of pipelining.

- **Types of pipeline hazards**:
  - **Structural hazards**: Hardware cannot support the combination of instructions to execute in the same clock cycle.
  - **Control hazards**: Which instruction to execute next depends on the results of a previous instruction still in the pipeline.
  - **Data hazards**: An instruction depends on the results of a previous instruction still in the pipeline.

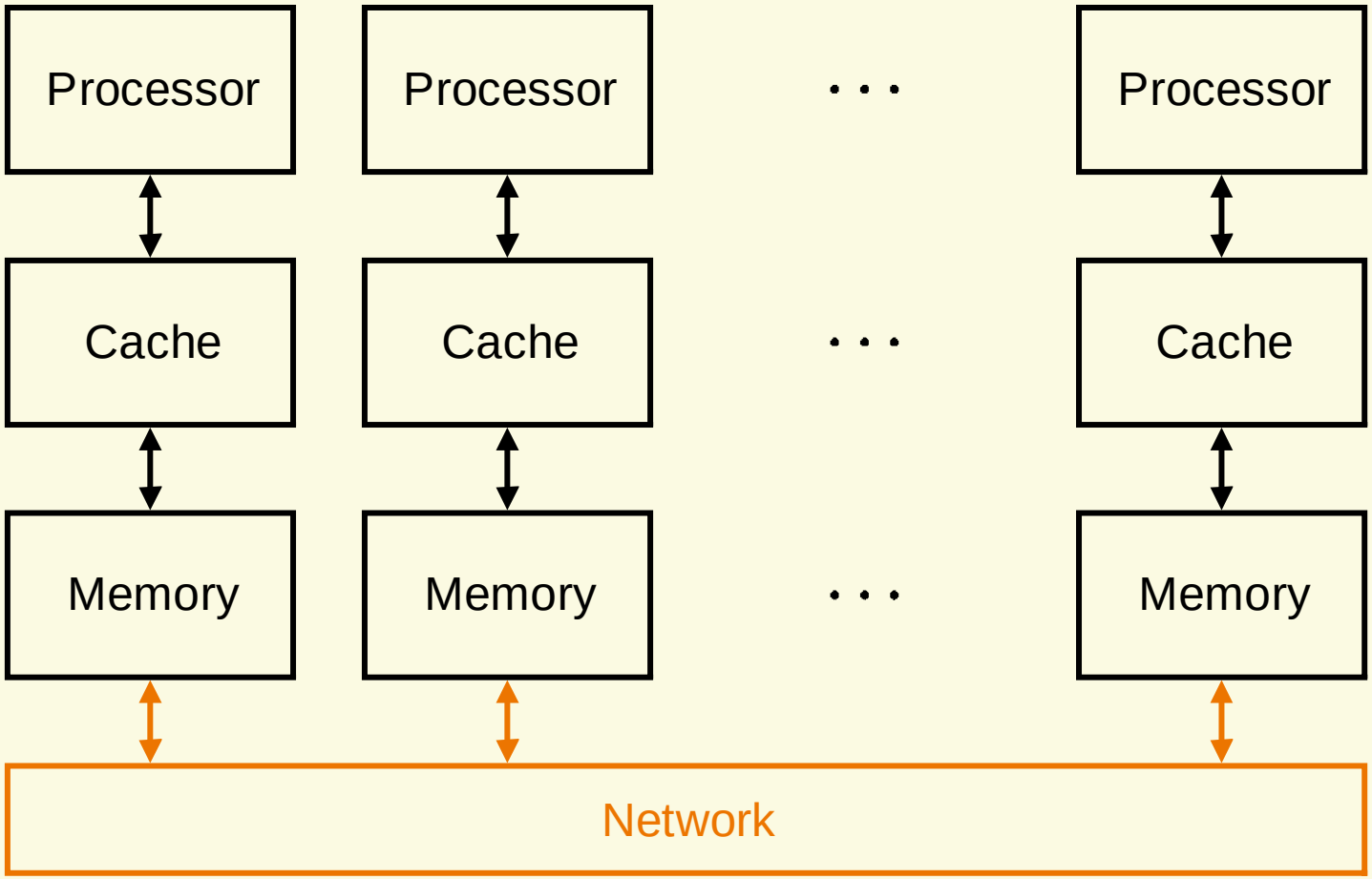
Multiprocessing

- It would be nice if we could connect uniprocessors together to create a scalable and powerful multiprocessor.
- Parallel processors can share data using either a **single address space** (**shared-memory processors**) or **message passing** (**clusters of computers**).
- Parallel processors can be connected using either a **single shared bus** (generally with less processors) or a **network** (generally with more processors).
- It is difficult to write efficient programs for multiprocessors.

Single-Bus Multiprocessor



Network-Connected Multiprocessor



Network Topologies

- Different network topologies affect the performance and cost of communications between connected components.

- A variety of network topologies exist:
  - E.g., **ring**, **grid**, **Omega network**