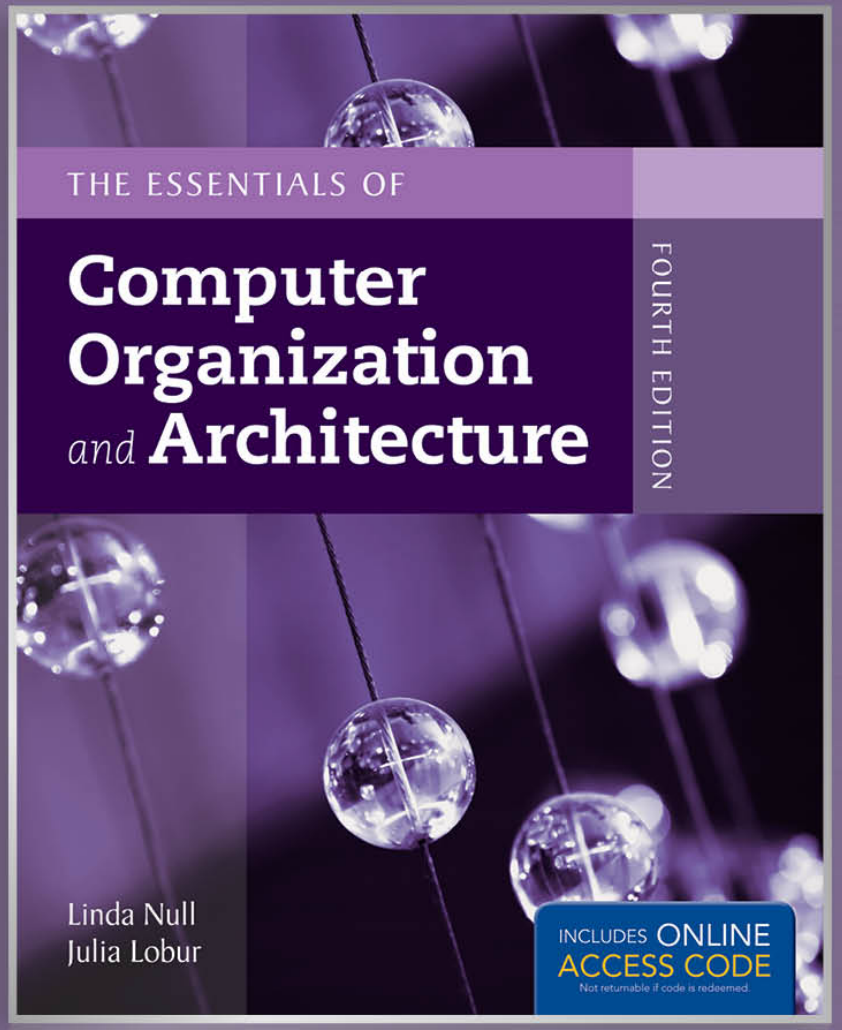
Chapter 9

Alternative Architectures



Chapter 9 Objectives

- Learn the properties that often **distinguish RISC** (Reduced Instruction Set Computer) from **CISC** (Complex Instruction Set Computer) **architectures**.
- Understand how **multiprocessor architectures** are **classified**.
- Appreciate the factors that create **complexity in multiprocessor systems**.

9.1 Introduction

- We have so far **studied** only the simplest models of computer systems; **classical single-processor von Neumann systems**.
- This chapter presents a number of **alternative approaches to computer organization and architecture**.
- Some of these **approaches are in place in today's commercial systems**. Others may form the basis for the computers of tomorrow.

9.2 RISC Machines

- The **philosophy of RISC** machines is that a system is **better able to manage program execution** when the program consists of **only a few different instructions** that are the **same length** and require the **same number of clock cycles** to execute.
- **RISC** systems **access memory only with** explicit **load and store** instructions.
- In **CISC** systems, **many different kinds of instructions access memory**, making **instruction length variable** and **execute time unpredictable**.

9.2 RISC Machines

- The **difference** between **CISC and RISC** becomes evident through the basic **computer performance equation**:

$$\text{CPU Time} = \frac{\text{seconds}}{\text{program}} = \frac{\text{instructions}}{\text{program}} \times \frac{\text{avg. cycles}}{\text{instruction}} \times \frac{\text{seconds}}{\text{cycle}}$$

- **RISC** systems **shorten execution** time by **reducing the clock cycles per instruction**.
- **CISC** systems **improve performance** by **reducing the number of instructions per program**.

9.2 RISC Machines

- The **simple instruction set** of **RISC** machines enables **control units to be hardwired** for maximum speed.
- The **more complex** -- and variable-- instruction set of **CISC** machines requires **microcode-based control** units that interpret instructions as they are fetched from memory. This **interpretation takes time**.
- With **fixed-length instructions**, **RISC** lends itself to **pipelining and speculative execution**.

9.2 RISC Machines

- Consider the program fragments:

CISC

```
mov ax, 10
mov bx, 5
mul bx, ax
```

RISC

```
        mov ax, 0
        mov bx, 10
        mov cx, 5
Begin   add ax, bx
        loop Begin
```

- The total **clock cycles** for the **CISC** version might be:

  `(2 movs × 1 cycle) + (1 mul × 30 cycles) = 32 cycles`

- While the **clock cycles** for the **RISC** version is:

  `(3 movs × 1 cycle) + (5 adds × 1 cycle) + (5 loops × 1 cycle) = 13 cycles`

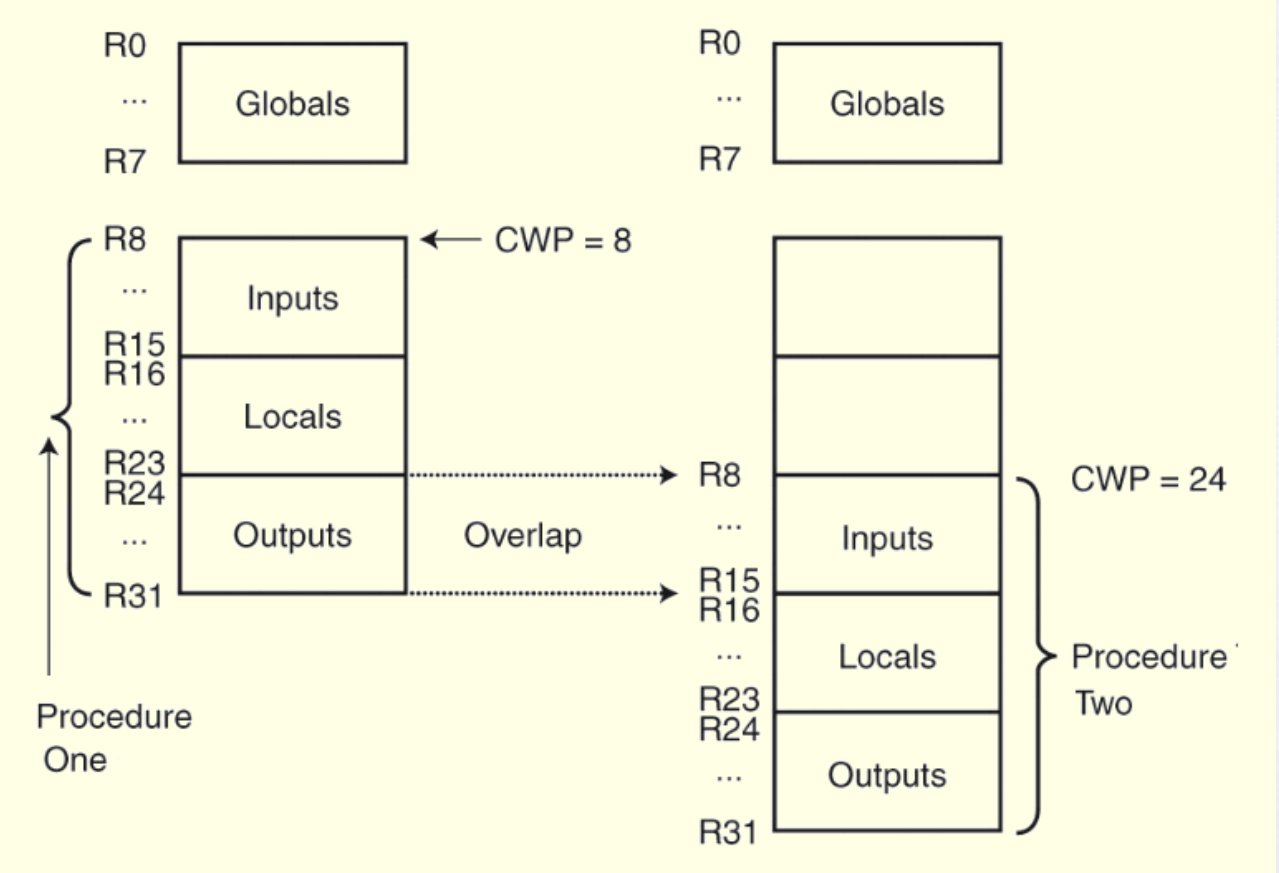
- With **RISC clock cycle** being **shorter**, RISC gives us much **faster execution speeds**.

9.2 RISC Machines

- Because of their **simpler instruction set**, **RISC** architectures **allow for a larger number of CPU registers**.
- These **registers provide fast access to data** during sequential program execution.
- The **registers** can also be employed to **reduce** the **overhead** typically caused **by passing parameters** to subprograms.
- **Instead** of pulling **parameters off of a stack**, the **subprogram** is directed to **use a subset of registers**.

9.2 RISC Machines

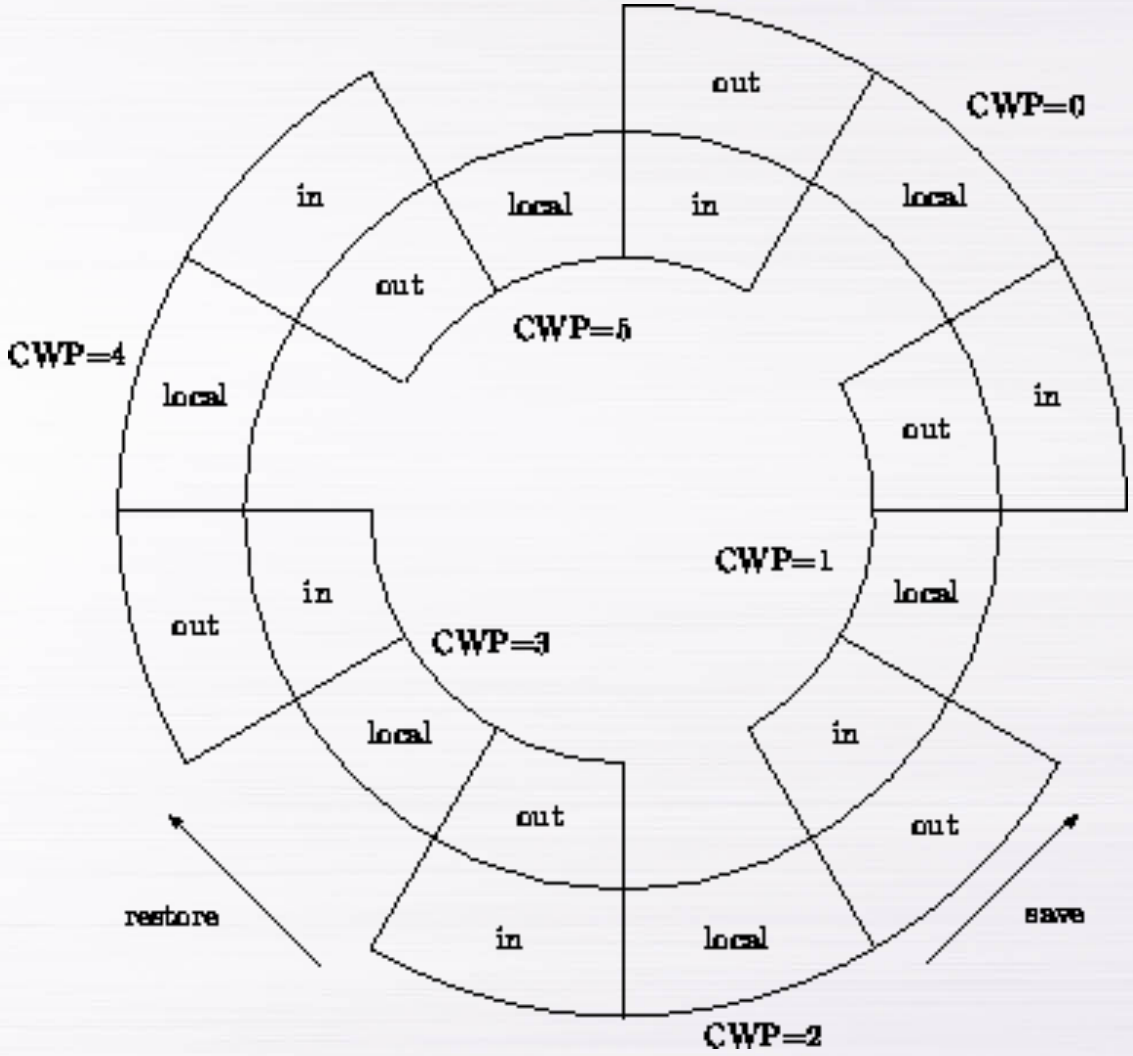
- **Registers can be overlapped in a RISC** system.
- The **current window pointer** (CWP) points to the **active register window**.



Procedure 1 and Procedure 2 share output/input registers for parameter passing

9.2 RISC Machines

- If **all register windows in use**, data of '**oldest**' procedures **saved to memory**
- **Registers** of 'most recent' procedures **wrap around**



9.2 RISC Machines

- In recent times, is becoming **difficult to distinguish RISC** architectures from **CISC** architectures.
- **Some RISC** systems have **more extravagant instruction sets than** some **CISC** systems.
- **Some systems combine both** approaches.
- The **following slides summarize** the **traditional differences** between these two architectures.

9.2 Quiz: RISC or CISC?

- Multiple register sets
- Complex instructions.
- Parameter passing through memory.
- Multiple cycle instructions.
- Hardwired control.
- Highly pipelined.
- Variable length instructions
- Many instructions can access memory.

9.2 RISC Machines

- **RISC**
  - Multiple register sets.
  - Three operands per instruction.
  - Parameter passing through register windows.
  - Single-cycle instructions.
  - Hardwired control.
  - Highly pipelined.
- **CISC**
  - Single register set.
  - One or two register operands per instruction.
  - Parameter passing through memory.
  - Multiple cycle instructions.
  - Microprogrammed control.
  - Less pipelined.

Continued....

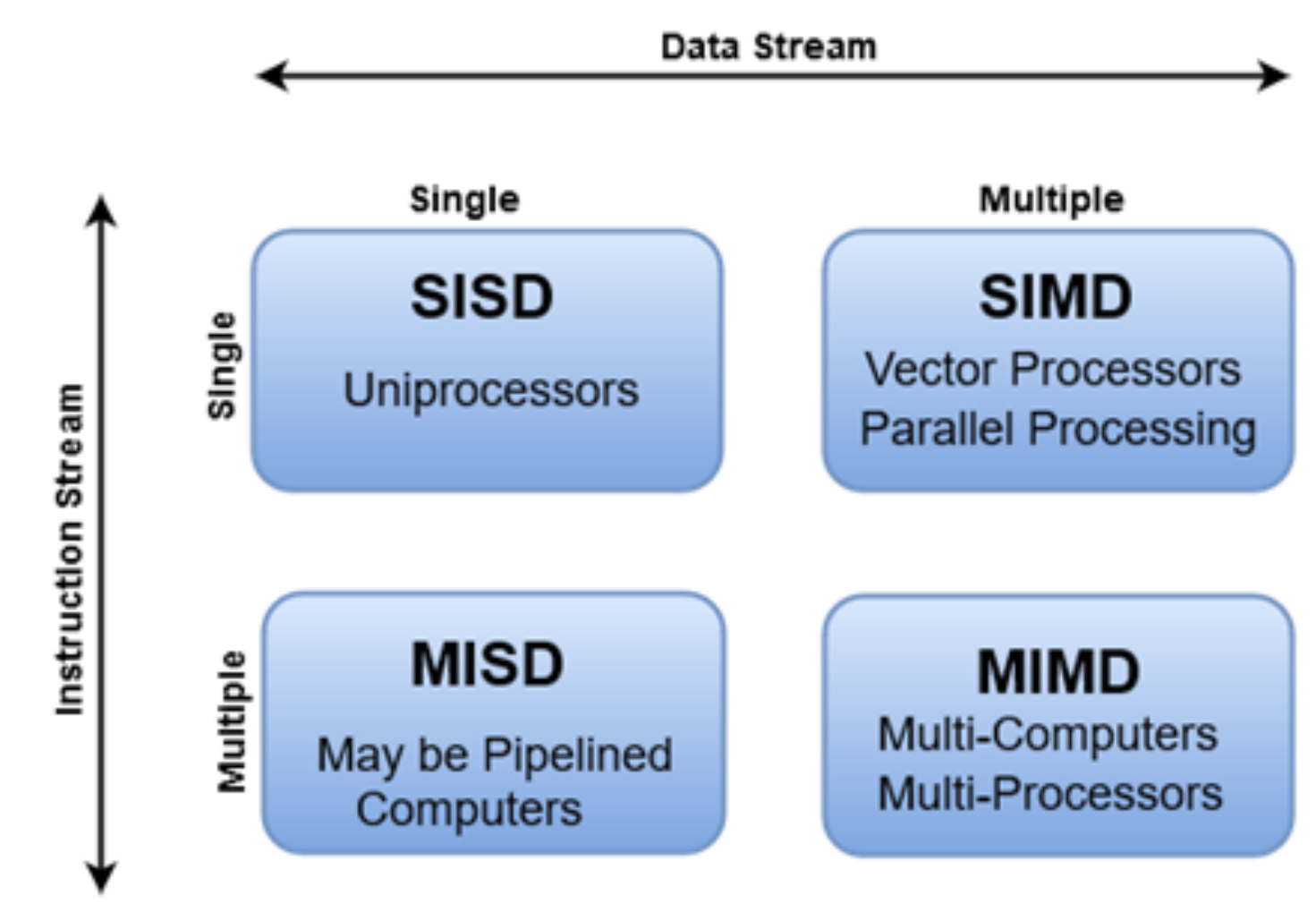
9.2 RISC Machines

- **RISC**
  - Simple instructions, few in number.
  - Fixed length instructions.
  - Complexity in compiler.
  - Only **`LOAD/STORE`** instructions access memory.
  - Few addressing modes.
- **CISC**
  - Many complex instructions.
  - Variable length instructions.
  - Complexity in microcode.
  - Many instructions can access memory.
  - Many addressing modes.

9.3 Flynn's Taxonomy

- Many attempts have been made to come up with a way to **categorize computer architectures**.
- ***Flynn's Taxonomy*** has been the **most common** of these, despite having some limitations.
- Flynn's Taxonomy takes into consideration the **number of processors** and the **number of data streams** incorporated into an architecture.
- **A machine can have one or many processors that operate on one or many data streams.**

9.3 Flynn's Taxonomy



9.3 Flynn's Taxonomy

- The **four combinations** of multiple processors and multiple data streams are described by Flynn as:
  - **SISD:** Single instruction stream, single data stream. These are **classic uniprocessor systems**.
  - **SIMD:** Single instruction stream, multiple data streams. Execute the same instruction on multiple data values, as in **vector processors**.
  - **MIMD:** Multiple instruction streams, multiple data streams. These are **today's parallel architectures**.
  - **MISD:** Multiple instruction streams, single data stream.

9.3 Flynn's Taxonomy

- **Flynn's Taxonomy falls short** in a number of ways:
- First, there appears to be no **need for MISD machines.**
- Second, it **ignores** the contribution of **specialized processors**.
- Third, it does not **distinguish different** architectures of the **MIMD** category.
  - One **idea** is to **divide MIMD systems into** those that **share memory**, and those that **don't**, as well as whether the **interconnections** are **bus-based** or **switch-based**.

9.3 Flynn's Taxonomy

- **Symmetric multiprocessors** (SMP) and **massively parallel processors** (MPP) are MIMDs that **differ in how they use memory**.
- SMP systems share the same memory and MPP do not:

SMP $\Rightarrow$ fewer processors + **shared memory** + **communication via memory**

MPP $\Rightarrow$ many processors + **distributed memory** + communication via **message passing**

9.3 Revised Taxonomy