Embedded Systems Architecture

Session #4

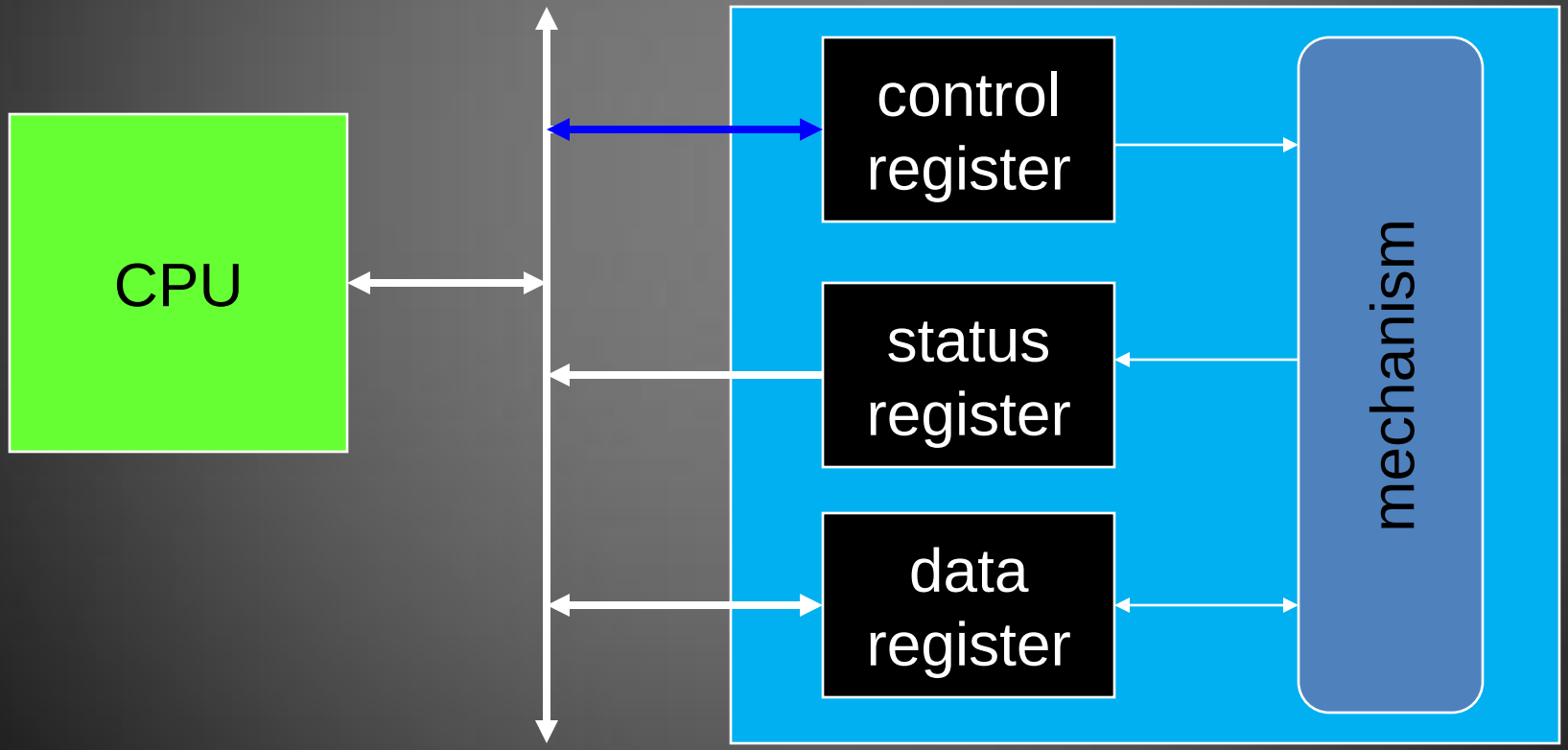
CPUs

- I/O
- Interrupts
- Co-Processors
- Caches & MMUs
- Performance
- Power Consumption

I/O

- I/O (Input/Output) is simply the interface between hardware & software.
- “One persons input is another’s output.”
- Consists of one or more of the following:
  - Control (or command) information
  - Status (or response) information
  - Data information
- Can be I/O space or memory space mapped.

I/O


CPU
control register
status register
data register
mechanism

I/O

- Hardware I/O devices include things like:
  - Switches
  - Analog to Digital Converters (ADC)
  - Digital to Analog Converters (DAC)
  - Universal Synchronous Asynchronous Receiver Transmitters (USART)
  - Interval Timers
  - Watch Dog Timers

Interrupts

- Generic Definition
- Types
- Basics
- Priorities
- Latency
- Vectors
- Overhead

Generic Interrupt Definition

- An interrupt is any event that:
  - Causes the current program code execution to be interrupted.
  - Interrupt specific code is executed.
  - Interrupted program code execution is resumed.

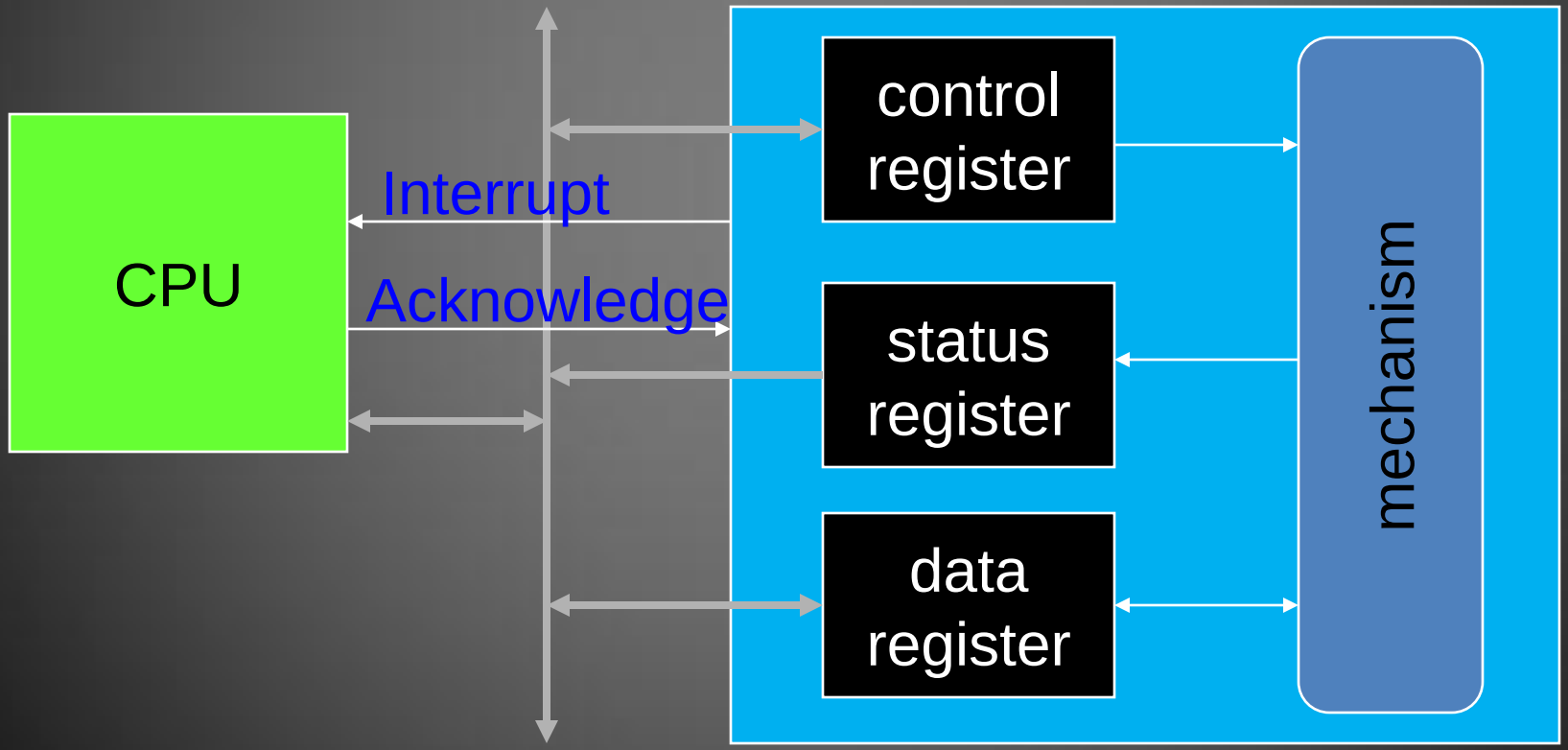
Interrupt Types

- Hardware Interrupts
- Software Interrupts

Hardware Interrupts

- Used to service I/O more efficiently than polling (Busy/Wait).
- Servicing a hardware interrupt could
  - Get data from hardware.
  - Give data to hardware.
  - Setup software and / or hardware for next interrupt.

Hardware Interrupts



Software Interrupts

- Traps
  - ICE (In Circuit Emulators)
  - OS Calls (mode switches)
- Exceptions
  - Divide by zero
  - Illegal instruction
  - Data access error (misalignment or non-existent memory)

Interrupt Basics

- Request
- Save context
- Acknowledge
- Service
- Restore context

Interrupt Priorities

- The CPU must have a mechanism to resolve simultaneously occurring hardware interrupts.
- What happens if a higher priority interrupt occurs *after* servicing has already begun of another interrupt?
- Is there a mechanism present to solve this problem?

Interrupt Latency

- After a hardware interrupt occurs, how long does it take for the CPU to service it?
- Does that time vary?
- How long does it take to service each interrupt?

Interrupt Vectors

- Each interrupt has it’s own Interrupt Service Routine which is pointed to by the interrupt vector.
- The vector number may be fixed by the CPU or provided by the interrupting device.
- The interrupt vector may point to a pointer of the ISR or the ISR itself.

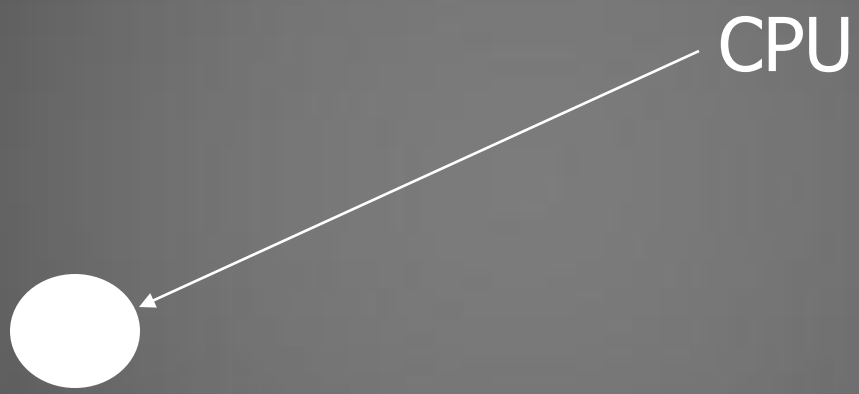
Interrupt Overhead

- Negatively impacts CPU performance by
  - *Stealing* CPU cycles.
  - Emptying execution pipeline.
  - Swapping out cached memory.

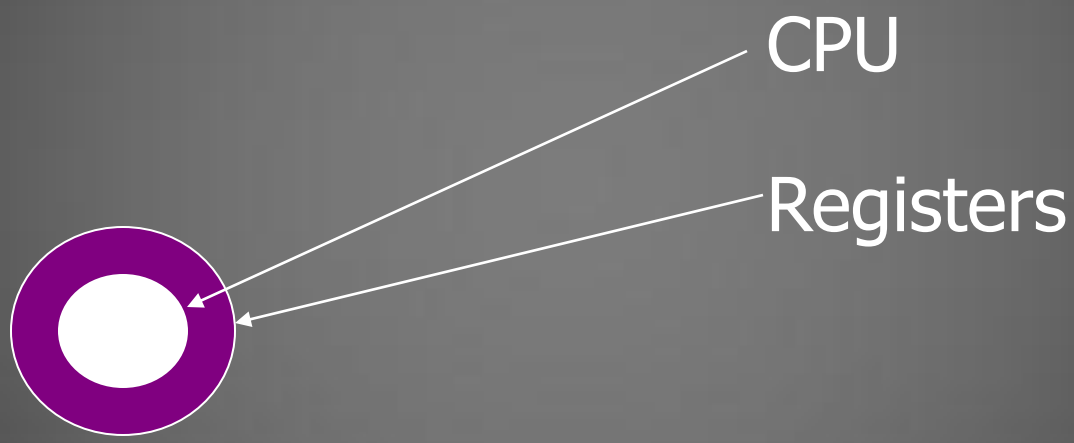
Co-Processors

- Provide additional functionality to the CPU such as floating point or I/O.
- CPU must reserve part of it's instruction set space for co-processor use.

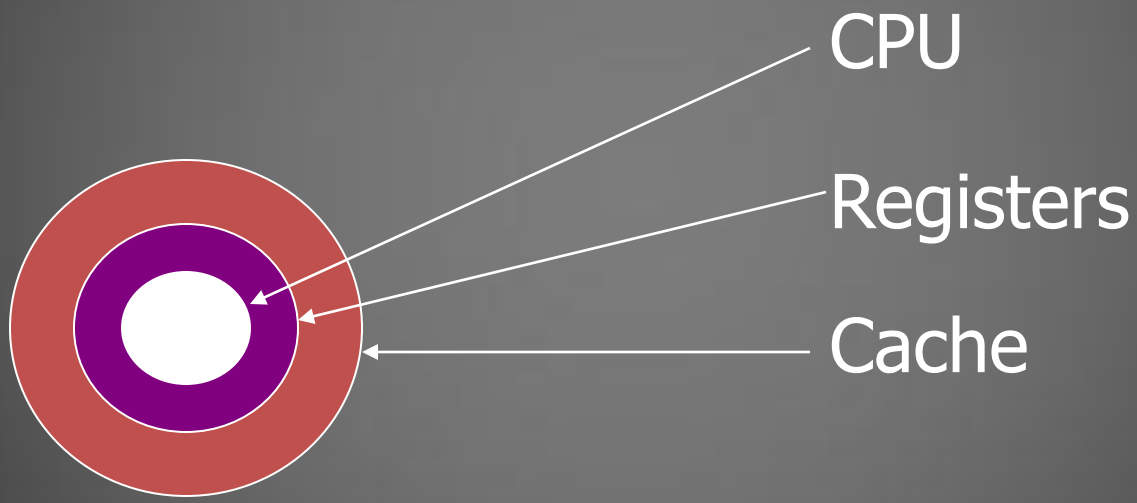
Memory Type Comparison



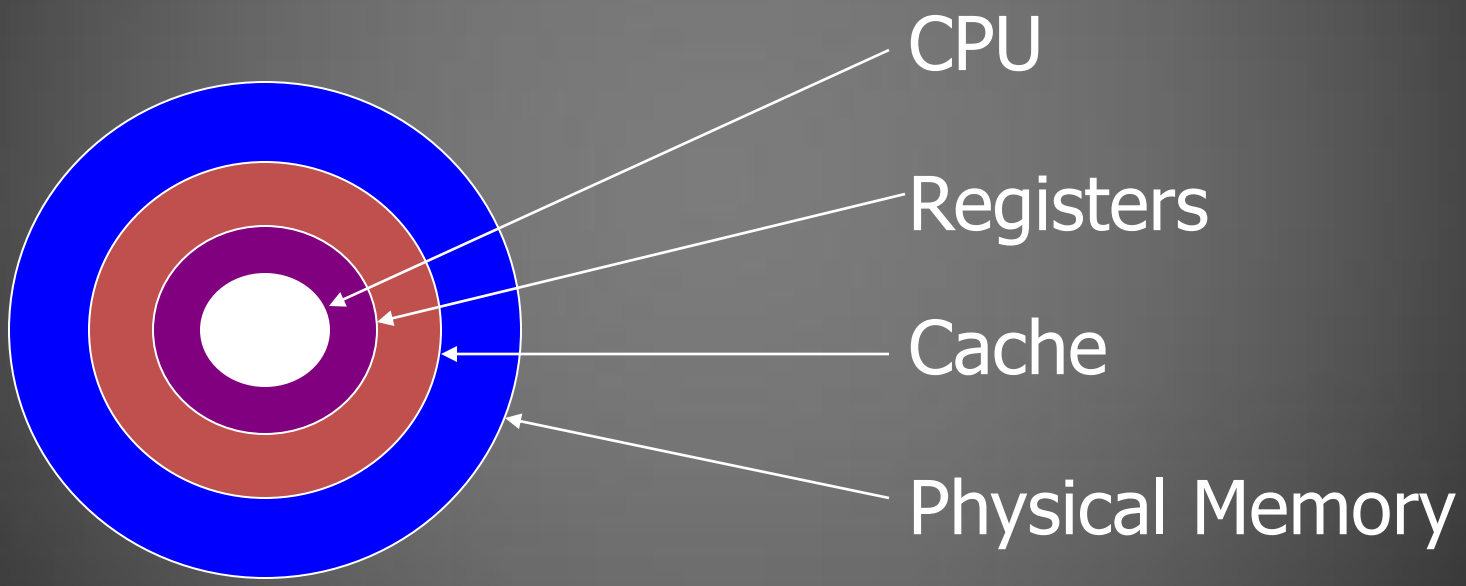
Memory Type Comparison



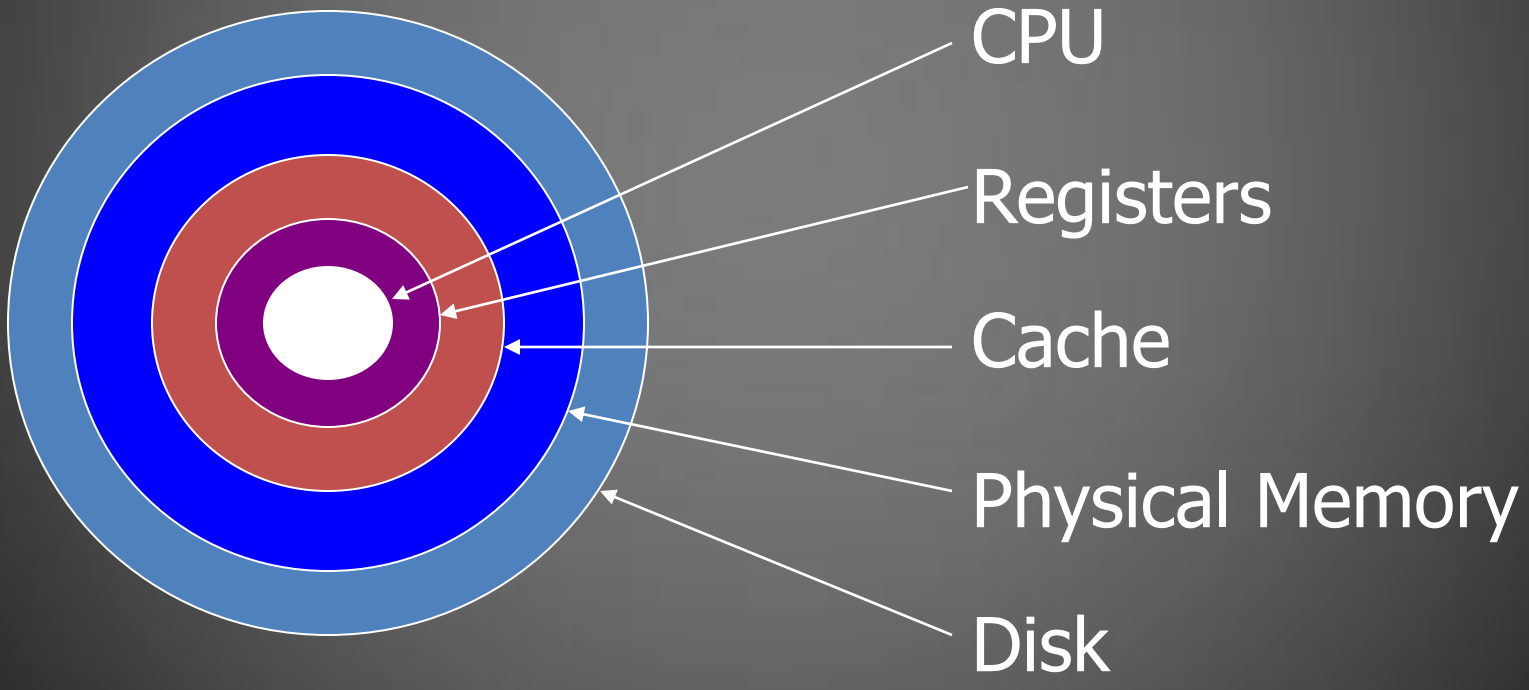
Memory Type Comparison



Memory Type Comparison



Memory Type Comparison



Caches & MMUs

- In general, caches & Memory Management Units (MMUs) are used to boost the memory system size & performance in a computer system by augmenting larger, slower, & cheaper storage devices with smaller, faster, & more expensive ones.

Caches

- Frequently accessed data and / or instructions are kept in a smaller but faster memory block.
- Implementation details are problematic.
- Types:
  - fully-associative
  - direct-mapped
  - set-associative

Memory Management Units (MMUs)

- Allows for the implementation of *Virtual Memory*.
- Physical memory space is mapped into virtual memory space.
- Virtual memory space image is maintained on a slower but larger secondary storage device like a hard disk.

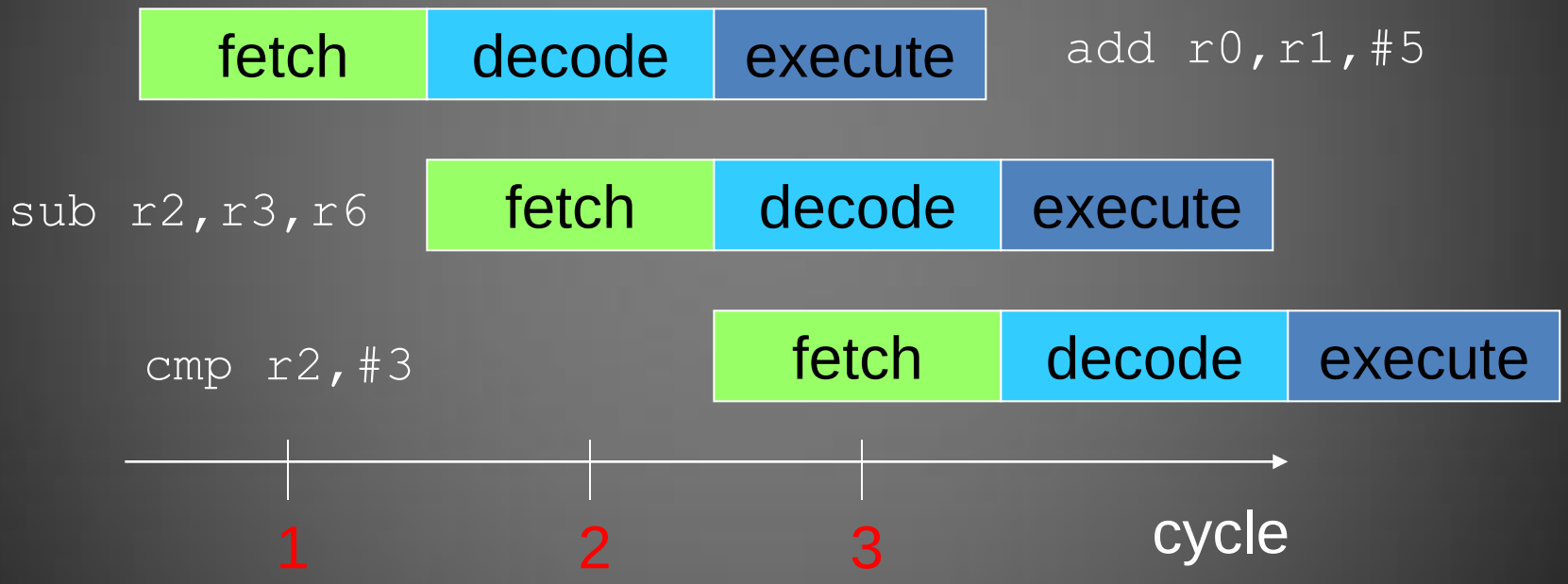
Caches & MMUs Caveat

- *How can the use of caches or MMUs be a problem in real-time embedded systems?*
- Difficult to quantify effect of *misses* of caches and MMUs.
- Real-time embedded require deterministic behavior.

Performance

- Execution Pipelining
- Superscalar Execution

Execution Pipelining



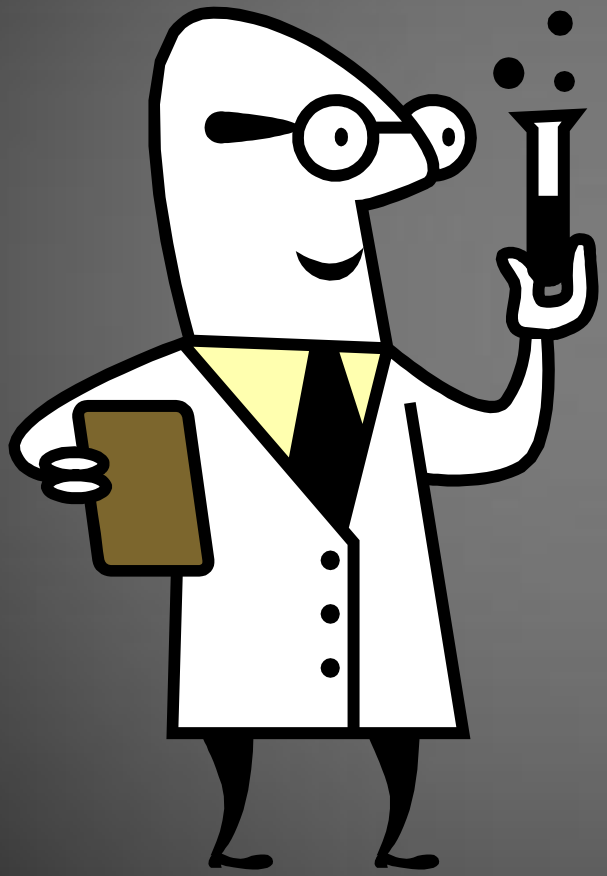
Superscalar Execution

- Multiple instructions executed per cycle.
- Uses pipeline execution and multiple data paths resources.
- Compiler optimizes instruction order of execution to keep the pipeline full and to use data paths efficiently.

Power Consumption (CMOS)

- Lower supply voltage means lower power consumption.
- *NOT*.
- Most of the power consumed by a CMOS output is during transition from high to low or low to high.

Lab Session #4



- Requirements Model Overview

So far we've covered

- Data Flow Diagrams
- Control Flow Diagrams

This week we'll cover

- Requirements Dictionary
- Timing Specification (TSPEC)
- Requirements Model Overview

Requirements Dictionary

- Defines the data & control stores & flows used in the CFDs and DFDs.
- Each entry defines either a primitive or a group of primitives.

Primitive

- Name
- Definition
- Data
  - Units
  - Range
  - Resolution
- Control
  - Value names

Primitive Data Example

- Name: Speed
- Definition: Speed of car
- Data
  - Units: MPH
  - Range: 0-120 MPH
  - Resolution: 0.1 MPH

Primitive Control Example

- Name: Braking
- Definition: Status of brakes
- Control
  - Value names: True or False

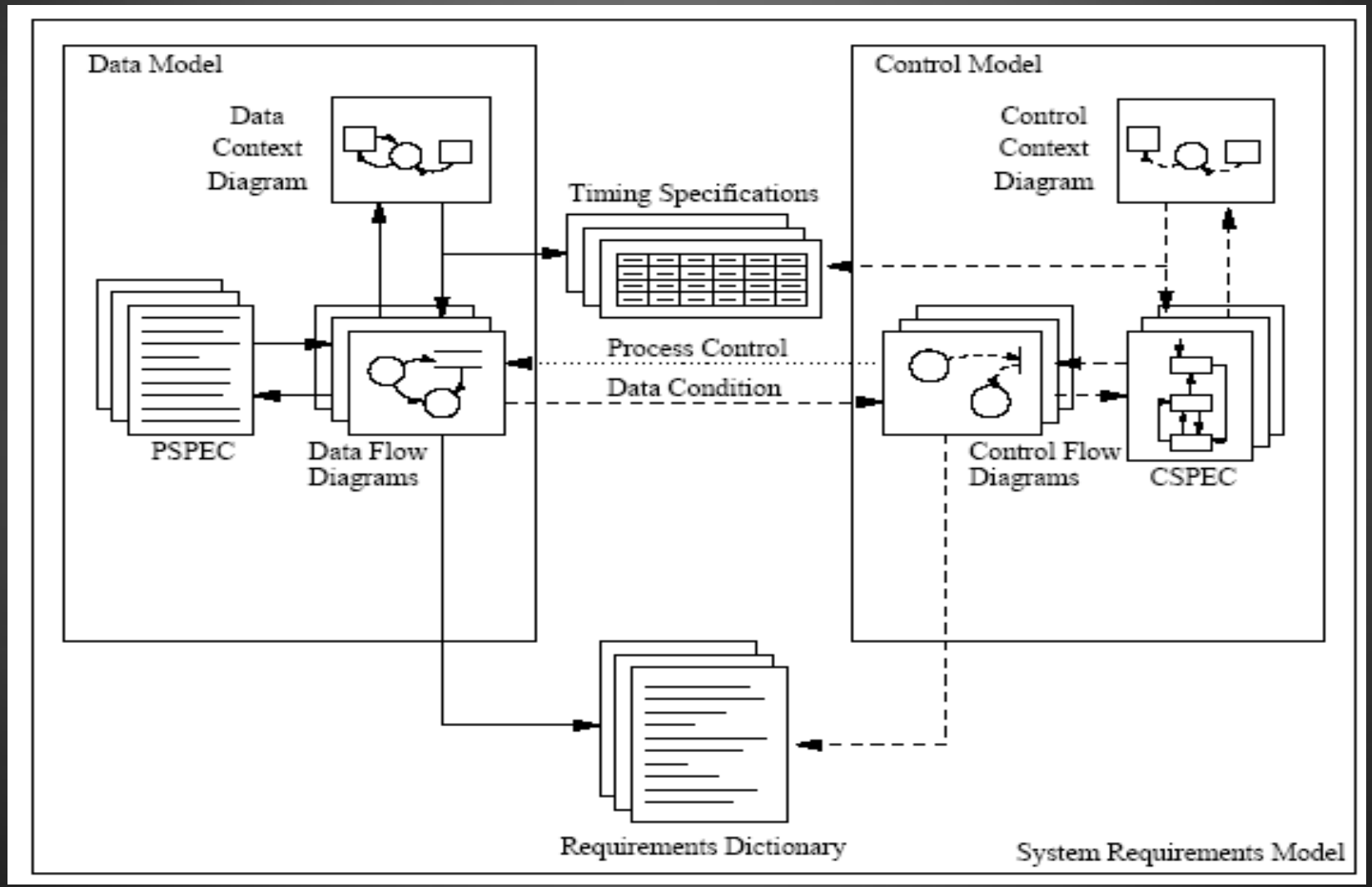
Timing Specification (TSPEC)

- Specifies the desired response time of an output to a change of input.
- Can apply to data and/or control.
- Applies to system level (i.e.: context diagram).

TSPEC Example

- Control & Monitor Auto Process shall respond from changes in inputs to:
  - Displays for Driver within 250 mS.
  - Throttle Position for Throttle Mechanism in 25 mS +/- 10%.

Requirements Model Overview



Design Project

- Chose a SIMPLE embedded application for your Design Project.
- We won't be building anything or writing any code.
- It's just a *paper* Design Project.
- *Relax!*

Design Project Requirements Model Assignment

- Includes
  - Verbal description.
  - Data & Control Context Diagram(s).
  - Data & Control Flow Diagrams.
  - Requirements Dictionary.
  - TSPECs, PSPECs, & CSPECs.
- Use CASE tools if you like but not necessary.
- 2 to 3 pages.
- Due in three weeks.