CS301
Embedded System and Microcomputer Principle

Lecture 11: Bus

2024 Fall

Recap

Storage
- Basics
  - Hierarchy
    - Cache
    - Main memory
    - Secondary storage
  - Type
    - Volatile
      - SRAM
      - DRAM
    - Non-volatile
      - ROM,
      - EPROM
      - EEPROM
- Flash Memory
  - Sector — erase a sector before writing
  - interface — SPI
  - Applications
    - SD Card
    - USB Flash Drive
    - SSD
- SD Card
  - interface — SPI, SD
  - driver — Read, Write, Initial, GetStatus
- FAT File System
  - Type — FAT, NTFS, etc
  - File allocation table
  - Cluster
    - bunch of sectors
    - logic concept
    - FAT16: Cluster address size = 16 bit
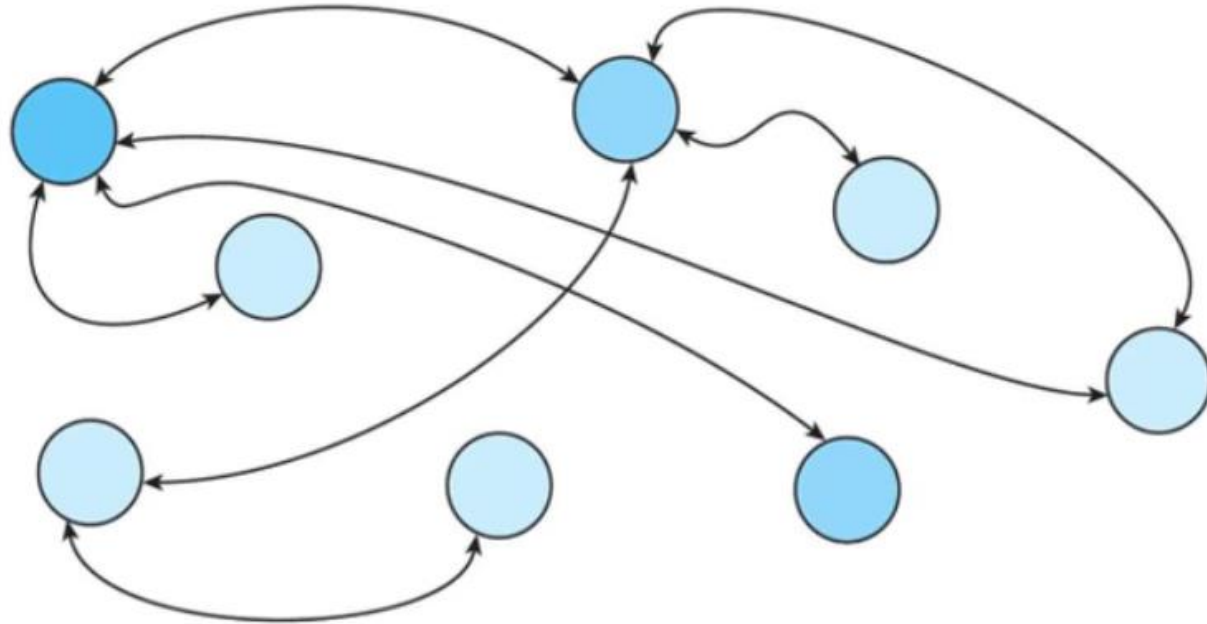 - Implementation
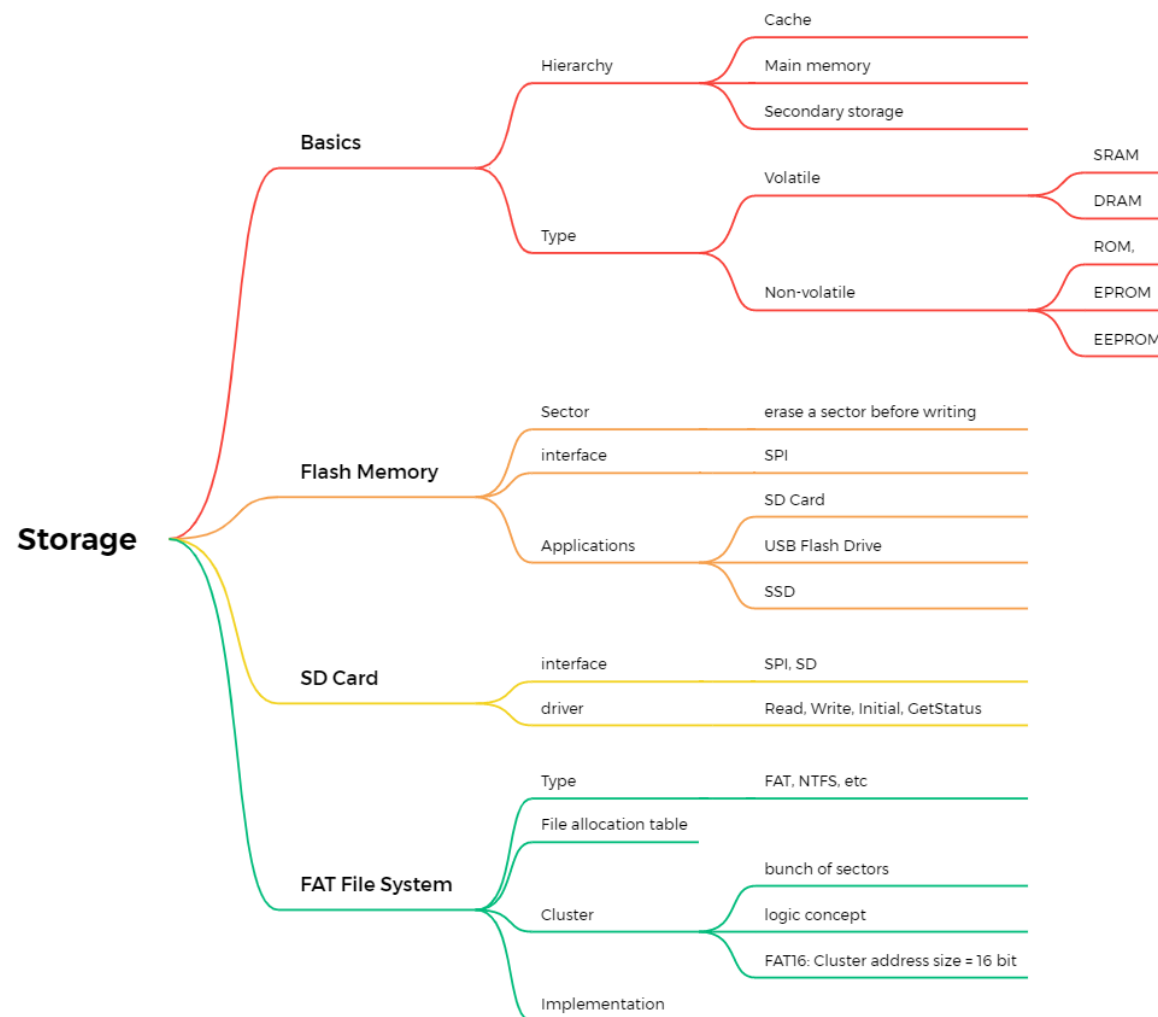
Outline

- **Bus Architecture**
- Bus Performance
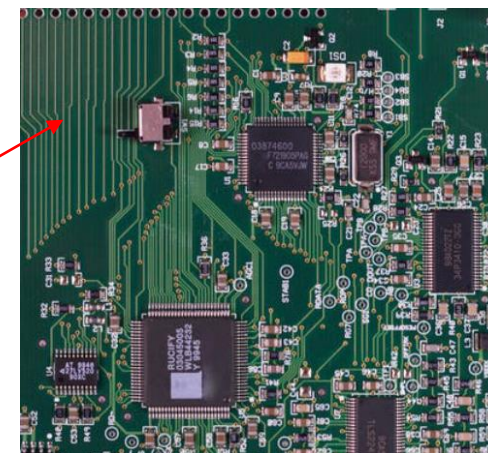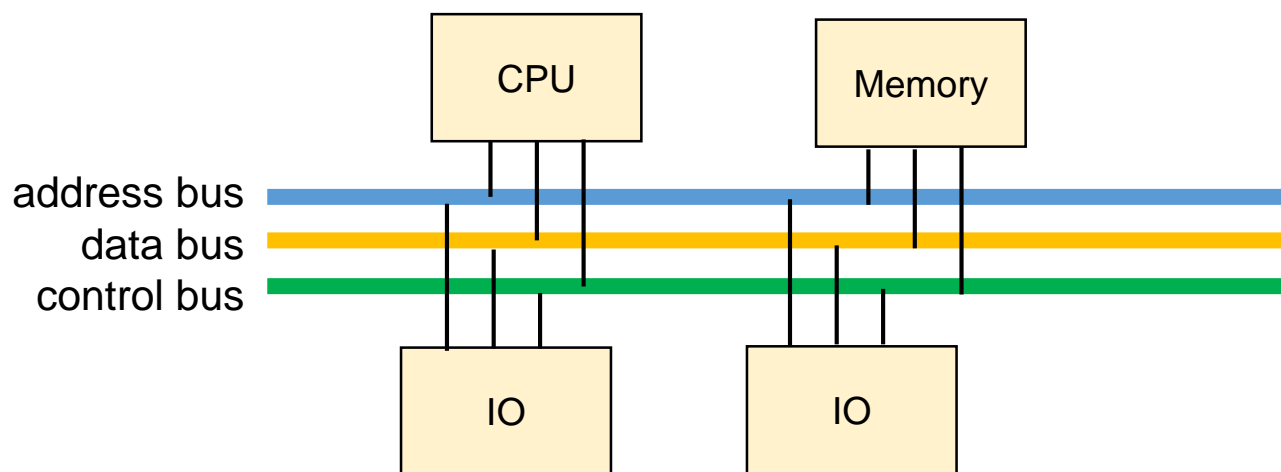- Arbitration
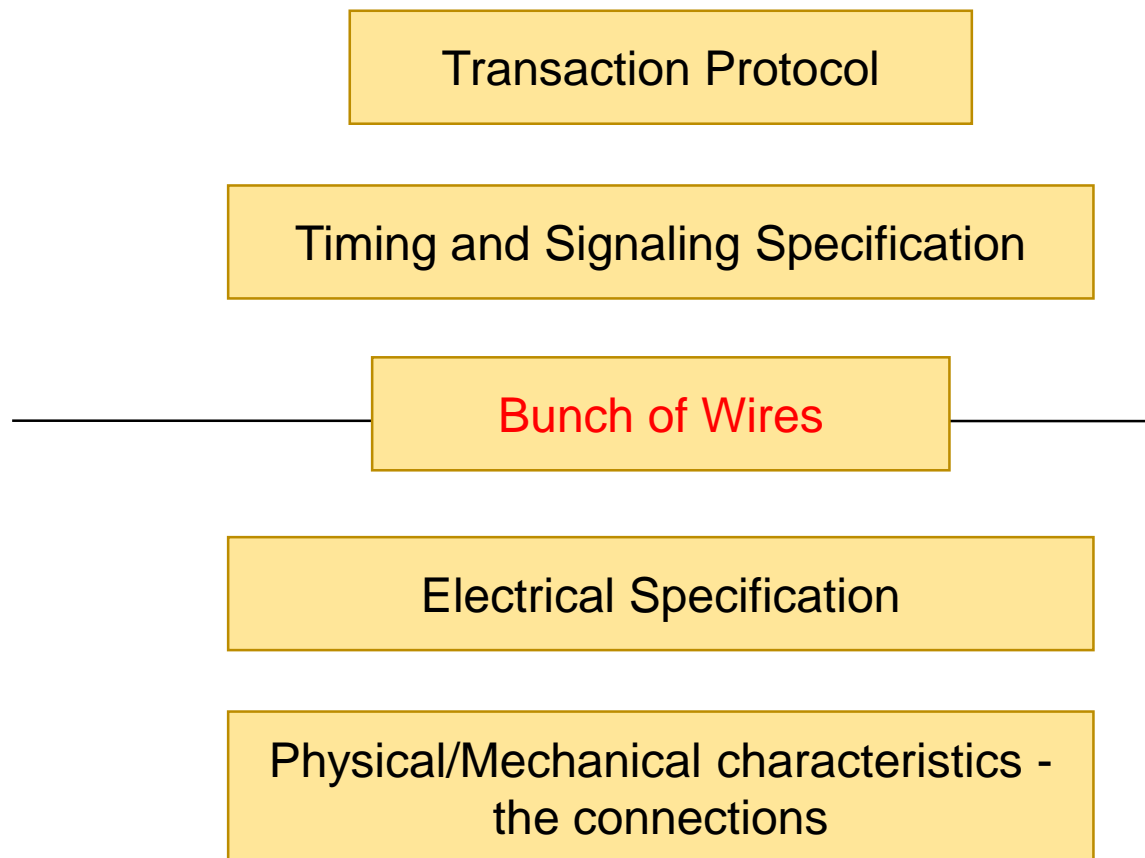- Bus Timing

Bus

- A system without Bus

Bus

- Bus: a shared communication link between subsystems
  - Each line or wire of a bus can at any one time contain a single binary digit.
  - Time Division:
    - At any given moment, only one component is allowed to send information to the bus. Multiple components send information to the bus in different time slots.
  - Shared:
    - Multiple components can be connected to the bus, and each component can exchange information through this set of lines.

What defines a bus?

Transaction Protocol

Timing and Signaling Specification

Bunch of Wires

Electrical Specification

Physical/Mechanical characteristics - the connections

Bus Classification

- By Data Transfer Format
  - Serial Bus: Transfers data one bit at a time.
  - Parallel Bus: Transfers multiple bits simultaneously.
- By Connectivity (Connected Components)
  - In-Chip Bus: Connections within a chip (e.g., Registers, ALU, and other components within CPU).
  - System Bus: Links CPU, Memory, and IO devices.
  - Communication Bus: Connects external devices (e.g., computer host and embedded systems).
- By Timing Control
  - Synchronous Bus: Operates based on a clock signal.
  - Asynchronous Bus: Operates without a clock signal, using handshaking protocols.

System Bus

- Address bus (uni-directional)
  - CPU reads/writes data from memory by addressing a unique location
  - Each I/O device has a unique address as well
- Data bus (bi-directional)
  - Carry information between source and destination
- Control bus (bi-directional)
  - Signal requests and acknowledgments
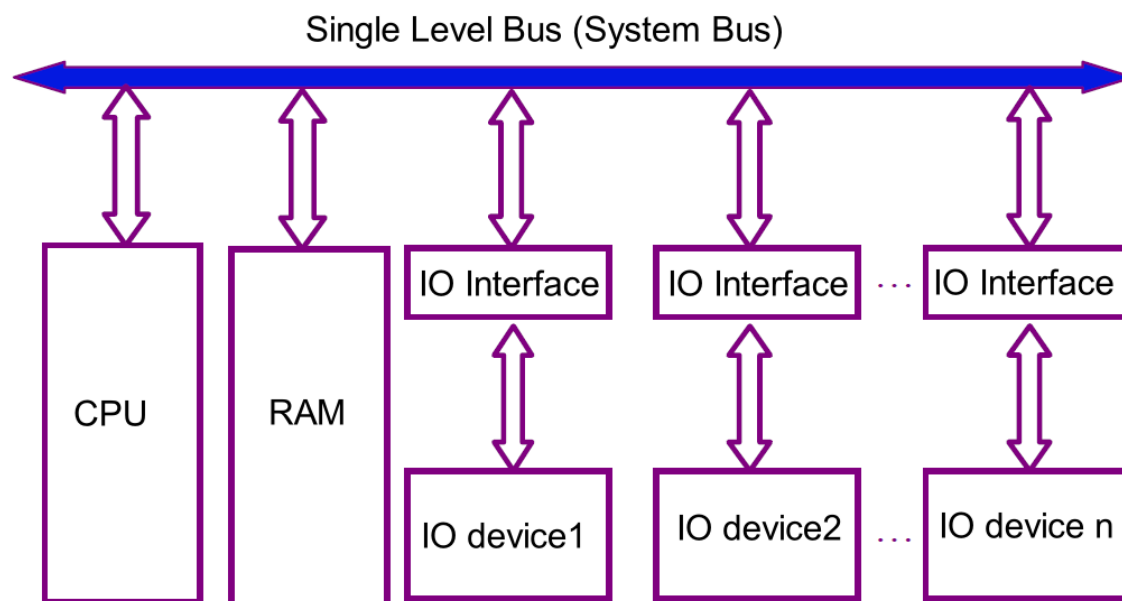 - Indicate what type of information is on the data lines
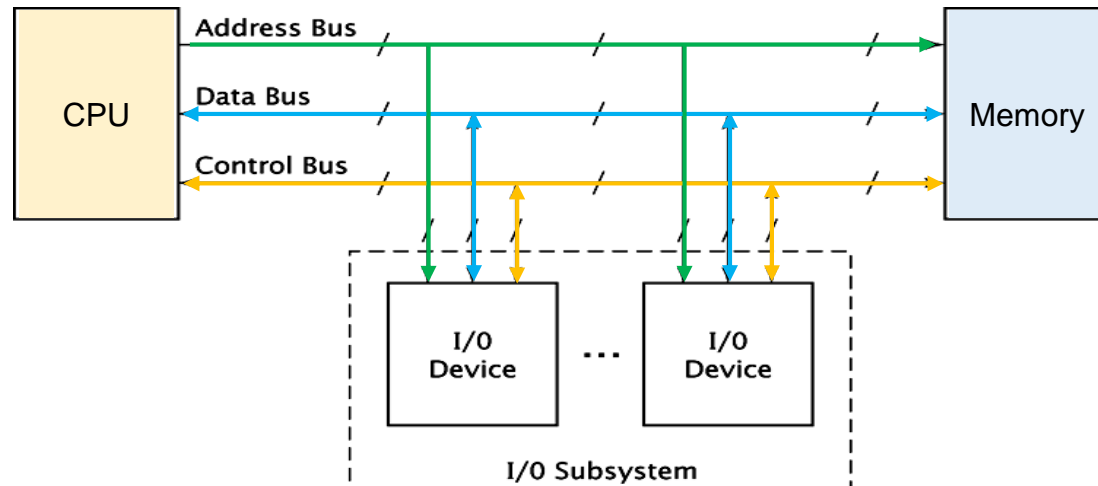
Bus Architecture

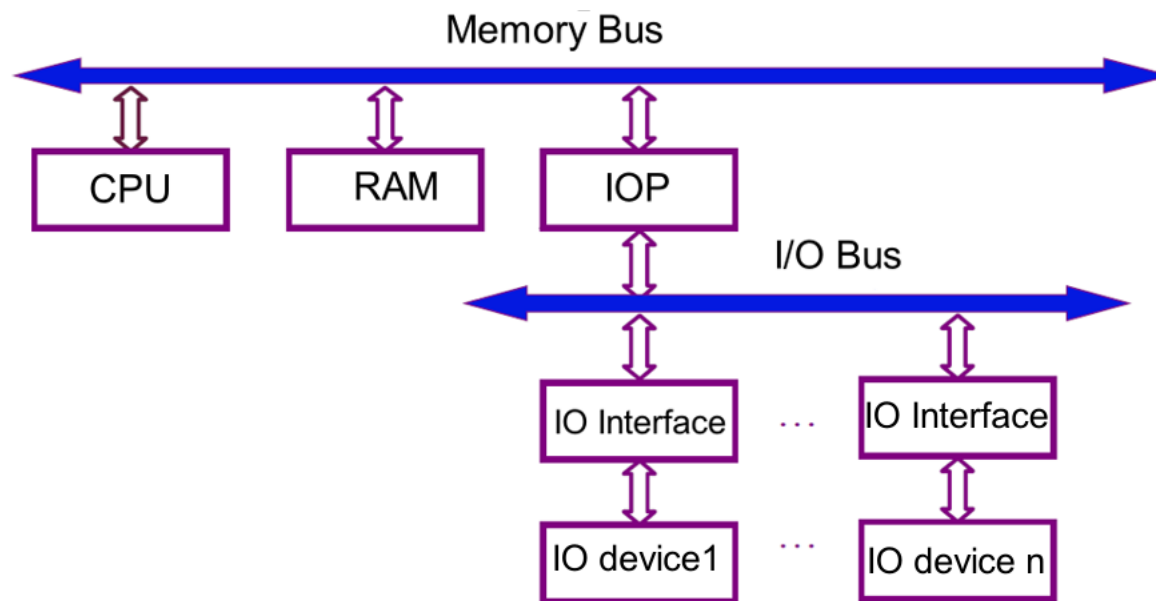- Single level bus architecture
  - In a single bus structure, one common bus is used to communicate between peripherals and microprocessors.
  - Facilitates adding and removing IO devices.
  - Conflicts occur when multiple components need to use the bus, and priority needs to be determined.

Single Level Bus (System Bus)

| CPU | RAM | IO Interface | IO Interface | ... | IO Interface |

| | | IO device1 | IO device2 | ... | IO device n |

Bus Architecture

- Two level bus architecture
  - IOP (channel): a processor with specialized functions that can uniformly manage I/O devices. The channel program is stored in main memory.
  - support burst transfer, refers to the consecutive transfer of multiple data elements, initiated by a single address.
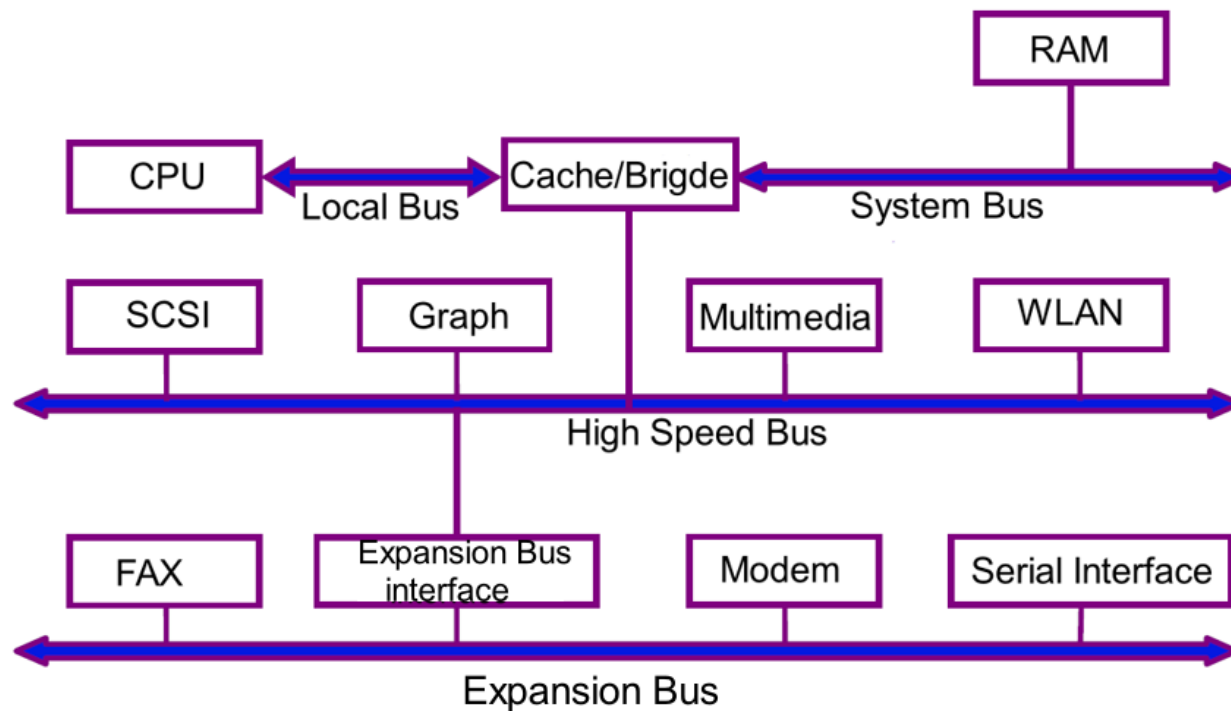
Bus Architecture

- Three level bus architecture
  - Memory centralized with DMA
    - Information exchange between IO devices and main memory doesn't require CPU intervention.
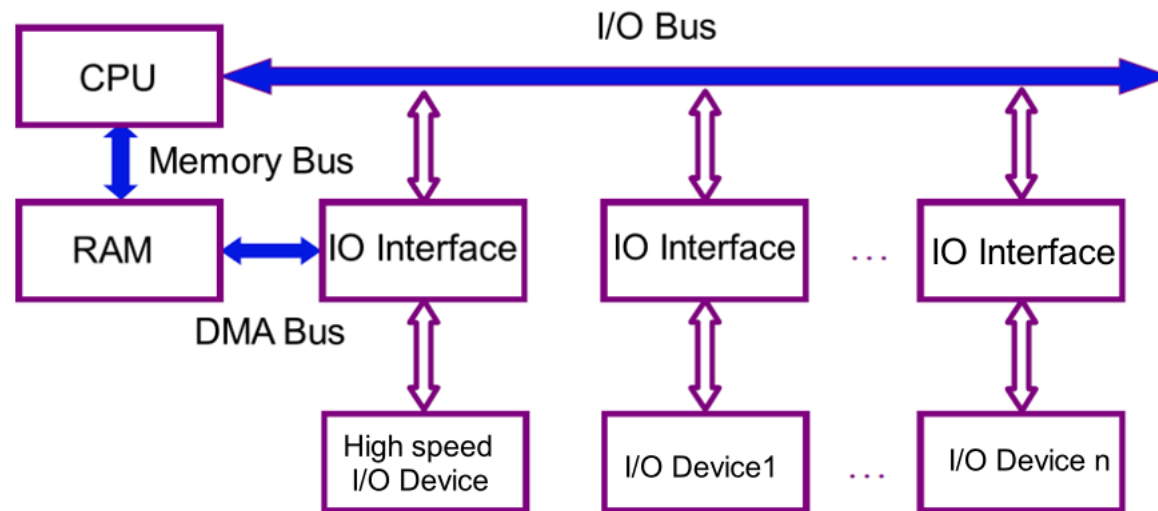
Bus Architecture

- Multilevel bus architecture
  - More advanced
    - Allowing various components to connect efficiently while maintaining high performance and data integrity.

Outline

- Bus Architecture
- **Bus Performance**
- Arbitration
- Bus Timing

Performance

- Bus Width
  - The number of data bits transmitted simultaneously.
  - Examples: 32-bit, 64-bit, etc.
  - Wider bus width ➔ Higher data transfer rate ➔ Increased bandwidth.
- Bus Bandwidth
  - The amount of data transmitted on a bus within a specific time.
  - Measurement: Max **data transfer rate** (usually in MB/s).
- Bus Frequency
  - Speed at which the bus operates, might be different than clock frequency.
  - Higher clock frequency ➔ Faster bus operation ➔ Increased bandwidth.

> Bus Bandwidth = Bus Width x Bus Frequency

Example 1

- Example: In the 32-bit bus system, the data and address are transferred on a shared bus. The clock frequency is 66MHz, during each clock cycle, two data transfers occur. (one on the rising edge and one on the falling edge).
  1. What is the maximum data transfer rate (bus bandwidth)?
  2. Assume it takes 1 clock cycle to transmit the address. How much time needed for CPU writing 128 bit data to the memory if it's in burst transfer mode.

- Solution Q1:
  - 1 clock cycle: 2 bus transfers → bus frequency = 2 * 66 MHz = 132 MHz
  - Bus width = 32 bit = 4 byte
  - Bus bandwidth = 4 * 132 = 528 MB/s

- Solution Q2:
  - To transfer 128bits → 4 words data, it takes 1 clock cycle for address, and 2 clock cycles for data
  - Latency = 3 * clock period = 3/66MHz = 45ns

Example 2

- In a 64-bits bus system, the bus clock frequency of 1GHz, and it takes 1 clock cycle to transmit the address/data. The burst transfer is not supported. The memory needs 6ns to prepare 64-bits data, how much time it takes to read 32 bytes data from memory.

- Solution:
  - Clock period = 1/1GHz = 1ns
  - Bus width = 64/8 = 8bytes
  - For each 8 bytes data transfer → 1 cycle for transferring address, 1 cycle for transferring data, 6 cycles for preparing data → 8 cycles
  - To read 32 bytes date → 4 bus transfers → 32 clock cycles
  - Latency = 32 * clock period = 32ns

Outline

- Bus Architecture
- Bus Performance
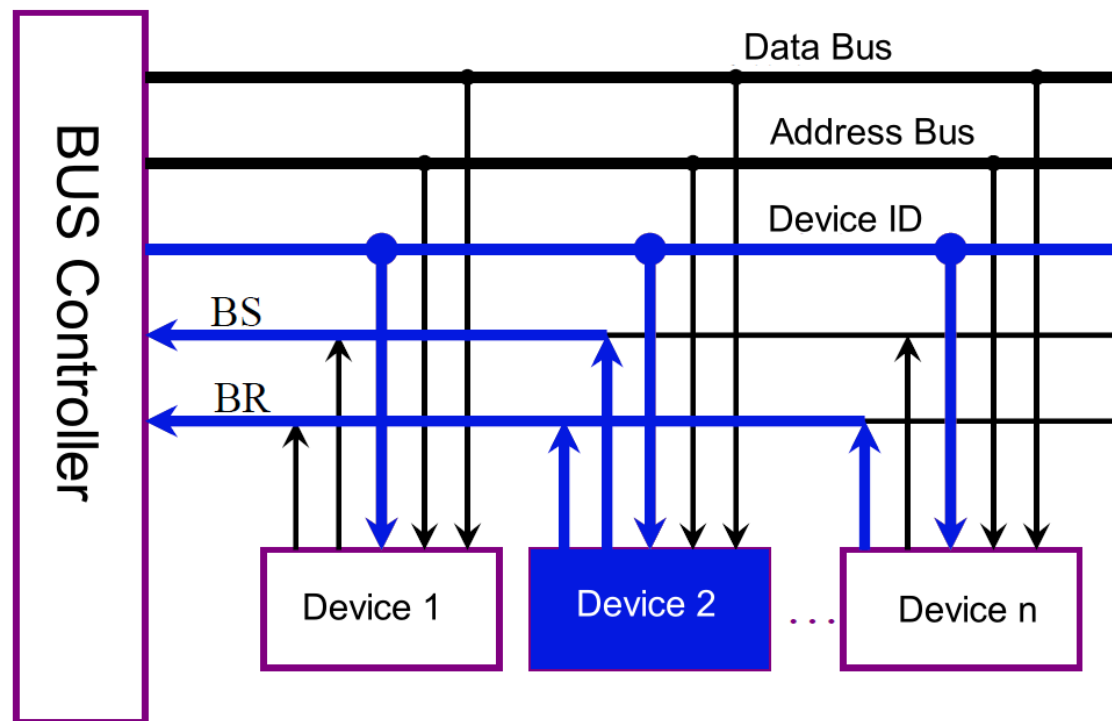- **Arbitration**
- Bus Timing

Bus Arbitration

- Conflicts when more than one module controlling the bus
  - e.g. CPU and DMA controller
  - Only one module may control bus at one time
- Arbitration: choosing a master among multiple requests
  - Centralized Arbitration
    - Daisy Chain
    - Counter Arbitration
    - Independent Request
  - Distributed Arbitration
    - Arbitration handled within individual bus devices.
    - Device failures don't affect others.

Daisy Chain Arbitration

- All bus master use the same line for bus request.
- If the bus busy line is inactive, the bus controller gives the bus grant signal.
- Bus grant signal is propagated serially through all masters starting from nearest one.

BS: Bus Busy
BR: Bus Request
BG: Bus Grant

Counter Arbitration

- An additional Device line is used
- A counter starts counting and sends the count value through the device lines

BS: Bus Busy
BR: Bus Request

Independent Request Arbitration

- All bus masters have their individual bus request and bus grant lines.
- The controller knows which master has requested, so bus is granted to that master.
- Priorities of the masters are predefined so on simultaneous bus requests, the bus is granted based on the priority.
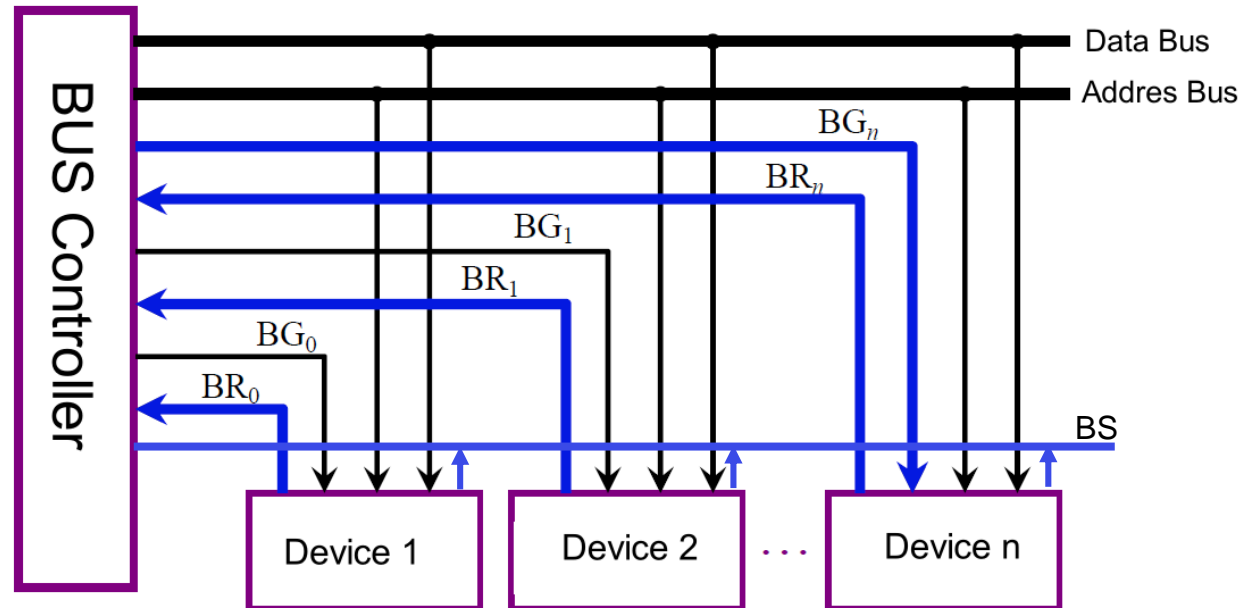
BR: Bus Request
BG: Bus Grant
BS: Bus Busy

Distributed Arbitration

- The arbiters are located within each device
  - The bus device arbiter sends its own arbitration number onto the shared bus.
  - The arbiter compares the arbitration number it receives from the bus with its own to determine priority.
  - If its own priority is lower, it withdraws its arbitration number.
  - The highest-priority arbiter remaining on the bus gains control of the bus.

Arbitration Bus

AB0
AB1
AB2
…
AB7

Master 1
controller

Master 2
controller

Master 3
controller
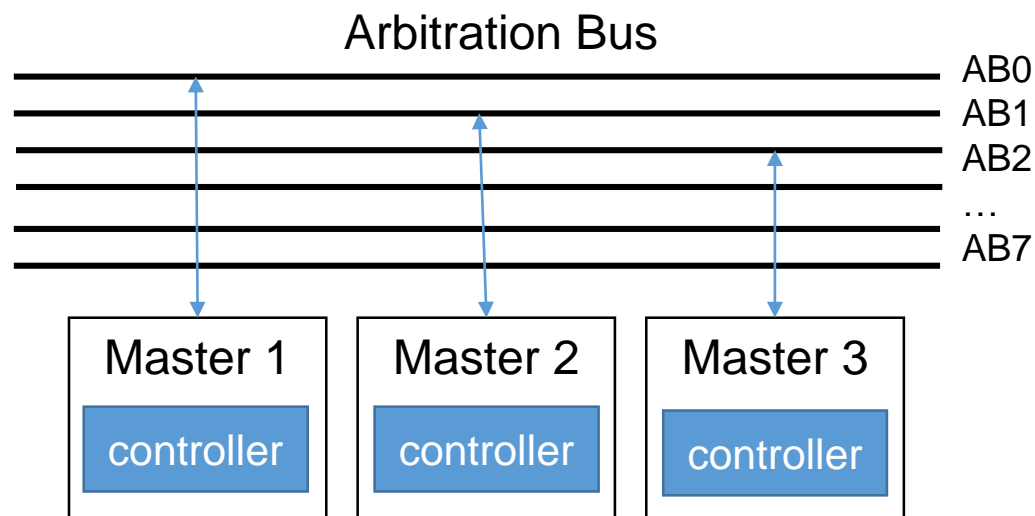
Outline

- Bus Architecture
- Bus Performance
- Arbitration
- **Bus Timing**

Bus Cycle

1. **Request:**
   - a device that wishes to initiate a transaction on the bus sends a request signal to the bus controller or arbiter
   - Arbitration: If there are multiple devices contending for bus access, an arbitration process occurs. The arbiter determines which device should be granted access based on arbitration scheme

2. **Addressing:**
   - Master sends address and control signal to the bus

3. **Transfer:**
   - Master and slave perform data trainsmission.

4. **Complete:**
   - After the data transfer is complete, master removes relevant information from the bus and release the control from the bus.
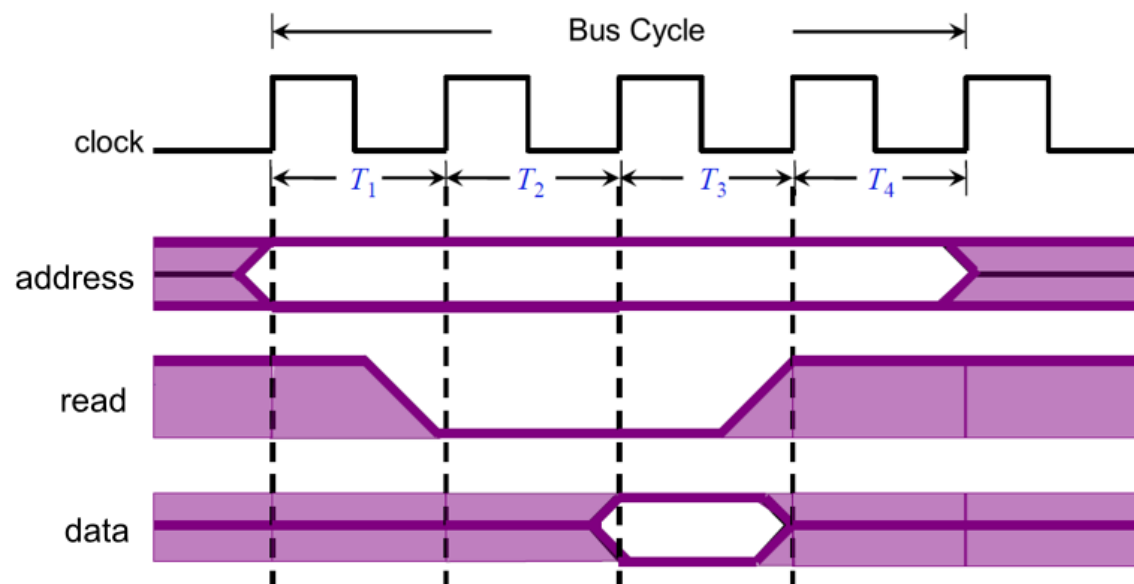
Bus Timing

- Co-ordination of events on bus
  - Synchronous – controlled by a clock,
    - support burst transfer
  - Asynchronous – timing is handled by well-defined specifications(handshaking), i.e., a response is delivered within a specified time after a request
  - An event like CPU Reading from Main Memory:
    - Sending the address to the main memory
    - Sending the read signal to the main memory
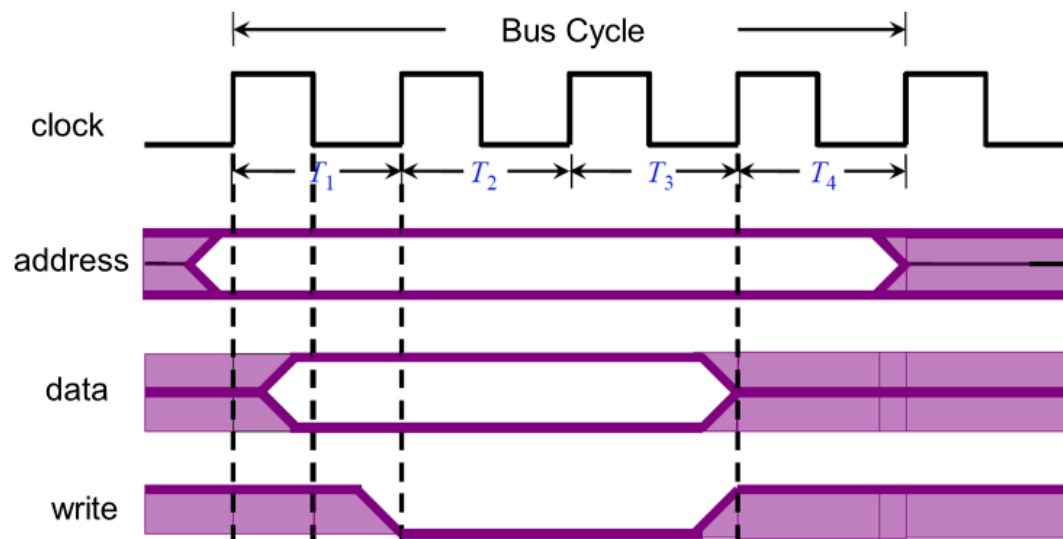   - Delivering the data on the data bus to the CPU

CLOCK

T_0 | T_1 | T_2 | T_3 | T_0 | T_1 | T_2 | T_3 | T_0 | T_1 | T_2 | T_3

Bus cycle   Bus cycle   Bus cycle
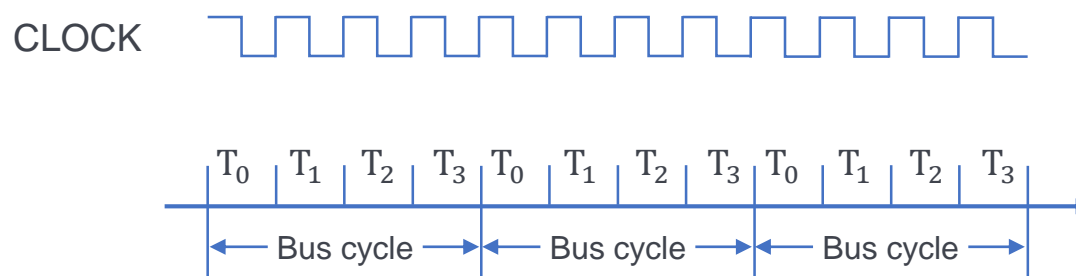
Synchronous Timing Diagram

- CPU(Master) reads data from memory(Slave)

Synchronous Timing Diagram

• CPU(Master) writes data to memory(Slave)

Asynchronous Timing- Read Diagram

- CPU(Master) reads data from memory(Slave)
  - ReadReq: Read request, informs the slave device to perform a read operation, and the address information is sent to the address/data lines simultaneously.
  - DataRdy: Data ready, indicates that the slave device has prepared the data, and the master device can retrieve the data, which is also sent to the address/data lines simultaneously.
  - Ack: Acknowledgment signal line, indicates that the request signal from the other party has been received.
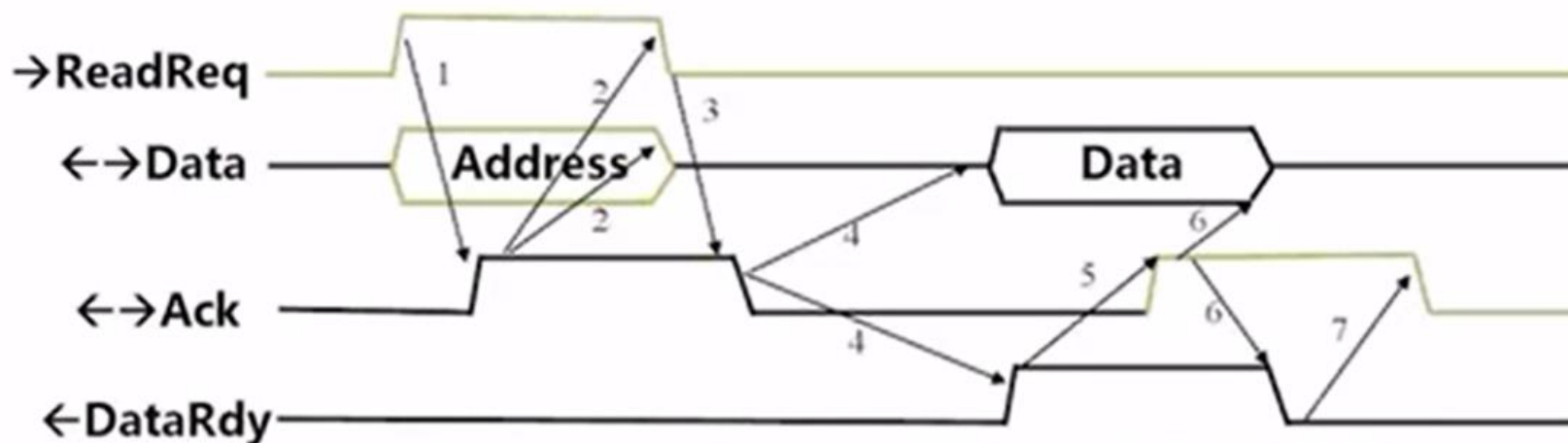
SOC Bus Architecture

- Advanced Microcontroller Bus Architecture
  - On-chip bus protocol from ARM
    - On-chip interconnect specification for the connection and management of functional blocks including processor and peripheral devices
- **AMBA2.0**
  - AHB
    - Widely used on ARM7, ARM9 and ARM Cortex-M based deisgns
  - ASB
  - APB
- AMBA3.0
  - AXI3 (or AXI v1.0)
    - widely used on ARM Cortex-A processors including Cortex-A9
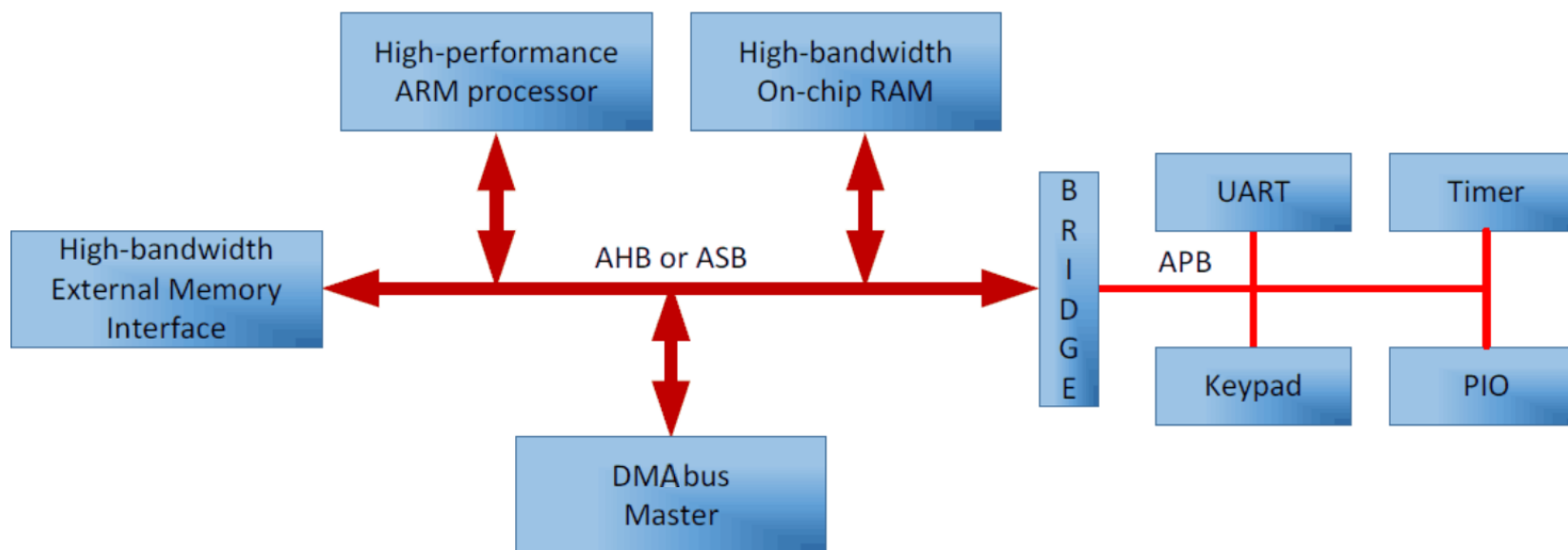 - AHB-Lite v1.0
  - APB3 v1.0
  - ATB v1.0

AHB: Advanced High-performance Bus
ASB: Advanced System Bus
APB: Advanced Peripheral Bus

AMBA2.0

- ARM Cortex-M processor uses AMBA2.0 to interface to the System Bus
  - Enables IP re-use
  - Flexibility
  - Compatibility
  - Well supported

AHB and APB

- AHB (Advanced High-performance Bus)
  - High Performance
  - Connects CPUs, High-Speed Memory, DMA Controllers
  - High Bandwidth Data Transfer
  - Multiple Masters Supported
  - Ideal for High-Performance Components
- APB (Advanced Peripheral Bus)
  - Low-Power Peripheral Bus
  - Connects Low-Speed Peripherals
  - Suitable for Timers, GPIO Controllers, UART, etc.
  - Designed for Low-Speed, Low-Power Applications
  - Connects Peripheral Devices to System Bus