Memory & Bus

Jin-Soo Kim (jinsookim@skku.edu)
Computer Systems Laboratory
Sungkyunkwan University
http://csl.skku.edu
Memory Architectures
Memory Components

- Several different types of memory
  - SRAM
  - DRAM
  - Flash
- Each type of memory comes in varying
  - Capacities
  - Widths
Random-Access Memory

- Dynamic RAM is dense, requires refresh
  - Synchronous DRAM is dominant type
  - SDRAM uses clock to improve performance, pipeline memory accesses

- Static RAM is faster, less dense, consumes more power
Read-Only Memory

- ROM may be programmed at factory
- Flash is dominant form of field-programmable ROM
  - Electrically erasable, must be block erased
  - Random access, but write/erase is much slower than read
  - NOR flash is more flexible
  - NAND flash is more dense
# Requirements

## Code

- Mobile
- Consumer Electronics
- Networking

<table>
<thead>
<tr>
<th></th>
<th>Read</th>
<th>Writes</th>
<th>Density</th>
<th>Reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile</td>
<td>Fast Random</td>
<td>Medium</td>
<td>Small – Medium</td>
<td>No bad bits</td>
</tr>
</tbody>
</table>

## Data

- Cards
- MP3
- USB Drives

<table>
<thead>
<tr>
<th></th>
<th>Read</th>
<th>Writes</th>
<th>Density</th>
<th>Reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cards</td>
<td>Fast Sequential</td>
<td>Fast</td>
<td>Large</td>
<td>Bad bits allowed</td>
</tr>
</tbody>
</table>

*Source: “Non-Volatile Memories”, Intel Corp.*
NOR XIP

**Pros**
- Simple, easy to design
- Execute-In-Place (XIP)
- Predictable read latency
- Code + Storage in NOR
- Firmware upgrades

**Cons**
- Slow read speed
- Much slower write speed
- The high cost of NOR
NOR Shadowing

- **Pros**
  - Faster read and write
  - Easy boot-up
  - Use a relatively pricey NOR only to boot up the system
  - Code can be compressed

- **Cons**
  - Larger DRAM needed
  - Require more design time
  - Not energy efficient
NAND Shadowing

- **Pros**
  - Faster read and write
  - Cost effective
  - NAND for both code and data storage

- **Cons**
  - Require a special boot mechanism
  - Extensive ECC for NAND
  - Larger DRAM needed
  - Require more design time
  - Not energy efficient
Hybrid NAND Shadowing

- **Pros**
  - Much faster read and write speed
  - ECC embedded
  - Cost effective
  - NAND for both code and data storage

- **Cons**
  - Larger DRAM needed
  - Not energy efficient
NAND Demand Paging

- **Pros**
  - Less DRAM required
  - Low cost
  - Energy efficient
  - NAND for both code and data storage

- **Cons**
  - Require MMU-enabled CPU
  - Unpredictable read latency
  - Complex to design and test
The CPU Bus
The CPU Bus

- Bus allows CPU, memory, devices to communicate
  - Shared communication medium

- A bus is:
  - A set of wires
  - A communications protocol
Bus Protocols

- Bus protocol determines how devices communicate.
- Devices on the bus go through sequences of states.
  - Protocols are specified by state machines, one state machine per actor in the protocol
- May contain asynchronous logic behavior
Four-cycle Handshake

- Device 1 raise enq
- Device 2 responds with ack
- Device 2 lowers ack once it has finished
- Device 1 lowers enq
Microprocessor Busses

- Clock provides synchronization
- R/W is true when reading
- Address is $a$-bit bundle of address lines
- Data is $n$-bit bundle of data lines
- Data ready signals when $n$-bit data is ready
Bus Multiplexing
DMA

- Direct memory access (DMA) performs data transfers without executing instructions
  - CPU sets up transfer
  - DMA engine fetches, writes
- DMA controller is a separate unit
Bus Mastership

- By default, CPU is bus master and initiates transfers.
- DMA must become bus master to perform its work
  - CPU can’t use bus while DMA operates
- Bus mastership protocol:
  - Bus request
  - Bus grant
DMA Operation

- CPU sets DMA registers for start address, and length
- DMA status register controls the unit
- Once DMA is bus master, it transfers automatically
  - May run continuously until complete
  - May use every $n^{th}$ bus cycle
System Bus Configurations

- Multiple busses allow parallelism
  - Slow devices on one bus
  - Fast devices on separate bus
- A bridge connects two busses
ARM AMBA
(Advanced Microcontroller Bus Architecture)
AMBA Specification

▪ Advanced eXtensible Interface (AXI)
▪ Advanced High-performance Bus (AHB)
  • High-performance system bus
▪ Advanced System Bus (ASB)
▪ Advanced Peripheral Bus (APB)
  • Lower speed, lower cost
  • All devices are slaves
▪ Advanced Trace Bus (ATB)
Typical Architecture

- High-performance ARM processor
- High-bandwidth on-chip RAM
- DMA bus master
- UART
- Timer
- Keypad
- PIO

AHB Bridge APB
AHB

- High performance
- Pipelined operation
- Burst transfers
- Split transactions
- Multiple bus masters
- Single-cycle bus master handover
- Single-clock edge operation
Simple Transfer

Diagram showing the timing for an address phase and a data phase.

- **HCLK**: Timing signal for the clock.
- **HADDR[31:0]**: Address signal.
- **Control**: Control signals.
- **HWDATA[31:0]**: Data signal for read operations.
- **HREADY**: Ready signal for the transaction.
- **HRDATA[31:0]**: Data signal for write operations.
Transfer with Wait States

- Address phase
- Data phase

- HCLK
- HADDR[31:0]: A
- Control
- HWDATA[31:0]: Data (A)
- HREADY
- HRDATA[31:0]: Data (A)
Burst Transfer

- **HCLK**
- **HTRANS[1:0]**
  - T1: NONSEQ
  - T2: SEQ
  - T3: SEQ
  - T4: SEQ
- **HADDR[31:0]**
  - T1: 0x38
  - T2: 0x3C
  - T3: 0x40
  - T4: 0x44
- **HBURST[2:0]**
  - INCR4
- **HWRITE**, **HSIZE[2:0]**
  - Control for burst
  - SIZE = Word
- **HPROT[3:0]**
- **HWDATA[31:0]**
  - Data (0x38)
  - Data (0x3C)
  - Data (0x40)
  - Data (0x44)
- **HREADY**
- **HRDATA[31:0]**
  - Data (0x38)
  - Data (0x3C)
  - Data (0x40)
  - Data (0x44)
APB

- Low power
- Latched address and control
- Simple interface
- Suitable for many peripherals