Embedded Computing Platforms

Chapter 4 (Sections 4.1-4.4)
Platform components

- CPUs.
- Interconnect buses.
- Memory.
- Input/output devices.

Implementations:
- System-on-Chip (SoC) vs. Multi-Chip
  - Microcontroller vs. microprocessor
- Commercial off-the-shelf (COTS) vs. custom
- FPGA & Platform FPGA
CPU Buses

- Mechanism for communication with memories and I/O devices

- Bus components:
  - signal wires with designated functions
  - protocol for data transfers
  - electrical parameters (voltage, current, capacitance, etc.)
  - physical design (connectors, cables, etc.)
CPU Bus Types

- **Synchronous vs. Asynchronous**
  - Sync: all op’s synchronized to a clock
  - Async: devices signal each other to indicate start/stop of operations
    - May combine sync/async (80x86 “Ready” signal)

- **Data transfer types:**
  - Processor to/from memory
  - Processor to/from I/O device
  - I/O device to/from memory (DMA)

- **Data bus types**
  - Parallel (data bits transferred in parallel)
  - Serial (data bits transferred serially)
Hierarchical Bus Architecture

- CPU
- Cache
- Main Memory
- System bridge
- Expansion LAN Controller
- Expansion Video Controller
- Expansion Mouse/Keyboard
- Expansion Disk Controller
- IDE/SCSI
- USB
- USB Device
- Local controller
- Expansion USB Controller
Typical bus data rates

Source: Peter Cheung “Computer Architecture & Systems Course Notes”
ARM Advanced Microcontroller Bus Architecture (AMBA)

- **On-chip** interconnect specification for SoC
- Promotes re-use by defining a common backbone for SoC modules using standard bus architectures
  - **AHB** – Advanced High-performance Bus (system backbone)
    - High-performance, high clock freq. modules
    - Processors to on-chip memory, off-chip memory interfaces
  - **APB** – Advanced Peripheral Bus
    - Low-power peripherals
    - Reduced interface complexity
  - **ASB** – Advanced System Bus
    - High performance alternate to AHB
  - **AXI** – Advanced eXtensible Interface
  - **ACE** – AXI Coherency Extension
  - **ATB** – Advanced Trace Bus
Example AMBA System

- High Performance ARM processor
- AHB
- APB Bridge
  - UART
  - Timer
  - Keypad
  - PIO

- High Bandwidth External Memory Interface
- High-bandwidth on-chip RAM
- DMA Bus Master

- High Performance Pipelined Burst Support Multiple Bus Masters

Low Power Non-pipelined Simple Interface
STM32F407 Microcontroller

- AHB 168MHz
- APB 142MHz
- APB 84MHz

External Bus
CoreLink peripherals for AMBA

“CoreLink” (orange blocks)

Interconnect + memory controllers for Cortex/Mali
## Cortex A9 System IP

### Interconnect SoC components

<table>
<thead>
<tr>
<th>Description</th>
<th>AMBA Bus</th>
<th>System IP Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced AMBA 3 Interconnect IP</td>
<td>AXI</td>
<td>NIC-301, PL301</td>
</tr>
<tr>
<td>DMA Controller</td>
<td>AXI</td>
<td>DMA-330, PL330</td>
</tr>
<tr>
<td>Level 2 Cache Controller</td>
<td>AXI</td>
<td>L2C-310, PL310</td>
</tr>
<tr>
<td>Dynamic Memory Controller</td>
<td>AXI</td>
<td>DMC-340, PL340</td>
</tr>
<tr>
<td>DDR2 Dynamic Memory Controller</td>
<td>AXI</td>
<td>DMC-342</td>
</tr>
<tr>
<td>Static Memory Controller</td>
<td>AXI</td>
<td>SMC-35x, PL35x</td>
</tr>
<tr>
<td>TrustZone Address Space Controller</td>
<td>AXI</td>
<td>PL380</td>
</tr>
<tr>
<td>CoreSight™ Design Kit</td>
<td>ATB</td>
<td>CDK-11</td>
</tr>
</tbody>
</table>
Microprocessor buses

- Clock provides synchronization.
- R/W is true when reading (R/W' is false 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 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 synchronous and/or asynchronous logic behavior.
Timing diagrams

- **A**: Low, High (10 ns), Rising, Falling
- **B**: Changing, Stable
- **C**: Timing constraint, Time
Typical bus read and write timing
State diagrams for bus read

CPU

Get data

Done

Ack?

Yes

No

Wait

DEVICE

Ack & Send data

Release ack

Ready?

Yes

No

Wait

Adrs

Adrs

start

Yes

No
Bus wait state

![Diagram showing Bus wait state with time axis (X) and various signals (Y): Clock, R/W, Address enable, Address, Data ready, Data. The wait state is highlighted in gray.](image)
Bus burst read
Bus multiplexing

Pins shared by address and data to minimize total # of bus connections.
Types of memory

- **ROM:**
  - Mask-programmable.
  - Flash programmable.

- **RAM: Static vs. Dynamic**
  - **SRAM:**
    - Faster.
    - Easier to integrate with logic.
    - Higher power consumption.
  - **DRAM:**
    - Denser.
    - Must be refreshed.
ROM/RAM device organization

Memory "organization" = $2^n \times d$
(from system designer's perspective)

- Size.
  - Address width. $n = r + c$
- Aspect ratio.
  - Data width $d$. 

Memory array

Address
- Row # $n$
- Column # $c$

Data bus connection
Typical generic SRAM

- Often have separate OE’ and WE’ instead of one R/W’ signal.
- Multi-byte Data bus devices usually have byte-select signals.
512K x 16 SRAM (on uCdragon board)
Generic SRAM timing

CE’
R/W’
Adrs
Data

From SRAM
From CPU

read
write
time
ISSI IS61LV51216 SRAM read cycle

- **ADDRESS**
- **OE**
- **CE**
- **LB, UB**
- **DOUT**
- **VDD Supply Current**

Key timing parameters:
- $t_{AA}$
- $t_{DOE}$
- $t_{LZOE}$
- $t_{LZCE}$
- $t_{ACE}$
- $t_{BA}$
- $t_{RC}$
- $t_{OH}$
- $t_{HZOE}$
- $t_{HZCE}$
- $t_{HZB}$

**DATA VALID**
### ISSI IS61LV51216 SRAM timing

#### READ CYCLE SWITCHING CHARACTERISTICS

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>-8 Min.</th>
<th>-10 Min.</th>
<th>-12 Min.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Max.</td>
<td>Max.</td>
<td>Max.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ns</td>
<td>ns</td>
<td>ns</td>
<td></td>
</tr>
<tr>
<td>trc</td>
<td>Read Cycle Time</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>ns</td>
</tr>
<tr>
<td>taa</td>
<td>Address Access Time</td>
<td>—</td>
<td>8</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>toha</td>
<td>Output Hold Time</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>ns</td>
</tr>
<tr>
<td>tace</td>
<td>CE Access Time</td>
<td>—</td>
<td>8</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>tdoe</td>
<td>OE Access Time</td>
<td>—</td>
<td>3.5</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>thzoed</td>
<td>OE to High-Z Output</td>
<td>—</td>
<td>3</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>tlzoe</td>
<td>OE to Low-Z Output</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>thzced</td>
<td>CE to High-Z Output</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>tlzce</td>
<td>CE to Low-Z Output</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>ns</td>
</tr>
<tr>
<td>tba</td>
<td>LB, UB Access Time</td>
<td>—</td>
<td>3.5</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>thzbd</td>
<td>LB, UB to High-Z Output</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>tlzbd</td>
<td>LB, UB to Low-Z Output</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>tpu</td>
<td>Power Up Time</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>tpd</td>
<td>Power Down Time</td>
<td>—</td>
<td>8</td>
<td>10</td>
<td>12</td>
</tr>
</tbody>
</table>
STM32 Flexible Static Memory Controller (FSMC) - STM32F4xx Tech. Ref. Manual, Chap. 36

- Control external memory on AHB bus in four 256M banks
- Upper address bits decoded by the FSMC

### Memory Mapping

<table>
<thead>
<tr>
<th>Address</th>
<th>Banks</th>
<th>Supported Memory Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>6000 0000h</td>
<td>Bank 1</td>
<td>4 x 64 MB</td>
</tr>
<tr>
<td>6FFF FFFFh</td>
<td>Bank 1</td>
<td></td>
</tr>
<tr>
<td>7000 0000h</td>
<td>Bank 2</td>
<td>4 x 64 MB</td>
</tr>
<tr>
<td>7FFF FFFFh</td>
<td>Bank 2</td>
<td></td>
</tr>
<tr>
<td>8000 0000h</td>
<td>Bank 3</td>
<td>4 x 64 MB</td>
</tr>
<tr>
<td>8FFF FFFFh</td>
<td>Bank 3</td>
<td></td>
</tr>
<tr>
<td>9000 0000h</td>
<td>Bank 4</td>
<td>4 x 64 MB</td>
</tr>
<tr>
<td>9FFF FFFFh</td>
<td>Bank 4</td>
<td></td>
</tr>
</tbody>
</table>

**Bank 1 addresses:**
- $A[27:26] = 64$MB chip select
- $A[25:0] = 64$MB chip offset

- 1 to 4 static memory-mapped devices:
  - SRAM
  - Pseudo-Static RAM
  - NOR flash

- 2 banks NAND flash

- 16-bit PC-Card devices
“N” = “negative” (active low)

NE[4:1] = NOR/PSRAM enable
- NE[1]: A[27:26]=00

NL = address latch/advance
NBL = byte lane
CLK for sync. Burst

A[25:0] = Address bus
D[15:0] = Data bus**
NOE = output enable
NEW = write enable
NWAIT = wait request

** Data bus = 8 or 16 bits
FSMC “Mode 1” memory read

Other modes:

* Provide ADV (address latch/advance)

* Activate OE and WE only in DATAST

* Multiplex A/D bits 15-0

* Allow WAIT to extend DATAST

ADDSET/DATAST programmed in chip-select timing register (HCLK = AHB clock)
FSMC “Mode 1” memory write
Flash memory issues

- Flash is programmed at system voltages.
- Erasure time is long.
- Must be erased in blocks.
- Available in NAND or NOR structures

<table>
<thead>
<tr>
<th></th>
<th>SLC NAND Flash (x8)</th>
<th>MLC NAND Flash (x8)</th>
<th>MLC NOR Flash (x16)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Density</td>
<td>512 Mbits(^1) – 4 Gbits(^2)</td>
<td>1Gbit to 16Gbit</td>
<td>16Mbit to 1Gbit</td>
</tr>
<tr>
<td>Read Speed</td>
<td>24 MB/s(^3)</td>
<td>18.6 MB/s</td>
<td>103MB/s</td>
</tr>
<tr>
<td>Write Speed</td>
<td>8.0 MB/s</td>
<td>2.4 MB/s</td>
<td>0.47 MB/s</td>
</tr>
<tr>
<td>Erase Time</td>
<td>2.0 mSec</td>
<td>2.0mSec</td>
<td>900mSec</td>
</tr>
<tr>
<td>Interface</td>
<td>I/O – indirect access</td>
<td>I/O – indirect access</td>
<td>Random access</td>
</tr>
<tr>
<td>Application</td>
<td>Program/Data mass storage</td>
<td>Program/Data mass storage</td>
<td>eXecuteInPlace</td>
</tr>
</tbody>
</table>

SLC – Single-Level Cell, MLC = Multi-Level Cell
SST39VF1601- 1M x 16 Flash (NOR)
(on uCdragon board)
SST39VF1601 characteristics

- Organized as 1M x 16
  - 2K word sectors, 32K word blocks

- Performance:
  - Read access time = 70ns or 90ns
  - Word program time = 7us
  - Sector/block erase time = 18ms
  - Chip erase time = 40ms

- Check status of write/erase operation via read
  - DQ7 = complement of written value until write complete
  - DQ7=0 during erase, DQ7=1 when erase done
### SST39VF1601 Command Sequences (assert WE# and CE# to write commands)

<table>
<thead>
<tr>
<th>Command Sequence</th>
<th>1st Bus Write Cycle</th>
<th>2nd Bus Write Cycle</th>
<th>3rd Bus Write Cycle</th>
<th>4th Bus Write Cycle</th>
<th>5th Bus Write Cycle</th>
<th>6th Bus Write Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Addr¹</td>
<td>Data²</td>
<td>Addr¹</td>
<td>Data²</td>
<td>Addr¹</td>
<td>Data²</td>
</tr>
<tr>
<td>Word-Program</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>A0H</td>
</tr>
<tr>
<td>Sector-Erase</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>80H</td>
</tr>
<tr>
<td>Block-Erase</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>80H</td>
</tr>
<tr>
<td>Chip-Erase</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>80H</td>
</tr>
<tr>
<td>Erase-Suspend</td>
<td>XXXXH</td>
<td>B0H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Erase-Resume</td>
<td>XXXXH</td>
<td>30H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Query Sec ID⁵</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>88H</td>
</tr>
<tr>
<td>User Security ID Word-Program</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>A5H</td>
</tr>
<tr>
<td>User Security ID Program Lock-Out</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>85H</td>
</tr>
<tr>
<td>Software ID Entry⁷,⁸</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>90H</td>
</tr>
<tr>
<td>CFI Query Entry</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>98H</td>
</tr>
<tr>
<td>Software ID Exit⁹,¹⁰ /CFI Exit/Sec ID Exit</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>F0H</td>
</tr>
<tr>
<td>Software ID Exit⁹,¹⁰ /CFI Exit/Sec ID Exit</td>
<td>XXH</td>
<td>F0H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SST39VF1601 read cycle timing

![Diagram of SST39VF1601 read cycle timing](image)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>SST39VFxx01/xx02-70</th>
<th>SST39VFxx01/xx02-90</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>T_{RC}</td>
<td>Read Cycle Time</td>
<td>Min: 70, Max: 90</td>
<td>Min: 70, Max: 90</td>
<td>ns</td>
</tr>
<tr>
<td>T_{CE}</td>
<td>Chip Enable Access Time</td>
<td>70</td>
<td>90</td>
<td>ns</td>
</tr>
<tr>
<td>T_{AA}</td>
<td>Address Access Time</td>
<td>70</td>
<td>90</td>
<td>ns</td>
</tr>
<tr>
<td>T_{OE}</td>
<td>Output Enable Access Time</td>
<td>35</td>
<td>45</td>
<td>ns</td>
</tr>
<tr>
<td>T_{CLZ}</td>
<td>CE# Low to Active Output</td>
<td>0</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>T_{OLZ}</td>
<td>OE# Low to Active Output</td>
<td>0</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>T_{CH-Z}</td>
<td>CE# High to High-Z Output</td>
<td>20</td>
<td>30</td>
<td>ns</td>
</tr>
<tr>
<td>T_{OH-Z}</td>
<td>OE# High to High-Z Output</td>
<td>20</td>
<td>30</td>
<td>ns</td>
</tr>
<tr>
<td>T_{OH}</td>
<td>Output Hold from Address Change</td>
<td>0</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>T_{RP}</td>
<td>RST# Pulse Width</td>
<td>500</td>
<td>500</td>
<td>ns</td>
</tr>
<tr>
<td>T_{RR}</td>
<td>RST# High before Read</td>
<td>50</td>
<td>50</td>
<td>ns</td>
</tr>
<tr>
<td>T_{RY}</td>
<td>RST# Pin Low to Read Mode</td>
<td>20</td>
<td>20</td>
<td>\mu s</td>
</tr>
</tbody>
</table>
SST39VF1601 word write

$T_{BP} = 10 \mu s$ max
2Gbit NAND flash organization

Register:
Holds 1 page

Page:
2048 + 64 bytes

Block:
64 pages

Chip:
2048 blocks
NAND flash functional block diagram

Micron: 2/4/8 Gbit, x8/x16 multiplexed NAND flash
### Micron Flash Mode Selection

<table>
<thead>
<tr>
<th>CLE</th>
<th>ALE</th>
<th>CE#</th>
<th>WE#</th>
<th>RE#</th>
<th>WP#¹</th>
<th>PRE²</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>X</td>
<td>X</td>
<td>Read mode</td>
</tr>
<tr>
<td>L</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>X</td>
<td>X</td>
<td>Command input</td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>X</td>
<td>Write mode</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>X</td>
<td>Command input</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>X</td>
<td>Address input</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>X</td>
<td>X</td>
<td>Data input</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>X</td>
<td>X</td>
<td>Sequential read and data output</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>X</td>
<td>X</td>
<td>During read (busy)</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>During program (busy)</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>H</td>
<td>X</td>
<td>During erase (busy)</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>L</td>
<td>Write protect</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>H</td>
<td>X</td>
<td>X</td>
<td>0V/0Vcc</td>
<td>Standby</td>
<td></td>
</tr>
</tbody>
</table>

Notes: 1. WP# should be biased to CMOS HIGH or LOW for standby.
2. PRE should be tied to Vcc or ground. Do not transition PRE during device operations. The PRE function is not supported on extended-temperature devices.
3. Mode selection settings for this table: H = Logic level HIGH; L = Logic level LOW; X = ViH or ViL.
Micron Flash Command Set

<table>
<thead>
<tr>
<th>Operation</th>
<th>Cycle 1</th>
<th>Cycle 2</th>
<th>Valid During Busy</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAGE READ</td>
<td>00h</td>
<td>30h</td>
<td>No</td>
</tr>
<tr>
<td>PAGE READ CACHE MODE START¹</td>
<td>31h</td>
<td>–</td>
<td>No</td>
</tr>
<tr>
<td>PAGE READ CACHE MODE START LAST¹</td>
<td>3Fh</td>
<td>–</td>
<td>No</td>
</tr>
<tr>
<td>READ for INTERNAL DATA MOVE²</td>
<td>00h</td>
<td>35h</td>
<td>No</td>
</tr>
<tr>
<td>RANDOM DATA READ³</td>
<td>05h</td>
<td>E0h</td>
<td>No</td>
</tr>
<tr>
<td>READ ID</td>
<td>90h</td>
<td>–</td>
<td>No</td>
</tr>
<tr>
<td>READ STATUS</td>
<td>70h</td>
<td>–</td>
<td>Yes</td>
</tr>
<tr>
<td>PROGRAM PAGE</td>
<td>80h</td>
<td>10h</td>
<td>No</td>
</tr>
<tr>
<td>PROGRAM PAGE CACHE¹</td>
<td>80h</td>
<td>15h</td>
<td>No</td>
</tr>
<tr>
<td>PROGRAM for INTERNAL DATA MOVE²</td>
<td>85h</td>
<td>10h</td>
<td>No</td>
</tr>
<tr>
<td>RANDOM DATA INPUT for PROGRAM ⁴</td>
<td>85h</td>
<td>–</td>
<td>No</td>
</tr>
<tr>
<td>BLOCK ERASE</td>
<td>60h</td>
<td>D0h</td>
<td>No</td>
</tr>
<tr>
<td>RESET</td>
<td>FFh</td>
<td>–</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Micron NAND Flash Page Read Operation

Cycle | I/07 | I/06 | I/05 | I/04 | I/03 | I/02 | I/01 | I/00 |
---|---|---|---|---|---|---|---|---|
First | CA7 | CA6 | CA5 | CA4 | CA3 | CA2 | CA1 | CA0 |
Second | LOW | LOW | LOW | LOW | CA11 | CA10 | CA9 | CA8 |
Third | RA19 | RA18 | RA17 | RA16 | RA15 | RA14 | RA13 | RA12 |
Fourth | RA27 | RA26 | RA25 | RA24 | RA23 | RA22 | RA21 | RA20 |
Fifth | LOW | LOW | LOW | LOW | LOW | LOW | LOW | RA28 |

Note: CAx = column address; RAx = row address.
Micron NAND Flash: Program & Erase Op’s

Program (data written to register)

Erase selected block

Data sequence

3 ms

300-700 µs
Generic DRAM device
Generic DRAM timing

- CE'
- R/W'
- RAS'
- CAS'
- Adrs
- Data

Time

Row Adrs

Col Adrs

Data
Page mode access
RAM refresh

- Value decays in approx. 1 ms.
- Refresh value by reading it.
  - Can’t access memory during refresh.
- RAS-only refresh
- CAS-before-RAS refresh.
- Hidden refresh.
Other types of memory

- Extended data out (EDO): improved page mode access.
- Synchronous DRAM: clocked access for pipelining.
  - Double Data Rate (DDR) – transfer on both edges of clock
    - DDR-1, DDR-2, DDR-3 support increasingly higher bandwidths
- Rambus: highly pipelined DRAM.
Bus mastership

- Bus master controls operations on the bus.
- CPU is default bus master.
- Other devices may request bus mastership.
  - Separate set of handshaking lines.
  - CPU can’t use bus when it is not master.
- Situations for multiple bus masters:
  - DMA data transfers
  - Multiple CPUs with shared memory
    - One CPU might be graphics/network processor
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.
CPU sets DMA registers for start address, length.

DMA status register controls the unit.

Once DMA is bus master, it transfers automatically.
  - May run continuously until complete.
  - May use every n\textsuperscript{th} bus cycle.
Bus transfer sequence diagram
System-level performance analysis

- Performance depends on all the elements of the system:
  - CPU.
  - Cache.
  - Bus.
  - Main memory.
  - I/O device.
Bandwidth as performance

- Bandwidth applies to several components:
  - Memory.
  - Bus.
  - CPU fetches.

- Different parts of the system run at different clock rates.
- Different components may have different widths (bus, memory).
Bandwidth and data transfers

- Video frame: $320 \times 240 \times 3 = 230,400$ bytes.
  - Transfer in $1/30$ sec.
- Transfer 1 byte/$\mu$sec, 0.23 sec per frame.
  - Too slow.
- Increase bandwidth:
  - Increase bus width.
  - Increase bus clock rate.
Bus bandwidth

- T: # bus cycles.
- P: time/bus cycle.
- Total time for transfer:
  - \( t = TP \).
- D: data payload length.
- \( O_1 + O_2 = \) overhead \( O \).

\[
T_{\text{basic}}(N) = (D+O)N/W
\]
Bus burst transfer bandwidth

- **T**: # bus cycles.
- **P**: time/bus cycle.
- **Total time for transfer**: \( t = TP \).
- **D**: data payload length.
- **O1 + O2 = overhead O**.

\[
T_{\text{burst}}(N) = \frac{(BD+O)N}{BW}
\]
Memory aspect ratios

64 M

16 M

8 M
Memory access times

- Memory component access times comes from chip data sheet.
  - Page modes allow faster access for successive transfers on same page.
- If data doesn’t fit naturally into physical words:
  - $A = \left\lfloor \frac{E}{w} \mod W \right\rfloor + 1$
Bus performance bottlenecks

- Transfer 320 x 240 video frame @ 30 frames/sec = 612,000 bytes/sec.
- Is performance bottleneck bus or memory?
Bus performance bottlenecks, cont’d.

- **Bus**: assume 1 MHz bus, D=1, O=3:
  \[ T_{\text{basic}} = (1+3)612,000/2 = 1,224,000 \text{ cycles} = 1.224 \text{ sec.} \]

- **Memory**: try burst mode B=4, width w=0.5.
  \[ T_{\text{mem}} = (4*1+4)612,000/(4*0.5) = 2,448,000 \text{ cycles} = 0.2448 \text{ sec.} \]
### Performance spreadsheet

<table>
<thead>
<tr>
<th>bus</th>
<th>memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>clock period</td>
<td>1.00E-06</td>
</tr>
<tr>
<td>W</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
</tr>
<tr>
<td>O</td>
<td>3</td>
</tr>
<tr>
<td>N</td>
<td>612000</td>
</tr>
<tr>
<td>T_basic</td>
<td>1224000</td>
</tr>
<tr>
<td>t</td>
<td>1.22E+00</td>
</tr>
</tbody>
</table>
Parallelism

- Speed things up by running several units at once.
- DMA provides parallelism if CPU doesn’t need the bus:
  - DMA + bus.
  - CPU.
Electrical bus design

- Bus signals are usually tri-stated.
- Address and data lines may be multiplexed.
- Every device on the bus must be able to drive the maximum bus load:
  - Bus wires.
  - Other bus devices.
  - Resistive and capacitive loads.
  - Bus specification may limit loads
- Bus may include clock signal.
  - Timing is relative to clock.
Tristate operation

<table>
<thead>
<tr>
<th></th>
<th>E2=0</th>
<th>E2=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1=0</td>
<td>float</td>
<td>D2</td>
</tr>
<tr>
<td>E1=1</td>
<td>D1</td>
<td>conflict</td>
</tr>
</tbody>
</table>

Must prevent E1=E2=1