Memory Systems for Embedded Applications

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.)
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)
Typical bus data rates

Source: Peter Cheung “Computer Architecture & Systems Course Notes”
Hierarchical Bus Architecture

CPU -> Local controller -> Cache

System

Main Memory -> Main Memory bridge

Expansion

LAN Controller -> Video Controller -> Mouse/Keyboard

USB

USB Controller -> USB Device

IDE/SCSI

Disk Controller

IDE/SCSI
ARM Advanced Microcontroller Bus Architecture (AMBA)

- **On-chip** interconnect specification for SoC
- Promote **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
    - Lower performance requirements
    - Low-power peripherals
    - Reduced interface complexity
  - **Others:**
    - **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
ARM CoreLink peripherals for AMBA

“CoreLink”
(orange blocks)

Interconnect +
memory controller
IP for Cortex/Mali
STM32L476G Microcontroller

Figure 1. STM32L476xx block diagram

- External Memory
- Quad SPI Memory
Microprocessor buses

- **Clock** provides synchronization.
- **R/W** true when reading, false when writing.
  - May replace CLK and R/W with RD and WR strobes
- **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.
- Bus protocol often defined by **timing diagrams**.
Timing diagrams

- **A**: Low to High transition, 10 ns delay
- **B**: Changing state
- **C**: Timing constraint
- **Time** as the horizontal axis
Typical bus read and write timing
Bus wait state

Extend read/write cycle if memory slower than CPU
Bus burst read

CPU sends start address, followed by burst of data from consecutive addresses
State diagrams for bus read

CPU

Get data

Yes

Ack?

No

Wait

Done


DEVICE

Ack & Send data

Yes

Ready?

No

Wait

Release ack

Adrs + CE

start

start
Read-only memory types

- **Mask-programmed ROM**
  - Programmed at factory *(high NRE cost)*

- **PROM (Programmable ROM)**
  - Programmable once by users *(low NRE cost)*
    - Electric pulses selectively applied to “fuses” or “antifuses”

- **EPROM (Erasable PROM)**
  - Repeatedly programmable/reprogrammable
    - Electric pulses for programming (seconds)
    - Ultraviolet light for erasing (15-20 minutes)

- **EEPROM (Electrically Erasable PROM)**
  - Electrically programmable and erasable at the single-byte level *(msec)*

- **Flash EPROM**
  - Electrically programmable *(µsec)* and
  - Electrically erasable *(block-by-block: msec to sec)*
  - Structures: NOR (random access); NAND (sequential access)
  - Most common program memory in embedded applications
  - Widely used in digital cameras, multimedia players, smart phones, etc.
Read-write memory types

- **Static RAM (SRAM)**
  - Each cell is a flip-flop, storing 1-bit information which is retained as long as power is on
  - Faster than DRAM
  - Requires a larger area per cell than DRAM

- **Dynamic RAM (DRAM)**
  - Each cell is a capacitor, which needs to be refreshed periodically to retain the 1-bit information
  - A refresh consists of reading followed by writing back
  - Refresh overhead
ROM/RAM device organization

Memory “organization” = $2^n \times d$
(from system designer’s perspective)

- **Size.**
  - $2^n$ addressable words
  - Address width = $n = r + c$

- **Aspect ratio.**
  - Data width $d$. 
Memory address decoding

- Select a sub-space of memory addresses
- A simple example
  - Microprocessor with 5 address bits (A₄ ... A₀) → 2⁵ = 32 bytes addressable
  - Assume 4 byte (4 x 8) memory chip → Decodes two address bits (A₁ A₀)
  - µP can address up to 8 chips (decode address bits (A₄A₃A₂) for chip enable

![Diagram showing memory address decoding](image_url)
Typical generic SRAM

CE# = chip enable: initiate memory access when active
OE# = output enable: drive Data lines when active
WE# = write enable: update SRAM contents with Data
(May have one R/W# signal instead of OE# and WE#)
Multi-byte data bus devices have a byte-enable signal for each byte.
**IS61LV51216-12T**: 512K x 16 SRAM (on uCdragon board)

- **Byte Lane Select**
  - Upper byte D15-8
  - Lower byte D7-0

- **Decoded A_{31-24}**
ISSI IS61LV51216 SRAM read cycle

**Timing Parameters:**
Max data valid times following activation of Address, CE, OE

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>-8 Min.</th>
<th>-8 Max.</th>
<th>-10 Min.</th>
<th>-10 Max.</th>
<th>-12 Min.</th>
<th>-12 Max.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>trc</td>
<td>Read Cycle Time</td>
<td>8</td>
<td>—</td>
<td>10</td>
<td>—</td>
<td>12</td>
<td>—</td>
<td>ns</td>
</tr>
<tr>
<td>tAA</td>
<td>Address Access Time</td>
<td>—</td>
<td>8</td>
<td>—</td>
<td>10</td>
<td>—</td>
<td>12</td>
<td>ns</td>
</tr>
<tr>
<td>tOHA</td>
<td>Output Hold Time</td>
<td>3</td>
<td>—</td>
<td>3</td>
<td>—</td>
<td>3</td>
<td>—</td>
<td>ns</td>
</tr>
<tr>
<td>tACE</td>
<td>CE Access Time</td>
<td>—</td>
<td>8</td>
<td>—</td>
<td>10</td>
<td>—</td>
<td>12</td>
<td>ns</td>
</tr>
<tr>
<td>tDOE</td>
<td>OE Access Time</td>
<td>—</td>
<td>3.5</td>
<td>—</td>
<td>4</td>
<td>—</td>
<td>5</td>
<td>ns</td>
</tr>
</tbody>
</table>
Control external memory on AHB bus in four 256M banks

- Upper address bits decoded by the FSMC

1 to 4 static memories:
* SRAM
* Pseudo-Static RAM
* NOR flash

Bank 1 addresses:
A[31:28] = 0110
A[27:26] = 64MB chip select
A[25:0] = 64MB chip offset

NAND flash devices

<table>
<thead>
<tr>
<th>Start address</th>
<th>End address</th>
<th>FMC bank</th>
<th>Memory space</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x8000 0000</td>
<td>0x8BFF FFFF</td>
<td>Bank 3</td>
<td>Attribute</td>
</tr>
<tr>
<td>0x8000 0000</td>
<td>0x83FF FFFF</td>
<td></td>
<td>Common</td>
</tr>
</tbody>
</table>
“N” = “negative” (active low)

\[ \text{NE}[4:1] = \text{NOR/PSRAM enable} \]
- \[ \text{NE}[1]: A[27:26]=00 \]
- \[ \text{NE}[2]: A[27:26]=01 \]
- \[ \text{NE}[3]: A[27:26]=10 \]
- \[ \text{NE}[4]: A[27:26]=11 \]

\[ \text{NL} = \text{address latch/advance} \]
\[ \text{NBL} = \text{byte lane} \]
\[ \text{CLK for sync. Burst} \]

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

\[^{**} \text{Data bus} = 8 \text{ or } 16 \text{ bits} \]
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

Programmable parameters
Flash memory devices

- Flash memory is programmed at system voltages.
- Erasure time is long.
- Must be erased in blocks.
- Available in NAND or NOR structures
  - NOR: memory cells in parallel – allows random access
  - NAND: memory cells in series – sequential access/60% smaller

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

SLC = Single-Level Cell, MLC = Multi-Level Cell
NAND and NOR flash comparision

NAND flash similar to a hard disk drive (sequential access to bits of a sector)

NOR flash similar to a Random-access memory (ROM/RAM)
SST39VF1601- 1M x 16 NOR Flash
(on uCdragon board)

Similar to SRAM connection
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 read cycle timing**

<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>TRC</td>
<td>Read Cycle Time</td>
<td>70</td>
<td>90</td>
<td>ns</td>
</tr>
<tr>
<td>TCE</td>
<td>Chip Enable Access Time</td>
<td>70</td>
<td>90</td>
<td>ns</td>
</tr>
<tr>
<td>TA#</td>
<td>Address Access Time</td>
<td>70</td>
<td>90</td>
<td>ns</td>
</tr>
<tr>
<td>TOE</td>
<td>Output Enable Access Time</td>
<td>35</td>
<td>45</td>
<td>ns</td>
</tr>
<tr>
<td>TCLZ(^1)</td>
<td>CE# Low to Active Output</td>
<td>0</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>TOLZ(^1)</td>
<td>OE# Low to Active Output</td>
<td>0</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>TCHZ(^1)</td>
<td>CE# High to High-Z Output</td>
<td>20</td>
<td>30</td>
<td>ns</td>
</tr>
<tr>
<td>TOHZ(^1)</td>
<td>OE# High to High-Z Output</td>
<td>20</td>
<td>30</td>
<td>ns</td>
</tr>
<tr>
<td>TOH(^1)</td>
<td>Output Hold from Address Change</td>
<td>0</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>TRP(^1)</td>
<td>RST# Pulse Width</td>
<td>500</td>
<td>500</td>
<td>ns</td>
</tr>
<tr>
<td>TR&lt;R#(^1)</td>
<td>RST# High before Read</td>
<td>50</td>
<td>50</td>
<td>ns</td>
</tr>
<tr>
<td>TR(^1,2)</td>
<td>RST# Pin Low to Read Mode</td>
<td>20</td>
<td>20</td>
<td>μs</td>
</tr>
</tbody>
</table>
### SST39VF1601 command sequences

Assert Address, Data, WE# and CE# to write a command

<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>Addr$^1$</td>
<td>Data$^2$</td>
<td>Addr$^1$</td>
<td>Data$^2$</td>
<td>Addr$^1$</td>
<td>Data$^2$</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 ID$^5$</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</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>A5H</td>
</tr>
<tr>
<td>Word-Program</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>85H</td>
</tr>
<tr>
<td>User Security ID</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>90H</td>
</tr>
<tr>
<td>Program Lock-Out</td>
<td>5555H</td>
<td>AAH</td>
<td>2AAAH</td>
<td>55H</td>
<td>5555H</td>
<td>98H</td>
</tr>
<tr>
<td>Software ID Entry$^7,8$</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 Exit$^9,10$ /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 Exit$^9,10$ /CFI Exit/Sec ID Exit</td>
<td>XXH</td>
<td>F0H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SST39VF1601 word program

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

System transfers data to/from the “Register”
Internal: page copied to Register

- **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

Bytes/words sent/received sequentially
## Micron Flash Mode Selection

CLE = command latch enable; ALE = address latch enable

<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>H</td>
<td>X</td>
<td>X</td>
<td>Command input</td>
</tr>
<tr>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>X</td>
<td></td>
<td>Write mode</td>
</tr>
<tr>
<td>L</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>X</td>
<td></td>
<td>Command input (write lock)</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>X</td>
<td></td>
<td>Data input</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>X</td>
<td>X</td>
<td></td>
<td>Sequential read and data output</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>X</td>
<td></td>
<td>During read (busy)</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>H</td>
<td>X</td>
<td>During program (busy)</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>H</td>
<td></td>
<td>During erase (busy)</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>L</td>
<td></td>
<td>Write protect</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>H</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>Standby</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 = VIL or VIH.
## 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 PROGRAM⁴</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

Page to register

Five address cycles

Capacity-dependent

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

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

RAS# = **Row Address Strobe**: row# on Address inputs

CAS# = **Column Address Strobe**: column# on Address inputs
Asynchronous DRAM timing

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

(row adrs) (col adrs) data
Asynchronous DRAM page mode access

- \( CE' \)
- \( R/W' \)
- \( RAS' \)
- \( CAS' \)
- \( Adrs \)
- \( Data \)

(row adrs) (col adrs) (col adrs) (col adrs)
(data) (data) (data)
SDRAM burst read (burst length 4)

$\text{Trcd} = \text{RAS-to-CAS delay}$
$\text{CL} = \text{CAS latency (CAS to data ready)}$
$\text{Tac} = \text{access time}$
Dynamic 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.

**Example: 4 Mbyte DRAM**

- Refreshed every 4 msec (one row at a time)
- Organized as 2048 rows x 2048 columns \(\Rightarrow\) 2048 refreshes
- Assume 1 refresh \(\Rightarrow\) 80 nsec

\[
\frac{2048 \times 80 \times 10^{-9}}{4 \times 10^{-3}} \approx 0.041 \quad \Rightarrow \quad 4.1\% \text{ of time spent refreshing}
\]
Other DRAM forms

- Extended data out (EDO): improved page mode access.
- Synchronous DRAM: clocked access for pipelining.
  - All operations clocked
    - Row address
    - Column address - increments on clock for each data transfer
    - Data transfer – burst transfers (one per clock) after initial latency
- Double Data Rate (DDR) – transfer on both edges of clock
  - Effectively doubles the bandwidth
  - DDR-2: doubles the clock rate of DDR
  - DDR-3, DDR-4 support increasingly higher bandwidths
- Rambus: highly pipelined DRAM.
DDR2 bank activate

Memory partitioned into 8 separate arrays called “banks”
Bank Activate command = CS# low, RAS# low, CAS# high, WE# high (and CKE high)
  - Bank address BA2-BA0 selects bank
  - Row address A15-A0 selects a row in the bank
Follow with read/write command in next clock cycle
Concurrent Bank Activate commands permitted (up to 8)
DDR2 burst read (burst length 4)

Burst read command = CS# low, CAS# low, RAS# high, WE# high (and CKE high)
Read Latency \( RL = AL + CL \)
- \( CL \) (programmable) = CAS latency (CAS to data ready)
- \( AL \) (programmable) = “Additive” Latency
Systems with multiple bus masters

- **Bus master** controls operations on the bus.

- CPU is default bus master.

- Other devices may request bus mastership.
  - Request mastership via separate handshaking lines.
  - Main CPU can’t use bus when it is not master.

- Situations for multiple bus masters:
  - **DMA** data transfers
  - Multiple CPUs/Cores with shared memory
  - Separate graphics/network processor
Direct Memory Access (DMA)

- DMA data transfers done without executing CPU instructions.
  - CPU sets up transfer.
  - DMA engine fetches, writes.
- DMA controller is a separate unit.
DMA operation

- CPU sets DMA registers for start address, length.
- DMA status register controls the unit.
  - Bus request to CPU – Bus grant back from CPU
- DMA controller requests bus mastership from CPU
- Once DMA is bus master, it transfers automatically.
  - May run continuously until complete.
  - May use every $n^{th}$ 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.
- Components may have different widths (bus, memory).
Bandwidth and data transfers

- Video frame: 320 x 240 x 3 = 230,400 bytes.
  - Need to transfer in 1/30 sec = 0.033 sec
- Transfer 1 byte/μsec, 0.23 sec per frame.
  - Too slow.

To increase bandwidth:
- Increase bus width.
- Increase bus clock rate.
- Minimize overhead (do burst transfers)
Bus bandwidth

- **T**: # bus cycles.
- **P**: bus clock period.
- **Total time for transfer**:
  - $t = TP$.

- **D**: data payload length.
- **O** = **O1** + **O2** = overhead.
  - (before & after data)
- **N**: total # data payloads.
- **W**: bus width (bits/xfer)

$$T_{basic}(N) = (D+O)N/W$$

Transfer ND bits
Bus burst transfer bandwidth

- T: # bus cycles.
- P: time/bus cycle.
- Total time for transfer:
  - \( t = TP \).
- D: data payload length.
- B: burst size
  - (#transfers of size D)
- O1 + O2 = overhead O.
- \( N = \) total # data payloads

\[
T_{\text{burst}}(N) = \frac{(BD+O)\times N}{BW}
\]
Bus performance bottlenecks

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

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

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

Memory chip formats

64 M

16 M

8 M

1

4

8
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