Computer Memory

Textbook: Chapter 1

ARM Cortex-M4 User Guide (Section 2.2 – Memory Model)
STM32F4xx Technical Reference Manual:
  Chapter 2  – Memory and Bus Architecture
  Chapter 3  – Flash Memory
  Chapter 36 – Flexible Static Memory Controller
Computer Memory Systems

- Memory system hierarchy
  - Disk, ROM, RAM, Cache
- Memory module (chip) organization
  - On-chip (address) decoder, cell array
- Memory system interfacing
  - Address decoding
  - Bus timing
- Direct memory access (DMA)
  - Transfer data directly between memory and I/O devices
  - Coordinated by a DMA controller
Computer Memory Hierarchy

Memory Content: $M_C \subseteq M_M \subseteq M_D$

Memory Parameters:
- **Access Time**: increase with distance from CPU
- **Cost/Bit**: decrease with distance from CPU
- **Capacity**: increase with distance from CPU
Semiconductor Memory

- **RAM (Random Access Memory)**
  - Constant access time, independent of location
  - A unique address for each location (generally a byte)
  - The address is decoded by one or more address decoders

- **RAM (Read/Write Memory) vs. ROM (Read Only Memory)**
  - **RAM**
    - User’s application programs and data
    - Information is lost when the power is off
  - **ROM**
    - Embedded system program code and operating system
    - Information is retained even without power
    - Each ROM cell is simpler than a RAM cell
Read-only memory types

- **Mask-programmed ROM**
  - Programmed at factory
- **PROM (Programmable ROM)**
  - Programmable once by users
  - Electric pulses selectively applied to “fuses”
- **EPROM (Erasable PROM)**
  - Repeatedly programmable/reprogrammable
  - Electric pulses for programming (seconds)
  - Ultraviolet light for erasing (minutes)
- **EEPROM (Electrically Erasable PROM)**
  - Electrically erasable at the single-byte level (msec) & programmable
- **Flash EPROM**
  - Electrically programmable (μsec) & erasable (block-by-block: msec~sec)
  - Most common program memory in embedded applications
  - Widely used in digital cameras, multimedia players, smart phones, etc.

ROM devices are “non-volatile” – they retain information, even when not powered.
Read-write memory types

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

4 Mbyte DRAM: Refreshed every 4 msec
Organized as 2048 rows x 2048 columns \(\rightarrow\) 2048 refreshes
1 refresh \(\rightarrow\) 80 nsec
\[
\frac{2048 \times 80 \times 10^{-9}}{4 \times 10^{-3}} \approx 0.041 \rightarrow 4.1\% \text{ of time spent refreshing}
\]
Memory organization (RAM)

- RAM Structure
  - Memory Cell
    - A byte consists of 8 memory cells, with common control signals, \textit{Select} and \textit{R/W}, and 8 bidirectional data lines.
    - Some RAMs have separate Din and Dout
    - With \( n \)-bit address, the memory system can contain up to \( 2^n \) bytes.
      - An \( n \)-bit address is decoded by one or more address decoders to generate the control signal, \textit{Select}. 

ROM/RAM device organization

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

- Aspect ratio.
  - Data width $d$.

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

- Selecting a sub-space of memory address
- A simple example
  - Microprocessor with 5 address bits \((A_4 \ldots A_0)\) \(\rightarrow\) \(2^5 = 32\) bytes addressable
  - Memory chip: \(4 \times 8\) (4 bytes) \(\rightarrow\) Decodes two address bits \((A_1 A_0)\)
  - Can address up to 8 chips (decode address bits \((A_4 A_3 A_2)\) for chip enable

![Diagram of address decoding](https://via.placeholder.com/150)
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.
Generic SRAM timing

- **CE’**
- **OE’**
- **WE’**
- **Adrs**
- **Data**

**Read**
- Chip enabled
- **Read Address**
- From SRAM captures Data

**Write**
- Chip enabled
- **Write Address**
- From CPU captures Data

**From CPU**
- CPU captures Data

**From SRAM**
- SRAM captures Data
Microprocessor 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.)
- 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.

** Instead of R/W’ some CPUs have separate RD’ and WR’
Timing diagrams

A

B

C

10 ns

High

Changing

Stable

Timing constraint

Rising

Falling

Time
Typical bus read and write timing

From memory to CPU
From CPU to memory
Bus wait state

Insert additional clock cycle(s) before completing read/write.
IS61LV51216-12T: 512K x 16 SRAM

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

Decoded A31-24
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</th>
<th>-10</th>
<th>-12</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min.</td>
<td>Max.</td>
<td>Min.</td>
<td>Max.</td>
</tr>
<tr>
<td>trc</td>
<td>8</td>
<td>—</td>
<td>10</td>
<td>—</td>
</tr>
<tr>
<td>tAA</td>
<td>—</td>
<td>8</td>
<td>—</td>
<td>10</td>
</tr>
<tr>
<td>toHA</td>
<td>3</td>
<td>—</td>
<td>3</td>
<td>—</td>
</tr>
<tr>
<td>tACE</td>
<td>—</td>
<td>8</td>
<td>—</td>
<td>10</td>
</tr>
<tr>
<td>tDOE</td>
<td>—</td>
<td>3.5</td>
<td>—</td>
<td>4</td>
</tr>
</tbody>
</table>
Flash memory devices

- Available in NAND or NOR structures
  - NOR flash system interface similar to SRAM (random access)
  - NAND flash system interface typically “serial” (indirect access)
- Read operations are the default, and similar to other memory devices
- Writing/erasing is initiated by writing “commands” to the Flash memory controller
  - Flash is programmed at system voltages.
  - Erasure time is long, and must be erased in blocks.

<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>
Ex: SST39VF1601- 1M x 16 NOR Flash

SST39VF3201=2M x 16 (4Mbyte: $2^{22}$) / SST39VF3201=4M x 16 (8Mbyte: $2^{23}$)

- Byte lane selects NBL[1:0] not used: all operations are “words”
- SST39VF3201 uses A[21..1], SST39VF6401 uses A[22..1]
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>
Generic DRAM device

RAS’ = Row Address Strobe: row# on Address inputs
CAS’ = Column Address Strobe: column# on Address inputs
Generic DRAM timing

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

row adrs

col adrs
data

time
Bus burst read (if supported)

![Diagram showing the timing of bus burst read with labels for Clock, R/W, Burst, Address enable, Address, Data ready, and Data over time.](image-url)
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.
Other DRAM forms

- 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.
Cortex-M4 memory map

Add external memory in this address range

On-chip
STM32F407 Microcontroller

AHB 168MHz

APB 142MHz

APB 84MHz

External Bus
STM32 Flexible Static Memory Controller (FSMC)

- Control external memory on AHB bus in 4 - 256K banks
- Upper address bits decoded by the FSMC

Static memory-mapped devices:
* SRAM
* Pseudo-Static RAM
* NOR flash

2 banks NAND flash

16-bit PC-Card devices

Bank 1 addresses:
A[31:28] = 0110
A[27:26] = 64MB bank select
A[25:0]  = 64MB bank offset
Example SRAM address decoding

SRAM/NE4 Addresses: [ 0x6C00 0000 … 0x6F00 0000]

4Gbyte address space
4-to-16 decoder

Ext SRAM bank
2-to-4 decoder

2^{26} = 64 Mbytes

SRAM/NE4 Addresses: [ 0x6C00 0000 … 0x6F00 0000]
"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
NWE = 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)
Example: 512K x 16 SRAM (1 Mbyte)

D[15..0]  
A[19..1]

1Mbyte ($2^{20}$) used of this 64Mbyte ($2^{26}$) address space for NEx

Therefore, 6 address bits not decoded: A[25..20]

A[0] is part of NBL[1:0]

Microcontroller decodes upper address bits – ADDR[31..26] – for NEx
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)
Typical bus data rates

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

[Diagram showing ARM System components: High Bandwidth External Memory Interface, High Performance ARM processor, APB Bridge, UART, Timer, Keypad, PIO, AHB, High-bandwidth on-chip RAM, DMA Bus Master, High Performance, Pipelined, Burst Support, Multiple Bus Masters, Low Power, Non-pipelined, Simple Interface]
DMA

- Direct memory access (DMA) performs data transfers without executing instructions.
  - CPU sets up transfer by programming the DMA controller.
  - DMA engine fetches, writes.
- DMA controller is a separate unit – can become bus master.
Bus mastership

- Bus master controls operations on the bus.
- By default, CPU is bus master and initiates transfers.
- Other devices may request bus mastership.
  - Separate set of handshaking lines.
  - CPU can’t use bus when it is not master.
- Bus mastership protocol:
  - Bus request – a device requests bus mastership from CPU
  - Bus grant – CPU relinquishes and grants mastership to device
- Situations for multiple bus masters:
  - DMA data transfers
  - Multiple CPUs with shared memory
    - One CPU might be graphics/network processor
DMA operation

- CPU configures DMA controller registers for:
  - peripheral address, memory start address, #xfers, direction (P->M or M->P)
- Peripheral issues DMA request to DMA controller.
- DMA controller takes bus mastership from CPU
- Once DMA is bus master, it transfers automatically.
  - Memory address incremented and count decremented for each transfer.
  - May run continuously until complete.
  - May use every n\textsuperscript{th} bus cycle.