Overview of Chapter 6

- Memory generators
- Random Access Memory
  - Function
  - Operation
  - Timing
- RAM integrated circuits
  - RAM Cell
  - RAM Bit Slice
  - 3-State Buffers
  - Cell Array and Coincident Selection
  - Dynamic RAM
- Array of RAM integrated circuits
  - Arrays of Static and Dynamic RAM

Memory Definitions

- Memory — A collection of storage cells together with the necessary circuits to transfer information to and from them
- Memory Organization — the basic architectural structure of a memory in terms of how data is accessed
- Random Access Memory (RAM) — a memory organized such that data can be transferred to or from any cell (or collection of cells) in a time that is not dependent upon the particular cell selected
- Memory Address — A collection of binary digits that identify a particular memory element (or collection of elements)

Memory Definitions ...Contd

- Typical data elements are:
  - bit — a single binary digit
  - byte — a collection of eight bits accessed together
  - word — a collection of binary bits whose size is a typical unit of access for the memory. It is typically a power of two multiple of bytes (e.g., 1 byte, 2 bytes, 4 bytes, 8 bytes, etc.)
- Memory Data — a bit or a collection of bits to be stored into or accessed from memory cells
- Memory Operations — operations on memory data supported by the memory unit. Typically, read and write operations over some sized data element (bit, byte, word, etc.)

Memory Organization

- Organized as an indexed array of words. Value of the index for each word is the memory address
- Memory is organized to fit the needs of a particular computer architecture.
- Some historically significant computer architectures and their associated memory organization:
  - Digital Equipment Corporation PDP-8 – used a 12-bit address to address 4096 12-bit words
  - IBM 360 – used a 24-bit address to address 16,777,216 8-bit bytes, or 4,194,304 32-bit words
  - Intel 8080 – (8-bit predecessor to the 8086 and the current Intel processors) used a 16-bit address to address 65,536 8-bit bytes

Memory Block Diagram

- A basic memory system is shown here:
- K Address Lines are decoded to address $2^k$ Words of memory
- Each Word is N bits
- Read and Write are single control lines defining the simplest of memory operations

Memory Organization (Example)

Example memory contents:
- A memory of 3 address bits, 8 data bits will have:
  - $k = 3$ and $N = 8$ so $2^k = 8$ Addresses labeled 0 to 7
  - $2^k = 8$ Words of 8-bit data

<table>
<thead>
<tr>
<th>Memory Address Binary</th>
<th>Memory Address Decimal</th>
<th>Memory Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>000 0</td>
<td>0</td>
<td>11111111</td>
</tr>
<tr>
<td>001 1</td>
<td>1</td>
<td>10111111</td>
</tr>
<tr>
<td>010 2</td>
<td>2</td>
<td>10110001</td>
</tr>
<tr>
<td>011 3</td>
<td>3</td>
<td>00000000</td>
</tr>
<tr>
<td>100 4</td>
<td>4</td>
<td>10111101</td>
</tr>
<tr>
<td>101 5</td>
<td>5</td>
<td>10000110</td>
</tr>
<tr>
<td>110 6</td>
<td>6</td>
<td>00110011</td>
</tr>
<tr>
<td>111 7</td>
<td>7</td>
<td>11001100</td>
</tr>
</tbody>
</table>
Basic Memory Operations

- Memory operations require the following:
  - **Data** — data written to, or read from, memory as required by the operation.
  - **Address** — used to specify a range of indices the memory is to operate on. The address lines carry this information to the memory. Typically: N bits specify locations of 2^N words.
  - **An operation** — Information sent to the memory and interpreted as control information which specifies the type of operation to be performed. Typical operations are READ DATA, and WRITE DATA. Others are READ followed by WRITE and a variety of new operations associated with delivering block of data.

Read and Write operations

- **Steps for a “Read” operation**
  - Apply binary address of the desired word to the address lines
  - Activate the “read” input
- **Steps for a “Write” operation**
  - Apply the binary address of the desired word to the address lines
  - Apply the data bits that must be stored in the memory to the data input lines
  - Active the “write” input

Memory Operation Timing

- The most basic memories are asynchronous
  - Storage is performed by latches or storage of electrical charge
  - Do not use a clock
  - Controlled by application of control inputs and address
  - Timing of signal application is critical to the operation
  - See Figure 6-4 in text
- **Control Signals**:
  - Relative timing of signals for Write and Read

Memory Operation Timing…Contd

- **Access time**:
  - is related to memory read operation
  - is the maximum time from the application of address (on address lines) to the appearance of data (on data output lines)
- **Write cycle time**:
  - is the maximum time from the application of the address to the completion of all internal memory operations required to store a word
  - The CPU must provide control signals at a rate that does not exceed the access frequency or write cycle frequency

Random Access Memory (RAM)

- **RAM is used for holding data and programs being executed.**
- **RAM differs from ROM (Read Only Memory) in that it can be “Read” and “Written”.**
- **Some facts about RAM**
  - RAM is volatile → contents are lost when power is turned off.
  - RAM is also called Read Write Memory (RWM)
  - RAM is much faster than ROM
- **Question:** Is Random Access Memory completely “Random”? What does “Random” refer to?
Types of Memory

- Types of memory
  - SRAM (Static Random Access Memory)
    - Consists of latches to store information
  - DRAM (Dynamic Random Access Memory)
    - Consists of a transistor and capacitor
  - FRAM (Fast Random Access Memory)
    - Same as SRAM but more compact and fast
  - MRAM (Magneto resistive Random Access Memory)
    - Advanced RAMs of the future. Use magnetic films to store data
  - RAM integrated circuits are built using arrays of 1-bit RAM cells

Static RAM (SRAM) features

- SRAM consist of latches to store information
- SRAM holds data without external refresh
  - It contrasts with DRAM which requires refresh to retain data
- SRAMs are implemented with latches and hence are faster than DRAMs
- SRAMs require large amount of logic for implementation and hence are costly

Static RAM Cell

- Array of storage cells used to implement static RAM
- Each storage cell consists of:
  - A latch
  - Cell write logic
  - Cell read logic
  - A logical representation of electronic circuitry
  - SR Latch for storage
  - Select input for control
  - Dual Rail Data Inputs B and \( \overline{B} \)
  - Dual Rail Data Outputs C and \( \overline{C} \)

Static RAM Bit Slice

- Represents all of the circuitry that is required to store multiple 1-bit words
- See Figure 6-6 in text as an example
- Multiple RAM cells
- Control Lines:
  - Word select \( i \) – one for each word
  - Bit Select
- Data Lines:
  - Data in
  - Data out

Dynamic RAM (DRAM) features

- DRAM is implemented with transistors and capacitors
- The capacitor holds logic 1 or 0 and the transistor performs switching action
- Capacitor is a discharging element so the logic level on the capacitor needs to be "refreshed"
- DRAM ICs consist of circuitry that refresh contents of every location hundreds of times per second otherwise it loses its memory
- DRAM uses very less area since it consists of a transistor and a capacitor
- DRAM is slower than SRAM because of capacitor charging and discharging

Dynamic RAM cell

- Implemented with a MOS transistor and a capacitor.
- If Row Select = 1, then the logic level on Digit In/Out Line is stored in the capacitor
- The contents of the capacitor are refreshed by reading its contents and loading them back into it.
- How many times should refresh be performed to retain the data?
Comparison between SRAM and DRAM

<table>
<thead>
<tr>
<th></th>
<th>SRAM</th>
<th>DRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implemented with latches (6-Transistor / 4-Transistor)</td>
<td>Capacitor based storage.</td>
<td></td>
</tr>
<tr>
<td>Require more area due to more logic</td>
<td>Requires very small area</td>
<td></td>
</tr>
<tr>
<td>Fast data access</td>
<td>Slow data access than SRAM</td>
<td></td>
</tr>
<tr>
<td>Low power consumption</td>
<td>Consume large amount of power due to periodic “refresh”</td>
<td>Used widely due to its high density and level of integration</td>
</tr>
</tbody>
</table>

- For more information on different memory types refer to the eRAMs presentation on course webpage

RAM Integrated Circuits

- RAM IC’s are specified by
  - number of words and
  - number of bits in each word
- Example: A 1K x 16 RAM is a memory with a capacity of 1K words of 16bits each
  - Total memory size is 1K( = 2^10 bits) x 16(=2^4 bits) = 2^14 bits.
  - What is the size of the memory expressed in KB
    - 2^10/2^10 = 2^4 K bits = 2 K bytes (since 2^4 bits = 1 byte)

n-Word x 1-Bit RAM IC

- To build a RAM IC from a RAM slice, we need:
  - A decoder decodes the log2n address lines to n word select lines
  - A 3-state buffer on the data output permits RAM ICs to be combined into a RAM with c x n words
- See Figure 6-7 in text as an example
  - Add 4-to 16 decoder with address inputs and word select outputs
  - Add 3-state buffer controlled by chip select

3-state Buffers and Logic

- Three-State Logic – Sometimes called tri-state logic but tri-state is a registered trademark of National Semiconductor – has three states for logic levels:
  - Active high state (output is driven high)
  - Active low state (output is driven low)
  - High impedance (Hi-Z) state (output is not driven)
- Especially useful for replacing "open collector" wire-OR or wire-AND oriented busses.
- Often registers made of latches and flip-flops have three-state outputs.
- Commonly used for memory components

3-State Buffer Basics

- The basic 3-State Buffer is shown:
  - Hi-Z: Means the output is not "driven"
  - The output “floats” to some level which is usually neither the "1" nor "0" in this state.
  - When EN = "1", the output follows the input.
  - The buffer can also invert data
  - EN can be an inverted signal

3-State Logic Basics…Contd

- Making Multiplexers with Tri-state Devices:

<table>
<thead>
<tr>
<th>EN</th>
<th>EN</th>
<th>IN</th>
<th>IN</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>X</td>
<td>X</td>
<td>Hi-Z</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>X</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>damage</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>damage</td>
</tr>
</tbody>
</table>

- 3-State Logic Basics…Contd
Cell Arrays and Coincident Selection

- Memory arrays can be very large
  - Large decoders
  - Large fanouts for the bit lines
  - The decoder size and fanouts can be reduced to approximately \( \sqrt{n} \) by using a coincident selection in a 2-dimensional array
    - Uses two decoders, one for words and one for bits
    - Word select becomes Row select
    - Bit select becomes Column select
- See Figure 6-10 for example
  - \( A_2 \) and \( A_1 \) used for Row select
  - \( A_0 \) and \( A_1 \) for Column select

RAM ICs with > 1 Bit/Word

- Word length can be quite high.
- To better balance the number of words and word length, use ICs with > 1 bit/word
- See Figure 6-11 for example
  - 2 Data input bits
  - 2 Data output bits
  - Row select selects 4 rows
  - Column select selects 2 pairs of columns