Memories

Lecture 8

Memory

- The register in a digital computer may be classified as either operational or storage type
  - An operational register is capable of storing binary information in its flips flops and in addition has combinational gates capable of data-processing tasks
  - A storage register is used solely for temporary storage of binary information
- A memory unit is a collection of storage registers together with the associated circuits needed to transfer information in and out of the registers
  - The storage registers in a memory unit are called memory registers

Semiconductor Memory Fundamentals

- In the design of all computers, semiconductor memories are used as primary storage for code and data
  - Semiconductor memories are connected directly to the CPU
  - they are the memory that the CPU first asks for information (code and data)
- For this reason, semiconductor memories are sometimes referred to as primary memory
  - The main requirement of primary memory is that it must be fast in responding to the CPU

Types of Memory

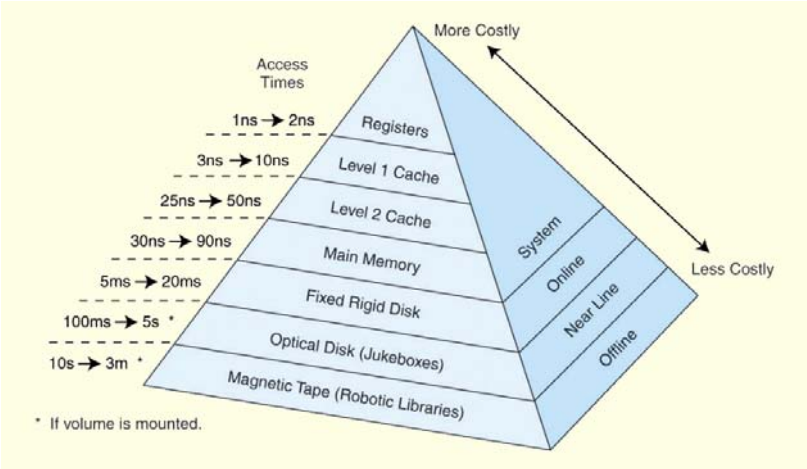
- There are two kinds of main memory:
  - *Random Access Memory, RAM, and* Read-only-*Memory, ROM*.
- There are two types of RAM:
  - Dynamic RAM (DRAM) and Static RAM (SRAM).
    - Dynamic RAM consists of capacitors that slowly leak their charge over time. Thus they must be refreshed every few milliseconds to prevent data loss.
    - DRAM is “cheap” memory owing to its simple design.
  - SRAM consists of circuits similar to the D flip-flop
    - SRAM is very fast memory and it doesn’t need to be refreshed like DRAM does. It is used to build cache memory, which we will discuss in detail later.

The Memory Hierarchy

- ROM also does not need to be refreshed, either.
  - In fact, it needs very little charge to retain its memory.
  - ROM is used to store permanent, or semi-permanent data that persists even while the system is turned off.
- Generally speaking, faster memory is more expensive than slower memory.
  - To provide the best performance at the lowest cost, memory is organized in a hierarchical fashion.
    - Small, fast storage elements are kept in the CPU, larger, slower main memory is accessed through the data bus.
    - Larger, (almost) permanent storage in the form of disk and tape drives is still further from the CPU.

The Memory Hierarchy

- This storage organization can be thought of as a pyramid:



Memory Organization

- Computer memory consists of a linear array of addressable storage cells that are similar to registers.
  - Memory can be byte-addressable, or word-addressable,
    - where a word typically consists of two or more bytes.
  - Memory is constructed of RAM chips,
    - often referred to in terms of length × width.
- If the memory word size is 16 bits, then a 4M × 16 RAM chip gives us 4 megabytes of 16-bit memory locations.

Memory organization

- Total number of bits that a memory chip can store is equal to the number of locations times the number of data bits per location.
- To summarize:

1. Each memory chip contains 2^x locations, where *x is* the number of address pins on the chip.
2. Each location contains *y* bits, where *y* is the number of data pins on the chip.
3. The entire chip will contain $2^x \times y$ bits, where *x* is the number of address pins a*nd y* is the number of data pins on the chip.

Speed

- One of the most important characteristics of a memory chip is the speed at which data can be accessed from it
  - To access the data, the address is presented to the address pins, and after a **certain amount** of time the data appears at the data pins
- The **shorter** this elapsed time, the **better**, and the more expensive the memory chip
- The speed of the memory chip is commonly referred to as its **access time**
  - The access time of memory chips varies from a few nanoseconds to hundreds of nanoseconds

Example

- A given memory chip has 12 address pins and 4 data pins. Find
  - (a) the organization
  - (b) the capacity
- Solution:
  a. This memory chip has 4096 locations (2^{12} = 4096), and each location can hold 4 bits of data. This gives an organization of 4096 x 4, often represented as 4Kx4
  b. The capacity is equal to 16K bits since there is a total of 4K locations and each location can hold 4 bits of data.

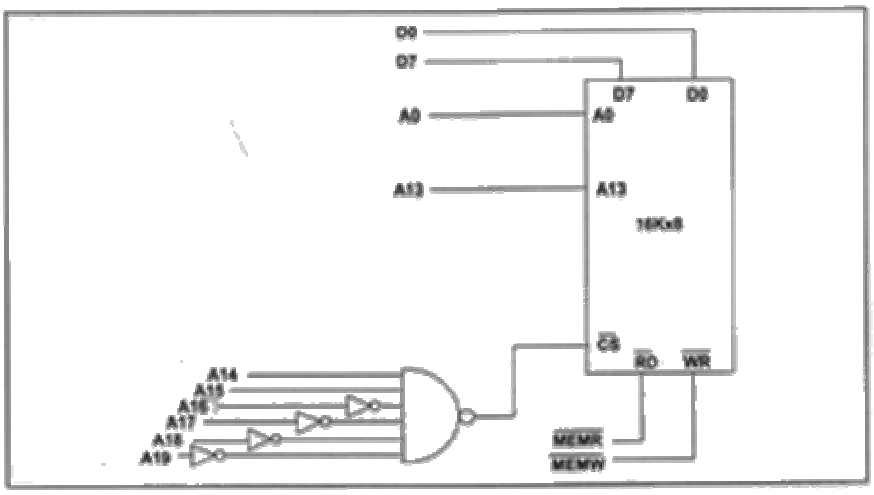
Methods of address decoding

- CPU provides the address of the data, but it is the job of the decoding circuitry to locate the data using the address bits provided by the CPU.
- Memory chips have one or more pins called CS (chip select),
  - which must be activated for the memory's contents to be accessed.
  - Sometimes the chip select is also referred to as chip enable (CE).
- In connecting a memory chip to the CPU, the data bus is connected directly to the data pins of the memory.

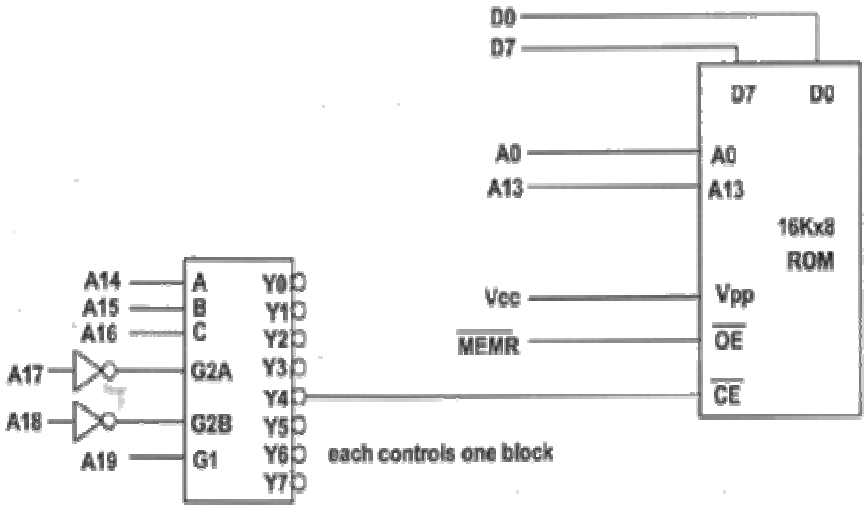
Simple logic gate as address decoder

- The simplest method of decoding circuitry is the use of NAND or other gates.
  - The fact that the output of the NAND gate is active low and that CS is also active low makes them a perfect match.
- In cases where the CS input is active high, an AND gate must be used.
- Using a combination of NAND and inverters, one can decode any address range.

Simple logic gate as address decoder



Example



Using the IC as Decoder

- one of the most widely used address decoders.
  - The 3 inputs A, B, and C generate 8 active-low outputs Y0 - Y7.
  - Each Y output is connected to CS of a memory chip, allowing control of 8 memory blocks by a single 74138.
- A, B, and C select which output is activated,
- Three additional inputs, G2A, G2B, and G1. G2A and G2B are both active low, and G1 is active high.
- If any one of the inputs G1, G2A, or G2B is not connected to an address signal (sometimes they are connected to a control signal), they must be activated permanently either by Vcc or ground, depending on the activation level.

Example

- Looking at the design in Figure, calculate the address range for Y4,
- Solution:
- The address range for Y4 is calculated as follows.

A19	A18	A17	A16	A15	A1	A13	A12	All	A10	A9	A8	A7	A6	A5	A4	A3	A2	Al	A0
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- This shows that the range for Y4 is F0000H to F3FFFH.
- In Figure notice that A19, A18, and A17 must be 1 for the decoder to be activated.
- Y4 will be selected when A16A15A14=100(4in binary). The remaining A13 - AO will be 0 for the lowest address and 1 for the highest address

Memory Organization

- How many address lines are required to access a memory location corresponds to a 4 M word address?
  - A 4M can be expressed as $2^2 \times 2^{20} = 2^{22}$ words.
    - The memory locations for this memory are numbered 0 through $2^{22}-1$.
    - The memory bus of this system requires at least 22 address lines.
- Example
  - Build 32K x 16 memory with 2K x 8 RAM chips

Memory Organization

- Build 32K x 16 memory with 2K x 8 RAM chips
  - Connect 16 rows and 2 columns of chips together
- Each row addresses 2k words
  - It requires two chips to handle the full width
- Address of this memory must be of 15 bits
  - For 32 k = $2^5 \times 2^{10}$
- But each chip pair (row) requires only 11 address lines 2^{11}
- So a decoder will be require to decode the left most 4 bits of the address to determine which chip is holding the desire address
- Once the proper chip has been identifies
  - Remaining 11 bits will be input to another decoder to find the desired address within that chip

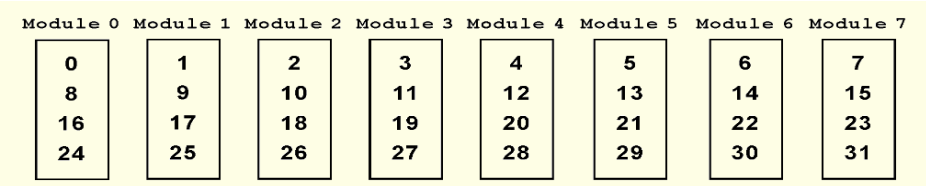
2k x 8	2k x8	Row 0
2k x 8	2k x8	Row 1
2k x 8	2k x8	Row15

Memory interleaving

- A single shared memory module causes sequentialization of access
- Memory interleaving splits memory across multiple memory modules or banks
  - Access is more efficient when memory is organized into banks of chips with the addresses interleaved across the chips
    - With low-order interleaving,
      - the **low order** bits of the address specify which memory bank contains the address of interest
      - Consecutive words of memory in different module .
    - With high-order interleaving,
      - the **high order** address bits specify the memory bank
      - Each module places consecutive addresses

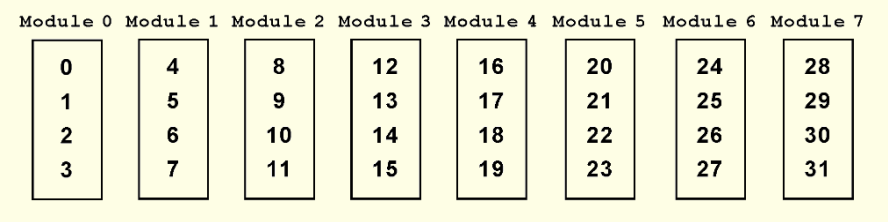
Memory Organization

- Physical memory usually consists of more than one RAM chip.
  - Access is more efficient when memory is organized into banks of chips with the addresses interleaved across the chips
- With low-order interleaving, the low order bits of the address specify which memory bank contains the address of interest.



Memory Organization

- With high-order interleaving, the high order address bits specify the memory bank.



Memory Organization and Addressing

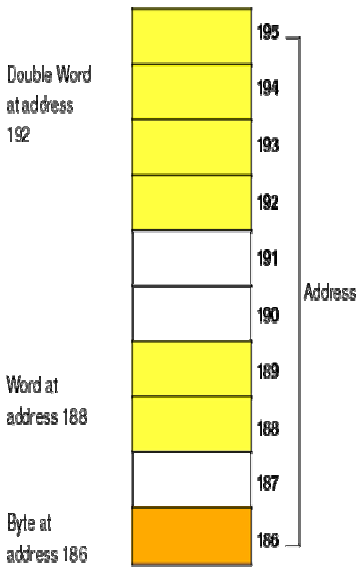
- Normally the memory is byte-addressable,
  - Some times it is word-addressable, where a word typically consists of two or more bytes.
    - A computer might handle 32 bit words, but still employ byte addressable architecture
      - In such cases the where word is of multiple bytes the byte at the lowest address determine the address of entire word
    - One can read or write a word even on an byte addressable machine
- We could have:
  - **byte-addressable** memory (each address stores eight bits) or
  - **word-addressable** memory (each address stores a word).
- The most common use of memory is to have one byte at each address.

Memory Organization

- Each byte / word has a unique address
  - Most current machines are byte addressable
- What if the machine is byte addressable and how to address a word of 16, 32 bits
  - The word is a basic unit of size used in instructions
- If machine architecture is of byte addressable, and the instruction set is larger than 1 byte
  - Issue of alignment must be addressed
    - To read a 32 bit word from a byte addressable machine ensure that:
      - The word was stored on natural alignment boundary
      - The access starts on that boundary
    - This can be achieved by requiring the address to be multiple of 4

Word, and DWord Storage in Memory

- It is quite possible for byte, word, and double word values to overlap in memory.
  - For example, you could have a word variable beginning at address 193, a byte variable at address 194, and a double word value beginning at address 192. These variables would all overlap.
- A processor with an eight-bit bus (like the old 8088 CPU) can transfer eight bits of data at a time.
  - Since each memory address corresponds to a byte, this turns out to be the most convenient arrangement (from the hardware perspective)

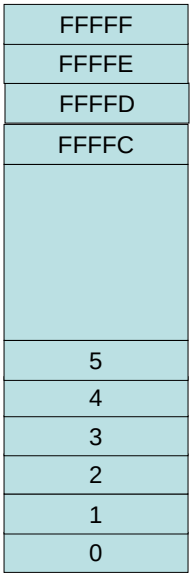


Important

- CPUs with an eight-bit bus can manipulate word and double word values, even through their data bus is only eight bits wide.
  - However, this requires multiple memory operations because these processors can only move eight bits of data at once.
    - To load a word requires two memory operations;
    - To load a double word requires four memory operations.
- If a 32 bit is to be read on a byte addressable machine, make sure:
  - The word was stored on a natural alignment boundary
  - The access starts on that boundary
    - This will achieve for 32 bit word, by requiring the address to be multiple of 4

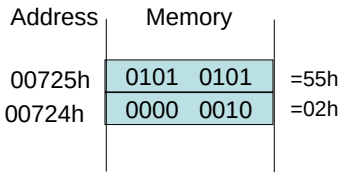
Memory address space

- To access any two consecutive bytes as a word of data
  - The lower addressed byte is the least significant byte of the word
  - The higher addressed byte is the most significant byte of the word



Storing a word of data

- To permit efficient use of memory
  - The word of data (16 bits) can be stored at even or odd addressed word boundaries
  - The least significant bit of the address determine the type of word boundary
    - If the bit is 0, the word is at an even address boundary corresponds to two consecutive byte located at even address



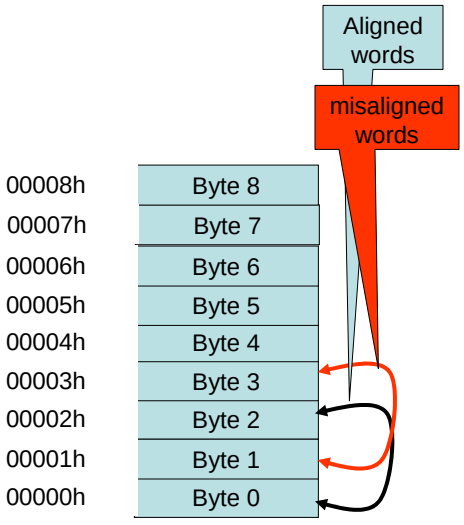
Collectively represents the word =5502h or

= **(01010101**00000010**)2**

The Least Significant byte of the word is stored at **00724**h, so it an even address boundary

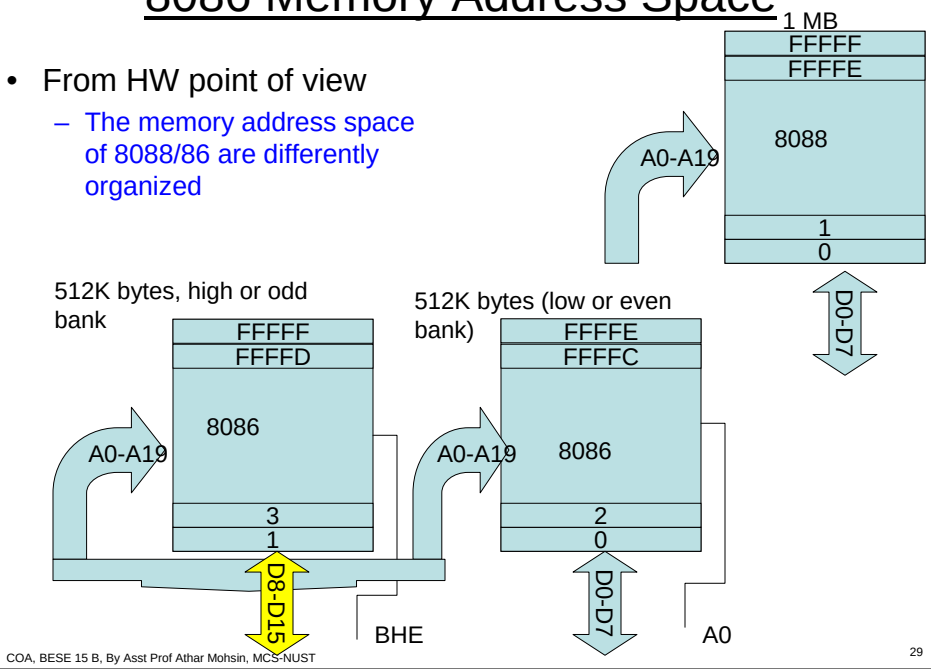
Aligned and misaligned words

- A word of data stored at an even address boundary like:
  - 00000, 00002,00004
  - said to be aligned words, located at an address that is multiple of 2
- A word of data stored at an odd address boundary like:
  - 00001, 00003, 00005
  - said to be misaligned words



8086 Memory Address Space

- From HW point of view
  - The memory address space of 8088/86 are differently organized



Low and high banks

- For 8086 address bits A1 to A19 selects the storage location that is to be accessed
  - These are applied to both the banks in parallel
- A0 and BHE are used as bank select signals
  - A0 = 0
    - identifies an even addressed byte of data and low bank of memory will be enabled
  - BHE = 0
    - enables the high bank and an odd addressed byte of data
  - Both the banks transfer byte size data