تركيب الحاسوب

Ghufran Saady

Computer Organization and Architecture

Computer

- technology has made incredible improvement in the past half century. In the early part of computer evolution, there were no stored-program computer, the computational power was less and on the top of it the size of the computer was a very huge one.
- Today, a personal computer has more computational power, more main memory , more disk storage, smaller in size and it is available in effordable cost. This rapid rate of improvement has come both from advances in the technology used to build computers and from innovation in computer design. In this course we will mainly deal with the innovation in computer design.
- The task that the computer designer handles is a complex one: Determine what attributes are important for a new machine, then design a machine to maximize performance while staying within cost constraints. This task has many aspects, including instruction set design, functional organization, logic design, and implementation. While looking for the task for computer design, both the terms computer organization and computer architecture come into picture.

Introduction

- It is difficult to give precise definition for the terms Computer Organization and Computer Architecture. But while describing computer system, we come across these terms, and in literature, computer scientists try to make a distinction between these two terms.
- Computer architecture refers to those parameters of a computer system that are visible to a programmer or those parameters that have a direct impact on the logical execution of a program. Examples of architectural attributes include the instruction set, the number of bits used to represent different data types, I/O mechanisms, and techniques for addressing memory.
- Computer organization refers to the operational units and their interconnections that realize the architectural specifications. Examples of organizational attributes include those hardware details transparent to the programmer, such as control signals, interfaces between the computer and peripherals, and the memory technology used.

- **The model of a computer can be described by four basic units in high level abstraction. These basic units are:**
- ❍ **Central Processor Unit**
- ❍ **Input Unit**
- ❍ **Output Unit**
- ❍ **Memory Unit**

- **A. Central Processor Unit [CPU] :**
- Central processor unit consists of :
- control unit has a set of registers and control circuit to generate control signals.
- Arithmatic and Logic Unit (ALU) for execution of arithmatic and logical operations.
- In addition, CPU may have some additional registers for temporary storage of data.

B. Input Unit :

With the help of input unit data from outside can be supplied to the computer. Program or data is read into main storage from input device or secondary storage under the control of CPU input instruction. Example of input devices: Keyboard, Mouse, Hard disk, Floppy disk, CD-ROM drive etc.

- **C. Output Unit :**
- With the help of output unit computer results can be provided to the user or it can be stored in storage device permanently for future use. Output data from main storage go to output device under the control of CPU output instructions.
- Example Printer, Monitor, Plotter, Hard Disk, Floppy Disk etc.
- **D. Memory Unit :**
- Memory unit is used to store the data and program. CPU can work with the information stored in memory unit. This memory unit is termed as primary memory or main memory module. These are basically semi conductor memories.

Basic computer (cont.)

- There are two types of semiconductor memories -

- **Volatile Memory** : RAM (Random Access Memory).

- **Non-Volatile Memory** : ROM (Read only Memory), PROM (Programmable ROM) EPROM (Erasable PROM), EEPROM (Electrically Erasable PROM).

- **Secondary Memory :**
- There is another kind of storage device, apart from primary or main memory, which is known as secondary memory. Secondary memories are non volatile memory and it is used for permanent storage of data and program.
- Example of secondary memories:
- Hard Disk, Floppy Disk, Magenetic Tape :These are magnetic devices,
- CD-ROM is optical device
- Thumb drive (or pen drive) is semiconductor memory.

- **First Generation (1940-1950) :: Vacuum Tube**

- Electron emitting devices
- Data and programs are stored in a single read-write memory
- Memory contents are addressable by location, regardless of the content itself
- Machine language/Assemble language
- Sequential execution

Second Generation (1950-1964) :: Transistors

- William Shockley, John Bardeen, and Walter Brattain invent the transistor that reduce size of computers and improve reliability. Vacuum tubes have been replaced by transistors.

- First operating Systems: handled one program at a time
- On-off switches controlled by electronically.
- High level languages
- Floating point arithmetic

Third Generation (1964-1974) :: Integrated Circuits (IC)

- Microprocessor chips combines thousands of transistors, entire circuit on one computer chip.
- Semiconductor memory
- Multiple computer models with different performance characteristics
- The size of computers has been reduced drastically

▶ Fourth Generation (Very Large-Scale Integration (VLSI) / Ultra Large Scale Integration

- Combines millions of transistors
- Single-chip processor and the single-board computer emerged
- Creation of the Personal Computer (PC)
- Use of data communications
- Massively parallel machine

The von Neumann Model

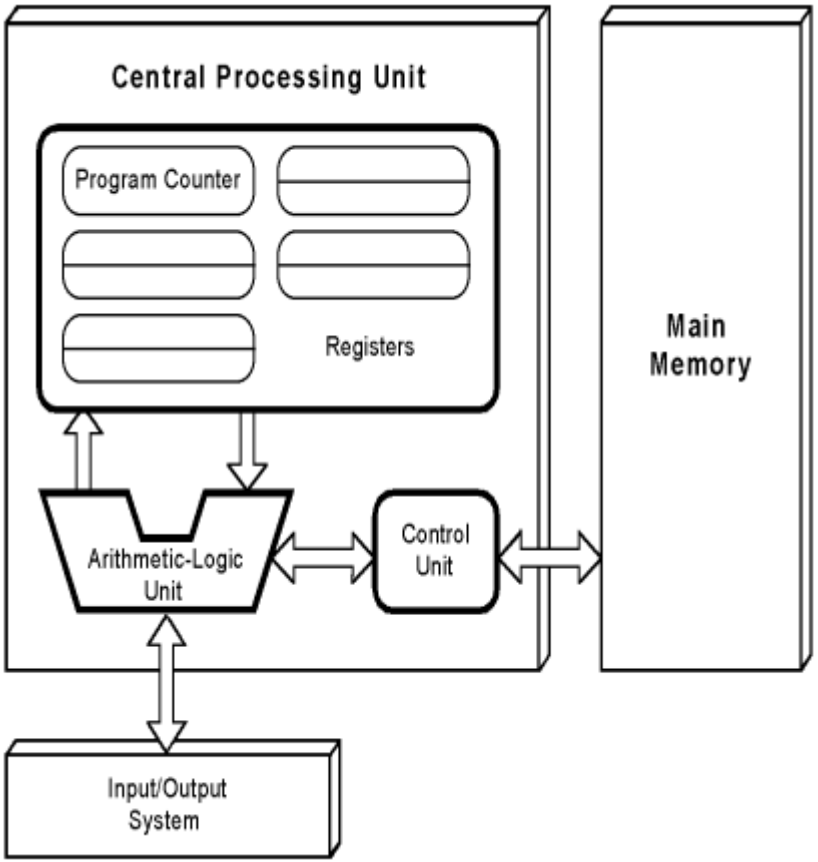
The invention of stored program computers has been ascribed to a mathematician, John von Neumann, who was a contemporary of Mauchley and Eckert.

Stored-program computers have become known as **von Neumann Architecture** systems.

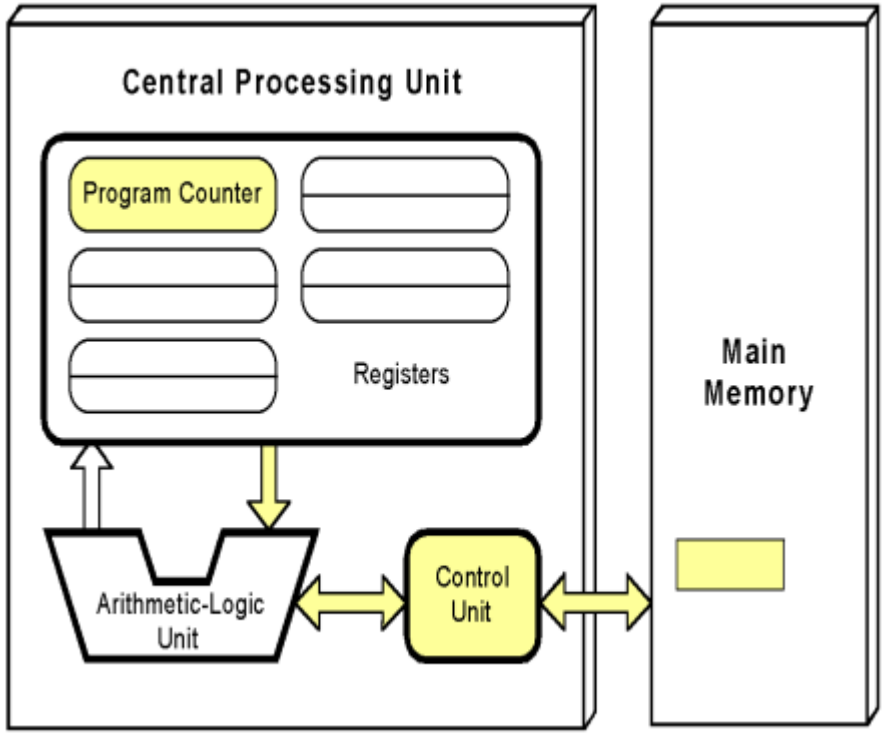
- Today's stored-program computers have the following characteristics:
  - Three hardware systems:
    - A central processing unit (CPU)
    - A main memory system
    - An I/O system
  - The capacity to carry out sequential instruction processing.
  - A single data path between the CPU and main memory.
    - This single path is known as the *von Neumann bottleneck*.

This is a general depiction of a von Neumann system:

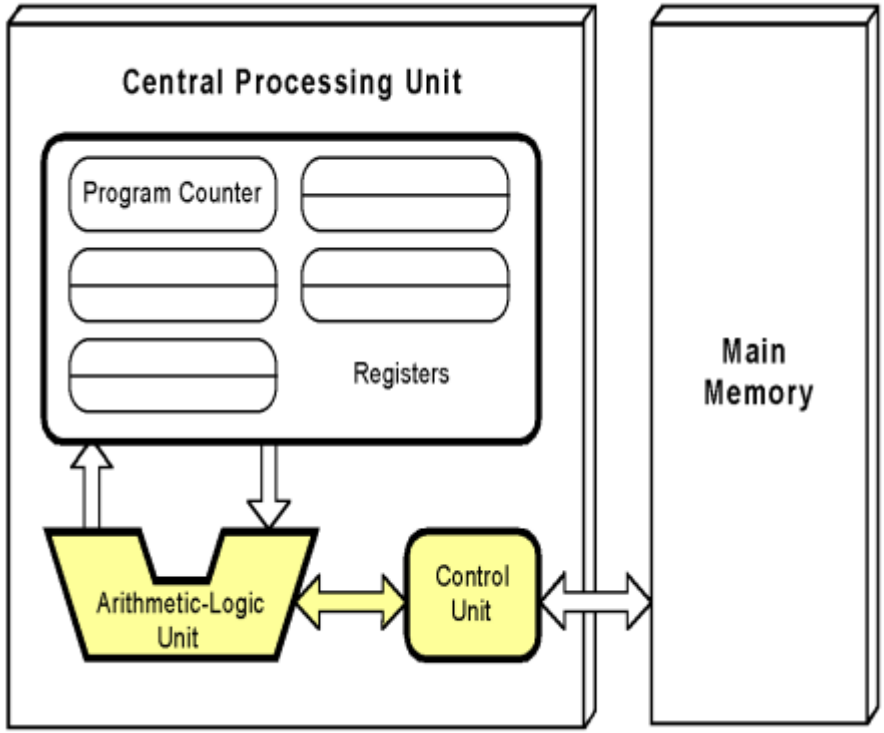
These computers employ a fetch-decode-execute cycle to run programs as follows . . .



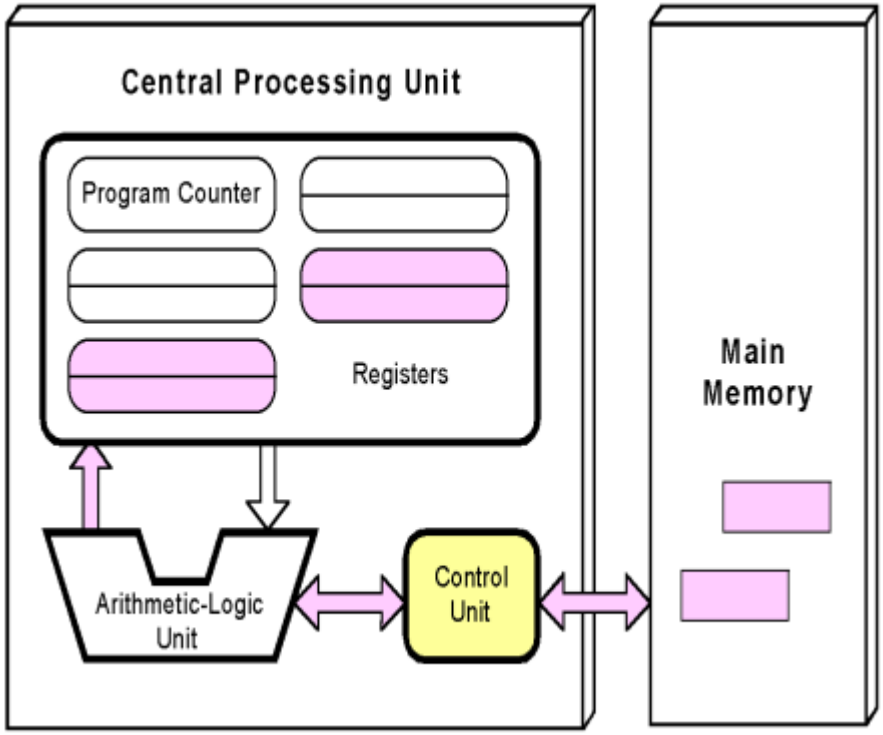
The control unit fetches the next instruction from memory using the program counter to determine where the instruction is located.



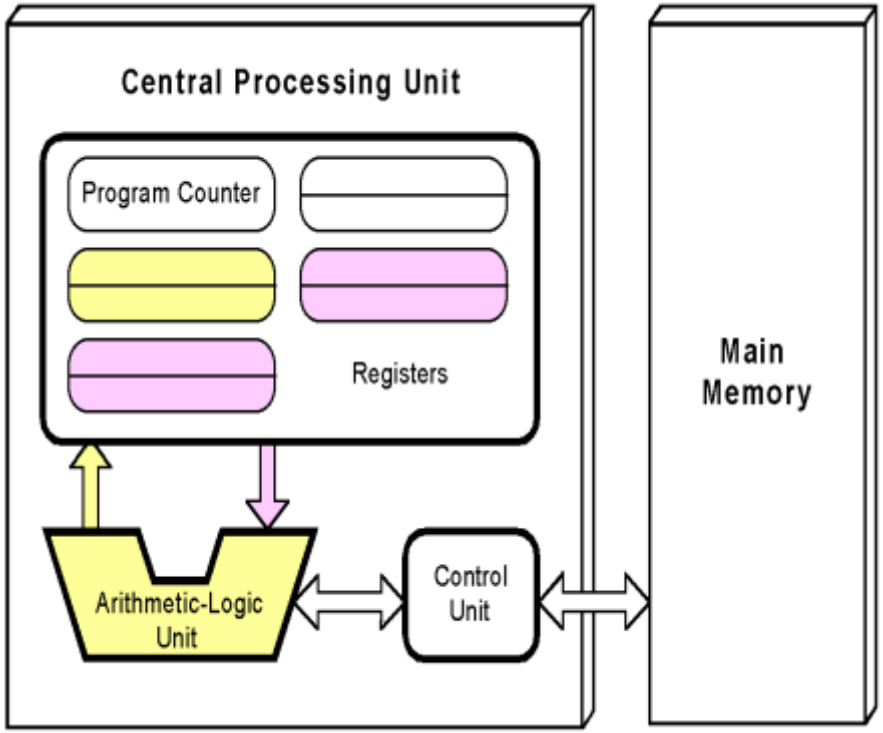
The instruction is decoded into a language that the ALU can understand.



Any data operands required to execute the instruction are fetched from memory and placed into registers within the CPU.



The ALU executes the instruction and places results in registers or memory



Conventional stored-program computers have undergone many incremental improvements over the years.

These improvements include adding specialized buses, floating-point units, and cache memories, to name only a few.

But enormous improvements in computational power require departure from the classic von

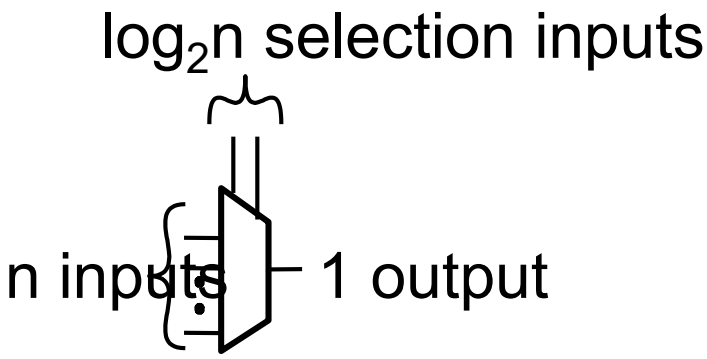
- In the late 1960s, high-performance computer systems were equipped with dual processors to increase computational throughput.
- In the 1970s supercomputer systems were introduced with 32 processors.
- Supercomputers with 1,000 processors were built in the 1980s.
- In 1999, IBM announced its Blue Gene system containing over 1 million processors.

- Parallel processing is only one method of providing increased computational power.

- DNA computers, quantum computers, and dataflow systems. At this point, it is unclear whether any of these systems will provide the basis for the next generation of computers.

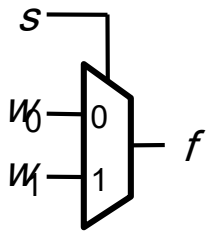
Leonard Adleman is often called the inventor of DNA computers. His article in a 1994 issue of the journal **Science** outlined how to use DNA to solve a well-known mathematical problem, called the "**traveling salesman**" problem. The goal of the problem is to find the shortest route between a number of cities, going through each city only once. As you add more cities to the problem, the problem becomes more difficult. Adleman

Multiplexers



- multiplexer
  - n binary inputs (binary input = 1-bit input)
  - $\log_2 n$ binary selection inputs
  - 1 binary output
  - Function: one of n inputs is placed onto output
  - Called **n-to-1** multiplexer

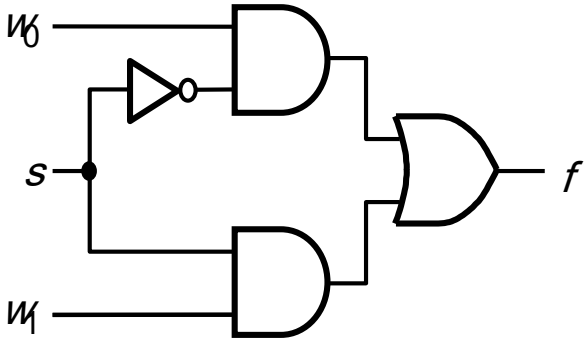
2-to-1 Multiplexer



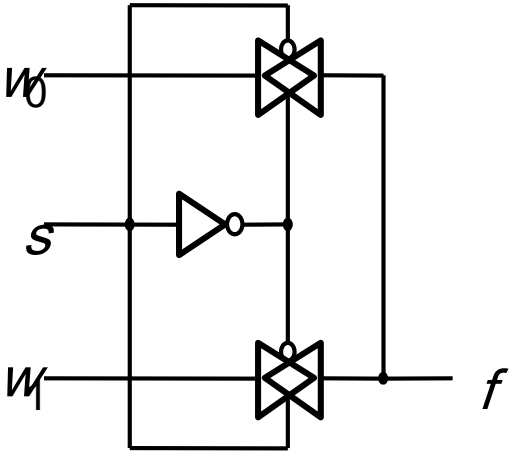
(a) Graphical symbol

s	f
0	w_0
1	w_1

(b) Truth table



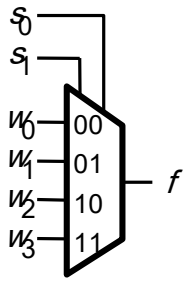
(c) Sum-of-products circuit



(d) Circuit with transmission gates

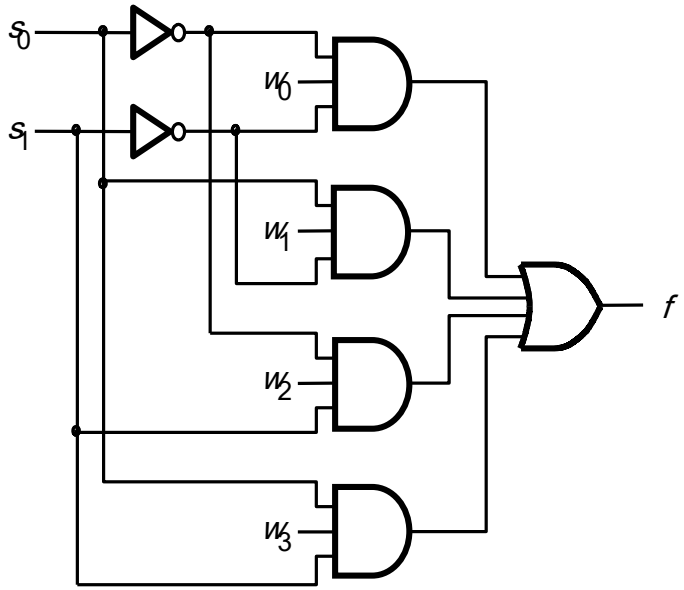
Source: Brown and Vranesic

4-to-1 Multiplexer



s_1	s_0	f
0	0	w_0
0	1	w_1
1	0	w_2
1	1	w_3

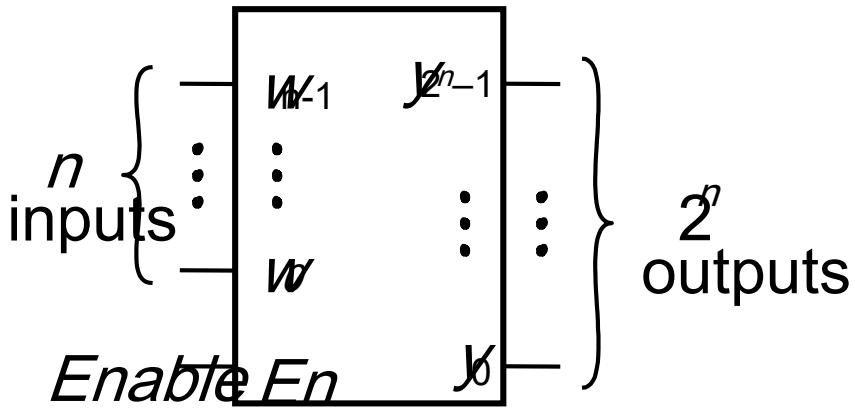
(a) Graphic symbol (b) Truth table



Source: Brown and Vranesic

(c) Circuit

Decoders

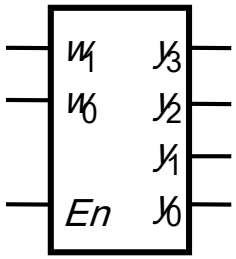


- Decoder
  - **n binary inputs**
  - **2^n binary outputs**
  - **Function: decode encoded information**
    - **If enable=1, one output is asserted high, the other outputs are asserted low**
    - **If enable=0, all outputs asserted low**
  - **Often, enable pin is not needed (i.e. the decoder is always enabled)**
  - **Called n-to-2^n decoder**
    - **Can consider n binary inputs as a single n-bit input**
    - **Can consider 2^n binary outputs as a single 2^n-bit output**
  - **Decoders are often used for RAM/ROM addressing**

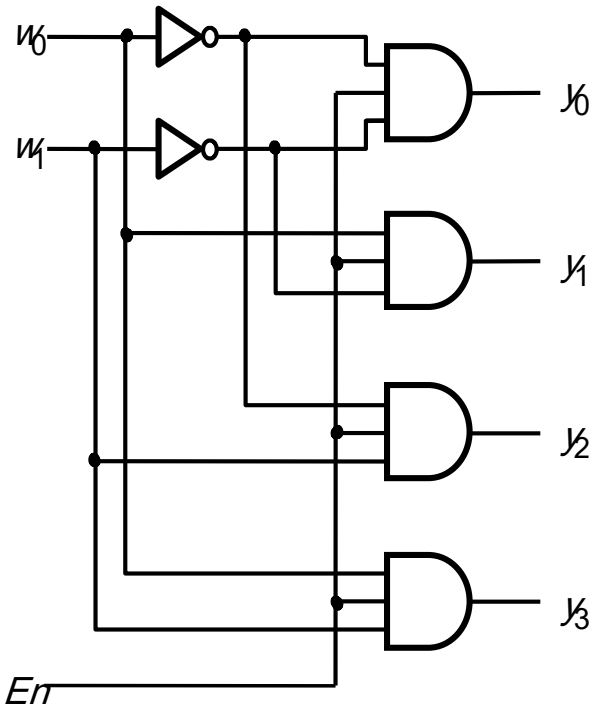
2-to-4 Decoder

En	w_1	w_0	y_3	y_2	y_1	y_0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0
0	-	-	0	0	0	0

(a) Truth table



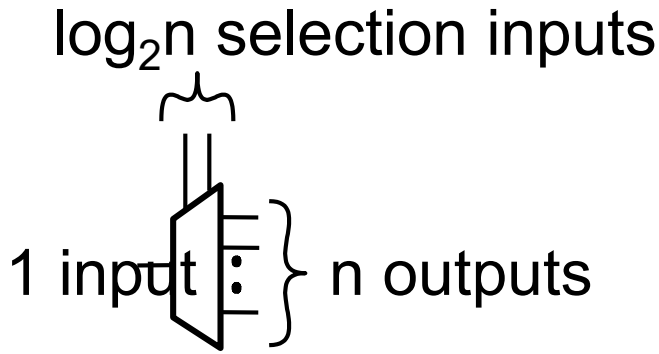
(b) Graphical symbol



(c) Logic circuit

Source: Brown and Vranesic

Demultiplexers

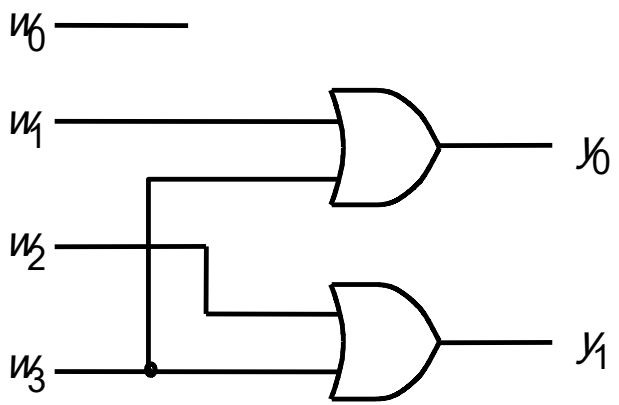


- Demultiplexer
  - 1 binary input
  - n binary outputs
  - $\log_2 n$ binary selection inputs
  - Function: places input onto one of n outputs, with the remaining outputs asserted low
  - Called **1-to-n** demultiplexer
- Closely related to decoder
  - Can build 1-to-n demultiplexer from $\log_2 n$-to-n decoder by using the decoder's enable signal as the demultiplexer's input signal, and using decoder's input signals as the demultiplexer's selection input signals.

4-to-2 Encoder

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

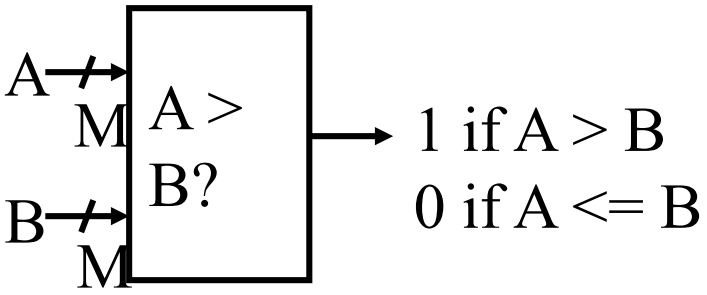
(a) Truth table



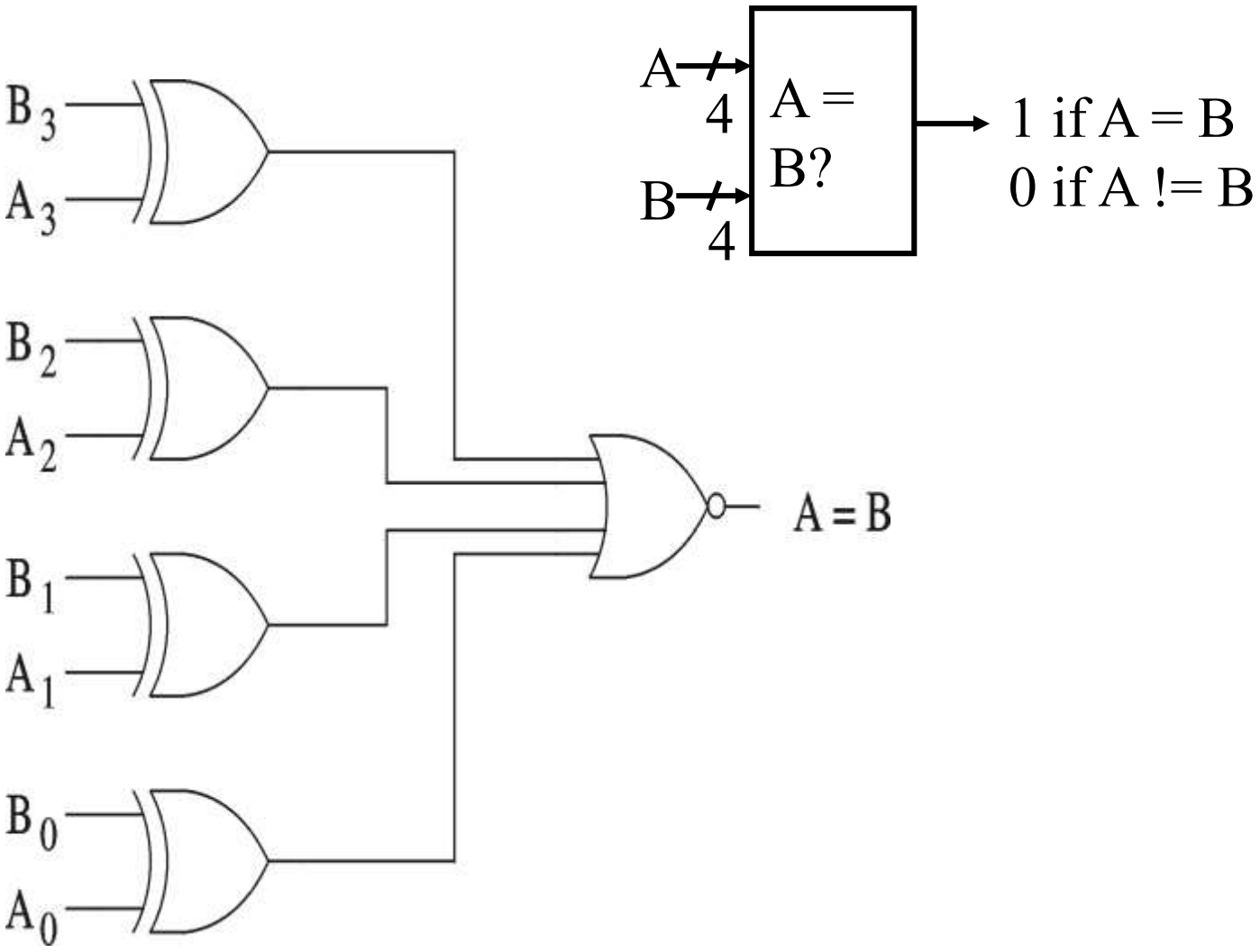
(b) Circuit

Comparator

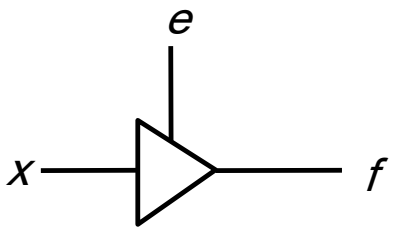
- Used two compare two M-bit numbers and produce a flag (M >1)
  - Inputs: M-bit input A, M-bit input B
  - Output: 1-bit output flag
    - 1 indicates condition is met
    - 0 indicates condition is not met
  - Can compare: >, >=, <, <=, =, etc.



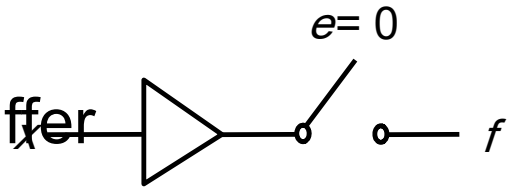
Example: 4-bit comparator (A = B)



Tri-state Buffer

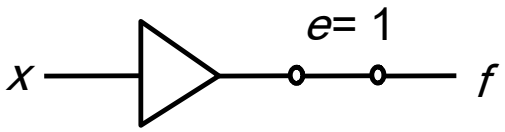


(a) A tri-state buffer



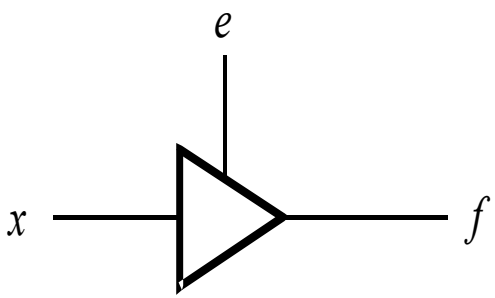
e	x	f
0	0	Z
0	1	Z
1	0	0
1	1	1

(c) Truth table

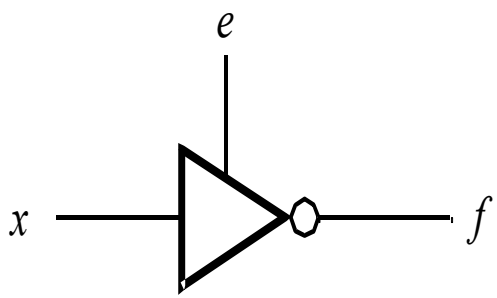


(b) Equivalent circuit

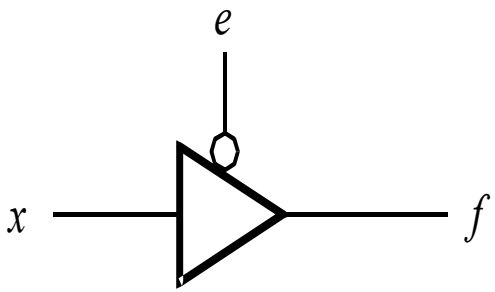
Four types of Tri-state Buffers



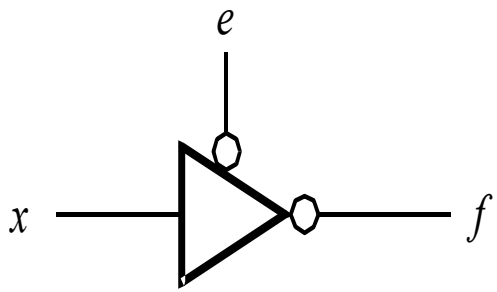
(a)



(b)

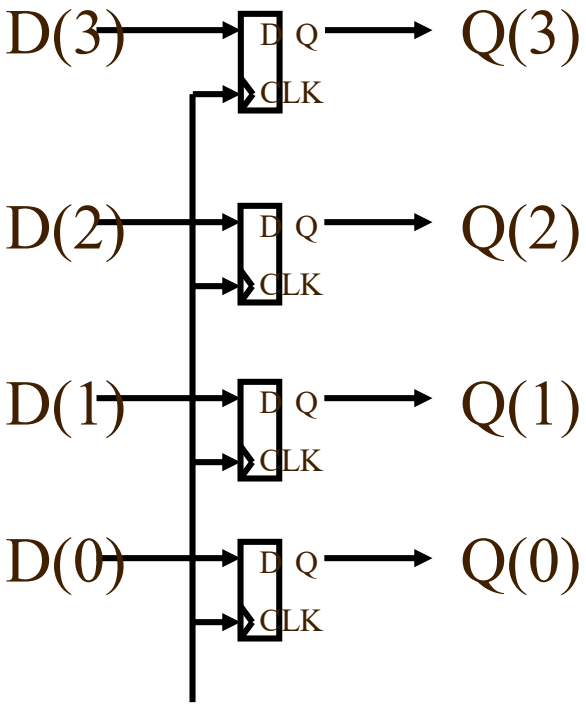


(c)



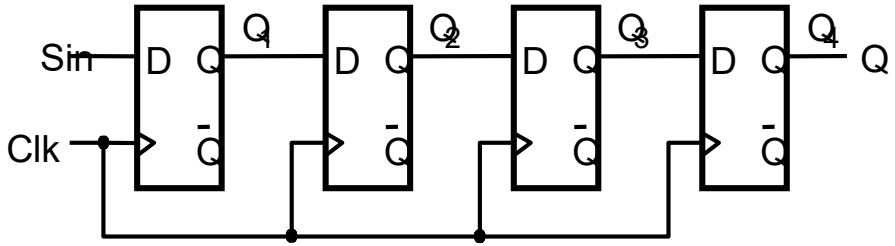
(d)

Register

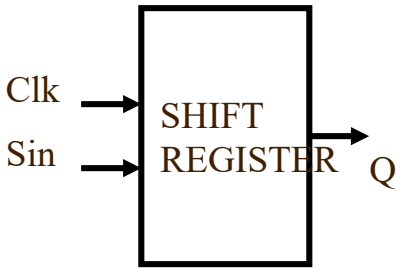


- In typical nomenclature, a register is a name for a collection of flip-flops used to hold a bus (i.e. std_logic_vector)

Shift Register



(a) Circuit



	Sin	Q_1	Q_2	Q_3	$Q_4 = Q$
t_0	1	0	0	0	0
t_1	0	1	0	0	0
t_2	1	0	1	0	0
t_3	1	1	0	1	0
t_4	1	1	1	0	1
t_5	0	1	1	1	0
t_6	0	0	1	1	1
t_7	0	0	0	1	1

(b) A sample sequence

Memory Hierarchy

Main memory

– Should store currently needed program instructions and data only

- **Secondary storage**

  – Stores data and programs that are not actively needed

- **Cache memory**

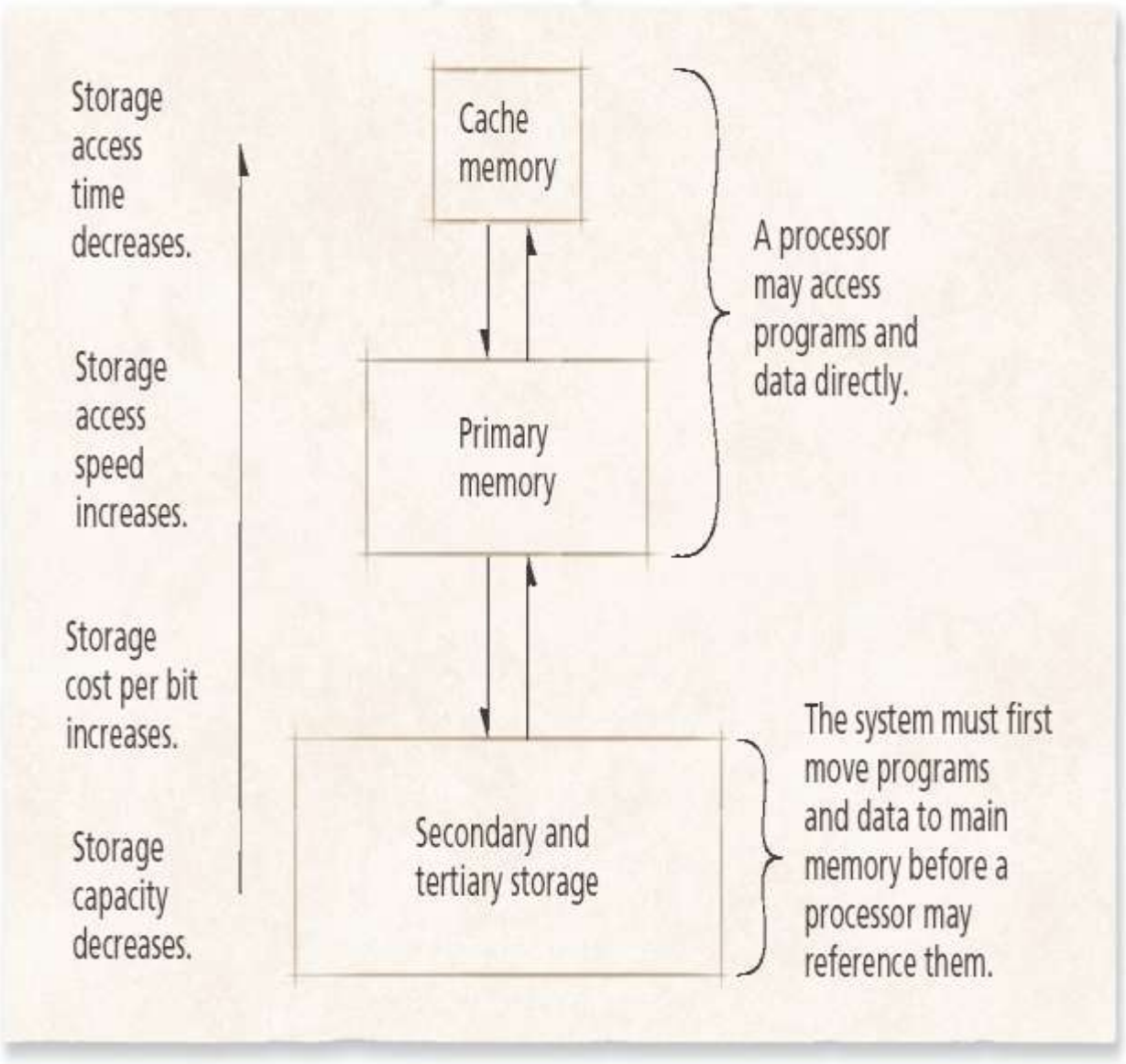
  – Extremely high speed

  – Usually located on processor itself

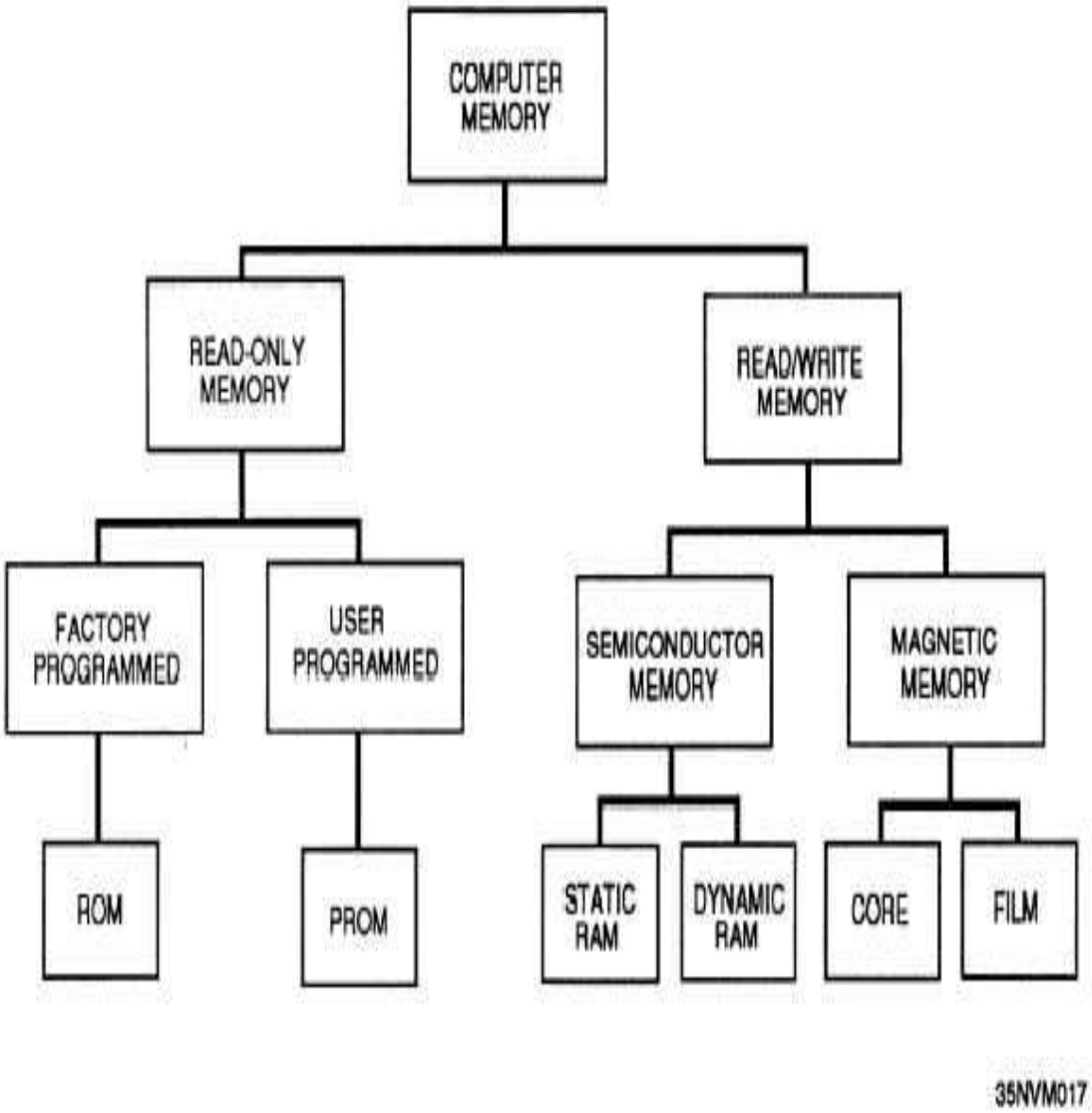
  – Most-commonly-used data copied to cache for faster access

  – Small amount of cache still effective for boosting performance

Memory Hierarchy



- Classification System of Memory



Two major types of Memory:

* RAM (Read Access Memory)

i. Perform
ii. Read
iii. Write

* ROM (Read-Only Memory)

i. A Programmable logic device
ii. Perform only the read operation

Registers associated with the memory system:

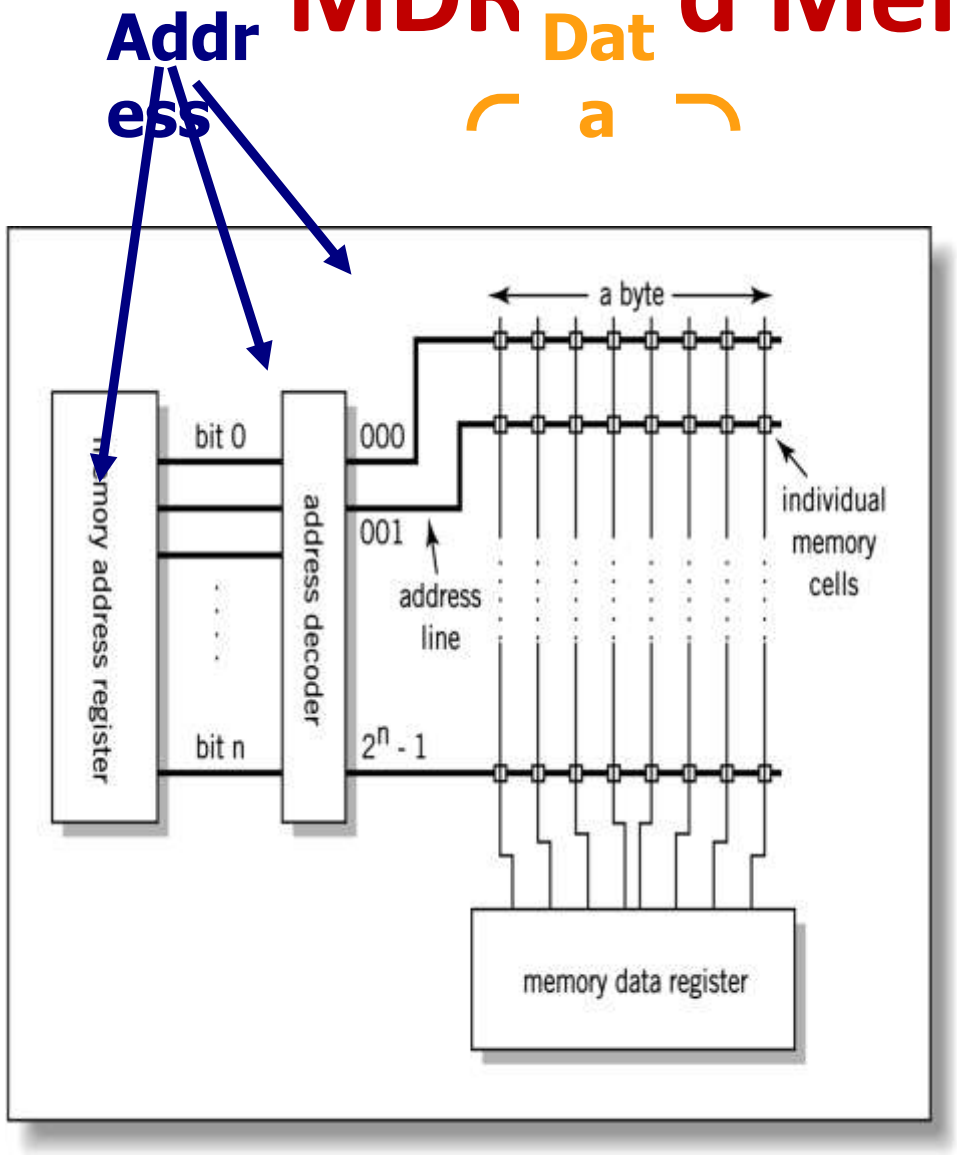
Both RAM and ROM can be characterized by two registers and a number of control signals.

For Example: Consider a memory of 2^n words, each having M bits. Then,

the Memory Address Register (MAR) is an n-bit register used to specify the memory address

The Memory Buffer Register (MBR) is an M-bit register used to hold data to be written to the memory or just read from the memory. This register is also called Memory Data Register (MDR)

Relationship between MAR, MDR and Memory



RAM

- **Memory** is a collection of storage cells with associated input and output circuitry
  - Possible to **read** and **write** cells
- **Random access memory (RAM)** contains **words** of information
- Data accessed using a sequence of signals
  - Leads to **timing waveforms**
- Decoders are an important part of

Preliminaries

- **RAMs contain a collection of data bytes**
  - **A collection of bytes is called a word**
    - **A sixteen bit word contains two bytes**
  - **Capacity of RAM device is usually described in bytes (e.g. 16 MB)**
- **Write operations write data to specific words**
- **R om**
- **M gate**

(a) Conventional symbol (b) Array logic symbol

Fig. 7-1 Conventional and Array Logic Diagrams for OR Gate

RAM Interface Signals

- **Data input and output lines carry data**
- **Memory contains 2^k words**
  - **k address lines select one word out of 2^k**
- **Read *asserted***

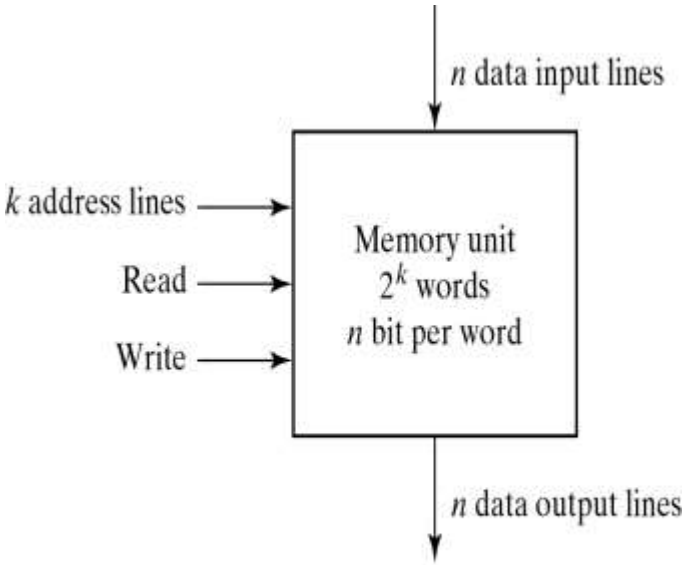
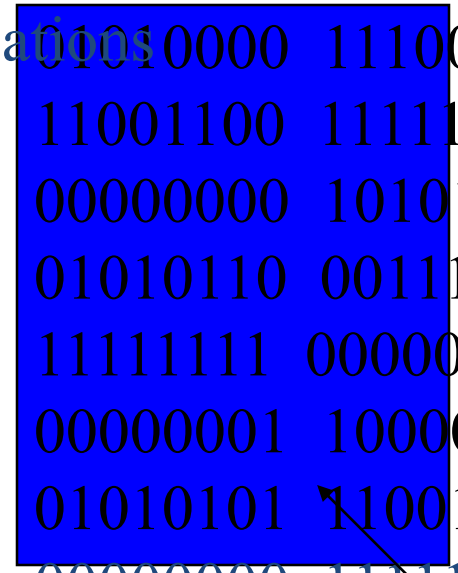


Fig. 7-2 Block Diagram of a Memory Unit

Random Access Memory Fundamentals

- **Lets consider a simple RAM chip**
  - **8 words of 2 bytes each (each word is 16 bits)**
  - **How many address bits do we need?**



RAM Size

- If memory has 2^k words, k address bits are needed
  - 2^3 words, 3 address bits
- Address locations are labelled 0 to 2^k-1
- Common subscripts:
  - Kilo – 2^{10}
  - Mega – 2^{20}
  - Giga - 2^{30}

Memory address		
Binary	decimal	Memory contest
0000000000	0	1011010101011101
0000000001	1	1010101110001001
0000000010	2	0000110101000110
⋮	⋮	⋮
1111111101	1021	1001110100010100
1111111110	1022	0000110100011110
1111111111	1023	1101111000100101

Fig. 7-3 Content of a 1024 × 16 Memory

Write Operation

1. **Apply binary address of word to address lines**
2. **Apply data bits to data input lines**
3. **Activate write input**

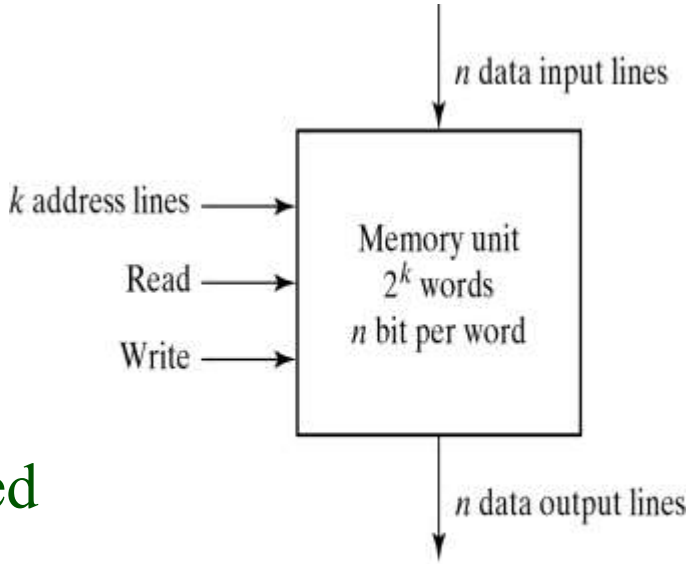


Fig. 2.1 Block Diagram of a Memory Unit

Data output lines unused

Read input signal should be inactive

Delay associated with write

Read Operation

1. **Apply binary address of word to address lines**

Data input lines unused

Write input signal should be inactive

2. **Activate read input**

Delay associated with read

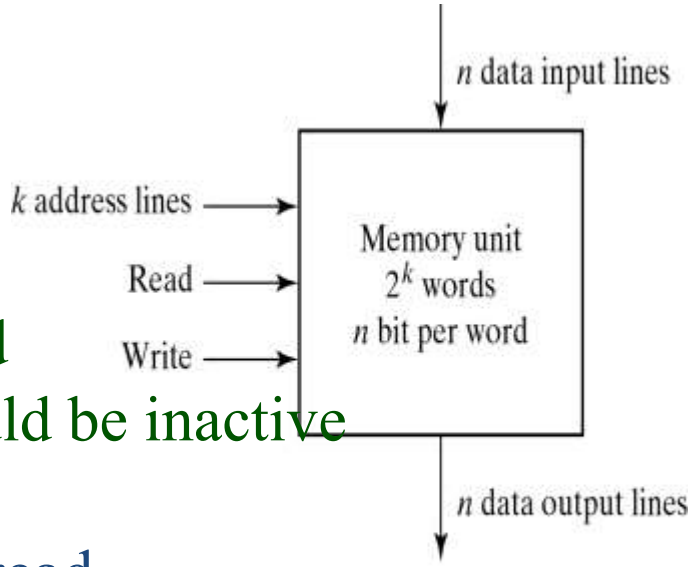


Fig. 7-2 Block Diagram of a Memory Unit

Memory enable used to allow read and writes

Types of Random Access Memories

- Static random access memory (SRAM)
  - Operates like a collection of latches
  - Once value is written, it is guaranteed to remain in the memory as long as power is applied
  - Generally expensive
  - Used inside processors (like the Pentium)
- Dynamic random access memory (DRAM)
  - Generally, simpler internal design than SRAM
  - Requires data to be rewritten (refreshed), otherwise data is lost
  - Often hold larger amount of data than SRAM
  - Longer access times than SRAM
  - Used as main memory in computer systems

Inside the RAM Device

- **Address inputs go into decoder**
  - Only one output active
- **Word line selects a row of bits (word)**
- **Data passes through OR gate**
- **Each binary cell (BC) stores one bit**
- **Input data stored if Read/Write is 0**
- **Output data driven if Read/Write is 1**

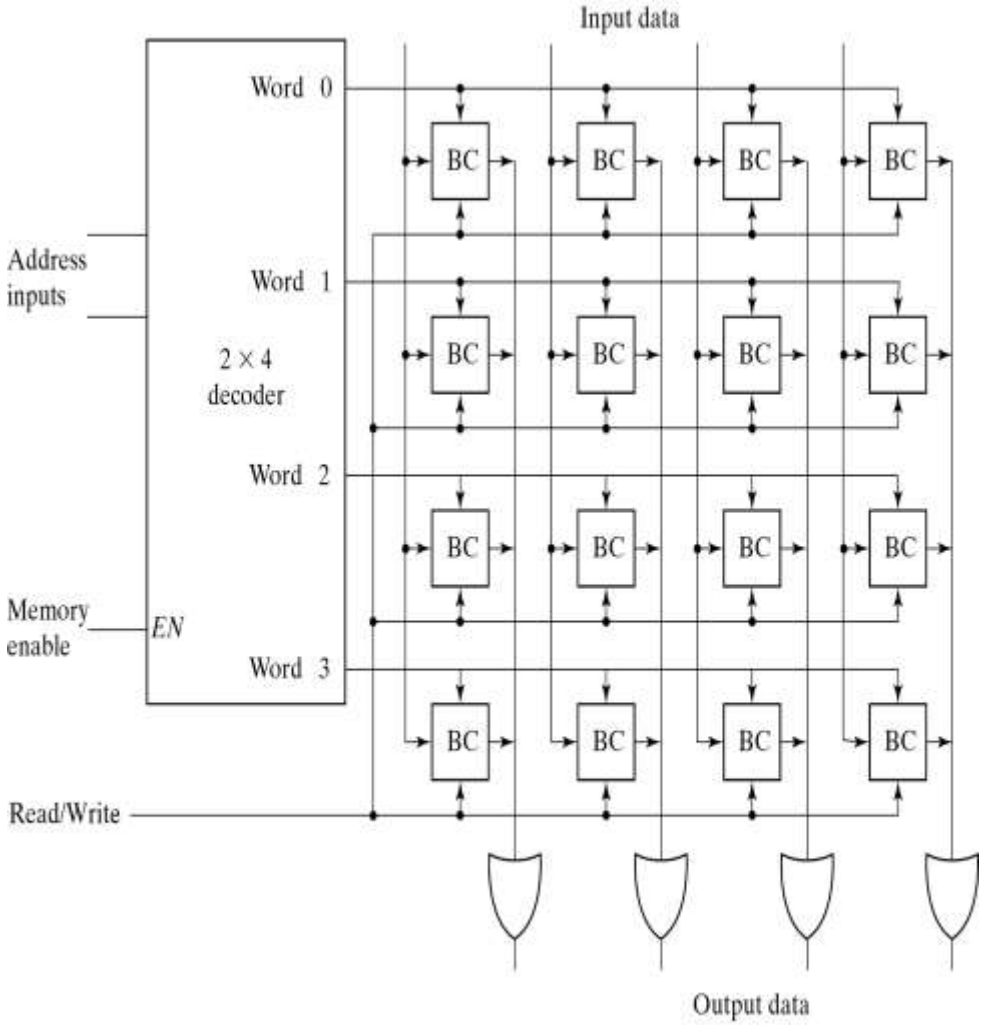


Fig. 7-6 Diagram of a 4 × 4 RAM

Inside the SRAM Device

- Basis of each SRAM cell is an S-R latch
- Note that data goes to both S and R
- Select enables operation
- Read/write enables read or write, but not both

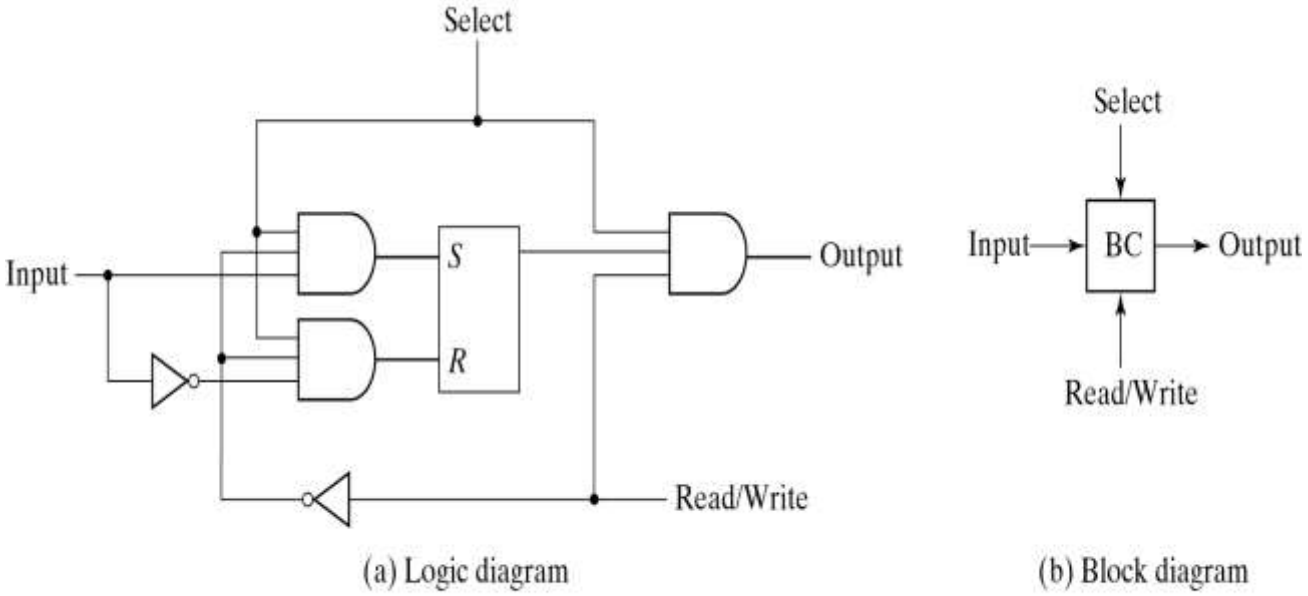


Fig. 7-5 Memory Cell

Inside the SRAM Device

- **Note: delay primarily depends on the number of words**
- **Delay not effected by size of words**

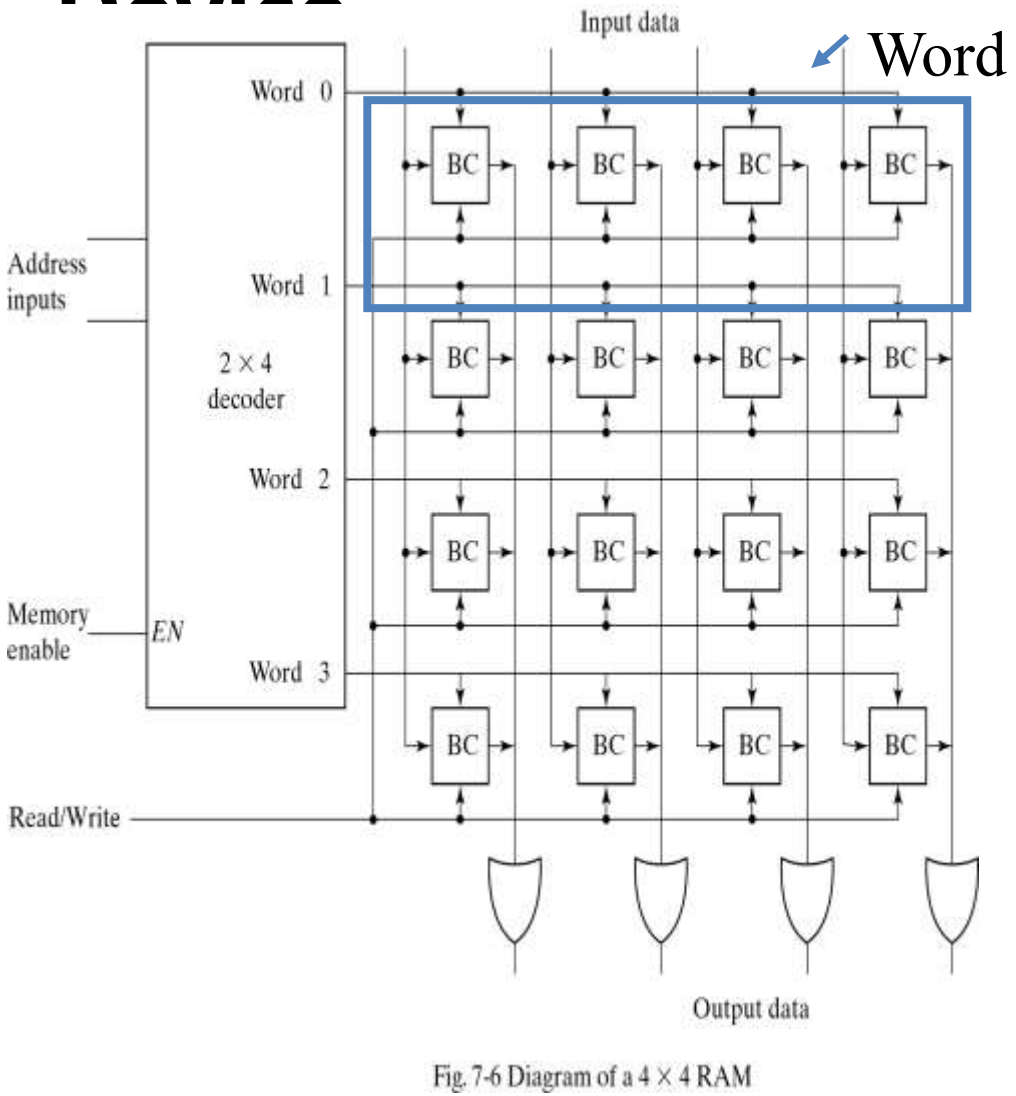


Fig. 7-6 Diagram of a 4 × 4 RAM

° **How many address bits would I need for 16 words?**

Cache memory

- If the active portions of the program and data are placed in a fast small memory, the average memory access time can be reduced,
- Thus reducing the total execution time of the program
- Such a fast small memory is referred to as cache memory
- The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component

Cache memory

- When CPU needs to access memory, the cache is examined
- If the word is found in the cache, it is read from the fast memory
- If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word

Cache memory

- When the CPU refers to memory and finds the word in cache, it is said to produce a **hit**
- Otherwise, it is a **miss**

- The performance of cache memory is frequently measured in terms of a quantity called **hit ratio**
- **Hit ratio = hit / (hit+miss)**

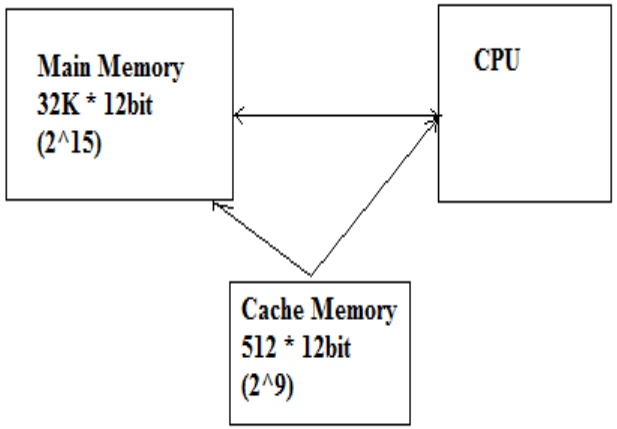
Cache memory

- The basic characteristic of cache memory is its fast access time,
- Therefore, very little or no time must be wasted when searching the words in the cache

- The transformation of data from main memory to cache memory is referred to as a **mapping** process, there are three types of mapping:
  - **Associative mapping**
  - **Direct mapping**
  - **Set-associative mapping**

Cache memory

- **To help understand the mapping procedure, we have the following example:**

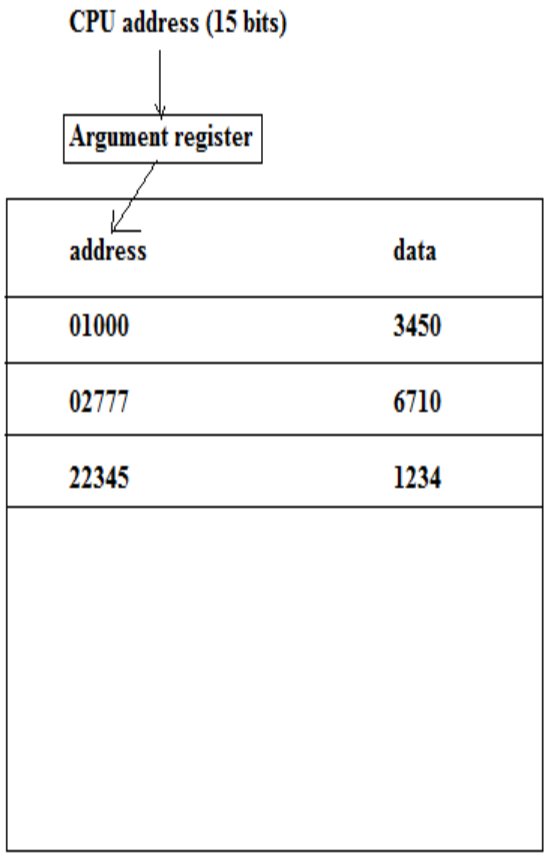


Associative mapping

- The fastest and most flexible cache organization uses an associative memory
- The associative memory stores both the address and data of the memory word
- This permits any location in cache to store any word from main memory

- The address value of 15 bits is shown as a five-digit **octal** number and its corresponding 12-bit word is shown as a four-digit octal number

Associative mapping



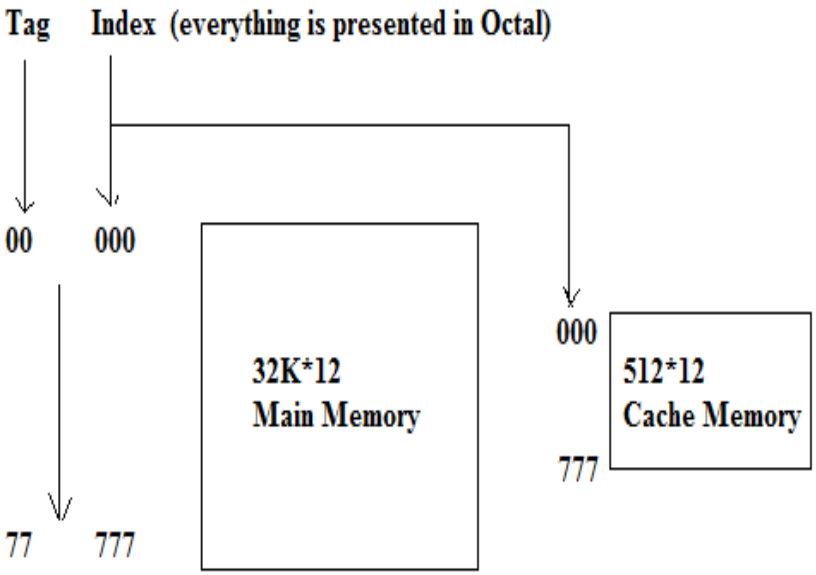
Associative mapping

- A CPU address of 15 bits is places in the argument register and the associative memory use searched for a matching address
- If the address is found, the corresponding 12-bits data is read and sent to the CPU
- If not, the main memory is accessed for the word
- If the cache is full, an address-data pair must be displaced to make room for a pair that is needed and not presently in the cache

Direct Mapping

- Associative memory is expensive compared to RAM
- In general case, there are 2^k words in cache memory and 2^n words in main memory (in our case, k=9, n=15)
- The n bit memory address is divided into two fields: k-bits for the index and n-k bits for the tag field

Direct Mapping



Direct Mapping

Memory Address	Memory Data
00000	1220
00777	2340
01000	3450
01111	2222
01777	4560
02000	5670
02777	6710

Index Address	Tag	Data
000	00	1220
111	01	2222
777	02	6710

Set-Associative Mapping

- **The disadvantage of direct mapping is that two words with the same index in their address but with different tag values cannot reside in cache memory at the same time**

- **Set-Associative Mapping is an improvement over the direct-mapping in that each word of cache can store two or more word of memory under the same index address**

Set-Associative Mapping

Memory Address	Memory Data
00000	1220
00777	2340
01000	3450
01111	2222
01777	4560
02000	5670
02777	6710

Index Address	Tag	Data	Tag	Data
000	01	3450	02	5670
111	01	2222		
777	02	6710	00	2340

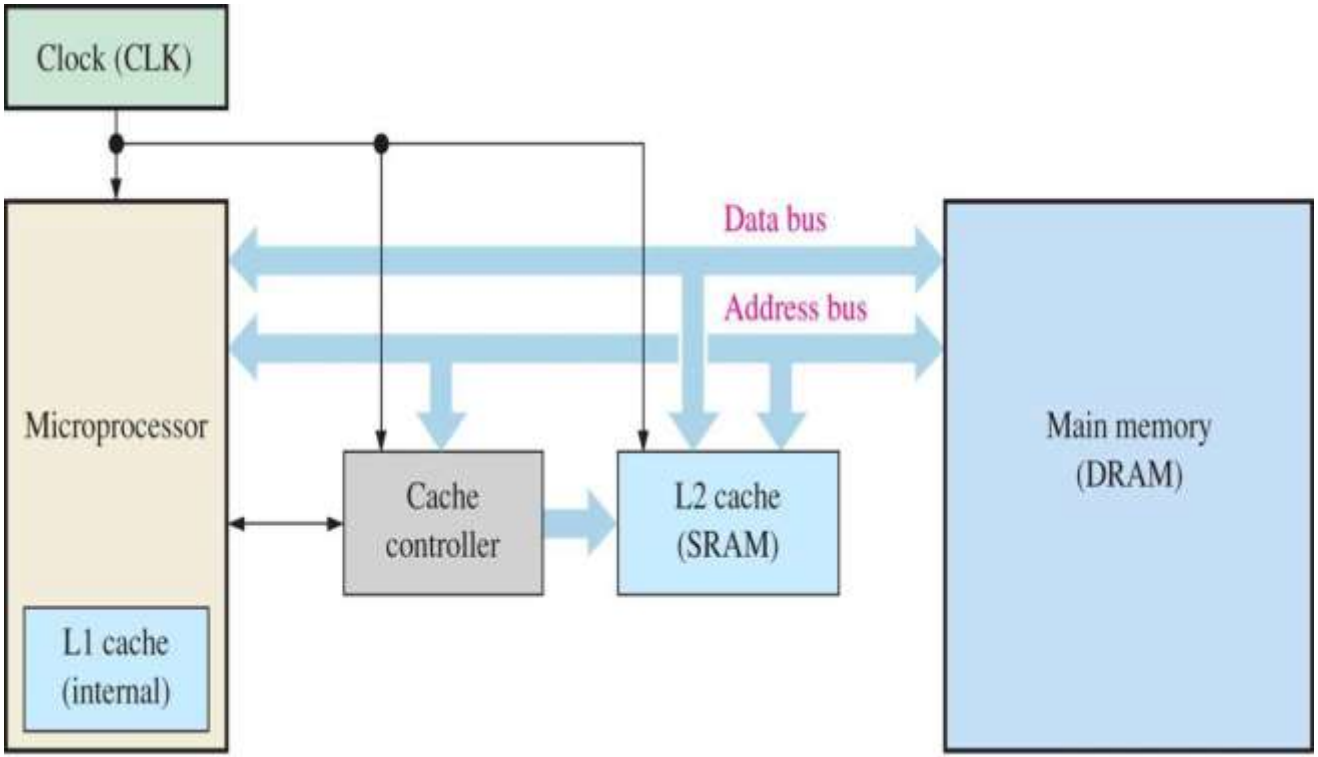
Set-Associative Mapping

- In the slide, each index address refers to two data words and their associated tags
- Each tag requires six bits and each data word has 12 bits, so the word length is 2*(6+12) = 36 bits

Cash Memory Example:

- Processors use 2 levels of cache
- Example used is a Intel 2800 MHz
  - **Level 1 (also called Primary)**
    - **Very small amount of cache 12kb**
    - **Fastest Memory**
    - **Stores recently used data and instructions**
  - **Level 2 (also called secondary)**
    - **512Kb**
    - **Faster than main memory, but slower than Level 1**
    - **Stores what can not fit into the smaller Level 1 Cache**

Block diagram showing L1 and L2 cache memories in a computer system.



CONTROL UNIT

- CPU is partitioned into *Arithmetic Logic Unit (ALU)* and *Control Unit (CU)*.
- The function of control unit is to generate relevant timing and control signals to all operations in the computer.
- It controls the flow of data between the processor and memory and peripherals

FUNCTIONS OF CONTROL UNIT

- The control unit directs the entire computer system to carry out stored program instructions.
- The control unit must communicate with both the arithmetic logic unit (ALU) and main memory.
- The control unit instructs the arithmetic logic unit that which logical or arithmetic operation is to be performed.
- The control unit co-ordinates the activities of the other two units as well as all peripherals and auxiliary storage devices linked to the computer.

DESIGN OF CONTROL UNIT

Control unit generates control signals using one of the two organizations:

- Hardwired Control Unit
- Micro-programmed Control Unit

HARDWIRED CONTROL UNIT

- It is implemented as logic circuits (gates, flip-flops, decoders etc.) in the hardware.
- This organization is very complicated if we have a large control unit.
- In this organization, if the design has to be modified or changed, requires changes in the wiring among the various components. Thus the modification of all the combinational circuits may be very difficult.

HARDWIRED CONTROL UNIT

ADVANTAGES

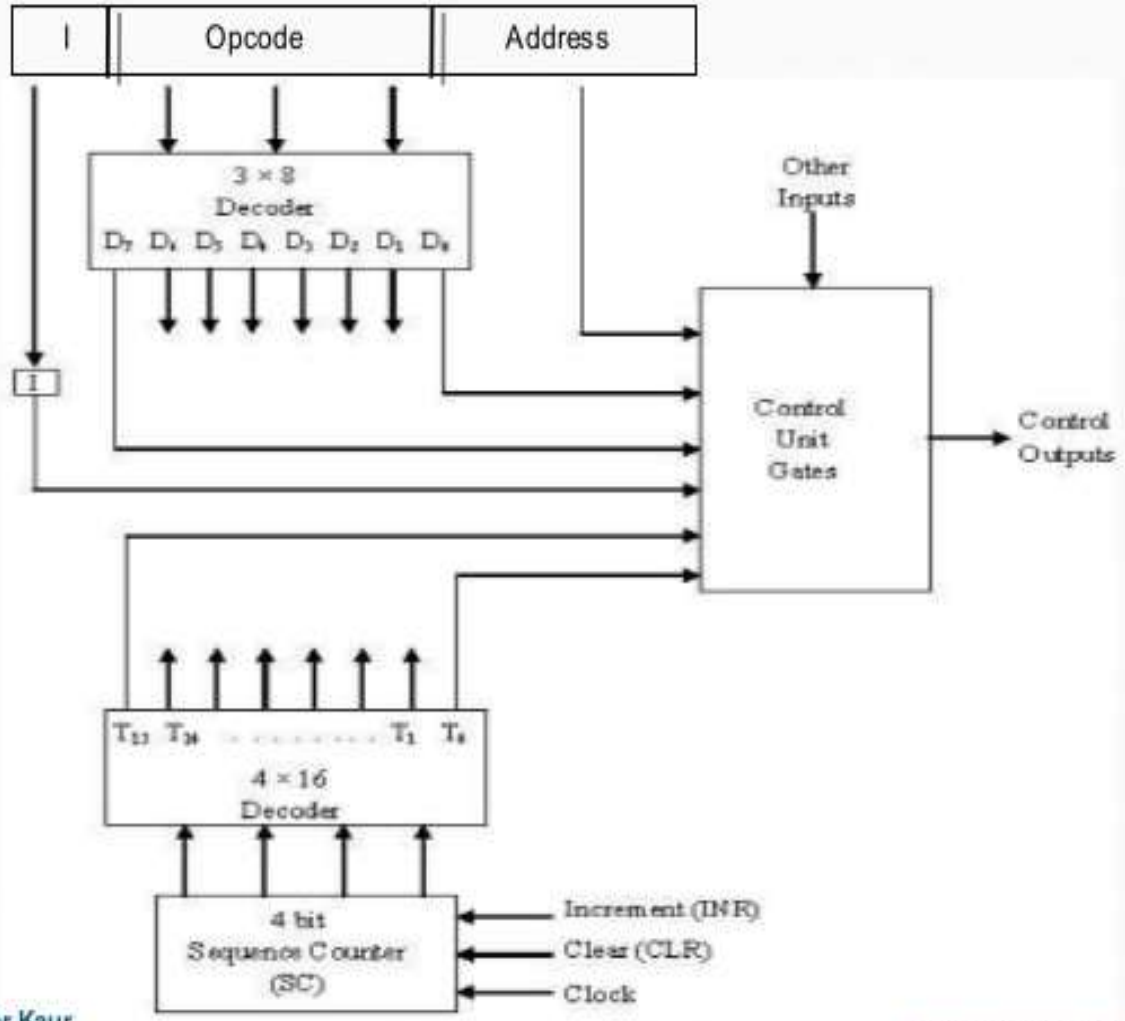
- Hardwired Control Unit is fast because control signals are generated by combinational circuits.
- The delay in generation of control signals depends upon the number of gates.

HARDWIRED CONTROL UNIT

DISADVANTAGES

- More is the control signals required by CPU; more complex will be the design of control unit.
- Modifications in control signal are very difficult. That means it requires rearranging of wires in the hardware circuit.
- It is difficult to correct mistake in original design or adding new feature in existing design of control unit.

ARCHITECTURE OF HARDWIRED CONTROL UNIT



Manindar Kaur
professormaninder@gmail.com

HARDWIRED CONTROL UNIT

Control unit consist of a:

- Instruction Register
- Number of Control Logic Gates,
- Two Decoders
- 4-bit Sequence Counter

HARDWIRED CONTROL UNIT

- An instruction read from memory is placed in the instruction register (IR).
- The instruction register is divided into three parts: the I bit, operation code, and address part.
- First 12-bits (0-11) to specify an address, next 3-bits specify the operation code (opcode) field of the instruction and last left most bit specify the addressing mode I.

  I = 0 for direct address

  I = 1 for indirect address

HARDWIRED CONTROL UNIT

- First 12-bits (0-11) are applied to the control logic gates.
- The operation code bits (12 – 14) are decoded with a 3 x 8 decoder.
- The eight outputs (D_0 through D_7) from a decoder goes to the control logic gates to perform specific operation.
- Last bit 15 is transferred to a I flip-flop designated by symbol I.

Maninder Kaur
professormaninder@gmail.com

HARDWIRED CONTROL UNIT

- The 4-bit sequence counter SC can count in binary from 0 through 15.
- The counter output is decoded into 16 timing pulses T_0 through T_{15}.
- The sequence counter can be incremented by INR input or clear by CLR input synchronously.

Maninder Kaur
professormaninder@gmail.com

HARDWIRED CONTROL UNIT

For example:

Consider the case where SC is incremented to provide timing signals T_0, T_1, T_2, T_3 and T_4 in sequence. At time T_4, SC is cleared to o if decoder output D_3 is active. This is expressed symbolically by the statement:

$$D_3 T_4: \quad SC \leftarrow o$$

- The timing diagram shows the time relationship of the control signals.

MICRO-PROGRAMMED CONTROL UNIT

- A micro-programmed control unit is implemented using programming approach. A sequence of micro-operations are carried out by executing a program consisting of micro-instructions.
- Micro-program, consisting of micro-instructions is stored in the control memory of the control unit.
- Execution of a micro-instruction is responsible for generation of a set of control signals.

MICRO-PROGRAMMED CONTROL UNIT

- A **micro-instruction** consists of:
  - One or more micro-operations to be executed.
  - Address of next microinstruction to be executed.

Micro-Operations: The operations performed on the data stored inside the registers are called *micro-operations*.

- **Micro-Programs:** Microprogramming is the concept for generating control signals using programs. These programs are called *micro-programs*.

Maninder Kaur
professormaninder@gmail.com

MICRO-PROGRAMMED CONTROL UNIT

- **Micro-Instructions:** The instructions that make micro-program are called *micro-instructions*.
- **Micro-Code:** Micro-program is a group of micro-instructions. The micro-program can also be termed as *micro-code*.
- **Control Memory:** Micro-programs are stored in the read only memory (ROM). That memory is called *control memory*.

Maninder Kaur
professormaninder@gmail.com

MICRO-PROGRAMMED CONTROL UNIT

ADVANTAGES

- The design of micro-program control unit is less complex because micro-programs are implemented using software routines.
- The micro-programmed control unit is more flexible because design modifications, correction and enhancement is easily possible.
- The new or modified instruction set of CPU can be easily implemented by simply rewriting or modifying the contents of control memory.
- The fault can be easily diagnosed in the micro-program control unit using diagnostics tools by maintaining the contents of flags, registers and counters.

Maninder Kaur
professormaninder@gmail.com

MICRO-PROGRAMMED CONTROL UNIT

DISADVANTAGES

- The micro-program control unit is slower than hardwired control unit. That means to execute an instruction in micro-program control unit requires more time.
- The micro-program control unit is expensive than hardwired control unit in case of limited hardware resources.
- The design duration of micro-program control unit is more than hardwired control unit for smaller CPU.

ARCHITECTURE OF MICRO-PROGRAMMED CONTROL UNIT

- The address of micro-instruction that is to be executed is stored in the control address register (CAR).
- Micro-instruction corresponding to the address stored in CAR is fetched from control memory and is stored in the control data register (CDR).
- This micro-instruction contains control word to execute one or more micro-operations.
- After the execution of all micro-operations of micro-instruction, the address of next micro-instruction is located.

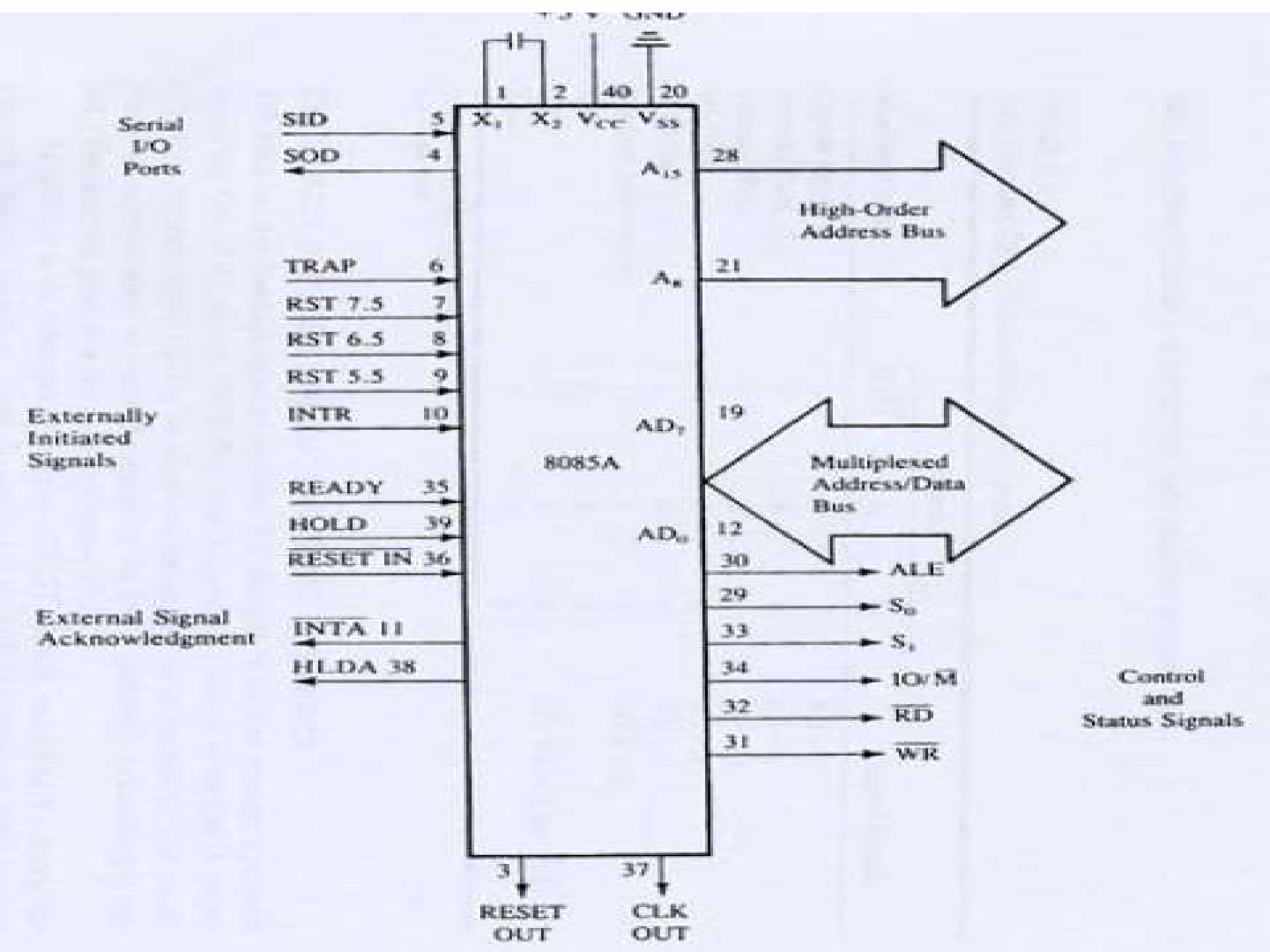
Maninder Kaur
professormaninder@gmail.com

COMPARISON BETWEEN HARDWIRED AND MICRO-PROGRAMMED CONTROL UNIT

Attributes	Hardwired Control	Micro-programmed Control
Speed	Fast	Slow
Cost of Implementation	More	Cheaper
Flexibility	Not flexible, difficult to modify for new instruction	Flexible, new instructions can easily be added
Ability to Handle Complex Instructions	Difficult	Easier
Decoding	Complex	Easy
Applications	RISC Microprocessor	CISC Microprocessor
Instruction Set Size	Small	Large
Control Memory	Absent	Present
Chip Area Required	Less	More

8085 Microprocessor Architecture

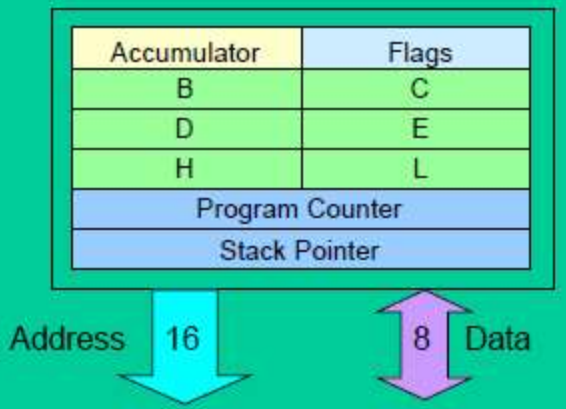
- 8-bit general purpose µp
- Capable of addressing 64 k of memory
- Has 40 pins
- Requires +5 v power supply
- Can operate with 3 MHz clock
- 8085 upward compatible


1
2
40
20
X_1
X_2
V_{CC}
V_{SS}
Serial I/O Ports
SID 5
SOD 4
TRAP 6
RST 7.5 7
RST 6.5 8
RST 5.5 9
Externally Initiated Signals
INTR 10
READY 35
HOLD 39
RESET IN 36
External Signal Acknowledgment
INTA 11
HLDA 38
8085A
A_{15} 28
A_8 21
High-Order Address Bus
AD_7 19
AD_0 12
Multiplexed Address/Data Bus
ALE 30
S_0 29
S_1 33
IO/M 34
RD 32
WR 31
Control and Status Signals
3
37
RESET OUT
CLK OUT

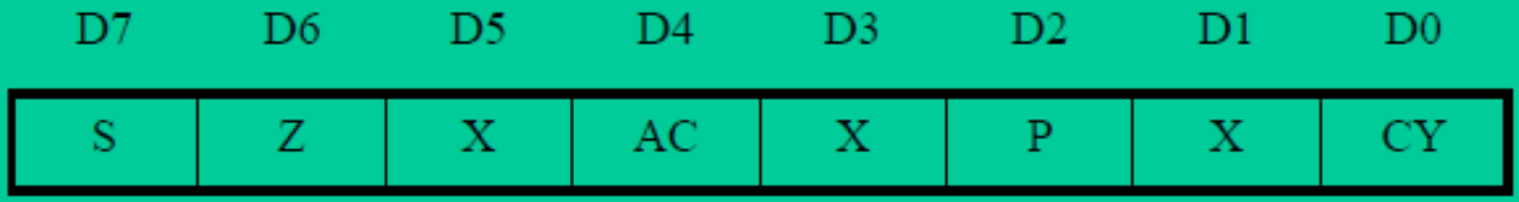
- System Bus – wires connecting memory & I/O to microprocessor
  - Address Bus
    - Unidirectional
    - Identifying peripheral or memory location
  - Data Bus
    - Bidirectional
    - Transferring data
  - Control Bus
    - Synchronization signals
    - Timing signals
    - Control signal

Intel 8085 Microprocessor

- Registers
  - General Purpose Registers
    - B, C, D, E, H & L (8 bit registers)
    - Can be used singly
    - Or can be used as 16 bit register pairs
      - BC, DE, HL
    - H & L can be used as a data pointer (holds memory address)
  - Special Purpose Registers
    - Accumulator (8 bit register)
      - Store 8 bit data
      - Store the result of an operation
      - Store 8 bit data during I/O transfer



- Flag Register
  - 8 bit register – shows the status of the microprocessor before/after an operation
  - S (sign flag), Z (zero flag), AC (auxillary carry flag), P (parity flag) & CY (carry flag)



  - Sign Flag
    - Used for indicating the sign of the data in the accumulator
    - The sign flag is set if negative (1 – negative)
    - The sign flag is reset if positive (0 –positive)

- Zero Flag
  - Is set if result obtained after an operation is 0
  - Is set following an increment or decrement operation of that register

```
  10110011
+ 01001101
----------
1 00000000
```

- Carry Flag
  - Is set if there is a carry or borrow from arithmetic operation

```
    1011 0101
+   0110 1100
-------------
Carry 1 0010 0001
```

```
    1011 0101
-   1100 1100
-------------
Borrow 1 1110 1001
```

- Auxillary Carry Flag
  - Is set if there is a carry out of bit 3
- Parity Flag
  - Is set if parity is even
  - Is cleared if parity is odd

The 8085 Instructions

- Since the 8085 is an 8-bit device it can have up to 2^8 (256) instructions.
  - However, the 8085 only uses 246 combinations that represent a total of 74 instructions.
    - Most of the instructions have more than one format.

- These instructions can be grouped into five different groups:
  - Data Transfer Operations
  - Arithmetic Operations
  - Logic Operations
  - Branch Operations
  - Machine Control Operations

The 8085 Instructions

- Since the 8085 is an 8-bit device it can have up to 2^8 (256) instructions.
  - However, the 8085 only uses 246 combinations that represent a total of 74 instructions.
    - Most of the instructions have more than one format.

- These instructions can be grouped into five different groups:
  - Data Transfer Operations
  - Arithmetic Operations
  - Logic Operations
  - Branch Operations
  - Machine Control Operations

Data Transfer Operations

- These operations simply **COPY** the data from the source to the destination.
- MOV, MVI, LDA, and STA

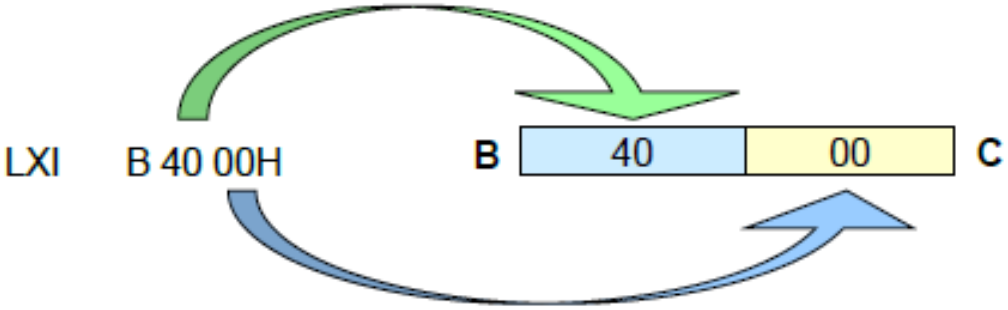
- They transfer:
  - Data between registers.
  - Data Byte to a register or memory location.
  - Data between a memory location and a register.
  - Data between an I\O Device and the accumulator.

- The data in the source is not changed.

The LXI instruction

- The 8085 provides an instruction to place the 16-bit data into the register pair in one step.
  - **LXI Rp, <16-bit address>** (Load eXtended Immediate)

  – The instruction **LXI B 4000H** will place the 16-bit number 4000 into the register pair B, C.
    - The upper two digits are placed in the 1st register of the pair and the lower two digits in the 2nd.



The Memory "Register"

- Most of the instructions of the 8085 can use a memory location in place of a register.
  - The memory location will become the "memory" register M.
    - **MOV M B**
      - copy the data from register B into a memory location.
  - Which memory location?

- The memory location is identified by the contents of the HL register pair.
  - The 16-bit contents of the HL register pair are treated as a 16-bit address and used to identify the memory location.

Using the Other Register Pairs

- There is also an instruction for moving data from memory to the accumulator without disturbing the contents of the H and L register.
  - **LDAX Rp** (LoaD Accumulator eXtended)
    - Copy the 8-bit contents of the memory location identified by the Rp register pair into the Accumulator.
    - This instruction only uses the **BC** or **DE** pair.
    - It does not accept the **HL** pair.

Arithmetic Operations

- Addition (ADD, ADI):
  - Any 8-bit number.
  - The contents of a register.
  - The contents of a memory location.
  - Can be added to the contents of the accumulator and the result is stored in the accumulator.
- Subtraction (SUB, SUI):
  - Any 8-bit number
  - The contents of a register
  - The contents of a memory location
  - Can be subtracted **from** the contents of the accumulator. The result is stored in the accumulator.

Arithmetic Operations Related to Memory

- These instructions perform an arithmetic operation using the contents of a memory location while they are still in memory.
  - ADD M
    - Add the contents of M to the Accumulator
  - SUB M
    - Sub the contents of M from the Accumulator
  - INR M / DCR M
    - Increment/decrement the contents of the memory location in place.
  - All of these use the contents of the HL register pair to identify the memory location being used.

Arithmetic Operations

- Increment (INR) and Decrement (DCR):
  - The 8-bit contents of any memory location or any register can be directly incremented or decremented by 1.
  - No need to disturb the contents of the accumulator.

Logic Operations

- These instructions perform logic operations on the contents of the accumulator.
  - ANA, ANI, ORA, ORI, XRA and XRI
    - Source: Accumulator and
      - An 8-bit number
      - The contents of a register
      - The contents of a memory location
    - Destination: Accumulator

ANA R/M	AND Accumulator With Reg/Mem
ANI #	AND Accumulator With an 8-bit number
ORA R/M	OR Accumulator With Reg/Mem
ORI #	OR Accumulator With an 8-bit number
XRA R/M	XOR Accumulator With Reg/Mem
XRI #	XOR Accumulator With an 8-bit number