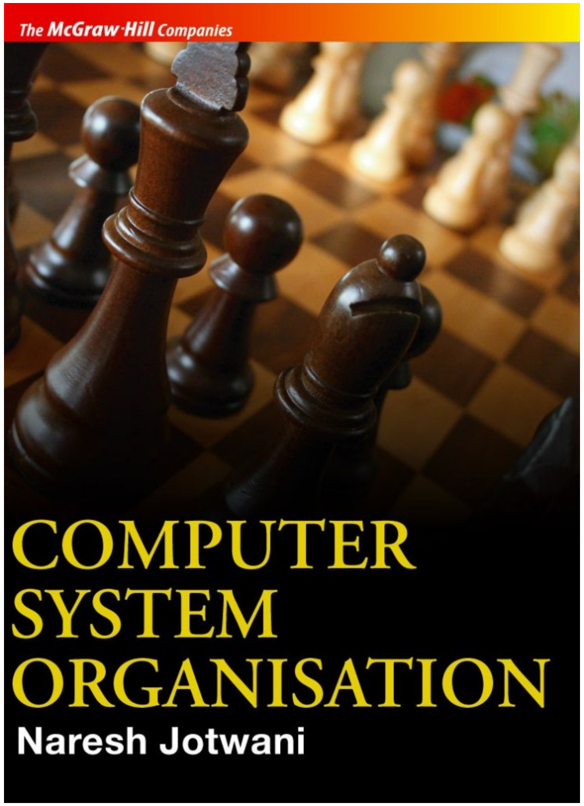
Mc Graw Hill Education

COMPUTER SYSTEM ORGANISATION

Naresh Jotwani

PowerPoint Slides



CHAPTER 3

HARDWARE BUILDING BLOCKS

Electronics

- Electron - an elementary particle
  - indivisible constituent of all matter
  - extremely light and electrically charged

- Movement of electrons through a medium generates electric *current*

- Electrons can also be made to bunch up together without moving, to create electric potential or *voltage*

- Electronics uses properties of electrons as well as various materials to achieve many different useful effects.

- Materials: conductors, insulators, semiconductors

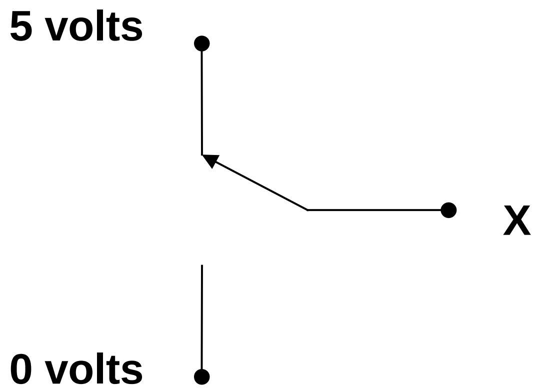
Digital Electronics

Utilizes two distinct states of an electronic device, to provide a physical realization of the notion of *one bit of information*

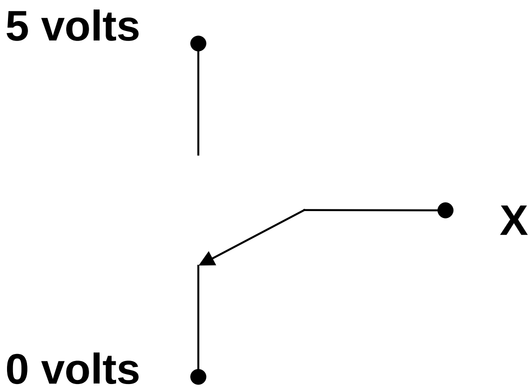
Allows us to build an electronic system in which a large number of bits can be reliably stored, written as input, or read as output

Also allows us to build a system for *processing* the information stored, according to well-defined instructions (i.e. program) provided by the user

Simple "ON-OFF" switch



(a) Switch is ON. Voltage at point X is 5 volts.



(a) Switch is OFF. Voltage at point X is 0 volts.

An ON-OFF switch and its two possible output voltages. A *transistor* is the electronic device which functions as a switch.

- To achieve electronic storage of a large number of bits, transistors are arranged in a rectangular array, or grid.

- At each intersection of a row and a column in this grid, a single bit of storage is provided.

- Appropriate means have to be provided to *read* or *write* any single bit in this grid.

- A typical semiconductor memory element is built on such a principle.

Logic Gates

- (i) *and* operation, also known as *conjunction*, has output 1 if and only if <u>all</u> of its inputs have value 1.

- (ii) *or* operation, also known as *disjunction*, has output 1 if and only if <u>at least one</u> of its inputs has value 1.

- (iii) unary *not* operation, also known as negation, has output 1 if and only if input is 0, and *vice versa*.

- By using a combination of these operations, we can define other complex functions.
- Electronic circuits implementing these operations are known as *logic gates*.

Value of input X_1	Value of input X_2	Value of output Y
0	0	0
0	1	0
1	0	0
1	1	1

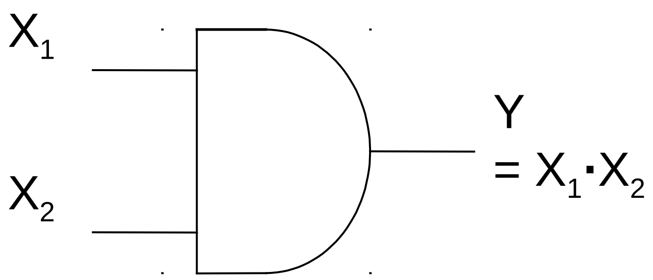


Table and graphical notation of *and* operation

Value of input X_1	Value of input X_2	Value of output Y
0	0	0
0	1	0
1	0	0
1	1	1

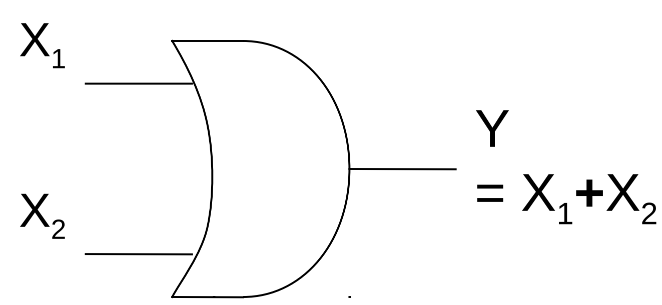


Table and graphical notation of *or* operation

Combinational Circuits

- Consider a digital circuit with three input variables X1, X2 and X3, and output variable Y, defined using *and*, *or* and *not* gates in the following way:

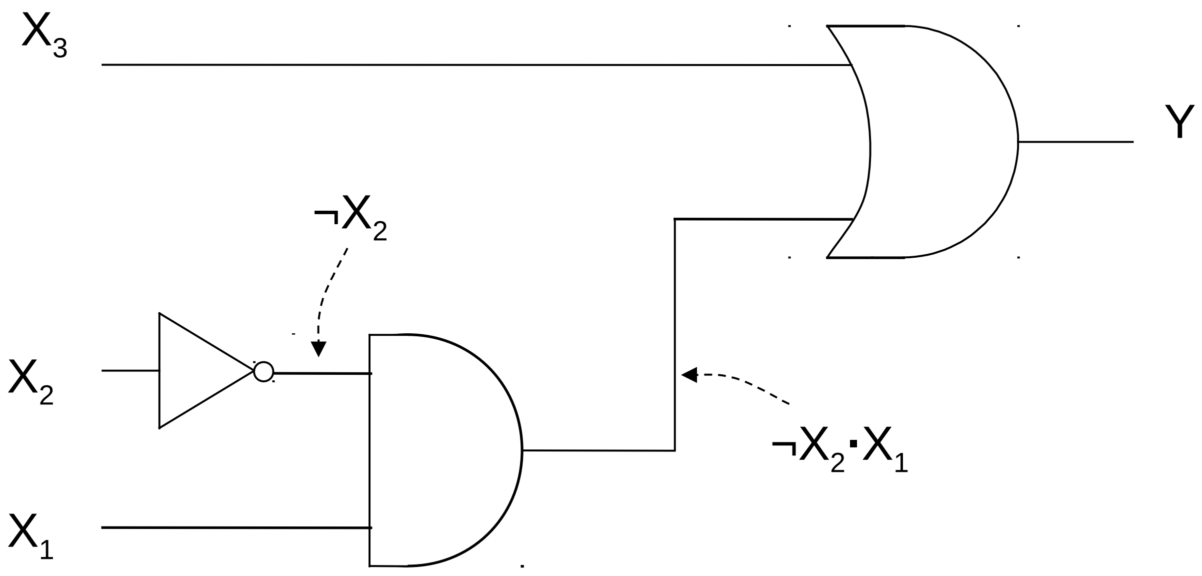
  Y = *or*(X3, *and*(X1, *not*(X2)))

- Note here that:
  - (i) The output of *not* is one of the inputs of *and*; the second input of *and* is X1.
  - (ii) The output of *and* is one of the inputs of *or*; the second input of *or* is X3.

- Replacing function notation by *operator* symbols +, ·, and ¬ for functions *or*, *and* & *not* respectively, we get:

  $$Y = X3 + X1 \cdot (\neg X2)$$

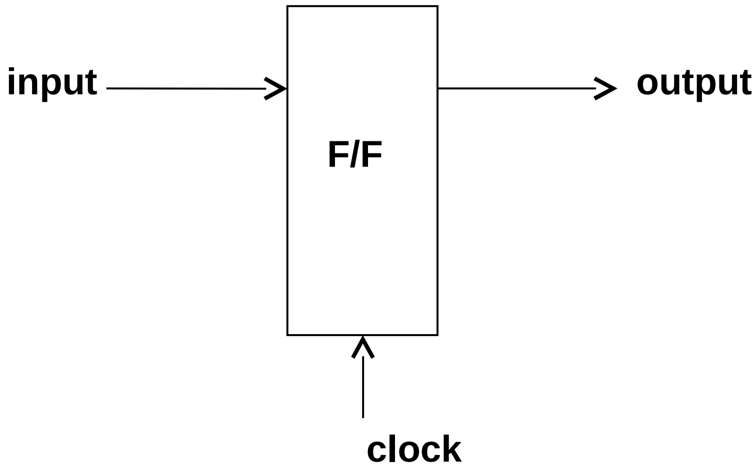
The same function can be represented graphically as under, in the form of a *schematic diagram* of a simple digital circuit:



Sequential circuits

- A combinational function such as Y = *f*(X1, X2 …. Xn) has ‘no memory’, because the output is determined solely by the inputs.

- Typical electronic circuit which can record one bit of information is the *flip-flop*.



A flip-flop with an input line, an output line, and a clock input at the bottom

- Once in every clock cycle, at an appropriate clock *transition* or *edge* – the flip-flop records the state of the input.

- If input is 0 at the clock transition, the flip-flop records 0;
- If input is 1, the flip-flop records 1.

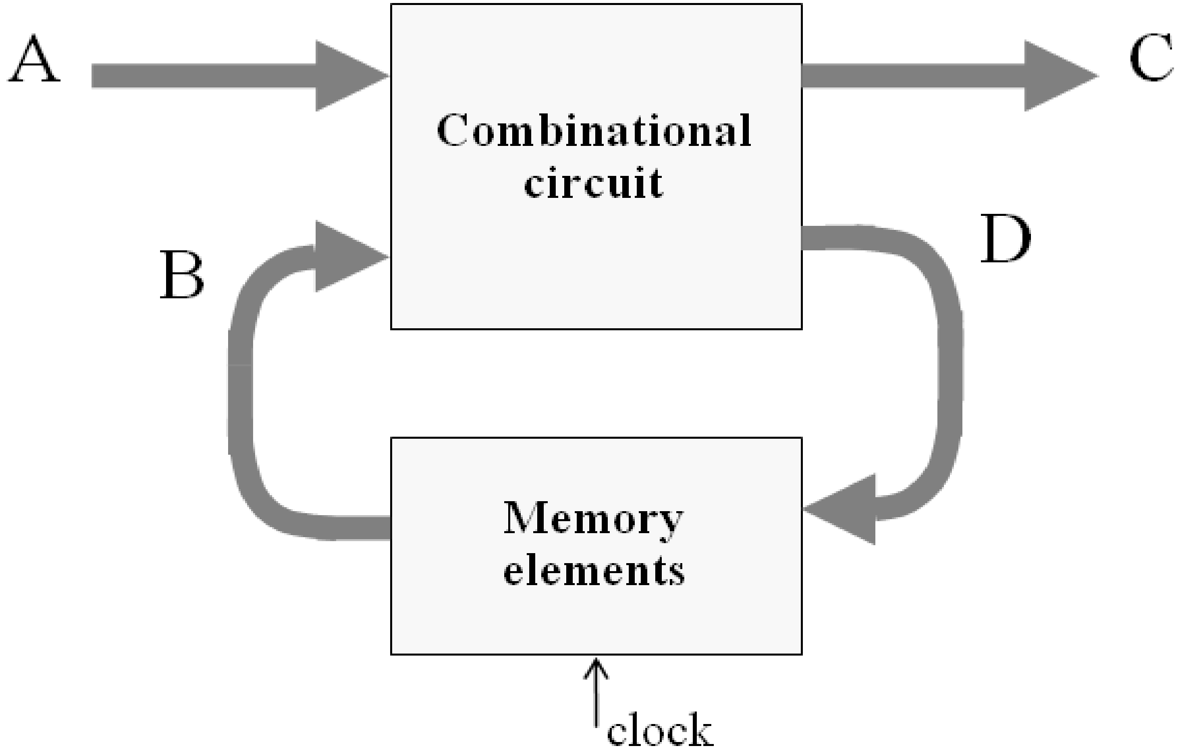
- When a flip-flop is in state 1, it is said to be *set*, and when it is in state 0, it is said to be *reset*.

- A flip-flop thus provides one bit of memory. By arranging *N* such flip-flops in a linear array, we obtain *N* bits of memory

SEQUENTIAL CIRCUIT

In the figure, A, B, C and D have the following meaning:

- A - Input signals to the sequential circuit
- B - State of the circuit at time *t*
- C - Output signals of the sequential circuit
- D - State of the circuit at time $t+\Delta t$



Integrated Circuits

- An IC has components such as transistors, resistors and capacitors created in silicon, with strips of conductors connecting them.

- Electrical isolation between components is achieved by creating insulating layers of silicon dioxide.

- To define required geometries on the silicon substrate, two-dimensional optical masks are placed over it during processing steps.

- A silicon chip with an IC must be provided with *leads* for making connections to other chips and components, and it must be packaged durably in ceramic or plastic.

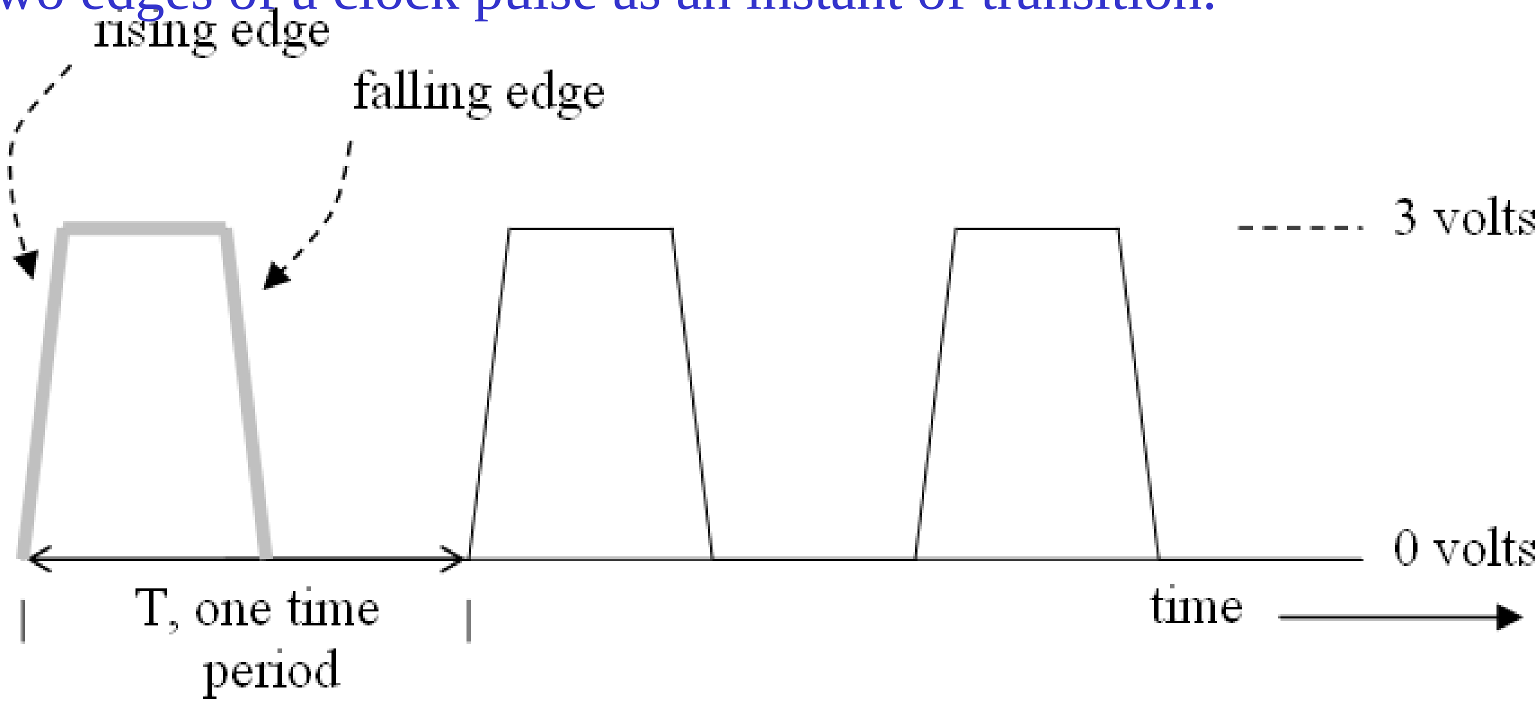
- ICs are available for a vast range of useful functions, including logic gates, counters, decoders, timers, memory chips, processors, and device controllers.

- Within a computer system, ICs are mounted on *printed circuit boards* (PCBs).

Clock signal

- Majority of circuits employed in today's computer systems employ a clock signal .
- In the next slide, three cycles of a typical clock signal are shown.
- Horizontal axis represents time, while the vertical axis represents clock voltage.

- In one time period, the clock signal exhibits two *transitions* or *edges*.
- Within one time period, the part of clock signal between the rising edge and the falling edge is known as a *clock pulse.*
- A flip-flop – or in general any sequential circuit – utilizes one of the two edges of a clock pulse as an instant of transition.



- From one *master clock*, circuit designers can derive other lower frequency clocks.
  - Thus different parts of a computer system can operate at different clock speeds, if required

- Any processing step in a computer system happens at the rising or falling edge of a clock pulse.

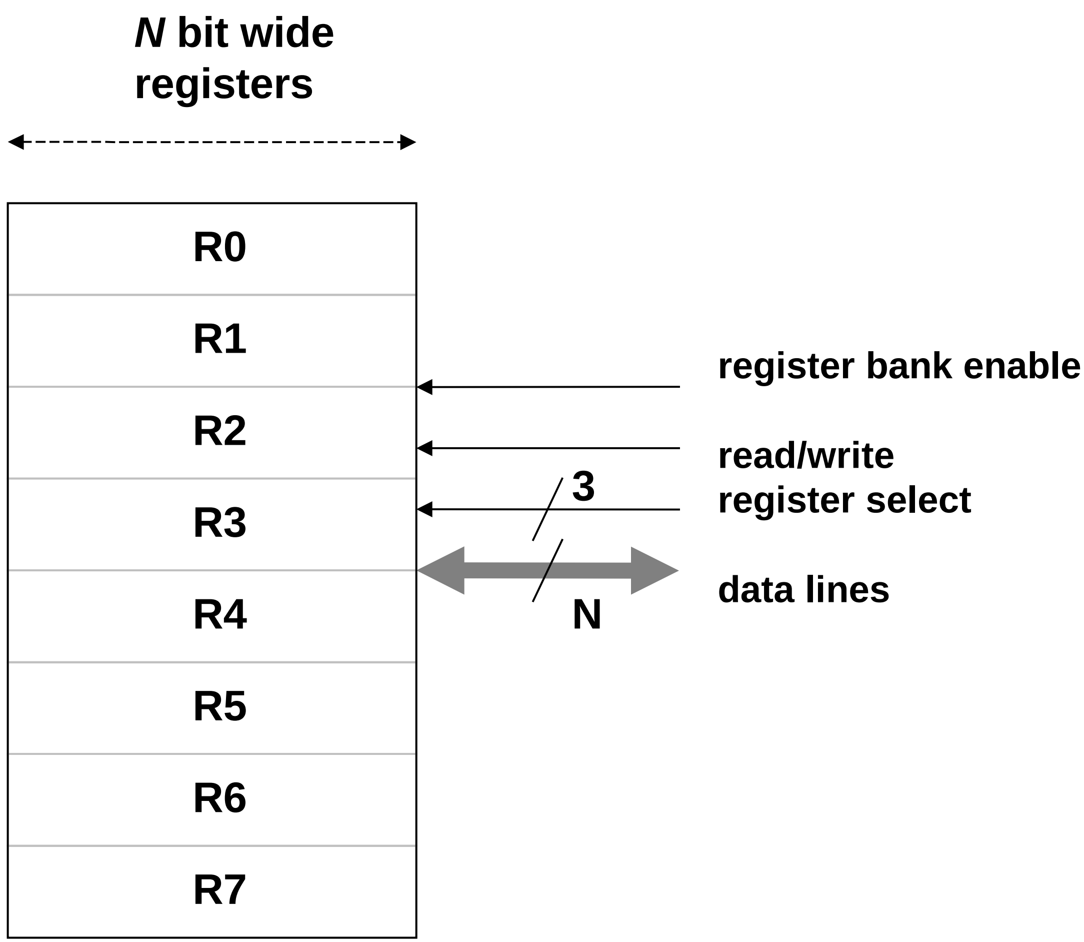
- It is important to view the clock signal as a *physical signal*, and not a logical signal.

- Several manufacturers have specialized clock generator circuits in their product line, which system designers can use as building blocks.

Registers

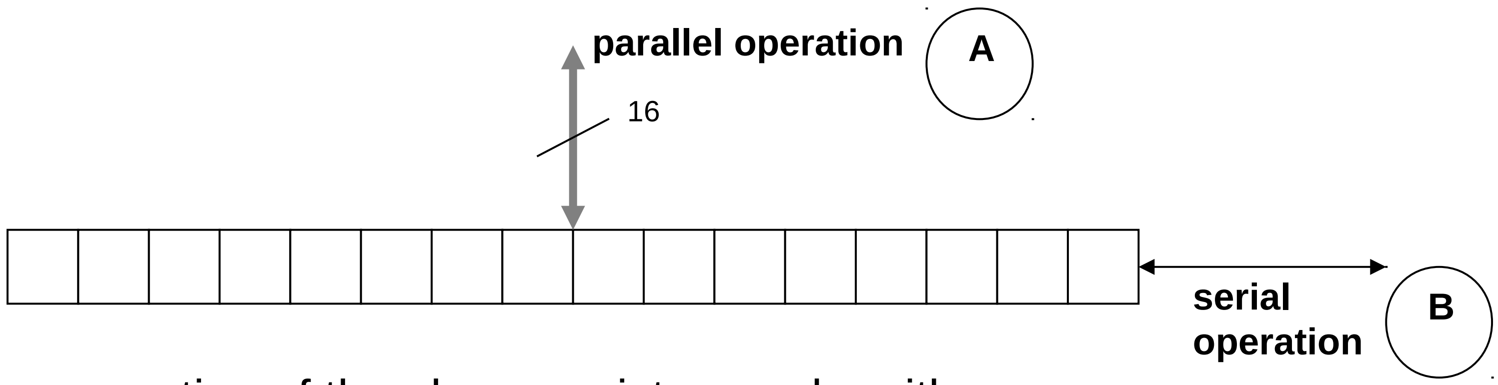
- When an array of *N* flip-flops is designed as a single unit of storage, we call the resulting element an *N-bit register*.
- If a register is used primarily as a counter, we may call it an *N-bit counter.*
- The unsigned integer count in an *N*-bit counter goes from 0 to 2^{N-1}, after which it is again reset to 0.

A bank of eight *N*-bit registers



- In the previous slide, the thick line shows *bi-directional data lines*
  - They connect this register bank with some other part of the system
- Above the data lines, we see a group of three lines labeled *register select.*
  - Any operation on the bank targets one of the eight registers, so we need three bits of information to select a register for a *read* or a *write* operation.
- Above the register select lines, there is a signal line labeled *read/write*.
  - In a given clock cycle, the *read/write* signal decides whether a read or write operation takes place

- At the top, we see a signal labeled *register bank enable*.
  - We must *enable* the register bank only when performing an operation on it, and not otherwise.
- In the previous figure, we read or write *N* bits in one clock cycle – i.e. the *N* bits of the selected register are read or written *in parallel*.
- The next slide shows a 16-bit register provided with both parallel and *serial* modes of operation.



The operation of the above register can be either

(i) a parallel write at ‘A’, followed by serial shift of the data out from ‘B’, or

(ii) serial shift of the data in from ‘B’, followed by parallel read at ‘A’.

- In serial mode, register bits are read in or written out one bit per clock cycle, in serial order. Shown at point ‘B’ in the figure; therefore on an *N*-bit register, a complete serial read or write operation takes *N* clock cycles.

- Serial mode of data transfer is required if a device exchanges data only in serial mode – i.e. one bit at a time. Most communication links function in this mode.

- Thus, a register such as the one shown above can be operated in either

  ‘parallel in, serial out’ mode

  or

  ‘serial in, parallel out’ mode.

- A *shift register* is a register whose contents can be shifted left or right by a specified number of bits.

Memory elements

Usually made byte addressable for flexibility and efficiency in programming.

- *Byte addressable:* i.e. every *memory address* points to one eight-bit byte in the memory block.

- For faster operations, often such memory is also made capable of multi-byte operations.

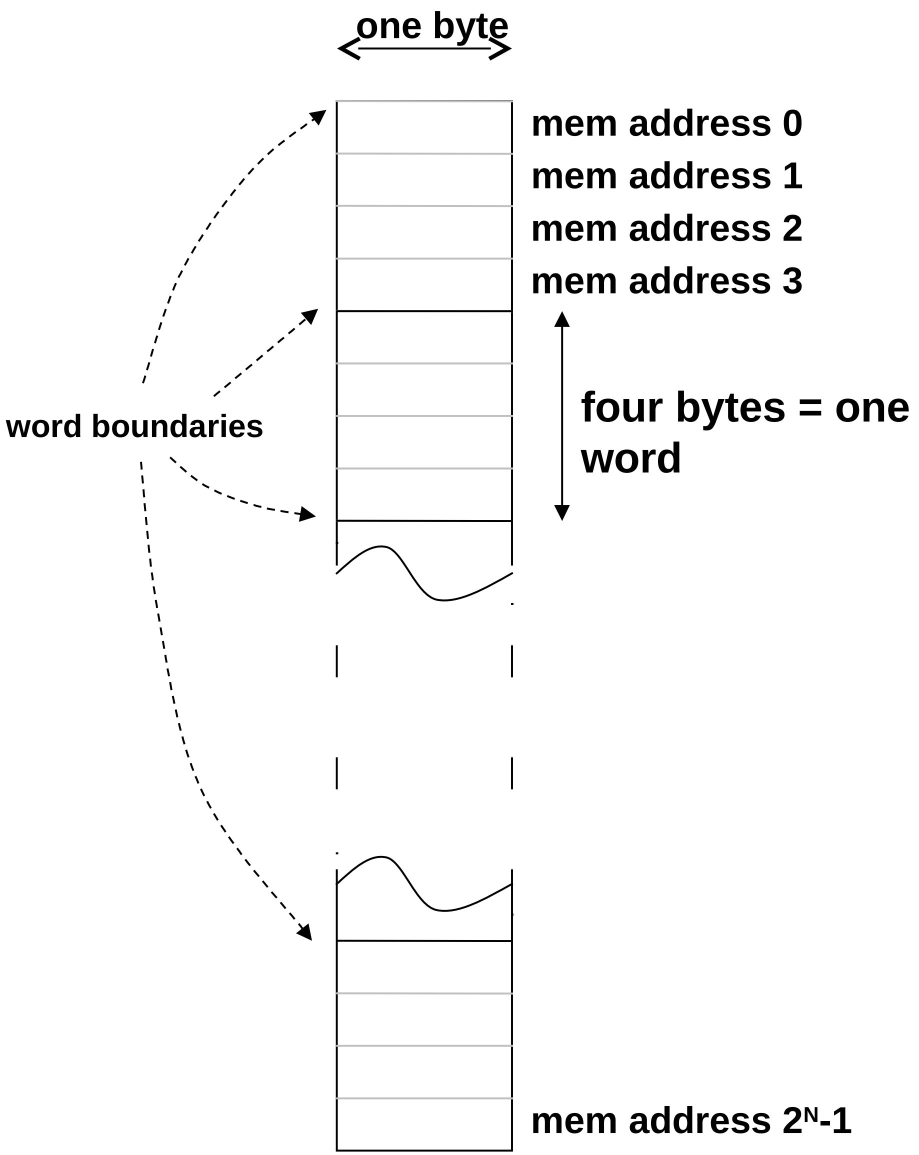
- The next slide shows the logical organization of byte addressable memory.

  – If a word equals four bytes, then every fourth byte of memory occurs at a *word boundary*. Counting memory bytes from 0 onwards, we then have word boundaries at the end of bytes 3, 7, 11, and so on.

  – Word operations on main memory are usually constrained to be aligned with word boundaries, and likewise for *half-word* and *double-word* operations.

- Any random byte or word in such a memory element can be accessed in one memory cycle. Therefore such memory is known as *random access memory*, or RAM.

Organization of byte-addressable memory

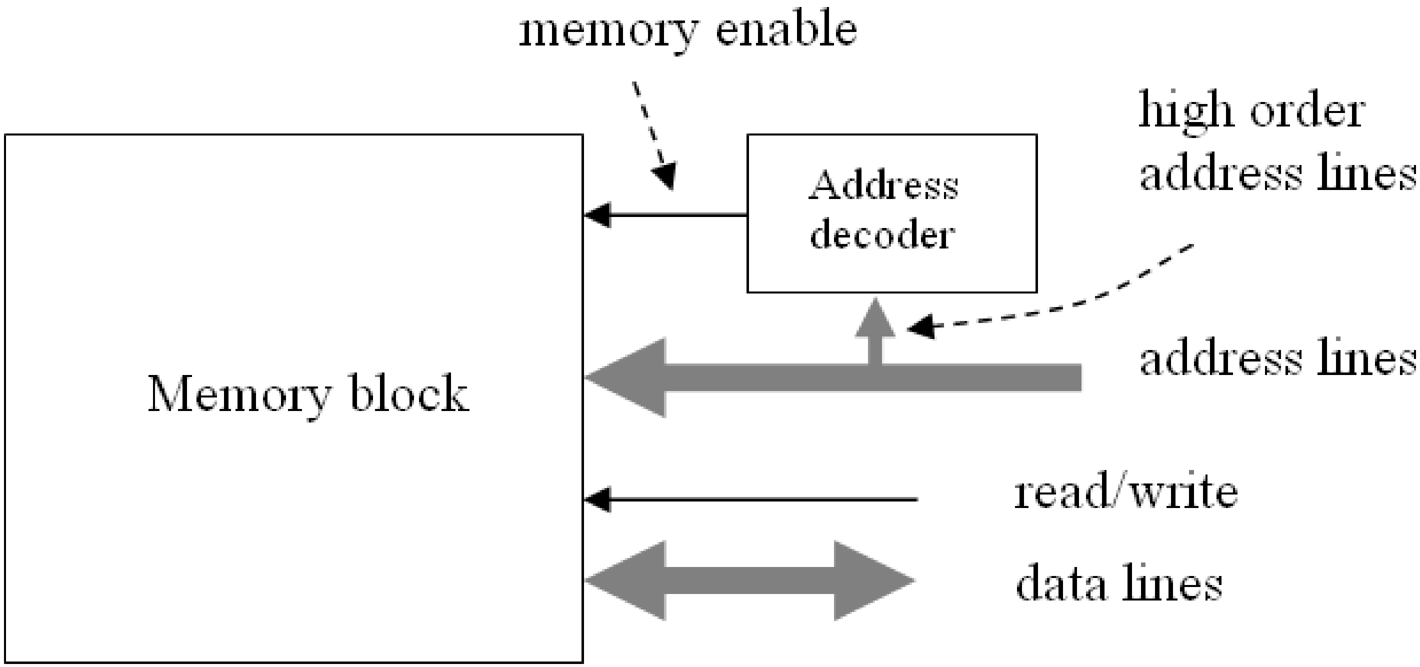


Input and output signals of a typical memory element are shown in the next slide. Note the *address lines*, *data lines*, and the *read/write* signal.

Other requirements:
(i) A memory element needs an *enable* signal, and
(ii) the memory is located at a specific *address*.

An address decoder element is a combinational circuit which generates the required *enable* signal.

This *enable* signal is generated when the higher order address lines have the value determined for this memory element. This results in the memory element being located at a specific address.


memory enable
high order
address lines
Address
decoder
Memory block
address lines
read/write
data lines

Types of memory elements

- Static memory element
  - faster, can hold contents indefinitely, relatively expensive, needs more chip area per bit
  - e.g. processor cache memory, register bank
- Dynamic memory element
  - relatively slower, needs periodic refresh, relatively inexpensive, needs less chip area per bit.
  - used in main memory.

Other useful elements:

- *Data bus* – Connects together the various system elements
- In one *bus cycle*, such a bus carries out a complete data transfer operation between any two elements connected to it.
- A typical bus may consist of data lines, address lines, read/write signal, various other control signals, clock signal, supply voltage, and ground.

- As seen above, address lines need to be decoded to generate the *enable* signal for a memory element.

- *Decoder* ICs simplify this work.

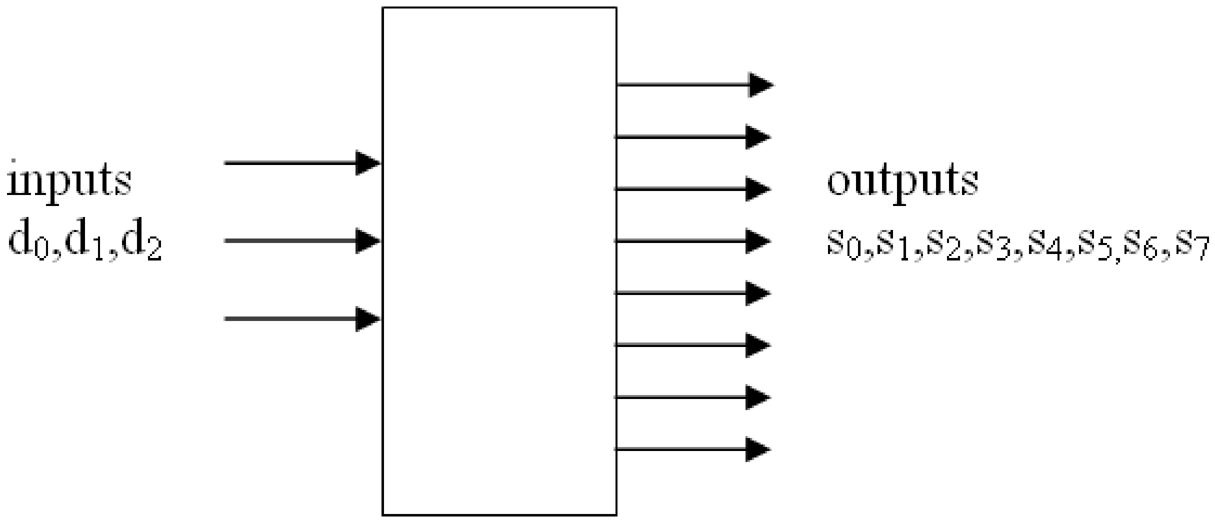
- The next slide shows a schematic diagram of the *3-to-8 decoder*

- The three inputs to such a decoder will have one of $2^3 = 8$ possible values, viz. 000, 001, 010, 011, 100, 101, 110 and 111.

- Accordingly, exactly one of the eight output lines will have value 1, while all others are 0.

- Thus the 3-bit binary number on the input determines which single one of the eight outputs has value 1.

- In the diagram, assume that the input & output signals are numbered in increasing order from top to bottom. The next slide shows the truth table of this device.



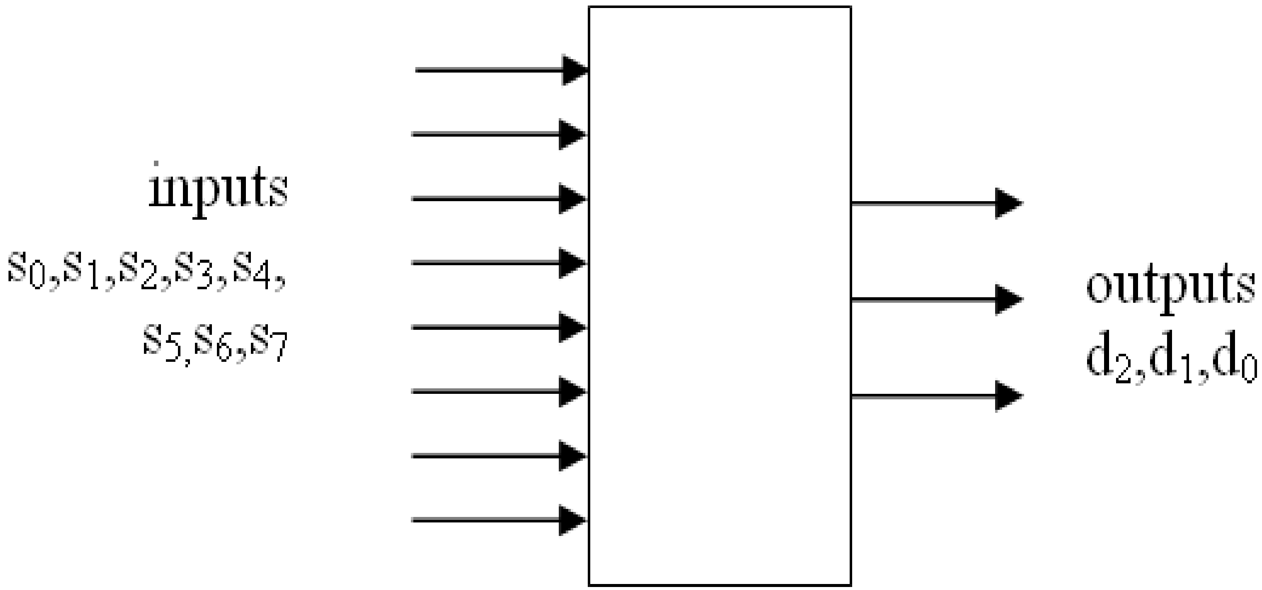
A 3-to-8 decoder

Inputs d_0,d_1,d_2	Outputs $s_0,s_1,s_2,s_3,s_4,s_5,s_6,s_7$
0,0,0	1,0,0,0,0,0,0,0
0,0,1	0,1,0,0,0,0,0,0
0,1,0	0,0,1,0,0,0,0,0
0,1,1	0,0,0,1,0,0,0,0
1,0,0	0,0,0,0,1,0,0,0
1,0,1	0,0,0,0,0,1,0,0
1,1,0	0,0,0,0,0,0,1,0
1,1,1	0,0,0,0,0,0,0,1

Truth table of the 3-to-8 decoder

- An *encoder* works in a manner which is exactly the converse of the *decoder*.
- An 8-to-3 encoder has eight inputs, exactly one of which must be set to 1 at any time.
- Its three output lines have a value from 000 to 111, depending on which input line is set to 1.

- The encoder must be operated under the constraint that exactly one of its inputs has value 1 at any time.



An 8-to-3 encoder

- A *comparator* is a combinational circuit which compares two binary values, say A and B, and generates output according to the result of the comparison.

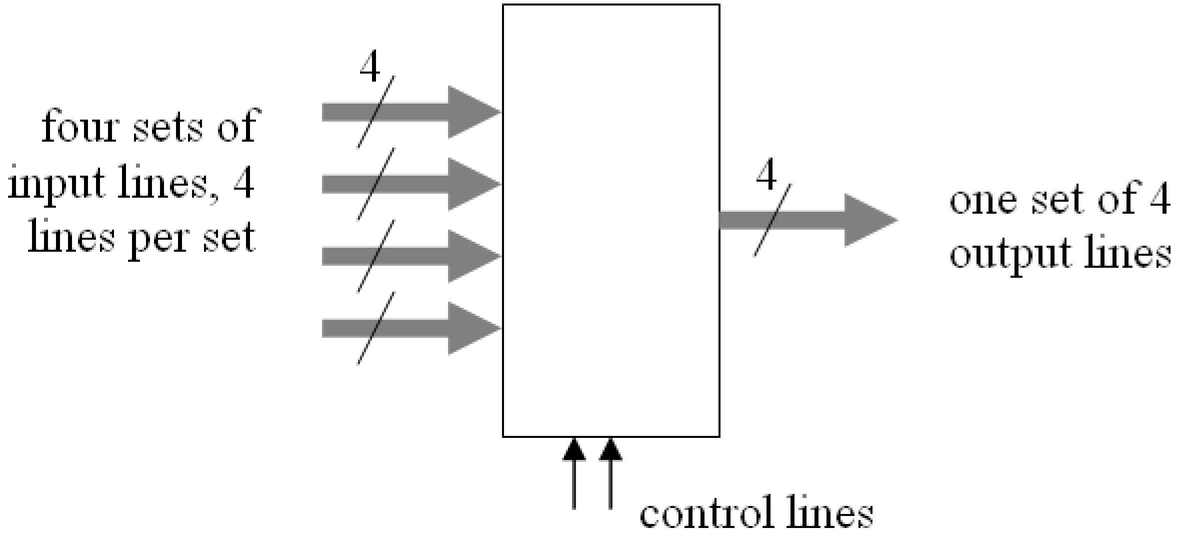
- The result on the output indicates one of the three possible results of comparison:

$$A < B, A = B, \text{ or } A > B.$$

- A *multiplexer* is a circuit with several sets of inputs, <u>exactly one of which</u> is connected to the outputs, depending on the control signals applied.

- The figure in the next slide shows a <u>4-line</u>, <u>4-to-1</u> multiplexer.

- The two control signals shown at the bottom of the device can have value 00, 01, 10 or 11.
- Data from the corresponding set of four input lines appears as the one set of four output lines.



A 4-line, 4-to-1 multiplexer

Truth table of a 4-to-1 multiplexer

Control inputs* c_0, c_1	Output set*
0,0	= input set A
0,1	= input set B
1,0	= input set C
1,1	= input set D

*Assume that, in the previous figure, control inputs are named c_0 and c_1, and data input sets are named A, B, C and D as we go from top to bottom

- In a *demultiplexer*, the single set of input lines is connected to exactly one out of several sets of output lines – as determined by the control signals.

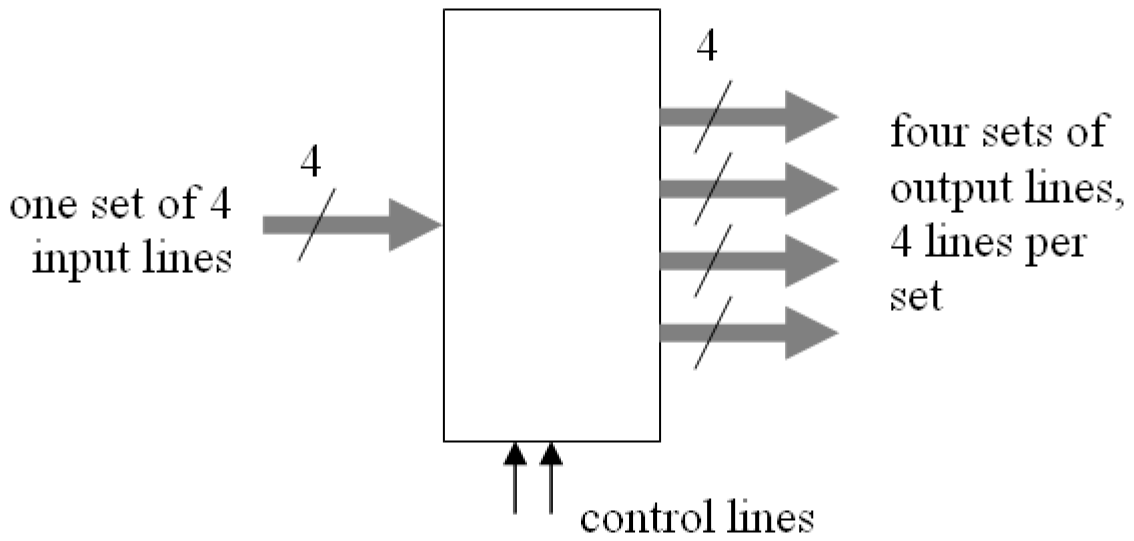
- The next slide shows a 4-line, 1-to-4 demultiplexer.

Control inputs* c_0, c_1	Input data appears on output set*
0,0	A
0,1	B
1,0	C
1,1	D

Truth table of a 1-to-4 demultiplexer

*Assume that, in the next figure, control inputs are named c_0 and c_1, and output sets are named A, B, C and D as we go from top to bottom.

- The two control signals shown at the bottom of the device can have value 00, 01, 10 or 11.
- Accordingly, data from the set of four input lines appears on exactly one of the four sets of output lines.

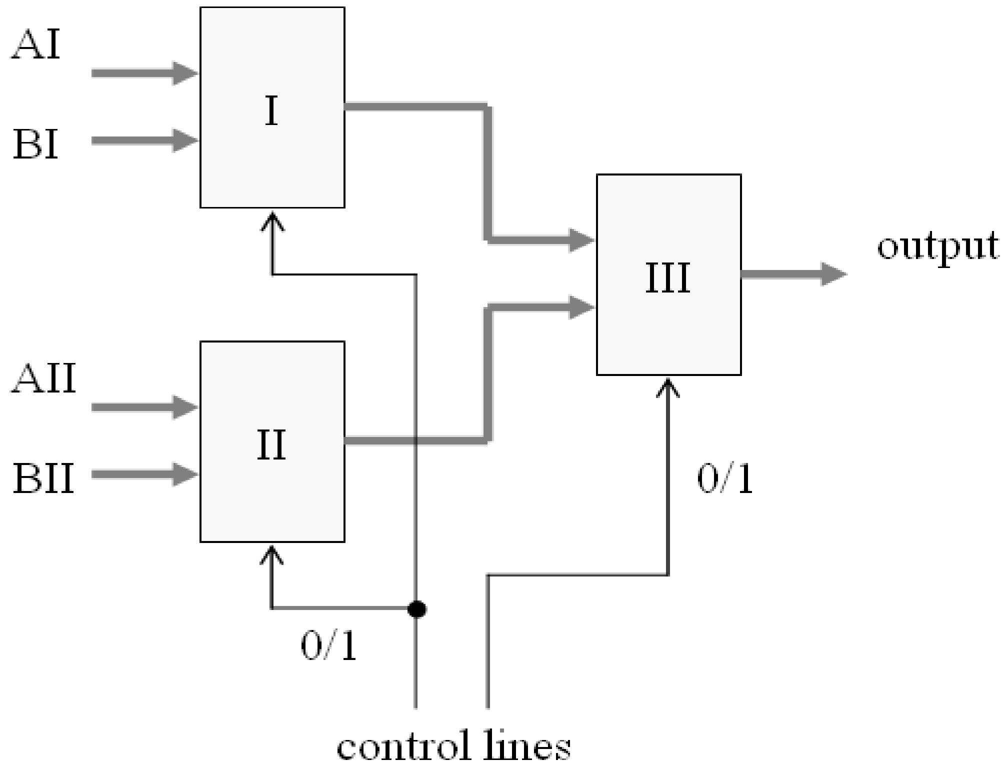


A 4-line, 1-to-4 demultiplexer

- A set of three 2-to-1 multiplexers can be connected together to achieve the functionality of a 4-to-1 multiplexer.
- This is shown schematically in the next slide.
- The four sets of inputs of the 4-to-1 multiplexer are denoted as A_I, B_I, A_{II} and B_{II}, respectively.
- Input sets A_I and B_I are connected to the inputs of 2-to-1 multiplexer I, while input sets A_{II} and B_{II} are connected to the inputs of multiplexer II.

- By applying the correct value (0 or 1) to the first control signal, we can select either
  - (i) A_I as the output of I and A_{II} as the output of II, or
  - (ii) B_I as the output of I and B_{II} as the output of II.

- Thus either (i) A_I and A_{II}, or (ii) B_I and B_{II} reach the input of multiplexer III.



Some other useful elements

- A *latch* is a flip-flop employed for capturing a single pulse or transition on a line
- An *8-bit latch* is a similar device with eight flip-flops
- A *buffer* is a device which strengthens the input signals without altering their logical value
- A *timer* device has an internal or external clock. Based on its pre-defined ‘timer count’, the device generates an output pulse once in that many clock periods

Summary

- F*unctional* point of view of Electronic technology – the building blocks of various sub-systems of a computer system
- Basic principles of the transistor serving as an ON-OFF switch, logic gates, combinational circuits, flip flops, and sequential circuits.
- Combinational function implemented using *and*, *or* and *not* gates
- Flip-flop - The basic circuit element which stores one bit of information
- Combinational and sequential circuits
- Integrated circuit technology – VLSI
- Crucial role of the clock signal in computer circuits
- Other useful building blocks of computer systems
  - registers and register banks, memory blocks, encoder, decoder, multiplexer, demultiplexer, and others.