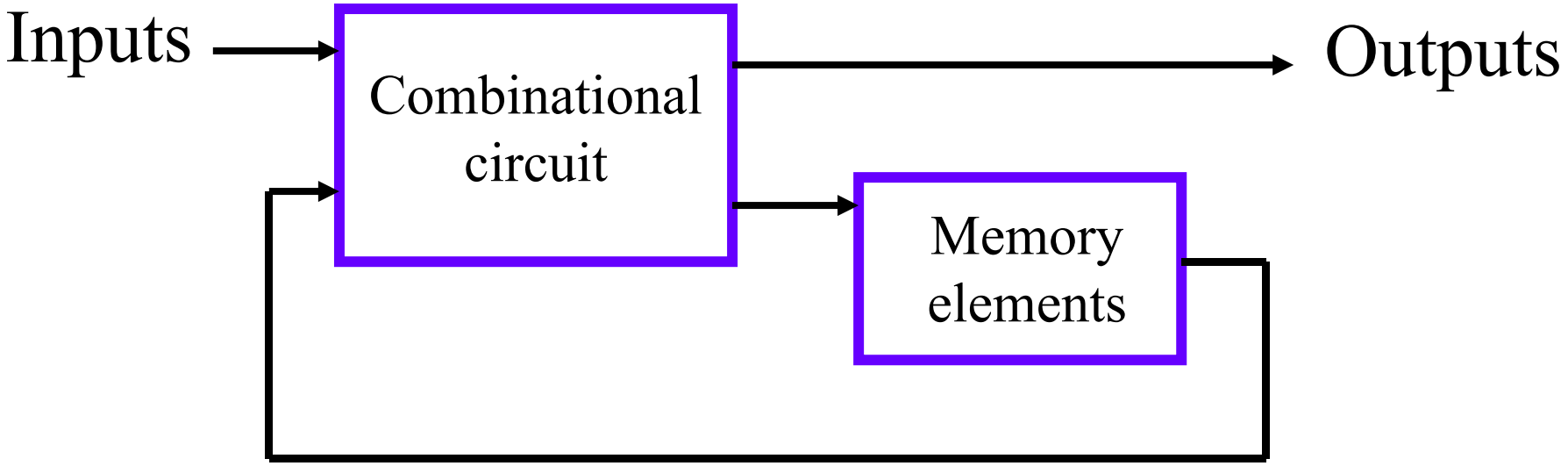
Sequential Logic Circuits

- Clock Signals
- Flip Flops
- Registers
- Design Flow

Sequential Logic

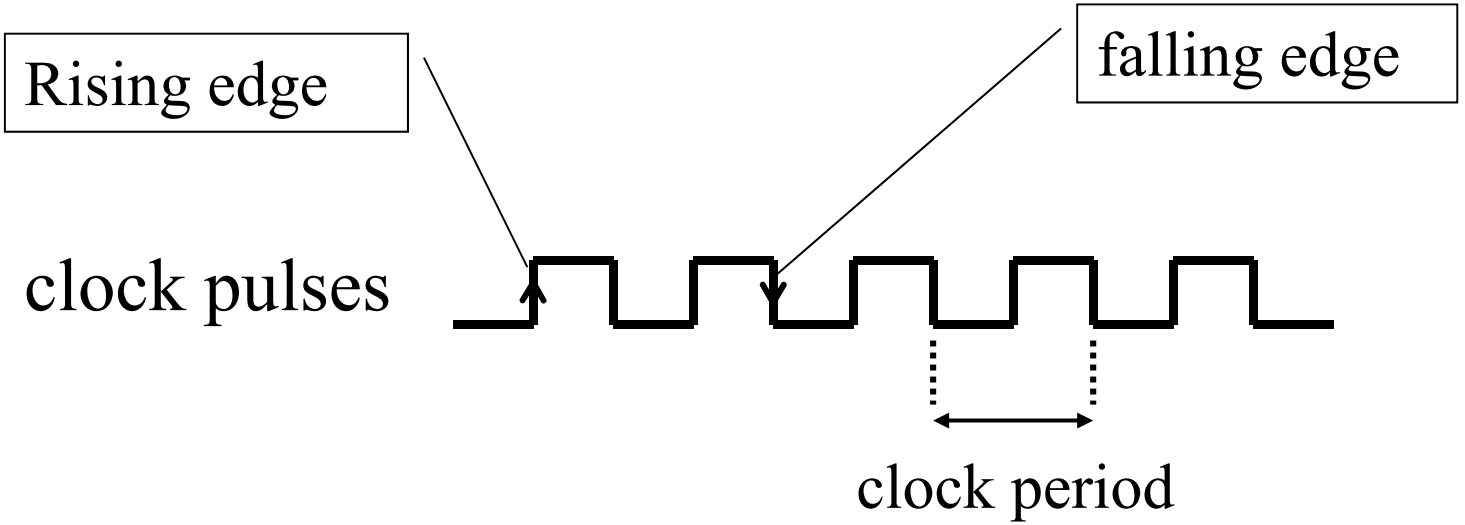
Sequential circuits contain memory elements which can store binary data. The information stored at any time defines the state of the circuit at that time.



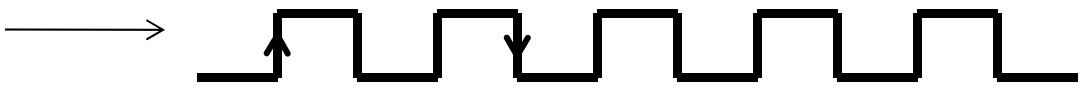
The inputs together with the present state of the storage elements, determines the binary value of the outputs.

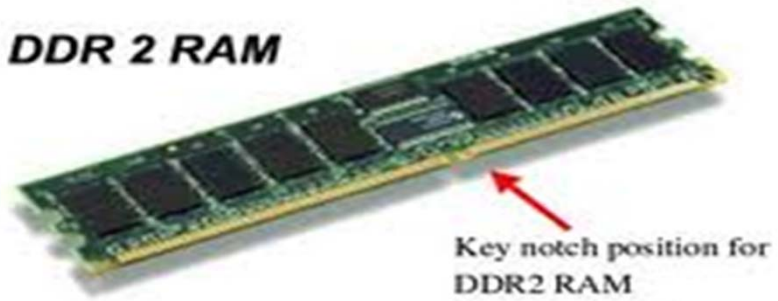
Synchronous Sequential Logic

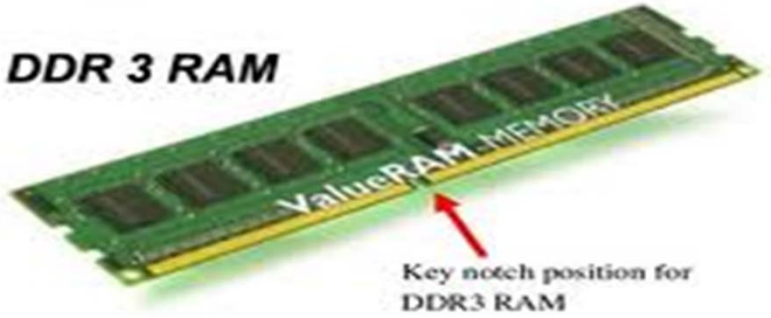
In a synchronous sequential circuit, changes occur at fixed points in time. These points are specified by the rising or falling edge of a **clock** signal.



The time between successive transitions in the same direction is known as the **clock period**. The reciprocal of of the clock period is known as the **clock frequency**.



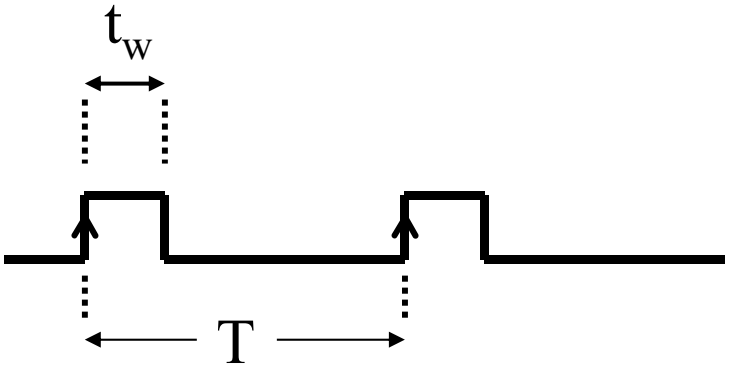




DDR SDRAM – double data rare synchronous DRAM

Period, Frequency, Duty Cycle

The frequency of the clock (f) is measured in cycles/second or Hertz.



$$f = \frac{1}{T}$$

$$T = \frac{1}{f}$$

1 KHz	10^3 Hz
1 MHz	10^6 Hz
1 GHz	10^9 Hz

The ratio of the pulse width (t_w) to the period (T) is known as the duty cycle.

$$\text{Duty cycle} = \frac{t_w}{T} * 100\%$$

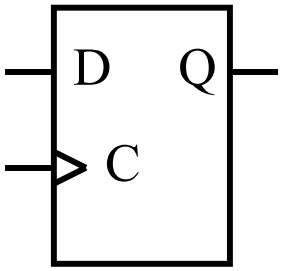
1 ms	10^{-3} s
1 μs	10^{-6} s
1 ns	10^{-9} s
1 ps	10^{-12} s

For pulse width of 25ns and a period of 100ns

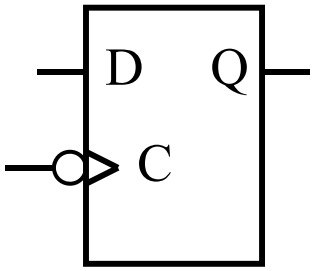
Frequency = 10 MHz , duty cycle = 25%

Flip-Flops

A flip-flop is a binary storage element capable of storing one bit of information, either a '**0**' or a '**1**'. There are several types but all can be constructed from the basic D-Type with some additional combinational logic.



rising edge triggered



falling edge triggered

D	$Q_{(next)}$
0	0
1	1

characteristic table

$Q_{(next)} = D$

characteristic equation

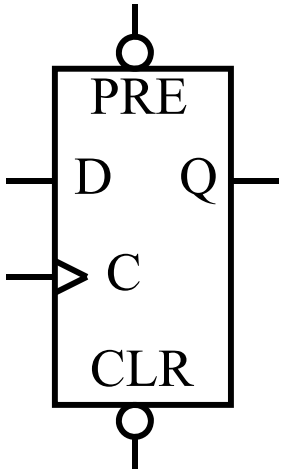
The flip-flop stores the value on the D input when an edge is present on the clock line. The output remains unchanged at other times.

Direct Inputs

It may be necessary to force flip-flops into a known state independent of the clock. For example, when power is turned on in a digital system, the state of the flip-flops is unknown.

preset (direct set) - sets the flip-flop to 1

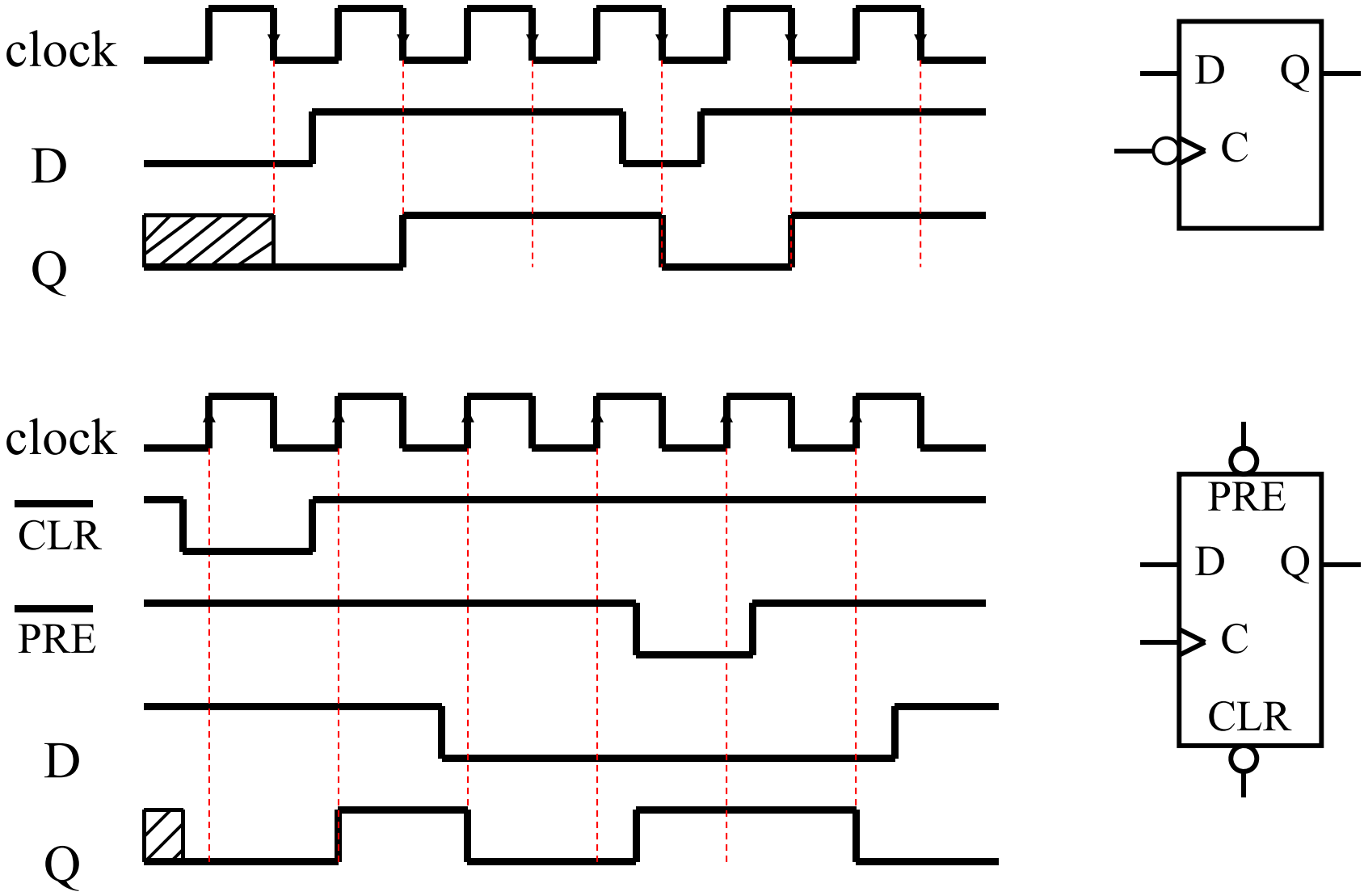
clear (direct reset) - sets the flip-flop to 0



PRE	CLR	Q
0	0	unstable
0	1	1
1	0	0
1	1	clocked operation

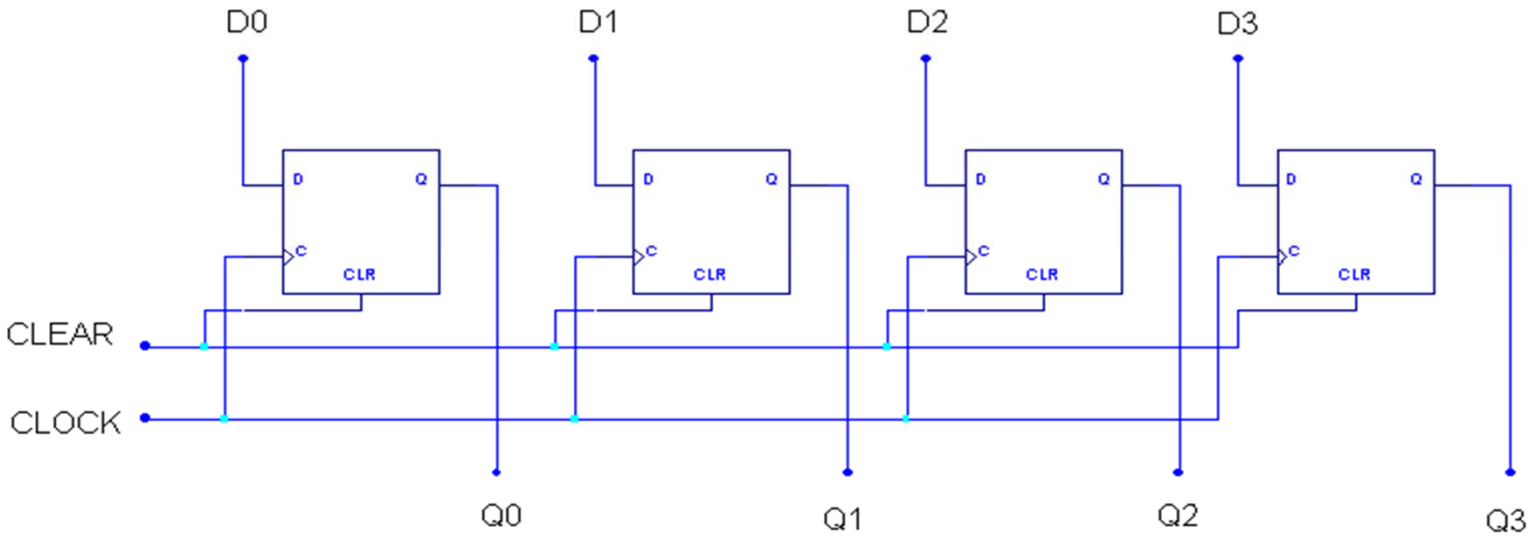
These are asynchronous inputs. A bubble on P or R indicates that the preset or clear is applied for a logic level ‘0’, i.e. active low.

D flip-flop Timing Diagrams



Registers

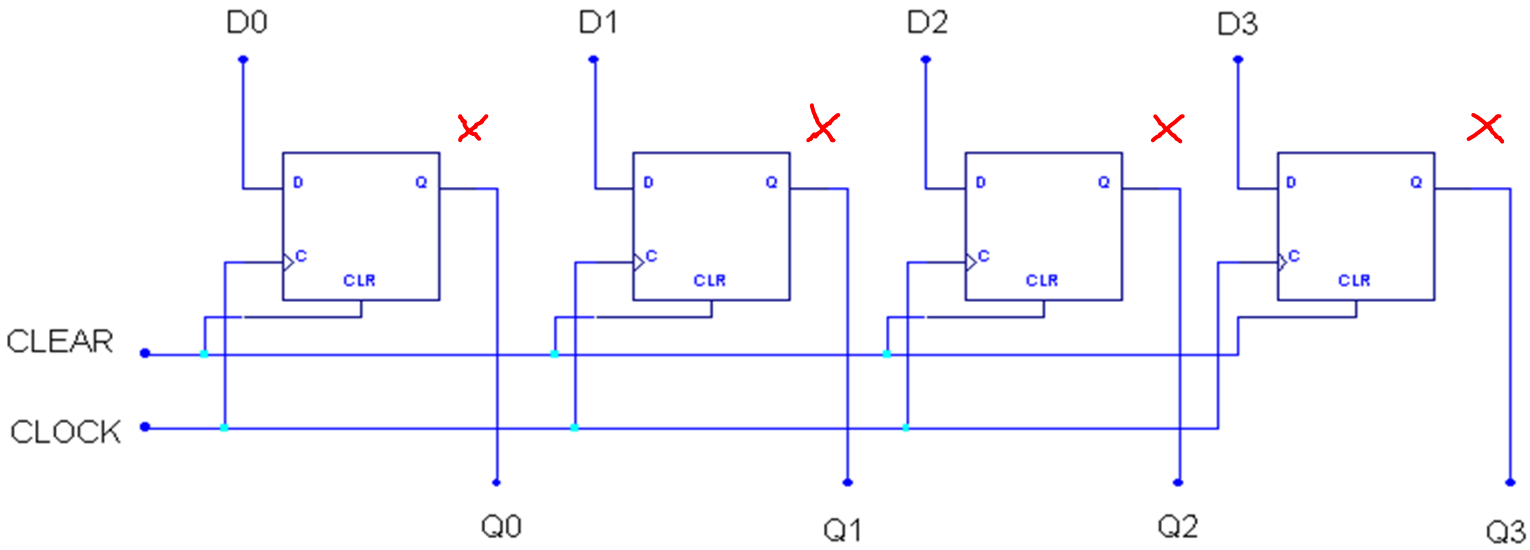
An n-bit register consists of a group of n flip-flops and can store n-bits of binary information.



The common clock triggers all flip-flops, on its rising edge in this case, transferring the binary data on the four inputs into the register. This is known as a parallel load. A '1' on the common clear line will reset all outputs asynchronously.

Registers

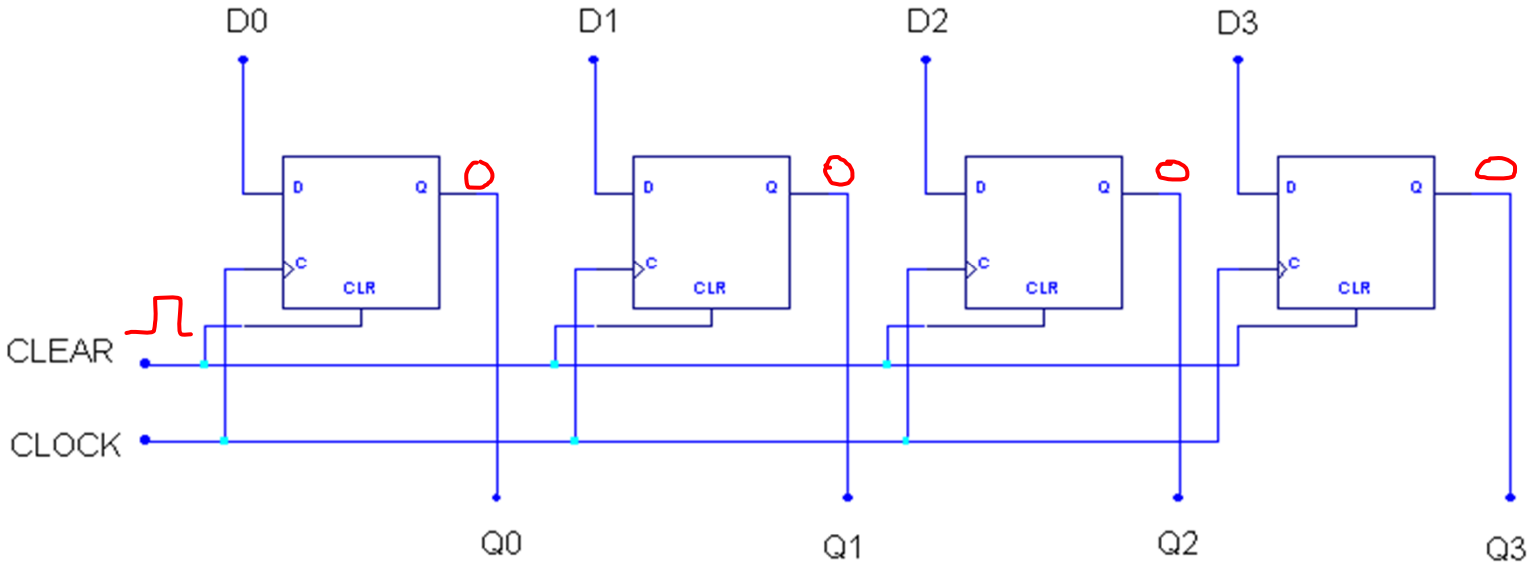
An n-bit register consists of a group of n flip-flops and can store n-bits of binary information.



The common clock triggers all flip-flops, on its rising edge in this case, transferring the binary data on the four inputs into the register. This is known as a parallel load. A '1' on the common clear line will reset all outputs asynchronously.

Registers

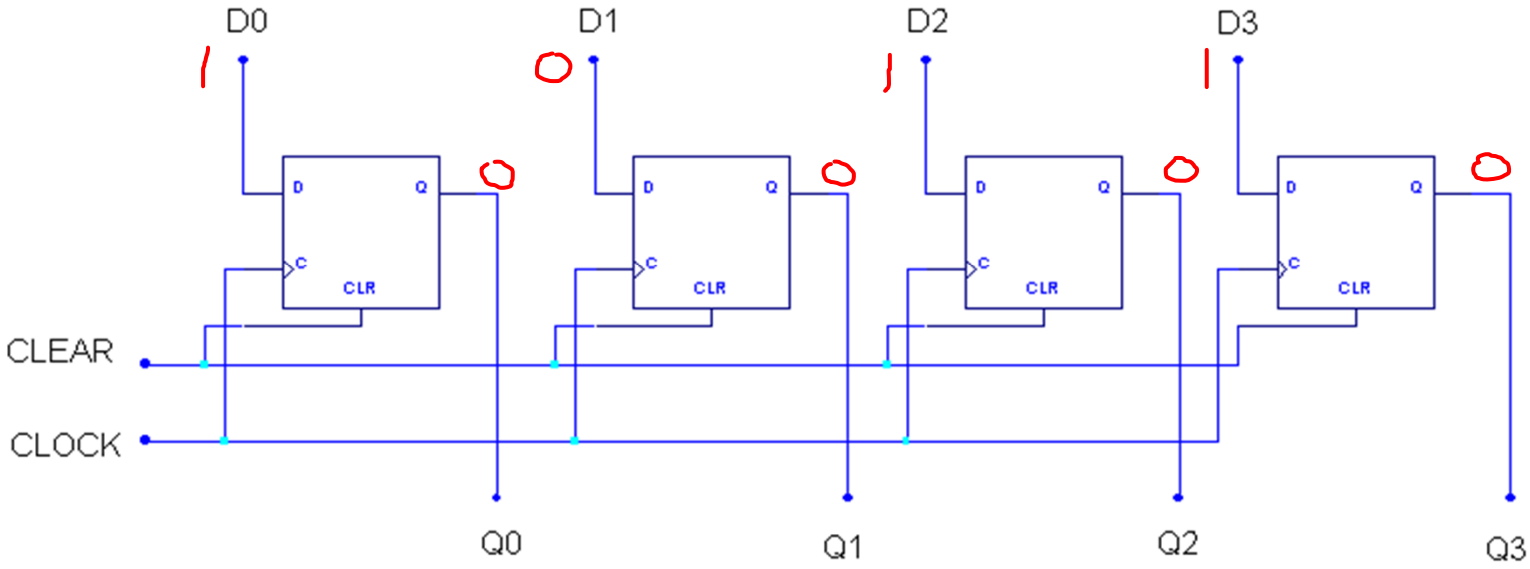
An n-bit register consists of a group of n flip-flops and can store n-bits of binary information.



The common clock triggers all flip-flops, on its rising edge in this case, transferring the binary data on the four inputs into the register. This is known as a parallel load. A '1' on the common clear line will reset all outputs asynchronously.

Registers

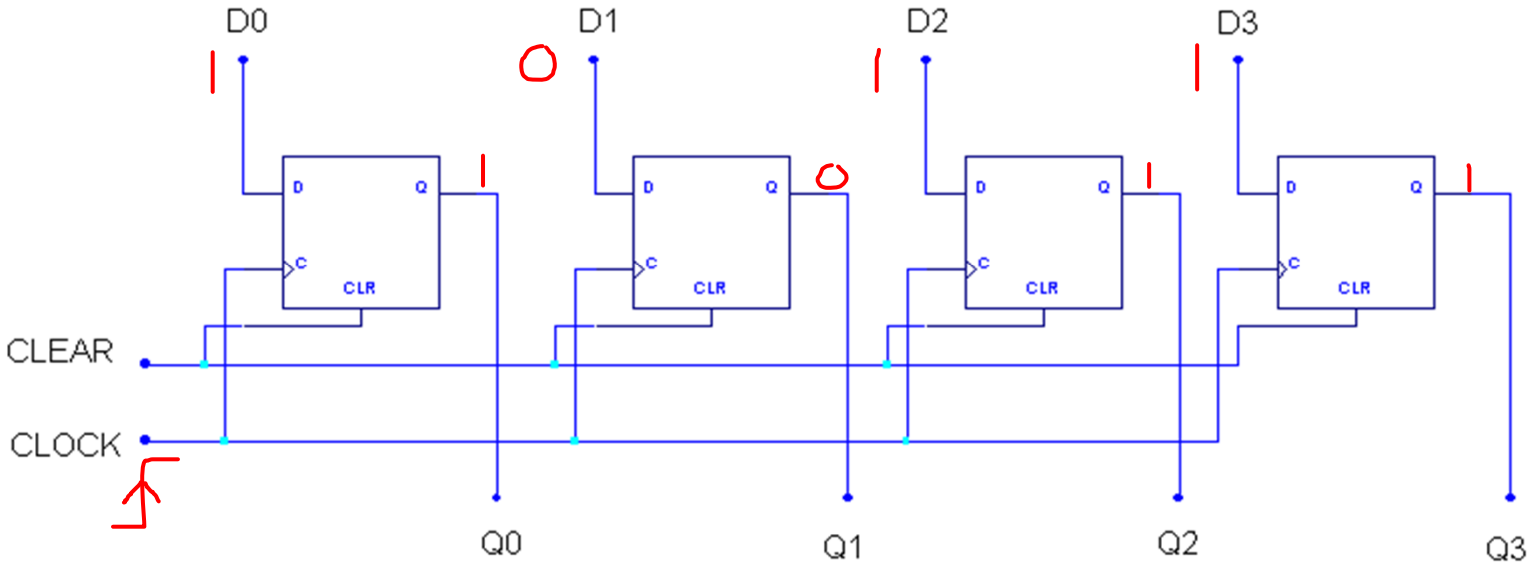
An n-bit register consists of a group of n flip-flops and can store n-bits of binary information.



The common clock triggers all flip-flops, on its rising edge in this case, transferring the binary data on the four inputs into the register. This is known as a parallel load. A '1' on the common clear line will reset all outputs asynchronously.

Registers

An n-bit register consists of a group of n flip-flops and can store n-bits of binary information.

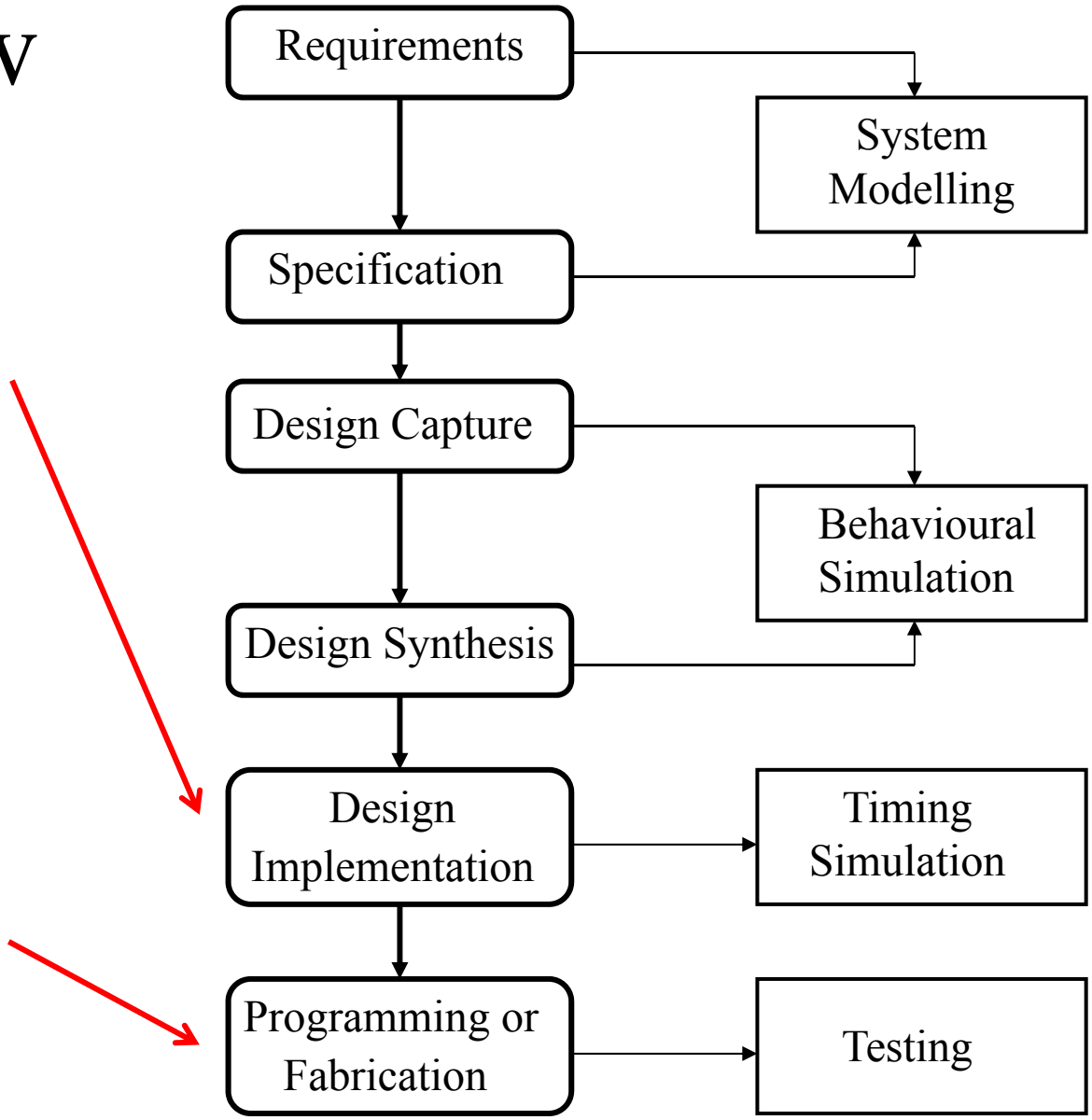


The common clock triggers all flip-flops, on its rising edge in this case, transferring the binary data on the four inputs into the register. This is known as a parallel load. A '1' on the common clear line will reset all outputs asynchronously.

Design Flow

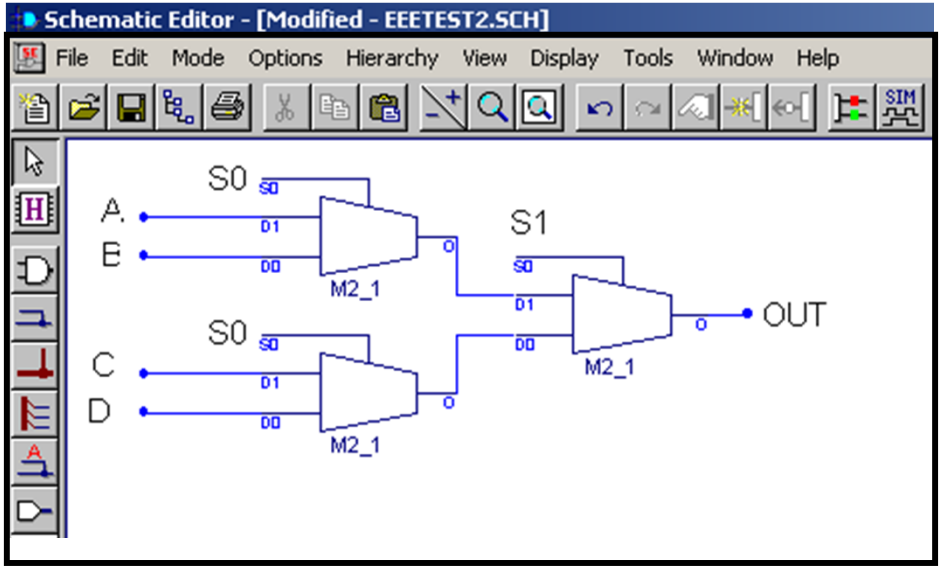
The design can now be implemented for a chosen target technology. A synthesis tool will generate the required implementation data.

When the results from simulation are satisfactory, the device can be fabricated. It may have been necessary to undergo many iterations of the design process to reach this point.



Design Capture

The design is entered into a CAD tool. It may be entered graphically using schematic capture or textually using a hardware description language such as Verilog.




```
Untitled - HDL Editor
File  Edit  Search  View  Synthesis  Project  Tools  Help

 1
 2  module mux3( select, d, q );
 3
 4
 5  input[1:0] select;
 6  input[3:0] d;
 7
 8  output q;
 9
10  reg q;
11  wire[1:0] select;
12  wire[3:0] d;
13
14
15
16  always @( select or d )
17  begin
18    if( select == 0)
19      q = d[0];
20
21
22    if( select == 1)
23      q = d[1];
24
25
26    if( select == 2)
27      q = d[2];
28
29
30    if( select == 3)
31      q = d[3];
32
33  end
34
35
36  endmodule
37
38
```

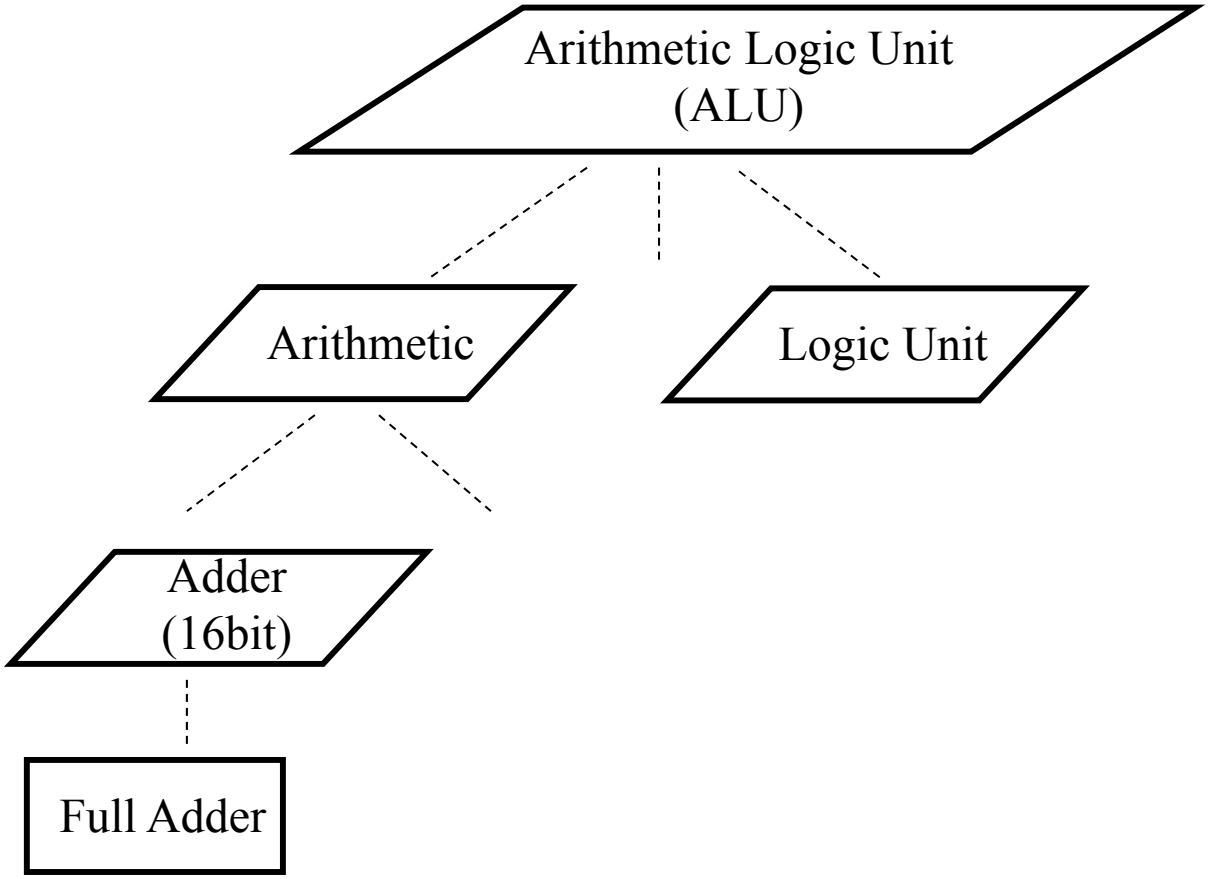
Verilog

- Verilog is a Hardware Description Language (HDL)
- Describes electronic systems in a textual form
- Documents designs
- Technology Independent
- It is an IEEE standard – number 1364

Important ! - You are only required to be able to read and understand a Verilog description.

Hierarchical Design

Hierarchy is used in digital design to **divide-and-conquer** a problem. The design is partitioned into components and subcomponents.

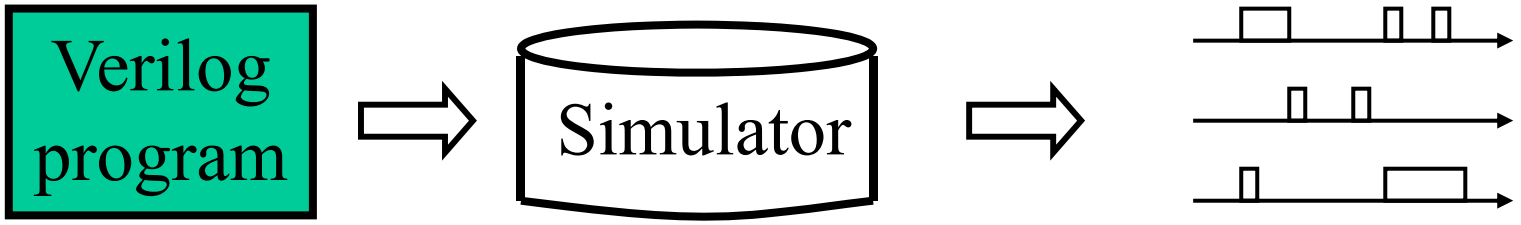


Subcomponents are designed and tested in isolation before being used in a higher level of the hierarchy. This is simpler than system testing and usually more thorough.

Subcomponents can be stored in libraries and used by other designers.

Simulation

Verilog can be used to describe digital systems. The Verilog model of a system can be executed by a CAD tool called a *simulator*.

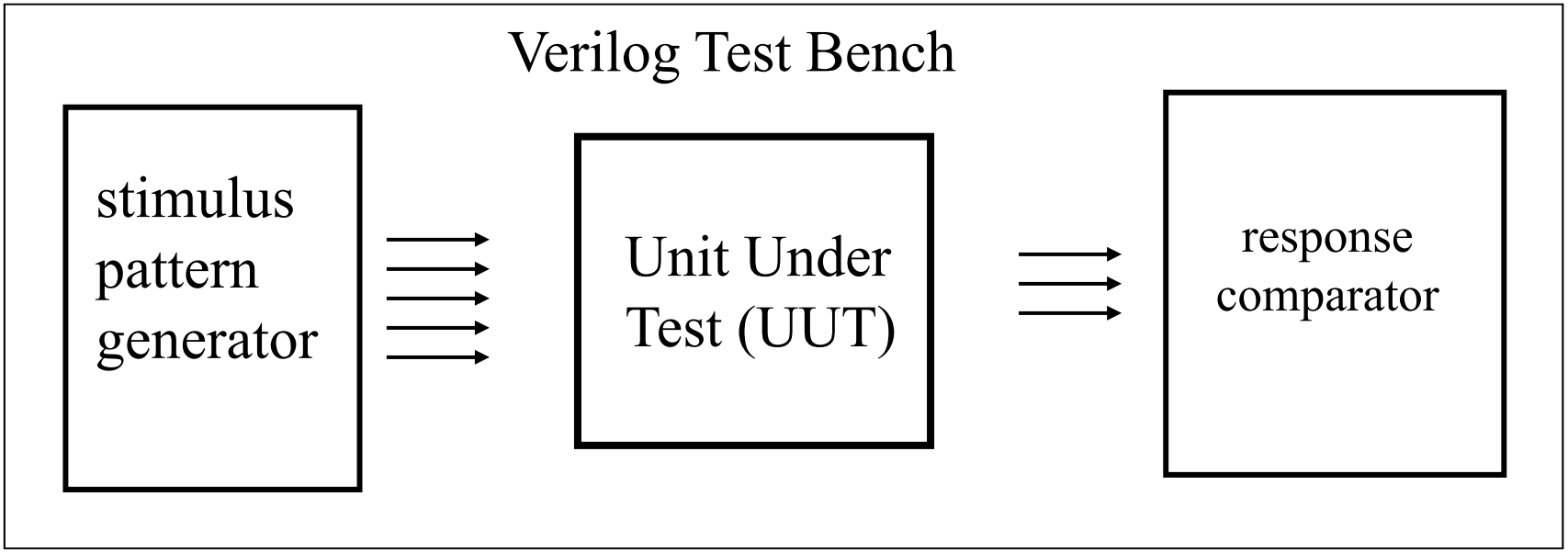


The simulator computes the outputs of the modelled system in response to a series of inputs applied over time. Simulation enables a designer to verify the design before committing to a hardware implementation.

The model can be progressively expanded and refined using the simulator to verify each design iteration.

Test Benches

The input values used to stimulate the design can be written in Verilog. This description is known as a Test Bench. The test bench can also check to see if the simulated outputs give the expected output value.



An exhaustive simulation covers all possible input combinations. For n inputs this gives 2^n test vectors.

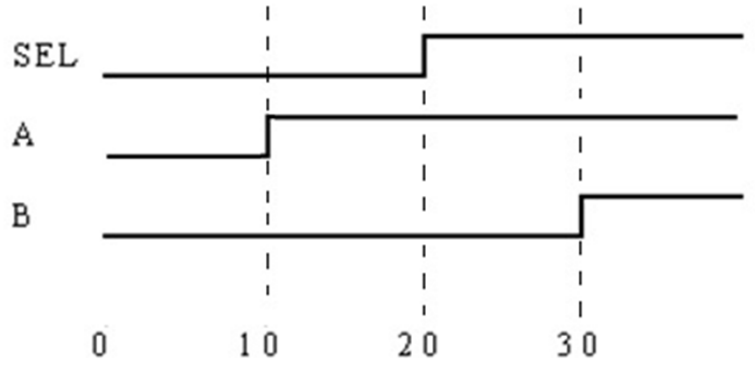
Verilog Stimulus File

1. Repetitive Waveform Generation

```
initial // initial blocks execute once starting at time 0
begin
    sys_clk = 0;
    forever #10 sys_clk = !sys_clk;      // clock changes state every 10 time units
end                                      //  the verilog NOT operator is !
```

2. Stimulus Generation

```
initial // test stimulus
begin
  SEL = 0; A = 0; B = 0;  // time 0
  #10   A = 1;
  #10   SEL = 1;
  #10   B = 1;
end
```



Summary

- Sequential digital circuits contain storage elements known as flip-flops.
- The system outputs depend upon current and previous inputs.
- Binary data can be stored in registers made from flip-flops.