**Objective:**

* Gain a practical understanding of State Diagrams and State Table
* Designing and implementation of a Synchronous Sequential Circuit given a State Diagram
* Understanding the concept of designing Sequential Circuits using Flip-Flop.

**List of Equipment**

* 1 \* IC 74107 JK Flip-Flop
* 1 \* IC 7408 2-input AND gates
* 1 \* IC 7404 Hex inverters (NOT gates)
* 1 \* IC 7432 2-input OR gates
* 1 \* IC 7474 Dual D Flip-Flop
* Trainer board
* Wires
* IC Extractor

**Theory:**

Synchronous Sequential Circuits:

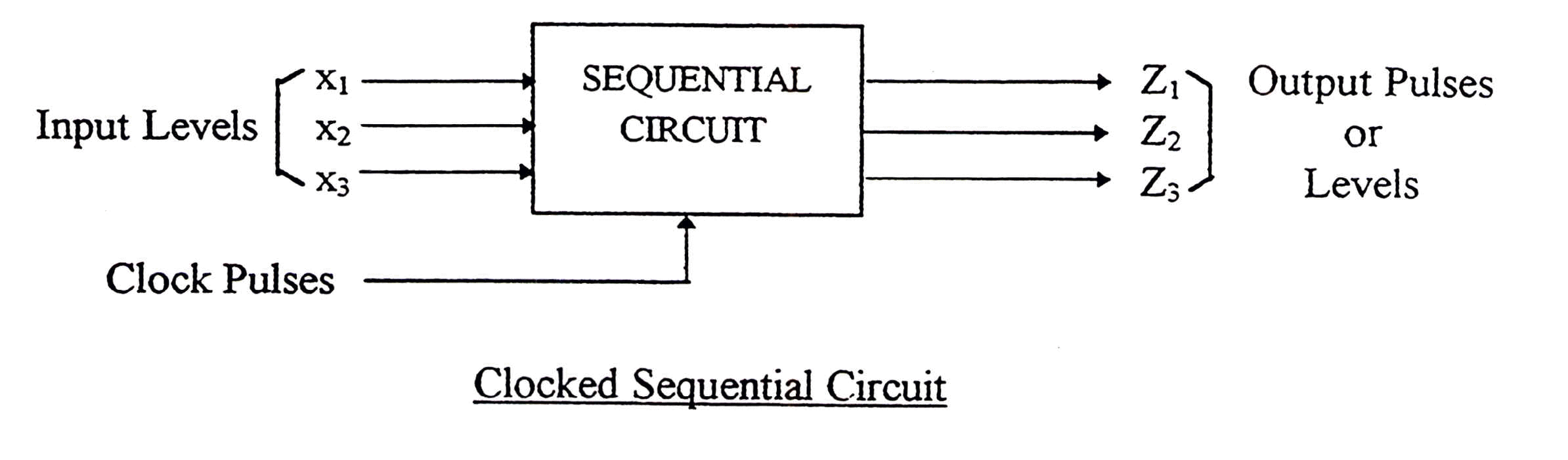
A synchronous circuit is a [digital circuit](https://en.wikipedia.org/wiki/Digital_circuit) in which the changes in the [state](https://en.wikipedia.org/wiki/State_(computer_science)) of memory elements are synchronized by a [clock signal](https://en.wikipedia.org/wiki/Clock_signal). In a [sequential](https://en.wikipedia.org/wiki/Sequential_logic) [digital logic](https://en.wikipedia.org/wiki/Digital_logic) circuit, data is stored in memory devices called [flip-flops](https://en.wikipedia.org/wiki/Flip-flop_(electronics)) or latches. The output of a flip-flop is constant until a pulse is applied to its "clock" input, upon which the input of the flip-flop is latched into its output. In a synchronous logic circuit, an [electronic oscillator](https://en.wikipedia.org/wiki/Electronic_oscillator) called the [*clock*](https://en.wikipedia.org/wiki/Clock_(computing)) generates a string of pulses, the "clock signal". This clock signal is applied to every storage element, so in an ideal synchronous circuit, every change in the [logical levels](https://en.wikipedia.org/wiki/Logic_level) of its storage components is simultaneous. Ideally, the input to each storage element has reached its final value before the next clock occurs, so the behavior of the whole circuit can be predicted exactly. Practically, some delay is required for each logical operation, resulting in a maximum speed at which each synchronous system can run.

To make these circuits work correctly, a great deal of care is needed in the design of the [Clock Distribution Networks](https://en.wikipedia.org/wiki/Clock_Distribution_Networks). [Static timing analysis](https://en.wikipedia.org/wiki/Static_timing_analysis) is often used to determine the maximum safe operating speed.

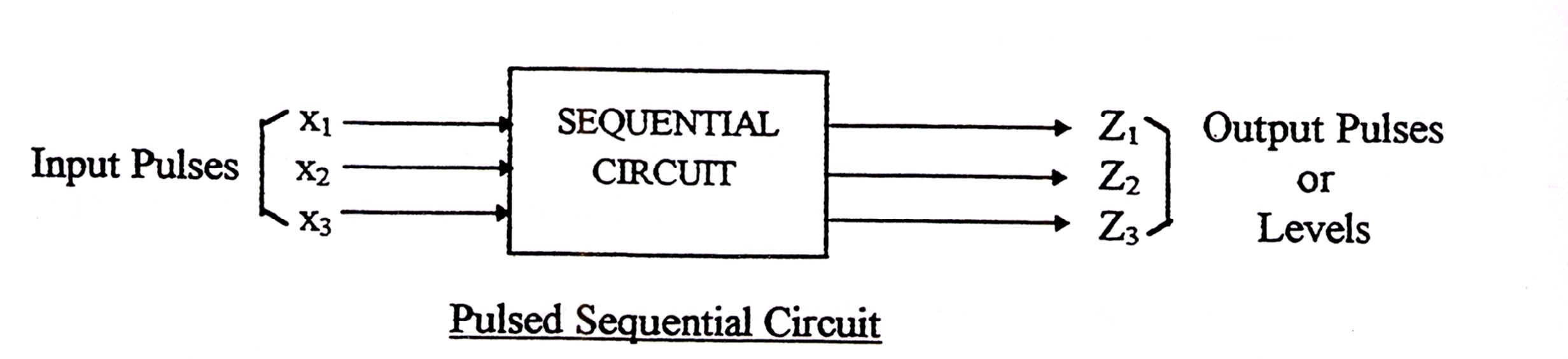
In synchronous circuits the input are pulses (or levels and pulses) with certain restrictions on pulse width and circuit propagation delay. Therefore synchronous circuits can be divided into clocked sequential circuits and uncklocked or pulsed sequential circuits.

In a clocked sequential circuit which has flip-flops or, in some instances, gated latches, for its memory elements there is a (synchronizing) periodic clock connected to the clock inputs of all the memory elements of the circuit, to synchronize all internal changes of state.

Hence the operation of the entire circuit is controlled and synchronized by the periodic pulses of the clock.



On the other hand in an unclocked or pulsed sequential circuit, such a clock is not present. Pulse mode circuits require two consecutive transitions between 0 and 1 - that is a 0-pulseor a 1 pulse to alter the circuit’s state. A pulse -mode circuit is designed to respond to pulses of certain duration; the constant signals between the pulses are “null” or “spacer” signals, which do not affect the circuit’s behavior



From the above block diagrams we can note the following:

1)     Pulse outputs: For pulsed sequential circuits these occur only for the duration of the respective input pulse and in some cases for duration considerably less. For clocked sequential circuits these outputs occur for the duration of the clock pulse.

2)     Level outputs: These change state at the start of the respective input or clock pulse and remain in that state until the next state of output is required.

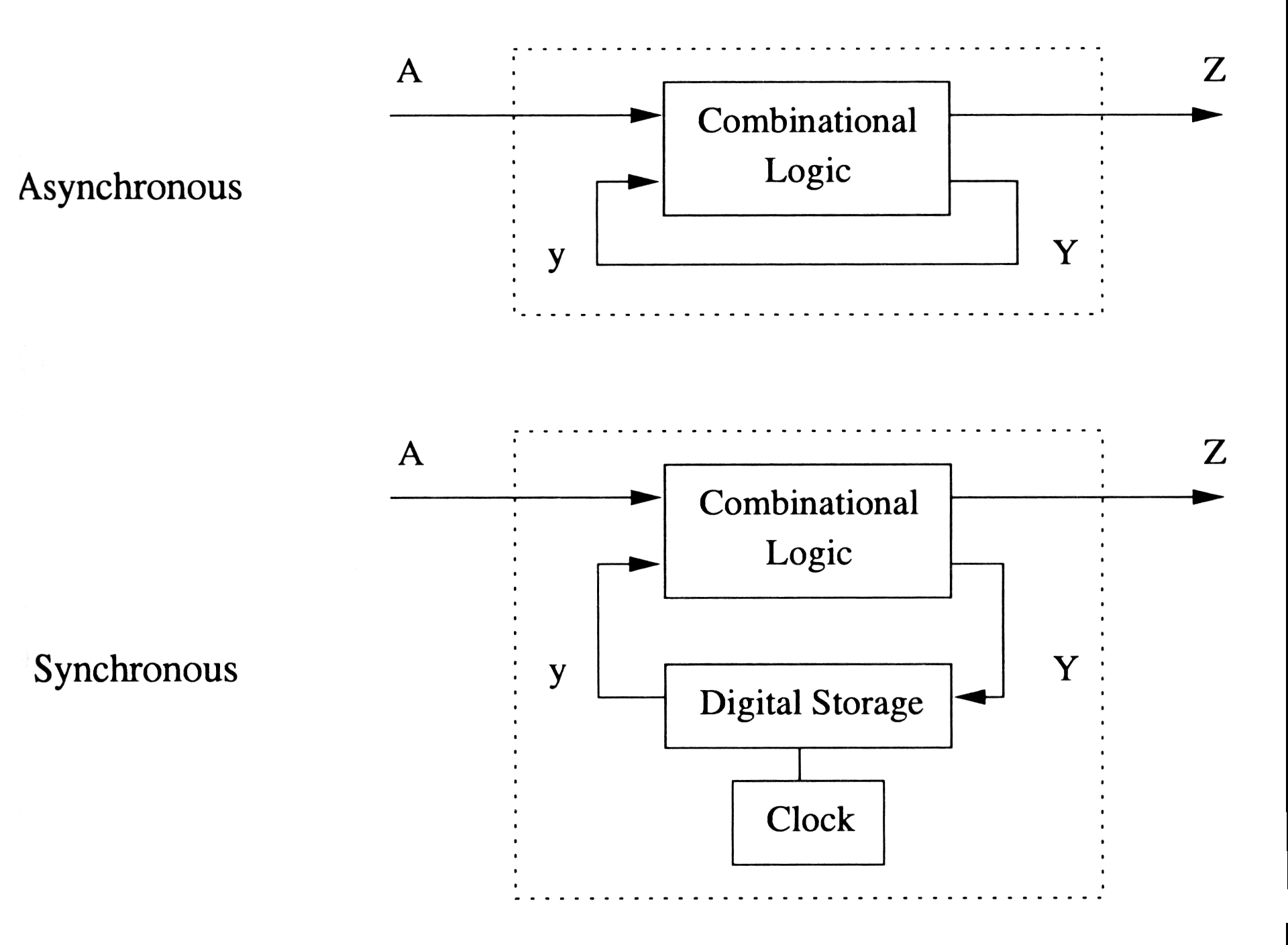
A requirement of synchronous sequential circuits is that the duration of the activating pulse or clock pulse should be sufficiently low in value that the pulse (or clock) has disappeared by the time the secondaries (the flip-flops outputs) have taken on their new value; otherwise the circuit will change state again. This means that the storage elements (flip-flops) should be edge-triggered devices (for example: D-type flip-flop, the JK flip-flop and their derivatives).

#### Asynchronous circuits:

The circuit is considered to be asynchronous if it does not employ a periodic clock signal C to synchronize its internal changes of state. Therefore the state changes occur in direct response to signal changes on primary (data) input lines, and different memory elements can change state at different times.

In asynchronous sequential circuits the inputs are levels and there are no clock pulses; the inputs events drive the circuit.

In general, an asynchronous circuit does not need the precise timing control supported by flip-flops. It may therefore contain latches rather than flip-flops. In many cases, an asynchronous circuit simply relies on the propagation delays of its component gates and connections, combined with the circuit’s feedback structure, to implement its memory functions.



**State Table**

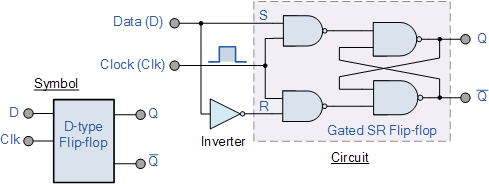
The state table representation of a sequential circuit consists of three sections labeled *present state*, *next state* and *output*. The present state designates the state of flip-flops before the occurrence of a clock pulse. The next state shows the states of flip-flops after the clock pulse, and the output section lists the value of the output variables during the present state.

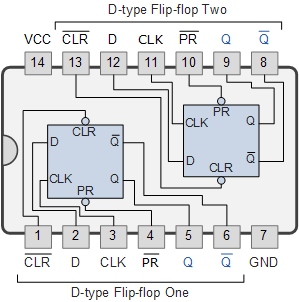
**State Diagram**

In addition to graphical symbols, tables or equations, flip-flops can also be represented graphically by a state diagram. In this diagram, a state is represented by a circle, and the transition between states is indicated by directed lines (or arcs) connecting the circles.

IC 7474 (Dual D Flip-Flop):

### D-type Flip-Flop Circuit



We remember that a simple SR flip-flop requires two inputs, one to “SET” the output and one to “RESET” the output. By connecting an inverter (NOT gate) to the SR flip-flop we can “SET” and “RESET” the flip-flop using just one input as now the two input signals are complements of each other. This complement avoids the ambiguity inherent in the SRlatch when both inputs are LOW, since that state is no longer possible.

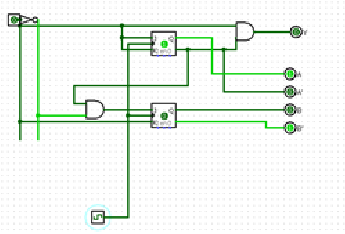
Thus this single input is called the “DATA” input. If this data input is held HIGH the flip flop would be “SET” and when it is LOW the flip flop would change and become “RESET”. However, this would be rather pointless since the output of the flip flop would always change on every pulse applied to this data input.

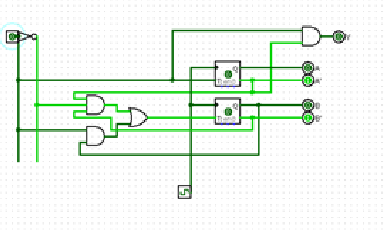
To avoid this an additional input called the “CLOCK” or “ENABLE” input is used to isolate the data input from the flip flop’s latching circuitry after the desired data has been stored. The effect is that D input condition is only copied to the output Q when the clock input is active. This then forms the basis of another sequential device called a **D Flip Flop**.

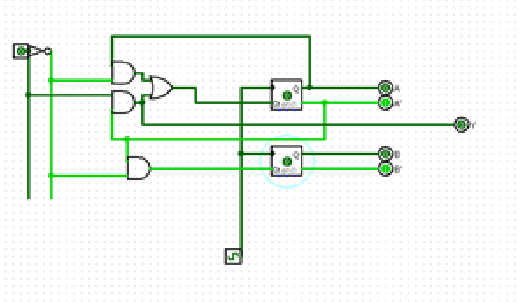
The “D flip flop” will store and output whatever logic level is applied to its data terminal so long as the clock input is HIGH. Once the clock input goes LOW the “set” and “reset” inputs of the flip-flop are both held at logic level “1” so it will not change state and store whatever data was present on its output before the clock transition occurred. In other words the output is “latched” at either logic “0” or logic “1”.

IC 74107 (Dual JK Flip-Flop):

**Circuit Diagram:**







**Data/Truth table:**

Constructing a Sequential Circuit using JK Flip-Flops:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Present  State |  | Input | Next  State | Output |  | Flip-Flops input Functions |  |  |  |
| A | B | X | A | B | Y | JA | KA | JB | KB |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | X | 1 | X |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | X | 0 | X |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | X | X | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | X | X | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | X | 0 | 0 | X |
| 1 | 0 | 1 | 0 | 0 | 0 | X | 1 | 0 | X |
| 1 | 1 | 0 | X | X | X | X | X | X | X |
| 1 | 1 | 1 | X | X | X | X | X | X | X |

Table: State Table for circuit using JK Flip-flops

|  |  |  |  |
| --- | --- | --- | --- |
| X | X | X | X |
| 0 | 1 | X | x |

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 1 | 0 |
| X | X | X | x |

**KA=X**

**JA=X**

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 0 | X | X |
| 0 | 0 | X | X |

**JB=A’X’**

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 1 | 0 |
| 0 | 0 | X | x |

|  |  |  |  |
| --- | --- | --- | --- |
| X | X | 1 | 0 |
| X | X | X | x |

**Y=A’X**

**KB=X**

Constructing a Sequential Circuit using T Flip-Flops:

Flip-Flops input Functions

Input

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Present  State |  |  | Next  State | Output |  |  |  |
| A | B | X | A | B | Y | TA | TB |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | X | X | X | X | X |
| 1 | 1 | 1 | X | X | X | X | X |

Table: State Table for circuit using JK Flip-flops

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 0 | 1 | 0 |
| 0 | 0 | X | x |

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 1 | 0 |
| 0 | 1 | X | x |

**TB=BX+A’B’X’**

**TA=X**

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 1 | 0 |
| 0 | 0 | X | X |

**Y=A’X**

Constructing a Sequential Circuit using D Flip-Flops:

Flip-Flops input Functions

Input

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Present  State |  |  | Next  State | Output |  |  |  |
| A | B | X | A | B | Y | DA | DB |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | X | X | X | X | X |
| 1 | 1 | 1 | X | X | X | X | X |

Table: State Table for circuit using JK Flip-flops

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 0 | 0 | 1 |
| 0 | 0 | X | x |

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 1 | 0 |
| 1 | 0 | X | x |

**DA=A’X+AX’**

**DB=A’X’**

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 1 | 0 |
| 0 | 0 | X | X |

**Y=A’X**

**Discussion:**

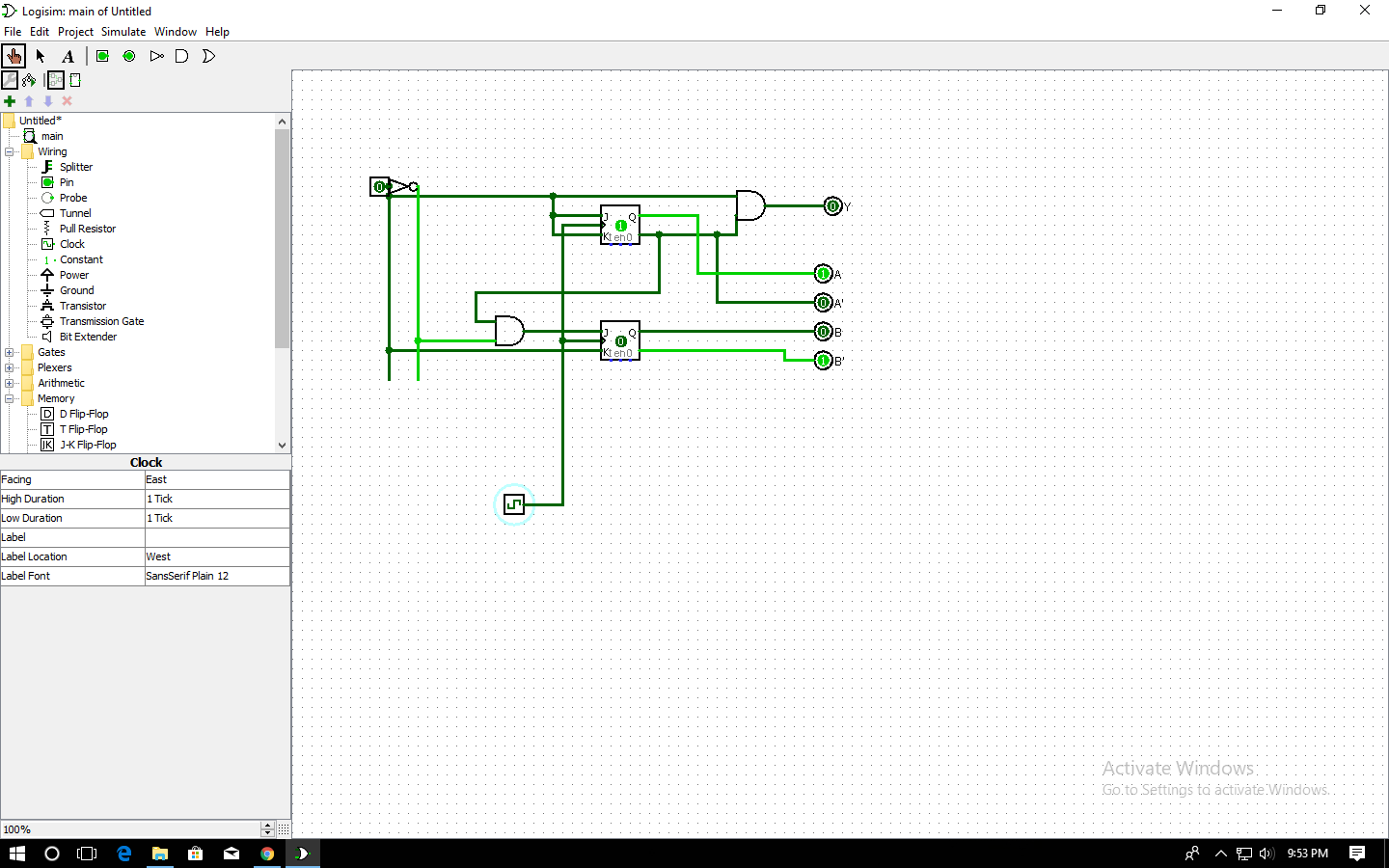
Today we learnt about the MUX and Decoders. At first we created a MUX using basic gates. While using the gates we didn’t get expected results. Then we changed our trainer board and then checked it again. But we got the same result. After that we checked every IC separately. Then we came to know that the AND gate IC had problem. Then we changed the IC. Then our result came as expected.

Next, we had to create a circuit using a MUX IC. In this case our IC was working properly. So, we did not face any problems. And this time we finished the circuit very quickly.

After that, we build a circuit using decoder. As decoders cannot be checked before implementing the circuit, we made the whole circuit to see whether it works properly or not. The IC worked properly. But couple of the output lights didn’t blow. But we managed with that. As all others bulbs worked properly and gave us the expected value we considered as the circuit is working properly.

Finally, after all our work has done cleared our table and rearranged our chairs. Then it was the end of the lab that day.

1.



2.

Yes, the output equation of Y is the same as the equation in the JK Flip-Flop. Because in JK Flip-Flop to make the next step 0 from 0 the J must be 0 and K is don’t care. If we take K as 0 it is the same equation as T. Again, to make Qnext is 1 from 0 we must take J = 1 and K will be don’t care. If we take K as 1 then it will be the same equation as T Flip-Flop. Similarly to make Qnext = 0/1 from Q = 1 we need to make K = 1/0 and J will be don’t care. If we take J the same as K then it will be the same as T Flip-Flop. So, Y will have the same equation as JK Flip-Flop in this.

3.

