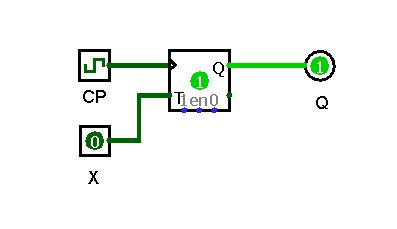
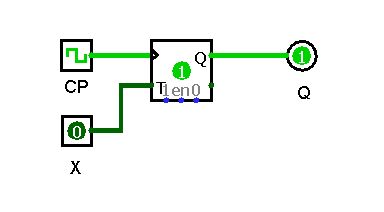
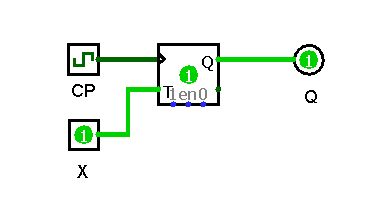
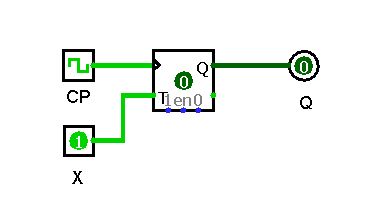
**Digital Electronics Bhargav Patel**

**Aim :** Design and Implementation of Character Display Device using counter

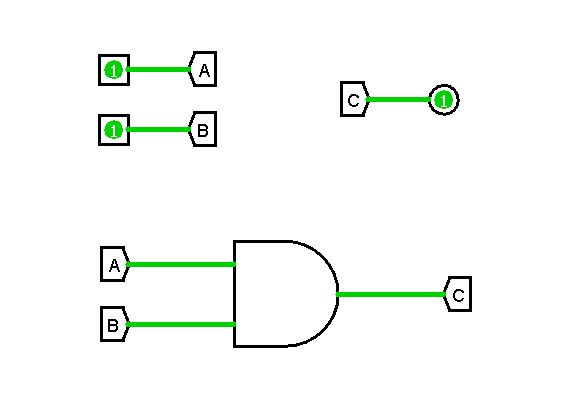
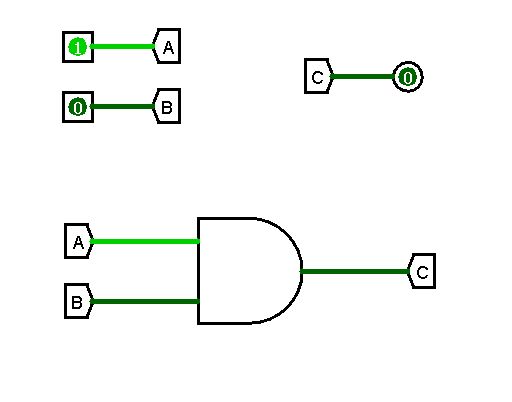
**Components Used :**

1. **T Flip Flop :**

* Excitation Table:

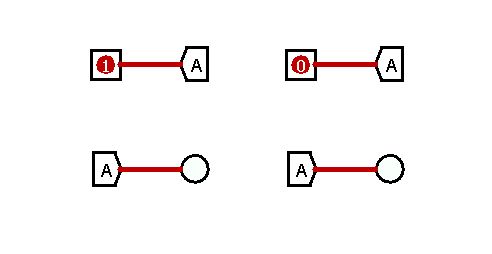
|  |  |  |
| --- | --- | --- |
| **Q(t)** | **T** | **Q(t+1)** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

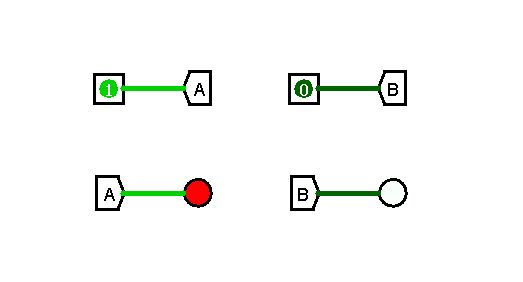
* If Input is 1 then State will get changed otherwise for 0 it Remains same
* And to get the Output of Next state We are Using Clock Pulse.

1. **Tunnel :**

* Tunnel is very useful tool in Logisim to avoid unnecessary wiring.

In Circuit each Tunnel will have its own value (ex. A,B,a,b,A1,etc.).

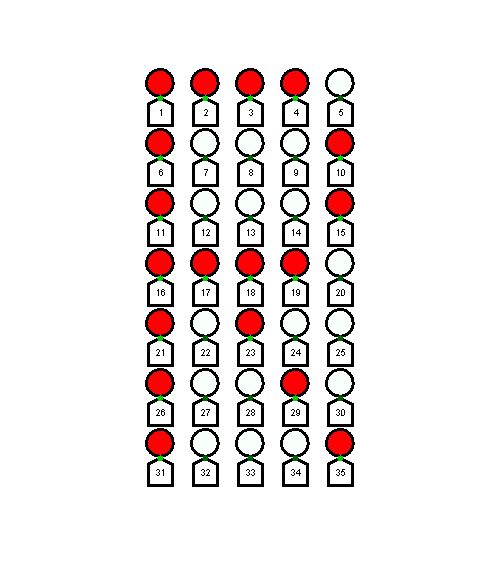
* We can use any number of tunnels in circuit but one of them should get input , so rest of the tunnels will act as an output terminal for same. And Two Tunnels having same value can’t be given Different Inputs.

1. **LED :**

* LED is sort of Output terminal in Logisim, in which it gets ON and OFF instead of showing output in terms of 0 and 1.
* If Input is 1 LED will glow with Bright Red Color and Remains in off mode for input 0 ( In Logisim We can change Color code for On and Off state of LED)

1. **LED Display :**

* In this Display Device We Have Arranged LEDs in form 7\*5 sized matrix.
* In this matrix we can Give Various Sequence of inputs to show Particular Pattern or Character . So it behaves like LED Display .
* So, we will be giving certain sequence of 35 inputs to Display All Alphabets.



* Present States :
* Present State of All T Flip Flops are Denoted by T1, T2, T3, …, T35.
* And to Determine next States for all of Them we are taking 8 Input States ( T16, T17, T18, T20, T31, T32, T33, T35).
* Because Sequence generated by present states of these 8 inputs are unique for all Alphabets. So, they can be used to determine Next State of all 35 Flip Flops
* Input States :

**a = T16 e = T31**

**b= T17 f = T32**

**c= T18 g = T33**

**d = T20 h = T35**

* State Table :

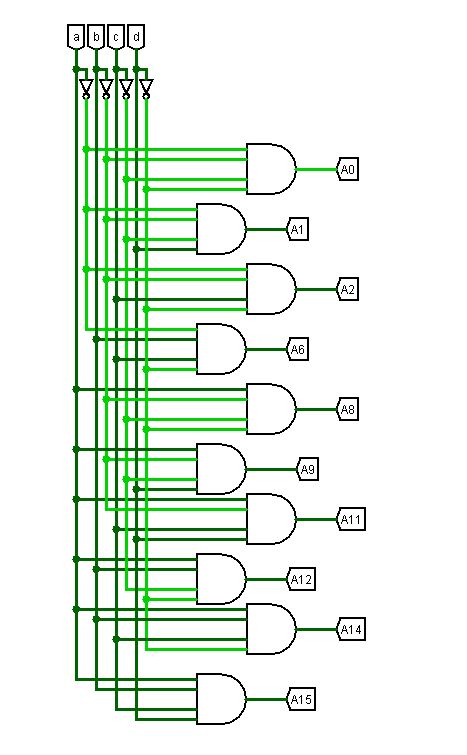
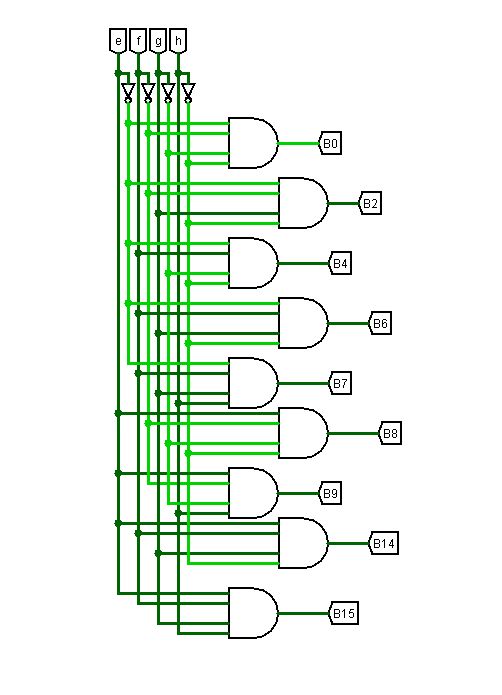
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Alphabet | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T 10 | T 11 | T 12 | T 13 | T 14 | T 15 | **T 16** | **T 17** | **T 18** | T 19 | **T 20** | T 21 | T 22 | T 23 | T 24 | T 25 | T 26 | T 27 | T 28 | T 29 | T 30 | **T 31** | **T 32** | **T 33** | T 34 | **T 35** |
| - | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **0** | **0** | **0** | 0 | **0** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **0** | **0** | **0** | 0 | **0** |
| A | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **1** | **1** | 1 | **1** | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **0** | **0** | **0** | 0 | **0** |
| B | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **1** | **1** | 1 | **0** | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | **1** | **1** | **1** | 1 | **0** |
| C | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | **1** | **0** | **0** | 0 | **0** | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | **0** | **0** | **0** | 0 | **0** |
| D | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **0** | **0** | 0 | **1** | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **1** | **1** | 1 | **0** |
| E | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | **1** | **1** | **1** | 1 | **0** | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | **1** | **1** | **1** | 1 | **1** |
| F | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | **1** | **0** | **0** | 0 | **0** | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | **1** | **0** | **0** | 0 | **0** |
| G | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | **1** | **0** | **0** | 0 | **0** | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | **0** | **1** | **1** | 1 | **0** |
| H | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **1** | **1** | 1 | **1** | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **0** | **0** | 0 | **1** |
| I | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | **0** | **0** | **1** | 0 | **0** | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | **0** | **1** | **1** | 1 | **0** |
| J | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | **0** | **0** | **0** | 0 | **1** | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **0** | **1** | **1** | 1 | **0** |
| K | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | **1** | **1** | **0** | 0 | **0** | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | **1** | **0** | **0** | 0 | **1** |
| L | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | **1** | **0** | **0** | 0 | **0** | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | **1** | **1** | **1** | 1 | **1** |
| M | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | **1** | **0** | **0** | 0 | **1** | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **0** | **0** | 0 | **1** |
| N | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | **1** | **0** | **1** | 0 | **1** | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **0** | **0** | 0 | **1** |
| O | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **0** | **0** | 0 | **1** | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | **0** | **0** | **0** | 0 | **0** |
| P | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **1** | **1** | 1 | **0** | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | **1** | **0** | **0** | 0 | **0** |
| Q | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **0** | **0** | 0 | **1** | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | **0** | **1** | **1** | 1 | **1** |
| R | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **1** | **1** | 1 | **0** | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | **1** | **0** | **0** | 0 | **1** |
| S | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | **0** | **1** | **1** | 1 | **0** | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **0** | **1** | **1** | 1 | **0** |
| T | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | **0** | **0** | **1** | 0 | **0** | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | **0** | **0** | **1** | 0 | **0** |
| U | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **0** | **0** | 0 | **1** | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **0** | **1** | **1** | 1 | **0** |
| V | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **0** | **0** | 0 | **1** | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | **0** | **0** | **1** | 0 | **0** |
| W | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **0** | **0** | 0 | **1** | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | **0** | **1** | **0** | 1 | **0** |
| X | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **0** | **1** | **1** | 1 | **0** | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **1** | **0** | **0** | 0 | **1** |
| Y | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | **0** | **1** | **1** | 1 | **0** | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | **0** | **0** | **1** | 0 | **0** |
| Z | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | **0** | **0** | **1** | 0 | **0** | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | **1** | **1** | **1** | 1 | **1** |

* Flip Flop Inputs :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Conversion | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | T9 | T 10 | T 11 | T 12 | T 13 | T 14 | T 15 | T 16 | T 17 | T 18 | T 19 | T 20 | T 21 | T 22 | T 23 | T 24 | T 25 | T 26 | T 27 | T 28 | T 29 | T 30 | T 31 | T 32 | T 33 | T 34 | T 35 |
| 0 -> A | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| A -> B | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| B -> C | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| C -> D | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| D -> E | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| E -> F | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| F -> G | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| G -> H | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| H -> I | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| I -> J | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| J -> K | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K -> L | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| L -> M | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| M -> N | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| N -> O | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| O -> P | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| P -> Q | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Q -> R | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| R -> S | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| S -> T | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| T -> U | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| U -> V | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 |
| V -> W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| W -> X | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| X -> Y | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| Y -> Z | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| Z -> 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |

* To Simplify Sequence of 8 State Inputs We are maintaining two different sequences having 4 inputs in each, So We can easily acquire any 8-state sequence by performing AND operation between sequences from each set.

|  |  |
| --- | --- |
| * First Set of Sequences :   A0 = a’b’c’d’  A1 = a’b’c’d  A2 = a’b’cd’  A6 = a’bcd’  A8 = ab’c’d’  A9 = ab’c’d  A11 = ab’cd  A12 = abc’d’  A14 = abcd’  A15 = abcd | * Second Set of Sequences :   B0 = e’f’g’h’  B2 = e’f’gh’  B4 = e’fg’h’  B6 = e’fgh’  B7 = e’fgh  B8 = ef’g’h’  B9 = ef’g’h  B14 = efgh’  B15 = efgh |

* **Sequence Generator :**
* To Generate These Sequence, We have used Two Sequence Generators.

Ex: For Alphabet D Present State for all State Inputs Would be

a = T16 = 1 e = T31 = 1

b = T17 = 0 f = T32 = 1

c = T18 = 0 g = T33 = 1

d = T20 = 1 h = T35 =0

* By Taking Sequence A9 from Set 1 and Sequence B14 from Set 2 We can Make Sequence of **A9 AND B14** Which is Unique for Alphabet D.
* We can Derive Flip Flop Inputs For Different Combinations of Sequence.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Alphabets | Combinations | | T1 | T2 | T3 |  | T34 | T35 |
| A | A15 | B0 | 1 | 0 | 0 | ….. | 1 | 1 |
| B | A14 | B14 | 1 | 0 | 0 | ….. | 0 | 0 |
| C | A8 | B0 | 1 | 0 | 0 | ….. | 0 | 0 |
| D | A9 | B14 | 0 | 0 | 0 | ….. | 0 | 1 |
| E | A14 | B15 | 0 | 0 | 0 | ….. | 1 | 1 |
| F | A8 | B8 | 1 | 0 | 0 | ….. | 1 | 0 |
| G | A8 | B6 | 1 | 1 | 1 | ….. | 1 | 1 |
| H | A15 | B9 | 1 | 1 | 1 | ….. | 1 | 1 |
| I | A2 | B6 | 0 | 0 | 0 | ….. | 0 | 0 |
| J | A1 | B6 | 1 | 1 | 1 | ….. | 1 | 1 |
| K | A12 | B9 | 0 | 0 | 0 | ….. | 1 | 0 |
| L | A8 | B15 | 0 | 0 | 0 | ….. | 1 | 0 |
| M | A9 | B9 | 0 | 0 | 0 | ….. | 0 | 0 |
| N | A11 | B9 | 1 | 1 | 1 | ….. | 1 | 1 |
| O | A9 | B0 | 1 | 0 | 0 | ….. | 1 | 0 |
| P | A14 | B8 | 1 | 0 | 0 | ….. | 1 | 1 |
| Q | A9 | B7 | 1 | 0 | 0 | ….. | 1 | 0 |
| R | A14 | B9 | 1 | 0 | 0 | ….. | 1 | 1 |
| S | A6 | B6 | 1 | 0 | 0 | ….. | 1 | 0 |
| T | A2 | B2 | 0 | 1 | 1 | ….. | 1 | 0 |
| U | A9 | B6 | 0 | 0 | 0 | ….. | 1 | 0 |
| V | A9 | B2 | 0 | 0 | 0 | ….. | 1 | 0 |
| W | A9 | B4 | 0 | 0 | 0 | ….. | 1 | 1 |
| X | A6 | B9 | 0 | 0 | 0 | ….. | 0 | 1 |
| Y | A6 | B2 | 0 | 1 | 1 | ….. | 1 | 1 |
| Z | A2 | B15 | 1 | 1 | 1 | ….. | 1 | 1 |

* From Above Given Table We Can Derive Equations for T1, T2 ,T3, ….T35 in Terms of Sum of Product (SOP) .
* **Next State Equations:**

T1 = (A15 + A8 + A9)B0 + (A1 + A8 + A6)B6 + (A14 + A11 + A115) + B8(A8 + A14) + B14A14 + B7A9 + B15A2

T2 = T3 = T4 = A0B0 + B6(A1 + A8) + A2B15 + B9(A11 + A15) + B2(A2 + A6)

T5 = B6(A2 + A8 + A6) + B9(A11 + A12 + A15) + B9A8 + B14A9 + B15(A2 + A8)

T6 = T11 = A0B0 + B6(A1 + A6) + B9A15 + B2(A2 + A6)

T7 = A6B15 + A9B9

T8 = A15B9 + B6(A2 + A6) + B2A2

T9 = A6B15 + B9(A9 + A12) + A1B6

T10 = A0B0 + B6(A1 + A2 + A6) + B14A9 + A2B2 + A8B8 + B9A15 + B15(A2 + B6)

T12 = B15A14 + B9(A9 + A11) + B8A8

T13 = B15(A8 + A14) + A8B8 + B9(A9 + A12 + A15) + A2B2 + B6(A1 + A2 + A6)

T14 = A6B2 + B15(A2 + A14) + A8B8

T15 = B14(A9 + A14) + B0A8 + B6(A1 + A2 + A8) + A8B15 + B9(A14 + A15) + B2(A2 + A6) + A0B0

T16 = A0B0 + B9(A14 + A15) + A2B2 + A1B6 + A9B4

T17 = A0B0 + A14(B8 + B14 + B15) + A8B6 + A9(B0 + B4 + B7 + B14) + A15B9 + A6(B2 + B6) + A1B6 + A12B9

T18 = A0B0 + A14(B9 + B14 + B15) + A11B9 + A9(B0 + B4 + B7 + B9 + B14) + A8B6 + A2(B2 + B6 + B15)

T19 = A0B0 + A14(B9 + B14 + B15) + A15B9 + A6(B2 + B6) + A8B6 + A9(B0 + B4 + B7 + B14)

T20 = A0B0 + A15(B0 + B9) + A9(B0 + B4 + B7 + B14) + A2(B2 + B6) + A8(B0 + B6 + B15) + A1B6 + A14B8

T21 = B9(A14 + A6 + A15) + A2B2 + A1B6 + A0B0

T22 = A6B2 + A2B15

T23 = B2(A2 + A6 + A9) + A9B4 + B6(A1 + A2 + A6) + A14B8 + B9(A6 + A12 + A14 + A16)

T24 = A8(B8 + B6) + B9(A9 + A11)

T25 = A0B0 + A14(B6 + B9) + A8(B8 + B15) + A9(B14 + B0 + B7) + A2(B6 + B2) + A6(B6 + B9) + A1B6 + A15B9

T26 = B0(A0 + A8 + A9) + B6(A2 + A6 + A9) + B2(A2 + A6 + A9) + B9(A6 + A11 + A15) + A2B15 + A14B14

T27 = A14B14 + A9(B2 + B0 + B6) + A8B0 + A11B9

T28 = A14B14 + B2(A2 + A6 + A9)B6(A2 + A6) + B9(A15 + A11 + A6) + B4A9 + B0(A8 + A9)

T29 = A14B14 + B6(A1 + A9) + B9(A12 + A11 + A14) + B8A14 + B2A9 + B0(A8 + A9)

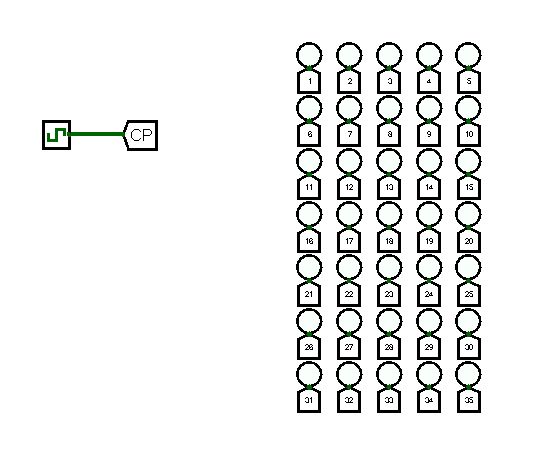
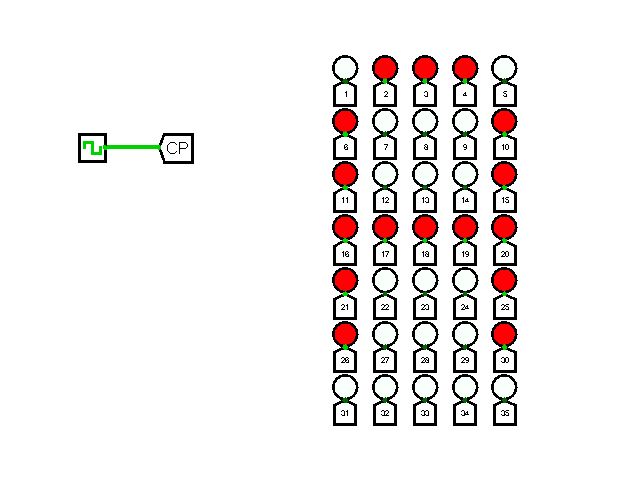
T30 = A15B0 + A14B15 + A8B6 + A12B9 + A9B9 + A9B0 + A9B4 + A6B2 + A2B15

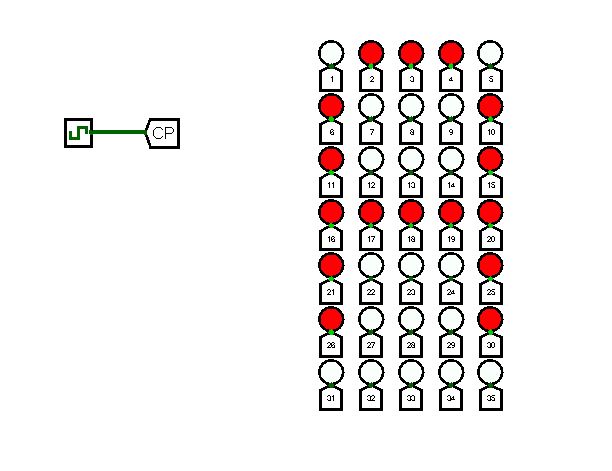
T31 = A15B0 + A14(B14 + B8 + B9) + A8(B8 + B0 + B6) + A9(B0 + B7 + B4) + A1B6 + A6(B2 + B9) + B9(A11 + A15) + A2B15

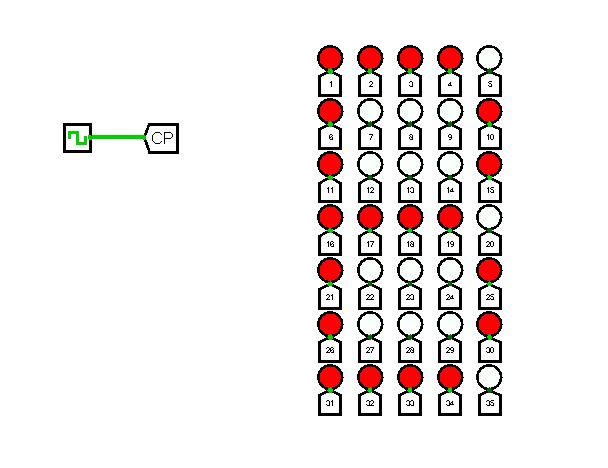
T32 = T34 = A14B14 + B6(A1 + A8 + A9 + A6) + B9(A15 + A12 + A14) + B15(A14 + A8 + A2) + B2(A2 + A9 + A6) + B0(A8 + A15) + B8(A8 + A14) + A9(B4 + B7)

T33 = A14B14 + B9(A12 + A15 + A14 + A6) + B15(A14 + A8 + A2) + B0(A15 + A8) + B8(A8 + A14) + B6(A8 + A1) + A9(B2 + B7)

T35 = A14(B8 + B15 + B9) + A15B9 + A9(B4 + B14) + B6(A1 + A8) + A6(B9 + B2) + A11B9 + A2B15

* **Circuit Implementation :**
* After Implementation of Circuit using Derived State Equations, all State would be ‘0’ Initially.
* So, Clock pulse would be OFF and LED Display Would be blank.
* After Turning on Clock Pulse for the first time all States will get Changed And Display will show Alphabet ‘A’.



* After that We will Turn off the Clock Pulse, so all state values will be stored. And while again turning on the Clock Pulse State values get changed and shows Alphabet ‘B’ at Display.
* By keep Turning the Clock pulse in ON and OFF mode we can show next Alphabets accordingly.(B,C,D,…)
* After ‘Z’ It will change state values for Alphabet ‘A’, so it moves like loop.
* **Real Life Applications :**
* This type of concept is highly used in working of Dot Matrix Display(DMD).
* Dot Matrix Display is the simple and more commonly used displays for showing advertisement information in shops, clocks, railway departure indicators, bus routes, etc where low cost displays are required with limited resolution.
* Generally, There are set of messages are supposed to be displayed, So we can’t use single character Display device. Instead of that we can use scrolling messages LED Display for Long Messages.
* DMD’s are now becoming more popular in industrial segments due to low cost and durability when compared to LCD technologies.
* Here are some advantages of the DMD displays used in industries :
* Easy update and dynamic message display to diverse audience
* Can display in any fonts/languages as generated by software
* Basic graphics and animation effects possible
* No need of powerful MCUs
* Relay messages to employees through one board or company-wide communication system
* Easy integration with the existing setup with just a tiny gateway between the system and the DMD panel
* Durable and practically no maintenance
* Low cost compared to LCD displays



* Commonly available display configurations are from 3 X 5 (15 LEDS) to 128 X 64 (8192 LEDS).