**TRAFFIC CONTROL SYSTEM**

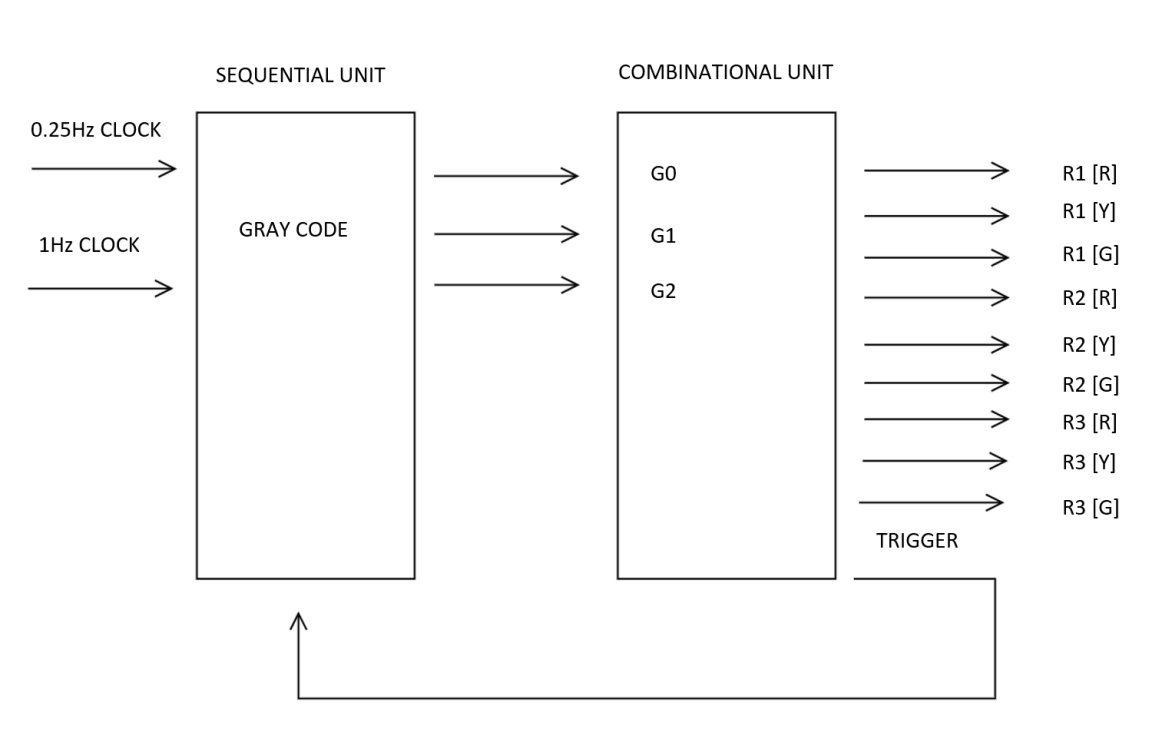
**INTRODUCTION**

The main objective of this project was to design a Traffic Control System for a T-intersection that consists of a timer at each road and operates the traffic lights in a proper order after a specific delay in time.

**COMPONENTS**

* And Gate (7408) ICs
* Not Gate (7404) ICs
* Or Gate (7432) ICs
* Xor Gate (7486) ICs
* J-K Flip-Flops (7473) ICs
* BCD 7-Segment Driver (7447) ICs
* 7-Segment Display Common Anode
* Breadboards
* Connecting Wires
* Jumper Wires
* Power Supply
* Function Generator
* LEDs (Red, Yellow and Green)

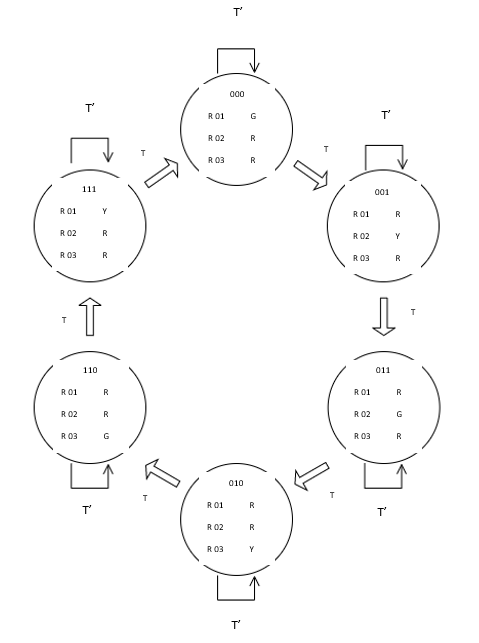
The Traffic Control System is responsible for controlling the sequence of Traffic lights (Red, Yellow, Green) of all signals and ensures that the sequences are changing after a specified amount of time, 4s in this particular case. The system is divided into 2 units (Combinational Unit and Sequential Unit). The Combinational Unit is responsible for controlling the sequence of lights whereas the transition of states depending on time is controlled by the Sequential Unit.



**FIGURE 01 BLOCK DIAGRAM OF TRAFFIC CONTROL SYSTEM**

**STATE DIAGRAM**

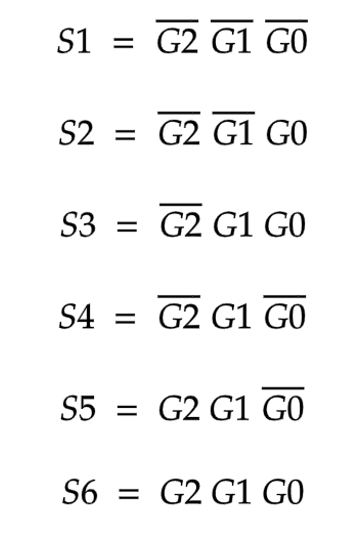
State diagram represents the sequence of states, the conditions for each state, and the requirements for transitions from one state to the next. Each state is assigned a 3-Bit Gray code as indicated. A looping arrow means that the system remains in a state, and an arrow between states means that the system transitions to the next state. T is the output of And Gate connected at the terminals of a 2-Bit Synchronous Counter of the Sequential Unit. System remains in each state until T is 0 and transitions to next state only when T is 1 or for T complement.



**FIGURE 02 STATE DIAGRAM**

**COMBINATIONAL UNIT**

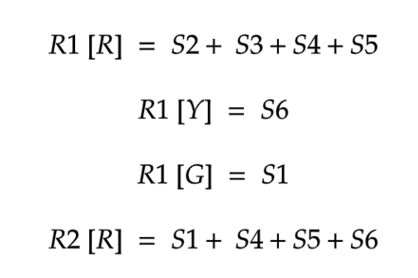
The combinational logic unit provides output to turn the traffic lights on or off. In the combinational unit all the And Gates act as state decoders. State decoder decodes the 3-bit Gray code from the sequential logic to determine which of the 6 states the system is in. The inputs to the state decoder are the 3 Gray code bits G2, G1 and G0. The 6 state outputs are S1, S2, S3, S4, S5 and S6. For each input code, one and only one output is activated. The Boolean expressions for the state outputs in terms of the inputs are as stated.

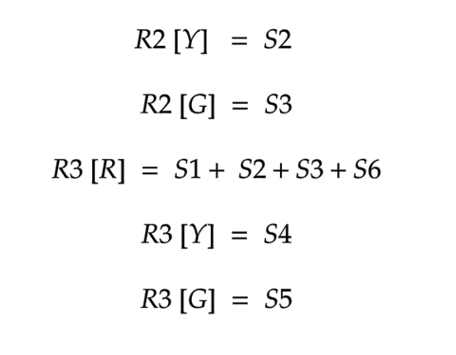


**TRUTH TABLE FOR STATE DECODER**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **STATE INPUTS [GRAY CODE]** | | | **STATE OUTPUTS** | | | | | |
| **G2** | **G1** | **G0** | **S1** | **S2** | **S3** | **S4** | **S5** | **S6** |
| **0** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** |
| **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** |
| **0** | **1** | **1** | **0** | **0** | **1** | **0** | **0** | **0** |
| **0** | **1** | **0** | **0** | **0** | **0** | **1** | **0** | **0** |
| **1** | **1** | **0** | **0** | **0** | **0** | **0** | **1** | **0** |
| **1** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **1** |

Light Output Logic has the state decoder outputs as its inputs and produces outputs to turn the traffic lights on and off for 3 specified roads. The Boolean expressions for Light Logic’s outputs in terms of the inputs are as stated.

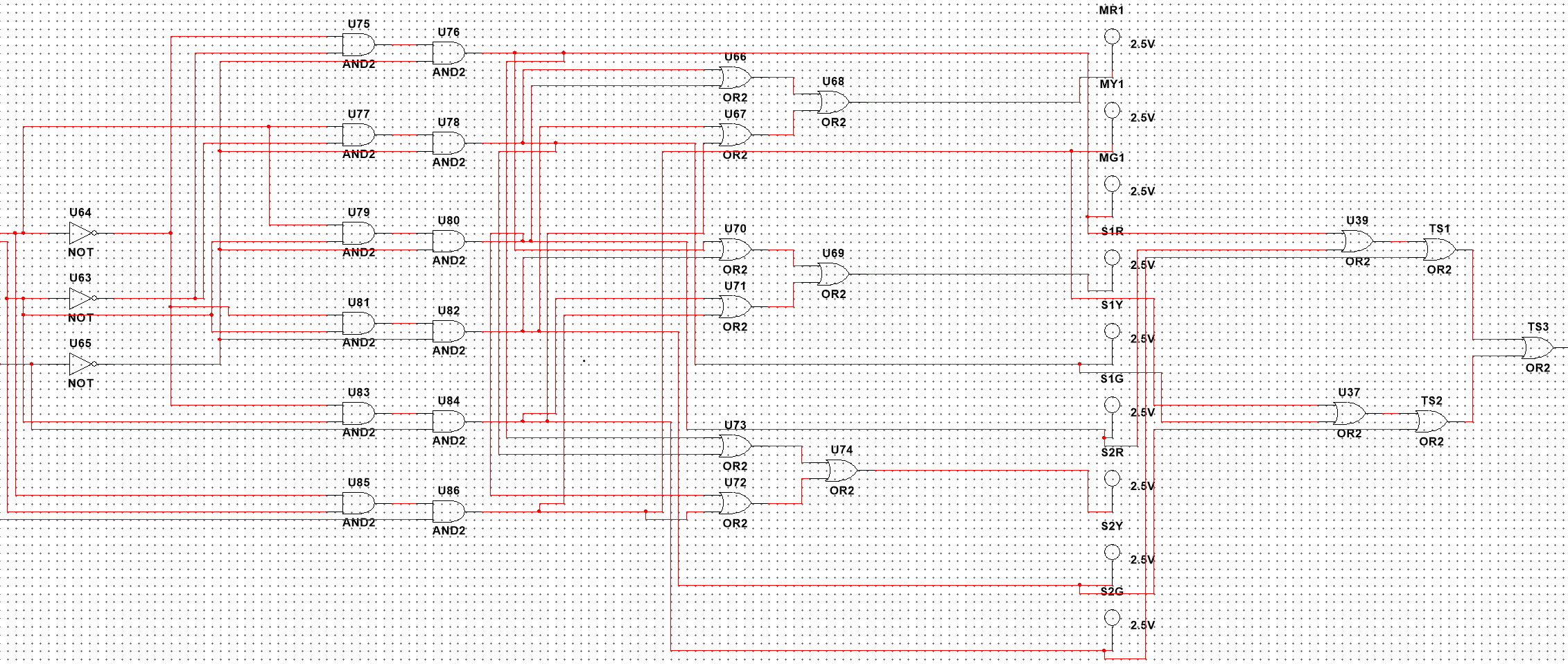




**TRUTH TABLE FOR LIGHT OUTPUT LOGIC**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **G2** | **G1** | **G0** | **R 01**  **R** | **R 01**  **Y** | **R 01**  **G** | **R 02**  **R** | **R 02**  **Y** | **R 02**  **G** | **R 03**  **R** | **R 03**  **Y** | **R 03**  **G** |
| **0** | **0** | **0** | **0** | **0** | **1** | **1** | **0** | **0** | **1** | **0** | **0** |
| **0** | **0** | **1** | **1** | **0** | **0** | **0** | **1** | **0** | **1** | **0** | **0** |
| **0** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **1** | **1** | **0** | **0** |
| **0** | **1** | **0** | **1** | **0** | **0** | **1** | **0** | **0** | **0** | **1** | **0** |
| **1** | **1** | **0** | **1** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **1** |
| **1** | **1** | **1** | **0** | **1** | **0** | **1** | **0** | **0** | **1** | **0** | **0** |

The Trigger Logic produces a single output that is 1 for every 3-Bit Gray Code combination. The Sequential Unit consists of 2 synchronous counters and the initial J-K Flip-Flop of 2-Bit Counter toggles when the output of Trigger is 1 as both J and K inputs of Flip-Flop are 1 at that time.



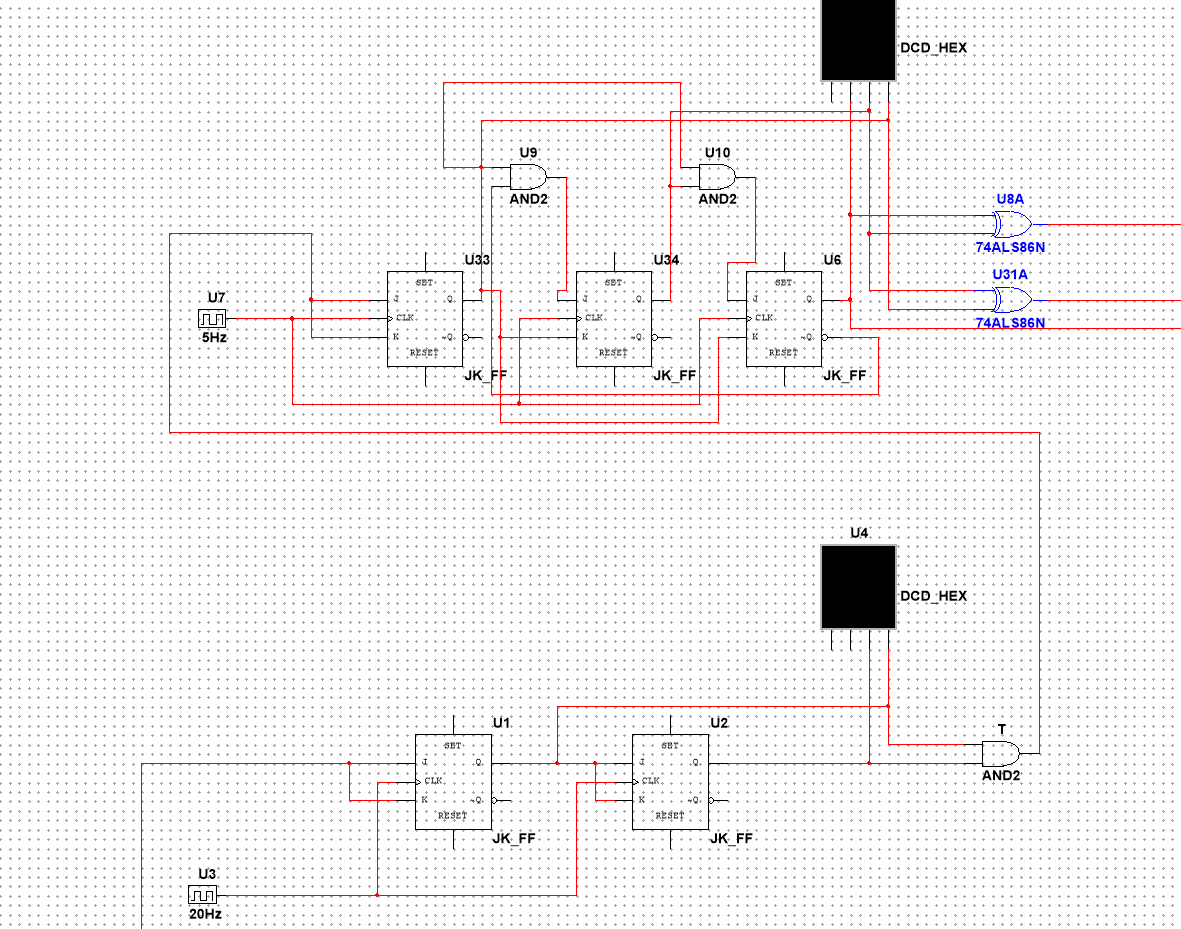
**FIGURE 03 COMBINATIONAL UNIT**

**SEQUENTIAL UNIT**

The Sequential Unit consists of Xor Gates, And Gates, 2-Bit Synchronous Counter, 3-Bit Synchronous Counter and 2 clocks. The 2-Bit Synchronous Counter counts from 0 to 3 and operates due to the Trigger of Combinational Unit. An And Gate is connected to its MSB and LSB terminals, the output of this gate is 1 only when digit 3 is displayed at the counter as both inputs of the gate are 1 at this particular time. The 3-Bit Synchronous counter operates due to output of the And Gate and counts from 0 to 7. A combination of Xor gates is connected at the 3 terminals of the counter, that take binary numbers as inputs and generates an equivalent gray code number by performing Modulo 2 Addition. Clocks are set at frequencies of 1Hz and 0.25Hz for 2 and 3 Bit Counters.

**CONVERSION TABLE**

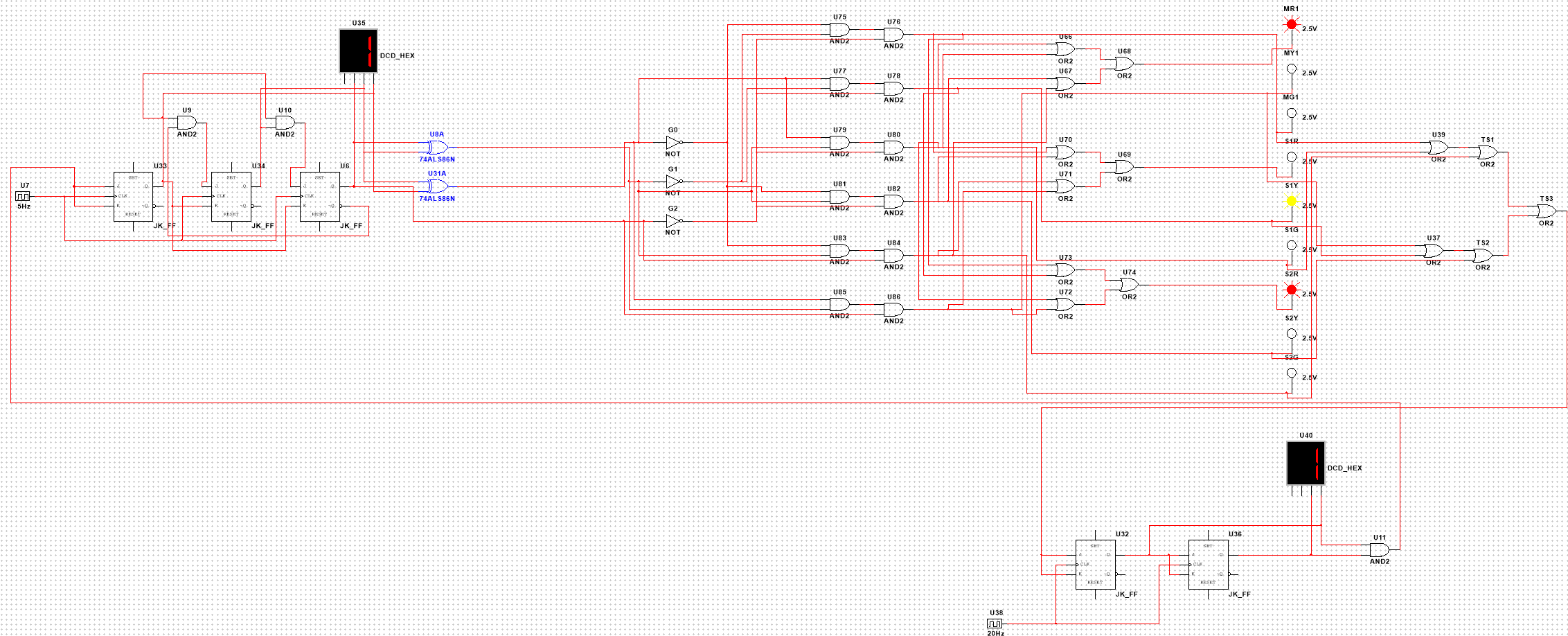
|  |  |  |
| --- | --- | --- |
| **DECIMAL EQUIVALENT** | **BINARY** | **GRAY CODE** |
| **0** | **000** | **000** |
| **1** | **001** | **001** |
| **2** | **010** | **011** |
| **3** | **011** | **010** |
| **4** | **100** | **110** |
| **5** | **101** | **111** |



**FIGURE 04 SEQUENTIAL UNIT**

**CONCLUSION**

Traffic control systems can be implemented using multiple logics depending upon the timing requirements and the type of intersection even for same timing requirements and intersection multiple logics can exist. For this particular system a simple logic that completes all the requirements of the control system is designed using basic gates.



**FIGURE 05 COMPLETE TRAFFIC CONTROL SYSTEM**