Heaven's light is our guide

Rajshahi University of Engineering and Technology

Department of Electrical and Computer Engineering

Lab Project

Design a Digital Clock using DEEDs Simulator with Verilog Code.

Course Code	ECE 2112
Course Title	Digital Techniques Sessional
Date of Experiment	02-02-2024
Date of Submission	14-05-2025

Submitted to

Md. Omaer Faruq Goni

Assistant Professor

Department of Electrical and Computer Engineering

RUET

Submitted by

Serial	Name	Roll	Task
01	Umme Sumaya Niha	2210025	BCD to 7 Segment
02	Okia Goni	2210027	MOD-10, 24 Detector, Cascade Connection,Verilog
03	Maheya Jannat Nilima	2210049	MOD-6,MOD-3, Documentation, Design

Files

Name	Link
MOD-3	MOD-3
MOD-6	MOD-6
MOD-10	MOD-10
BCD to 7 Segment Decoder	Decoder
24 Detector	24 Detector
Verilog Module	Module
Verilog Testbench	Testbench

Contents

Abstract :

This report thoroughly explores the design and implementation of a digital clock, constructed using synchronous counters and modular arithmetic circuits. The entire project was developed in strict accordance with the instructions and guidelines provided by our respected course instructor, ensuring both academic alignment and practical accuracy. The clock system is structured using a series of counters MOD-3, MOD-6, and MOD-10 to accurately represent and manage the progression of seconds and minutes. For hour tracking, a dedicated 24-hour detector circuit is integrated to maintain the correct 24-hour time format. Each counter stage is carefully linked through cascade connections, which allow for a seamless and synchronized flow of timing signals from one stage to the next, ensuring consistent and reliable timekeeping. To display the time output in a clear and user-friendly format, a BCD (Binary-Coded Decimal) to 7-segment display decoder is used. This decoder translates binary values into corresponding decimal digits, which are then presented on the 7-segment displays. The report encompasses all aspects of the clock's construction, including detailed block diagrams, comprehensive circuit schematics, and the logical equations that govern each component's operation. All files, diagrams, and supporting documentation are neatly organized and referenced in the index section for ease of navigation.

This report aims to provide an in-depth, structured explanation of the digital clock's architecture, component interconnections, and overall functionality, reflecting the theoretical and practical knowledge acquired during the course.

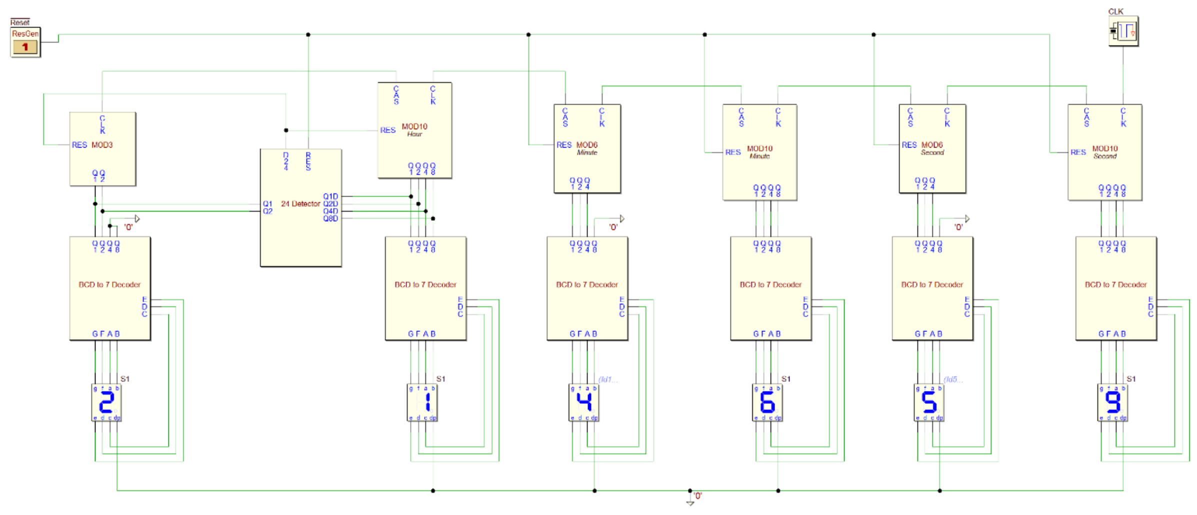


Figure 1 : Simulated Block Diagram of Digital Clock

Theory :

The digital clock is designed using synchronous counters, modular arithmetic circuits, and display decoding logic to accurately track and display time. Synchronous counters, triggered by a common clock signal, ensure precise timing without propagation delays. MOD-10, MOD-6, and MOD-3 counters are used to represent seconds, minutes, and hours, respectively, with a 24-hour detector circuit resetting the hour count after 23. The counters are connected in cascade, enabling smooth transitions between time units. Binary outputs from the counters are converted into readable digits using a BCD to 7-segment display decoder, which drives the visual output. This modular and synchronized design ensures reliable and clear timekeeping functionality.

Required Components :

- MOD-3 Counter
- MOD-3 Counter
- MOD-3 Counter
- BCD to 7 Segment Decoder
- 24 Detector
- Reset Button
- Clock Sources(1 hz)
- Logic Gates
- T Flip-Flop Pet

MOD-3

A MOD-3 counter is a sequential circuit that counts from 0 to 2 and then resets to 0, completing one cycle in three clock pulses. Using T (Toggle) flip-flops, the counter changes state on each clock edge based on the toggle input. For a MOD-3 counter, two T flip-flops are sufficient, as $2^2 = 4 > 3$. However, since only three states (00, 01, 10) are valid, the unused state (11) must be handled—typically by applying a reset condition to bring the counter back to 00. The toggle inputs and logic are designed so that the counter follows the sequence 00 → 01 → 10 → 00, and continues cyclically.

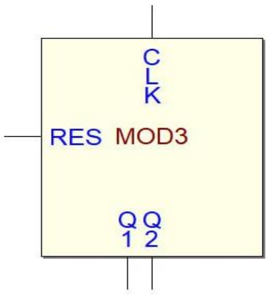


Figure 2 : Block Diagram of MOD-3

In this configuration, the clock (CLK) pin is connected to the carry-out (CAS) pin of the MOD-10 hour counter. The reset (RES) line is connected to the D24 output of the 24-hour detector circuit. Additionally, the Q1 and Q2 outputs from the counter are used as inputs to the BCD to 7-segment decoder for display purposes.

Truth Table:

Present State	Next State	T2	T1
00	01	0	1
01	10	1	1
10	00	1	0

Logic Equations:

T1 = Q2'
T2 = Q1+Q2

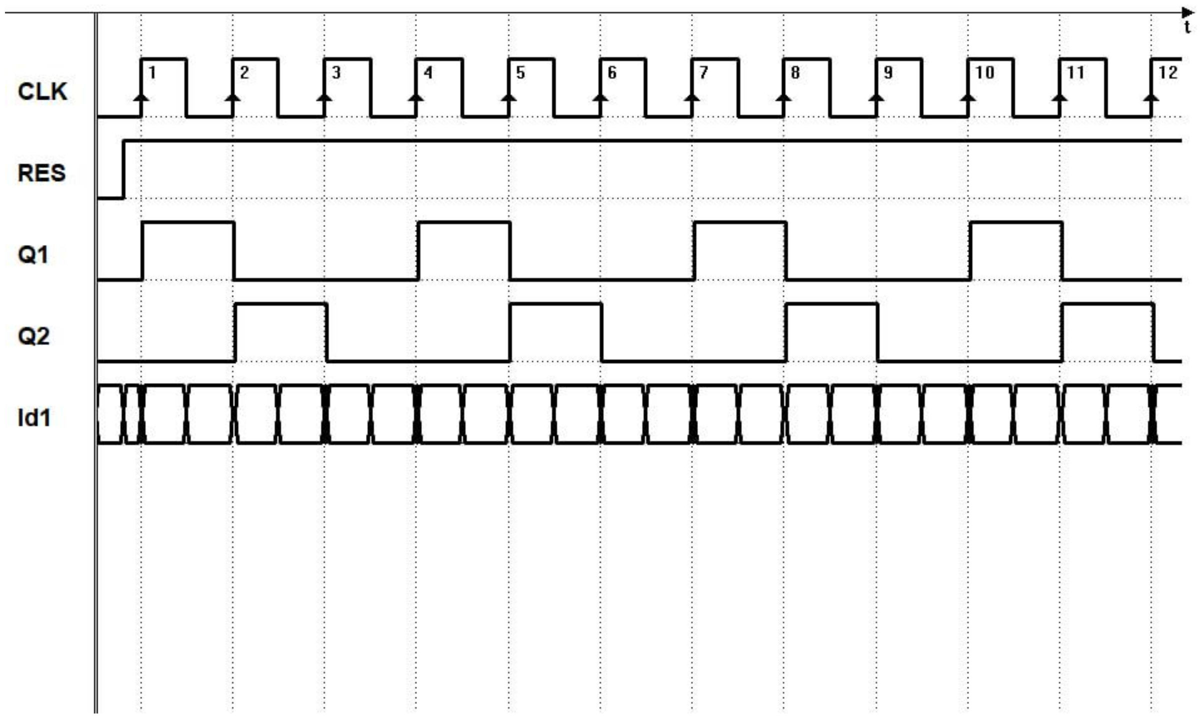


Figure 3 : Timing Diagram of MOD-3

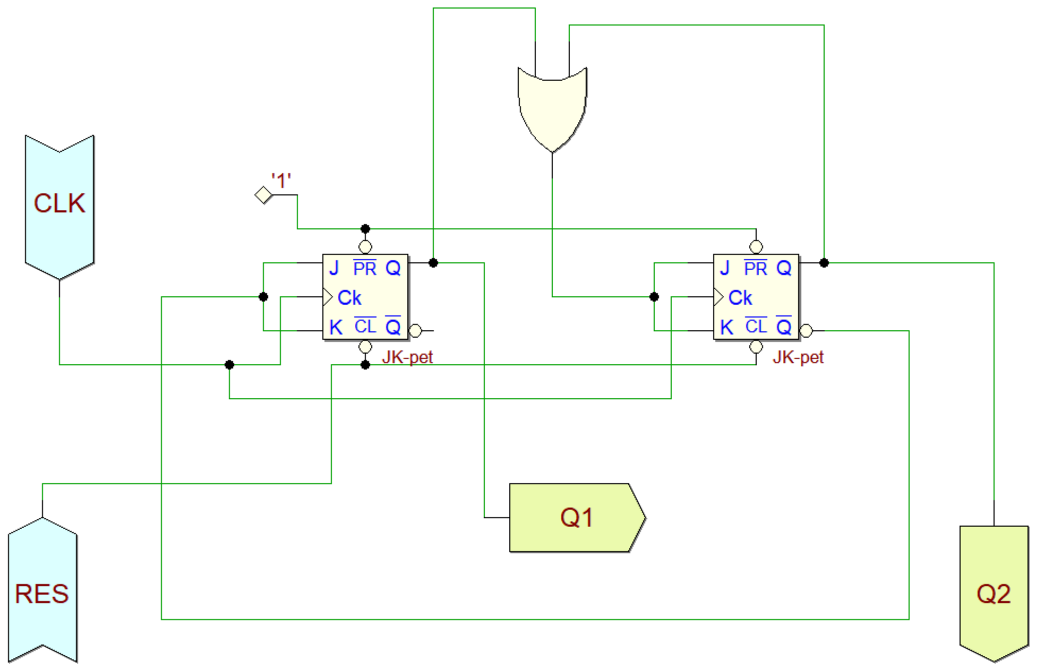


Figure 4 : Circuit Diagram of MOD-3

MOD-6

A MOD-6 counter is a sequential circuit that counts from 0 to 5 and then resets to 0, completing one full cycle in six clock pulses. To design a MOD-6 counter using T (Toggle) flip-flops, we need three flip-flops, since $2^3=8 > 6$, which provides enough states to cover the required six (000 to 101). However, two states (110 and 111) are unused and must be handled appropriately. This is typically done by introducing a logic condition that detects invalid states and resets the counter to 000 whenever an unused state appears.

The flip-flops toggle on the clock edge depending on the logic designed for the T inputs. The desired count sequence is:
000 → 001 → 010 → 011 → 100 → 101 → 000
and this repeats cyclically. The toggle logic ensures the counter follows this specific 6-state pattern and maintains stability by preventing it from entering undefined states.

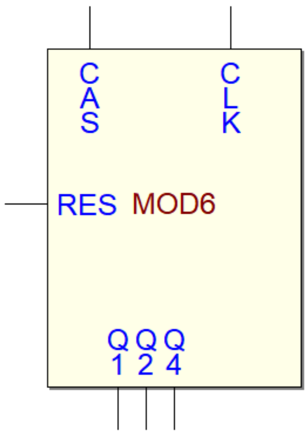


Figure 5 : Block Diagram of MOD-6

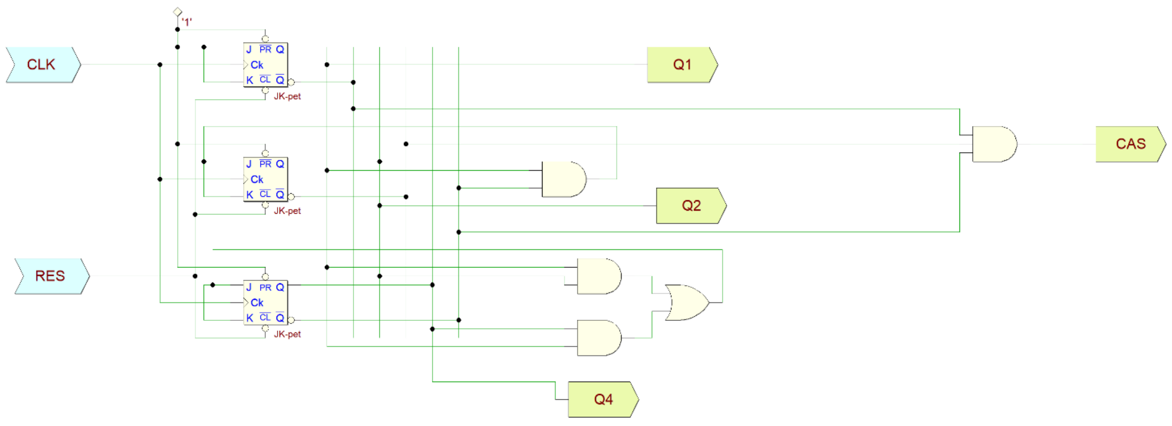


Figure 6 : MOD-6 Circuit

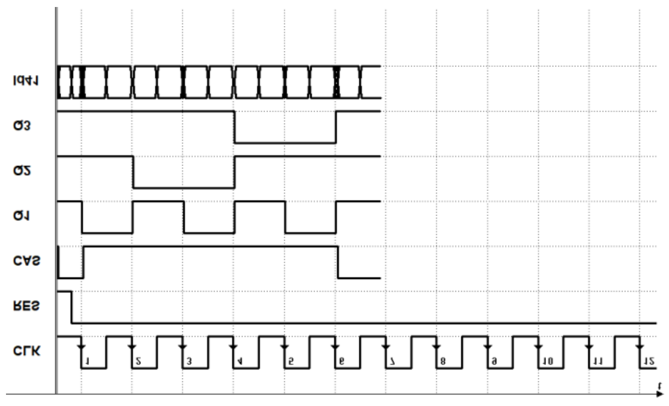


Figure 7 : Timing Diagram of MOD-6

Truth Table :

Present State	Next State	T4	T2	T1
000	001	0	0	1
001	010	0	1	1
010	011	0	0	1
011	100	1	1	1
100	101	0	0	1
101	000	1	0	1

Logic Equations:

T1=1
T2=Q1Q4'
T4=Q1Q2+Q1Q4

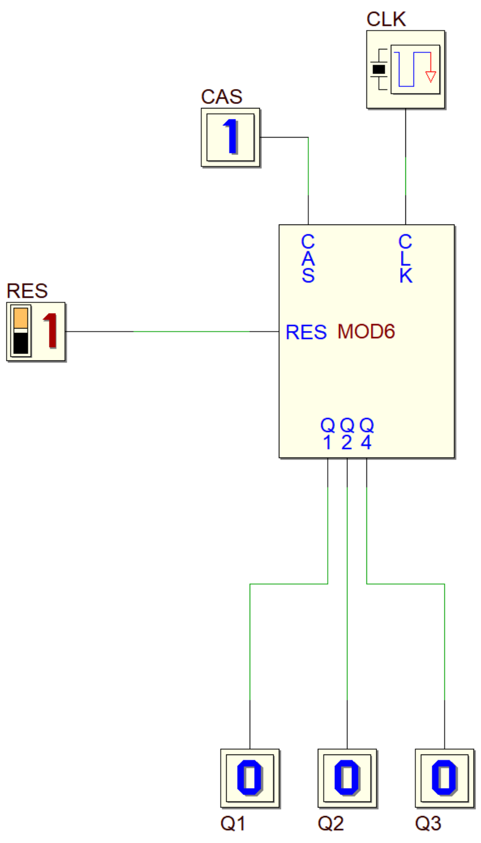


Figure 8 : Detection of Logic 1 in CAS pin

All the output pins (Q1, Q2, Q3, and Q4) are passed through NOT gates, and their outputs are combined using an AND gate to detect when all are logic LOW (0). This condition represents the 0000 state, and the resulting signal is used to generate the clock input for the next modulus counter stage.

MOD-10

A MOD-10 synchronous counter using T flip-flops is a digital circuit designed to count from 0 to 9 in binary with all flip-flops triggered simultaneously by a common clock signal. It consists of four T flip-flops connected in a synchronous configuration, where the toggling of each flip-flop depends on the outputs of the previous flip-flops and specific combinational logic. The T inputs are controlled using logic gates to ensure the correct toggling sequence, and an additional reset logic is implemented to detect the count of 1010 (decimal 10) and asynchronously reset the counter to 0000. This type of counter provides faster and more reliable operation compared to asynchronous designs.

In the overall design, three MOD-10 counters were utilized to track time for seconds, for minutes, for hours. Each counter is responsible for counting up to its respective limit before resetting and triggering the next stage, effectively forming the basis of a functional digital clock system.

Truth Table:

State	T8	T4	T2	T1
0000	0	0	0	1
0001	0	0	1	1
0010	0	0	0	1
0011	0	1	1	1
0100	0	0	0	1
0101	0	0	1	1
0110	0	0	0	1
0111	1	1	1	1
1000	0	0	0	1
1001	1	0	0	1

Logic Equations :

T1=1
T2=Q1Q8'
T4=Q1Q2
T8=Q1Q8 + Q1Q2Q4

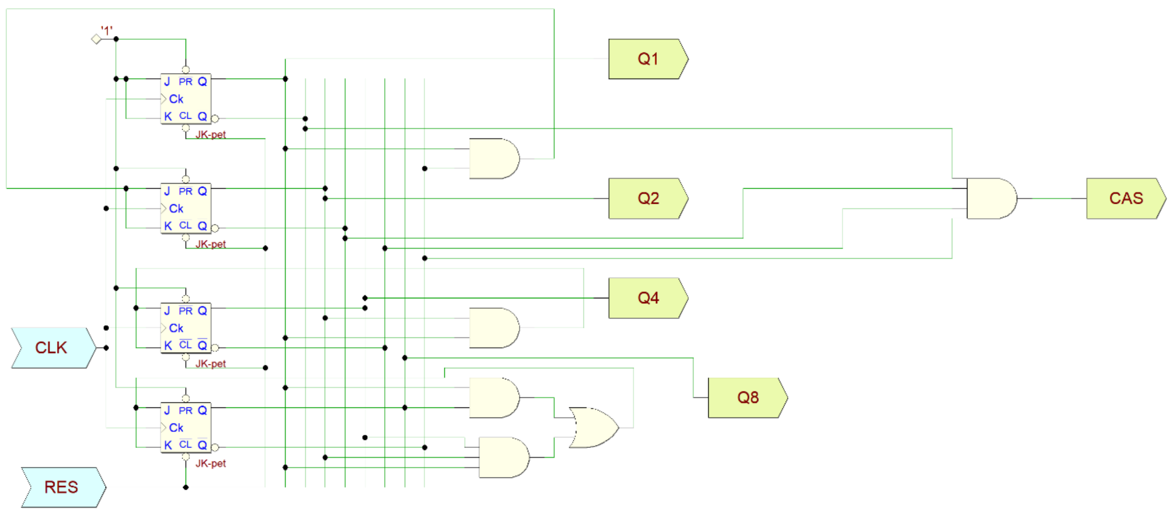


Figure 9 : Circuit Diagram of MOD-10

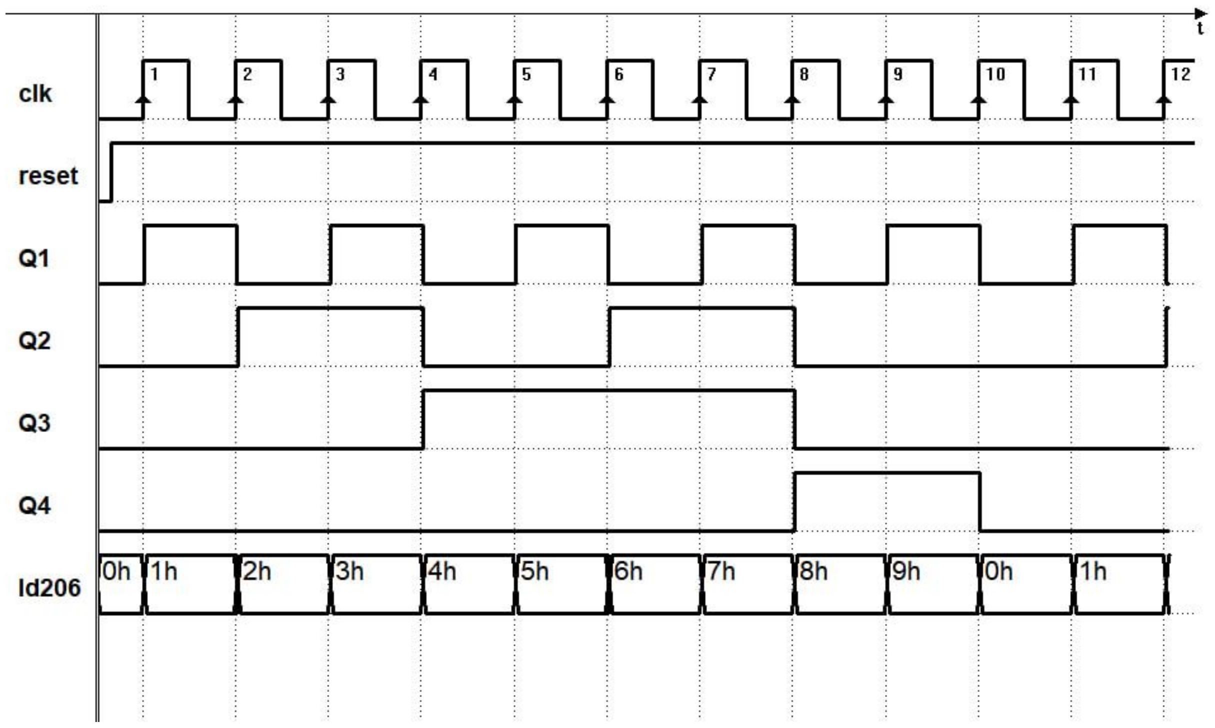


Figure 10 : Timing Diagram of MOD-10

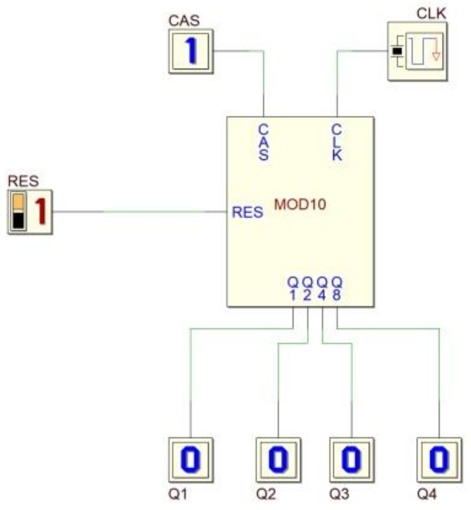


Figure 11 : Simulation of MOD-10

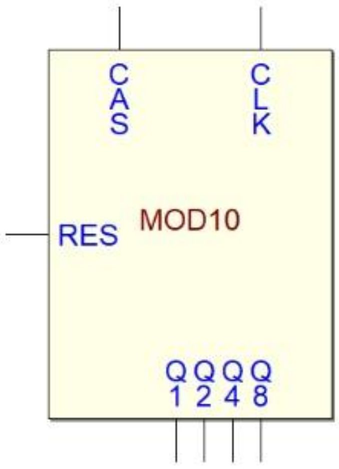


Figure 12 : Block Diagram of MOD-10

All the output pins (Q1, Q2, Q3, and Q4) are passed through NOT gates, and their outputs are combined using an AND gate to detect when all are logic LOW (0). This condition represents the 0000 state, and the resulting signal is used to generate the clock input for the next modulus counter stage.

24 Detector

In this digital clock circuit, accurate timekeeping in the 24-hour format is ensured using a special logic block that detects when the clock reaches 24:00:00 and immediately resets it to 00:00:00. This mechanism is extremely important because the hour section of the circuit is built using mod-3 and mod-6 counters, which by default allow the clock to count up to 29:59:59. If this logic is not implemented, the time will continue beyond 24 hours, which is logically incorrect for a 24-hour digital clock.

Detect Logic : To detect 24 (24:00:00), we use the BCD (Binary Coded Decimal) representation of the number:

- 2 = 0010 (from the tens place of the hour, generated by the Mod-3 counter)
- 4 = 0100 (from the units place of the hour, generated by the Mod-10 counter)

So, to identify when the hour counter reaches 24, we use a combination of logic gates (AND, OR, and NOT) to match this exact pattern. The gates are connected to the outputs of the BCD counters. The logic circuit becomes active only when:

- The tens digit of hour = 2 (from the Mod-3 counter), which occurs when Q1 = 0 and Q2 = 1. This condition is achieved by passing Q1 through a NOT gate and then feeding both Q2 and NOT(Q1) into an AND gate. This ensures that only the binary pattern 10 (i.e., 2) activates the logic.

- The units digit of hour = 4 (from the Mod-10 counter), which happens when Q4 = 1 and Q1, Q2, and Q3 = 0. Here, Q1, Q2, and Q3 are first passed through NOT gates, and then all four conditions (NOT Q1, NOT Q2, NOT Q3, and Q4) are fed into a single AND gate. This uniquely identifies the binary pattern 0100 (i.e., 4).

The output of this logic block acts as a detection pulse.

Reset Logic : The outputs from both identification blocks (for 2 from the Mod-3 counter and 4 from the Mod-10 counter) are first fed into a NAND gate. This ensures that the detection pulse will be active low (the output becomes 0 only when both inputs are high, exactly at 24).Now, to finalize the reset logic, the output of this NAND gate is then combined with the Reset control signal (manually triggered or coming from another part of the circuit) using an AND gate. This AND gate ensures that the counters will reset only when the detection condition is true and the reset is enabled. The output of this AND gate is then connected to the reset (RES) inputs of both the Mod-3 and Mod-10 counters (which represent the tens and units place of the hour respectively). As a result, when the clock reaches exactly 24:00:00, the detection circuit triggers the AND gate, and the counters are immediately reset to 00:00:00.This logic flow guarantees that the digital clock never exceeds 24 hours, ensuring accurate and cyclic 24-hour timekeeping

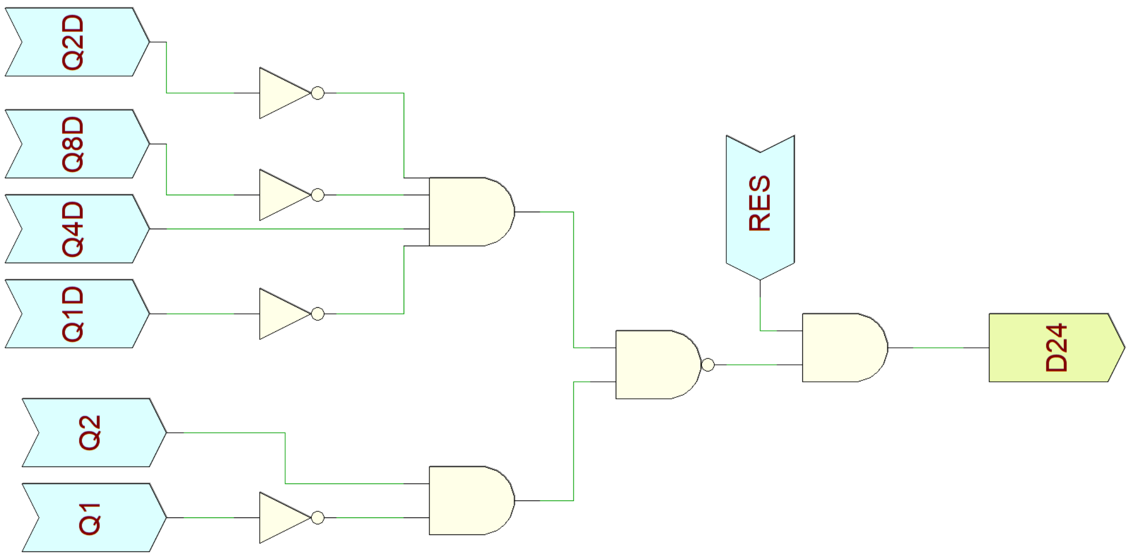


Figure 13 : 24 Hour Detector and Reset Circuit

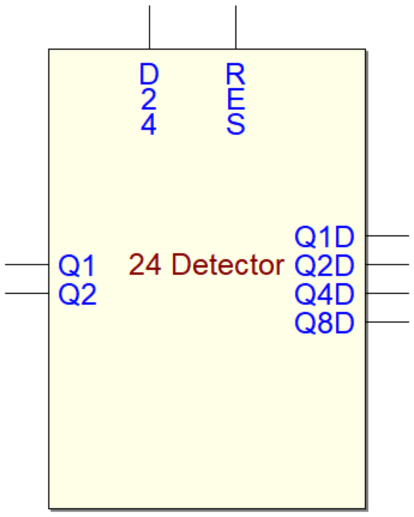


Figure 14 : Block diagram of detector of 24 hour and reset

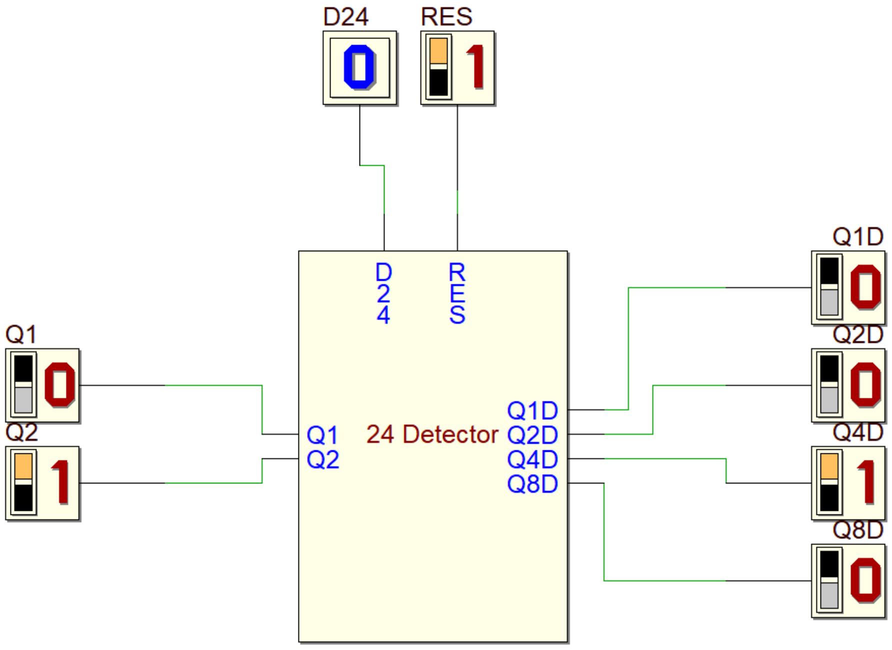


Figure 15 : D24 Simulation

The D24 button output goes LOW (0) precisely when the Mod-3 counter outputs a 2 and the Mod-10 counter outputs a 4, effectively indicating the time has reached 24:00:00.

BCD to 7 Segment Decoder

A BCD to 7-segment display circuit is used to visually represent numerical data in digital systems. BCD (Binary-Coded Decimal) is a 4-bit representation of decimal digits (0–9). Each digit is encoded in a 4-bit binary number, with values from `0000` to `1001`.A 7-segment display is a type of LED display that uses seven individual segments to represent digits. These segments are labeled as a to g, and by illuminating specific segments, any digit from 0 to 9 can be displayed. To drive the display automatically, a BCD to 7-segment decoder IC (like 7447 or 4511) is used. This IC receives a 4-bit BCD input and sends output signals to the appropriate segments, allowing the correct digit to appear on the display.

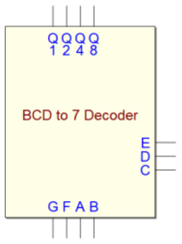


Figure 16 : Block Diagram of BCD to 7 Segment Decoder

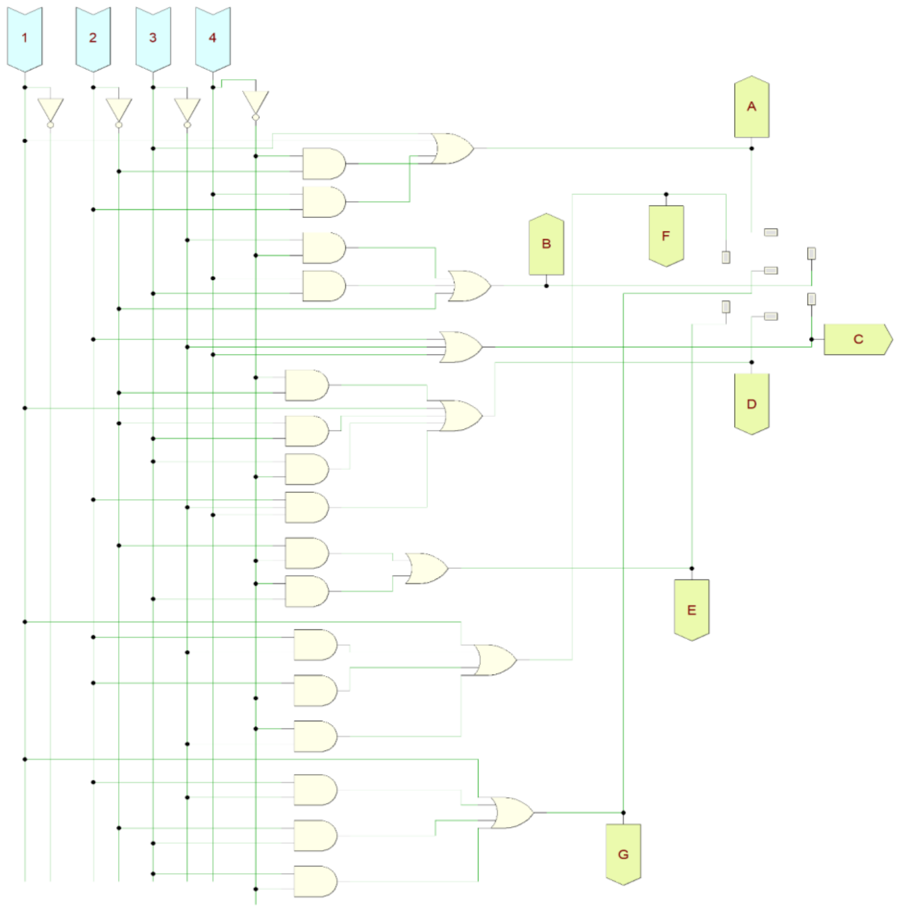


Figure 17 : Circuit Diagram of BCD to 7 Segment Decoder

Seconds and Minutes Counting : The mod-10 counter outputs a standard 4-bit BCD (Binary-Coded Decimal) value. This output is directly connected to the BCD-to-7-segment decoder which then drives the corresponding 7-segment display. The mod-6 counter only outputs 3 bits (as it counts up to 5), but the decoder requires a 4-bit BCD input. Therefore:

- The 3 output bits from the mod-6 counter are connected to the lower 3 inputs of the decoder (D0–D2).
- The unused 4th bit (D3) is tied to logic 0 to maintain a valid 4-bit input.

This ensures that all decoder inputs are defined and that the 7-segment display correctly shows digits 0 to 5 for the higher digit of seconds or minutes.

Hour Counting : The mod-10 counter behaves the same as in the seconds and minutes section. Its 4-bit BCD output goes directly to a 7-segment decoder and display. The mod-3 counter outputs only 2 bits (enough to count 0 to 2). To form a valid 4-bit input for the decoder

- The 2 output bits are connected to the lower bits of the decoder.
- The remaining 2 input bits are tied to logic 0.

This approach ensures a consistent 4-bit BCD input to the decoder, allowing accurate representation of the tens digit of the hour (0, 1, or 2) on the 7-segment display.

Common Logic and Display configuaration : All 7-segment displays are assumed to be of common anode type, as logic 0 is used as the common ground across all display units. The outputs of the BCD counters go through BCD-to-7-segment decoder ICs (for common anode displays), which convert the binary value into signals that control the segments (a–g) of the display.

Circuit Explanation :

Input: The entire digital clock circuit is driven by a 1 Hz clock signal, which means it ticks once per second—perfect for simulating real-time second counting. This clock pulse acts as the base input to the first counter in the seconds section.

Seconds Counter: A MOD-10 counter is used for the units place of the seconds, and a MOD-6 counter is used for the tens place, allowing the seconds to count from 00 to 59. The CAS (carry out) pin of the MOD-10 counter detects when the count rolls over to 0 (i.e., after reaching 9 and resetting), and this output is connected to the clock input of the MOD-6 counter. As a result, each time the units counter resets from 9 to 0, the tens counter receives a clock pulse and increments by 1. This cascading setup ensures accurate counting from 00 to 59 seconds before rolling over and incrementing the minutes counter.

Minutes Counter: The MOD-6 counter used for the tens place of seconds is connected to the MOD-10 counter representing the units place of minutes. Specifically, the CAS (carry out) pin of the seconds' tens counter is connected to the clock input of the minute's units counter. This means that when the seconds counter completes a full cycle from 00 to 59 and resets back to 00, a clock pulse is sent to the minute's units counter, causing it to increment by 1.This cascading arrangement ensures that the minutes counter advances by one after every full 60-second cycle. By using a MOD-10 counter for the units place and a MOD-6 counter for the tens place of minutes, the minutes are able to count accurately from 00 to 59 before triggering the hour counter.

Hour Counter : The MOD-6 counter used for the tens place of the hour is connected to the MOD-10 counter representing the units place of the hour. This cascading setup allows the hour counter to increment correctly, and without any additional logic, it would naturally count from 00 up to 29 (since 6×10=606 \times 10 = 606×10=60). However, in a 24-hour clock format, the time must stop at 23:59:59 and reset to 00:00:00.

Reset: To solve this, a special "Detect 24 and Reset" block is introduced. This block monitors the outputs of the hour counters and identifies when the time reaches 24:00:00 using logic gates (NOT and AND gates). Specifically, it detects when the tens digit of the hour is 2 (output 10 from the MOD-6) and the units digit is 4 (output 0100 from the MOD-10). When this condition is met, the circuit triggers a reset signal that clears all counters—seconds, minutes, and hours—bringing the time back to 00:00:00. This mechanism ensures that the clock functions within the proper 24-hour format, preventing it from rolling over to invalid times like 29:00:00, and making the design more robust and realistic.

Display: Each of the modulus counters (MOD-10, MOD-6, and MOD-3) provides binary outputs that are used as inputs to BCD to 7-segment decoders, which in turn control the digital display. The MOD-10 counter generates 4 binary outputs (Q1 to Q4) corresponding to the numbers 0 to 9. These outputs are connected to the 4-bit input of a BCD to 7-segment decoder, which converts the binary value into 9 segment control signals (a, b, c, d, e, f, g, and DP along with an optional enable). These signals are then used to drive a 7-segment display configured in common cathode mode (0), meaning all the cathodes of the segments are tied together to ground, and segments light up when a HIGH signal is applied. For the MOD-6 counter, which counts from 0 to 5, only 3 output lines are needed. To form a 4-bit input for the BCD decoder, the most significant bit (MSB) is set to 0, while the remaining three outputs from the counter are connected to the corresponding lower bits. The decoder then generates the segment outputs accordingly. Similarly, the MOD-3 counter, with 2 output lines, uses two leading zeros for the MSBs and connects the remaining two outputs to the LSBs of the decoder input. This ensures compatibility with the 4-bit BCD input format. In each case, the DP (decimal point) pin is manually set to 0 (OFF) since it is not used in the current design. The common cathode configuration ensures correct segment illumination when the decoder outputs are HIGH.

Simulations :

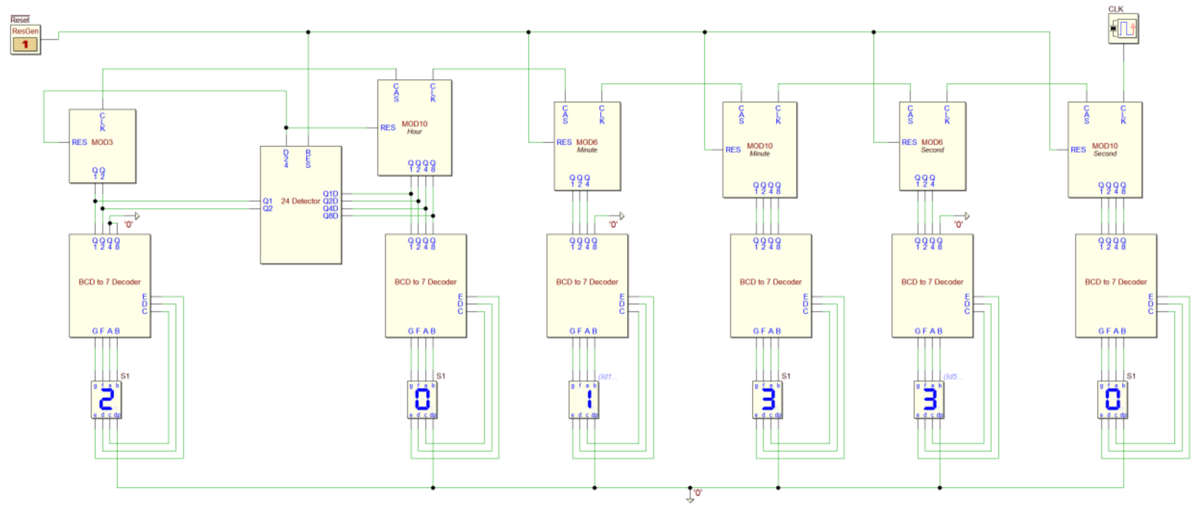


Figure 18 : Simulation for checking the Hour Clock

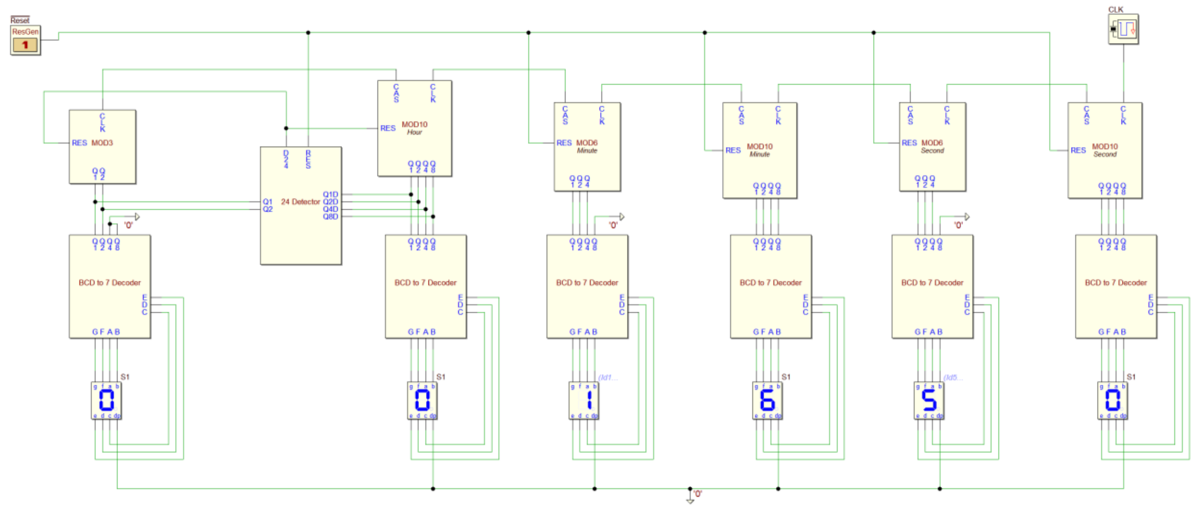


Figure 19 : Simulation to check the Minute Clock

Verilog Code :

```verilog
// digital_clock.v
module digital_clock(
    input clk_1hz,
    input reset,
    output reg [6:0] sec_units_display,
    output reg [6:0] sec_tens_display,
    output reg [6:0] min_units_display,
    output reg [6:0] min_tens_display,
    output reg [6:0] hour_units_display,
    output reg [6:0] hour_tens_display
);

    reg [3:0] sec_units = 0;
    reg [2:0] sec_tens = 0;
    reg [3:0] min_units = 0;
    reg [2:0] min_tens = 0;
    reg [3:0] hour_units = 0;
    reg [1:0] hour_tens = 0;

    wire sec_units_carry, sec_tens_carry, min_units_carry,
min_tens_carry;
    wire hours_reset;

    assign sec_units_carry = (sec_units == 4'd9);
    assign sec_tens_carry = (sec_tens == 3'd5) && sec_units_carry;
    assign min_units_carry = (min_units == 4'd9) && sec_tens_carry;
    assign min_tens_carry = (min_tens == 3'd5) && min_units_carry;
    assign hours_reset = (hour_tens == 2'd2) && (hour_units == 4'd3)
&& min_tens_carry;

    always @(posedge clk_1hz or posedge reset) begin
        if (reset) begin
            sec_units <= 0;
            sec_tens <= 0;
            min_units <= 0;
            min_tens <= 0;
            hour_units <= 0;
            hour_tens <= 0;
        end else begin
            if (sec_units == 9)
                sec_units <= 0;
            else
                sec_units <= sec_units + 1;

            if (sec_units_carry) begin
                if (sec_tens == 5)
                    sec_tens <= 0;
                else
                    sec_tens <= sec_tens + 1;
            end

            if (sec_tens_carry) begin
                if (min_units == 9)
                    min_units <= 0;
                else
                    min_units <= min_units + 1;
            end

            if (min_units_carry) begin
                if (min_tens == 5)
                    min_tens <= 0;
                else
                    min_tens <= min_tens + 1;
            end

            if (min_tens_carry) begin
                if (hours_reset) begin
                    hour_units <= 0;
                    hour_tens <= 0;
                end else if ((hour_tens == 2 && hour_units == 3))
```

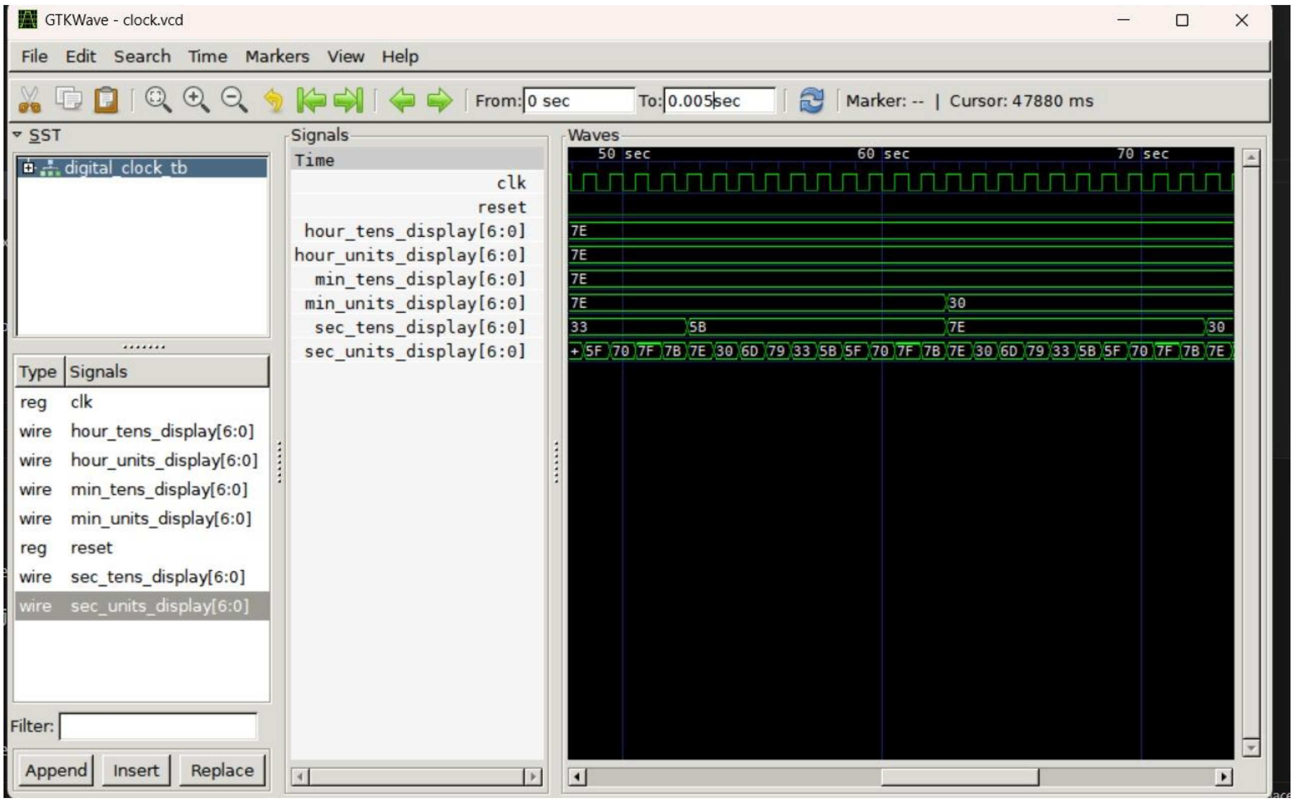
```
begin
                    hour_units <= 0;
                    hour_tens <= 0;
                end else if (hour_units == 9 || (hour_tens == 1 &&
hour_units == 9)) begin
                    hour_units <= 0;
                    hour_tens <= hour_tens + 1;
                end else begin
                    hour_units <= hour_units + 1;
                end
            end
        end
    end

    function [6:0] bcd_to_7seg(input [3:0] bcd);
        case (bcd)
            4'd0: bcd_to_7seg = 7'b1111110;
            4'd1: bcd_to_7seg = 7'b0110000;
            4'd2: bcd_to_7seg = 7'b1101101;
            4'd3: bcd_to_7seg = 7'b1111001;
            4'd4: bcd_to_7seg = 7'b0110011;
            4'd5: bcd_to_7seg = 7'b1011011;
            4'd6: bcd_to_7seg = 7'b1011111;
            4'd7: bcd_to_7seg = 7'b1110000;
            4'd8: bcd_to_7seg = 7'b1111111;
            4'd9: bcd_to_7seg = 7'b1111011;
            default: bcd_to_7seg = 7'b0000000;
        endcase
    endfunction

    always @(posedge clk_1hz or posedge reset) begin
        sec_units_display  = bcd_to_7seg(sec_units);
        sec_tens_display   = bcd_to_7seg({1'b0, sec_tens});
        min_units_display  = bcd_to_7seg(min_units);
        min_tens_display   = bcd_to_7seg({1'b0, min_tens});
        hour_units_display = bcd_to_7seg(hour_units);
        hour_tens_display  = bcd_to_7seg({2'b00, hour_tens});
    end
endmodule
```

Test-bench :

```
`timescale 1s/1ms

module digital_clock_tb;

    reg clk = 0;
    reg reset = 0;

    wire [6:0] sec_units_display;
    wire [6:0] sec_tens_display;
    wire [6:0] min_units_display;
    wire [6:0] min_tens_display;
    wire [6:0] hour_units_display;
    wire [6:0] hour_tens_display;

```

```verilog
    digital_clock uut (
        .clk_1hz(clk),
        .reset(reset),
        .sec_units_display(sec_units_display),
        .sec_tens_display(sec_tens_display),
        .min_units_display(min_units_display),
        .min_tens_display(min_tens_display),
        .hour_units_display(hour_units_display),
        .hour_tens_display(hour_tens_display)
    );

    always #0.5 clk = ~clk; // 1Hz clock

    initial begin
        $dumpfile("clock.vcd");
        $dumpvars(0, digital_clock_tb);
        reset = 1;
        #2 reset = 0;

        #100;
        $finish;
    end

endmodule
```

Output :



From: 0 sec To: 0.005sec Marker: -- | Cursor: 47880 ms

Type	Signals
reg	clk
wire	hour_tens_display[6:0]
wire	hour_units_display[6:0]
wire	min_tens_display[6:0]
wire	min_units_display[6:0]
reg	reset
wire	sec_tens_display[6:0]
wire	sec_units_display[6:0]

```
Signals                   Waves
Time                      50 sec                         60 sec                         70 sec
clk
reset
hour_tens_display[6:0]    7E
hour_units_display[6:0]   7E
min_tens_display[6:0]     7E
min_units_display[6:0]    7E                                    30
sec_tens_display[6:0]     33          5B                        7E                           30
sec_units_display[6:0]    + 5F 70 7F 7B 7E 30 6D 79 33 5B 5F 70 7F 7B 7E 30 6D 79 33 5B 5F 70 7F 7B 7E
```

Potential Improvements :

The current digital clock circuit successfully demonstrates timekeeping using modular synchronous counters and BCD to 7-segment display decoders; however, several improvements can enhance its functionality, accuracy, and user interaction. Integrating a real-time clock (RTC) module or crystal oscillator can provide more precise and stable timekeeping compared to a software-generated clock signal. Adding manual control features such as time-setting buttons, reset, and pause functionality would make the system more user-friendly. The circuit can also benefit from debouncing logic for physical switches and better synchronization during reset conditions to prevent timing errors. Upgrading the display to an LCD or graphical module could allow additional features such as AM/PM indicators, date, or alarm settings. Furthermore, implementing the design using programmable logic devices or simulating it on an FPGA platform would reduce hardware complexity, improve modularity, and offer flexibility for future upgrades. These enhancements would significantly improve the reliability, usability, and scalability of the digital clock system.

Conclusion :

In conclusion, the implementation of the digital clock circuit using MOD-10, MOD-60, and MOD-24 counters successfully demonstrated the use of sequential logic to represent real-time counting for seconds, minutes, and hours. By integrating T flip-flops, BCD to 7-segment decoders, and reset logic, the design achieved accurate time tracking and display functionality. This experiment reinforced the practical understanding of counter circuits, synchronous design, and digital display interfacing, forming a strong foundation for more advanced digital system applications.