BCD to Seven-Segment Display Clock using Verilog

Supratim Saha
School of Computer Science Engineering and Technology
Bennett Univeristy
Greater Noida, India
E23CSEU0525

***Abstract*—The evolution of digital timekeeping devices encompasses a rich interplay between hardware and software technologies. This project elucidates a novel approach in digital clock design, embodying the synthesis of Verilog-based hardware description for Binary-Coded Decimal (BCD) to seven-segment display logic, with a Python-driven graphical user interface. The Verilog module, intricately fashioned, interprets BCD inputs and governs the illumination of corresponding segments on a simulated display, effectively representing numerical time data. Complementing this, a Python script harnesses the Tkinter library to vividly render the clock interface, facilitating real-time display updates reflective of the current time. This integrated system not only serves as a practical digital clock but also as an educational archetype in digital system design, bridging theoretical constructs with tangible applications. The report, prepared by Supratim Saha from Bennett University, meticulously delineates the design rationale, descriptive code insights, and multifaceted applications, encapsulating the project's essence in the domain of embedded systems and digital electronics education.**

***Index Terms*—Digital Clock, Verilog, BCD to Seven-Segment Decoder, Python, Tkinter GUI, Hardware Description Language, Real-time Simulation, Digital Electronics Education, Embedded System Design, Graphical User Interface.**

I. Introduction

TIMEKEEPING devices are ubiquitous and vital in modern society, functioning as fundamental instruments for organization and synchronization. The portrayal of time has transitioned from analog representations to digital manifestations, reflecting advancements in electronic design and user interface engineering. This project report discusses the design and implementation of a digital clock through the Binary-Coded Decimal (BCD) to seven-segment display logic using Verilog, paired with a Python-based graphical user interface (GUI) for real-time visualization.

The adoption of BCD in digital systems is a direct consequence of the need for a reliable and straightforward method to represent decimal numbers in binary form, especially in systems that interact with human operators. BCD's congruence with human-readable decimal systems makes it particularly advantageous for displaying numerical information, such as time. Seven-segment displays are prevalent in digital clocks, owing to their simplicity and ease of interpretation. The design challenge lies in effectively converting BCD inputs into appropriate signals that activate specific segments to render the desired numerals.

Verilog, a Hardware Description Language (HDL), enables the precise modeling of digital circuits. It stands as an integral tool for simulating the BCD to seven-segment decoding logic that underpins the operation of digital time displays. The inherent nature of Verilog to model concurrent processes provides a simulation environment that closely mirrors the actual hardware behavior, offering invaluable insights during the design phase.

Complementing the Verilog simulation, the Python programming language, known for its versatility and readability, is employed to develop a GUI. Utilizing the Tkinter library, Python serves as a bridge between the hardware logic and a user-friendly interface. This synergy facilitates the observation of the Verilog simulation's outputs in a more interactive and engaging manner, rendering the clock's dynamics visible in real time.

Furthermore, the project encapsulates an educational dimension, providing a practical experience in digital electronics and embedded systems design. It serves as a didactic tool that demonstrates the integration of HDL simulation with software visualization, thereby nurturing the skills necessary for the development of embedded systems and digital interfaces.

In summary, the digital clock project leverages the strengths of Verilog for hardware logic simulation and Python for GUI representation to deliver a comprehensive timekeeping device. This paper unfolds the methodology behind the BCD to seven-segment display conversion, elucidates the Python-based visualization strategy, and explores the broader implications and applications of the project.

II. Description

A. Binary-Coded Decimal

Binary-Coded Decimal (BCD) is a class of binary encodings where each digit of a decimal number is represented by its own binary sequence. In standard BCD, which is also known as 8421 BCD, each decimal digit from 0 to 9 is represented by a four-bit binary code. For instance, the decimal number 5 is represented as 0101 in BCD. Unlike pure binary representation, where decimal numbers are converted into a continuous binary sequence, BCD ensures that each decimal digit is distinctly represented, facilitating conversions and calculations that require decimal accuracy, such as in digital time displays.

Decimal	Binay (BCD) 8 4 2 1
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1

Fig. 1. Decimal to BCD. [1]

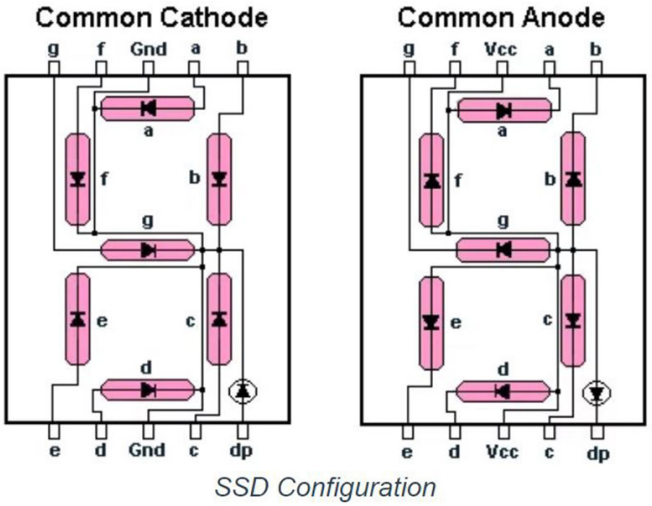

Fig. 2. Seven Segment disipay circuit. [2]

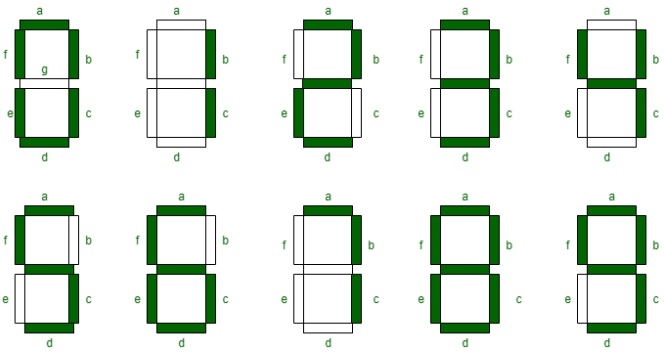

Fig. 3. Seven Segment disipays. [3]

B. Seven-Segment Display

A seven-segment display is an electronic display device that consists of seven individual segments arranged in a rectangle, with an additional segment for the decimal point in certain configurations. Each of the seven segments can be independently illuminated, typically using LEDs or liquid crystals, to represent the Arabic numerals from 0 to 9. These segments are labeled from 'A' to 'G', with each segment corresponding to a specific part of the numerals. The ability to control each segment allows for a simple and cost-effective method of displaying numerical information in many electronic devices, including clocks, watches, and calculators.

C. Digital Clock using BCD to Seven-Segment Display Logic

A digital clock utilizing BCD to seven-segment display logic employs the principles of digital electronics to represent time in a human-readable form. The core of this system is a decoder circuit that translates BCD-encoded time data into signals that drive the seven-segment displays. In this implementation, each part of the time - hours, minutes, and seconds - is converted into BCD form and subsequently fed into a dedicated BCD to seven-segment decoder. The decoder activates the corresponding segments to render the current time on the display.

In the context of the project, a Verilog module simulates this decoder logic, handling the conversion from BCD inputs to the seven-segment output signals. The simulation reflects how an actual hardware decoder would perform in real-world conditions, making it a valuable asset for the verification and testing of the design before any physical implementation.

Moreover, to visualize the output of the Verilog simulation in real-time, a Python script with a Tkinter-based GUI is deployed. The GUI displays a series of seven-segment digits that represent hours, minutes, and seconds, which update as time progresses. The interactive interface bridges the gap between the abstract simulation data and a tangible user experience, illustrating the practical utility of integrating software with hardware design.

The amalgamation of BCD encoding, seven-segment displays, and digital logic culminates in a digital clock that not only demonstrates the interaction between hardware and software but also serves as a functional timekeeping device. Its design is a testament to the efficiency and effectiveness of using BCD for numerical displays in electronic devices.

III. Code Details

This section provides an in-depth exploration of the code implemented in the BCD to Seven-Segment Display Clock project, including both the Verilog hardware description for

the logic circuit and the Python script for the graphical user interface.

A. Verilog Implementation

The Verilog module, `bcd_to_7seg`, is designed to convert a 4-bit BCD input into a 7-bit output that drives a seven-segment display. Each bit of the output corresponds to one segment of the display, where a '1' indicates that the segment is lit, and a '0' indicates it is off.

1) BCD to Seven-Segment Decoder: The core functionality of this module is encapsulated within a combinational logic block that uses a case statement to map each possible BCD input (0 to 9) to its corresponding seven-segment display pattern.

```
module bcd_to_7seg(input [3:0] bcd,
output reg [6:0] seg);
    always @(*) begin
        case(bcd)
           4'd0: seg = 7'b1111110;
            4'd1: seg = 7'b0110000;
            4'd2: seg = 7'b1101101;
            4'd3: seg = 7'b1111001;
            4'd4: seg = 7'b0110011;
            4'd5: seg = 7'b1011011;
            4'd6: seg = 7'b1011111;
            4'd7: seg = 7'b1110000;
            4'd8: seg = 7'b1111111;
            4'd9: seg = 7'b1111011;
            default: seg = 7'b0000000;
        endcase
    end
endmodule
```

2) Testbench for Verification: The testbench `tb_segment7` is used to simulate the decoder's functionality. It reads BCD values from a file, applies them to the decoder, and writes the resulting segment patterns to another file. The simulation includes timing to mimic real-world operation delays.

```
module tb_segment7;
    reg [3:0] bcd;
    wire [6:0] seg;
    integer file_bcd, file_seg, scan_file;

    bcd_to_7seg uut (
        .bcd(bcd),
        .seg(seg)
    );

    initial begin
        file_bcd = $fopen("bcd_value.dat", "r");
        file_seg = $fopen("seg_output.dat", "w");
        while (!$feof(file_bcd)) {
            scan_file = $fscanf(file_bcd, "%b\n", bcd);
            #10;
            $fwrite(file_seg, "%b\n", seg);
        }
        $fclose(file_bcd);
        $fclose(file_seg);
        $finish;
    end
endmodule
```

B. Python GUI Implementation

The graphical interface is created using Python's Tkinter library, which allows for the straightforward creation of graphical elements such as windows and canvas areas where graphical representations of the seven-segment display are drawn.

```
def draw_segment(canvas, coords, fill='grey'):
    return canvas.create_polygon(coords, fill=fill, outline='black')
```

1) Real-time Simulation Display: The GUI updates in real time by reading the segment data from the output file of the Verilog simulation. It adjusts the display colors to reflect the current time, dynamically updating every second.

```
def update_display():
    binary_values = run_verilog_simulation(get_time_bcd())
    for part, seg_string in zip(segment_ids.keys(), binary_values):
        for segment_id, bit in zip(segment_ids[part], seg_string):
            color = 'green' if bit == '1' else 'grey'
            canvas.itemconfig(segment_id, fill=color)
    root.after(1000, update_display)
```

This comprehensive code description offers a detailed insight into both the hardware and software components of the project, emphasizing their interconnectivity and functionality in creating a digital clock.

IV. Applications

The BCD to Seven-Segment Display Clock, realized through the synergy of Verilog hardware description and Python-based graphical representation, extends its utility beyond a mere timekeeping apparatus. This section delineates various applications of the project, underscoring its potential influence in educational, commercial, and research domains.

A. Educational Tool

In educational settings, this project serves as a practical example to elucidate core principles of digital logic and embedded system design. By dissecting the Verilog code, students can gain insights into combinational logic, binary representations, and the operational underpinnings of electronic displays. Similarly, the Python interface provides an exemplary case study in software integration with hardware, offering students hands-on experience in GUI development.

B. Prototyping and Embedded Systems

As a prototype, the project's architecture offers a foundational model for embedded systems that require numerical displays, such as in home appliances, industrial machines, and public information boards. The principles demonstrated in the clock can be adapted and scaled to develop more complex systems, including those with additional functionalities such as alarms, timers, and user interactions.

C. Timekeeping Devices

Commercially, the project has implications for the design and development of digital clocks, stopwatches, and countdown timers. Its minimalistic yet robust framework allows for customization and branding, making it suitable for integration into consumer electronics where time display is essential.

D. Research and Development

In research settings, the project provides a baseline for experiments in optimizing digital logic circuits and exploring alternative time representation formats. Moreover, it can be used as a starting point for studies in power efficiency, display technologies, and human-computer interaction.

In conclusion, the BCD to Seven-Segment Display Clock is not just an academic exercise but a versatile project with wide-ranging applications. Its adaptability and the principles it embodies make it a valuable asset across various fields and industries.

V. Conclusion

This project presented a coherent integration of hardware simulation using Verilog with a software interface developed in Python, culminating in the successful creation of a BCD to Seven-Segment Display Clock. The implementation demonstrated not only the feasibility of such an integrated approach but also its potential for educational purposes, prototyping, and real-world applications. The simplicity of the design, paired with the robustness of the technologies employed, provides a platform that is both instructional and practical. As digital systems continue to evolve, the methodologies showcased in this project can be applied to a broader spectrum of applications, affirming the project's value as a tool for learning, innovation, and advancement in digital system design. The experience and knowledge gained through this project lay a foundation for further exploration into the vast realm of embedded systems and digital electronics.

References

[1] Electronic Clinic. Codes in digital electronics, bcd, excess-3, error detection, ascii, gray code. Accessed: 2024-04-24. [Online]. Available: https://www.electroniclinic.com/codes-in-digital-electronics-bcd-excess-3-error-detection-ascii-gray-code/

[2] Digikey. How to interface a seven-segment display with an arduino. Accessed: 2024-04-24. [Online]. Available: https://www.digikey.com/en/maker/projects/how-to-interface-a-seven-segment-display-with-an-arduino/9c05f147618c4fe3b8bb79acce5c60e3

[3] Geek for geeks. (2023) Seven segment displays. Accessed: 2024-04-24. [Online]. Available: https://www.geeksforgeeks.org/seven-segment-displays/

[1] [2] [3]