ACTIVITY 7: Basketball Scoring System

Jason Harvey Lorenzo, April 22, 2024
ENGG 125.03, Department of Electronics, Computer, and Communications Engineering
School of Science and Engineering, Ateneo de Manila University
Quezon City, 1108, Philippines
jason.lorenzo@obf.ateneo.edu

***Abstract*—This paper presents the design and implementation of a basketball scoreboard system using Verilog HDL. The system features displays for the time remaining in the period, scores for two teams, and a shot clock. The functionalities and input controls of the system are detailed, facilitating precise control over game timing and scoring. The Verilog HDL code for the system is provided, showcasing the modular design approach with separate modules for the shot clock, timer, and scoring components. The operation of each module is explained, highlighting their roles and interactions within the overall system. Simulation results demonstrate the system's normal operation and response to various input controls, confirming its accurate and reliable performance according to the defined functionalities.**

***Index Terms*—Verilog HDL, basketball scoreboard**

I. Introduction

This project involves the construction of a basketball scoreboard system in Verilog HDL. This scoreboard will display the time left in the period and both the teams' scores, as well as the shot clock. Table I shows the outputs of the system while II the input controls for this system.

TABLE I
Scoreboard Outputs

Name	Output
Score1	First Team's Score (255 max, initially 0)
Score2	Second Team's Score (255 max, initially 0)
timer_M	Minutes left in timer (initially 12)
timer_S	Seconds left in timer (initially 0)
shotClock_dS	Deciseconds left in shot clock (initially 0)
shotClock_S	Seconds left in shot clock (initially 24)

TABLE II
Scoreboard Input Controls

Input	Function
PB0	Resets all outputs to initial state
PB1	Increments Score1 by 1
PB2	Decrements Score1 by 1
PB3	Increments Score2 by 1
PB4	Decrements Score2 by 1
SW1	If on, resets timer and shot clock to initial state
SW2	If on, counts down timer; If off, pauses timer
SW3	Resets shot clock to initial state
SW4	If on, counts down shot clock; If off, pauses shot clock

II. HDL Code

Listing 1 shows an HDL code implementation of the basketball scoreboard system discussed.

Listing 1. Ping Pong Game FSM HDL Code

```
module bball_scoring(
  input CLK, SW3, SW1, SW4, SW2,
        PB0, PB1, PB2, PB3, PB4,
  output wire [4:0] shotClock_S,
  output wire [3:0] shotClock_dS, timer_M,
  output wire [5:0] timer_S,
  output wire [7:0] Score1, Score2
);

wire RST;

shotClock shotClk(CLK, SW3, SW1, SW4, RST,
                  shotClock_S, shotClock_dS);
timer gameTime(CLK, SW1, SW2, RST,
                  timer_S, timer_M);
scoreTally scoring(CLK, PB0, PB1, PB2,
        PB3, PB4, Score1, Score2, RST);

endmodule


module shotClock(
  input CLK, SW3, SW1, SW4, RST,
  output reg [4:0] seconds,
  output reg [3:0] deciseconds
);

wire pe_deciseconds;

assign pe_deciseconds =
   (deciseconds == 4'd0);

initial begin
  seconds <= 5'd24;
  deciseconds <= 4'd0;
end

always @(posedge CLK or posedge SW3
  or posedge SW1 or posedge RST) begin
  if (SW3 | SW1 | RST) begin
```

```
    seconds <= 5'd24;
    deciseconds <= 4'd0;
  end
  else begin
    seconds <= (SW4 & pe_deciseconds) ?
      ((seconds == 5'd0) ? 5'd24 :
      (seconds - 1'b1)) : seconds;
    deciseconds <= SW4 ? ((pe_deciseconds)
      ? 4'd9 : (deciseconds - 1'b1)) :
      deciseconds;
  end
end

endmodule


module timer(
  input CLK, SW1, SW2, RST,
  output reg [5:0] seconds,
  output reg [3:0] minutes
);

wire pe_deciseconds, p_seconds;
reg [3:0] deciseconds;

assign pe_deciseconds =
  (deciseconds == 4'd0);
assign p_seconds = (seconds == 6'd0);

initial begin
  seconds <= 6'd0;
  deciseconds <= 4'd0;
  minutes <= 4'd12;
end

always @(posedge CLK or posedge SW1
 or posedge RST) begin
  if (SW1 | RST) begin
    seconds <= 5'd0;
    deciseconds <= 4'd0;
    minutes <= 4'd12;
  end
  else begin
    if (SW2) begin
      deciseconds <= pe_deciseconds ?
        4'd9 : (deciseconds - 1'b1);
      seconds <= (pe_deciseconds) ?
        ((minutes == 4'd0) ? 6'd0 :
        ((p_seconds) ? 6'd59 :
        (seconds - 1'b1))) : seconds;
      minutes <= (p_seconds &
      pe_deciseconds) ?
      ((minutes == 4'd0) ? 4'd0 :
      (minutes - 1'b1)) : minutes;
    end
    else begin
      deciseconds <= deciseconds;
      seconds <= seconds;
      minutes <= minutes;
    end
  end
end

endmodule

module scoreTally(
  input CLK, PB0, PB1, PB2, PB3, PB4,
  output reg [7:0] Score1, Score2,
  output wire PE0
);

reg RPB0, RPB1, RPB2, RPB3, RPB4;
wire PE1, PE2, PE3, PE4;

assign PE0 = PB0 & ~RPB0;
assign PE1 = PB1 & ~RPB1;
assign PE2 = PB2 & ~RPB2;
assign PE3 = PB3 & ~RPB3;
assign PE4 = PB4 & ~RPB4;

initial begin
  RPB0 <= 1'b0;
  RPB1 <= 1'b0;
  RPB2 <= 1'b0;
  RPB3 <= 1'b0;
  RPB4 <= 1'b0;
  Score1 <= 8'd0;
  Score2 <= 8'd0;
end
always @(posedge CLK) begin
  RPB0 <= PB0;
  RPB1 <= PB1;
  RPB2 <= PB2;
  RPB3 <= PB3;
  RPB4 <= PB4;

  if (PE0) begin
    Score1 <= 8'd0;
    Score2 <= 8'd0;
  end
  else begin
    if (PE1 & ~PE2) begin
      Score1 <= (&Score1) ?
        8'd0 : (Score1 + 1'b1);
    end
    else if (~PE1 & PE2) begin
      Score1 <= (|Score1) ?
        (Score1 - 1'b1) : 8'd0;
    end
    else begin
      Score1 <= Score1;
    end
```

```
    if (PE3 & ~PE4) begin
      Score2 <= (&Score2) ?
        8'd0 : (Score2 + 1'b1);
    end
    else if (~PE3 & PE4) begin
      Score2 <= (|Score2) ?
        (Score2 - 1'b1) : 8'd0;
    end
    else begin
      Score2 <= Score2;
    end
  end
end

endmodule
```

This code defines the basketball scoring system designed to manage various aspects of a basketball game, including the shot clock, game timer, and score tracking for two teams.

The bball_scoring module serves as the top-level entity, connecting the other modules and interfacing with the external environment. It takes input signals from switches (SW1-SW4) and push buttons (PB0-PB4), as well as the system clock (CLK), and provides output signals representing the shot clock, game timer, and scores for two teams.

The shotClock module is responsible for managing the shot clock, which counts down from 24 seconds. It receives inputs from switches (SW1, SW3, SW4) and the system clock (CLK) and generates output signals representing the remaining seconds (shotClock_S) and deciseconds (shotClock_dS). The shot clock is reset to 24 seconds whenever one of the switches or the reset signal (RST) is triggered.

The timer module handles the game timer, which tracks the elapsed time of the basketball game. It receives inputs from switches (SW1, SW2) and the system clock (CLK) and generates output signals representing the elapsed minutes (timer_M) and seconds (timer_S). Similar to the shot clock, the game timer can be reset to 12 minutes whenever the appropriate switch or the reset signal (RST) is activated.

The scoreTally module tracks and updates the scores for two basketball teams. It receives inputs from push buttons (PB0-PB4) and the system clock (CLK) and generates output signals representing the scores for two teams (Score1, Score2). For the push buttons, only their positive edge are considered as input to prevent incessant scoring or resetting. The module also produces the pulse output (PE0) indicating a game reset (RST) condition. The scores are incremented or decremented based on the button presses, with the score for each team resetting to zero when the game reset condition is detected.

III. Output Circuit

Fig. 1 depicts the output circuit derived from the code discussed earlier. This circuit is composed of three constituent modules responsible for the shot clock, timer, and scoring functionalities. Each of these modules is represented by individual boxes within the larger output circuit diagram, as illustrated in Fig. 1. The modularity of this design approach enhances the overall organization and clarity of the system architecture.

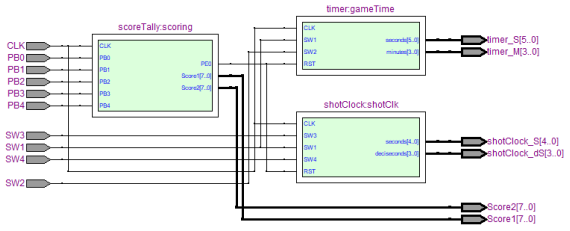

Fig. 1. Basketball Scoreboard Main Module (bball_scoring) RTL Netlist

The shot clock and timer modules are designed to operate with a clock frequency of 1 decisecond (0.1 seconds) to ensure precise timekeeping. In the shot clock module, the decisecond value is continuously decremented until it reaches zero, triggering a borrow from the seconds variable. This process repeats until the seconds variable also reaches zero or is reset, at which point the shot clock resets to 24.0 seconds. The decrementing operation is facilitated by components such as adders, multiplexers, and flip-flops, depicted on the left side of Fig. 3, which illustrates the output circuit of the shot clock module. Notably, the flip-flop enables for the deciseconds and seconds variables are directly connected to SW4, allowing counting to be paused when SW4 is toggled off. Additionally, the reset inputs (SW1, SW3, and the positive edge of the master reset switch, PB0) are logically ORed to the clear (CLR) inputs of the flip-flops controlling the deciseconds and seconds variables, ensuring they are reset to their initial states as needed. The remaining components of the circuit manage the decrementing of seconds and handle the inputs and outputs of the overall system.

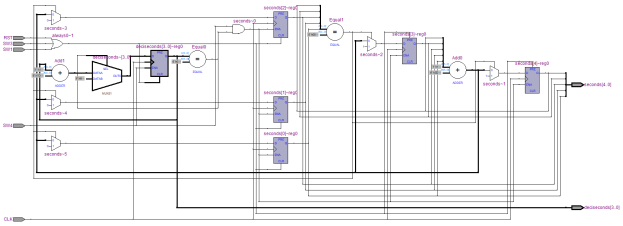

Fig. 2. Baskteball Scoreboard Shot Clock Module (shotClk) RTL Netlist

The timer module, depicted in Fig. 3, functions similarly to the shot clock module but with additional functionality to track minutes alongside seconds. In this module, the seconds variable has a maximum value of 59, and a new layer representing minutes (with a maximum value of 12) is introduced. Like the shot clock, when the deciseconds reach zero, it triggers a borrow for the seconds variable. Similarly, when seconds reach zero, it triggers a borrow from the minutes variable. However, unlike the shot clock module, once the minutes variable reaches zero, it does not automatically reset to its

maximum value unless manually reset with the designated inputs. The enable of the flip-flops controlling the variables is governed by SW2, allowing for the pausing and resuming of counting. SW1 or the positive edge of PB0 is connected to the clear (CLR) or preset (PRE) inputs of the flip-flops, ensuring they reset to their initial states when triggered.

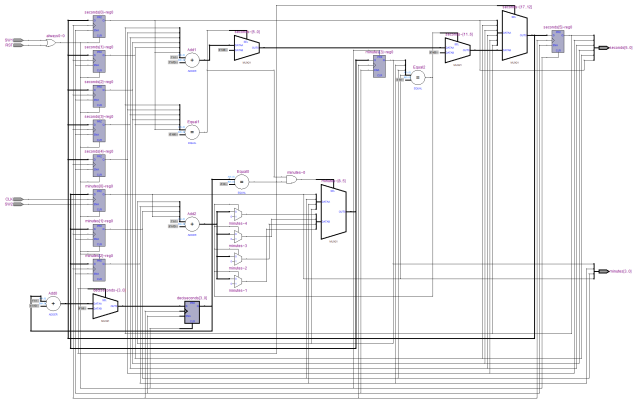

Fig. 3. Basketball Scoreboard Timer Module (gameTime) RTL Netlist

Fig. 4 illustrates the scoring module, also known as scoreTally, within the scoreboard system. The flip-flops on the left-hand side store the current values of Score1 (at the bottom) and Score2 (at the top). In the middle, a larger flip-flop, along with the combinational circuitry on its right, detects the positive edges of the push buttons until the next clock (CLK) pulse. This mechanism ensures that these inputs are only registered once per clock cycle, maintaining accuracy in the scoring process. Subsequently, these registered inputs serve as conditions to determine whether to increment or decrement the values of Score1 and Score2 accordingly.

IV. Simulation

Fig. 5 depicts the simulation of the basketball scoreboard system's normal operation. Here, SW1 and SW3 remain in a LOW state, while SW2 and SW4 remain HIGH. As a result, both the shot clock and timer initiate at their maximum values of 24.0 seconds and 12:00, respectively, and decrement accordingly. Regarding the scoring system, activating PB1 increments Score1, while PB2 decrements it. Likewise, PB3 and PB4 perform similar functions for Score2, incrementing and decrementing it, respectively.

Fig. 6 demonstrates the utilization of various input controls for both the timer and shot clock within the basketball scoreboard system. Notably, SW1 is responsible for resetting both the timer and shot clock to their initial values, while SW3 exclusively resets the shot clock. This distinction becomes evident when SW1 is transitioned back to a LOW state, prompting the timer to commence counting while the shot clock remains in its initial state due to the continued activation of SW3. When SW4 is set to LOW, the shot clock's seconds and deciseconds maintain their current state, effectively pausing the timer. Similarly, toggling SW2 to LOW pauses the timer's minutes and seconds. Additionally, the system prevents Score1

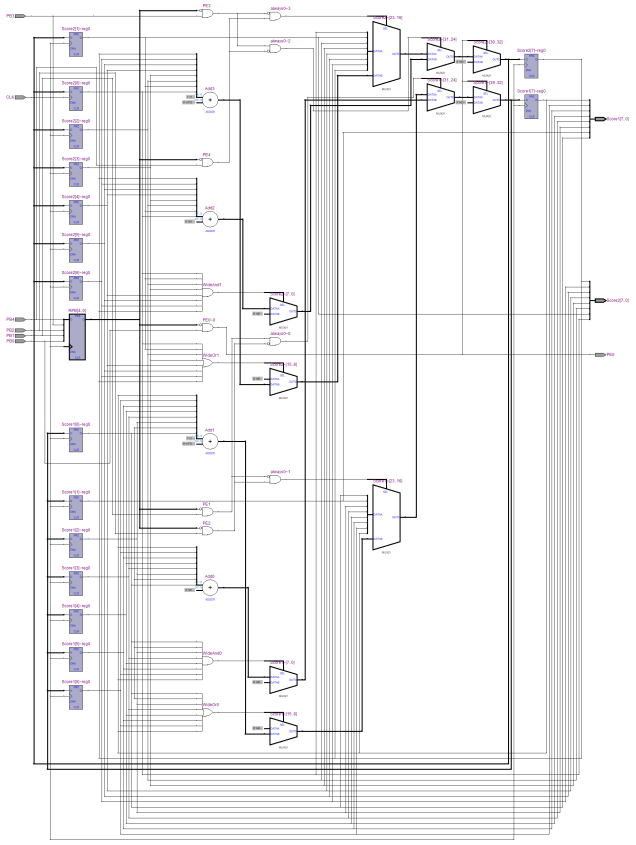

Fig. 4. Basketball Scoreboard Scoring Module (scoreTally) RTL Netlist

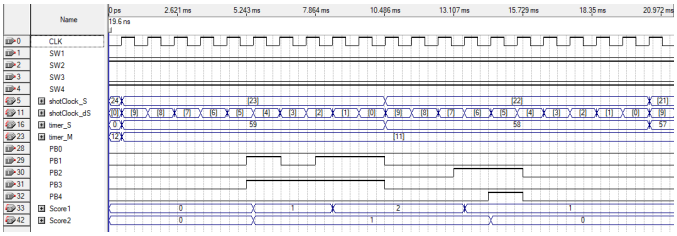

Fig. 5. Basketball Scoreboard Normal Operation Simulation Waveforms

and Score2 from being decremented below zero, and upon reaching 255, their values reset to 0. Finally, activating PB0 (master reset) restores all output values to their initial states, ensuring the system returns to its starting configuration.

In Fig. 7, the scenario where the timer reaches 00:00 is depicted. Notably, the timer remains at this value indefinitely, while the shot clock continuously resets to its maximum value. Only when SW1 is toggled to HIGH or when PB0 is activated (which also resets the score) will the timer reset, initiating a new countdown.

The simulation results presented are consistent with the defined functionalities, affirming the effective operation of the basketball scoreboard system. These outcomes demonstrate that the shot clock, timer, and scoring functionalities are correctly implemented and functioning as intended, validating the system's reliability and accuracy.

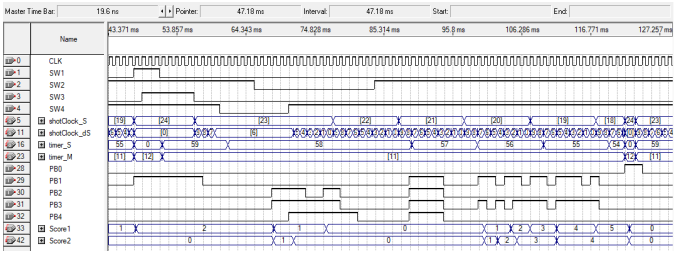

Fig. 6. Timer and Shot Clock Input Controls Simulation Waveforms

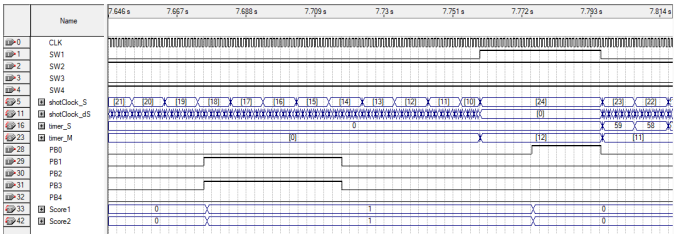

Fig. 7. Timer Runs Out Simulation Waveform