Key pad Project

Min Han

02.15.2024

Introduction

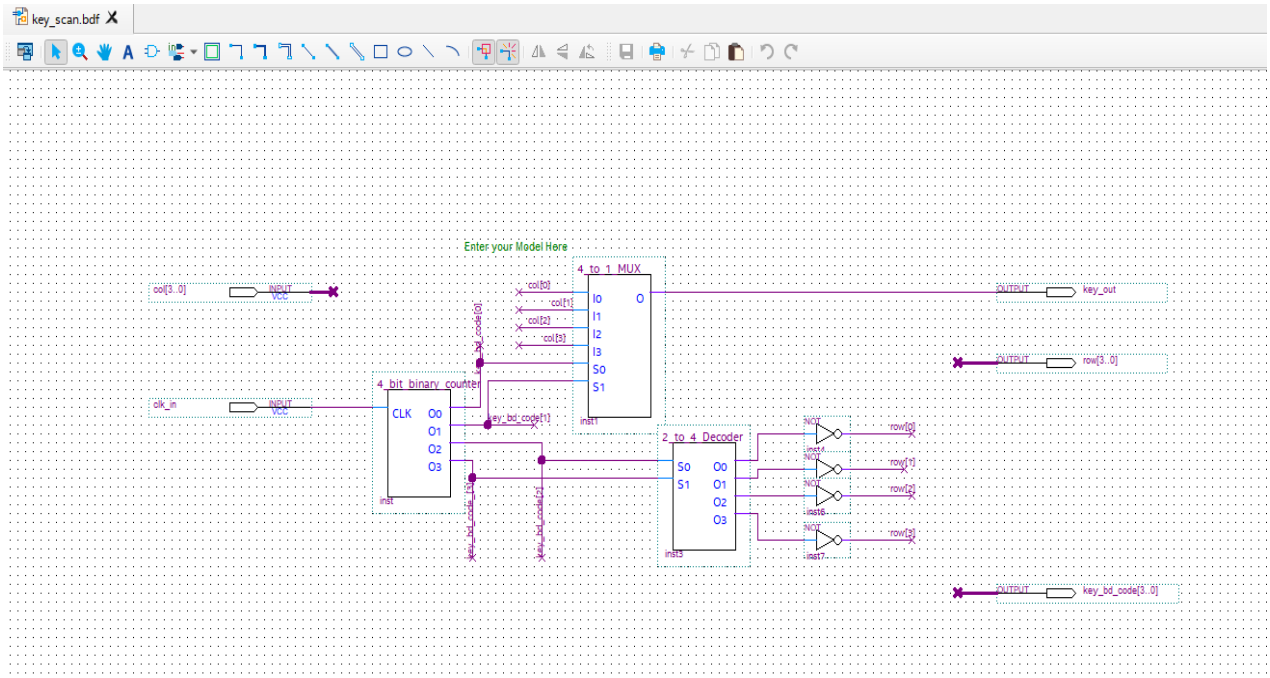
: In this laboratory, the task focuses on addressing the technical challenges associated with interfacing digital hardware with mechanical input devices, notably a 16-key keypad. I am required to employ schematic capture and hardware description language to devise a solution that mitigates key bounce issues and employs time multiplexing to streamline the connection between the keypad and digital systems. The outcome, a design that effectively decodes keypad inputs to display corresponding hexadecimal symbols on a seven-segment LED, emphasizes the critical thinking and technical skills necessary for real-world digital logic applications.

<Phase 1>

Experiment Description:

In Phase 1 of the "Scanning Keypad Interface", I embark on designing the key_scan module, which is pivotal for interfacing a 16-key keypad with the Terasic DE2-115 FPGA platform. Utilizing schematic capture techniques, this module is crafted to decode signals from keypad presses, incorporating a binary counter, multiplexer, decoder, and basic logic gates to scan and interpret key presses. The objective is to translate these presses into unique codes for display on the FPGA's seven-segment LEDs, with an emphasis on functionality and the ability to produce discernible output patterns for validation by the instructor, setting the stage for further refinement in subsequent phases.

Design and Implementation:



Results:

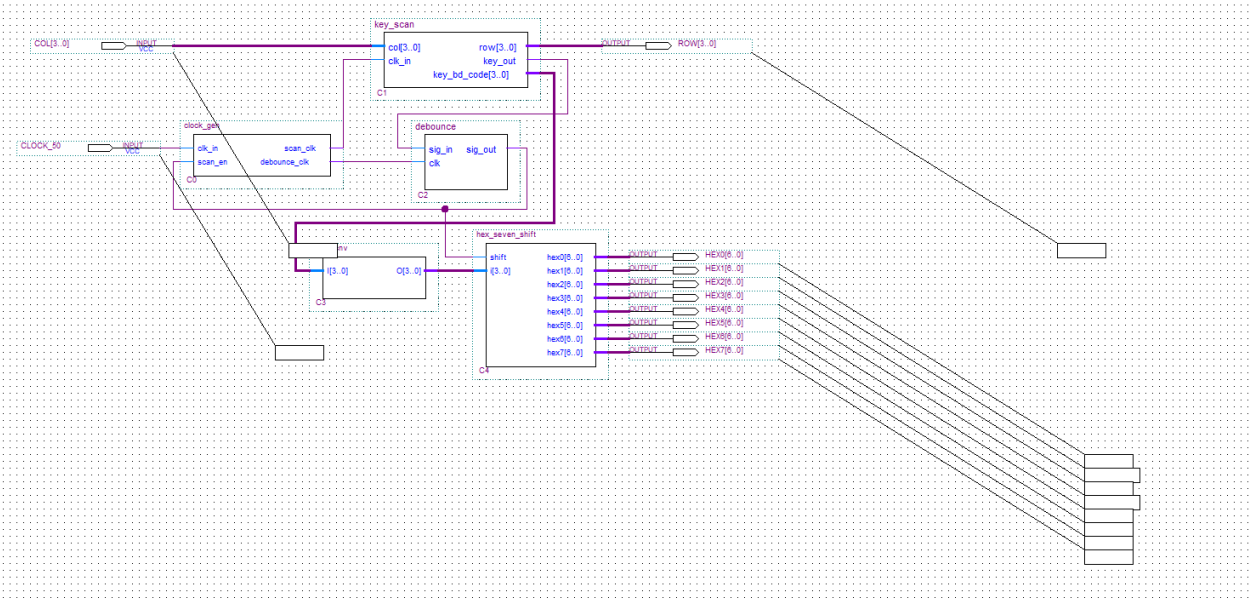
The keypad is functioning, but pressing the numbers does not result in the appropriate output.

<Phase 2>

Experiment Description:

I address the key bounce problem by creating a debounce module. In this step, schematic capture is used to build a circuit that filters out unwanted electrical noise that is produced when a switch is pressed, causing it to connect several times quickly. Through the use of components such as shift registers and D-flip flops, the design guarantees that a key press occurs only after the input has remained constant for a set number of clock cycles—eight in this case.

Design and Implementation:



Results:

The numbers are sorted in order, but the corresponding outputs for each number input are not sorted.

<Phase 3>

Experiment Description:

Phase 3 of the "Scanning Keypad Interface" requires me to develop the code_cnv module, which maps keypad inputs to equivalent hexadecimal symbols for display using Verilog HDL. The task is to make sure that, in accordance with the guidelines provided in the lab handbook, the correct output symbol appears after pressing each key.

Design and Implementation:

```
// 4-bit to 4-bit keypad code converter model
// converts 4-bit keypad code into a 4-bit
// value expressed by Table 1 in Lab 4
// assignment
module code_cnv(O, I);
   input [3:0] I;
   output reg [3:0] O;

   always @ (I) begin
      case(I)
         4'b0000: O = 4'h0; // Key 0
         4'b0001: O = 4'h1; // Key 1
         4'b0010: O = 4'h2; // Key 2
         4'b0011: O = 4'h3; // Key 3
         4'b0100: O = 4'h4; // Key 4
         4'b0101: O = 4'h5; // Key 5
         4'b0110: O = 4'h6; // Key 6
         4'b0111: O = 4'h7; // Key 7
         4'b1000: O = 4'h8; // Key 8
         4'b1001: O = 4'h9; // Key 9
         4'b1010: O = 4'hA; // Key A
         4'b1011: O = 4'hb; // Key B
         4'b1100: O = 4'hC; // Key C
         4'b1101: O = 4'hd; // Key D
         4'b1110: O = 4'hE; // Key *
         4'b1111: O = 4'hF; // Key #
         default: O = 4'h0;  // Default case
      endcase
   end
endmodule
```

Results:

Now the numbers are finally sorted correctly.

Conclusion:

To sum up, the lab gave me a thorough understanding of how to interface high-speed digital hardware with a mechanical device, like a keypad. Design elements were easier to integrate into a bigger system because to our practical knowledge with hardware description language design entry methods and schematic capture. Notable successes included resolving the key bounce issue and putting in place a time-multiplexed interface to reduce the number of I/O connections. I learned a lot about practical difficulties in digital system design through this lab, like debouncing and using hardware description languages to implement complex logic functions. The knowledge and abilities gained are useful not just for complex logic design but also for a variety of technical issues pertaining to digital hardware interfaces.