Lab 06: Da Vinci Code

Submission deadlines:

Source Code:	2019/11/19 18:30
Report:	2019/11/24 23:59

Objective

- ➢ To be familiar with finite state machines with Verilog.
- ➢ To be familiar with the keyboard control on FPGA

Description

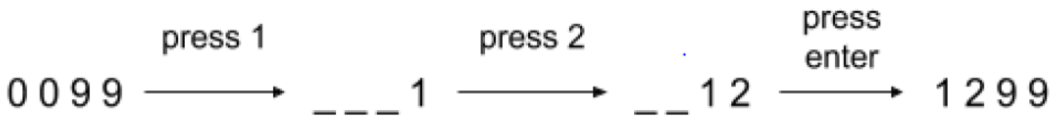
Da Vinci Code is a well-known mathematical game. In this lab, we want you to design the Da Vinci Code game for us and implement it on the FPGA board with a keyboard. The game starts with guessing an untold secret number among the initial range of 00 ~ 99. The range will get smaller and smaller as long as you make more and more guesses. It won't stop until you figure out the right number.

- Upon the reset, the secret number **goal** is set to zero, and the 7-segment will display "- - - -" (four dashes) at the beginning of the game until you press the button **start (BTNU)**.
- Generate a number among 01 and 98 randomly and update it to the number **goal.** This is the secret number to be guessed! (Ex: the goal is 25.)
- Start the game after pressing the **start,** and the 7-segment will display "0099" before the first input (i.e., the first guess). The display "0099" indicates that the number to be guessed is from 01 to 98 (00 < goal < 99).
- Enter the number you want to guess with the keyboard. You should enter **exactly two digits** of the number, i.e., 05, 37, and so on. A single digit is invalid, such as '7'.
- If you enter more than two digits, only the last two digits you entered will be taken into account.
- Press the **enter (keyboard)** after your typing of the number.
- If you key in an out-of-range number, or if you just key in a single digit, then press enter, it will do nothing and display the latest range before your input.
- When you make a wrong guess, the range will shrink accordingly.

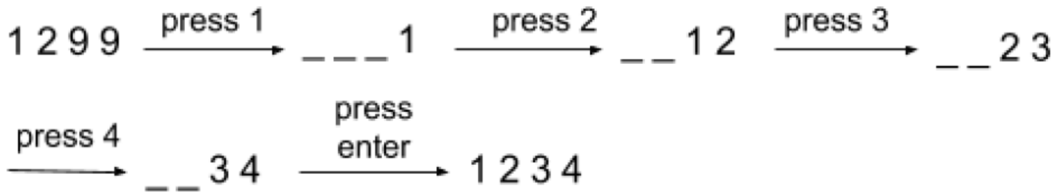
See the following Cases for example.

- Repeat the guesses until your answer is the same as the number **goal**.
- Whenever you press the button **reset (BTNC)**, the number **goal** should be reset to zero with the 7-segment displaying “- - - -”. Then the Da Vinci code game can restart after pressing **start**.
- Once you make the right guess, the 7-segment should display the number **goal** at both left two and right two digits (see the Cases). And all the **LED (LD15 ~ LD0)** will light up simultaneously for a clock cycle (clk/2^25).
- Get back to the beginning of the game, which displays “- - - -”, after the **goal** is displayed.
- There is one **cheat (BTNL)** button. After pressing **start,** you can press the **cheat**, the two leftmost 7-segment digits will display the number of the **goal**. The other two digits are left blank. When you release the cheat button, the display resumes to its original operation. The cheat **button** cannot be pressed when displays “- - - -”.
- Cases:
  There are some examples as follows, assuming the secret number is 25:

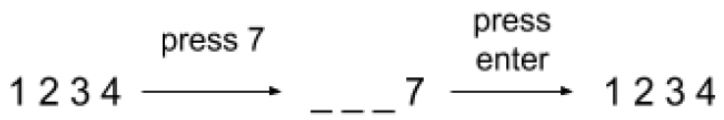
Case 1*:* Key in “1” and then key in “2” and press **enter** . The range becomes 12-99 because 12 < 25 < 99.



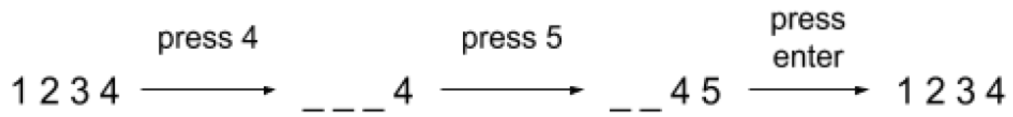
Case 2: Key in “1”, “2”, “3”, “4” and press **enter**. The effective input will be 34. The range updates to 12-34 because now 12 < 25 < 34.



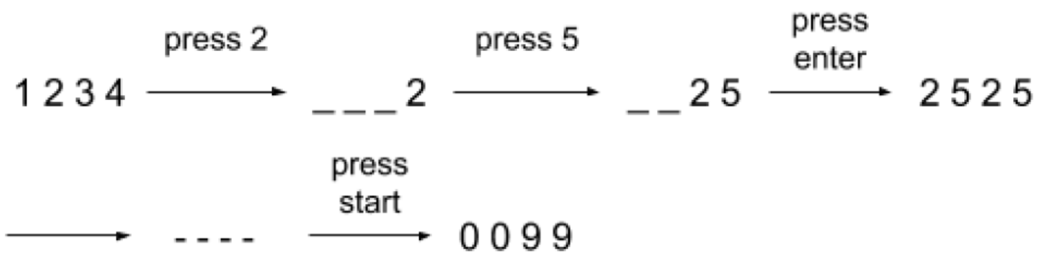
Case 3: Key in ”7” and press **enter**. It is an error so the range will not change.



Case 4: Key in an out of range number. It is an error again. The range will not change.



Case 5: The number goal is 25 and you key in “2”, “5”, and **enter**. It hits the secret goal! It will be display on the 4 digits and all the LED will be turned on at the same time for a clock cycle of clk/(2^25). Afterward it starts all over again and displays “- - - -”.
Note: The underlines here (e.g., “_ _ _”) indicate blank digits on the 7-segment display.



I/O signal specification:

- ***clk***: the clock signal with the frequency of 100MHz (connected to pin **W5**).
- ***rst***: the asynchronous active-high reset (connected to **BTNC**).
- ***start*:** to start the game (connected to **BTNU**).
- ***cheat***: to show **goal** on the two leftmost 7-segment digit (connected to **BTNL**).
- ***LED[15:0]***: to be on when you figure out the answer (connected to **LD15 ~ LD0**).
- **DIGIT[3:0]**: the signals to enable one of the 7-segment digits.
- **DISPLAY[6:0]**: the signals to show the digits on the 7-segment display.

Note:

1. **All the signals of the pushbutton should be properly processed with the debounce and one pulse (except the *cheat* button).**
2. **The clock frequency of each debounce or one-pulse circuit is clk/(2^16)**
3. **The clock frequency of the seven-segment display controller is clk/(2^13)**
4. **The clock frequency of the LED display is clk/(2^25).**
5. **Demo video:** **https://drive.google.com/open?id=1mi5uDdlaRtdiCJM68cMD95GAljOU0PnN**

Hint:

1. You can generate the random number with the counter of the clock divider. Use the start button to sample a number from the counter, say, counter [6:0]. And make sure it is within the range of 01~98 (Simply adjust it if not).

2. You can use the following template for your design.

```
module lab06(
  output wire [6:0] DISPLAY,
  output wire [3:0] DIGIT,
  output reg [15:0] LED,
  inout wire PS2_DATA,
  inout wire PS2_CLK,
  input wire rst,
  input wire clk,
  input wire start,
  input wire cheat
  );
        //add your design here
Endmodule
```

Attention:

1. You should hand in only one Verilog file, lab06.v. If you have several modules in your design, integrate them in lab06.v. (Please do not hand in any compressed files, which will be considered as an incorrect format.)
2. You should also hand in your report as lab06_report_StudentID.pdf (i.e., lab06_report_107066666.pdf).
3. You should be able to answer questions of this lab from TA during the demo.
4. You need to generate bitstream before demo.