Term Project

Important dates:

Simulation demo	18-20/01/2023 (will be scheduled and announced in SuCourse) You can do your implementation demo also if you are ready.
Submission deadline to SuCourse	22/01/2023, 23:55
Implementation demo	23/01/2023 (will be scheduled and announced in SuCourse)

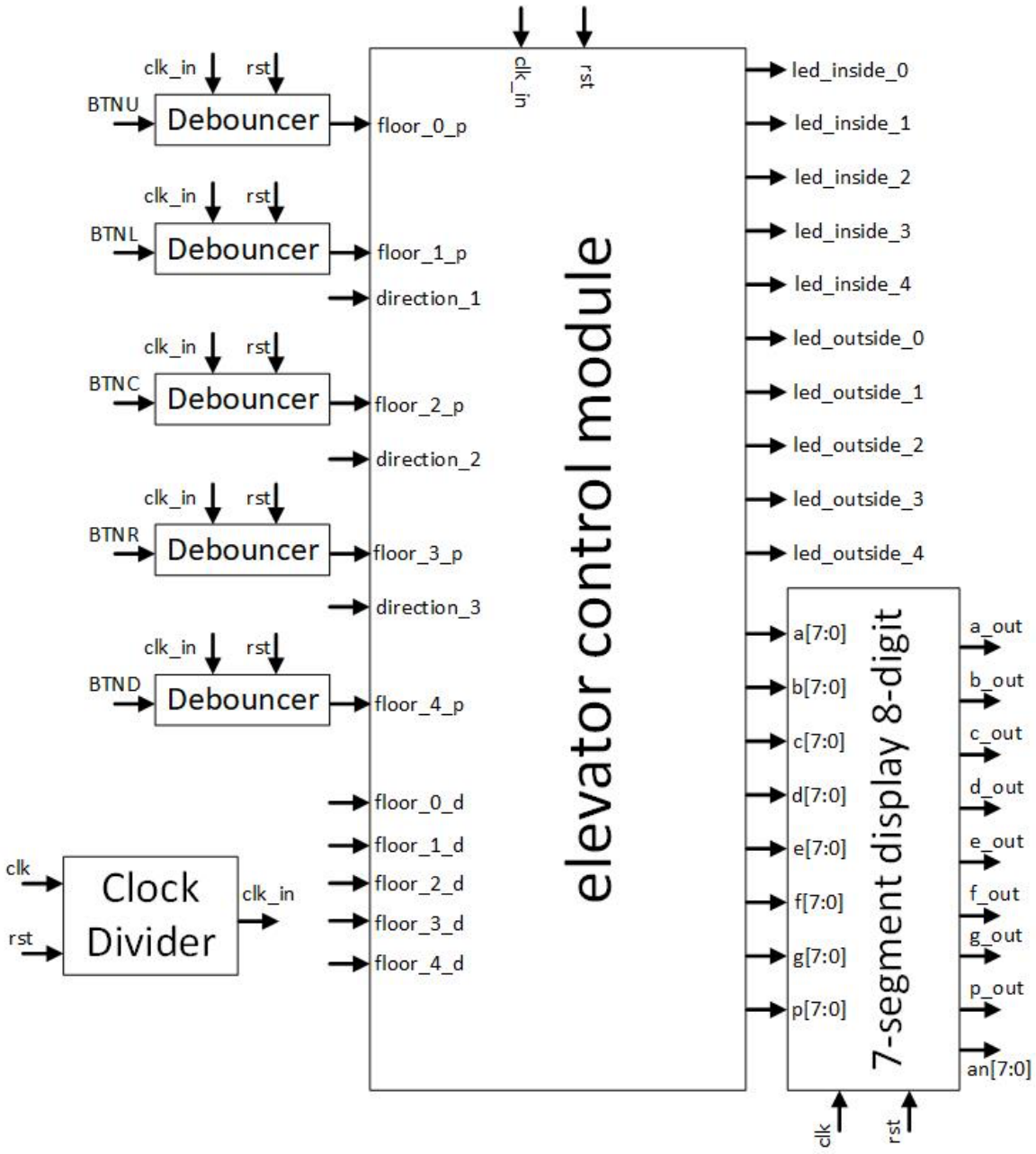

Figure 1: Term Project block diagram

An Elevator Control System

You will design a simple elevator control system and implement it on the FPGA device. Users can control the elevator using buttons and switches. Users can monitor the status of the elevator from the LEDs and seven-segment display in your FPGA. The elevator will be designed for a building with 5 floors.

- **You can work in groups of two and you can select your teammate.**
- **There will be simulation demo dates, in which you can show your simulation results and work on FPGAs to implement your design.**
- **If you can implement your design on FPGA in the simulation demo day, you do not have to wait for the implementation demo day. You can submit your project.**
- **If you can not finish your design and implement it in simulation demo, you will have a couple of days extra time to complete your design. You can show your implementation in the implementation demo in this case.**
- **There will not be any extra time after the implementation demo. You also need to submit your project (detailed report, your files) the day before the implementation demo.**
- **Please check important dates and do not miss them!**

Details of the system are given below:

Note: Pin names that you should use **in your Verilog code** are given **in italic**. **Pin names** that you need to use in your **constraint file** are given **in parentheses ( ).**

Inputs

There will be 15 inputs in your design:

- *clk_in* is the clock signal required for sequential operation of your design. You need to use a clock divider module to reduce your clock speed. Your FPGA has 100 MHz internal clock (clk). The *clock_divider* module will take the 100 MHz clock and generate a 50 Hz (20 ms) clock signal *clk_in* as an output, which you use as an input to your elevator control module.
- *rst* will reset your design. This input should be provided from CPU Reset (C12) push button of the Nexys board. This push button works differently than the other push buttons. If it is not pressed, it provides a logic high '1' and if it is pressed it provides a logic low '0'.

- 5 push buttons and 3 switches will be used to call the elevator from any floor outside the elevator:
    - *floor_0_p*: Push button BTNU (M18) will be used to call the elevator. Since floor 0 is the lowermost floor, there is no need for a direction switch.

- *floor_1_p*: Push button BTNL (P17) and switch SW1 (L16) (*direction_1*) will be used to call the elevator. If SW1 (*direction_1*) is **low**, the elevator is called to go **down**. If SW1 is **high**, the elevator is called to go **up**.
- *floor_2_p*: Push button BTNC (N17) and switch SW2 (M13) (*direction_2*) will be used to call the elevator. If SW2 (*direction_2*) is **low**, the elevator is called to go **down**. If SW2 is **high**, the elevator is called to go **up**.
- *floor_3_p*: Push button BTNR (M17) and switch SW3 (R15) (*direction_3*) will be used to call the elevator. If SW3 (*direction_3*) is **low**, the elevator is called to go **down**. If SW3 is **high**, the elevator is called to go **up**.
- *floor_4_p*: Push button BTND (P18) will be used to call the elevator. Since floor 4 is the uppermost floor, there is no need for a direction switch.

- 5 switches will be used to enter the destination floor inside the elevator.

  - *floor_0_d*: User should change the position of the switch SW15 (V10) to **high** if he/she wants **to go to floor 0**. User should change the position of the switch SW15 (V10) **back to low** to let the switch be free to be used again.
  - *floor_1_d*: User should change the position of the switch SW14 (U11) to **high** if he/she wants **to go to floor 1**. User should change the position of the switch SW14 (U11) **back to low** to let the switch be free to be used again.
  - *floor_2_d*: User should change the position of the switch SW13 (U12) to **high** if he/she wants **to go to floor 2**. User should change the position of the switch SW13 (U12) **back to low** to let the switch be free to be used again.
  - *floor_3_d*: User should change the position of the switch SW12 (H6) to **high** if he/she wants **to go to floor 3**. User should change the position of the switch SW12 (H6) **back to low** to let the switch be free to be used again.
  - *floor_4_d*: User should change the position of the switch SW11 (T13) to **high** if he/she wants **to go to floor 4**. User should change the position of the switch SW11 (T13) **back to low** to let the switch be free to be used again.

Outputs

There will be 2 (two) different types of outputs in your design, LEDs and 7-segment displays (SSDs):

- 5 LEDs will be used to show which switches are used to go destination floors for the people inside the elevator:

  - *led_inside_0*: If a user changes the position of the switch SW15 (V10) to **high to go to floor 0**, led LD15 (V11) will **turn ON** and will **stay turned-ON** till the elevator goes to floor 0 and stops there.

- *led_inside_1*: If a user changes the position of the switch SW14 (U11) to **high to go to floor 1**, led LD14 (V12) will **turn ON** and will **stay turned-ON** till the elevator goes to floor 1 and stops there.
- *led_inside_2*: If a user changes the position of the switch SW13 (U12) to **high to go to floor 2**, led LD13 (V14) will **turn ON** and will **stay turned-ON** till the elevator goes to floor 2 and stops there.
- *led_inside_3*: If a user changes the position of the switch SW12 (H6) to **high to go to floor 3**, led LD12 (V15) will **turn ON** and will **stay turned-ON** till the elevator goes to floor 3 and stops there.
- *led_inside_4*: If a user changes the position of the switch SW11 (T13) to **high to go to floor 4**, led LD11 (T16) will **turn ON** and will **stay turned-ON** till the elevator goes to floor 4 and stops there.

- 5 LEDs will be used to show which push buttons are pressed to call the elevator for the people outside the elevator:

  - *led_outside_0*: If user presses push button BTNU (M18) to call the elevator to floor 0, led LD0 (H17) will **turn ON** and will **stay turned-ON** till the elevator comes to floor 0 and stops there.
  - *led_outside_1*: If user presses push button BTNL (P17) to call the elevator to floor 1, led LD1 (K15) will **turn ON** and will **stay turned-ON** till the elevator comes to floor 1 and stops there.
  - *led_outside_2*: If user presses push button BTNC (N17) to call the elevator to floor 2, led LD2 (J13) will **turn ON** and will **stay turned-ON** till the elevator comes to floor 2 and stops there.
  - *led_outside_3*: If user presses push button BTNR (M17) to call the elevator to floor 3, led LD3 (N14) will **turn ON** and will **stay turned-ON** till the elevator comes to floor 3 and stops there.
  - *led_outside_4*: If user presses push button BTND (P18) to call the elevator to floor 4, led LD4 (R18) will **turn ON** and will **stay turned-ON** till the elevator comes to floor 4 and stops there.

- SSDs will show the status and current position of the elevator. The left SSD will show the status and the right SSD will show the current position (in which floor the elevator is) of the elevator. The left and the right SSD have four 7-segment displays (8 SSDs in total). So, it is possible to write a word or number with 8 letters/digits. In this project, the left four displays will be used to write --UP', --DO', --Id' to show the status (UP, DOWN, IDLE). The right displays will be used to write FL-X. Here '-' means nothing to write and 'X' means the digit showing the floor information (0,1,2,3,4).

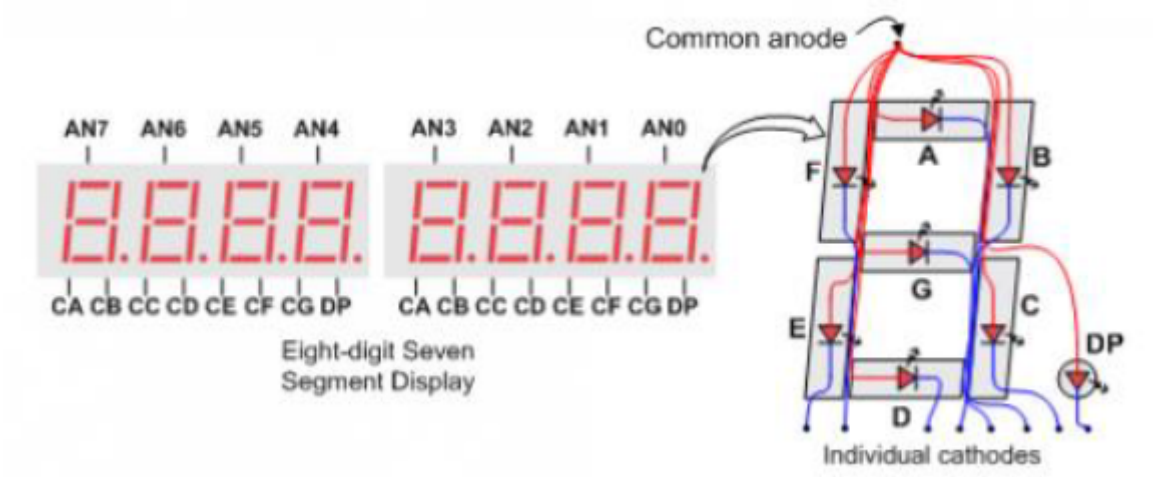

Figure 2: 8-digit seven-segment display in Nexys A7 FPGA board

- You will be given an 'ssd' module to display your messages on SSDs. Figure 2 shows how SSDs operate in Nexys A7 FPGA board. Each SSD (digit) has 8 diodes (A, B, C, D, E, F, G, P). The anodes of all 8 diodes are connected, so they have a common anode. Their cathodes can be controlled separately. To display anything on the SSD, you need to activate the common anode of that SSD (digit) and the cathodes you need to activate. Both anode and cathodes work active-low, meaning you need to assign logic-0 to activate the diodes. For example, if you want to display A on the leftmost SSD, you need to activate AN7 and you need to activate A, B, C, E and F of the leftmost digit. You don't need to do anything for AN7, the 'ssd' module handles it. To do that you need to provide the following inputs to 'ssd' module:
    - a[7:0]=8'b01111111, b[7:0]=8'b01111111, c[7:0]=8'b01111111, d[7:0]=8'b11111111, e[7:0]=8'b01111111, f[7:0]=8'b01111111, g[7:0]=8'b11111111, p[7:0]=8'b01111111
- P is used to display a dot (.), so you do not need to activate P in this project. If you want to display B on the rightmost SSD, you need to activate AN0 and you need to activate A, B, C, D, E, F and G. You don't need to do anything for AN0, the 'ssd' module handles it. To do that you need to provide the following inputs to 'ssd' module:
    - a[7:0]=8'b11111110, b[7:0]=8'b11111110, c[7:0]=8'b11111110, d[7:0]=8'b11111110, e[7:0]=8'b11111110, f[7:0]=8'b11111110, g[7:0]=8'b11111110, p[7:0]=8'b11111111
- If you want to display '--Id' on the left SSD, and 'FL-2' on the right SSD, you need to provide the following inputs to 'ssd' module:
    - a[7:0]=8'b11110110, b[7:0]=8'b11001110, c[7:0]=8'b11001111, d[7:0]=8'b11101010, e[7:0]=8'b11100010, f[7:0]=8'b11110011, g[7:0]=8'b11100110, p[7:0]=8'b11111111
- This is already done in module 'ssd'. You just need to know what you need to write and which anodes and cathodes you need to activate to write your message. If you are interested in how the SSDs work and understand the 'ssd' module better, you can refer to Nexys A7 FPGA Reference Manual, which is given under 'Term Project' folder of SuCourse.

Operation

- The elevator starts in IDLE status and at floor 0 when ‘rst’ input is asserted, in which it displays ‘--Id’ on the left SSD and ‘FL-0’ on the right SSD.
- In IDLE status, users can call the elevator from any floor by using push buttons and direction switches. Please note that the user should first select the direction (UP/DOWN) and then should press the push button in his/her floor.
- If the elevator stopped at any floor and there is no other command (there is no pressed push button outside the elevator and any of the destination switches are not changed to high and back to low position inside the elevator), the elevator status is IDLE and the left display should show ‘--Id’ and the right display should show the floor information. For example, if the elevator stopped at floor 2, the right display should show ‘FL-2’.
- If the elevator is moving up, the status should be UP and the left SSD should display ‘--UP’. While the elevator is moving, the right display should display the floor information. For example, if the user is going from floor 0 to floor 3, the right display should display ‘FL-0’, ‘FL-1’, ‘FL-2’ and finally ‘FL-3’ respectively.
- If the elevator is moving down, the status is DOWN and the left SSD should display ‘--DO’. While the elevator is moving, the right display should display the floor information. For example, if the user is going from floor 3 to floor 0, the right display should display ‘FL-3’, ‘FL-2’, ‘FL-1’ and finally ‘FL-0’ respectively.
- While the elevator is moving, if someone calls the elevator by using push buttons and direction switches, there are 3 scenarios:
    - If the direction is NOT same, the elevator should ignore the request. For example, if the elevator is going from floor 1 to floor 3 and someone calls the elevator from floor 0, the request should be ignored.
    - If the direction is same and the request comes from a floor in the range between the current floor and destination floor, the elevator should stop at the requested floor and then should continue in the same direction. The elevator should first stop at the newly requested floor in the range and should continue to the destination floor requested first. For example, the elevator is initially going from floor 0 to floor 4. While it is moving and at floor 1, someone calls the elevator from floor 2 to go UP. Then he/she wants to go to floor 3 by using destination switches inside the elevator. In this example, the elevator should stop at floor 2 and take this person and should stop at floor 3 to drop this person. Then, it should continue to floor 4, which was initially requested.
    - If the direction is same and the request comes from a floor NOT in the range between the current and destination floors, the elevator should take the request into the priority list. For example, the elevator is going from floor 0 to floor 2. Someone calls the elevator from floor 3 to go UP or DOWN. Then, the elevator should take the request into priority list, since floor 3 is NOT in the range between floor 0 and floor 2. Once the elevator stops at floor 2 and drops the person who is the owner of the initial request, it should go to floor 3, which is the second in the priority list. When the elevator comes to floor 3, this person

should enter the destination floor using the destination switches inside the elevator.

- People in the elevator may change their mind and decide to go to another floor before they reach their destination floor initially requested. Alternatively, when the elevator stops at any floor and new people come into the cabin, they may enter new destinations. The same rules, that are specified above for 3 different scenarios, are also valid for the requests got inside the elevator using destination switches. For example, the elevator is going from floor 0 to floor 4. Then, someone wanted to stop at floor 3 and used the destination switch by changing its position first to high and then back to low when the elevator is at floor 2. Since, the elevator is going UP and floor 3 is in the range between the current floor and destination floor, the elevator should accept this request and should drop this person at floor 3.

- Users can monitor the status of the elevator from the LEDs inside the elevator and outside the elevator:

    - If any of the destination switches are used, the corresponding LED inside the elevator is turned-ON. This LED should stay turned-ON till the elevator reaches the requested floor and stops there. The LED should turn-OFF when the elevator reaches the destination floor and stops there.
    - If any of the push buttons are used, the corresponding LED outside the elevator is turned-ON. This LED should stay turned-ON till the elevator comes to this floor and stops there. The LED should turn-OFF when the elevator comes to this floor and stops there.
    - Example: Someone wants to go from floor 0 to floor 3. Assume that the elevator is at floor 4 and in IDLE status. So, this person should use push button *floor_0_p* to go UP. Since, floor 0 is the lowest floor, there is no direction switch. When this person presses the push button, the corresponding LED *led_outside_0* should turn-ON. While the elevator is moving from floor 4 to floor 0, before it reaches floor 2, someone presses the push button *floor_2_p* and the position of the direction switch (*direction_2*) is low. Since the direction of the elevator is same with the request and the requested floor is in the range, this request is accepted, and the corresponding LED *led_outside_2* should turn-ON. Then, the elevator should go to floor 2 and stop there. When the elevator reaches floor 2 and stops there, *led_outside_2* should turn-OFF. Then, the elevator should continue to floor 0. When the elevator reaches floor 0 and stops there, *led_outside_0* should turn-OFF. Then, the person should change the position of the destination switch *floor_3_d* first to high and then back to low position. Thus, the corresponding LED *led_inside_3* should turn-ON. When the elevator reaches floor 3 and stops there, *led_inside_3* should turn-OFF.

- Users can monitor the status of the elevator from the seven-segment display (SSD):

    - If the elevator is moving and going UP, then the left SSD should display '--UP'.
    - If the elevator is moving and going DOWN, then the left SSD should display '--DO'.
    - If the elevator is NOT moving, then the left SSD should display '--Id'.
    - If the elevator is called from any floor by the push buttons or someone inside the elevator enters a destination floor by using destination switches, and the elevator reaches the called floor or destination floor, it should stop there and the left SDD should display '--Id'.
    - If the elevator is moving UP or DOWN, the right SSD should display the floor information. The elevator should go from one floor to another in 5 seconds. For example, if the elevator moves from floor 0 to floor 4, the right SSD should display 'FL-0', 'FL-1', 'FL-2', 'FL-3' and 'FL-4' respectively with 5 seconds intervals.
    - If the elevator is NOT moving, the right SSD should display the current position of the elevator. For example, if the elevator stays IDLE at floor 3, the right SSD should display 'FL-3'.

- If the elevator is called from any floor by the push buttons or someone inside the elevator enters a destination floor by using destination switches, and the elevator reaches the called floor or destination floor, it should stay in IDLE status at least 5 seconds and the left SSD should display '--Id'. If the elevator is not called from any floor or destination floor information is NOT entered, then the elevator can stay IDLE for a long time and the left SSD should display '--Id'. The right SSD should display the current position of the elevator.

- You need to use a clock divider module (clock_divider) that is given to you in "Term Project" folder of SuCourse. This module takes the FPGA internal clock *clk* and generates a clock output *clk_in* for your elevator control module with a reduced frequency of 50 Hz (20 ms). You need to use *clk_in* also in your 'debouncer' module required for push buttons.

- You need to use the debouncer module (debouncer.v) that gets the input from a push button and generates a one clock pulse output, which you will use in your elevator control module as a push button input (*floor_0_p* etc.). The clock input for the debouncer module is also *clk_in*. The debouncer module is located under 'Term Project' folder of SuCourse.