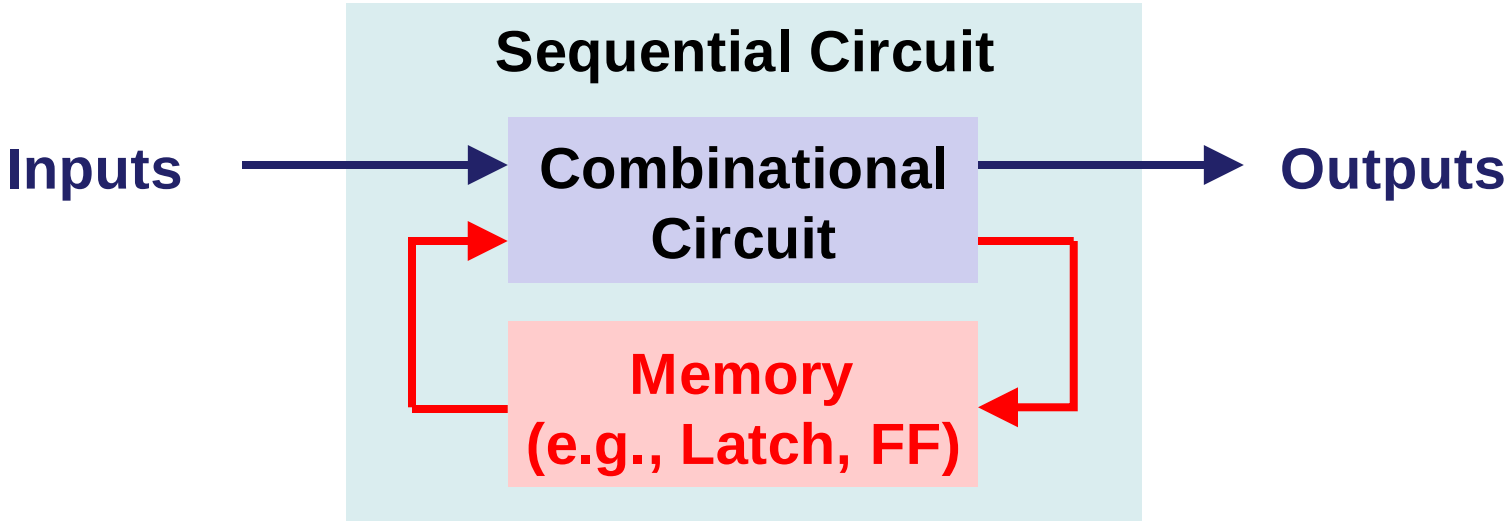
香港中文大學
The Chinese University of Hong Kong

CENG3430 Rapid Prototyping of Digital Systems

Lab03: Shift Register

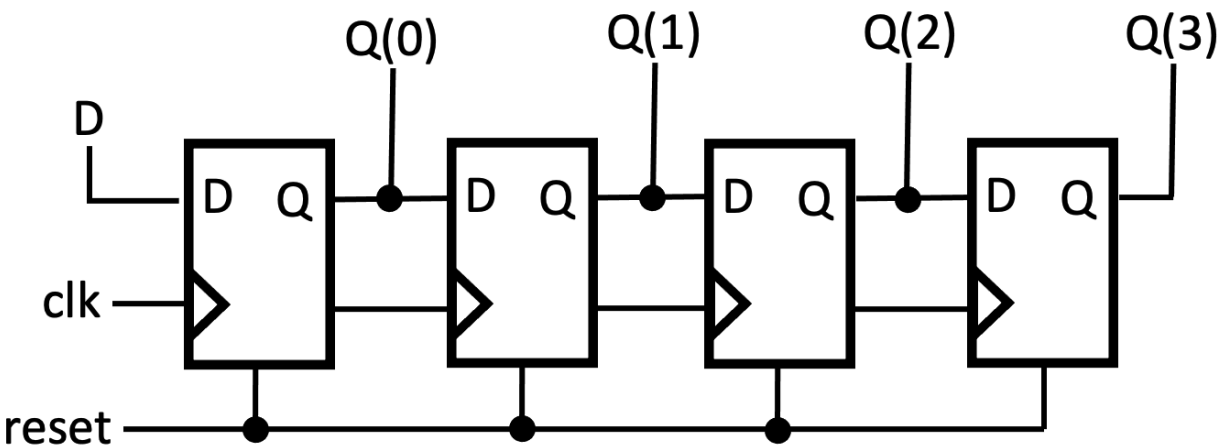
Combinational vs. Sequential Circuit

- **Combinational Circuit**: **no memory**
  - Outputs depend on the *present* inputs only.
  - **Rule**: Use **either** concurrent **or** sequential statements.
- **Sequential Circuit**: **has memory**
  - Outputs depend on *present* inputs and *previous* outputs.
  - **Rule**: **MUST** use sequential statements (i.e., `process`) .

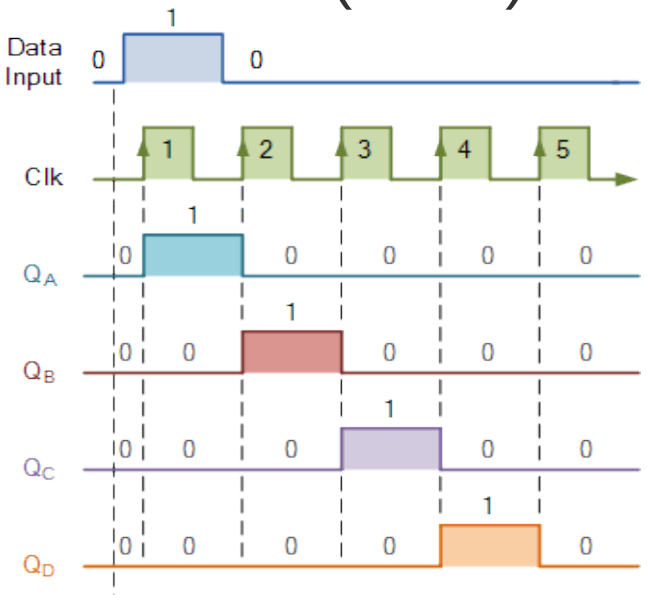


Seq. Circuit Example: Shift Register

- A **register** is a device that can be composed of a group of FFs to store multiple bits of data.
- A **shift register** allows the stored data being moved from one FF to another.
  - There are basically **four** types: ① Serial-In-Serial-Out (SISO), ② Serial-In-Parallel-Out (SIPO), ③ Parallel-In-Serial-Out (PISO), and ④ Parallel-In-Parallel-Out (PIPO).



Ex: Serial-In-Parallel-Out (SIPO) Shift Register



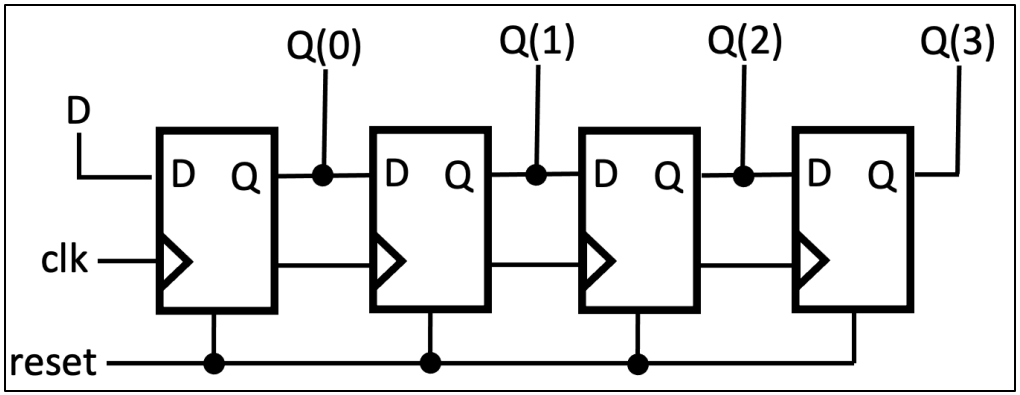
https://www.geeksforgeeks.org/shift-registers-in-digital-logic/ & https://www.electronics-tutorials.ws/sequential/seq_5.html

Example: SIPO

SIPO Shift Register in VHDL

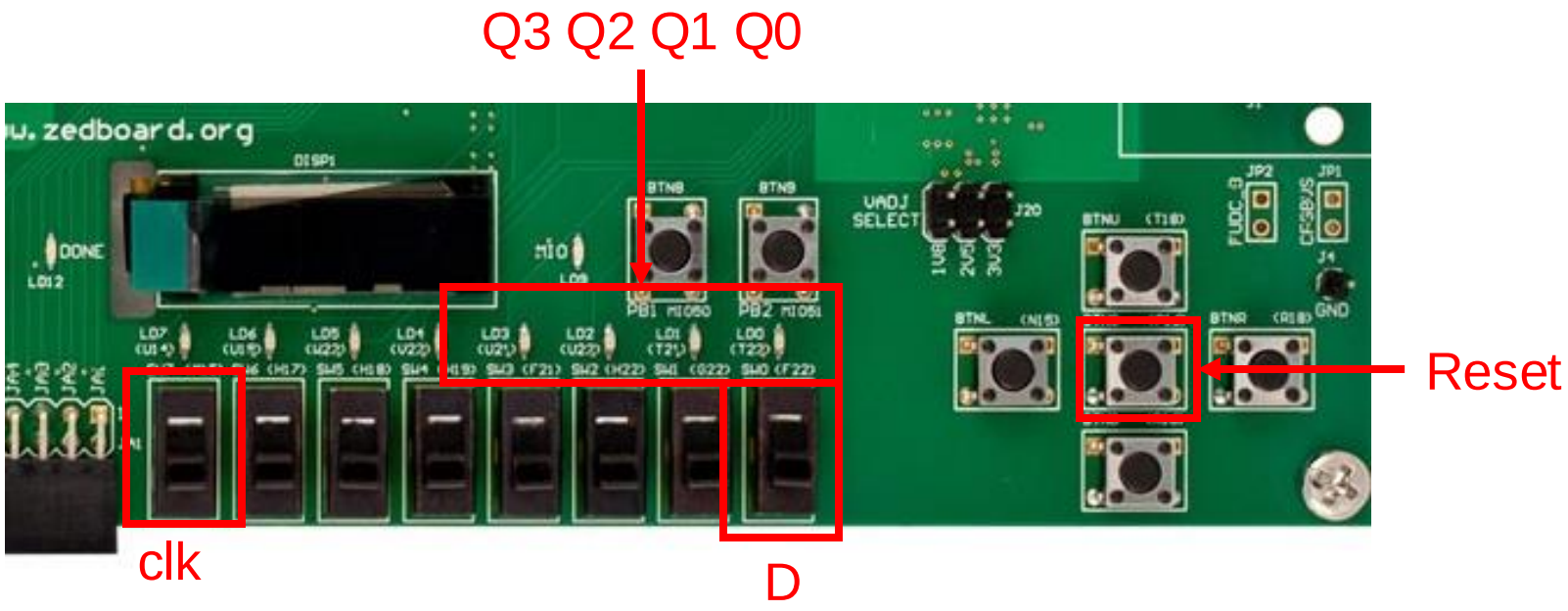
```
entity sipo is
    port (D, clk, reset : in std_logic;
    Q : buffer std_logic_vector(3 downto 0));
end sipo;

architecture arch_sipo of sipo is
    component D_FF is
        port (D, clk, reset : in std_logic;
        Q : buffer std_logic);
    end component;
begin
    DFF0 : D_FF port map(D, clk, reset, Q(0));
    DFF1 : D_FF port map(Q(0), clk, reset, Q(1));
    DFF2 : D_FF port map(Q(1), clk, reset, Q(2));
    DFF3 : D_FF port map(Q(2), clk, reset, Q(3));
end arch_sipo;
```

```
architecture arch_dff of D_FF is
begin
    process (clk, reset)
    begin
        if reset = '1' then
            Q <= '0';
        elsif rising_edge(clk) then
            Q <= D;
        end if;
    end process;
end arch_dff;
```



Constraints (XDC file)

- Bind the I/O ports and physical pins as following:
  - Input: clk=SW7, reset=BTNC, D=SW0
  - Output: Q0~Q3 = LD0~LD3



Notes on Constraints (XDC file)

- Signals in the XDC file is case-sensitive.
  - The case of signals should be same with the entity declaration.
  - Because XDC can be also used with Verilog, which is case-sensitive.
- Vector signals in the XDC file are indexed with [ ] instead of ( ).

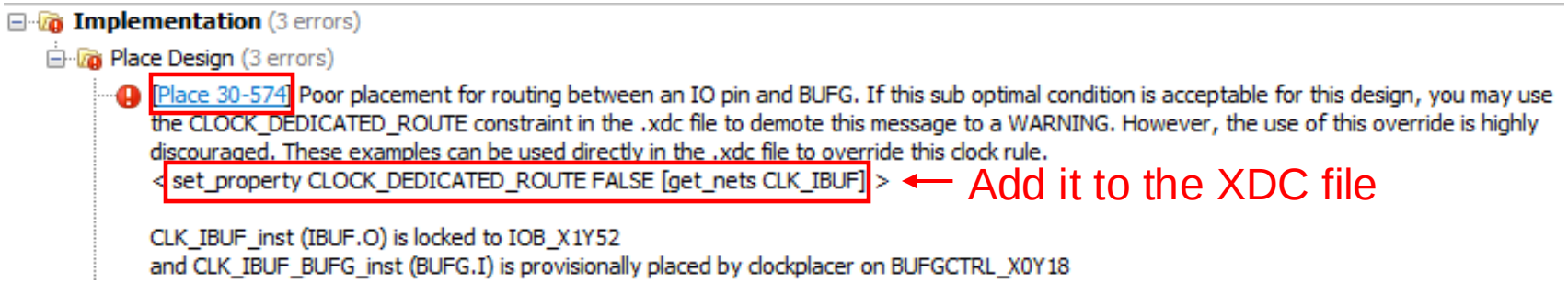
```
set_property PACKAGE_PIN T22 [get_ports Q[0]];  #LD0
set_property PACKAGE_PIN T21 [get_ports Q[1]];  #LD1
set_property PACKAGE_PIN U22 [get_ports Q[2]];  #LD2
set_property PACKAGE_PIN U21 [get_ports Q[3]];  #LD3

set_property PACKAGE_PIN F22 [get_ports D];   #SW0
set_property  PACKAGE_PIN M15 [get_ports clk];  #SW7
set_property  PACKAGE_PIN P16  [get_ports reset];#BTNC

# Electrical specification…
```

Q(0) here will be wrong.

Using ‘RESET’ here will cause an error.

Notes on Constraints (XDC file)

- Add the following statement into XDC file when encountering error [Place 30-574].

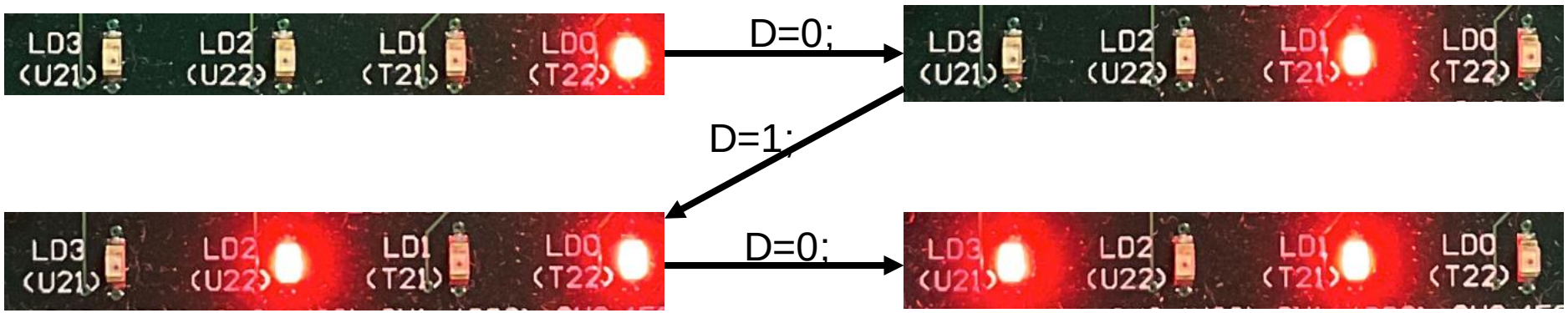


  - The reason is that We bind the signal clk to a switch, which is not a real clock.
  - see how to use a real clock in Lec04 and Lab04.

```
architecture arch_dff of D_FF is
begin
    process (clk, reset)
    begin
        if reset = '1' then
            Q <= '0';
        elsif rising_edge(clk) then
            Q <= D;
        end if;
    end process;
end arch_dff;
```

Verfication on Zedboard

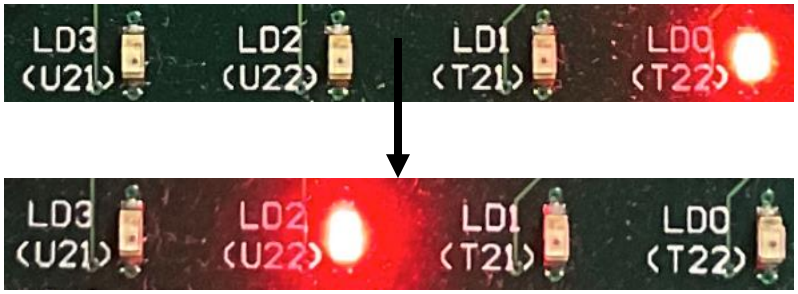
- Generate the Bitstream.
- Program the Zedboard.
- Verify the functions of the SIPO shift register.
  - Test shifting by switching SW7(clk) with BTNC(reset)=0.

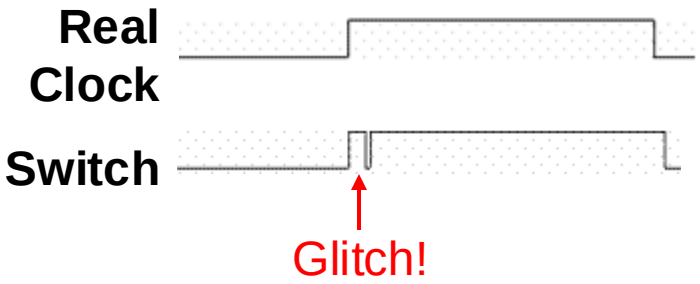


  - Test async reset by turning on BTNC.

Little Problem: Glitch

- If you switch once SW7 but the SIPO shifts twice, you encounter a glitch, it's OK to happen



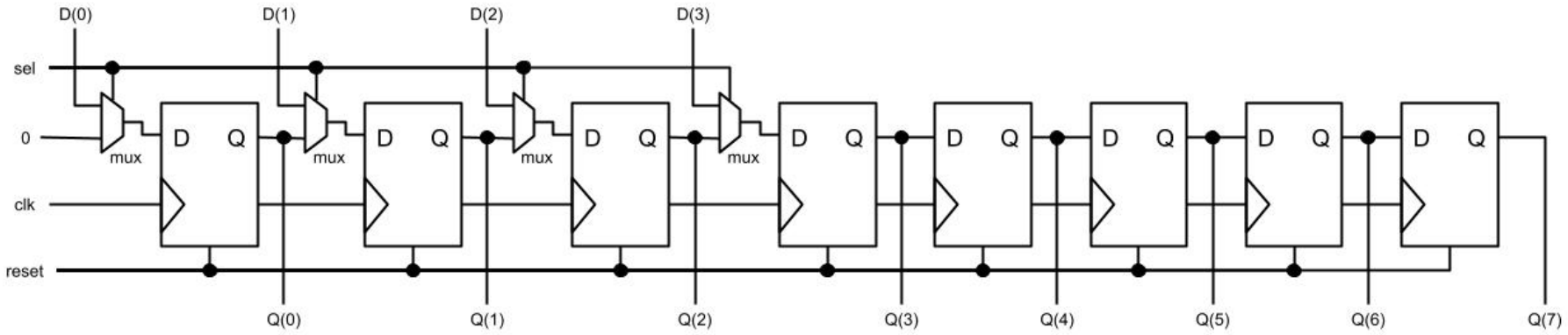


- The glitch is caused by the poor switch contact.
  - Operating once (by human) causes multiple positive edge in the signal.
  - It is also called by switch bouncing.
- Debouncing is out of the lab03 scope.

Lab03 Requirements

Lab03: PIPO Shift Register

- Implement a **parallel-in-parallel-out (PIPO) shift register**, which is composed of **multiplexers (mux)** and **D-Flip-Flops (D-FF)** as follows:
  - It allows **parallel input** (through either external inputs (**Din**) or the shifted outputs from previous D-FFs (**Q**) based on the select signal (**sel**) at rising edges of clock (**clk**));
  - It produces **parallel output** (through external outputs (**Q**)).



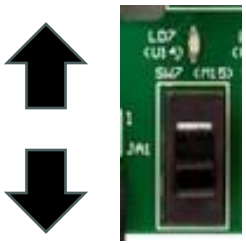
Lab03: PIPO Shift Register

- When **reset** equals to 1:
  - All the **Q(s)** are reset to zero(s) **asynchronously** (immediately).

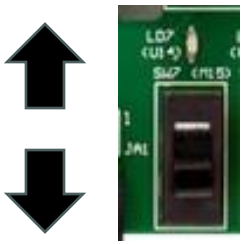
- Otherwise (when **reset** equals to 0), at every **rising edge** of **clk**:
  - When **sel** is 1, the values of **Din** are assigned to **Q** of D-FFs in parallel;
  - Otherwise (when **sel** is 0), the value kept by each D-FF is shifted to the D-FF on its right.
    - A '0' is shifted into to the "leftmost" D-FF;
    - The value kept by the "rightmost" D-FF is shifted out (i.e., discarded) from the PIPO shift register.

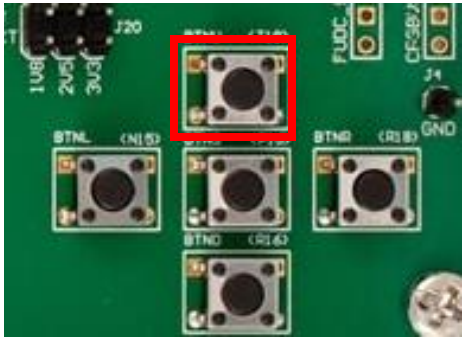
Lab03: PIPO Shift Register

- Instructions:
  - Select the digit using SW 0-3
  - Press BTNU and toggle the SW7 at the same time
    - The digit will display on the LED 0-3
  - Then toggle the SW7 only
    - The digit will shift left
  - Press BTNC to reset the LED 0-7



Shift Left





Display digit on the right

Lab03: Step-by-Step Instructions (1/2)

1. Use the structural design (port map) to implement the PIPO shift register.
   - That is, implement the D-FF first, and then use D-FF to implement the PIPO shift register using port map.
   - There is no special requirement on the implementation method of multiplexers.
2. Create a XDC file with the following assignments:
   - Input: **clk** = SW7, reset = BTNC, **sel** = BTNU, **D0~D3** = SW0~SW3
   - Output: **Q0~Q7** = LD0~LD7
   - Add the following statement to your XDC file:

```
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets clk_IBUF];
```

Lab03: Step-by-Step Instructions (2/2)

4. Generate the bitstream and download it to Zedboard.
5. Toggle SW7 regularly to simulate the **clk** and verify the behavior of your design as follows:
   ① Set D to “0101”, press BTNU and toggle SW7;
   ② Toggle SW7 only until “0101” move to SW4-7
   ③ Set D to “1100”, press BTNU and toggle SW7
   ④ Toggle SW7 several times to shift the digit
   ⑤ Asynchronously **reset** PIPO by pressing BTNC.
   ⑥ Record the above verification as a video for submission.
6. Submit your source code.
   – Submit the zip file of source codes and verification video to Blackboard.
   – Deadline: **14:30 on 12 Feb. 2025**
      ➢ Late submission is NOT acceptable (unless otherwise approved)

香港中文大學
The Chinese University of Hong Kong

Thank You!