|  |
| --- |
|  |
| **RTL\_EXERCISE\_1 BOUND FLASHER** |
|  |
| |  |  | | --- | --- | | Author | LE GIA HUY - 1952717 | | Date | 2024/03/01 | | Version | 1.1 | |
|  |
|  |

|  |
| --- |
|  |

Contents

[1. Interface 2](#_heading=h.1fob9te)

[2. Functional implementation. 3](#_heading=h.3znysh7)

[3. Internal implementation. 5](#_heading=h.2et92p0)

[3.1. Overall. 5](#_heading=h.tyjcwt)

[3.2. State Machine 6](#_heading=h.3dy6vkm)

[4. History 7](#_heading=h.1t3h5sf)

# 1. Interface

|  |
| --- |
| **Module bounder\_flasher(**  input flick,  input clk,  input rst,  output led[15:0]  **)**  LED[15:0]  flick  16  rst  clk |

|  |
| --- |
| Figure 1: the figure of Bound Flasher System |

|  |  |  |  |
| --- | --- | --- | --- |
| Signal | Width | In/Out | Description |
| flick | 1 | In | When the output led turns on gradually, at LED[5] or LED[10] if the flick=1 the output will turn back to the previous state. |
| clk | 1 | In | Operate state’s transition at the rising edge of the clock signal. |
| rst | 1 | In | When reset is high, return to initial state. |
| LED | 16 | Out | Led display depends on the state. |

Table 1: Description of signals in Bound Flasher

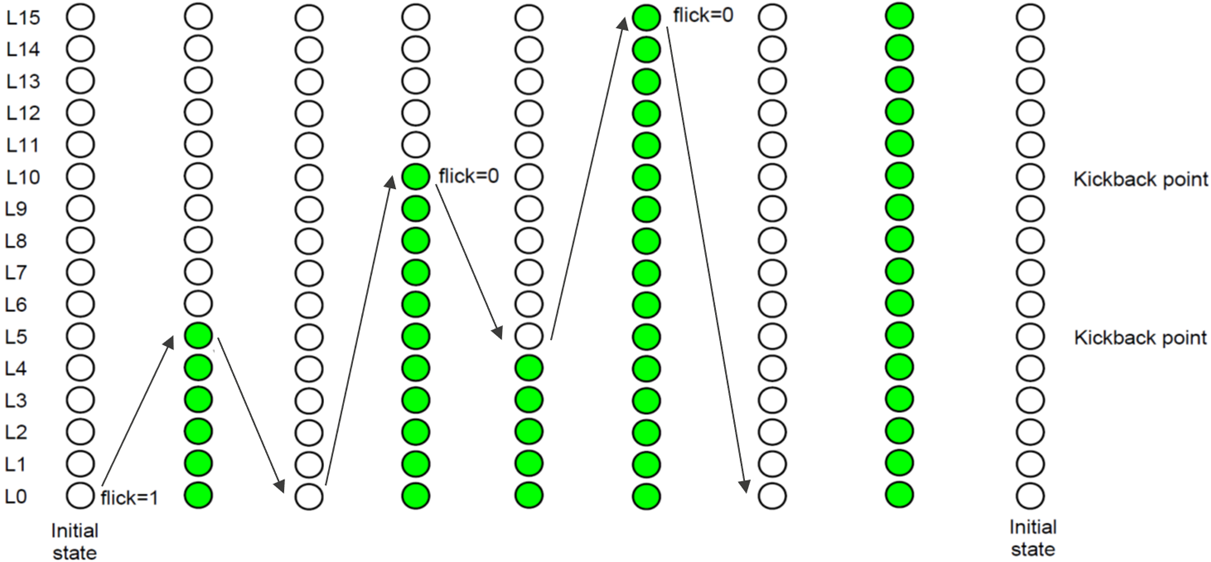
# 2. Functional implementation.

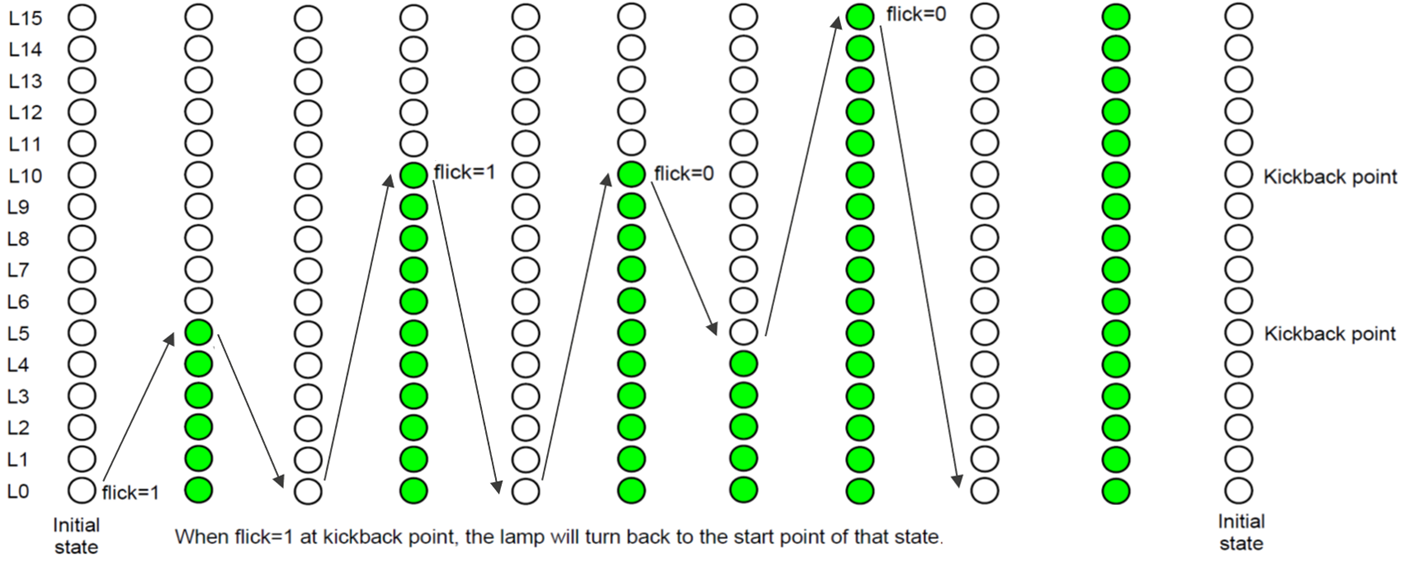
* Implement a 16-bits LEDs system
* System’s Operation base on three input signal
  + Reset
  + Clock
  + Flick
* The system specification
* Clock signal is provided for system inspire of function status. The function operate state’s transition at positive edge of the clock signal.
* Reset signal:
* LOW-ACTIVE Reset = 0: System is restarted to Initial State.
* HIGH-ACTIVE Reset = 1: System is started with initial state.
* Flick signal: special input for controlling state transfer.
* At the initial state, all lamps are OFF. If flick signal is ACTIVE, the flasher start operating:
* The lamps are turned ON gradually from lamp[0]to lamp[5]**.**
* The lamps are turned OFF gradually from lamp[5] **(max)** to lamp[0] **(min)**.
* The lamps are turned ON gradually from lamp[0]to lamp[10].
* The lamps are turned OFF gradually from lamp[10] **(max)** to lamp[5] **(min)**.
* The lamps are turned ON gradually from lamp[5] to lamp[15].
* The lamps are turned OFF gradually from lamp[15] to lamp[0].
* Finally, the lamps are turned ON then OFF simultaneously (blink), return to the initial state.

Additional condition:

* At each kickback point (lamp[5] and lamp[10]), if flick signal is ACTIVE, the lamps will turn OFF gradually again to the **min** lamp of the previous state, then continue operation as above description.

For simplicity, kickback point is considered only when the lamps are turned ON gradually, except the first state.

* Some insulations:
* When flick = 0 at kickback points

* When flick = 1 at kickback points (lamp[10])

# 3. Internal implementation.

## 3.1. Overall.

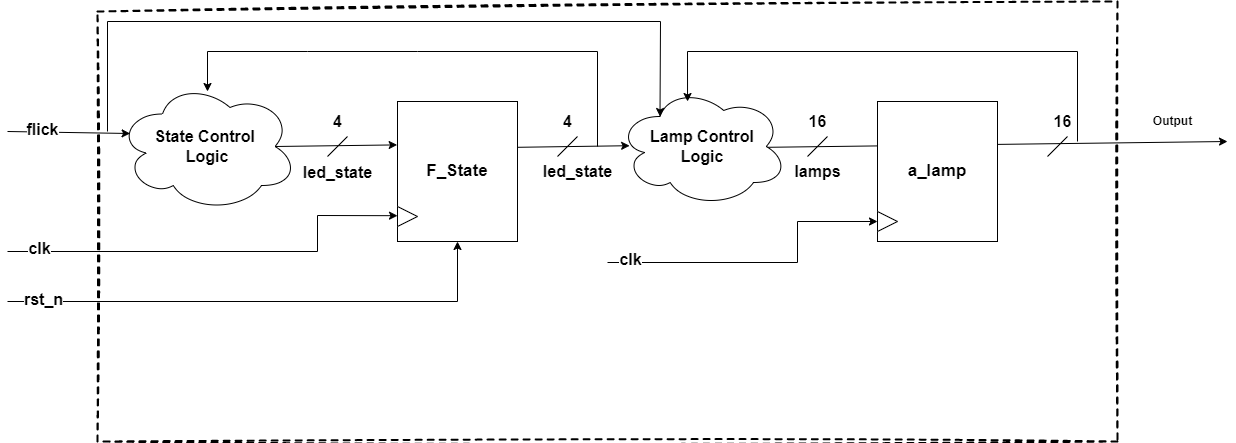


Figure 3.1: Block diagram of Bound Flasher

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Element name | Element description | | State control logic | Combinational logic block used to control the logic of the state, combining with the input, ie, when one state progresses to the next with the right input. | | F\_State | A flip flop used to control the states seen in the state machine in the following part. | | Lamp control logic | Combinational logic block used to control which lights should be turned on or off during a particular state. | | a\_lamp | A flip flop used to control the actual lamps, with one clock cycle corresponding to one light being turned on or off. | |

Table 3.1: Block diagram of Bound Flasher Description

## 3.2. State Machine

Figure 3.2: State Machine of Bound Flasher

|  |  |
| --- | --- |
| Variable Name | Possible value |
| flick | Asynchronous input signal. At kickback points (LED[5] & LED[10]), if flick signal = 1, the LEDs will turn OFF gradually again to the **min** LED of the previous state. |
| LED | 16-bit output, LED[0] is the Least Significant Bit, LED[15] is the Most Significant Bit |

Table 3.2: Variable name of State machine

|  |  |
| --- | --- |
| **State Name** | **Explanation** |
| Initial State | All LEDs are OFF (from LED[0] to LED[15]). If flick signal = 1 then moving to **State ZERO\_TO\_FIVE**. |
| ZERO\_TO\_FIVE | The LEDs are turned ON gradually from LED[0] to LED[5], then moving to **State OFF\_TO\_ZERO**. |
| OFF\_TO\_ZERO | The LEDs are turned OFF gradually from LED[5] to LED[0]. If LED[0] is OFF then moving to **State ZERO\_TO\_TEN.** |
| ZERO\_TO\_TEN | The LEDs are turned ON gradually from LED[0] to LED[10]. When LED[5] is ON, flick signal =1 but LED[6] is OFF, then moving back to **State OFF\_TO\_ZERO.** If (LED[10] is ON && flick signal = 0) then moving to **State OFF\_TO\_FIVE**. Otherwise, if (LED[10] is ON && flick signal = 1) then moving to **State OFF\_TO\_FIVE.** |
| OFF\_TO\_FIVE | The LEDs are turned OFF gradually from LED[10] to LED[5]. If LED[5] is OFF then moving to **State FIVE\_TO\_FIFTEEN**. |
| FIVE\_TO\_FIFTEEN | The LEDs are turned ON gradually from LED[5] to LED[15]. If (LED[5] is ON && LED[6] is OFF) and flick signal = 1 then moving to **State OFF\_TO\_FIVE**. If (LED[15] is ON && flick signal = 0) then moving to **State BLINK** |
| BLINK | The LEDs are turned ON immediatelly from LED[0] to LED[15] then moving to **State BLINK\_RESET**. |
| BLINK\_RESET | The LEDs are turned OFF immediatelly from LED[15] to LED[0] then moving to **State Initial State**. |

Table 3.3: State name of State machine

# 4. History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Author | Modified part | Description |
| 2020/02/22 | Gia Huy | Design & Textbench | Complete the source code |
| 2022/03/01 | Gia Huy | Word, excel | Complete the report |