|  |
| --- |
|  |
| **RTL EXERCISE 1** |
|  |
| |  |  | | --- | --- | | Supporter | My Nguyen | | Author | An Tran | | Date | 2017/03/27 | | Version | 1.4 | |
|  |
|  |

|  |
| --- |
|  |

Contents

[1. **Interface** 2](#_Toc476912617)

[2. **Functional implementation.** 3](#_Toc476912618)

[2.1 **Function 1** 3](#_Toc476912619)

[2.2 **Function 2** 3](#_Toc476912620)

[3. **Internal implementation** 4](#_Toc476912621)

[3.1 **Overall** 4](#_Toc476912622)

[3.2 **State control logic module.** 5](#_Toc476912623)

[3.2.1 **Overall.** 5](#_Toc476912624)

[3.2.2 **Function state control logic.** 6](#_Toc476912625)

[3.3 **Lamp control logic module.** 12](#_Toc476912627)

[3.3.1 **Overall.** 12](#_Toc476912628)

[3.3.2 **Function lamp control logic** 12](#_Toc476912629)

[4. **History** 13](#_Toc476912630)

1. **Interface**

input output

Module lfu\_finder

clk

rst buf\_num\_replc[1:0]

new\_buf\_req 2

ref\_buf\_numbr[1:0]

Figure 1: Interface of LFU

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Signal** | **Width** | **In/Out** | **Mode** | **Description** |
| clk | 1 | In | Rising edge | The clock signal |
| rst | 1 | In | Falling edge | The reset signal |
| new\_buf\_req | 1 | In | Data | 0 if no need to replace a buffer. 1 if a buffer must be replaced |
| ref\_buf\_numbr[1:0] | 2 | In | Data | The number of the buffer accessed. This signal is valid only when new\_buf\_req = 0. |
| buf\_num\_replc[1:0] | 2 | Out | Data | The number of the buffer to be replaced. This signal is valid only during a clock cycle right after new\_buf\_req is set to 1. |

Note 1: The buffer defined by this signal must be treated to be accessed 1 time.

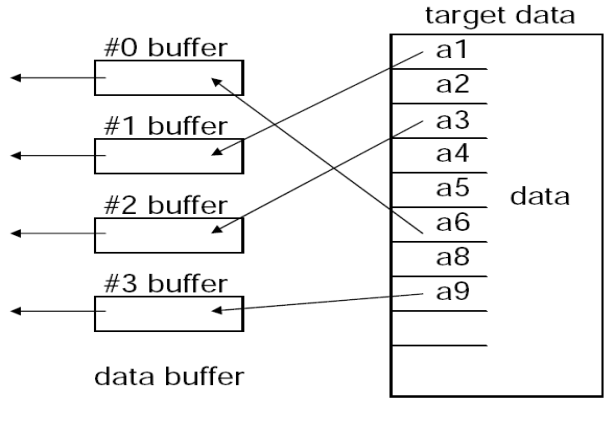
Note 2: After reset, the access time of buffers shall be reset to 1 and the buf\_num\_replc will be reset to 2’b00.

Note 3: In actual usage, the LFU algorithm is only applicable after all entries are filled up.

1. **SYSTEM DESCRIPTION.**

Create a module that finds out the Least Frequently Used entry in the four entries specified below.

The target data are accessible only through data buffers named: #0 buffer, #1 buffer, #2 buffer, and #3 buffer. The following figure shows that data a1 using #1 buffer, a3 using #2 buffer, a6 using #0 buffer, and a9 using #3 buffer. Now, if data a2 is requested, one of the buffers must be replaced for a2. To select the buffer to be replaced for the new request, apply LFU strategy.



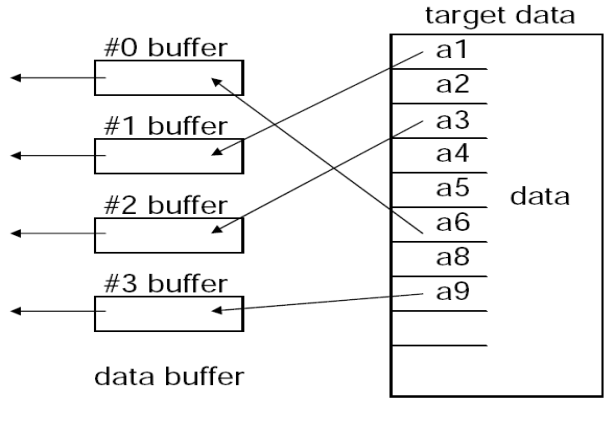


Figure 2. Basic operation of buffers

* Objective: Suppose the target data are randomly accessed, create a module which can output the buffer number to be replaced based on LFU algorithm.

The following figure shows the LFU management rule. If ref\_buf\_numbr =2 means that data in cache #2 buffer is referenced. If buf\_num\_replc = 1 means cache #1 buffer shall be replaced (saved back to external memory and shall be filled in with new data).

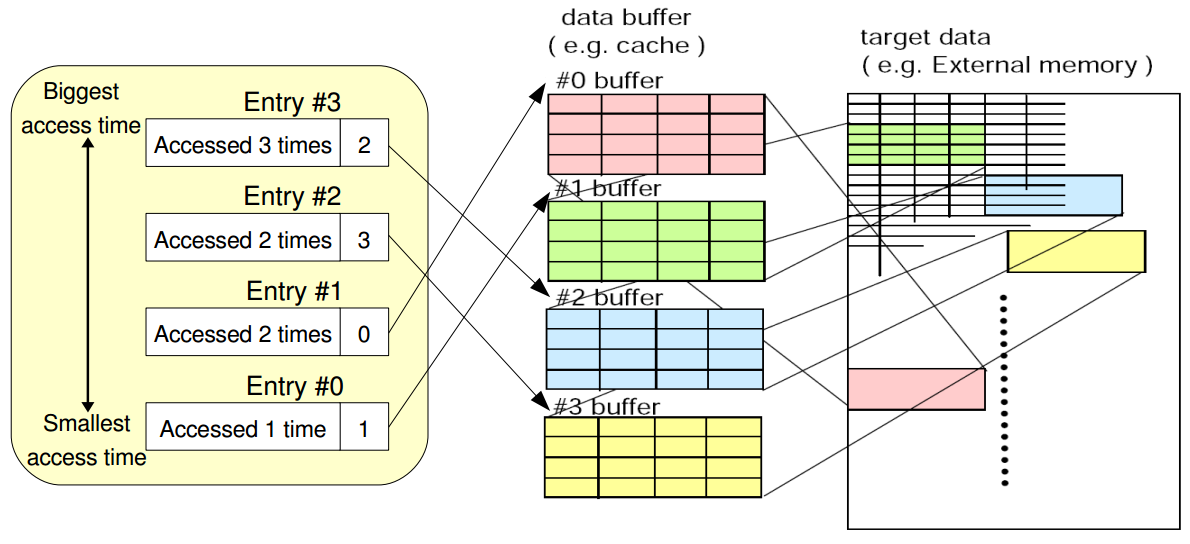


Figure 3: LFU management table

In case where two or more buffers have the same minimum access time, the default replacement order is #0 -> #1 -> #2 -> #3. That means the buffer with lowest ID is replaced first. The access time of newly filled buffer is 1.

1. **BLOCK DIAGRAM**

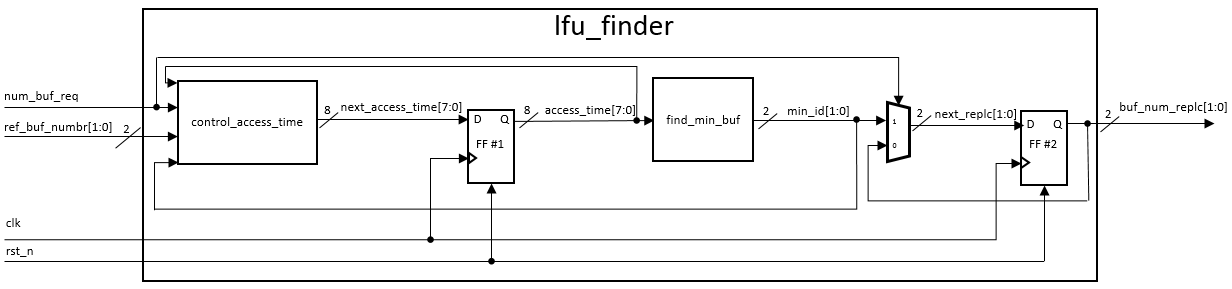


Figure 4: Block diagram of lfu\_fi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Port** | **I/O** | **Mode** | **Description** |
| 1 | clk | I | Rising edge | Clock signal |
| 2 | rst\_n | I | Active low | Reset signal |
| 3 | new\_buf\_req | I | Control signal | 0 if no need to replace a buffer. 1 if a buffer must be replaced. |
| 4 | ref\_buf\_numbr[1:0] | I | Data | The number of the buffer accessed. This signal is valid only when new\_buf\_req = 0. |
| 5 | buf\_num\_replc[1:0] | O | Data | The number of the buffer to be replaced. This signal is valid only during a clock cycle right after new\_buf\_req is set to 1. |
| 6 | next\_access\_time[7:0] | - | Data | The next cycle access time. |
| 7 | access\_time[7:0] | - | Data | The current access time. |
| 8 | min\_id[1:0] | - | Data | The id of the buffer which has the minimum access time. |
| 9 | next\_replc[1:0] | - | Data | The next cycle buf\_num\_replc. |

* 1. **State control logic module.**
     1. **Overall.**

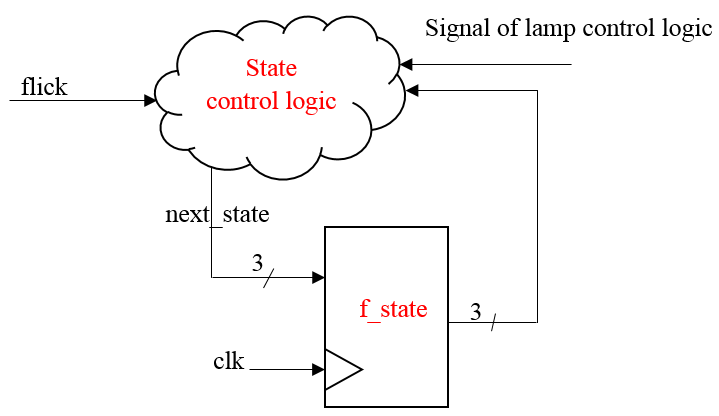
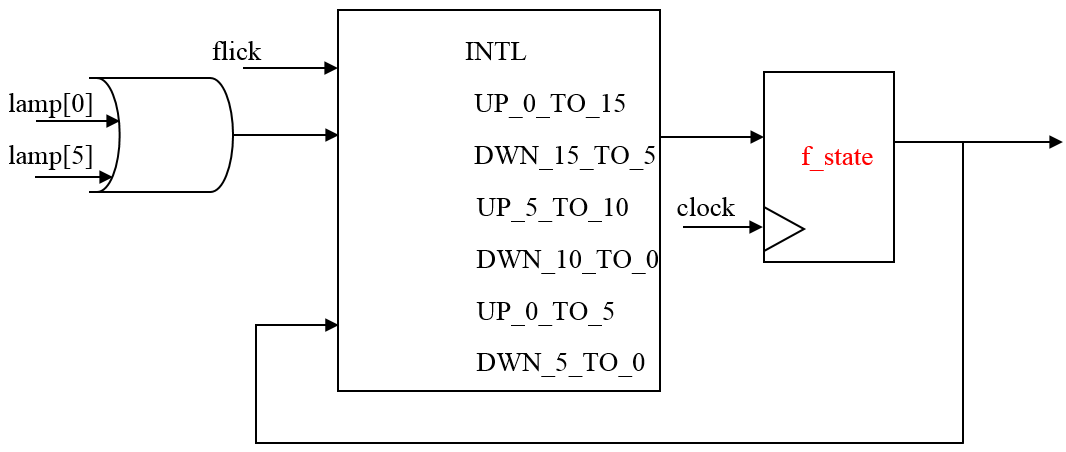


Figure 3: The block diagram of state control module

The input receive the signal from flick and sent the signal go to state control logic module to control translation of signal. If the condition is true, module transfer signal control the circuit to next state and transfer it through flip – flop circuit and when condition of clock is true the signal transfer two ways. The first way transfer signal change to next state to the lamp control logical. In the other hand, signal turn back function 1 to compare with condition.

* + 1. **Function state control logic.**

 **How to implement.**

|  |
| --- |
|  |
| Figure 4: The logic circuit of state control logic module |

In overall, when the flick change to active signal and receive to signal of kickback point with the current state will transition to de-mux 3 to 8 to decide the next state of circuit will be change.

* 1. **Lamp control logic module.**
     1. **Overall.**

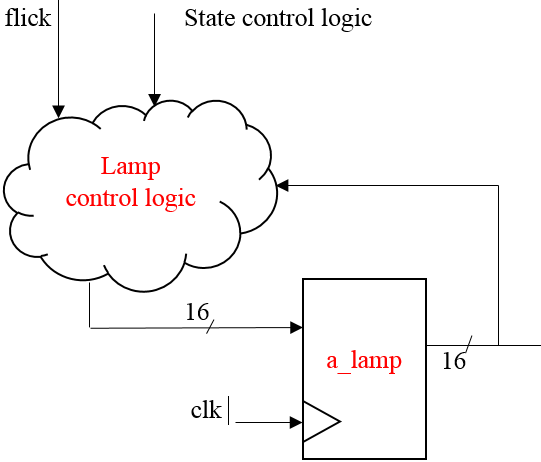


Figure 5: The block diagram of lamp control logic module

The lamp control logic module receive to two ways signal. One signal receive through the signal of flick from input and one signal receive from state control logic module. Two signal help this module decide turn on or turn off and how many lamps work. After control the lamps, the signal to control it to return to module control state to compare with condition and decide the next state for circuit will change to.

* + 1. **Function lamp control logic**

**How to implement**

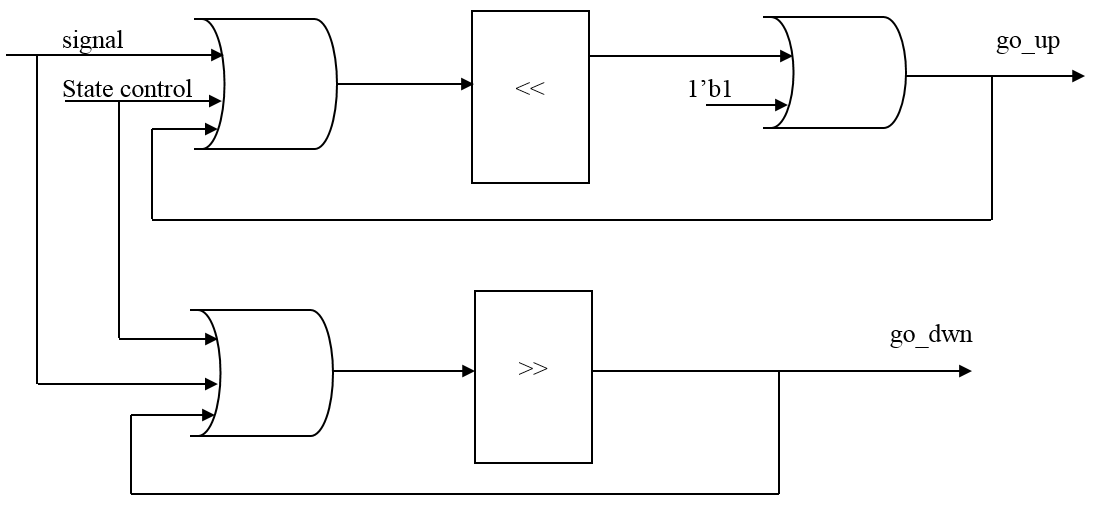


Figure 6: The logic circuit of lamp control logic module

The signal and state control logic to control the state of lamp to go up or go down. If you want control the lamp turn on gradually the signal will shift left and or with 1 bit 1. In the other hand, we turn off gradually by put the signal to shift right. After that, the go up and go down transition to state control for check condition and feedback it to this module.

* 1. **The states transition table**

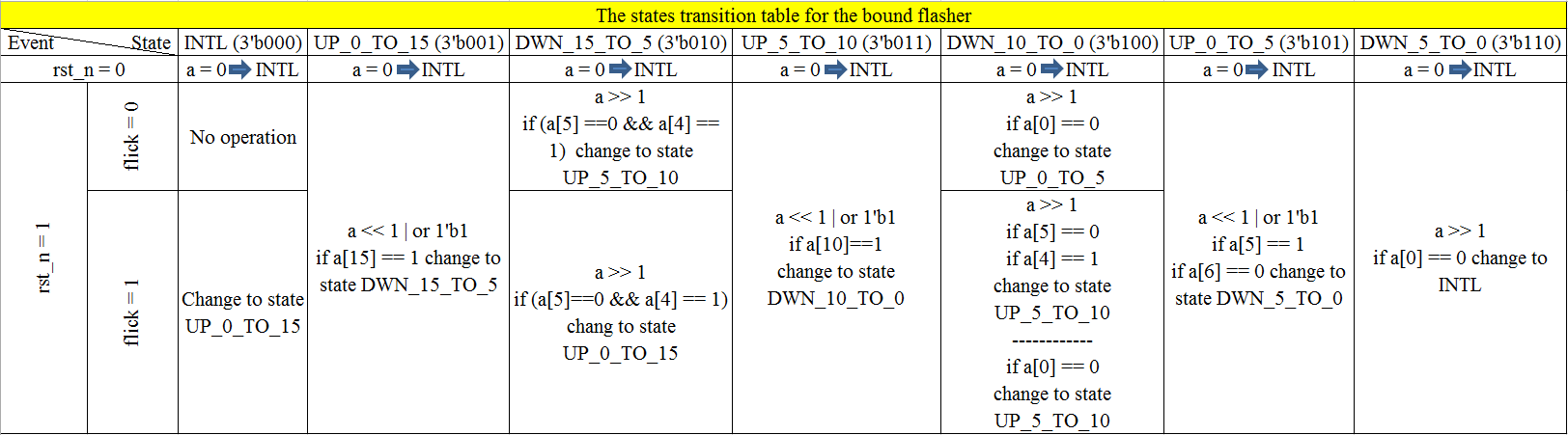


Figure 7: The states transition of te bound flasher

|  |  |
| --- | --- |
| State name | Description |
| INTL | The initial state |
| UP\_0\_TO\_15 | Turn on the lamp gradually from 0 to 15 |
| DWN\_15\_TO\_5 | Turn off the lamp gradually from 15 to 5 |
| UP\_5\_TO\_10 | Turn on the lamp gradually from 5 to 10 |
| DWN\_10\_TO\_0 | Turn off the lamp gradually from 10 to 0 |
| UP\_0\_TO\_5 | Turn on the lamp gradually from 0 to 5 |
| DWN\_5\_TO\_0 | Turn off the lamp gradually from 5 to 0 |

Table 1: The state of bound flasher

|  |  |  |
| --- | --- | --- |
| Current state | Next state | Description |
| INTL | INTL | When the circuit open, the state is initial with every lamps turn off. If the value of flick is 0, the state don’t change to another state and wait the flick is ACTIVE. |
| UP\_0\_TO\_15 | When flick is active. The state change to next state for turn on gradually from 0 to 15. |
| UP\_0\_TO\_15 | DWN\_15\_TO\_5 | The lamps in current state turn on gradually when the fifteenth lamp is active and the state change to next state. |
| DWN\_15\_TO\_5 | UP\_0\_TO\_15 | The lamps in current state turn off gradually when the fifth lamp is passive and fourth lamp is active with flick is active, the next state is return the state before. |
| UP\_5\_TO\_10 | The lamps in current state turn off gradually when the fifth lamp is passive and fourth lamp is active with flick is passive, the next state is turn on gradually from the fifth lamp to the tenth lamp. |
| UP\_5\_TO\_10 | DWN\_10\_TO\_0 | The lamps in current state turn on gradually when the tenth lamp is active and the state change to next state. |
| DWN\_10\_TO\_0 | UP\_5\_TO\_10 | The lamps in current state turn off gradually when flick is active with the fifth lamp is passive and the fourth lamp is active or the lamp turn off to the last lamp, the state is return the state before. |
| UP\_0\_TO\_5 | The lamps in current state turn off gradually when the last lamp is passive, the next state is turn on gradually from the last lamp to the fifth lamp. |
| UP\_0\_TO\_5 | DWN\_5\_TO\_0 | The lamps in current state turn on gradually when the fifth lamp is active and the sixth lamp is passive, the state change to next state is turn off gradually from the fifth lamp to the last lamp. |
| DWN\_5\_TO\_0 | INTL | The final state don’t care kickback point and turn off gradually when the last lamp is passive, the state is return to the initial state. |

Table 2: The transition of state

1. **History**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Author | Modified part | Description |
| 2017/03/22 | An Tran | - | New creation |
| 2017/03/23 | An Tran | State control logic | Add logic circuit of state control logic module |
| 2017/03/24 | An Tran | Lamp control logic | Add logic circuit of lamp control logic module |
| 2017/03/27 | An Tran | State transition table | Make the transition of state |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |