**RTL EXERCISE #2**

**LEAST FREQUENTLY USED ALGORITHM**

**Supporter: Nguyen Dinh Minh Nhat**

**Trainee: Tran Hong An**

**ID: 1792**

# REVISIONS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Version | Modification | | | Author | Review by |
| Page | Details | Reason |
| 00.01 14/10/2016 | ― | New creation | ― | An Tran |  |
| 00.02 23/10/2016 | 2, 3, 6-16 | Edit the revisions and content. Add the Block diagrams and detail design. Add the timing chart and test cases. | ― | An Tran |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# CONTENTS

[REVISIONS 2](#_Toc464990853)

[CONTENTS 3](#_Toc464990854)

[LIST OF FIGURES 3](#_Toc464990855)

[LIST OF TABLES 3](#_Toc464990856)

[1. SYSTEM DESCRIPTION 4](#_Toc464990857)

[2. ARCHITECHTURE 6](#_Toc464990858)

[3. BLOCK DIAGRAM 7](#_Toc464990859)

[4. DETAIL DESIGN 8](#_Toc464990860)

[4.1. find\_min\_buf 8](#_Toc464990861)

[4.2. control\_access\_time 9](#_Toc464990862)

[5. TIMING CHART DIAGRAM 11](#_Toc464990863)

[6. TEST CASES 15](#_Toc464990864)

# LIST OF FIGURES

[Figure 1. Basic operation of buffers 4](#_Toc464991012)

[Figure 2. LFU management table 5](#_Toc464991013)

[Figure 3. Block diagram of architecture 6](#_Toc464991014)

[Figure 4. Block diagram of lfu\_finder module 7](#_Toc464991015)

[Figure 5. Block diagram of find\_min\_buf block 8](#_Toc464991016)

[Figure 6. Block diagram of control\_access\_time block 9](#_Toc464991017)

[Figure 7. Timing chart when rst\_n is ACTIVE 11](#_Toc464991018)

[Figure 8. Timing chart when rst\_n and num\_buf\_req is off 12](#_Toc464991019)

[Figure 9. Timing chart when rst\_n is off and the new\_buf\_req is active 13](#_Toc464991020)

[Figure 10. Timing chart when a buffer's access time = 2'b11 and all buffers’ access time = 2'b11 14](#_Toc464991021)

# LIST OF TABLES

[Table 1. IF table of architecture 6](#_Toc464991022)

[Table 1. IF Table of top module 7](#_Toc464991023)

[Table 3. IF Table of the find\_min\_buf block 8](#_Toc464991024)

[Table 2. IF Table of the control\_access\_time block 9](#_Toc464991025)

# 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.

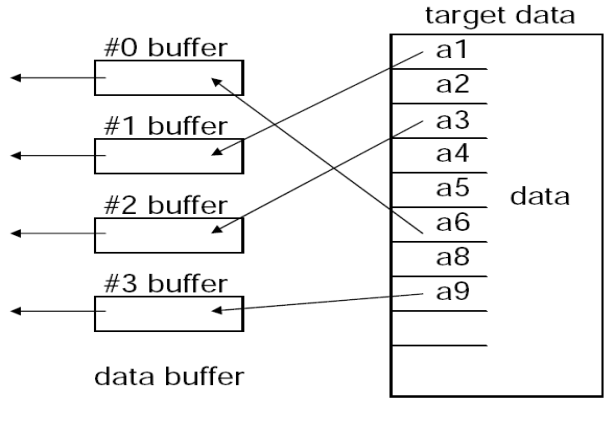


Figure 1. Basic operation of buffers

* 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).

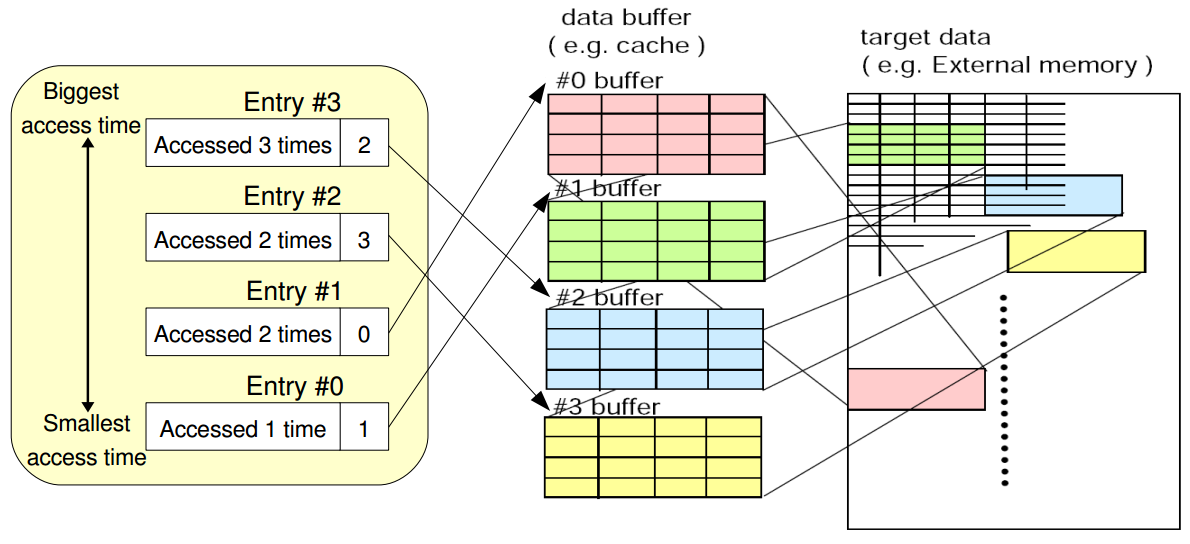


Figure 2. LFU management table

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.

# ARCHITECHTURE

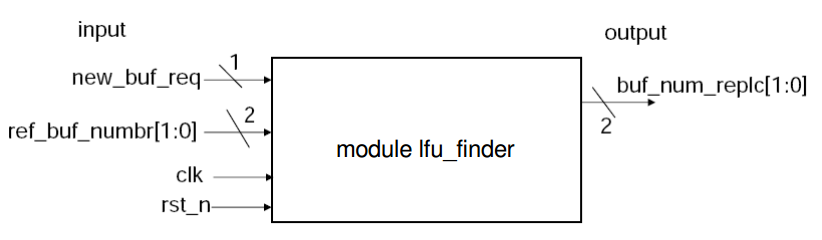


Figure 3. Block diagram of architecture

Table 1. IF table of architecture

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Port** | **I/O** | **Mode** | **Description** |
| 1 | clk | I | Rising edge | Clock signal |
| 2 | rst\_n | I | Falling edge | Reset signal |
| 3 | new\_buf\_req | I | Data | 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. |

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.

# BLOCK DIAGRAM

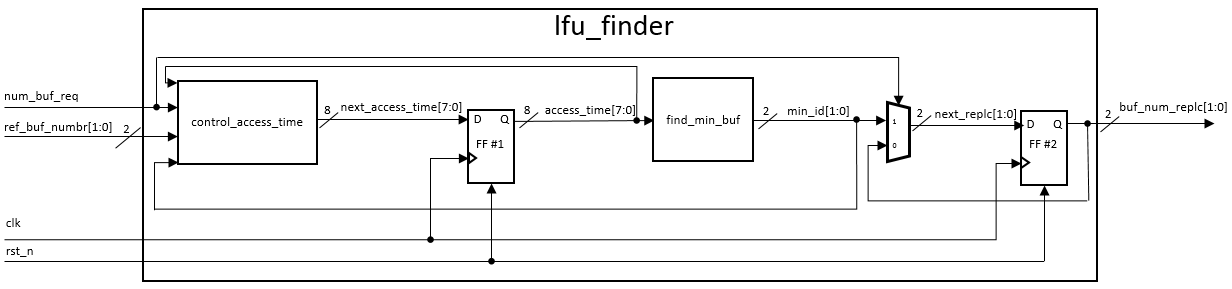


Figure 4. Block diagram of lfu\_finder module

Table 2. IF Table of top module

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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. |

Description

* The control\_access\_time block uses 4 inputs “min\_buf\_req”, “ref\_buf\_numbr”, “access\_time” and “min\_id” to provide the next\_access\_time.
* The find\_min\_buf block is used to provide the buffer with minimum access time or the lowest order buffer if there are 2 or more buffer having the same number of access time.
* The mux is used to select the “min\_id” buffer for the “buf\_num\_replc” when there is a request for new buffer or keep the same value.
* The FF #1 is used to return the access\_time value to calculate for the next\_access\_time. When reset is active, the access\_time = 8’b01010101.
* The FF #2 is used to return the current buf\_num\_replc to keep this value for the next cycle if the new\_buf\_req is not 0.

# DETAIL DESIGN

* 1. **find\_min\_buf**
  2. **control\_access\_time**

## 4.1. find\_min\_buf

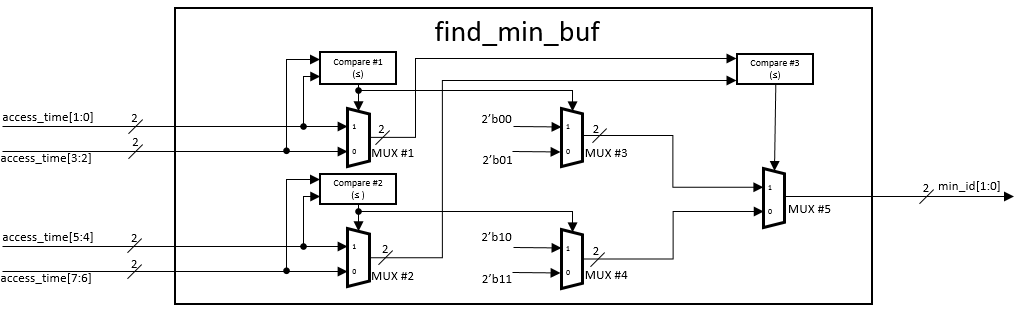


Figure 5. Block diagram of find\_min\_buf block

Table 3. IF Table of the find\_min\_buf block

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Port** | **I/O** | **Mode** | **Description** |
| 1 | access\_time[7:0] | I | Data | Each 2 bit represent the number of access time for each buffer. access\_time[1:0] for buffer #0, access\_time[3:2] for buffer #1; access\_time[5:4] for buffer #2 and access\_time[7:6] for buffer #3. |
| 2 | min\_id[1:0] | O | Data | The id of the buffer which has the minimum access time. |

Description:

* The Compare #1 is used to find the smaller access time between buffer #0 and buffer #1. Its output is used to control MUX #1 and MUX #3. The same for the Compare #2, MUX #2 and MUX #4.
* The Compare #3 is used to find the smaller access time between MUX #1 and MUX #2. The output is used to control the MUX #5.
* The output MUX #1 and MUX #2 is the smaller access time between the buffer #0 and #1, #2 and #3 respectively.
* The output MUX #3 and MUX #4 is the id of the buffer having smaller access time between the buffer #0 and #1, #2 and #3 respectively.
* The output MUX #5 is the id of the buffer having minimum access time.

## 4.2. control\_access\_time

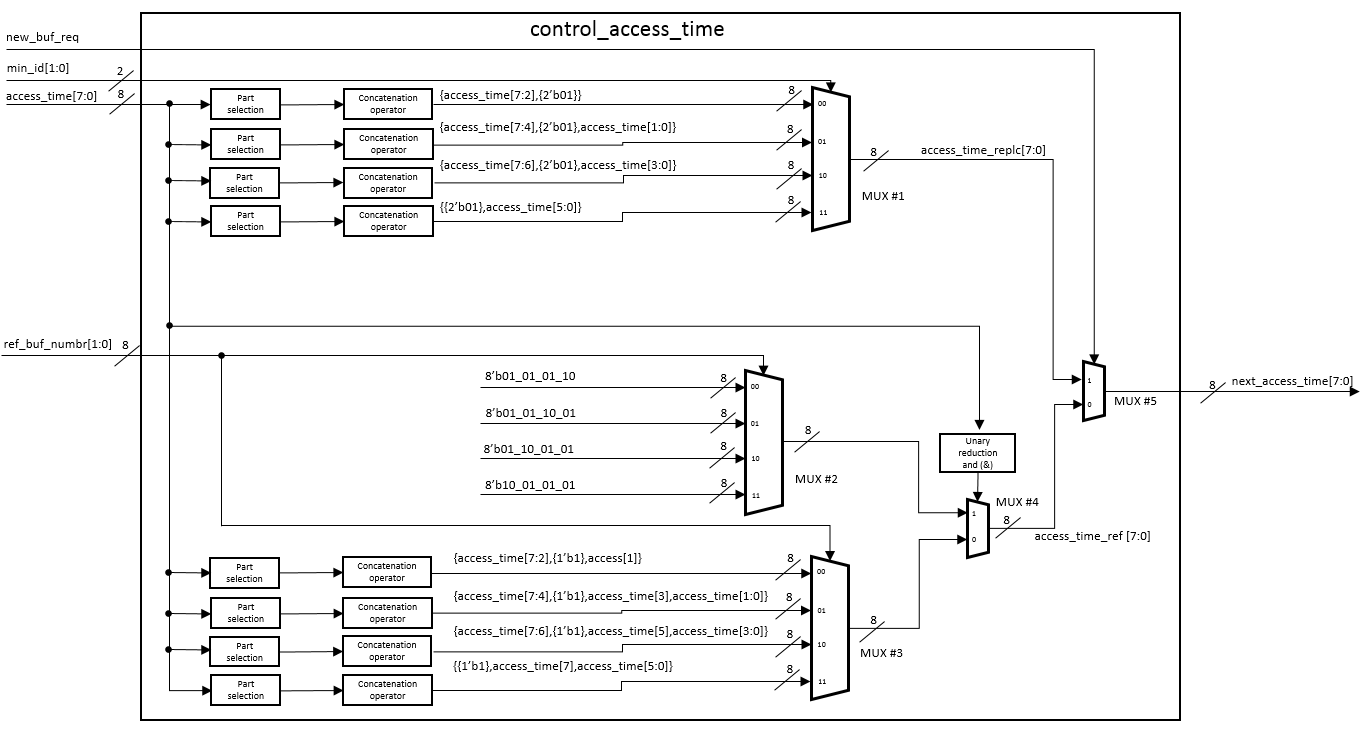


Figure 6. Block diagram of control\_access\_time block

Table 4. IF Table of the control\_access\_time block

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Port** | **I/O** | **Mode** | **Description** |
| 1 | new\_buf\_req | I | Control signal | 0 if no need to replace a buffer. 1 if a buffer must be replaced. |
| 2 | min\_id[1:0] | I | Data | The id of the buffer which has the minimum access time. |
| 3 | access\_time[7:0] | I | Data | Each 2 bit represent the number of access time for each buffer. access\_time[1:0] for buffer #0, access\_time[3:2] for buffer #1; access\_time[5:4] for buffer #2 and access\_time[7:6] for buffer #3. |
| 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 | next\_access\_time[7:0] | O | Data | The next cycle value of access\_time |
| 6 | access\_time\_replc[7:0] | - | Data | Access time when a buffer is requested |
| 7 | access\_time\_ref[7:0] | - | Data | Access time when a buffer is referenced and new\_buf\_req = 0. |

Description

* The Mux #1‘s output is the access\_time in case of the new\_buf\_req = 1. The requested buffer’s access time is set to 2’b01 while the others do not change.
* The Mux #2’s output is the access\_time in case of the access time of 4 buffers is all 2’b11 and a buffer is referenced.
* The Mux #3’s output is the access\_time in case of the access time of 4 buffers is not all 2’b11 and a buffer is referenced.
* The “Unary reduction and (&)” is used to AND 8 bit of the access\_time. If the access\_time = 8’hff, the output will be 1’b1 or 1’b0 for other cases. The output is used to choose between access\_time from Mux #2 and Mux #3.
* The Mux #5 is used to choose between the access time when the new\_buf\_req is 1 or 0.

# TIMING CHART DIAGRAM

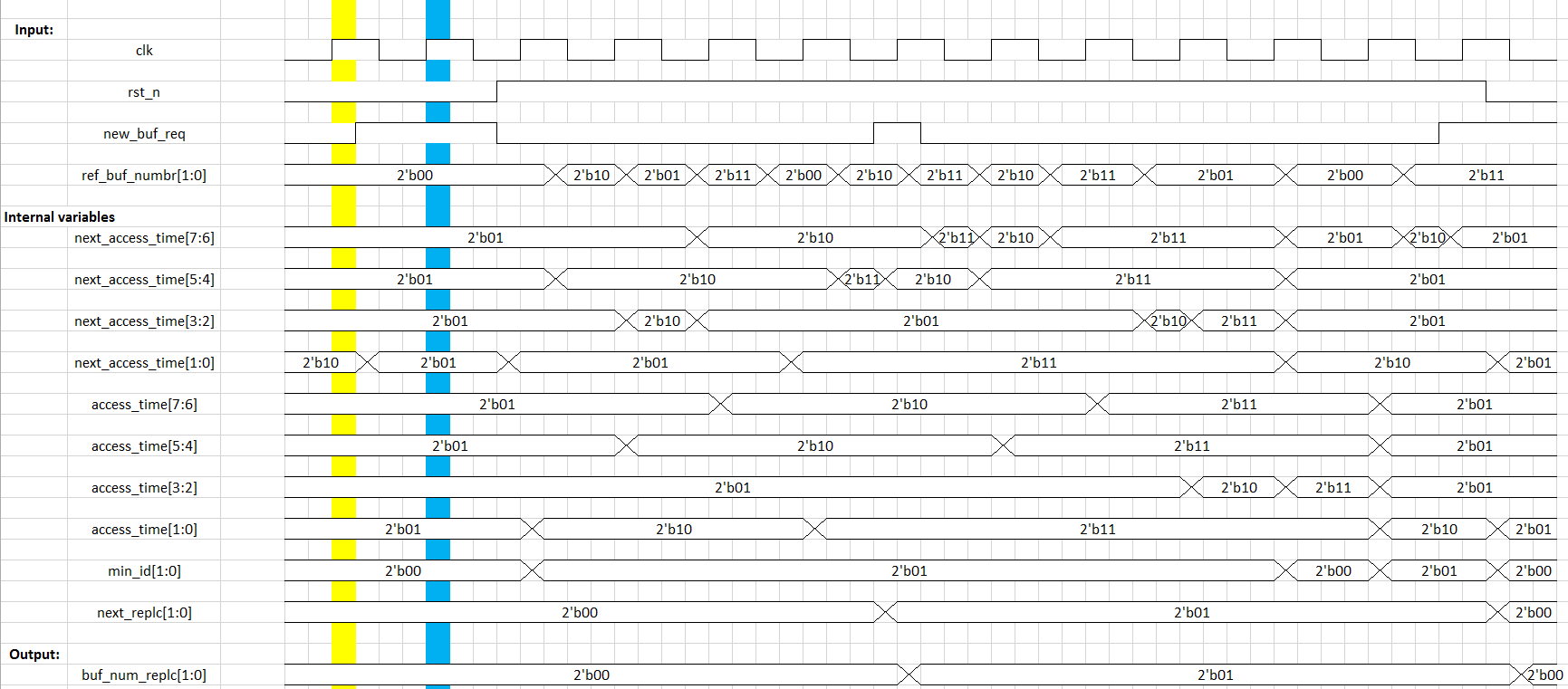


Figure 7. Timing chart when rst\_n is ACTIVE

Yellow line:

* rst\_n is ACTIVE ( = 0) 🡺Initial values: access\_time = 8’b01010101, buf\_num\_replc = 2’b00.
* next\_access\_time = 2’b01010110 but the acces\_time won’t change the value (rst\_n is active).

Blue line:

* rst\_n is ACTIVE ( = 0) 🡺 Initial values: access\_time = 8’b01010101, buf\_num\_replc = 2’b00.
* buf\_num\_req = 1 🡺 buf\_num\_replc = 2’b00 (not change) and next\_access\_time = 8’h55.

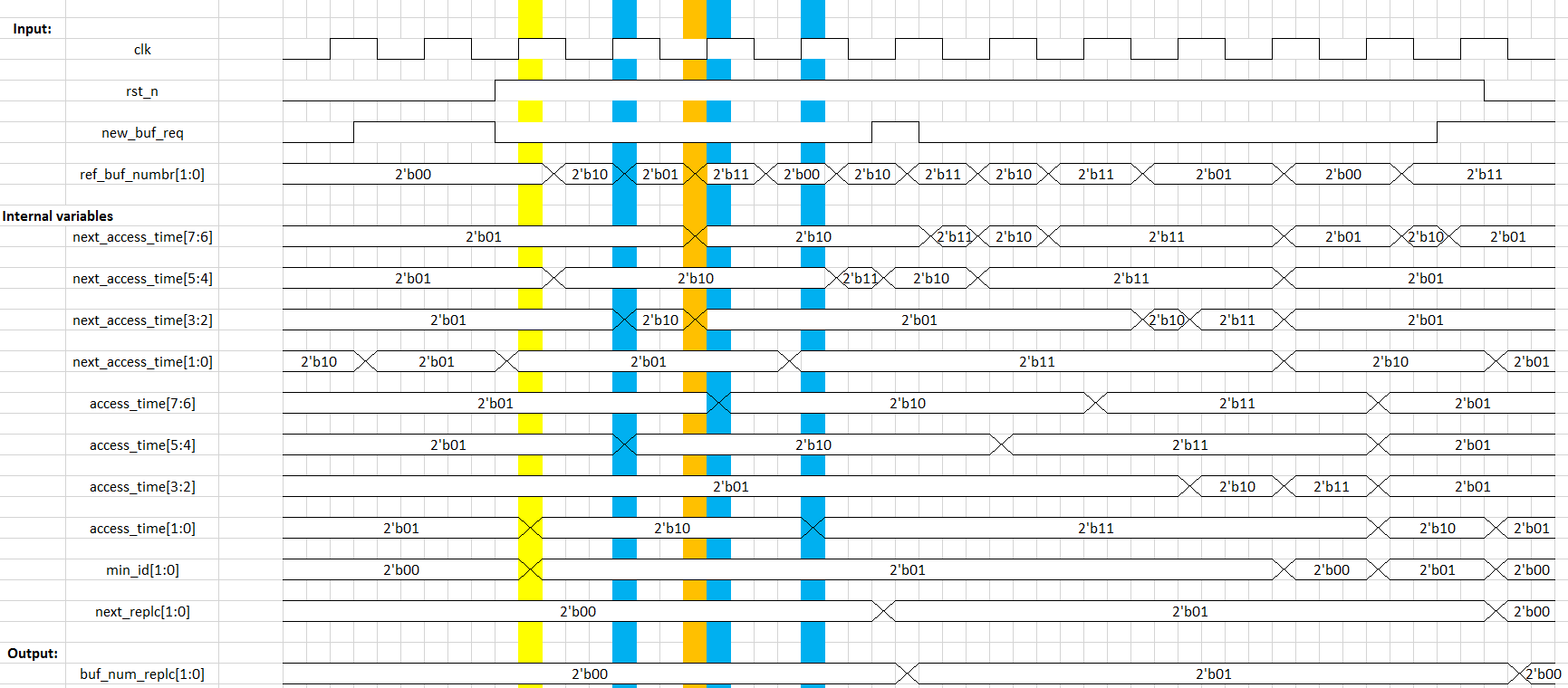


Figure 8. Timing chart when rst\_n and num\_buf\_req is off

Yellow line:

* next\_accesss\_time increased based on the ref\_buf\_numbr.
  + ref\_buf\_numbr = 2’b00 🡺next\_access\_time[1:0] = 2’b10.
* access\_time get the value of next\_access\_time (8’b01010110).
* The min\_id changed according to the change of access\_time (from 2’b00 to 2’b01)

Blue lines:

* next\_access\_time changed according to the ref\_buf\_numbr.
* access\_time get the values of next\_access\_time

Orange line:

* ref\_buf\_numbr change value after the clock rise and before the next clock rise.
* The next\_access\_time[3:2] change to 2’b01 (from 2’b10) since the access\_time[3:2] = 2’b01.
* Only the change of next\_access\_time[7:6] effect the access\_time

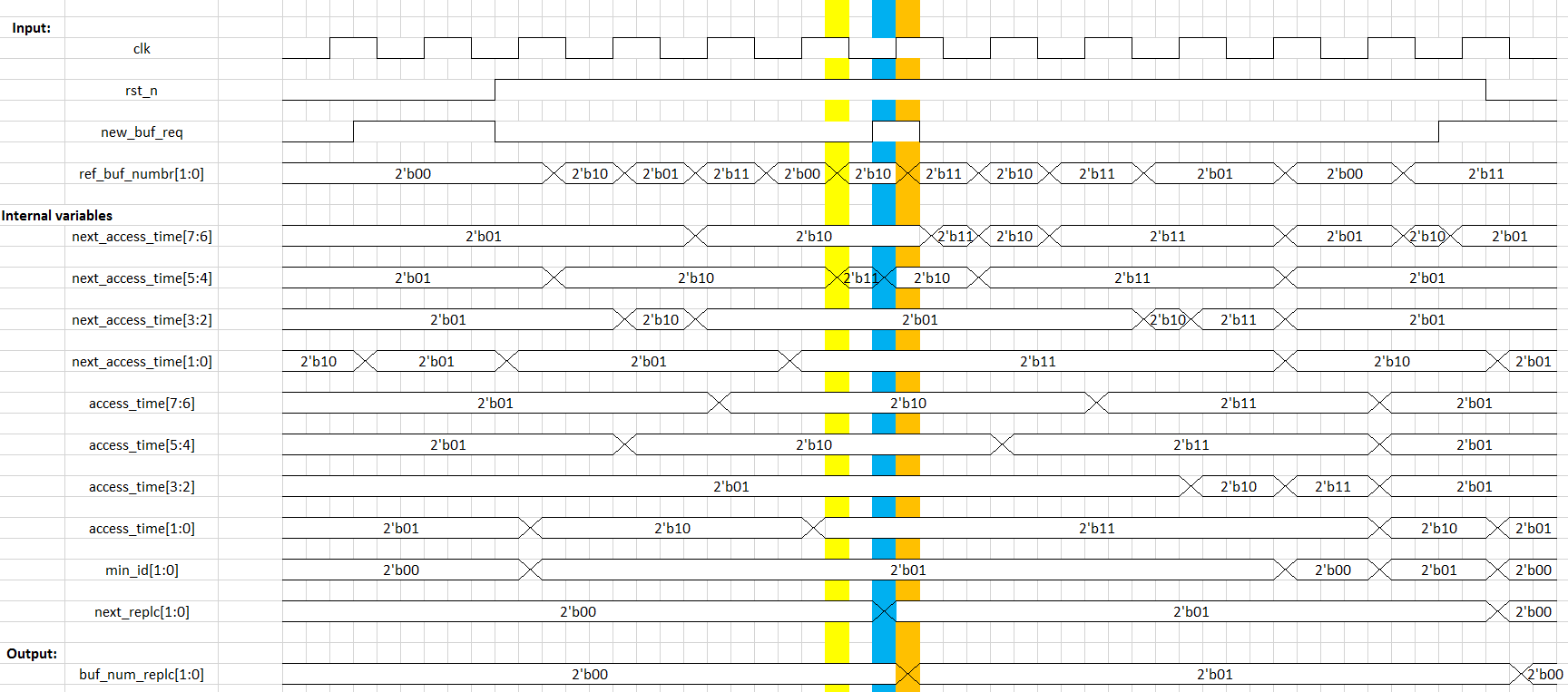


Figure 9. Timing chart when rst\_n is off and the new\_buf\_req is active

Yellow line:

* next\_accesss\_time increased based on the ref\_buf\_numbr.

Blue line:

* new\_buf\_req is active🡺the next\_access\_time forget the above increment and change according to the min\_id.
  + min\_id = 2’b01 🡺the next\_access\_time[3:2] = 2’b01, other bit keep the same value of the access\_time.
* next\_replc changed according to the min\_id.

Orange line:

* buf\_num\_replc change according to next\_replc (from 2’b00 to 2’b01)

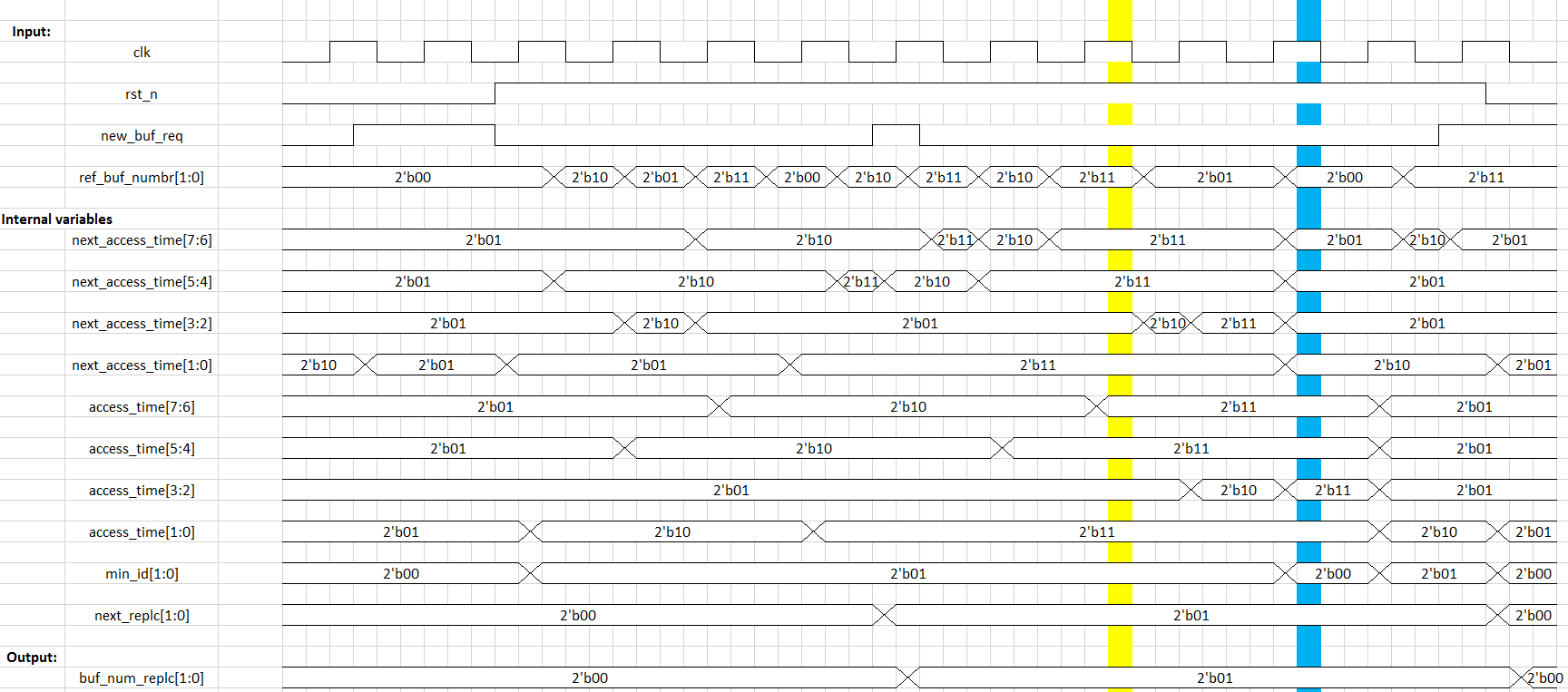


Figure 10. Timing chart when a buffer's access time = 2'b11 and all buffers’ access time = 2'b11

Yellow line:

* access\_time[7:6] = 2’b11, ref\_buf\_numbr = 2’b11 🡺next\_access\_time[7:6] keep its value (2’b11).

Blue line:

* access\_time = 8’hff, ref\_buf\_numbr = 2’b00 🡺next\_access\_time = 6’b01010110
* min\_id = 2’b00 when access\_time = 8’hff.

# TEST CASES

|  |  |  |
| --- | --- | --- |
| **No.** | **Test items** | **Expected** |
| * **When rst\_n is active, change value of new\_buf\_req:** | | |
| 1 | rst\_n = 1'b0 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'bxx | access\_time = 8'h55 buf\_num\_replc = 2'b00 |
| 2 | rst\_n = 1'b0 new\_buf\_req = 1'b1 ref\_buf\_numbr = 2'bxx | access\_time = 8'h55 buf\_num\_replc = 2'b00 |
| * **When rst\_n is off and new\_buf\_req is on:** | | |
| 3 | rst\_n = 1'b1 new\_buf\_req = 1'b1 ref\_buf\_numbr = 2'b00 | access\_time = 8'h55 buf\_num\_replc = 2'b00 |
| * **When new\_buf\_req is off. Operating normally:** | | |
| 4 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b00 | access\_time = 8'h56 buf\_num\_replc = 2'b00 |
| 5 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b00 | access\_time = 8'h57 buf\_num\_replc = 2'b00 |
| 6 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b01 | access\_time = 8'h5B buf\_num\_replc = 2'b00 |
| 7 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b01 | access\_time = 8'h5F buf\_num\_replc = 2'b00 |
| 8 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b10 | access\_time = 8'h6F buf\_num\_replc = 2'b00 |
| 9 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b11 | access\_time = 8'hAF buf\_num\_replc = 2'b00 |
| * **When access\_time is not 8’hff and new\_buf\_req is on:** | | |
| 10 | rst\_n = 1'b1 new\_buf\_req = 1'b1 ref\_buf\_numbr = 2'b11 | access\_time = 8'h9F buf\_num\_replc = 2'b10 |
| * **When new\_buf\_req is off, increase the access\_time to 8’hff** | | |
| 11 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b11 | access\_time = 8'hDF buf\_num\_replc = 2'b10 |
| 12 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b11 | access\_time = 8'hDF buf\_num\_replc = 2'b10 |
| 13 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b10 | access\_time = 8'hEF buf\_num\_replc = 2'b10 |
| 14 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b10 | access\_time = 8'hFF buf\_num\_replc = 2'b10 |
| * **When access\_time = 8’hff and new\_buf\_req is on:** | | |
| 15 | rst\_n = 1'b1 new\_buf\_req = 1'b1 ref\_buf\_numbr = 2'b10 | access\_time = 8'hFD buf\_num\_replc = 2'b00 |
| * **When the new\_buf\_req is off. Check the access\_time increment when a buffer’s access time is 2’b11 and referenced.** | | |
| 16 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b01 | access\_time = 8'hFD buf\_num\_replc = 2'b00 |
| * **When the new\_buf\_req is off. Check the access\_time increment when all four buffers’ access time are 2’b11 and referenced.** | | |
| 17 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b00 | access\_time = 8'hFE buf\_num\_replc = 2'b00 |
| 18 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b00 | access\_time = 8'hFF buf\_num\_replc = 2'b00 |
| 19 | rst\_n = 1'b1 new\_buf\_req = 1'b0 ref\_buf\_numbr = 2'b00 | access\_time = 8'h56 buf\_num\_replc = 2'b00 |