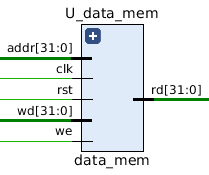
# 2.2 Component design & test

## 2.2.1 Component1 Data Mem

Contributor: yd1257 Yiren DAI

## 2.2.1.1 Implementation

* Block diagram:



* IN PORT:

|  |  |  |
| --- | --- | --- |
| Type | Name | Description |
| In | clk | clock signal |
| In | Rst | reset signal, active-high |
| In | We | Flag signal in sensitive list. Active-high |
| In | Addr[31:0] | We use addr to select which row we want |
| In | Wd[31:0] | Write the input into data mem |

* OUT PORT:

|  |  |  |
| --- | --- | --- |
| Out | Rd[31:0] | Read the input from data mem |

* What is this component for? How this component work?

*This part of VHDL code covers read from RAM and write to RAM vhdl code. Data Memory is a 128 rows \* 32 columns matrix. For the 32 columns (32 bits), we allocated them into 4 combined columns, each part is 8-bits byte. For the 128 rows, we use addr to select which row we want. As we know, addr have 32 bits, 7 bits is enough to represent 128 rows(2^7 = 128)*

## 2.2.1.2 Testbench

What you do in the testbench? Explain what you did with screenshots of your code.

* I used Excel to generate the 1000 test bench:

1. addr from 0 – 127 and wd = 00000001000000100000001100000100
2. addr from 0 – 127 and wd = 00000100000000110000001000000001
3. addr from 0 – 127 and wd = 11111111111111101111110111111100
4. addr from 0 – 127 and wd = 11111100111111011111111011111111
5. addr from 0 – 127 and wd = 00000001000000101111110111111100
6. addr from 0 – 127 and wd = 11111100111111010000001000000001
7. addr from 0 – 127 and wd = 00000100000000111111111011111111
8. addr from 0 – 127 and wd = 11111110111111110000001100000100

…...



* At first, I set the reset signal to ‘1’ for 20 seconds to reset the whole data memory. Then, I gave 1000 different kinds of test bench to test if it could generate the correct output. I set we = ‘1’ and gave different combination of addr and wd.

## Functional simulation

2.2.1.3.1 Functional simulation

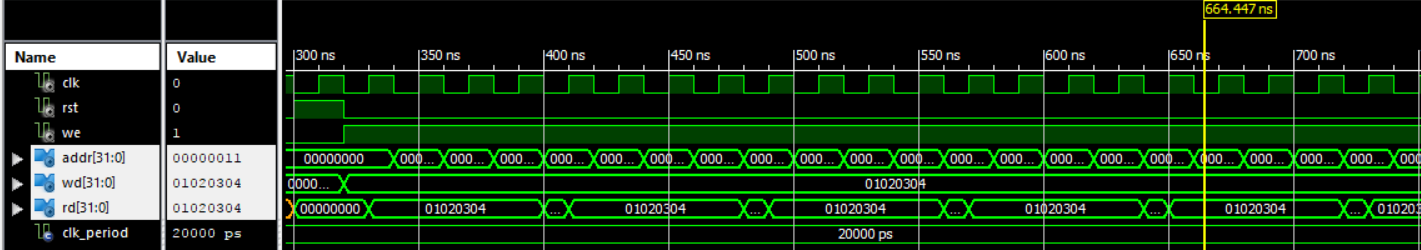


Figure 2.2.1.3.1 for all cases

2.2.1.3.2 Functional simulation

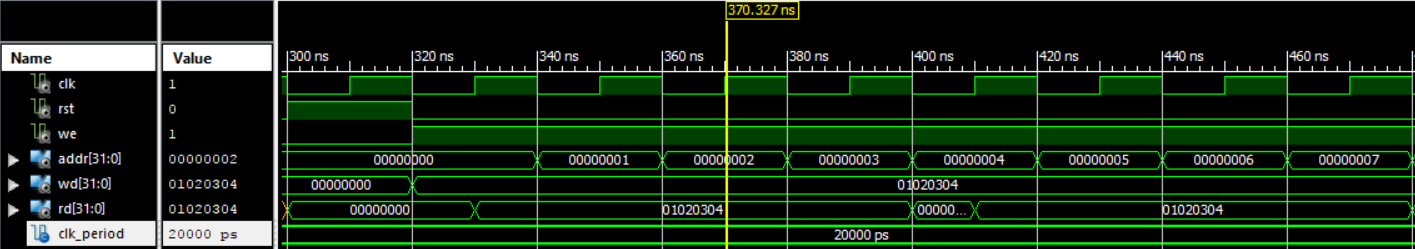


Figure 2.2.1.3.2 set the rst = ‘1’ and reset everything and then continue change the addr

2.2.1.3.3 Functional simulation

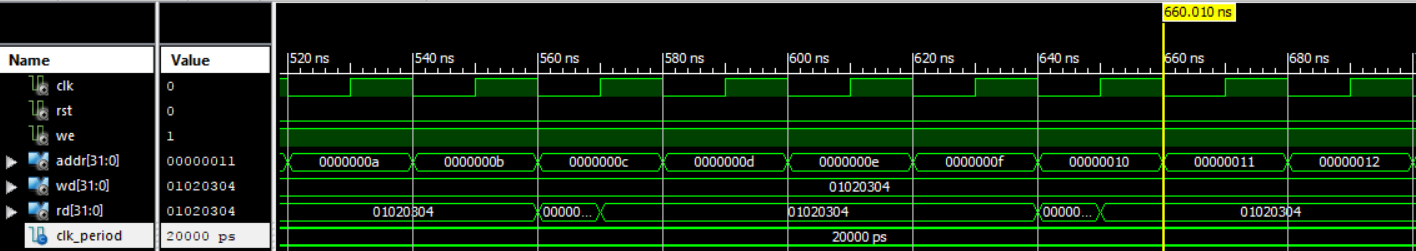


Figure 2.2.1.3.3 continue change the addr and check the rd[31:0]

## Timing simulation

2.2.1.4.1 Timing simulation

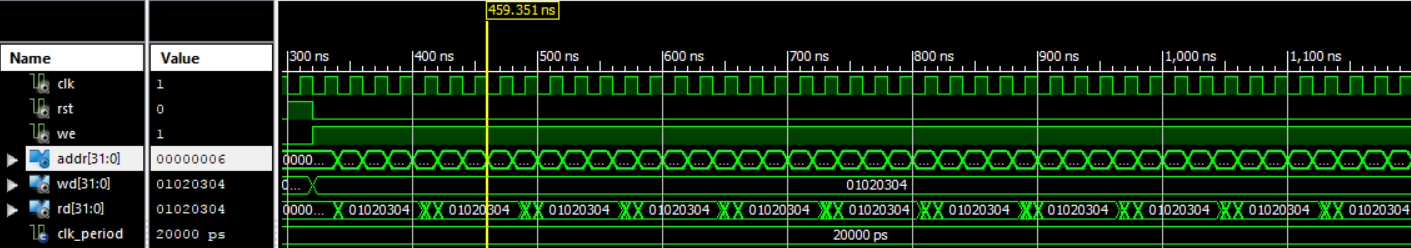


Figure 2.2.1.4.1 for all cases

2.2.1.4.2 Functional simulation

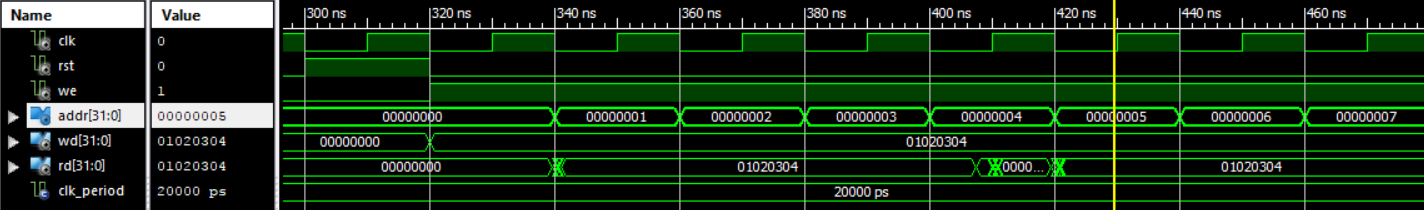


Figure 2.2.1.4.2 set the rst = ‘1’ and reset everything and then continue change the addr

2.2.1.4.3 Functional simulation

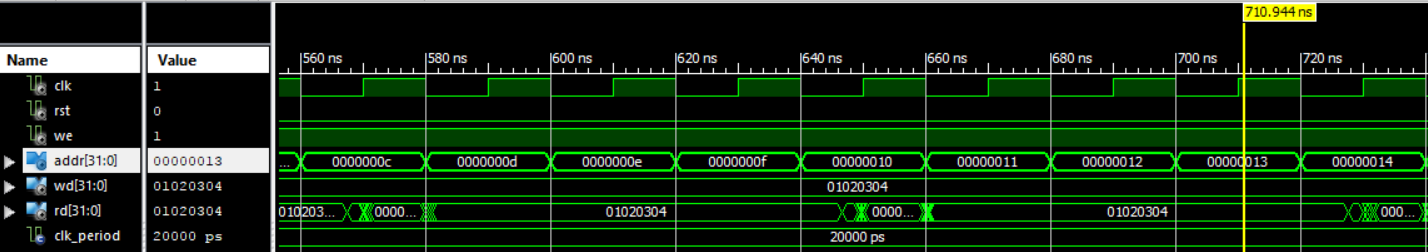


Figure 2.2.1.4.3 continue change the addr and check the rd[31:0]

## Timing Analysis

|  |  |  |
| --- | --- | --- |
|  | **Post-synthesis** | **Post-PAR** |
| Time |  |  |

|  |  |  |
| --- | --- | --- |
| **Resources** | **Post-synthesis** | **Post-PAR** |
| Slice registers | 4096 out of 126800 3% | 4,096 out of 126,800 3% |
| Slice LUTs | 1480 out of 63400 2% | 1,476 out of 63,400 2% |
| IOBs | 74 out of 210 35% | 74 out of 210 35% |