LAB-4

Due to:
Part 1: Sunday, April 24, 2016, 23:55hrs

Submission: via COW

Part 1: ISBN_Memory (Individual Work)

(This part of the lab will be performed and submitted individually.)

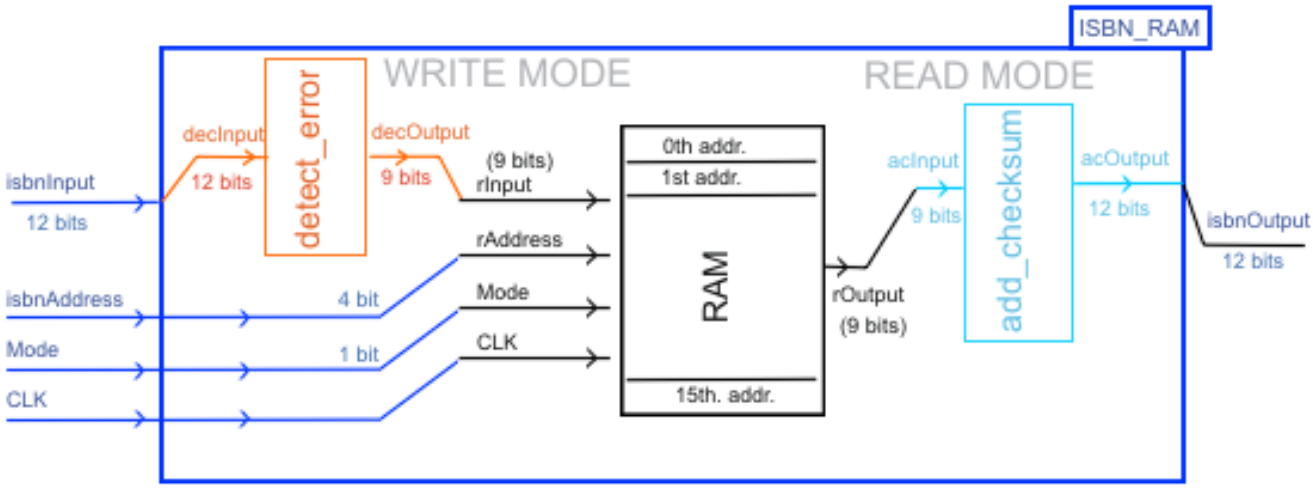
Problem Definition

In this part of your assignment you are going to implement a specific kind of memory called ISBN_RAM as a verilog module.

There are two modes in this ISBN_RAM, write and read. In the write mode, the module takes a 12-bit binary number with **data** and **check** bits as input. Then it controls the **data** and **check** bits and writes 9 **data** bits to the memory.

In the read mode, the module reads 9-bit **data** from RAM, adds 3 **check** bits and produces 12 bit output that contains both **data** and **check** bits.

The module structure is given in the figure below:



Add_checksum Module:

Add_checksum module takes a 9 bit binary number(**acInput**) that consist of only **data** bits. For error checking, it adds 3 **checksum** bits and gives 12 bit output.

Input(**acInput**) Format									
Bit position	1	2	3	4	5	6	7	8	9
	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9

Above, d_1 is the most significant bit and d_9 is the least significant bit.

- Checksum is a 3-bit binary number ($c_1c_2c_3$) where c_1 is the most significant bit and c_3 is the least significant bit. The checksum is calculated using the following formula;

$$checksum = \left(\sum_{i=1}^{9} i * d_i \right) \ modulus \ 8$$

Equation 1

- Output (**acOutput**) consists of 9 data and 3 checksum bits. 3 checksum bits c_1, c_2 and c_3 must be located at positions 10, 11 and 12 respectively. Also data bits d_1 to d_9 must be located at positions 1, 2, ..., 9 respectively.

Output(**acOutput**) Format												
Bit position	1	2	3	4	5	6	7	8	9	10	11	12
	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	c_1	c_2	c_3

- Use the following Verilog definition for the module:

```
module add_checksum(
        input [1:9] acInput,
        output reg [1:12] acOutput
   );
```

- Sample input and outputs are given below;

INPUT										OUTPUT											
d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9		d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	c_1	c_2	c_3
1	0	1	0	1	0	1	0	1		1	0	1	0	1	0	1	0	1	0	0	1
1	1	1	1	1	1	1	1	1		1	1	1	1	1	1	1	1	1	1	0	1

Detect_error Module:

Detect_error module takes a 12 bit binary number (**decInput**) that consist of both **checksum** and **data** bits. It calculates checksum ($c_1{'}c_2{'}\ c_3{'}$) of 9 **data** bits and compares it with 3-bit checksum part ($c_1c_2c_3$) of the decInput. If they are equal then gives the **data** bits ($d_1 \dots d_9$) as output, otherwise gives 000000000 as output.

Input(**decInput**) Format												
Bit position	1	2	3	4	5	6	7	8	9	10	11	12
	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	c_1	c_2	c_3

- 3 checksum bits c_1, c_2 and c_3 are located at positions 10,11 and 12 respectively.
- The rest are the 9 data bits where d_1 is the most significant bit and d_9 is the least significant bit.
- You can assume that 9 data bits will not be equal to all zeros.
- The checksum ($c_1{'}c_2{'}\ c_3{'}$) of 9 input data bits is calculated according to the Equation 1.
- If the calculated checksum ($c_1{'}c_2{'}\ c_3{'}$) equals to the given checksum ($c_1\ c_2\ c_3$) then 9 data bits ($d_1 \dots d_9$) are given as output.
- If the calculated checksum ($c_1{'}c_2{'}\ c_3{'}$) does **not** equal to the given checksum ($c_1c_2c_3$) then 000000000 (9 zero bits) is given as output.

Output(**decOutput**) Format									
Bit position	1	2	3	4	5	6	7	8	9
	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9

- Use the following Verilog definition for the module:

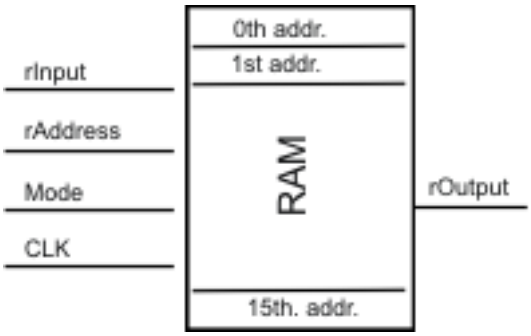
```
module detect_error(
        input [1:12] decInput,
        output reg [1:9] decOutput
   );
```

- Sample input and outputs are given below;

INPUT												OUTPUT								
d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	c_1	c_2	c_3	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9
1	0	1	0	1	0	1	0	1	0	0	1	1	0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0

RAM Module

The RAM will use 4-bit addresses. Each address will contain 9 bits of data, i.e. a memory *word* consists of 9 bits. Design the following RAM in verilog:



- Initially all RAM contains 9'b111111111 values.
- Mode(1 bit):
    - **0 - read mode:** Data stored at **rAddress** location of the RAM must be represented in **rOutput** when the positive edge **CLK** pulse comes.
    - **1 - write mode**: **rInput** must be stored at **rAddress** location of the RAM when the

positive edge **CLK** pulse comes.

- Use the following lines as the port definition of the **RAM**:

```
module RAM(
        input [8:0] rInput,
        input CLK,
        input Mode, //0:read, 1:write
        input [3:0] rAddress,
        output reg [8:0] rOutput
);
```

ISBN_RAM Module:

Three modules (RAM, detect_error and add_checksum) described above will be used by an upper module in which inputs and outputs of the other modules are defined. **You should not edit this module.**

- The module definition for the main module is below:

```
module ISBN_RAM(
        input [11:0] isbnInput,
        input CLK,
        input Mode, //0:read, 1:write
        input [3:0] isbnAddress,
        output  [11:0] isbnOutput
);
```

- Sample write and read operations are given below;

isbnAddress	**Mode**	**isbnInput**(red bits are erroneous bits)												**isbnOutput**
		d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	c_1	c_2	c_3	
0000	1(write)	1	0	1	0	1	0	1	0	1	0	0	1	will not be checked
0001		1	0	1	0	1	0	1	0	1	0	1	1	will not be checked

isbnAddress	**Mode**	**isbnInput**	**isbnOutput**											
			d_1	d_2	d_1	d_4	d_5	d_6	d_7	d_8	d_9	c_1	c_2	c_3
0000	0(read)	X	1	0	1	0	1	0	1	0	1	0	0	1
0001			0	0	0	0	0	0	0	0	0	0	0	0
0010			1	1	1	1	1	1	1	1	1	1	0	1

Hints

- You should not edit ISBN_RAM Module. It will be provided for you and will not be implemented by you.
- "RAM", "detect_error", "add_checksum" and "ISBN_RAM" modules will be tested **separately**.

Deliverables:

- Implement both modules in a single Verilog file: lab4_1.v. Do NOT submit your testbenches, or project file (.xise).
- You can share your testbenches on the newsgroup.
- Submit the file through the COW system before the given deadline.
- This part is required to be done individually, not with your partner. Any kind of cheating is not allowed.
- Use the newsgroup metu.ceng.course.232 for any questions regarding to the homework.