硬體描述語言設計　Homework Assignment #5

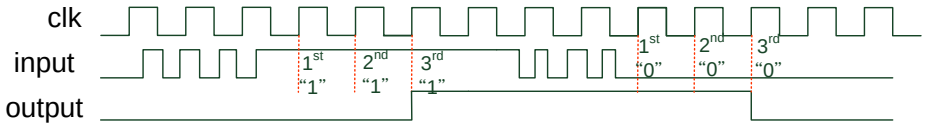
Due 12/29/2017 (Friday)

(** 12/22 上課將會說明題目，test bench 檔將另行公布**)

Q1. (25%) Debouncing circuit

Debouncing circuit will sample input signal at the rising edges of the clock and will change its output state only when a consistent signal is sampled in 3 consecutive clock cycles.

Note: 講義的 code 只能 debounce 0→1 的彈跳，作業要能同時處理 0→1 以及 1→0 的彈跳




```
module debouncing(clk,rst,in,out);
input     clk;
input     rst;              // synchronous reset
input     in;
output    reg   out;

endmodule
```

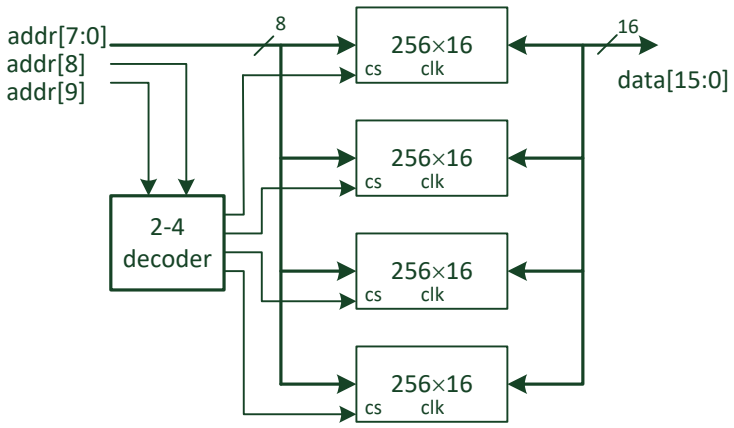
Q2. (30%) 1024X16 memory system design

Please design a 1024X16 memory system consisting of four 256X16 single port memory modules

- The memory module is addressed by a 8-bit address addr[7:0] and has a bi-directional data port “data[15:0]”. The module has 2 control signals
- rw: write if rw = 1, read if rw = 0
- cs: active low chip select signal, the RAM functions only if cs = 0. The data port is high impedance if cs = 1

The two MSBs addr[9:8] are used to select memory module.

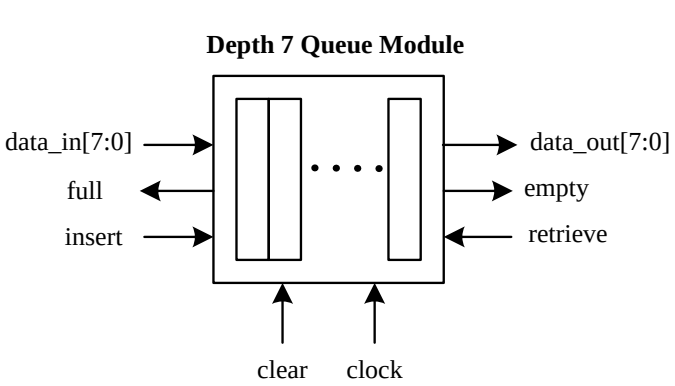
```
module ram1024X16(clk, addr, data,rw,cs);
input clk, rw, cs;
input [9:0] addr;
inout [15:0] data;

endmodule
```

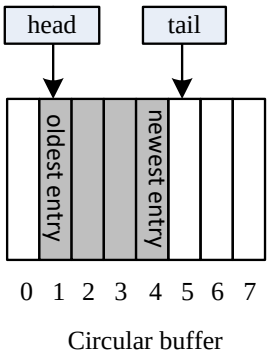


Q3. (45%) Queue design

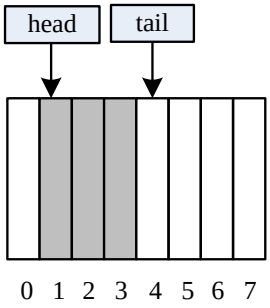
Design a queue of depth 7 and 8-bit wide

- It has two I/O ports, data_in[7:0] as input port and data_out[7:0] as output port
- When “insert” signal (active high) is asserted, an 8-bit data entry will be inserted into the queue at the positive edge of the clock if flag “full” is false (0). Nothing happens if flag “full” is true
- When “retrieve” signal (active high) is asserted, an 8-bit data entry will be retrieved from the queue at the negative edge of the clock if flag “empty” is false (0). “FF” will appear on data_out[7:0] if flag “empty” is true
- The queue consists of a circular buffer and two pointers. The circular buffer size is 8 in depth but can accommodate at most 7 entries. The “head” pointer always points to the location of the earliest data inserted to the queue. It is also the data to be read out if a retrieve command is asserted. The “tail” pointer always points to the vacant location next to the newset data in the queue. It is also the location where the next inserted data will be stored.
- The circular buffer is empty if (head == tail) and is full if (head == tail + 1)
- Each insert instruction is accomplised in two clock cycles. In clock cyle 1, the data is written to the location indicated by the “tail” pointer. In cycle 2, the content of the “tail” pointer is increased by 1.
- Each retrieve instruction is also accomplished in two clock cycles. In clock cycle 1, the data is read from the location indicated by the “head” pointer. In cycle 2, the content of the “head” pointer is increased by 1.
- When “clear” (active high) is applied, the “head” and the “tail” pointers are both reset to 0, the “empty” flag is set to 1 and the “full” flag is set to 0
- Both “full” and “empty” flags are updated at the negative edge of the clock

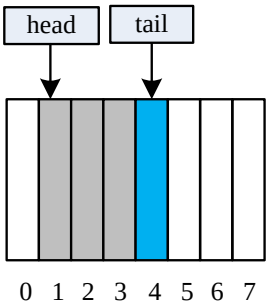




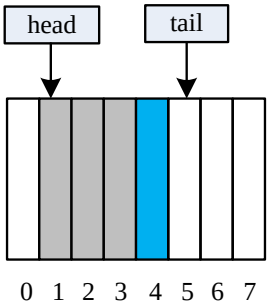
Circular buffer



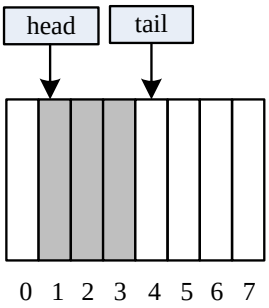
Before insert



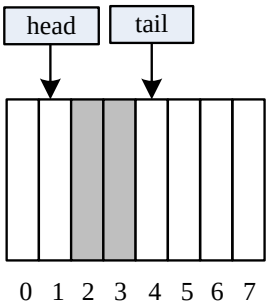
Insert cycle 1



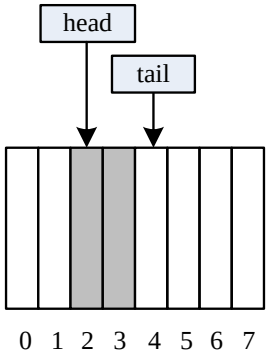
Insert cycle 2



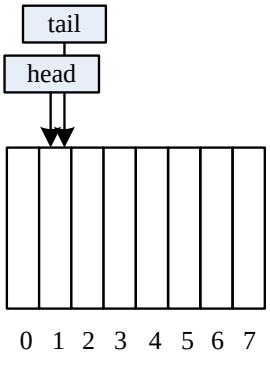
Before retrieve



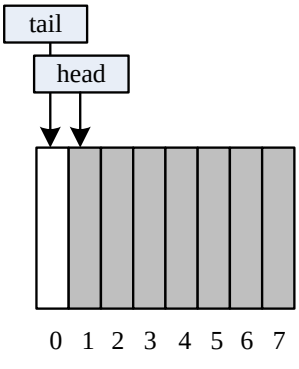
retrieve cycle 1



retrieve cycle 2



Queue is empty



Queue is full

```
module Queue(clk,clear,data_in,insert,full,data_out,retrieve,empty);
input       clk;
input       clear;
//write side
```

```
input       [7:0]data_in;
input       insert;
output      reg    full;
//read side
output      reg [7:0] data_out;
input       retrieve;
output      reg empty;

endmodule
```