1. **[Design and Implementation of Simple Hardware Router]** In this assignment, you have to implement a primitive hardware router. The task of the router is to take the packets from the input buffer I and route them through the proper output interface. Each packet contains the (a) Packet ID (8bit) (b) Destination IP address (32bit) and (c) Transmit count field n ($n \leq 4$) (3 bit). Each output buffer can store 4 packets and maintains a FIFO queue. The router implements a routing table with entries of the form (X,Y), which indicates that if the destination IP address of the packet (coming from the input buffer I) is X, the packet should be routed to the output buffer Y. Once the router receives the packet from the input buffer, first it decides on the interface/output buffer Y to route the packet, next it makes n copies of the packet (here n=0 indicates dropping the packet) and finally enques/stores those n packets to the respective output buffer Y. If required, router transmits (in FIFO order) the existing packets of the respective output buffer Y to the network, to make space for (copies of) this incoming packet. The design and implementation of the hardware to process the given packets has to be performed in Verilog, and the design description should be behaviorally simulated using a proper testbench. The routing table is given in Table 1.

Table 1: Routing Table

IP address range	Output interface
188.39.0.0 to 188.39.255.255	4
170.153.0.0 to 170.153.255.255	2
83.168.0.0 to 83.168.255.255	1
104.148.0.0 to 104.148.255.255	3

The IP address is usually represented in dot-decimal notation, consisting of four decimal numbers, each ranging from 0 to 255, separated by dots, e.g., 172.16.254.1 (you can simply ignore the dots while storing IP addresses in memory; you may consider them as sequence of decimal numbers). The Routing table uses the first two components (say, 172.16, indicating the range 172.16.0.0 to 172.16.255.255) to decide on the output interface. The router also maintains a list of the Drop history, indicating the list of dropped packets by the router. The list can contain 8 packets at maximum.

Test-bench: Consider the the state of the input buffer I as shown in Table 2 at any point of time. Transmit all the packets of I in sequence and show the state of the simulation in (a) output buffers (b) drop table. (20 marks)

Clock scheme: Your router shall use one clock cycle to sample the input data and decide the suitable output buffer. For writing n copies of the packet in the output buffer, your design shall take n clock cycles. Only when processing of one packet is completed with suitable writing of output, the router shall again sample the input port. Your test bench code shall provide input data to the router at suitable intervals so that the no input is overwritten without being sampled by the router.

I/O interfaces:

Following Fig.1

```
module buffer( input clk, input rst, input [42:0] buffer_in, output [42:0] buffer_out);

module router( input [42:0] data_in, input clk, input rst, output [42:0] channel_1,
output [42:0] channel_2, output [42:0] channel_3, output [42:0] channel_4, output reg
ready);
```

Commenting: Add suitable comments to your code in order to explain the I/O interfaces

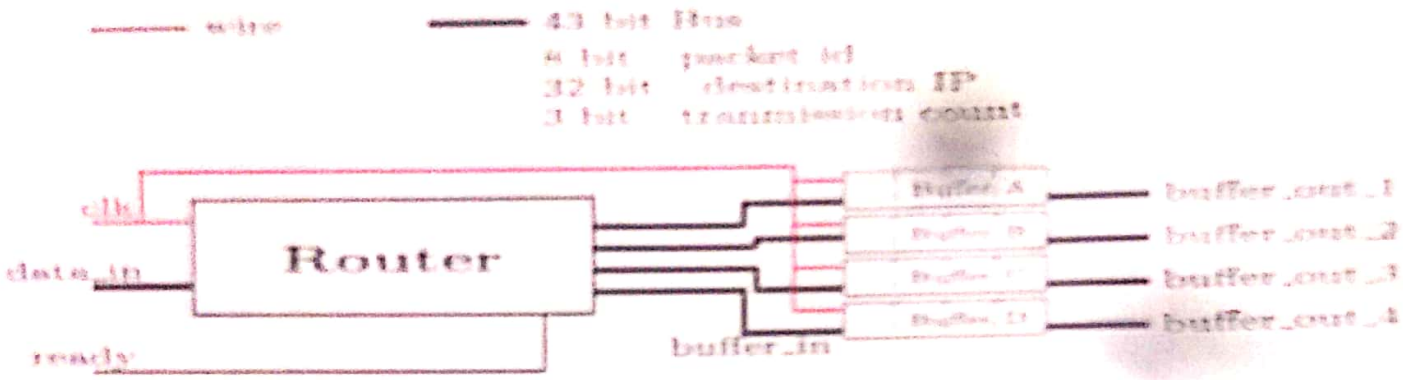

Figure 1: Block Digram

Table 2: Input Buffer

Packet Id	IP	Transmit Count
1	170 153 26 104	2
2	170 153 127 218	4
3	83 168 234 22	2
4	104 148 200 16	2
5	83 168 252 74	1
6	188 39 221 216	2
7	104 148 115 57	4
8	170 153 78 224	3
9	188 39 3 195	4
10	83 168 164 223	3
11	104 148 202 212	3
12	83 168 90 183	2
13	83 168 150 92	0
14	104 148 19 46	1
15	166 30 150 38	3
16	170 154 75 38	0
17	170 153 12 220	4
18	170 153 59 168	0
19	83 168 1 82	4
20	104 148 23 173	0