```
module lab2_tb();
// constant declarations
localparam PKT_WIDTH = 96;
localparam DATA_WIDTH = 32;

// one reg type var for each input of the circuit
reg clk;
reg [DATA_WIDTH-1:0] data_i;
reg valid_i;

// one wire type var for the outputs
wire [DATA_WIDTH-1:0] data_o;
wire valid_o;


lab2 lab2 (
        .clk (clk),
        .data_i (data_i),
        .valid_i (valid_i),
        .data_o (data_o),
        .valid_o (valid_o)
);

initial begin
        clk = 0;
end

always begin
        clk = ~clk;
        #50;
end

task waitalot;
begin
        @(negedge clk);
        @(negedge clk);
        @(negedge clk);
        @(negedge clk);
        @(negedge clk);
        @(negedge clk);
        @(negedge clk);
        @(negedge clk);
        @(negedge clk);
end
endtask

initial begin
        data_i = 0;
        valid_i = 0;

        // let's wait a few clock cycles before sending any data in
        @(negedge clk);
        @(negedge clk);
        @(negedge clk);

        valid_i = 1'b1;

        @(negedge clk);

        data_i = {29'd123,3'b101}; // data + header
        valid_i = 1'b1;

        @(negedge clk);
```

```
data_i = 32'd321; // data
valid_i = 1'b1;

@(negedge clk);

data_i = {3'b010,10'd98,10'd99,6'd5,3'b101}; // tail + source + destination + hops + data
valid_i = 1'b1;

@(negedge clk);
data_i = 32'd0;
valid_i = 1'b0;

waitalot;
// building a package with a broken header

valid_i = 1'b1;
@(negedge clk);
data_i = {29'd1234,3'b111}; // data + header
valid_i = 1'b1;
@(negedge clk);
data_i = 32'd3210; // data
valid_i = 1'b1;
@(negedge clk);
data_i = {3'b010,10'd980,10'd990,6'd8,3'b101}; // tail + source + destination + hops + data
valid_i = 1'b1;
@(negedge clk);
data_i = 32'd0;
valid_i = 1'b0;

waitalot;
// building a package with a broken tail

valid_i = 1'b1;
@(negedge clk);
data_i = {29'd1234,3'b101}; // data + header
valid_i = 1'b1;
@(negedge clk);
data_i = 32'd3210; // data
valid_i = 1'b1;
@(negedge clk);
data_i = {3'b000,10'd980,10'd990,6'd9,3'b101}; // tail + source + destination + hops + data
valid_i = 1'b1;
@(negedge clk);
data_i = 32'd0;
valid_i = 1'b0;

waitalot;
// building a package with too many hops

valid_i = 1'b1;
@(negedge clk);
data_i = {29'd1234,3'b101}; // data + header
valid_i = 1'b1;
@(negedge clk);
data_i = 32'd3210; // data
valid_i = 1'b1;
@(negedge clk);
data_i = {3'b010,10'd980,10'd990,6'd21,3'b101}; // tail + source + destination + hops + data
valid_i = 1'b1;
@(negedge clk);
data_i = 32'd0;
valid_i = 1'b0;

waitalot;
$finish();
```

```
end

endmodule
```