```
module control_unit( input[2:0] opcode,
                           input reset,
                           output reg reg_dst,mem_to_reg,
                           output reg jump,branch,mem_read,mem_write,alu_src,reg_write

 );
 always @(*)
 begin
      if(reset == 1'b1) begin
                reg_dst = 1'b0;  //0
                mem_to_reg = 1'b0;
                jump = 1'b0;
                branch = 1'b0;
                mem_read = 1'b0;
                mem_write = 1'b0;
                alu_src = 1'b0;
                reg_write = 1'b0;
      end
      else begin
      case(opcode)
    3'b000: begin  reg_dst = 1'b1;  // R-type
                   mem_to_reg = 1'b0;
                   jump = 1'b0;
                   branch = 1'b0;
                   mem_read = 1'b0;
                   mem_write = 1'b0;
                   alu_src = 1'b0;
                   reg_write = 1'b1;  // 28/7
            end
    3'b010: begin    // LB //0
                   reg_dst = 1'b0;
                   mem_to_reg = 1'b1;
                   jump = 1'b0;
                   branch = 1'b0;
                   mem_read = 1'b1;
                   mem_write = 1'b0;
                   alu_src = 1'b1;
                   reg_write = 1'b1;
            end
    3'b011: begin    // SB
                   reg_dst = 1'b0;
                   mem_to_reg = 1'b0;
                   jump = 1'b0;
                   branch = 1'b0;
                   mem_read = 1'b0;
                   mem_write = 1'b1;
                   alu_src = 1'b1;
                   reg_write = 1'b0;
            end
    3'b001: begin    // Addi
                   reg_dst = 1'b0;
                   mem_to_reg = 1'b0;
                   jump = 1'b0;
                   branch = 1'b0;
                   mem_read = 1'b0;
                   mem_write = 1'b0;
                   alu_src = 1'b1;
                   reg_write = 1'b1;
                    //
            end
   3'b100:  begin    //beq //0
                   reg_dst = 1'b0;
```

```
                  mem_to_reg = 1'b0;
                  jump = 1'b0;
                  branch = 1'b1;
                  mem_read = 1'b0;
                  mem_write = 1'b0;
                  alu_src = 1'b0;
                  reg_write = 1'b0;
           end
   3'b101: begin    //jump
                  reg_dst = 1'b0;
                  mem_to_reg = 1'b0;
                  jump = 1'b1;
                  branch = 1'b0;
                  mem_read = 1'b0;
                  mem_write = 1'b0;
                  alu_src = 1'b0;
                  reg_write = 1'b0;
           end
     default: begin
                  reg_dst = 1'b1;
                  mem_to_reg = 1'b0;
                  jump = 1'b0;
                  branch = 1'b0;
                  mem_read = 1'b0;
                  mem_write = 1'b0;
                  alu_src = 1'b0;
                  reg_write = 1'b1;
              end
   endcase
   end
end
endmodule
```