```
  1: library STD;
  2: library IEEE;
  3: use IEEE.std_logic_1164.all;
  4:
  5: entity rd_wr_hit_miss_reg is
  6:     port(
  7:         rd_wr       :   in  std_logic;
  8:         rd_wr_set_en:   in  std_logic;
  9:         hit_miss    :   in  std_logic;
 10:         hit_miss_en :   in  std_logic; --signal for latching hit miss
 11:         clk         :   in  std_logic;
 12:         reset       :   in  std_logic;
 13:         Gnd         :   in  std_logic;
 14:         rd_wr_o     :   out std_logic;
 15:         rd_hit      :   out std_logic;
 16:         wr_hit      :   out std_logic;
 17:         rd_miss     :   out std_logic;
 18:         wr_miss     :   out std_logic
 19:     );
 20: end rd_wr_hit_miss_reg;
 21:
 22: architecture structural of rd_wr_hit_miss_reg is
 23:
 24:     component dff_reset
 25:         port(
 26:             d       :   in  std_logic;
 27:             clk     :   in  std_logic;
 28:             reset   :   in  std_logic;
 29:             Gnd     :   in  std_logic;
 30:             q       :   out std_logic
 31:         );
 32:     end component;
 33:
 34:     component Dlatch_Reset
 35:     port(
 36:         d       :   in  std_logic;
 37:         clk     :   in  std_logic;
 38:         reset   :   in  std_logic;
 39:         Gnd     :   in  std_logic;
 40:         q       :   out std_logic
 41:     );
 42:     end component;
 43:
 44:     component and2
 45:         port(
 46:             in1     :   in  std_logic;
 47:             in2     :   in  std_logic;
 48:             out1    :   out std_logic
 49:         );
 50:     end component;
 51:
 52:     component and4
 53:         port(
 54:             in1     :   in  std_logic;
 55:             in2     :   in  std_logic;
 56:             in3     :   in  std_logic;
 57:             in4     :   in  std_logic;
 58:             out1    :   out std_logic
 59:         );
 60:     end component;
 61:
 62:     component and3
 63:         port(
 64:             in1     :   in  std_logic;
 65:             in2     :   in  std_logic;
 66:             in3     :   in  std_logic;
 67:             out1    :   out std_logic
 68:         );
 69:     end component;
 70:
 71:     component or2
 72:         port(
 73:             in1     :   in  std_logic;
 74:             in2     :   in  std_logic;
 75:             out1    :   out std_logic
 76:         );
 77:     end component;
 78:
 79:     component invX1
 80:         port(
 81:             in1     :   in  std_logic;
 82:             out1    :   out std_logic
 83:         );
 84:     end component;
 85:
 86:     signal n_rd_wr      :   std_logic;
 87:     signal n_hit_miss   :   std_logic;
 88:     signal rhit         :   std_logic;
 89:     signal rmiss        :   std_logic;
 90:     signal whit         :   std_logic;
 91:     signal wmiss        :   std_logic;
 92:
 93:     signal reg1_in      :   std_logic;
 94:     signal reg2_in      :   std_logic;
 95:
 96:     signal latched_rd_wr    :   std_logic;
 97:     signal latched_hit_miss :   std_logic;
 98:     signal nen              :   std_logic;
 99:     signal clk_en1          :   std_logic;
100:     signal clk_en2          :   std_logic;
101:
102:     signal enabled_clk      :   std_logic;
103:
104:     signal nreset           :   std_logic;
105:
106:     for inv1, inv2, n_en, nrst  :   invX1 use entity work.invX1(structural);
107:     for and_1, and_2, and_3, and_4, asdf, reg1in  : and2 use entity work.and2(
structural);
108:     for clk_en, nrsten  :   or2 use entity work.or2(structural);
109:     for reg1    :   dff_reset use entity work.dff_reset(structural);
110:     for latch   :   Dlatch_Reset use entity work.Dlatch_Reset(structural);
111:
112:
113: begin
114:
115:     rd_wr_o <=  latched_rd_wr;
116:
117:     inv1    :   invX1       port map(latched_rd_wr, n_rd_wr);
118:     inv2    :   invX1       port map(latched_hit_miss, n_hit_miss);
119:     n_en    :   invX1       port map(rd_wr_set_en, nen);
120:     nrst    :   invX1       port map(reset, nreset);
121:
122:     and_1   :   and2        port map(latched_rd_wr, latched_hit_miss, rd_hit);
123:     and_2   :   and2        port map(n_rd_wr, latched_hit_miss, wr_hit);
```

```
  124:     and_3   :   and2        port map(latched_rd_wr, n_hit_miss, rd_miss);
  125:     and_4   :   and2        port map(n_rd_wr, n_hit_miss, wr_miss);
  126:
  127:
  128:     clk_en  :   or2         port map(nen, clk, clk_en1);
  129:     nrsten  :   or2         port map(nreset, clk, clk_en2);
  130:     asdf    :   and2        port map(clk_en1, clk_en2, enabled_clk);
  131:
  132:     reg1in  :   and2        port map(rd_wr,  rd_wr_set_en, reg1_in);
  133:
  134:     reg1    :   dff_reset   port map(reg1_in, enabled_clk, reset, Gnd, latched
_rd_wr);
  135:     latch   :   Dlatch_Reset    port map(hit_miss, hit_miss_en, reset, Gnd, la
tched_hit_miss);
  136:
  137: end structural;
```