

Cache_Cell_Row.vhd

```

1: library STD;
2: library IEEE;
3: use IEEE.std_logic_1164.all;
4:
5: entity Cache_Cell_Row is
6:     port(
7:         Data_In      : in  std_logic_vector(7 downto 0);
8:         Tag_In       : in  std_logic_vector(2 downto 0);
9:         Set_Valid    : in  std_logic;
10:        Col0_Rd_En    : in  std_logic;
11:        Col1_Rd_En    : in  std_logic;
12:        Col2_Rd_En    : in  std_logic;
13:        Col3_Rd_En    : in  std_logic;
14:        Col0_Wr_En    : in  std_logic;
15:        Col1_Wr_En    : in  std_logic;
16:        Col2_Wr_En    : in  std_logic;
17:        Col3_Wr_En    : in  std_logic;
18:        Tag_Wr_En     : in  std_logic;
19:        Row_Rd_En     : in  std_logic;
20:        Row_Wr_En     : in  std_logic;
21:        Row_En        : in  std_logic;
22:        Gnd           : in  std_logic;
23:        reset         : in  std_logic;
24:        Data_Out      : out std_logic_vector(7 downto 0);
25:        Tag_Out       : out std_logic_vector(2 downto 0);
26:        Valid_Out     : out std_logic
27:    );
28: end Cache_Cell_Row;
29:
30: architecture structural of Cache_Cell_Row is
31:
32:     component Cache_Cell_Valid
33:     port(
34:         s      : in  std_logic;
35:         r      : in  std_logic;
36:         Rd_En  : in  std_logic; --row read enable
37:         Wr_En  : in  std_logic; --row write enable
38:         q      : out std_logic
39:     );
40:     end component;
41:
42:     component Cache_Cell_Tag
43:     port(
44:         Data  : in  std_logic_vector(2 downto 0);
45:         Tag_Wr : in  std_logic;
46:         reset : in  std_logic;
47:         Gnd   : in  std_logic;
48:         Output : out std_logic_vector(2 downto 0);
49:         Row_En : in  std_logic
50:     );
51:     end component;
52:
53:     component Cache_Cell_Data_Block
54:     port(
55:         Data      : in  std_logic_vector(7 downto 0);
56:         W_En_r    : in  std_logic; --row wr enable
57:         W_En_c    : in  std_logic; --column wr enable
58:         reset     : in  std_logic;
59:         Gnd       : in  std_logic;
60:         Output    : out std_logic_vector(7 downto 0);
61:         Rd_En_r   : in  std_logic; --row rd enable
62:         Rd_En_c   : in  std_logic --column rd enable
63:     );
64:     end component;
65:
66:     component mux2_1
67:     port(
68:         in1      : in  std_logic;
69:         in2      : in  std_logic;
70:         sel      : in  std_logic;
71:         out1     : out std_logic
72:     );
73:     end component;
74:
75:     component and2
76:     port(
77:         in1      : in  std_logic;
78:         in2      : in  std_logic;
79:         out1     : out std_logic
80:     );
81:     end component;
82:
83:     component invX1
84:     port(
85:         in1      : in  std_logic;
86:         out1     : out std_logic
87:     );
88:     end component;
89:
90:     signal Tag_Wr      : std_logic;
91:     signal Val_Set     : std_logic;
92:     signal s_in        : std_logic;
93:     signal n_reset     : std_logic;
94:
95:     for nrst      : invX1 use entity work.invX1(structural);
96:     for andtag, valid_s : and2 use entity work.and2(structural);
97:     for rst_mux   : mux2_1 use entity work.mux2_1(structural);
98:     for data0, data1, data2, data3 : Cache_Cell_Data_Block use entity work.
Cache_Cell_Data_Block(structural);
99:     for tag       : Cache_Cell_Tag use entity work.Cache_Cell_Tag(structural);
100:    for valid      : Cache_Cell_Valid use entity work.Cache_Cell_Valid(structur
al);
101:
102:    begin
103:
104:        nrst      : invX1    port map(reset, n_reset);
105:        andtag    : and2     port map(Tag_Wr_En, Row_En, Tag_Wr);
106:        valid_s   : and2     port map(Set_Valid, Row_En, Val_Set);
107:
108:        rst_mux   : mux2_1   port map(Val_Set, n_reset, reset, s_in);
109:
110:        data0     : Cache_Cell_Data_Block port map(Data_In, Row_Wr_En, Col0_Wr_E
n, reset, Gnd, Data_Out, Row_Rd_En, Col0_Rd_En);
111:        data1     : Cache_Cell_Data_Block port map(Data_In, Row_Wr_En, Col1_Wr_E
n, reset, Gnd, Data_Out, Row_Rd_En, Col1_Rd_En);
112:        data2     : Cache_Cell_Data_Block port map(Data_In, Row_Wr_En, Col2_Wr_E
n, reset, Gnd, Data_Out, Row_Rd_En, Col2_Rd_En);
113:        data3     : Cache_Cell_Data_Block port map(Data_In, Row_Wr_En, Col3_Wr_E
n, reset, Gnd, Data_Out, Row_Rd_En, Col3_Rd_En);
114:
115:        tag       : Cache_Cell_Tag      port map(Tag_In, Tag_Wr, reset, Gnd, T
ag_Out, Row_En);
116:
117:        valid     : Cache_Cell_Valid    port map(s_in, reset, Row_En, Row_En,

```

12/11/17
19:59:42

Cache_Cell_Row.vhd

2

```
Valid_Out);  
118:  
119: end structural;
```