```vhdl
library STD;
library IEEE;
use IEEE.std_logic_1164.all;

entity Cache_Cell_Row is
    port(
        Data_In     :   in  std_logic_vector(7 downto 0);
        Tag_In      :   in  std_logic_vector(2 downto 0);
        Set_Valid   :   in  std_logic;
        Col0_Rd_En  :   in  std_logic;
        Col1_Rd_En  :   in  std_logic;
        Col2_Rd_En  :   in  std_logic;
        Col3_Rd_En  :   in  std_logic;
        Col0_Wr_En  :   in  std_logic;
        Col1_Wr_En  :   in  std_logic;
        Col2_Wr_En  :   in  std_logic;
        Col3_Wr_En  :   in  std_logic;
        Tag_Wr_En   :   in  std_logic;
        Row_Rd_En   :   in  std_logic;
        Row_Wr_En   :   in  std_logic;
        Row_En      :   in  std_logic;
        Gnd         :   in  std_logic;
        reset       :   in  std_logic;
        Data_Out    :   out std_logic_vector(7 downto 0);
        Tag_Out     :   out std_logic_vector(2 downto 0);
        Valid_Out   :   out std_logic
    );
end Cache_Cell_Row;

architecture structural of Cache_Cell_Row is

    component Cache_Cell_Valid
    port(
        s       :   in  std_logic;
        r       :   in  std_logic;
        Rd_En   :   in  std_logic; --row read enable
        Wr_En   :   in  std_logic; --row write enable
        q       :   out std_logic
    );
    end component;

    component Cache_Cell_Tag
    port(
        Data    :   in  std_logic_vector(2 downto 0);
        Tag_Wr  :   in  std_logic;
        reset   :   in  std_logic;
        Gnd     :   in  std_logic;
        Output  :   out std_logic_vector(2 downto 0);
        Row_En  :   in  std_logic
    );
    end component;

    component Cache_Cell_Data_Block
    port(
        Data    :   in  std_logic_vector(7 downto 0);
        W_En_r  :   in  std_logic; --row wr enable
        W_En_c  :   in  std_logic; --column wr enable
        reset   :   in  std_logic;
        Gnd     :   in  std_logic;
        Output  :   out std_logic_vector(7 downto 0);
        Rd_En_r :   in  std_logic; --row rd enable
        Rd_En_c :   in  std_logic  --column rd enable
    );
    end component;

    component mux2_1
    port(
        in1     :   in  std_logic;
        in2     :   in  std_logic;
        sel     :   in  std_logic;
        out1    :   out std_logic
    );
    end component;

    component and2
    port(
        in1     :   in  std_logic;
        in2     :   in  std_logic;
        out1    :   out std_logic
    );
    end component;

    component invX1
    port(
        in1     :   in  std_logic;
        out1    :   out std_logic
    );
    end component;

    signal Tag_Wr   :   std_logic;
    signal Val_Set  :   std_logic;
    signal s_in     :   std_logic;
    signal n_reset  :   std_logic;

    for nrst    :   invX1 use entity work.invX1(structural);
    for andtag, valid_s :   and2 use entity work.and2(structural);
    for rst_mux :   mux2_1 use entity work.mux2_1(structural);
    for data0, data1, data2, data3  :   Cache_Cell_Data_Block use entity work.Cache_Cell_Data_Block(structural);
    for tag     :   Cache_Cell_Tag use entity work.Cache_Cell_Tag(structural);
    for valid   :   Cache_Cell_Valid use entity work.Cache_Cell_Valid(structural);

begin

    nrst    :   invX1   port map(reset, n_reset);
    andtag  :   and2    port map(Tag_Wr_En, Row_En, Tag_Wr);
    valid_s :   and2    port map(Set_Valid, Row_En, Val_Set);

    rst_mux :   mux2_1  port map(Val_Set, n_reset, reset, s_in);

    data0   :   Cache_Cell_Data_Block   port map(Data_In, Row_Wr_En, Col0_Wr_En, reset, Gnd, Data_Out, Row_Rd_En, Col0_Rd_En);
    data1   :   Cache_Cell_Data_Block   port map(Data_In, Row_Wr_En, Col1_Wr_En, reset, Gnd, Data_Out, Row_Rd_En, Col1_Rd_En);
    data2   :   Cache_Cell_Data_Block   port map(Data_In, Row_Wr_En, Col2_Wr_En, reset, Gnd, Data_Out, Row_Rd_En, Col2_Rd_En);
    data3   :   Cache_Cell_Data_Block   port map(Data_In, Row_Wr_En, Col3_Wr_En, reset, Gnd, Data_Out, Row_Rd_En, Col3_Rd_En);

    tag     :   Cache_Cell_Tag          port map(Tag_In, Tag_Wr, reset, Gnd, Tag_Out, Row_En);

    valid   :   Cache_Cell_Valid        port map(s_in, reset, Row_En, Row_En,
Valid_Out);

end structural;
```