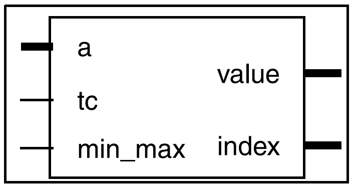
DW_minmax

Minimum/Maximum Value

Version, STAR and Download Information: IP Directory

Features and Benefits

- Parameterized number of inputs
- Parameterized word length
- Unsigned and signed (two's complement) data operation
- Dynamically selectable mode (minimum or maximum)
- Additional output gives an index of the minimum or maximum input
- Inferable using a function call



Description

DW_minmax determines the minimum or maximum value of multiple inputs. The *num_inputs* input operands of *width* length must be concatenated into a single input vector (a) of *num_inputs* × *width* length. The `value` output is the minimum of all inputs if `min_max`=0, and the maximum if `min_max`=1. The inputs and the `value` output are interpreted as unsigned numbers if `tc`=0, and as signed numbers if `tc`=1.

Table 1-1 Pin Description

Pin Name	Width	Direction	Function
a	*num_inputs* × *width* bit(s)	Input	Concatenated input data
tc	1 bit	Input	Two's complement control
min_max	1 bit	Input	Minimum/maximum control 0 = minimum (a) 1 = maximum (a)
value	*width* bit(s)	Output	Minimum/maximum value
index	ceil($\log_2$[*num_inputs*]) bit(s)	Output	Index of minimum/maximum input

Table 1-2 Parameter Description

Parameter	Values	Description
width	≥ 1	Input word length
num_inputs	≥ 2 Default: 2	Number of inputs

Table 1-3 Synthesis Implementations[a]

Implementation Name	Function	License Feature Required
cla	Carry-lookahead tree synthesis model	DesignWare
clas	Carry-lookahead/select tree synthesis model	DesignWare

a. During synthesis, Design Compiler will select the appropriate architecture for your constraints. However, you may force Design Compiler to use any architectures described in this table. For more, see *DesignWare Building Block IP User Guide*

Table 1-4 Simulation Models

Model	Function
DW01.DW_minmax_cfg_sim	Design unit name for VHDL simulation
dw/dw01/src/DW_minmax_sim.vhd	VHDL simulation model source code
dw/sim_ver/DW_minmax.v	Verilog simulation model source code

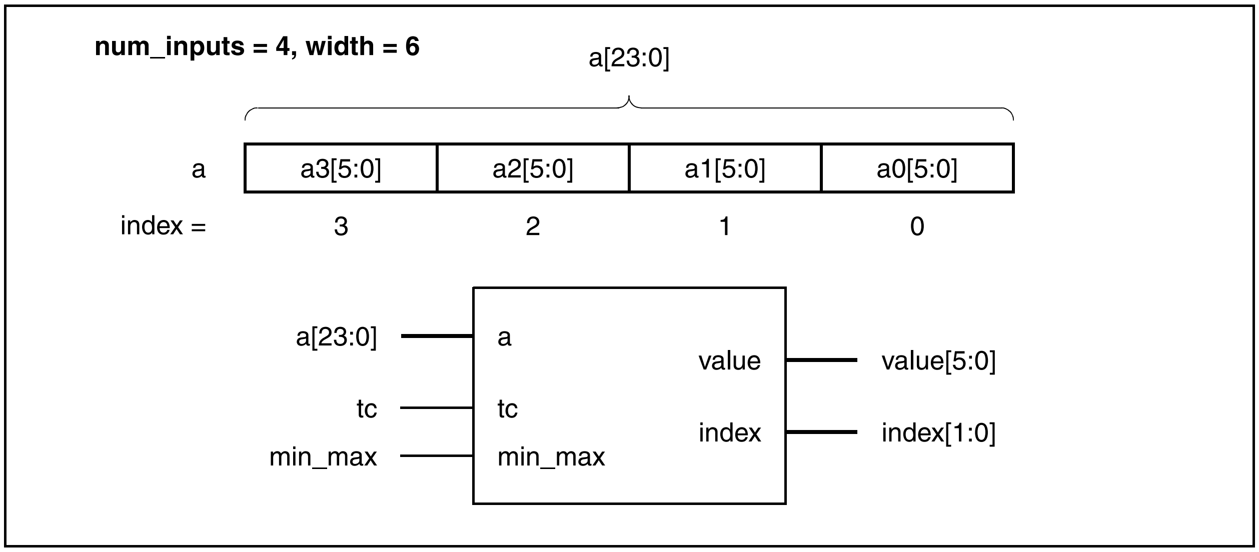
Table 1-5 Functional Description

min_max	a	value
0	a*n* & ... & a1 & a0[a]	min (a*n*, ..., a1, a0)
1	a*n* & ... & a1 & a0	max (a*n*, ..., a1, a0)

a. "&" denotes concatenation in VHDL (Verilog: "{a*n*, ..., a1, a0}")

The `index` output gives the index of the minimum or maximum input as a binary coded number. Therefore, the right-most input within the concatenated input vector has index 0, and the left-most input has index *num_inputs*–1. If multiple inputs are equal and minimum, the lowest of the indices is given; if multiple inputs are equal and maximum, the highest of the indices is given.

Figure 1-1 Functional Operation



Related Topics

- Math – Arithmetic Overview
- DesignWare Building Block IP Documentation Overview

HDL Usage Through Function Inferencing - VHDL

```
library IEEE, DWARE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;
use DWARE.DW_Foundation_arith.all;

-- example: functional inference of 4-input minimum/maximum component
entity DW_minmax_func is
  generic ( wordlength : natural := 8);
  port ( a0, a1, a2, a3 : in  std_logic_vector(wordlength-1 downto 0);
         tc             : in  std_logic;
         max            : in  std_logic;
         val            : out std_logic_vector(wordlength-1 downto 0) );
end DW_minmax_func;

architecture func of DW_minmax_func is
begin

  -- function calls for unsigned/signed minimum/maximum
  -- inputs are concatenated into the first function call operand "a"
  -- the second function call operand is "num_inputs"
  process (a0, a1, a2, a3, max, tc)
  begin
    if max = '0' then
      if tc = '0' then
        val <= std_logic_vector(DWF_min(unsigned(a3 & a2 & a1 & a0), 4));
      else
        val <= std_logic_vector(DWF_min(signed(a3 & a2 & a1 & a0), 4));
      end if;
    else
      if tc = '0' then
        val <= std_logic_vector(DWF_max(unsigned(a3 & a2 & a1 & a0), 4));
      else
        val <= std_logic_vector(DWF_max(signed(a3 & a2 & a1 & a0), 4));
      end if;
    end if;
  end process;

end func;
```

HDL Usage Through Function Inferencing - Verilog

```
// example: functional inference of 4-input minimum/maximum component
module DW_minmax_func (a0, a1, a2, a3, tc, max, val);

  parameter wordlength = 8;

  input  [wordlength-1 : 0] a0, a1, a2, a3;
  input  tc, max;
  output [wordlength-1 : 0] val;

  // pass "width" and "num_inputs" parameters to the inference functions
  parameter width = wordlength;
  parameter num_inputs = 4;

  // Please add search_path = search_path + {synopsys_root + "/dw/sim_ver"}
  // to your .synopsys_dc.setup file (for synthesis) and add
  // +incdir+$SYNOPSYS/dw/sim_ver+ to your verilog simulator command line
  // (for simulation).
  `include "DW_minmax_function.inc"

  // function calls for unsigned/signed minimum/maximum
  // inputs are concatenated into the function call operand
  assign val =
    (max == 1'b0) ? ((tc == 1'b0) ? DWF_min_uns({a3, a2, a1, a0}) :
                                    DWF_min_tc ({a3, a2, a1, a0})) :
                    ((tc == 1'b0) ? DWF_max_uns({a3, a2, a1, a0}) :
                                    DWF_max_tc ({a3, a2, a1, a0}));
endmodule
```

HDL Usage Through Component Instantiation - VHDL

```
library IEEE, DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DW_Foundation_comp_arith.all;

-- example: instantiation of 4-input minimum/maximum component
entity DW_minmax_inst is
  generic ( wordlength : natural := 8);
  port ( a0, a1, a2, a3 : in  std_logic_vector(wordlength-1 downto 0);
         tc             : in  std_logic;
         max            : in  std_logic;
         val            : out std_logic_vector(wordlength-1 downto 0);
         idx            : out std_logic_vector(1 downto 0));
end DW_minmax_inst;

architecture inst of DW_minmax_inst is
  signal a : std_logic_vector(4*wordlength-1 downto 0);
begin

  -- concatenation of inputs
  a <= a3 & a2 & a1 & a0;

  -- instantiation of DW_minmax
  U1  : DW_minmax
    generic map (width => wordlength, num_inputs => 4)
    port map (a => a, tc => tc, min_max => max, value => val, index => idx);
end inst;

-- pragma translate_off
configuration DW_minmax_inst_cfg_inst of DW_minmax_inst is
  for inst
  end for;
end DW_minmax_inst_cfg_inst;
-- pragma translate_on
```

HDL Usage Through Component Instantiation - Verilog

```
// example: instantiation of 4-input minimum/maximum component
module DW_minmax_inst (a0, a1, a2, a3, tc, max, val, idx);

  parameter wordlength = 8;

  input  [wordlength-1 : 0] a0, a1, a2, a3;
  input  tc, max;
  output [wordlength-1 : 0] val;
  output [1 : 0] idx;

  // instantiation of DW_minmax
  // inputs are concatenated into the input vector
  DW_minmax #(wordlength, 4)
    U1 (.a({a3, a2, a1, a0}), .tc(tc), .min_max(max),
        .value(val), .index(idx));

endmodule
```

Copyright Notice and Proprietary Information

© 2018 Synopsys, Inc. All rights reserved. This Synopsys software and all associated documentation are proprietary to Synopsys, Inc. and may only be used pursuant to the terms and conditions of a written license agreement with Synopsys, Inc. All other use, reproduction, modification, or distribution of the Synopsys software or the associated documentation is strictly prohibited.

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html.

All other product or company names may be trademarks of their respective owners.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
690 E. Middlefield Road
Mountain View, CA 94043

www.synopsys.com