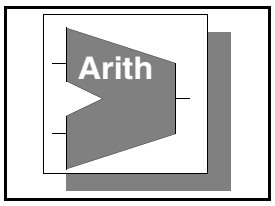
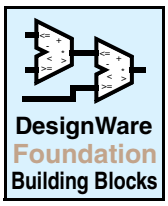




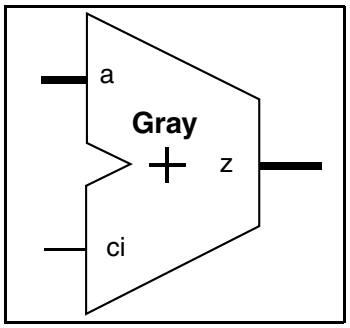
DW_inc_gray

Gray Incrementer

Version, STAR and Download Information: IP Directory

Features and Benefits

- Parameterized word length
- Inferable using a function call



Description

DW_inc_gray adds input at `ci` to Gray coded input `a` and produces the Gray coded output `z`.

Table 1-1 Pin Description

Pin Name	Width	Direction	Function
a	*width* bit(s)	Input	Gray coded input data
ci	1 bit	Input	Carry-in
z	*width* bit(s)	Output	Gray coded output data

Table 1-2 Parameter Description

Parameter	Values	Description
width	≥ 1	Input word length

Table 1-3 Synthesis Implementations[a]

Implementation Name	Function	License Feature Required
rpl	Ripple-carry synthesis model	DesignWare
cla	Carry-lookahead 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_inc_gray_cfg_sim	Design unit name for VHDL simulation
dw/dw01/src/DW_inc_gray_sim.vhd	VHDL simulation model source code
dw/sim_ver/DW_inc_gray.v	Verilog simulation model source code

Table 1-5 Truth Table

a - Gray Input	ci - Carry-in	z - Gray Output
000	0	000
001	0	001
011	0	011
010	0	010
110	0	110
111	0	111
101	0	101
100	0	100
000	1	001
001	1	011
011	1	010
010	1	110
110	1	111
111	1	101
101	1	100
100	1	000

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 DWARE.DW_Foundation_arith.all;

entity DW_inc_gray_func is
  generic (func_width : positive := 8);
  port (func_a  : in  std_logic_vector(func_width-1 downto 0);
        func_ci : in  std_logic;
        z_func  : out std_logic_vector(func_width-1 downto 0));
end DW_inc_gray_func;

architecture func of DW_inc_gray_func is
begin
  -- function inference of DW_inc_gray
  z_func <= DWF_inc_gray (func_a, func_ci);
end func;
```

HDL Usage Through Function Inferencing - Verilog

```
module DW_inc_gray_func (func_a, func_ci, z_func);

  parameter func_width = 8;

  input  [func_width-1 : 0] func_a;
  input                     func_ci;
  output [func_width-1 : 0] z_func;

  // pass "width" parameters to the inference functions
  parameter width = func_width;

  // 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_inc_gray_function.inc"

  // function inference of DW_inc_gray
  assign z_func = DWF_inc_gray (func_a, func_ci);
endmodule
```

HDL Usage Through Component Instantiation - VHDL

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

entity DW_inc_gray_inst is
  generic (inst_width : positive := 8);
  port (inst_a  : in  std_logic_vector(inst_width-1 downto 0);
        inst_ci : in  std_logic;
        z_inst  : out std_logic_vector(inst_width-1 downto 0));
end DW_inc_gray_inst;

architecture inst of DW_inc_gray_inst is
begin
  -- instance of DW_inc_gray
  U1 : DW_inc_gray
    generic map (width => inst_width)
    port map (a  => inst_a,
              ci => inst_ci,
              z  => z_inst);
end inst;

-- pragma translate_off
configuration DW_inc_gray_inst_cfg_inst of DW_inc_gray_inst is
  for inst
  end for;
end DW_inc_gray_inst_cfg_inst;
-- pragma translate_on
```

HDL Usage Through Component Instantiation - Verilog

```
module DW_inc_gray_inst (inst_a, inst_ci, z_inst);

  parameter inst_width = 8;

  input  [inst_width-1 : 0] inst_a;
  input                     inst_ci;
  output [inst_width-1 : 0] z_inst;

  // Please add +incdir+$SYNOPSYS/dw/sim_ver+ to your verilog simulator
  // command line (for simulation).

  // instance of DW_inc_gray
  DW_inc_gray #(inst_width)
    U1 (.a(inst_a),
        .ci(inst_ci),
        .z(z_inst));
endmodule
```

Copyright Notice and Proprietary Information

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