

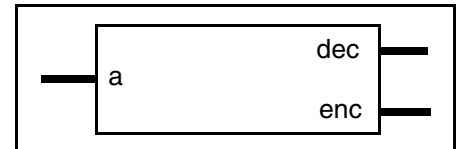
# DW\_lod

## Leading Ones Detector

Version, STAR and Download Information: [IP Directory](#)

### Features and Benefits

- Parameterized word length
- Inferable using a function call



### Description

DW\_lod contains two outputs, dec and enc. The dec output is a decoded one-hot value of the a input vector assuming there is at least one 0 on a. The output enc represents the number of 1s found (from the most significant bit) before the first occurrence of a 0 from the input port a. All lower order bits (to the right) from the first occurrence of the 0 on the a input port are “don't care.” If no 0 is found and only 1s are present, the resulting value of enc is all 1s and of dec is all 0s.

The output port enc width is automatically derived from the input port width parameter, *a\_width*, and is defined as  $\text{ceil}(\log_2[a\_width]) + 1$  as listed in [Table 1-1](#). Output port dec has the same width as the a input.

**Table 1-1 Pin Description**

| Pin Name | Width                               | Direction | Function  |
|----------|-------------------------------------|-----------|---|
| a        | a_width                             | Input     | Input vector  |
| enc      | $\text{ceil}(\log_2[a\_width]) + 1$ | Output    | Number of leading 1s in input a before first 0<br>All 1s if input a is all 1s |
| dec      | a_width                             | Output    | One-hot decode of input a<br>All 0s if input a is all 1s                      |

**Table 1-2 Parameter Description**

| Parameter | Values   | Description             |
|-----------|----------|-------------------------|
| a_width   | $\geq 1$ | Vector width of input a |

**Table 1-3 Synthesis Implementations**

| Implementation Name | Function        | License Feature Required |
|---------------------|-----------------|--------------------------|
| cla                 | Synthesis model | DesignWare               |
| rtl                 | Synthesis model | DesignWare               |

**Table 1-4 Simulation Models**

| Model                      | Function                             |
|----------------------------|--------------------------------------|
| DW01.DW_LOD_CFG_SIM        | Design unit name for VHDL simulation |
| dw/dw01/src/DW_lod_sim.vhd | VHDL simulation model source code    |
| dw/sim_ver/DW_lod.v        | Verilog simulation model source code |

**Table 1-5 Truth Table (*a\_width* = 7, *dec\_width* = 7, *enc\_width* = 4)**

| a(6:0) |   |   |   |   |   |   | enc(3:0) | dec(6:0) |   |   |   |   |   |   |
|--------|---|---|---|---|---|---|----------|----------|---|---|---|---|---|---|
| 1      | 1 | 1 | 1 | 1 | 1 | 1 | 1111     | 0        | 0 | 0 | 0 | 0 | 0 | 0 |
| 1      | 1 | 1 | 1 | 1 | 1 | 0 | 0110     | 0        | 0 | 0 | 0 | 0 | 0 | 1 |
| 1      | 1 | 1 | 1 | 1 | 0 | X | 0101     | 0        | 0 | 0 | 0 | 0 | 1 | 0 |
| 1      | 1 | 1 | 1 | 0 | X | X | 0100     | 0        | 0 | 0 | 0 | 1 | 0 | 0 |
| 1      | 1 | 1 | 0 | X | X | X | 0011     | 0        | 0 | 0 | 1 | 0 | 0 | 0 |
| 1      | 1 | 0 | X | X | X | X | 0010     | 0        | 0 | 1 | 0 | 0 | 0 | 0 |
| 1      | 0 | X | X | X | X | X | 0001     | 0        | 1 | 0 | 0 | 0 | 0 | 0 |
| 0      | X | X | X | X | X | X | 0000     | 1        | 0 | 0 | 0 | 0 | 0 | 0 |

## 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.all;
-- If using numeric_std data types of std_logic_arith, uncomment the
-- following line:
-- use DWARE.DW_Foundation_arith.all;

entity DW_lod_func is
  generic (
    func_a_width : POSITIVE := 8
  );
  port (
    func_a : in std_logic_vector(func_a_width-1 downto 0);
    dec_func : out std_logic_vector(func_a_width-1 downto 0);
    enc_func : out std_logic_vector(bit_width(func_a_width) downto 0)
  );
end DW_lod_func;

architecture func of DW_lod_func is
begin

  -- Inferred function of DW_lod
  dec_func <= DWF_lod(func_a);
  enc_func <= DWF_lod_enc(func_a);

end func;

-- pragma translate_off
configuration DW_lod_func_cfg_func of DW_lod_func is
for func
end for; -- func
end DW_lod_func_cfg_func;
-- pragma translate_on
```

## HDL Usage Through Function Inferencing - Verilog

```
module DW_lod_func( func_a, dec_func, enc_func );

parameter func_a_width = 8;

`define enc_width 4 // ceil(log2(func_a_width))+1

// Passes the widths to DW_lod_function
parameter a_width      = func_a_width;
parameter addr_width   = `enc_width;
`include "DW_lod_function.inc"

input  [func_a_width-1 : 0] func_a;
output [func_a_width-1 : 0] dec_func;
output [`enc_width-1 : 0] enc_func;

    // Function inference of DW_lod and DW_lod_enc
    assign dec_func = DWF_lod(func_a);
    assign enc_func = DWF_lod_enc(func_a);

endmodule
```

## HDL Usage Through Component Instantiation - VHDL

```
library IEEE,DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DW_Foundation_comp.all;
use DWARE.DWpackages.all;
-- If using numeric_std data types of std_logic_arith, uncomment the
-- following line:
-- use DWARE.DW_Foundation_arith.all;

entity DW_lod_inst is
  generic (
    inst_a_width : POSITIVE := 8
  );
  port (
    inst_a : in std_logic_vector(inst_a_width-1 downto 0);
    dec_inst : out std_logic_vector(inst_a_width-1 downto 0);
    enc_inst : out std_logic_vector(bit_width(inst_a_width) downto 0)
  );
end DW_lod_inst;

architecture inst of DW_lod_inst is
begin

  -- Instance of DW_lod
  U1 : DW_lod
    generic map ( a_width => inst_a_width )
    port map ( a => inst_a, dec => dec_inst, enc => enc_inst );
end inst;

-- pragma translate_off
configuration DW_lod_inst_cfg_inst of DW_lod_inst is
  for inst
    end for; -- inst
end DW_lod_inst_cfg_inst;
-- pragma translate_on
```

## HDL Usage Through Component Instantiation - Verilog

```
module DW_lod_inst( inst_a, dec_inst, enc_inst );

parameter a_width      = 8;
parameter enc_width    = 4;  // ceil(log2(a_width))+1

input  [a_width-1 : 0] inst_a;
output [a_width-1 : 0] dec_inst;
output [enc_width-1 : 0] enc_inst;

    // Instance of DW_lod
    DW_lod #(a_width)
        U1 ( .a(inst_a), .dec(dec_inst), .enc(enc_inst) );

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](http://www.synopsys.com)

