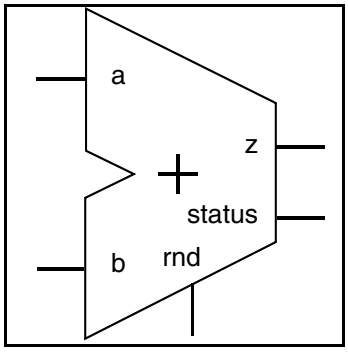
DW_fp_add

Floating-Point Adder

Version, STAR, and myDesignWare Subscriptions: IP Directory

Features and Benefits

Revision History



- The precision format is parameterizable for either IEEE single, double precision, or a user-defined custom format
- Exponents can range from 3 to 31 bits
- Fractional part of the floating-point number can range from 2 to 253 bits
- Configurable to be fully compliant with the IEEE Std 754-1985 standard
- Configurable for NaN representation compatible with the IEEE Std 754-2008 standard (controlled by the *ieee_compliance* parameter)
- DesignWare datapath generator is employed for better timing and area

Description

DW_fp_add is a floating-point component that adds two floating-point values, `a` and `b`, to produce a floating-point sum, `z`.

Component pins are described in Table 1-1 and configuration parameters are described in Table 1-2.

Table 1-1 Pin Description

Pin Name	Width	Direction	Function
a	*sig_width* + *exp_width* +1 bits	Input	Input data
b	*sig_width* + *exp_width* +1 bits	Input	Input data
z	*sig_width* + *exp_width* +1 bits	Output	`a + b`
status	8 bits	Output	Status flags for the result For details, see STATUS Flags in the *Datapath Floating-Point Overview*.
rnd	3 bits	Input	Rounding mode For details, see Rounding Modes in the *Datapath Floating-Point Overview*.

Table 1-2 Parameter Description

Parameter	Values	Description
sig_width	2 to 253 bits Default: 23	Word length of fraction field of floating-point numbers `a`, `b`, and `z`
exp_width	3 to 31 bits Default: 8	Word length of biased exponent of floating-point numbers `a`, `b`, and `z`
ieee_compliance	0, 1, or 3 Default: 0	Level of support for IEEE Std 754 standards: ■ 0: No support for NaNs and denormals; NaNs are considered infinities and denormals are considered zeros ■ 1: Fully compliant with the IEEE Std 754-1985 standard, including support for NaNs and denormals ■ 2: Reserved ■ 3: Fully compliant with the IEEE Std 754-1985 standard plus NaN representation that matches the IEEE Std 754-2008 standard[a] For details, see Compatibility with IEEE Std 754 Standards in the *Datapath Floating-Point Overview*

a. Propagating payload information to the output during the NaN process, which is an optional feature specified in the IEEE Std 754-2008 standard, is not supported.

Table 1-3 Synthesis Implementations

Implementation Name	Function	License Feature Required
rtl	Synthesis model	DesignWare

Table 1-4 Simulation Models

Model	Function
DW02.DW_FP_ADD_CFG_SIM	Design unit name for VHDL simulation
dw/dw02/src/DW_fp_add_sim.vhd	VHDL simulation model source code
dw/sim_ver/DW_fp_add.v	Verilog simulation model source code

Table 1-5 Functional Description

a	b	status	z[a]
a (floating-point)	b (floating-point)	*	a + b (floating-point)

a. The value for any a + b is defined by the rounding mode.

Suppressing Warning Messages During Verilog Simulation

The Verilog simulation model includes macros that allow you to suppress warning messages during simulation.

To suppress all warning messages for all DWBB components, define the DW_SUPPRESS_WARN macro in either of the following ways:

- Specify the Verilog preprocessing macro in Verilog code:

  ```
  `define DW_SUPPRESS_WARN
  ```

- Or, include a command line option to the simulator, such as:

  `+define+DW_SUPPRESS_WARN` (which is used for the Synopsys VCS simulator)

The warning messages for this model include the following:

- If an invalid rounding mode has been detected on `rnd`, the following message is displayed:

  ```
  WARNING: <instance_path>:
      at time = <timestamp>: Illegal rounding mode.
  ```

  To suppress this message, use the DW_SUPPRESS_WARN macro explained earlier.

Related Topics

- Datapath Floating-Point Overview
- DesignWare Building Block IP User Guide

HDL Usage Through Component Instantiation - VHDL

```
library IEEE,DWARE;
use IEEE.std_logic_1164.all;
use DWARE.dw_foundation_comp.all;
-- If using numeric types from std_logic_arith package,
-- comment the preceding line and uncomment the following line:
-- use DWARE.DW_Foundation_comp_arith.all;

entity DW_fp_add_inst is
      generic (
        inst_sig_width : POSITIVE := 23;
        inst_exp_width : POSITIVE := 8;
        inst_ieee_compliance : INTEGER := 0
        );
      port (
       inst_a : in std_logic_vector(inst_sig_width+inst_exp_width downto 0);
       inst_b : in std_logic_vector(inst_sig_width+inst_exp_width downto 0);
       inst_rnd : in std_logic_vector(2 downto 0);
       z_inst : out std_logic_vector(inst_sig_width+inst_exp_width downto 0);
       status_inst : out std_logic_vector(7 downto 0)
        );
    end DW_fp_add_inst;


architecture inst of DW_fp_add_inst is

begin

    -- Instance of DW_fp_add
    U1 : DW_fp_add
    generic map ( sig_width => inst_sig_width,
                  exp_width => inst_exp_width,
                  ieee_compliance => inst_ieee_compliance )
    port map ( a => inst_a, b => inst_b, rnd => inst_rnd, z => z_inst, status =>
status_inst );


end inst;

-- pragma translate_off
configuration DW_fp_add_inst_cfg_inst of DW_fp_add_inst is
 for inst
 end for; -- inst
end DW_fp_add_inst_cfg_inst;
-- pragma translate_on
```

HDL Usage Through Component Instantiation - Verilog

```
module DW_fp_add_inst( inst_a, inst_b, inst_rnd, z_inst, status_inst );

parameter sig_width = 23;
parameter exp_width = 8;
parameter ieee_compliance = 0;


input [sig_width+exp_width : 0] inst_a;
input [sig_width+exp_width : 0] inst_b;
input [2 : 0] inst_rnd;
output [sig_width+exp_width : 0] z_inst;
output [7 : 0] status_inst;

    // Instance of DW_fp_add
    DW_fp_add #(sig_width, exp_width, ieee_compliance)
      U1 ( .a(inst_a), .b(inst_b), .rnd(inst_rnd), .z(z_inst), .status(status_inst) );

endmodule
```

Revision History

For notes about this release, see the *DesignWare Building Block IP Release Notes*.

For lists of both known and fixed issues for this component, refer to the STAR report.

For a version of this datasheet with visible change bars, click here.

Date	Release	Updates
October 2020	DWBB_202009.1	■ For enhanced NaN compatibility with the IEEE Std 754 standards, added a new value for *ieee_compliance* in Table 1-2 on page 2
July 2020	DWBB_201912.5	■ Adjusted the description of the *ieee_compliance* parameter in Table 1-2 on page 2 ■ Added "Suppressing Warning Messages During Verilog Simulation" on page 3 ■ Added this Revision History table and the document links on this page

Copyright Notice and Proprietary Information

© 2022 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.

Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

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.
www.synopsys.com