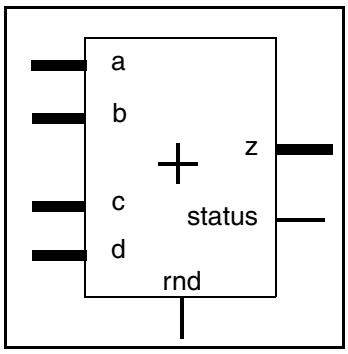
DW_fp_sum4

4-Input Floating-Point Adder

Version, STAR and Download Information: IP Directory

Features and Benefits



- The precision is controlled by parameters, and covers formats in the IEEE standard
- Exponents can range from 3 to 31 bits
- Fractional part of floating-point number ranges from 2 to 253 bits
- A parameter controls the use of denormal values
- Faster than an equivalent logic using three FP adders for some parameter values
- Result is always more accurate than using three FP adders

Description

DW_fp_sum4 is a floating-point component that is capable of adding four floating-point values, `a`, `b`, `c`, and `d`, to produce a floating-point result `z`. This component generates results with more accuracy than independent FP additions, given that only one rounding computation is done, and special conditions on the inputs can be detected and corrected.

The input `rnd` is a 3-bit rounding mode (see Rounding Modes in the *Datapath Floating-Point Overview*) and the output `status` is an 8-bit vector of status flags

For the case of zero result, when *ieee_compliance* = 0, the sign of a zero result is negative when rounding to –infinity (`rnd` = 3), and positive for any other rounding mode. When *ieee_compliance* = 1, the following is the behavior for the sign of zero:

1. When the zero output is a result of the addition of zero inputs (all inputs are zeros), the sign of the zero output depends on the rounding mode:
   - Rounding to –infinity (`rnd` = 3): the output is +0 when all the inputs are +0, otherwise, it is -0.
   - Other rounding modes: the output is -0 when all the inputs are -0, otherwise, it is +0.
2. When the zero output is a result of the addition of non-zero inputs, the output is -0 when the rounding mode is –infinity (`rnd` = 3), otherwise the output is +0.

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

Table 1-1 Pin Description (Continued)

Pin Name	Width	Direction	Function
c	*sig_width* + *exp_width* + 1 bits	Input	Input data
d	*sig_width* + *exp_width* + 1 bits	Input	Input data
z	*sig_width* + *exp_width* + 1 bits	Output	((a + b) + c) + d
status	8 bits	Output	See STATUS Flags in the *Datapath Floating-Point Overview*
rnd	3 bits	Input	Rounding mode

Table 1-2 Parameter Description

Parameter	Values	Description
sig_width	2 to 253 bits	Word length of fraction field of floating-point numbers a, b, c, d, and z
exp_width	3 to 31 bits	Word length of biased exponent of floating-point numbers a, b, c, d, and z
ieee_compliance	0 or 1	When 1, the generated architecture is fully compliant with IEEE 754 standard, including the use of denormals and NaNs.
arch_type	0 or 1	Controls the use of an alternative architecture. Default value is 0 (previous architecture).

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_SUM4_CFG_SIM	Design unit name for VHDL simulation
dw/dw02/src/DW_fp_sum4_sim.vhd	VHDL simulation model source code
dw/sim_ver/DW_fp_sum4.v	Verilog simulation model source code

Table 1-5 Functional Description

a	b	c	d	status[a]	z[b]
a (FP)	b (FP)	c (FP)	d (FP)	*	a + b + c + d (FP)

a. The details of status flags, if used, can be found in Table 9 of the *Datapath Floating-Point Overview*.

b. The actual value of the result is defined by the rounding mode.

The parameters *ieee_compliance* and *arch_type* control the functionality of this component. Different values of *arch_type* result in slightly different numeric behaviors, but in any case, the component is more accurate than the implementation of the same function using basic floating-point adders.

When the parameter *arch_type* = 0, the component provides an output that is independent of the input order. This feature implies that the operation is done as if infinite precision was internally used, and only at the end the rounding operation is performed to obtain the final result.

When *arch_type* = 1, the component is sensitive to the input order, in the same way that a tree of FP adders would be, when configured as ((`a` + `b`) + (`c` + `d`)). Only one rounding operation is performed to compute the output value. The logic produced by this component when *arch_type* = 1 is smaller and faster than when *arch_type* = 0. This configuration is faster or competitive with a network of 3 FP adders, with superior accuracy.

When the parameter *ieee_compliance* = 0, the component considers denormal values as zeros and NaN values as infinity. When *ieee_compliance* = 1, the component operates with denormals and NaNs as described in the IEEE Standard 754.

For more information about the floating-point system defined for all the DW_fp components, including `status` flag bits, and integer and floating-point formats, refer to the *Datapath Floating-Point Overview*.

Related Topics

- Datapath – Floating-Point Overview
- DesignWare Building Block IP Documentation Overview

HDL Usage Through Component Instantiation - VHDL

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

entity DW_fp_sum4_inst is
      generic (
        inst_sig_width : POSITIVE := 23;
        inst_exp_width : POSITIVE := 8;
        inst_ieee_compliance : INTEGER := 0;
        inst_arch_type : 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_c : in std_logic_vector(inst_sig_width+inst_exp_width downto 0);
        inst_d : 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_sum4_inst;


architecture inst of DW_fp_sum4_inst is

begin

    -- Instance of DW_fp_sum4
    U1 : DW_fp_sum4
    generic map (
          sig_width => inst_sig_width,
          exp_width => inst_exp_width,
          ieee_compliance => inst_ieee_compliance,
          arch_type => inst_arch_type
          )
    port map (
          a => inst_a,
          b => inst_b,
          c => inst_c,
          d => inst_d,
          rnd => inst_rnd,
          z => z_inst,
          status => status_inst
          );


end inst;
```

```
-- pragma translate_off
configuration DW_fp_sum4_inst_cfg_inst of DW_fp_sum4_inst is
 for inst
 end for; -- inst
end DW_fp_sum4_inst_cfg_inst;
-- pragma translate_on
```

HDL Usage Through Component Instantiation - Verilog

```
module DW_fp_sum4_inst( inst_a, inst_b, inst_c, inst_d, inst_rnd,
          z_inst, status_inst );

parameter inst_sig_width = 23;
parameter inst_exp_width = 8;
parameter inst_ieee_compliance = 0;
parameter inst_arch_type = 0;


input [inst_sig_width+inst_exp_width : 0] inst_a;
input [inst_sig_width+inst_exp_width : 0] inst_b;
input [inst_sig_width+inst_exp_width : 0] inst_c;
input [inst_sig_width+inst_exp_width : 0] inst_d;
input [2 : 0] inst_rnd;
output [inst_sig_width+inst_exp_width : 0] z_inst;
output [7 : 0] status_inst;

    // Instance of DW_fp_sum4
    DW_fp_sum4 #(inst_sig_width, inst_exp_width, inst_ieee_compliance, inst_arch_type)
U1 (
                .a(inst_a),
                .b(inst_b),
                .c(inst_c),
                .d(inst_d),
                .rnd(inst_rnd),
                .z(z_inst),
                .status(status_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