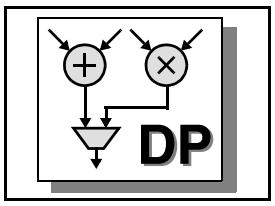

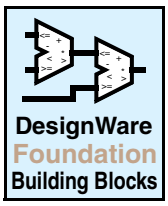

DWF_dp_simd_mult functions

SIMD multiply

Version, STAR and Download Information: IP Directory

Description

The DWF_dp_simd_mult functions implement a configurable SIMD multiplier. They allow you to either multiply arguments `a` and `b` as full-width vectors (for example, one 32-bit multiplication) or to multiply smaller partitions of `a` and `b` using multiple parallel multipliers (for example, two 16-bit multiplications or four 8-bit multiplications). The argument `no_confs` specifies the number of possible configurations, and argument `conf` dynamically selects one of the configurations. Configuration with number `conf` has 2^{conf} partitions of size $width/2^{conf}$. Arguments `a` and `b` and the return value are all either unsigned or signed (two's complement).

Table 1-1 Function Names

Function Name	Description
DWF_dp_simd_mult	VHDL unsigned SIMD multiply
DWF_dp_simd_mult	VHDL signed (two's complement) SIMD multiply
DWF_dp_simd_mult_uns	Verilog unsigned SIMD multiply
DWF_dp_simd_mult_tc	Verilog signed (two's complement) SIMD multiply

Table 1-2 Argument Description

Argument Name	Type	Width / Values	Description
a	Vector [a]	width	Input multiplier
b	Vector [a]	width	Input multiplicand
no_confs	Integer	≥ 2	Number of configurations (VHDL only, constant)
conf	Vector [b]	ceil($\log_2$[no_confs])	Configuration selection: 2^{conf} partitions of size $width/2^{conf}$
DWF_dp_simd_mult	Vector [a]	width	Return value

a. `unsigned` or `signed` in VHDL
b. `std_logic_vector` in VHDL

Table 1-3 Parameter Description (Verilog)

Parameter	Values	Description
width	≥ 2, must be a multiple of $2^{no_confs-1}$	Word length
no_confs	≥ 2	Number of configurations

Verilog Include File: `DW_dp_simd_mult_function.inc`

Functional Description

```
z[2*width-1:0] = DWF_dp_simd_mult (a[width-1:0], b[width-1:0], no_confs,
                 conf[bit_width(no_confs)-1:0])

conf = 0:
  z[2*width-1:0]           = a[width-1:0]           * b[width-1:0]

conf = 1:
  z[2*width-1:2*width/2]   = a[width-1:width/2]     * b[width-1:width/2]
  z[2*width/2-1:0]         = a[width/2-1:0]         * b[width/2-1:0]

conf = 2:
  z[2*width-1:2*width*3/4]   = a[width-1:width*3/4]   * b[width-1:width*3/4]
  z[2*width*3/4-1:2*width/2] = a[width*3/4-1:width/2] * b[width*3/4-1:width/2]
  z[2*width/2-1:2*width/4]   = a[width/2-1:width/4]   * b[width/2-1:width/4]
  z[2*width/4-1:0]           = a[width/4-1:0]         * b[width/4-1:0]

...
```

Example: `width = 32, no_confs = 3`

```
conf = 0:
  z[63: 0] = a[31: 0] + b[31: 0]

conf = 1:
  z[63:32] = a[31:16] + b[31:16]
  z[31: 0] = a[15: 0] + b[15: 0]

conf = 2:
  z[63:48] = a[31:24] + b[31:24]
  z[47:32] = a[23:16] + b[23:16]
  z[31:16] = a[15: 8] + b[15: 8]
  z[15: 0] = a[ 7: 0] + b[ 7: 0]
```

For more information about the DesignWare datapath functions, refer to the topic titled DesignWare Datapath Functions Overview.

Related Topics

- DesignWare Datapath Functions Overview
- DesignWare Building Block IP Documentation Overview

VHDL Example

```
library IEEE, DWARE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
use DWARE.DW_dp_functions.all;
-- DWARE.DW_dp_functions_arith package if IEEE.std_logic_arith is used

entity DWF_dp_simd_mult_test is
  port (op1, op2  : in  signed(31 downto 0);
        config_no : in  std_logic_vector(1 downto 0);
        product   : out signed(63 downto 0));
end DWF_dp_simd_mult_test;

architecture rtl of DWF_dp_simd_mult_test is
begin
  product <= DWF_dp_simd_mult (a => op1, b => op2,
                               no_confs => 3, conf => config_no);
end rtl;
```

Verilog Example

```
module DWF_dp_simd_mult_test (op1, op2, config_no, product);

  input  signed [31:0] op1, op2;
  input         [1:0]  config_no;
  output signed [63:0] product;

  // Passes the parameters to the function
  parameter width    = 32;
  parameter no_confs = 3;

  // add "$SYNOPSYS/dw/sim_ver" to the search path for simulation
  `include "DW_dp_simd_mult_function.inc"

  assign product = DWF_dp_simd_mult_tc (op1, op2, config_no);

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