Arora V Digital Signal Processing (DSP)

User Guide

UG305-1.0E, 04/20/2023

Disclaimer

GOWINSEMI assumes no liability and provides no warranty (either expressed or implied) and is not responsible for any damage incurred to your hardware, software, data, or property resulting from usage of the materials or intellectual property except as outlined in the GOWINSEMI Terms and Conditions of Sale. GOWINSEMI may make changes to this document at any time without prior notice. Anyone relying on this documentation should contact GOWINSEMI for the current documentation and errata.

Revision History

Date	Version	Description
04/20/2023	1.0E	Initial version published.

Contents

List of Figures

List of Tables

1 About This Guide

1.1 Purpose

This manual provides a description of the GOWINSEMI Arora V DSP architecture, signal definition, and user calling methods, etc., to help you make the most use of the Arora V DSP and enhance design efficiency.

1.2 Related Documents

The latest user guides are available on the GOWINSEMI Website. You can find the related documents at www.gowinsemi.com :

- DS981, GW5AT series of FPGA Products Data Sheet
- DS1103, GW5A series of FPGA Products Data Sheet
- DS1104, GW5AST series of FPGA Products Data Sheet
- SUG100, Gowin Software User Guide

1.3 Terminology and Abbreviations

The terminology and abbreviations used in this manual are as shown in Table 1-1.

Table 1-1 Terminology and Abbreviations

Terminology and Abbreviations	Meaning
CFU	Configurable Function Unit
DSP	Digital Signal Processing
FIR	Finite Impulse Response
FFT	Fast Fourier Transformation
MULT	MULT
PADD	PADD
48-bit ALU	48-bit Arithmetic Logic Unit

1.4 Support and Feedback

Gowin Semiconductor provides customers with comprehensive technical support. If you have any questions, comments, or suggestions, please feel free to contact us directly by the following ways.

Website: www.gowinsemi.com

E-mail: support@gowinsemi.com

2 Overview

GOWINSEMI Arora V FPGA products have abundant DSP resources to meet customers' needs for high performance digital signal processing, such as FIR and FFT design. DSP blocks deliver the advantages of stable timing performance, high-usage, and low-power. This manual is designed to help you quickly understand the structure and use of Arora V DSP.

The functions and features of the DSP blocks are as follows:

- Multiplier with three widths: 12X12, 27X18, 27X36
- 26-bit pre-adder
- 48-bit ALU
- Supports Shift function
- Multiple multipliers can achieve multiply of larger data widths by cascading
- Supports the accumulation and multiply-add functions of the 27X18 multiplier
- Supports the accumulation after summation function of two 12X12 multipliers
- Supports the pipeline and bypass functions of registers
- All operands for arithmetic operation are signed numbers

3 DSP Architecture

The DSP of GOWINSEMI Arora V FPGA products are distributed in FPGA arrays in the form of rows. And the DSP includes MULT, PADD, and 48-bit ALU function modules, which can implement multiplication, pre-addition, accumulation, shift functions, etc. Each DSP occupies 3 CFUs, and each DSP has 2 independent clock signals, 2 independent clock enable signals, and 2 independent reset signals. Register hierarchy has up to 4 levels, i.e. input reg, pipe reg, out reg, and fb preg.

Figure 3-1 DSP Architecture

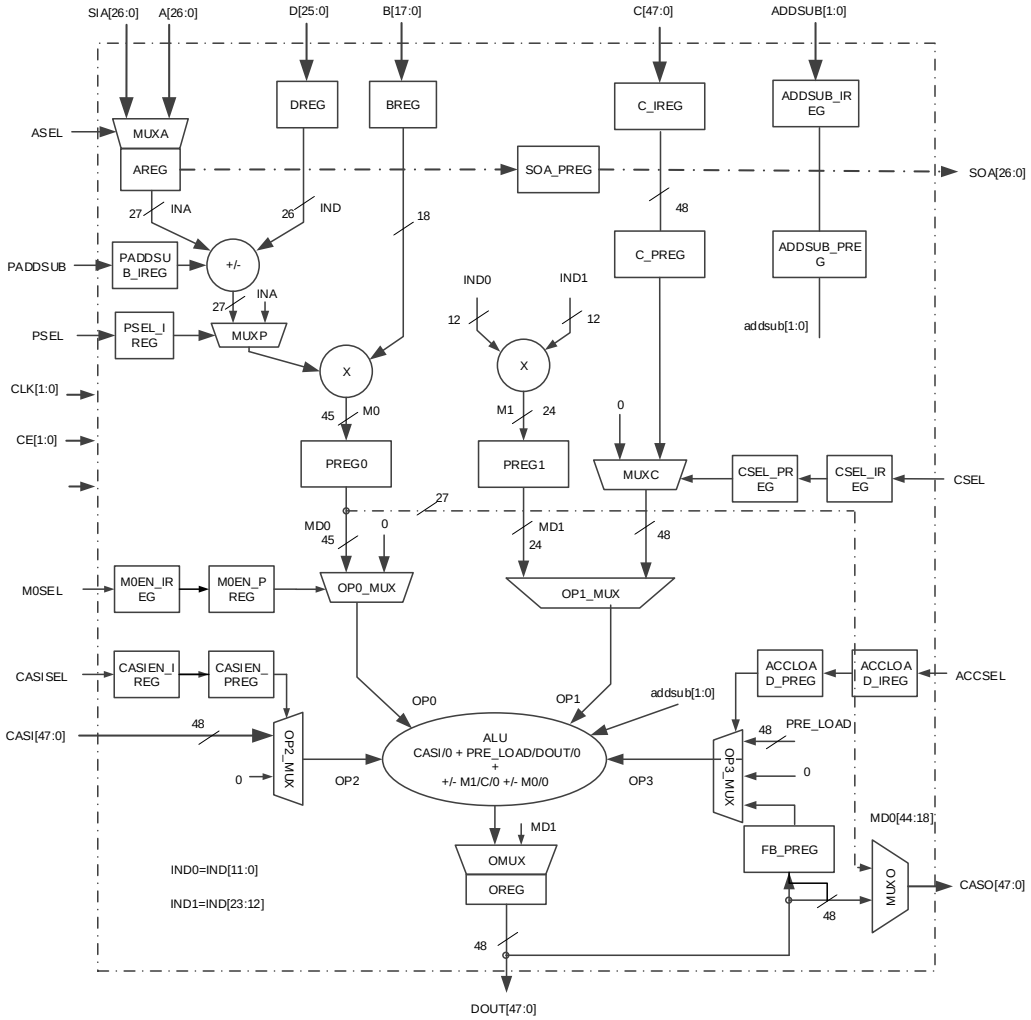


Table 3-1 presents DSP ports description. The internal staged registers are as shown in Table 3-2. In addition, input signals CLK, CE, and RESET are used to control the registers.

Table 3-1 Description of the DSP Ports

Port Name	I/O	Description
A[26:0]	I	27-bit data input A
B[17:0]	I	18-bit data input B
C[47:0]	I	48-bit data input C
D[25:0]	I	26-bit data input D
SIA[26:0]	I	Shift data input A, used for cascade connection. The input signal “SIA” is directly connected to the output signal “SOA” of the previously adjacent DSP.

Port Name	I/O	Description
CASI[47:0]	I	CASO from previous DSP, 48-bit ALU cascading input for cascade connection.
CASISEL	I	48-bit ALU input CASI/O control signal
ASEL	I	Input A selection of pre-adder
PSEL	I	Input A selection of multiplier
PADDSUB	I	Operation control signal of pre-adder for pre-adder logic addition/subtraction selection
CLK[1:0]	I	Clock input
CE[1:0]	I	Clock enable signal, active-high.
RESET[1:0]	I	Reset signal, supports synchronous/asynchronous mode, active-high
ADDSUB[1:0]	I	Operation control signal of 48-bit ALU for addition/subtraction selection of M0/0, M1/C/0
CSEL	I	48-bit ALU input C/0 control signal
ACCSEL	I	48-bit ALU input PRE_LOAD/DOUT control signal
M0SEL	I	48-bit ALU input M0/0 control signal
SOA[26:0]	O	Shift data output A
DOUT[47:0]	O	DSP output data
CASO[47:0]	O	48-bit ALU output to next DSP block for cascade connection

Table 3-2 DSP Internal Registers Description

Register	Description and Associated Attributes
AREG	A input register
BREG	B input register
C_IREG	C input register
DREG	D input register
ADDSUB_IREG	ADDSUB input register
PADDSUB_IREG	PADDSUB input register
PSEL_IREG	PSEL input register
M0SEL_IREG	M0SEL input register
CASISEL_IREG	CASISEL input register
CSEL_IREG	CSEL input register

Register	Description and Associated Attributes
ACCSEL_IREG	ACCSEL input register
C_PREG	C pipeline input register
ADDSUB_PREG	ADDSUB pipeline input register
M0SEL_PREG	M0SEL pipeline input register
CASISEL_PREG	CASISEL pipeline input register
CSEL_PREG	CSEL pipeline input register
ACCSEL_PREG	ACCSEL pipeline input register
OREG	DOUT output register
PREG0	Left multiplier pipeline output register
PREG1	Right multiplier pipeline output register
FB_PREG	Feedback output pipeline register
SOA_PREG	SOA pipeline shift output register

4 DSP Primitive

4.1 MULT

MULT is the multiplier unit of DSP. The multiplier input signal is defined as A and B; The multiplier output signal is defined as DOUT, which can implement multiplication:

$$DOUT = A * B$$

$$DOUT = (A \pm D) * B$$

Each DSP macro unit has two multipliers to implement multiplication. To meet the needs of different multiplication bit widths, MULT mode can be configured to 12x12 and 27x36 multipliers depending on the data width, corresponding to MULT12x12 and MULT27x36 primitives, respectively. The 27x36 multiplier requires 2 DSP blocks to configure.

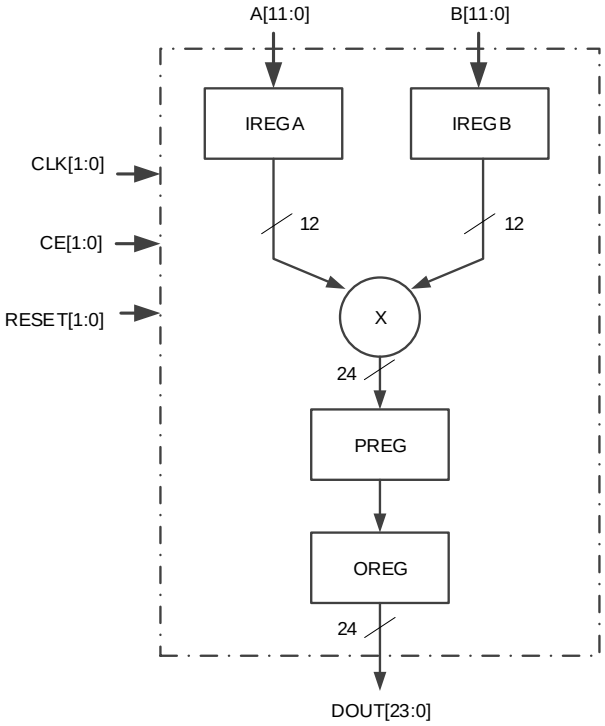
4.1.1 MULT12X12

Primitive Introduction

MULT12X12 (12x12 Multiplier) implements 12-bit multiplication.

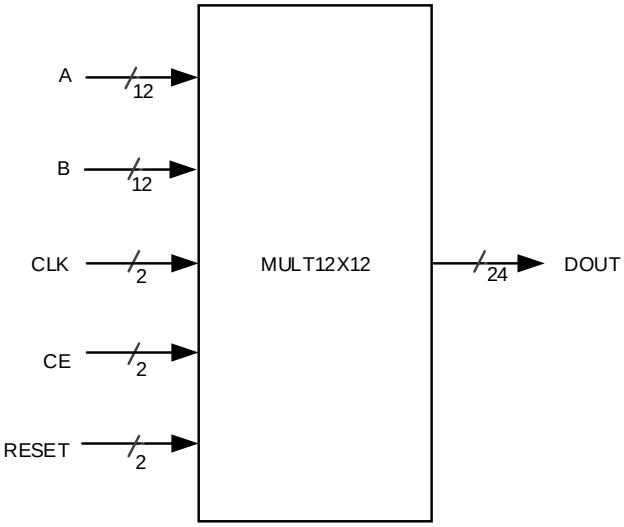
Logic Structure Diagram

Figure 4-1 MULT12X12 Logic Structure Diagram



Port Diagram

Figure 4-2 MULT12X12 Port Diagram



Port Description

Table 4-1 MULT12X12 Port Description

Port	I/O	Description
A[11:0]	Input	12-bit data input signal A
B[11:0]	Input	12-bit data input signal B
CLK[1:0]	Input	Clock input signal
CE[1:0]	Input	Clock enable signal, active-high
RESET[1:0]	Input	Reset input signal, active-high
DOUT[23:0]	Output	Data output signal

Parameter Description

Table 4-2 MULT12X12 Parameter Description

Parameter	Range	Default Value	Description
AREG_CLK	BYPASS, CLK0, CLK1	BYPASS	A input register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
AREG_CE	CE0, CE1	CE0	A input register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
AREG_RESET	RESET0, RESET1	RESET0	A input register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
BREG_CLK	BYPASS, CLK0, CLK1	BYPASS	B input register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].

Parameter	Range	Default Value	Description
BREG_CE	CE0, CE1	CE0	B input register clock Enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
BREG_RESET	RESET0, RESET1	RESET0	B input register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Pipeline register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
PREG_CE	CE0, CE1	CE0	Pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
PREG_RESET	RESET0, RESET1	RESET0	Pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
OREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Output register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
OREG_CE	CE0, CE1	CE0	Output register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].

Parameter	Range	Default Value	Description
OREG_RESET	RESET0, RESET1	RESET0	Output register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
MULT_RESET_MODE	SYNC, ASYNC	SYNC	Reset configuration mode ● SYNC: synchronized reset ● ASYNC: asynchronous reset

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 5 IP Generation.

Verilog Instantiation:

```
MULT12X12 mult12x12_inst (
    .DOUT(dout),
    .A(a),
    .B(b),
    .CLK(clk),
    .CE(ce),
    .RESET(reset)
);

defparam mult12x12_inst.AREG_CLK = "BYPASS";
defparam mult12x12_inst.AREG_CE = "CE0";
defparam mult12x12_inst.AREG_RESET = "RESET0";
defparam mult12x12_inst.BREG_CLK = "BYPASS";
defparam mult12x12_inst.BREG_CE = "CE0";
defparam mult12x12_inst.BREG_RESET = "RESET0";
defparam mult12x12_inst.PREG_CLK = "BYPASS";
defparam mult12x12_inst.PREG_CE = "CE0";
defparam mult12x12_inst.PREG_RESET = "RESET0";
```

```
defparam mult12x12_inst.OREG_CLK = "BYPASS";
defparam mult12x12_inst.OREG_CE = "CE0";
defparam mult12x12_inst.OREG_RESET = "RESET0";
defparam mult12x12_inst.MULT_RESET_MODE="SYNC";
```

Vhdl Instantiation:

```
COMPONENT MULT12X12
    GENERIC (
            AREG_CLK : string := "BYPASS";
            AREG_CE : string := "CE0";
            AREG_RESET : string := "RESET0";
            BREG_CLK : string := "BYPASS";
            BREG_CE : string := "CE0";
            BREG_RESET : string := "RESET0";
            PREG_CLK : string := "BYPASS";
            PREG_CE : string := "CE0";
            PREG_RESET : string := "RESET0";
            OREG_CLK : string := "BYPASS";
            OREG_CE : string := "CE0";
            OREG_RESET : string := "RESET0";
            MULT_RESET_MODE:string:="SYNC"
    );
    PORT(
        DOUT: out std_logic_vector(23 downto 0);
        A: in std_logic_vector(11 downto 0);
        B: in std_logic_vector(11 downto 0);
        CLK: in std_logic_vector(1 downto 0);
        CE: in std_logic_vector(1 downto 0);
        RESET: in std_logic_vector(1 downto 0)
    );
end COMPONENT;
mult12x12_inst: MULT12X12
```

```
GENERIC MAP(
    AREG_CLK => "BYPASS",
    AREG_CE => "CE0",
    AREG_RESET => "RESET0",
    BREG_CLK => "BYPASS",
    BREG_CE => "CE0",
    BREG_RESET => "RESET0",
    PREG_CLK => "BYPASS",
    PREG_CE => "CE0",
    PREG_RESET => "RESET0",
    OREG_CLK => "BYPASS",
    OREG_CE => "CE0",
    OREG_RESET => "RESET0",
    MULT_RESET_MODE=>"SYNC"
)
PORT MAP (
    DOUT => dout,
    A=>a,
    B=>b,
    CLK => CLK_i,
    CE => CE_i,
    RESET => RESET_i
);
```

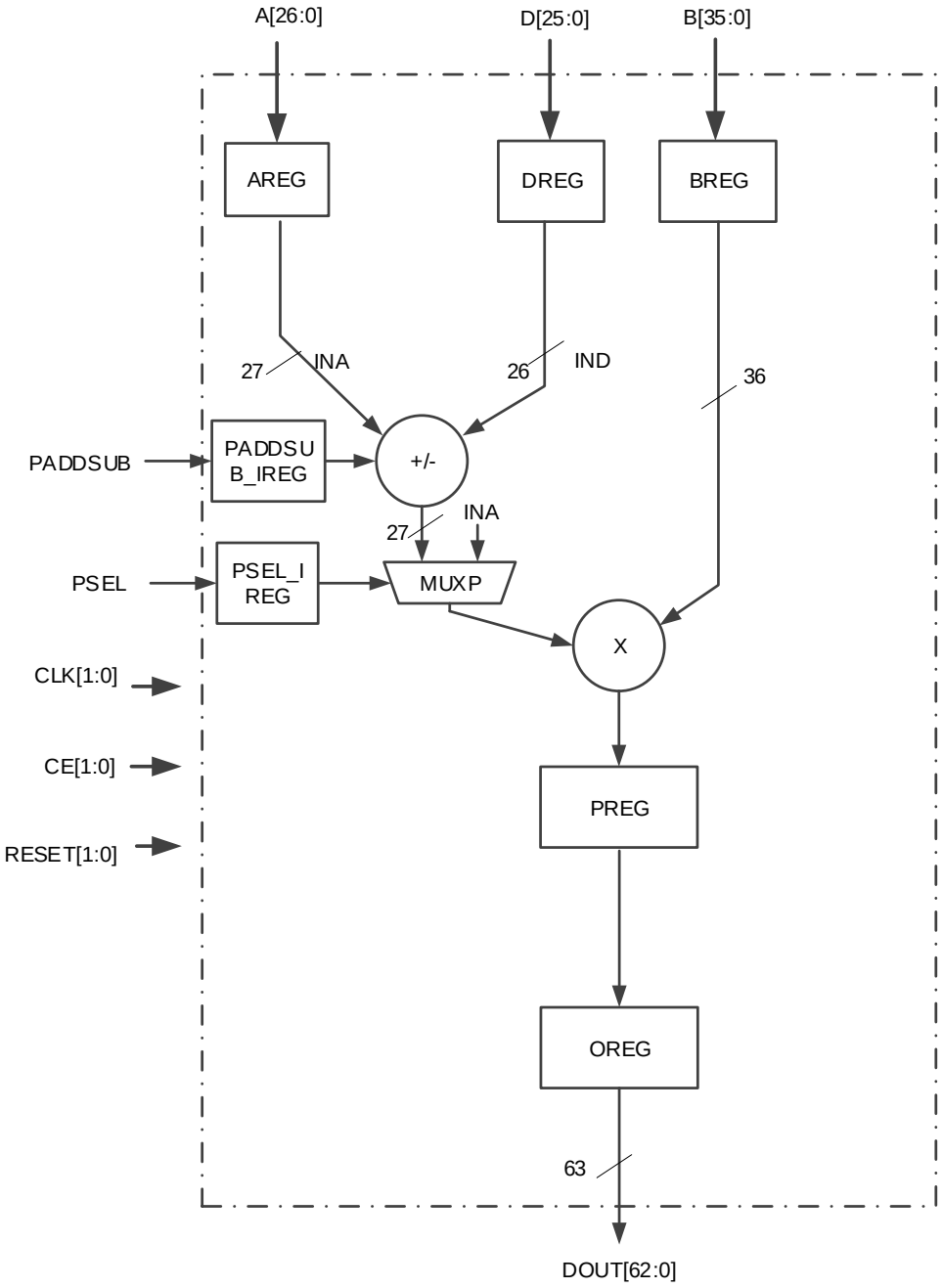
4.1.2 MULT27X36

Primitive Introduction

MULT27X36 (27x36 Multiplier) implements 27bit x 36bit multiplication.

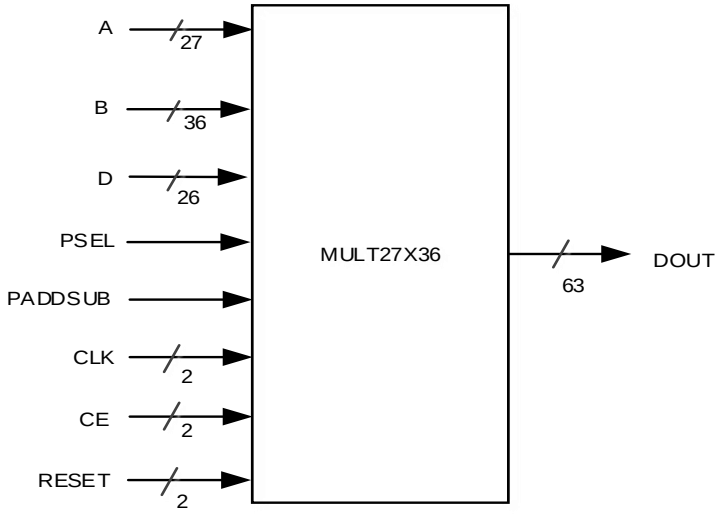
Logic Structure Diagram

Figure 4-3 MULT27X36 Logic Structure Diagram



Port Diagram

Figure 4-4 MULT27X36 Port Diagram



Port Description

Table 4-3 MULT27X36 Port Description

Port	I/O	Description
A[26:0]	Input	27-bit data input signal A
B[35:0]	Input	36-bit data input signal B
D[25:0]	Input	26-bit data input signal D
PSEL	Input	A input source selection of multiplier
PADDSUB	Input	Operation control signal of pre-adder for pre-adder logic addition/subtraction selection
CLK[1:0]	Input	Clock input signal
CE[1:0]	Input	Clock enable signal, active-high
RESET[1:0]	Input	Reset input signal, active-high
DOUT[62:0]	Output	Data output signal

Parameter Description

Table 4-4 MULT12X12 Parameter Description

Parameter	Range	Default Value	Description
AREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input A register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register control signal clk is from CLK[0]; ● CLK1: Register mode, register control signal clk is from CLK[1].
AREG_CE	CE0, CE1	CE0	Input A register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
AREG_RESET	RESET0, RESET1	RESET0	Input A register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
BREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input B register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register control signal clk is from CLK[0]; ● CLK1: Register mode, register control signal clk is from CLK[1].
BREG_CE	CE0, CE1	CE0	Input B register clock Enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
BREG_RESET	RESET0, RESET1	RESET0	Input B register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
DREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input D register clock control signal ● BYPASS: bypass mode;

Parameter	Range	Default Value	Description
			● CLK0: Register mode, register control signal clk is from CLK[0]; ● CLK1: Register mode, register control signal clk is from CLK[1].
DREG_CE	CE0, CE1	CE0	Input D register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
DREG_RESET	RESET0, RESET1	RESET0	Input D register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
PADDSUB_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input PADDSUB register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register control signal clk is from CLK[0]; ● CLK1: Register mode, register control signal clk is from CLK[1].
PADDSUB_IREG_CE	CE0, CE1	CE0	Input PADDSUB register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
PADDSUB_IREG_RESET	RESET0, RESET1	RESET0	Input PADDSUB register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
PSEL_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input PSEL register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register control signal clk is from CLK[0]; ● CLK1: Register mode, register control signal clk is from CLK[1].

Parameter	Range	Default Value	Description
PSEL_IREG_CE	CE0, CE1	CE0	Input PSEL register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
PSEL_IREG_RESET	RESET0, RESET1	RESET0	Input PSEL register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Mult Pipeline register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register control signal clk is from CLK[0]; ● CLK1: Register mode, register control signal clk is from CLK[1].
PREG_CE	CE0, CE1	CE0	Mult Pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
PREG_RESET	RESET0, RESET1	RESET0	Mult Pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
OREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Output register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register control signal clk is from CLK[0]; ● CLK1: Register mode, register control signal clk is from CLK[1].
OREG_CE	CE0, CE1	CE0	Output register clock enable control signal ● CE0: Register clock enable control signal is from CE[0];

Parameter	Range	Default Value	Description
			● CE1: Register clock enable control signal is from CE[1].
OREG_RESET	RESET0, RESET1	RESET0	Output register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
MULT_RESET_MODE	SYNC, ASYNC	SYNC	Reset configuration mode
P_SEL	1'b0, 1'b1	1'b0	Static control to select to directly connect A or A +/- D ● 1'b0: select to directly connect INA ● 1'b1: select pre-adder
DYN_P_SEL	FALSE, TRUE	FALSE	Dynamic control selection, INA or INA +/- D ● FALSE: P_SEL static control mult0 to select INA or INA +/- D ● TRUE: Input PSEL dynamic control mult0 to select INA or INA +/- D
P_ADDSUB	1'b0, 1'b1	1'b0	Static control pre-adder to select addition/subtraction ● 1'b0: addition ● 1'b1: subtraction
DYN_P_ADDSUB	FALSE, TRUE	FALSE	Dynamic control pre-adder to select addition/subtraction ● FALSE: P_ADDSUB static control pre-adder to select addition/subtraction ● TRUE: Input PSEL to dynamic control pre-adder to select addition/subtraction

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 5 IP Generation.

Verilog Instantiation:

```
MULT27X36 mult27x36_inst (
    .DOUT(dout),
```

```
        .A({gw_gnd,a[25:0]}),
        .B(b),
        .D(d),
        .PSEL(gw_gnd),
        .PADDSUB(gw_gnd),
        .CLK({gw_gnd,clk}),
        .CE({gw_gnd,ce}),
        .RESET({gw_gnd,reset})
);

defparam mult27x36_inst.AREG_CLK = "CLK0";
defparam mult27x36_inst.AREG_CE = "CE0";
defparam mult27x36_inst.AREG_RESET = "RESET0";
defparam mult27x36_inst.BREG_CLK = "CLK0";
defparam mult27x36_inst.BREG_CE = "CE0";
defparam mult27x36_inst.BREG_RESET = "RESET0";
defparam mult27x36_inst.DREG_CLK = "CLK0";
defparam mult27x36_inst.DREG_CE = "CE0";
defparam mult27x36_inst.DREG_RESET = "RESET0";
defparam mult27x36_inst.PADDSUB_IREG_CLK = "BYPASS";
defparam mult27x36_inst.PADDSUB_IREG_CE = "CE0";
defparam mult27x36_inst.PADDSUB_IREG_RESET = "RESET0";
defparam mult27x36_inst.PREG_CLK = "BYPASS";
defparam mult27x36_inst.PREG_CE = "CE0";
defparam mult27x36_inst.PREG_RESET = "RESET0";
defparam mult27x36_inst.PSEL_IREG_CLK = "BYPASS";
defparam mult27x36_inst.PSEL_IREG_CE = "CE0";
defparam mult27x36_inst.PSEL_IREG_RESET = "RESET0";
defparam mult27x36_inst.OREG_CLK = "CLK0";
defparam mult27x36_inst.OREG_CE = "CE0";
defparam mult27x36_inst.OREG_RESET = "RESET0";
```

```
defparam mult27x36_inst.MULT_RESET_MODE="SYNC";
defparam mult27x36_inst.DYN_P_SEL = "FALSE";
defparam mult27x36_inst.P_SEL = "1'b1";
defparam mult27x36_inst.DYN_P_ADDSUB = "FALSE";
defparam mult27x36_inst.P_ADDSUB = "1'b0";
```

Vhdl Instantiation:

```
COMPONENT MULT27X36
        GENERIC (AREG_CLK:string:="CLK0";
                AREG_CE:string:="CE0";
                AREG_RESET:string:="RESET0";
                BREG_CLK:string:="CLK0";
                BREG_CE:string:="CE0";
                BREG_RESET:string:="RESET0";
                DREG_CLK:string:="CLK0";
                DREG_CE:string:="CE0";
                DREG_RESET:string:="RESET0";
                PADDSUB_IREG_CLK:string:="CLK0";
                PADDSUB_IREG_CE:string:="CE0";
                PADDSUB_IREG_RESET:string:="RESET0";
                PREG_CLK:string:="CLK0";
                PREG_CE:string:="CE0";
                PREG_RESET:string:="RESET0";
                PSEL_IREG_CLK:string:="CLK0";
                PSEL_IREG_CE:string:="CE0";
                PSEL_IREG_RESET:string:="RESET0";
                OREG_CLK:string:="CLK0";
                OREG_CE:string:="CE0";
                OREG_RESET:string:="RESET0";
                MULT_RESET_MODE:string:="ASYNC";
                DYN_P_SEL:string:="FALSE";
                P_SEL:bit:='0';
```

```
                    DYN_P_ADDSUB:string:="FALSE";
                    P_ADDSUB:bit:='0';
                );
                PORT(
                    DOUT:OUT std_logic_vector(62 downto 0);
                    A:IN std_logic_vector(26 downto 0);
                    B:IN std_logic_vector(35 downto 0);
                    D:IN std_logic_vector(25 downto 0);
                    PSEL:IN std_logic;
                    PADDSUB:IN std_logic;
                    CLK:IN std_logic_vector(1 downto 0);
                    CE:IN std_logic_vector(1 downto 0);
                    RESET:IN std_logic_vector(1 downto 0)
                );
END COMPONENT;
uut:MULT27X36
        GENERIC MAP (AREG_CLK=>"CLK0",
                    AREG_CE=>"CE0",
                    AREG_RESET=>"RESET0",

                    BREG_CLK=>"CLK0",
                    BREG_CE=>"CE0",
                    BREG_RESET=>"RESET0",
                    DREG_CLK=>"CLK0",
                    DREG_CE=>"CE0",
                    DREG_RESET=>"RESET0",
                    PADDSUB_IREG_CLK=>"CLK0",
                    PADDSUB_IREG_CE=>"CE0",
                    PADDSUB_IREG_RESET=>"RESET0",
                    PREG_CLK=>"CLK0",
                    PREG_CE=>"CE0",
```

```
                PREG_RESET=>"RESET0",
                PSEL_IREG_CLK=>"CLK0",
                PSEL_IREG_CE=>"CE0",
                PSEL_IREG_RESET=>"RESET0",
                OREG_CLK=>"CLK0",
                OREG_CE=>"CE0",
                OREG_RESET=>"RESET0",
                MULT_RESET_MODE=>"ASYNC",
                DYN_P_SEL=>"FALSE",
                P_SEL=>'1',
                DYN_P_ADDSUB=>"FALSE",
                P_ADDSUB=>'0'
    )
    PORT MAP (
        DOUT => dout,
        A=>A_i,
        B=>b,
        D=>d,
        PSEL=>gw_gnd,
        PADDSUB=>gw_gnd,
        CLK=>clk,
        CE=>ce,
        RESET=>reset
    );
```

4.2 MULTALU

4.2.1 MULTALU27X18

The MULTALU mode implements a multiplier output after 48-bit ALU operations, corresponding to MULTALU27X18 primitive. The MULTALU27X18 has sixteen operation modes:

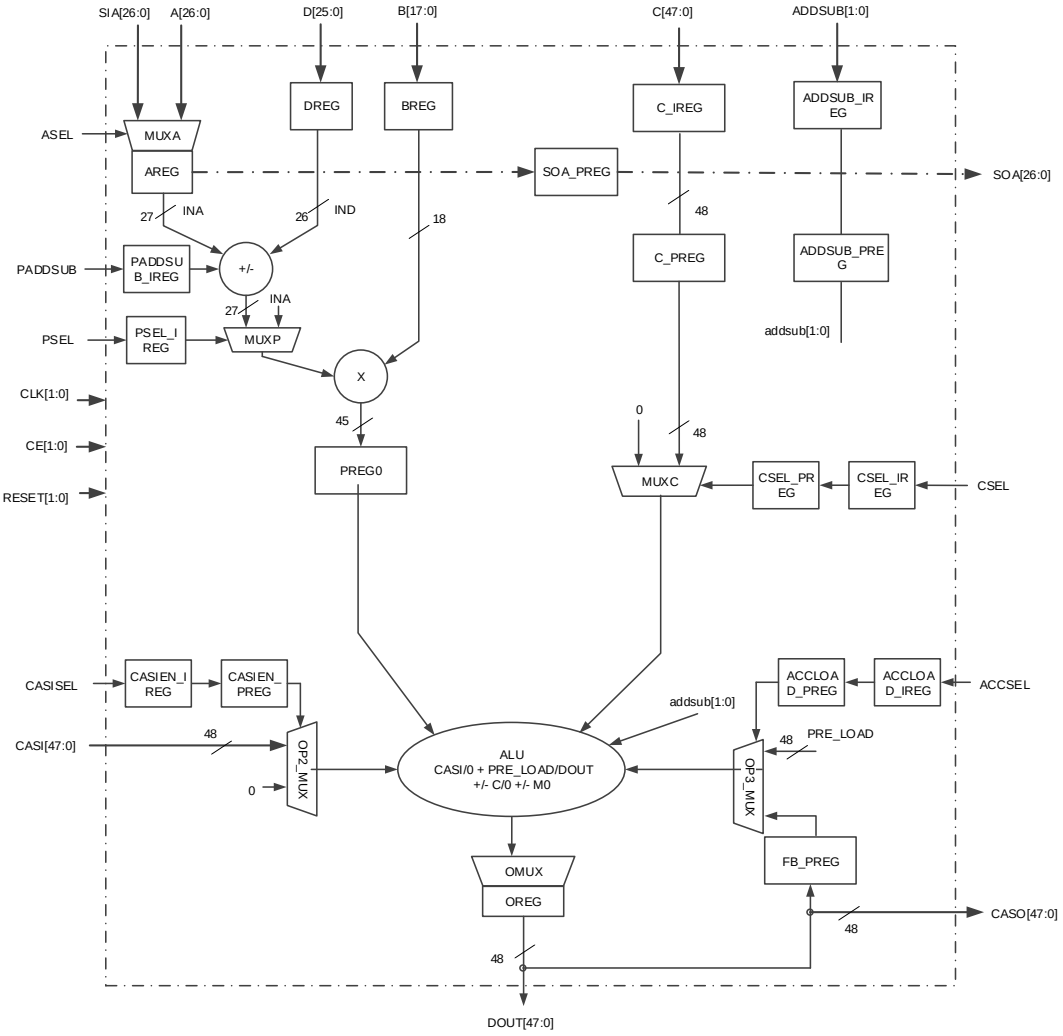
$$DOUT = \pm(A * B)$$
$$DOUT = \pm(A * B) \pm C$$
$$DOUT = \pm(A * B) + DOUT$$
$$DOUT = \pm(A * B) \pm C + DOUT$$
$$DOUT = \pm\big((A \pm D) * B\big)$$
$$DOUT = \pm\big((A \pm D) * B\big) \pm C$$
$$DOUT = \pm\big((A \pm D) * B\big) + DOUT$$
$$DOUT = \pm\big((A \pm D) * B\big) \pm C + DOUT$$
$$DOUT = \pm(A * B) + CASI$$
$$DOUT = \pm(A * B) + CASI \pm C$$
$$DOUT = \pm(A * B) + CASI + DOUT$$
$$DOUT = \pm(A * B) + CASI + DOUT \pm C$$
$$DOUT = \pm(SIA * B)$$
$$DOUT = \pm(SIA * B) \pm C$$
$$DOUT = \pm(SIA * B) + DOUT$$
$$DOUT = \pm(SIA * B) + DOUT \pm C$$

Primitive Introduction

The MULTALU27X18 (27x18 Multiplier with ALU) implements multiplication, multiply-add, accumulation, multiply-accumulate, shift based on multiplication/multiply-add/accumulation/multiply-accumulate, cascade based on multiplication/multiply-add/accumulation/multiply-accumulate, pre-addition and pre-subtraction based on multiplication/multiply-add/accumulation/multiply-accumulate, etc.

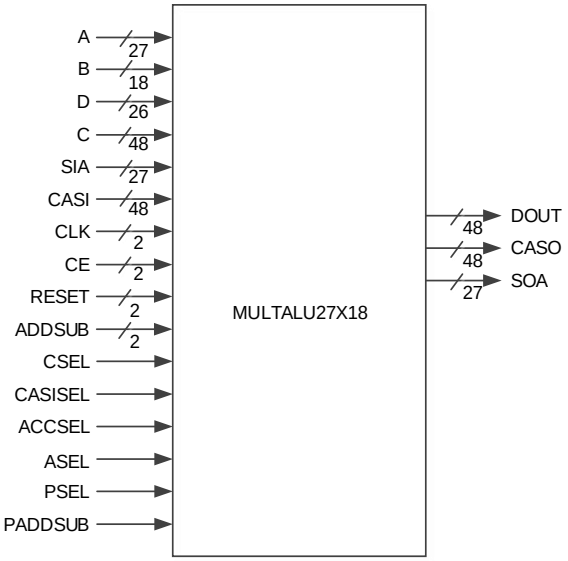
Logic Structure Diagram

Figure 4-5 MULT27X18 Logic Structure Diagram



Port Diagram

Figure 4-6 MULT27X18 Port Diagram



Port Description

Table 4-5 MULT27X18 Port Description

Port	I/O	Description
A[26:0]	Input	27-bit data input signal A
B[17:0]	Input	18-bit data input signal B
D[25:0]	Input	26-bit pre-adder data input signal D
C[47:0]	Input	48-bit ALU data input signal C
CASI[47:0]	Input	48-bit ALU cascade input signal
SIA[26:0]	Input	27-bit shift data input signal for shifting The input signal “SIA” is directly connected to the output signal “SOA” of the previously adjacent DSP.
CLK[1:0]	Input	Clock input signal
CE[1:0]	Input	Clock enable signal, active-high
RESET[1:0]	Input	Reset input signal, active-high
ADDSUB[1:0]	Input	Dynamic addition/subtraction control input signal
CSEL	Input	48-bit ALU input C, the control selection input signal of 0
CASISEL	Input	48-bit ALU input CASI, the control selection input signal of 0
ACCSEL	Input	48-bit ALU input DOUT, the control selection input signal of PRE_LOAD
ASEL	Input	A of pre-adder or multiplier, the control selection input signal of SIA
PSEL	Input	INA of multiplier, the control selection input signal of INA +/- D
PADDSUB	Input	Operation control signal of pre-adder for pre-adder logic addition/subtraction selection
SOA[26:0]	Output	Shift data output signal
DOUT[47:0]	Output	Data output signal
CASO[47:0]	Output	48-bit cascading output signal

Parameter Description

Table 4-6 MULTALU27X18 Parameter Description

Parameter	Range	Default Value	Description
AREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input A (A or SIA) register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
AREG_CE	CE0, CE1	CE0	Input A (A or SIA) register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
AREG_RESET	RESET0, RESET1	RESET0	Input A (A or SIA) register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
BREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input B register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
BREG_CE	CE0, CE1	CE0	Input B register clock Enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
BREG_RESET	RESET0, RESET1	RESET0	Input B register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control

Parameter	Range	Default Value	Description
			signal is from RESET[1].
DREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input D register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
DREG_CE	CE0, CE1	CE0	Input D register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
DREG_RESET	RESET0, RESET1	RESET0	Input D register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
C_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input C register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
C_IREG_CE	CE0, CE1	CE0	Input C register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
C_IREG_RESET	RESET0, RESET1	RESET0	Input C register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
C_PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input C pipeline register clock control signal

Parameter	Range	Default Value	Description
			● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
C_PREG_CE	CE0, CE1	CE0	Input C pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
C_PREG_RESET	RESET0, RESET1	RESET0	Input C pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
ADDSUB0_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ADDSUB[0] register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
ADDSUB0_IREG_CE	CE0, CE1	CE0	Input ADDSUB[0] register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
ADDSUB0_IREG_RESET	RESET0, RESET1	RESET0	Input ADDSUB[0] register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
ADDSUB1_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ADDSUB[1] register clock control signal

Parameter	Range	Default Value	Description
			● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
ADDSUB1_IREG_CE	CE0, CE1	CE0	Input ADDSUB[1] register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
ADDSUB1_IREG_RESET	RESET0, RESET1	RESET0	Input ADDSUB1[1] register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
ADDSUB0_PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ADDSUB[0] pipeline register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
ADDSUB0_PREG_CE	CE0, CE1	CE0	Input ADDSUB[0] pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
ADDSUB0_PREG_RESET	RESET0, RESET1	RESET0	Input ADDSUB[0] pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
ADDSUB1_PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ADDSUB[1] pipeline register clock control signal

Parameter	Range	Default Value	Description
			● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
ADDSUB1_PREG_CE	CE0, CE1	CE0	Input ADDSUB[1] pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
ADDSUB1_PREG_RESET	RESET0, RESET1	RESET0	Input ADDSUB[1] pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
PADDSUB_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input PADDSUB register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
PADDSUB_IREG_CE	CE0, CE1	CE0	Input PADDSUB register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
PADDSUB_IREG_RESET	RESET0, RESET1	RESET0	Input PADDSUB register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
PSEL_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input PSEL register clock control signal

Parameter	Range	Default Value	Description
			● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
PSEL_IREG_CE	CE0, CE1	CE0	Input PSEL register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
PSEL_IREG_RESET	RESET0, RESET1	RESET0	Input PSEL register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
CSEL_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input CSEL register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
CSEL_IREG_CE	CE0, CE1	CE0	Input CSEL register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
CSEL_IREG_RESET	RESET0, RESET1	RESET0	Input CSEL register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
CSEL_PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input CSEL pipeline register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock

Parameter	Range	Default Value	Description
			control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
CSEL_PREG_CE	CE0, CE1	CE0	Input CSEL pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
CSEL_PREG_RESET	RESET0, RESET1	RESET0	Input CSEL pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
CASISEL_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input CASISEL register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
CASISEL_IREG_CE	CE0, CE1	CE0	Input CASISEL register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
CASISEL_IREG_RESET	RESET0, RESET1	RESET0	Input CASISEL register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
CASISEL_PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input CASISEL pipeline register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock

Parameter	Range	Default Value	Description
			control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
CASISEL_PREG_CE	CE0, CE1	CE0	Input CASISEL pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
CASISEL_PREG_RESET	RESET0, RESET1	RESET0	Input CASISEL pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
ACCSEL_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ACCSEL register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
ACCSEL_IREG_CE	CE0, CE1	CE0	Input ACCSEL register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
ACCSEL_IREG_RESET	RESET0, RESET1	RESET0	Input ACCSEL register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
ACCSEL_PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ACCSEL pipeline register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock

Parameter	Range	Default Value	Description
			control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
ACCSEL_PREG_CE	CE0, CE1	CE0	Input ACCSEL pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
ACCSEL_PREG_RESET	RESET0, RESET1	RESET0	Input ACCSEL pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	M0 Pipeline register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
PREG_CE	CE0, CE1	CE0	M0 Pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
PREG_RESET	RESET0, RESET1	RESET0	M0 Pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
FB_PREG_EN	FALSE, TRUE	FALSE	Feedback output pipeline register control parameter ● FALSE: Bypass mode; ● TRUE: Register mode, control signal clk/ce/reset is consistent with OREG

Parameter	Range	Default Value	Description
SOA_PREG_EN	FALSE, TRUE	FALSE	Shift output SOA pipeline register control parameter ● FALSE: Bypass mode; ● TRUE: Register mode, control signal clk/ce/reset is consistent with AREG
OREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Output register clock control signal ● BYPASS: bypass mode; ● CLK0: Register mode, register clock control signal is from CLK[0]; ● CLK1: Register mode, register clock control signal is from CLK[1].
OREG_CE	CE0, CE1	CE0	Output register clock enable control signal ● CE0: Register clock enable control signal is from CE[0]; ● CE1: Register clock enable control signal is from CE[1].
OREG_RESET	RESET0, RESET1	RESET0	Output register reset control signal ● RESET0: Register reset control signal is from RESET[0]; ● RESET1: Register reset control signal is from RESET[1].
MULT_RESET_MODE	SYNC, ASYNC	SYNC	Reset mode configuration ● SYNC: synchronized reset ● ASYNC: asynchronous reset
PRE_LOAD	48’h000000000000 ~48’hFFFFFFFFFFFF	48’h0	PRE_LOAD initialization value
A_SEL	1’b0, 1’b1	1’b0	Static control A, SIA source selection ● 1’b0: select A ● 1’b1: select SIA
DYN_A_SEL	FALSE, TRUE	FALSE	Dynamic control A, SIA source selection ● FALSE: A_SEL static control A, SIA source selection; ● TRUE: Input ASEL to dynamic control A, SIA source selection
P_SEL	1’b0, 1’b1	1’b0	Static control INA, INA +/- D selection

Parameter	Range	Default Value	Description
			● 1’b0: select INA ● 1’b1: select INA +/- D
DYN_P_SEL	FALSE, TRUE	FALSE	Dynamic control INA, INA +/- D selection ● FALSE: P_SEL static control INA, INA +/- D selection ● TRUE: Input PSEL to dynamic control INA, INA +/- D selection
P_ADDSUB	1’b0, 1’b1	1’b0	Static control pre-adder addition/subtraction selection ● 1’b0: addition ● 1’b1: subtraction
DYN_P_ADDSUB	FALSE, TRUE	FALSE	Dynamic control pre-adder addition/subtraction selection ● FALSE: P_ADDSUB static control pre-adder addition/subtraction selection ● TRUE: Input PADDSUB to dynamic control pre-adder, addition/subtraction selection
ADD_SUB_0	1’b0, 1’b1	1’b0	Static control the addition/subtraction selection of M0/0 ● 1’b0: addition ● 1’b1: subtraction
DYN_ADD_SUB_0	FALSE, TRUE	FALSE	Dynamic control the addition/subtraction selection of M0/0 ● FALSE: ADD_SUB_0 static control the addition/subtraction selection of M0/0 ● TRUE: Input ADDSUB[0] dynamic control the addition/subtraction selection of M0/0
ADD_SUB_1	1’b0, 1’b1	1’b0	Static control the addition/subtraction selection of C/0 ● 1’b0: addition ● 1’b1: subtraction
DYN_ADD_SUB_1	FALSE, TRUE	FALSE	Dynamic control the addition/subtraction selection of C/0

Parameter	Range	Default Value	Description
			● FALSE: ADD_SUB_1 static control the addition/subtraction selection of C/0 ● TRUE: Input ADDSUB[1] dynamic control addition/subtraction of M1/C/0 selection
CASI_SEL	1'b0, 1'b1	1'b0	Static control CASI/0 selection ● 1'b0: select 0 ● 1'b1: select CASI
DYN_CASI_SEL	FALSE, TRUE	FALSE	Dynamic control CASI/0 selection ● FALSE: CASI_SEL static control CASI/0 source selection ● TRUE: Input CASISEL to dynamic control CASI/0 source selection
ACC_SEL	1'b0, 1'b1	1'b0	Static control PRE_LOAD, DOUT selection ● 1'b0: select PRE_LOAD ● 1' b1: select output feedback
DYN_ACC_SEL	FALSE, TRUE	FALSE	Dynamic control PRE_LOAD, DOUT selection ● FALSE: ACC_SEL static control PRE_LOAD, output feedback source selection ● TRUE: Input ACCSEL to dynamic control PRE_LOAD, output feedback source selection
C_SEL	1'b0, 1'b1	1'b0	Static control C, 0 selection ● 1'b0: select 0 ● 1'b1: select C
DYN_C_SEL	FALSE, TRUE	FALSE	Dynamic control C, 0 selection ● FALSE: C_SEL static control C/0 source selection ● TRUE: Input CSEL to dynamic control C/0 source selection
MULT12X12_EN	FALSE, TRUE	FALSE	Control M0 Mode ● FALSE: 27X18 mode ● TRUE: 12X12 mode

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 5 IP Generation.

Verilog Instantiation:

```
MULTALU27X18 multalu27x18_inst (
    .DOUT(dout),
    .CASO(caso),
    .SOA(soa),
    .A({a[25],a[25:0]}),
    .B(b),
    .C(c),
    .D(d),
    .SIA({gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd,gw_gnd}),
    .CASI(casi),
    .ACCSEL(gw_gnd),
    .CASISEL(gw_gnd),
    .ASEL(gw_gnd),
    .PSEL(gw_gnd),
    .CSEL(gw_gnd),
    .ADDSUB({gw_gnd,gw_gnd}),
    .PADDSUB(gw_gnd),
    .CLK({gw_gnd,clk}),
    .CE({gw_gnd,ce}),
    .RESET({gw_gnd,reset})
);
defparam multalu27x18_inst.AREG_CLK = "CLK0";
defparam multalu27x18_inst.AREG_CE = "CE0";
defparam multalu27x18_inst.AREG_RESET = "RESET0";
defparam multalu27x18_inst.BREG_CLK = "CLK0";
```

```
defparam multalu27x18_inst.BREG_CE = "CE0";
defparam multalu27x18_inst.BREG_RESET = "RESET0";
defparam multalu27x18_inst.DREG_CLK = "CLK0";
defparam multalu27x18_inst.DREG_CE = "CE0";
defparam multalu27x18_inst.DREG_RESET = "RESET0";
defparam multalu27x18_inst.C_IREG_CLK = "CLK0";
defparam multalu27x18_inst.C_IREG_CE = "CE0";
defparam multalu27x18_inst.C_IREG_RESET = "RESET0";
defparam multalu27x18_inst.PSEL_IREG_CLK = "BYPASS";
defparam multalu27x18_inst.PSEL_IREG_CE = "CE0";
defparam multalu27x18_inst.PSEL_IREG_RESET = "RESET0";
defparam multalu27x18_inst.PADDSUB_IREG_CLK = "BYPASS";
defparam multalu27x18_inst.PADDSUB_IREG_CE = "CE0";
defparam multalu27x18_inst.PADDSUB_IREG_RESET = "RESET0";
defparam multalu27x18_inst.ADDSUB0_IREG_CLK = "BYPASS";
defparam multalu27x18_inst.ADDSUB0_IREG_CE = "CE0";
defparam multalu27x18_inst.ADDSUB0_IREG_RESET = "RESET0";
defparam multalu27x18_inst.ADDSUB1_IREG_CLK = "BYPASS";
defparam multalu27x18_inst.ADDSUB1_IREG_CE = "CE0";
defparam multalu27x18_inst.ADDSUB1_IREG_RESET = "RESET0";
defparam multalu27x18_inst.CSEL_IREG_CLK = "BYPASS";
defparam multalu27x18_inst.CSEL_IREG_CE = "CE0";
defparam multalu27x18_inst.CSEL_IREG_RESET = "RESET0";
defparam multalu27x18_inst.CASISEL_IREG_CLK = "BYPASS";
defparam multalu27x18_inst.CASISEL_IREG_CE = "CE0";
defparam multalu27x18_inst.CASISEL_IREG_RESET = "RESET0";
defparam multalu27x18_inst.ACCSEL_IREG_CLK = "BYPASS";
defparam multalu27x18_inst.ACCSEL_IREG_CE = "CE0";
defparam multalu27x18_inst.ACCSEL_IREG_RESET = "RESET0";
defparam multalu27x18_inst.PREG_CLK = "BYPASS";
defparam multalu27x18_inst.PREG_CE = "CE0";
```

```
defparam multalu27x18_inst.PREG_RESET = "RESET0";
defparam multalu27x18_inst.ADDSUB0_PREG_CLK = "BYPASS";
defparam multalu27x18_inst.ADDSUB0_PREG_CE = "CE0";
defparam multalu27x18_inst.ADDSUB0_PREG_RESET = "RESET0";
defparam multalu27x18_inst.ADDSUB1_PREG_CLK = "BYPASS";
defparam multalu27x18_inst.ADDSUB1_PREG_CE = "CE0";
defparam multalu27x18_inst.ADDSUB1_PREG_RESET = "RESET0";
defparam multalu27x18_inst.CSEL_PREG_CLK = "BYPASS";
defparam multalu27x18_inst.CSEL_PREG_CE = "CE0";
defparam multalu27x18_inst.CSEL_PREG_RESET = "RESET0";
defparam multalu27x18_inst.CASISEL_PREG_CLK = "BYPASS";
defparam multalu27x18_inst.CASISEL_PREG_CE = "CE0";
defparam multalu27x18_inst.CASISEL_PREG_RESET = "RESET0";
defparam multalu27x18_inst.ACCSEL_PREG_CLK = "BYPASS";
defparam multalu27x18_inst.ACCSEL_PREG_CE = "CE0";
defparam multalu27x18_inst.ACCSEL_PREG_RESET = "RESET0";
defparam multalu27x18_inst.C_PREG_CLK = "CLK0";
defparam multalu27x18_inst.C_PREG_CE = "CE0";
defparam multalu27x18_inst.C_PREG_RESET = "RESET0";
defparam multalu27x18_inst.FB_PREG_EN = "FALSE";
defparam multalu27x18_inst.SOA_PREG_EN = "FALSE";
defparam multalu27x18_inst.OREG_CLK = "CLK0";
defparam multalu27x18_inst.OREG_CE = "CE0";
defparam multalu27x18_inst.OREG_RESET = "RESET0";
defparam multalu27x18_inst.MULT_RESET_MODE="SYNC";
defparam multalu27x18_inst.PRE_LOAD=48'h000000000000;
defparam multalu27x18_inst.DYN_P_SEL = "FALSE";
defparam multalu27x18_inst.P_SEL=1'b0;
defparam multalu27x18_inst.DYN_P_ADDSUB = "FALSE";
defparam multalu27x18_inst.P_ADDSUB=1'b0;
defparam multalu27x18_inst.DYN_A_SEL = "FALSE";
```

```
defparam multalu27x18_inst.A_SEL=1'b0;
defparam multalu27x18_inst.DYN_ADD_SUB_0 = "FALSE";
defparam multalu27x18_inst.ADD_SUB_0=1'b0;
defparam multalu27x18_inst.DYN_ADD_SUB_1 = "FALSE";
defparam multalu27x18_inst.ADD_SUB_1=1'b0;
defparam multalu27x18_inst.DYN_C_SEL = "FALSE";
defparam multalu27x18_inst.C_SEL=1'b1;
defparam multalu27x18_inst.DYN_CASI_SEL = "FALSE";
defparam multalu27x18_inst.CASI_SEL=1'b1;
defparam multalu27x18_inst.DYN_ACC_SEL = "FALSE";
defparam multalu27x18_inst.ACC_SEL=1'b0;
defparam multalu27x18_inst.MULT12X12_EN = "FALSE";
```

Vhdl Instantiation:

```
COMPONENT MULTALU27X18
    GENERIC (
        AREG_CLK : string := "BYPASS";
        AREG_CE : string := "CE0";
        AREG_RESET : string := "RESET0";
        BREG_CLK : string := "BYPASS";
        BREG_CE : string := "CE0";
        BREG_RESET : string := "RESET0";
        DREG_CLK : string := "BYPASS";
        DREG_CE:string:="CE0";
        DREG_RESET : string := "RESET0";
        C_IREG_CLK : string := "BYPASS";
        C_IREG_CE:string:="CE0";
        C_IREG_RESET : string := "RESET0";
        PSEL_IREG_CLK : string := "BYPASS";
        PSEL_IREG_CE:string:="CE0";
        PSEL_IREG_RESET:string:="RESET0";
        PADDSUB_IREG_CLK : string := "BYPASS";
```

```
PADDSUB_IREG_CE:string:="CE0";
PADDSUB_IREG_RESET:string:="RESET0";
ADDSUB0_IREG_CLK : string := "BYPASS";
ADDSUB0_IREG_CE:string:="CE0";
ADDSUB0_IREG_RESET:string:="RESET0";
ADDSUB1_IREG_CLK : string := "BYPASS";
ADDSUB1_IREG_CE:string:="CE0";
ADDSUB1_IREG_RESET:string:="RESET0";
CSEL_IREG_CLK : string := "BYPASS";
CSEL_IREG_CE:string:="CE0";
CSEL_IREG_RESET:string:="RESET0";
CASISEL_IREG_CLK : string := "BYPASS";
CASISEL_IREG_CE:string:="CE0";
CASISEL_IREG_RESET:string:="RESET0";
ACCSEL_IREG_CLK : string := "BYPASS";
ACCSEL_IREG_CE:string:="CE0";
ACCSEL_IREG_RESET:string:="RESET0";
PREG_CLK : string := "BYPASS";
PREG_CE : string := "CE0";
PREG_RESET : string := "RESET0";
ADDSUB0_PREG_CLK : string := "BYPASS";
ADDSUB0_PREG_CE:string:="CE0";
ADDSUB0_PREG_RESET:string:="RESET0";
ADDSUB1_PREG_CLK : string := "BYPASS";
ADDSUB1_PREG_CE:string:="CE0";
ADDSUB1_PREG_RESET:string:="RESET0";
CSEL_PREG_CLK : string := "BYPASS";
CSEL_PREG_CE:string:="CE0";
CSEL_PREG_RESET:string:="RESET0";
CASISEL_PREG_CLK : string := "BYPASS";
CASISEL_PREG_CE:string:="CE0";
```

```
CASISEL_PREG_RESET:string:="RESET0";
ACCSEL_PREG_CLK : string := "BYPASS";
ACCSEL_PREG_CE:string:="CE0";
ACCSEL_PREG_RESET:string:="RESET0";
C_PREG_CLK : string := "BYPASS";
C_PREG_CE:string:="CE0";
C_PREG_RESET:string:="RESET0";
FB_PREG_EN:string:="FALSE";
SOA_PREG_EN:string:="FALSE";
OREG_CLK : string := "BYPASS";
OREG_CE : string := "CE0";
OREG_RESET : string := "RESET0";
MULT_RESET_MODE:string:="SYNC";
PRE_LOAD : bit_vector := X"000000000000";
DYN_P_SEL : string := "FALSE";
P_SEL : bit := '0';
DYN_P_ADDSUB : string := "FALSE";
P_ADDSUB : bit := '0';
DYN_A_SEL : string := "FALSE";
A_SEL : bit := '0';
DYN_ADD_SUB_0 : string := "FALSE";
ADD_SUB_0:bit:='0';
DYN_ADD_SUB_1 : string := "FALSE";
ADD_SUB_1:bit:='0';
DYN_C_SEL : string := "FALSE";
C_SEL : bit := '0';
DYN_CASI_SEL : string := "FALSE";
CASI_SEL : bit := '0';
DYN_ACC_SE : string := "FALSE";
ACC_SEL : bit := '0';
MULT12X12_EN : string := "FALSE"
```

```
        );
        PORT(
            DOUT: out std_logic_vector(47 downto 0);
            CASO: out std_logic_vector(47 downto 0);
            SOA: out std_logic_vector(26 downto 0);
            A: in std_logic_vector(26 downto 0);
            B: in std_logic_vector(17 downto 0);
            C: in std_logic_vector(47 downto 0);
            D: in std_logic_vector(25 downto 0);
            SIA: in std_logic_vector(26 downto 0);
            CASI: in std_logic_vector(47 downto 0);
            ACCSEL: in std_logic;
            CASISEL: in std_logic;
            ASEL: in std_logic;
            PSEL: in std_logic;
            CSEL: in std_logic;
            ADDSUB: in std_logic_vector(1 downto 0);
            PADDSUB: in std_logic;
            CLK: in std_logic_vector(1 downto 0);
            CE: in std_logic_vector(1 downto 0);
            RESET: in std_logic_vector(1 downto 0)
        );
    end COMPONENT;
begin
    gw_gnd <= '0';

    A_i <= a[25] & a(25 downto 0);
    SIA_i <= gw_gnd & gw_gnd & gw_gnd & gw_gnd & gw_gnd &
gw_gnd & gw_gnd & gw_gnd & gw_gnd & gw_gnd & gw_gnd & gw_gnd &
gw_gnd & gw_gnd & gw_gnd & gw_gnd & gw_gnd & gw_gnd & gw_gnd &
gw_gnd & gw_gnd & gw_gnd & gw_gnd & gw_gnd & gw_gnd & gw_gnd &
gw_gnd;
```

```
ADDSUB_i <= gw_gnd & gw_gnd;
CLK_i <= gw_gnd & clk;
CE_i <= gw_gnd & ce;
RESET_i <= gw_gnd & reset;

multalu27x18_inst: MULTALU27X18
    GENERIC MAP(
        AREG_CLK => "CLK0",
        AREG_CE => "CE0",
        AREG_RESET => "RESET0",
        BREG_CLK => "CLK0",
        BREG_CE => "CE0",
        BREG_RESET => "RESET0",
        DREG_CLK => "CLK0",
        DREG_CE => "CE0",
        DREG_RESET => "RESET0",
        C_IREG_CLK => "CLK0",
        C_IREG_CE => "CE0",
        C_IREG_RESET => "RESET0",
        PSEL_IREG_CLK => "BYPASS",
        PSEL_IREG_CE => "CE0",
        PSEL_IREG_RESET => "RESET0",
        PADDSUB_IREG_CLK => "BYPASS",
        PADDSUB_IREG_CE => "CE0",
        PADDSUB_IREG_RESET => "RESET0",
        ADDSUB0_IREG_CLK => "BYPASS",
        ADDSUB0_IREG_CE => "CE0",
        ADDSUB0_IREG_RESET => "RESET0",
        ADDSUB1_IREG_CLK => "BYPASS",
        ADDSUB1_IREG_CE => "CE0",
        ADDSUB1_IREG_RESET => "RESET0",
```

```
CSEL_IREG_CLK => "BYPASS",
CSEL_IREG_CE => "CE0",
CSEL_IREG_RESET => "RESET0",
CASISEL_IREG_CLK => "BYPASS",
CASISEL_IREG_CE => "CE0",
CASISEL_IREG_RESET => "RESET0",
ACCSEL_IREG_CLK => "BYPASS",
ACCSEL_IREG_CE => "CE0",
ACCSEL_IREG_RESET => "RESET0",
PREG_CLK => "BYPASS",
PREG_CE => "CE0",
PREG_RESET => "RESET0",
ADDSUB0_PREG_CLK => "BYPASS",
ADDSUB0_PREG_CE => "CE0",
ADDSUB0_PREG_RESET => "RESET0",
ADDSUB1_PREG_CLK => "BYPASS",
ADDSUB1_PREG_CE => "CE0",
ADDSUB1_PREG_RESET => "RESET0",
CSEL_PREG_CLK => "BYPASS",
CSEL_PREG_CE => "CE0",
CSEL_PREG_RESET => "RESET0",
CASISEL_PREG_CLK => "BYPASS",
CASISEL_PREG_CE => "CE0",
CASISEL_PREG_RESET => "RESET0",
ACCSEL_PREG_CLK => "BYPASS",
ACCSEL_PREG_CE => "CE0",
ACCSEL_PREG_RESET => "RESET0",
C_PREG_CLK => "CLK0",
C_PREG_CE => "CE0",
C_PREG_RESET => "RESET0",
FB_PREG_EN => "FALSE",
```

```
        SOA_PREG_EN => "FALSE",
        OREG_CLK => "CLK0",
        OREG_CE => "CE0",
        OREG_RESET => "RESET0",
        MULT_RESET_MODE => "SYNC",
        PRE_LOAD => X"000000000000",
        DYN_P_SEL => "FALSE",
        P_SEL => '0',
        DYN_P_ADDSUB => "FALSE",
        P_ADDSUB => '0',
        DYN_A_SEL => "FALSE",
        A_SEL => '0',
        DYN_ADD_SUB_0 => "FALSE",
        ADD_SUB_0 => '0',
        DYN_ADD_SUB_1 => "FALSE",
        ADD_SUB_1 => '0',
        DYN_C_SEL => "FALSE",
        C_SEL => '1',
        DYN_CASI_SEL => "FALSE",
        CASI_SEL => '1',
        DYN_ACC_SEL => "FALSE",
        ACC_SEL => '0',
        MULT12X12_EN => "FALSE"
    )
    PORT MAP (
        DOUT => dout,
        CASO=>caso,
        SOA => soa,
        A=>A_i,
        B=>b,
        C => c,
```

```
D=>d,
SIA => SIA_i,
CASI=>casi,
ACCSEL => gw_gnd,
CASISEL => gw_gnd,
ASEL => gw_gnd,
PSEL=>gw_gnd,
CSEL => gw_gnd,
ADDSUB => ADDSUB_i,
PADDSUB=>gw_gnd,
CLK => CLK_i,
CE => CE_i,
RESET => RESET_i
);
```

4.3 MULTADDALU

4.3.1 MULTADDALU12X12

The MULTADDALU mode implements two 12 x 12 multiplier outputs after the 48-bit ALU operation, corresponding to MULTADDALU12x12 primitive.

The MULTADDALU12X12 has four operation modes:

$$DOUT = A0 * B0 \pm A1 * B1$$

$$DOUT = DOUT \pm (A0 * B0 \pm A1 * B1)$$

$$DOUT = CASI \pm A0 * B0 \pm A1 * B1$$

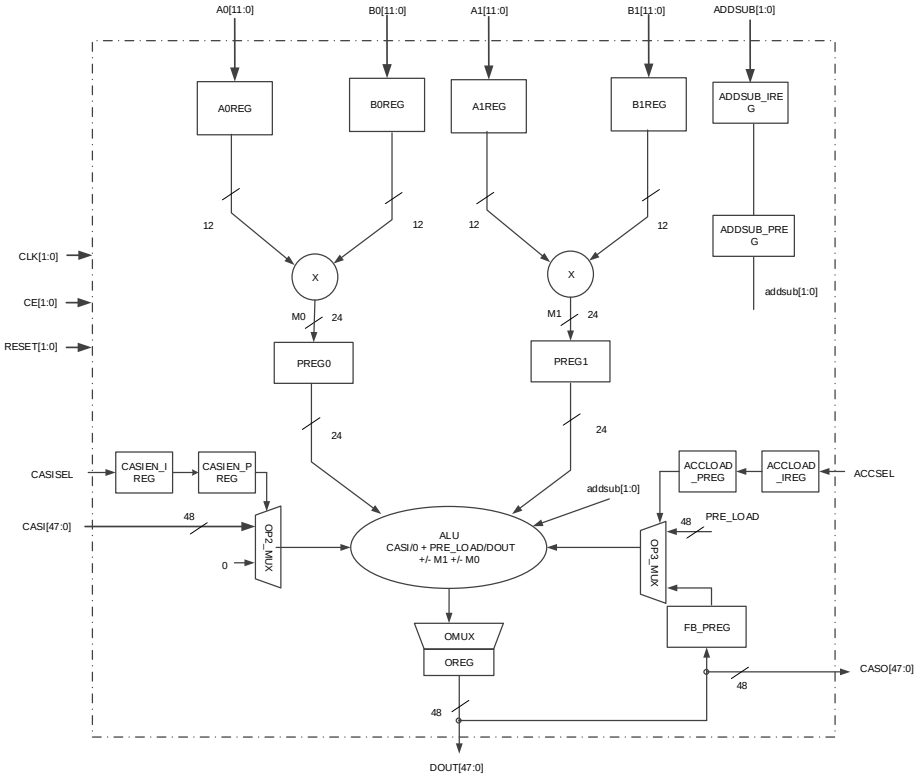
$$DOUT = CASI \pm (A0 * B0 \pm A1 * B1) + DOUT$$

Primitive Introduction

The MULTADDALU12x12 (The Sum of Two 12x12 Multipliers with ALU) implements 12-bit accumulation operation after summing up the multiplication.

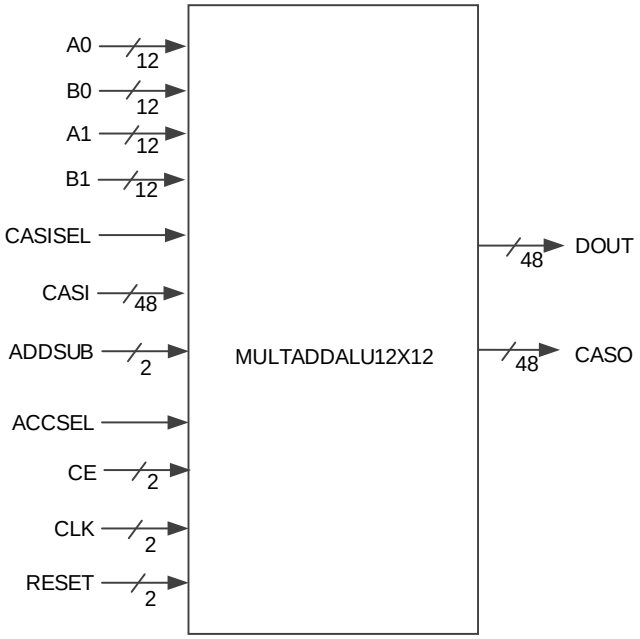
Logic Structure Diagram

Figure 4-7 MULTADDALU12X12 Logic Structure Diagram



Port Diagram

Figure 4-8 MULTADDALU12X12 Port Diagram



Port Description

Table 4-7 MULTADDALU12X12 Port Description

Port	I/O	Description
A0[11:0]	Input	12-bit data input signal A0
B0[11:0]	Input	12-bit data input signal B0
A1[11:0]	Input	12-bit data input signal A1
B1[11:0]	Input	12-bit data input signal B1
CASI[47:0]	Input	48-bit cascading input signal of previous DSP
CASISEL	Input	CASI/O control Input signal of 48-bit ALU
ADDSUB[1:0]	Input	Dynamic +/- control input signal
ACCSEL	Input	DOUT/PRE_LOAD control input signal of 48-bit ALU
CLK[1:0]	Input	Clock input signal
CE[1:0]	Input	Clock enable signal
RESET[1:0]	Input	Reset input signal
DOUT[53:0]	Output	Data output signal
CASO[54:0]	Output	48-bit cascading output signal

Parameter Description

Table 4-8 MULTADDALU12X12 Parameter Description

Parameter	Range	Default Value	Description
A0REG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input A0 register clock control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
A0REG_CE	CE0, CE1	CE0	Input A0 register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
A0REG_RESET	RESET0, RESET1	RESET0	Input A0 register reset control signal

Parameter	Range	Default Value	Description
			● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
B0REG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input B0 register clock control signal ● BYPASS: Bypass mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
B0REG_CE	CE0, CE1	CE0	Input B0 register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
B0REG_RESET	RESET0, RESET1	RESET0	Input B0 register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
A1REG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input A1 register clock control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
A1REG_CE	CE0, CE1	CE0	Input A1 register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
A1REG_RESET	RESET0, RESET1	RESET0	Input A1 register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal

Parameter	Range	Default Value	Description
			is from RESET[1]
B1REG_CLK	BYPASS, CLK0, CLK1	CLK0	Input B1 register clock control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
B1REG_CE	CE0, CE1	CE0	Input B1 register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
B1REG_RESET	RESET0, RESET1	RESET0	Input B1 register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
ADDSUB0_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ADDSUB[0] register clock control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
ADDSUB0_IREG_CE	CE0, CE1	CE0	Input ADDSUB[0] register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
ADDSUB0_IREG_RESET	RESET0, RESET1	RESET0	Input ADDSUB[0] register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]

Parameter	Range	Default Value	Description
ADDSUB1_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ADDSUB[1] register clock control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
ADDSUB1_IREG_CE	CE0, CE1	CE0	Input ADDSUB[1] register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
ADDSUB1_IREG_RESET	RESET0, RESET1	RESET0	Input ADDSUB[1] register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
ADDSUB0_PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ADDSUB[0] pipeline register clock control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
ADDSUB0_PREG_CE	CE0, CE1	CE0	Input ADDSUB[0] pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
ADDSUB0_PREG_RESET	RESET0.RESET1	RESET0	Input ADDSUB[0] pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]

Parameter	Range	Default Value	Description
ADDSUB1_PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ADDSUB[1] pipeline register clock control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
ADDSUB1_PREG_CE	CE0, CE1	CE0	Input ADDSUB[1] pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
ADDSUB1_PREG_RESET	RESET0, RESET1	RESET0	Input ADDSUB[1] pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
CASISEL_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input CASISEL register clock control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
CASISEL_IREG_CE	CE0, CE1	CE0	Input CASISEL register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
CASISEL_IREG_RESET	RESET0, RESET1	RESET0	Input CASISEL register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
CASISEL_PREG_	BYPASS, CLK0,	BYPASS	Input CASISEL pipeline register clock

Parameter	Range	Default Value	Description
CLK	CLK1		control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
CASISEL_PREG_CE	CE0, CE1	CE0	Input CASISEL pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
CASISEL_PREG_RESET	RESET0, RESET1	RESET0	Input CASISEL pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
ACCSEL_IREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ACCSEL register clock control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
ACCSEL_IREG_CE	CE0, CE1	CE0	Input ACCSELSEL register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
ACCSEL_IREG_RESET	RESET0, RESET1	RESET0	Input ACCSEL register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
ACCSEL_PREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Input ACCSEL pipeline register clock control signal

Parameter	Range	Default Value	Description
			● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
ACCSEL_PREG_CE	CE0, CE1	CE0	Input ACCSEL pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
ACCSEL_PREG_RESET	RESET0, RESET1	RESET0	Input ACCSEL pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
PREG0_CLK	BYPASS, CLK0, CLK1	BYPASS	Mult0 pipeline register clock control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
PREG0_CE	CE0, CE1	CE0	Mult0 Pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
PREG0_RESET	RESET0, RESET1	RESET0	Mult0 Pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
PREG1_CLK	BYPASS, CLK0, CLK1	BYPASS	Mult1 Pipeline register clock control signal ● BYPASS: Bypass Mode ● CLK0: Register mode, register clock

Parameter	Range	Default Value	Description
			control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
PREG1_CE	CE0, CE1	CE0	Mult1 Pipeline register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
PREG1_RESET	RESET0, RESET1	RESET0	Mult1 Pipeline register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
FB_PREG_EN	FALSE, TRUE	FALSE	Feedback output pipeline register mode control parameter ● FALSE: Bypass mode: ● TRUE: Register enable, control signal clk/ce/reset is consistent with OREG
OREG_CLK	BYPASS, CLK0, CLK1	BYPASS	Output register clock control signal ● BYPASS: Bypass mode: ● CLK0: Register mode, register clock control signal is from CLK[0] ● CLK1: Register mode, register clock control signal is from CLK[1]
OREG_CE	CE0, CE1	CE0	Output register clock enable control signal ● CE0: Register clock enable control signal is from CE[0] ● CE1: Register clock enable control signal is from CE[1]
OREG_RESET	RESET0, RESET1	RESET0	Output register reset control signal ● RESET0: Register reset control signal is from RESET[0] ● RESET1: Register reset control signal is from RESET[1]
MULT_RESET_MODE	SYNC, ASYNC	SYNC	Synchronous or asynchronous

Parameter	Range	Default Value	Description
PRE_LOAD	48bits value	48'h0	PRE_LOAD instantiation value
ADD_SUB_0	1'b0, 1'b1	1'b0	Static control addition/subtraction selection of M0/0 ● 1'b0: Addition ● 1'b1: Subtraction
DYN_ADD_SUB_0	FALSE, TRUE	FALSE	Dynamic control addition/subtraction selection of M0/0 ● FALSE: ADD_SUB_0 static control addition/subtraction selection of M0/0 ● TRUE: Input ADDSUB[0] dynamic control addition/subtraction selection of M0/0
ADD_SUB_1	1'b0, 1'b1	1'b0	Static control addition/subtraction of M1/0 ● 1'b0: Addition ● 1'b1: Subtraction
DYN_ADD_SUB_1	FALSE, TRUE	FALSE	Dynamic control addition/subtraction selection of M1/0 ● FALSE: ADD_SUB_1 static control addition/subtraction selection of M1/0 ● TRUE: Input ADDSUB[1] dynamic control addition/subtraction selection of M1/0
CASI_SEL	1'b0, 1'b1	1'b0	Static control CASI/0 selection ● 1'b0: 0 ● 1'b1: CASI
DYN_CASI_SEL	FALSE, TRUE	FALSE	Dynamic control CASI/0 selection ● FALSE: CASI_SEL static control CASI/0 selection ● TRUE: Input CASISEL dynamic control CASI/0 selection
ACC_SEL	1'b0, 1'b1	1'b0	Static control PRE_LOAD/DOUT selection ● 1'b0: PRE_LOAD ● 1'B1: DOUT of feedback
DYN_ACC_SEL	FALSE, TRUE	FALSE	Dynamic control PRE_LOAD/DOUT selection ● FALSE: ACC_SEL static control PRE_LOAD/DOUT selection

Parameter	Range	Default Value	Description
			● TRUE: Input ACCSEL dynamic control PRE_LOAD/DOUT selection

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 5 IP Generation.

Verilog Instantiation:

```
MULTADDALU12X12 multaddalu12x12_inst (
    .DOUT(dout),
    .CASO(caso),
    .A0(a0),
    .B0(b0),
    .A1(a1),
    .B1(b1),
    .CASI(casi),
    .ACCSEL(gw_gnd),
    .CASISEL(gw_gnd),
    .ADDSUB({gw_gnd,gw_gnd}),
    .CLK({gw_gnd,clk}),
    .CE({gw_gnd,ce}),
    .RESET({gw_gnd,reset})
);

defparam multaddalu12x12_inst.A0REG_CLK = "CLK0";
defparam multaddalu12x12_inst.A0REG_CE = "CE0";
defparam multaddalu12x12_inst.A0REG_RESET = "RESET0";
defparam multaddalu12x12_inst.A1REG_CLK = "CLK0";
defparam multaddalu12x12_inst.A1REG_CE = "CE0";
defparam multaddalu12x12_inst.A1REG_RESET = "RESET0";
defparam multaddalu12x12_inst.B0REG_CLK = "CLK0";
```

```
defparam multaddalu12x12_inst.B0REG_CE = "CE0";
defparam multaddalu12x12_inst.B0REG_RESET = "RESET0";
defparam multaddalu12x12_inst.B1REG_CLK = "CLK0";
defparam multaddalu12x12_inst.B1REG_CE = "CE0";
defparam multaddalu12x12_inst.B1REG_RESET = "RESET0";
defparam multaddalu12x12_inst.ACCSEL_IREG_CLK = "BYPASS";
defparam multaddalu12x12_inst.ACCSEL_IREG_CE = "CE0";
defparam multaddalu12x12_inst.ACCSEL_IREG_RESET = "RESET0";
defparam multaddalu12x12_inst.CASISEL_IREG_CLK = "BYPASS";
defparam multaddalu12x12_inst.CASISEL_IREG_CE = "CE0";
defparam multaddalu12x12_inst.CASISEL_IREG_RESET = "RESET0";
defparam multaddalu12x12_inst.ADDSUB0_IREG_CLK = "BYPASS";
defparam multaddalu12x12_inst.ADDSUB0_IREG_CE = "CE0";
defparam multaddalu12x12_inst.ADDSUB0_IREG_RESET = "RESET0";
defparam multaddalu12x12_inst.ADDSUB1_IREG_CLK = "BYPASS";
defparam multaddalu12x12_inst.ADDSUB1_IREG_CE = "CE0";
defparam multaddalu12x12_inst.ADDSUB1_IREG_RESET = "RESET0";
defparam multaddalu12x12_inst.PREG0_CLK = "BYPASS";
defparam multaddalu12x12_inst.PREG0_CE = "CE0";
defparam multaddalu12x12_inst.PREG0_RESET = "RESET0";
defparam multaddalu12x12_inst.PREG1_CLK = "BYPASS";
defparam multaddalu12x12_inst.PREG1_CE = "CE0";
defparam multaddalu12x12_inst.PREG1_RESET = "RESET0";
defparam multaddalu12x12_inst.FB_PREG_EN = "FALSE";
defparam multaddalu12x12_inst.ACCSEL_PREG_CLK = "BYPASS";
defparam multaddalu12x12_inst.ACCSEL_PREG_CE = "CE0";
defparam multaddalu12x12_inst.ACCSEL_PREG_RESET = "RESET0";
defparam multaddalu12x12_inst.CASISEL_PREG_CLK = "BYPASS";
defparam multaddalu12x12_inst.CASISEL_PREG_CE = "CE0";
defparam multaddalu12x12_inst.CASISEL_PREG_RESET = "RESET0";
defparam multaddalu12x12_inst.ADDSUB0_PREG_CLK = "BYPASS";
```

```
  defparam multaddalu12x12_inst.ADDSUB0_PREG_CE = "CE0";
  defparam multaddalu12x12_inst.ADDSUB0_PREG_RESET =
"RESET0";
  defparam multaddalu12x12_inst.ADDSUB1_PREG_CLK = "BYPASS";
  defparam multaddalu12x12_inst.ADDSUB1_PREG_CE = "CE0";
  defparam multaddalu12x12_inst.ADDSUB1_PREG_RESET =
"RESET0";
  defparam multaddalu12x12_inst.OREG_CLK = "CLK0";
  defparam multaddalu12x12_inst.OREG_CE = "CE0";
  defparam multaddalu12x12_inst.OREG_RESET = "RESET0";
  defparam multaddalu12x12_inst.MULT_RESET_MODE="SYNC";
  defparam multaddalu12x12_inst.PRE_LOAD=48'h000000000000;
  defparam multaddalu12x12_inst.DYN_ADD_SUB_0 = "FALSE";
  defparam multaddalu12x12_inst.ADD_SUB_0=1'b0;
  defparam multaddalu12x12_inst.DYN_ADD_SUB_1 = "FALSE";
  defparam multaddalu12x12_inst.ADD_SUB_1=1'b0;
  defparam multaddalu12x12_inst.DYN_CASI_SEL = "FALSE";
  defparam multaddalu12x12_inst.CASI_SEL=1'b1;
  defparam multaddalu12x12_inst.DYN_ACC_SEL = "FALSE";
  defparam multaddalu12x12_inst.ACC_SEL=1'b0;
```

Vhdl Instantiation:

```
  COMPONENT MULTADDALU12X12
        GENERIC (
            A0REG_CLK : string := "BYPASS";
            A0REG_CE:string:="CE0";
            A0REG_RESET:string:="RESET0";
            A1REG_CLK : string := "BYPASS";
            A1REG_CE:string:="CE0";
            A1REG_RESET:string:="RESET0";
            B0REG_CLK : string := "BYPASS";
            B0REG_E:string:="CE0";
```

```
B0REG_RESET:string:="RESET0";
B1REG_CLK : string := "BYPASS";
B1REG_CE:string:="CE0";
B1REG_RESET:string:="RESET0";
ACCSEL_IREG_CLK : string := "BYPASS";
ACCSEL_IREG_CE:string:="CE0";
ACCSEL_IREG_RESET : string := "RESET0";
CASISEL_IREG_CLK : string := "BYPASS";
CASISEL_IREG_CE:string:="CE0";
CASISEL_IREG_RESET:string:="RESET0";
ADDSUB0_IREG_CLK : string := "BYPASS";
ADDSUB0_IREG_CE:string:="CE0";
ADDSUB0_IREG_RESET:string:="RESET0";
ADDSUB1_IREG_CLK : string := "BYPASS";
ADDSUB1_IREG_CE:string:="CE0";
ADDSUB1_IREG_RESET:string:="RESET0";
PREG0_CLK : string := "BYPASS";
PREG0_CE:string:="CE0";
PREG0_RESET:string:="RESET0";
PREG1_CLK : string := "BYPASS";
PREG1_CE:string:="CE0";
PREG1_RESET:string:="RESET0";
FB_PREG_EN:string:="FALSE";
ACCSEL_PREG_CLK : string := "BYPASS";
ACCSEL_PREG_CE:string:="CE0";
ACCSEL_PREG_RESET:string:="RESET0";
CASISEL_PREG_CLK : string := "BYPASS";
CASISEL_PREG_CE:string:="CE0";
CASISEL_PREG_RESET:string:="RESET0";
ADDSUB0_PREG_CLK : string := "BYPASS";
ADDSUB0_PREG_CE:string:="CE0";
```

```
        ADDSUB0_PREG_RESET:string:="RESET0";
        ADDSUB1_PREG_CLK : string := "BYPASS";
        ADDSUB1_PREG_CE:string:="CE0";
        ADDSUB1_PREG_RESET:string:="RESET0";
        OREG_CLK : string := "BYPASS";
        OREG_CE : string := "CE0";
        OREG_RESET : string := "RESET0";
        MULT_RESET_MODE:string:="SYNC";
        PRE_LOAD : bit_vector := X"000000000000";
        DYN_ADD_SUB_0 : string := "FALSE";
        ADD_SUB_0:bit:='0';
        DYN_ADD_SUB_1 : string := "FALSE";
        ADD_SUB_1:bit:='0';
        DYN_CASI_SEL : string := "FALSE";
        CASI_SEL : bit := '0';
        DYN_ACC_SE : string := "FALSE";
        ACC_SEL : bit := '0';
    );
    PORT(
        DOUT: out std_logic_vector(47 downto 0);
        CASO: out std_logic_vector(47 downto 0);
        A0: in std_logic_vector(11 downto 0);
        B0: in std_logic_vector(11 downto 0);
        A1: in std_logic_vector(11 downto 0);
        B1: in std_logic_vector(11 downto 0);
        CASI: in std_logic_vector(47 downto 0);
        ACCSEL: in std_logic;
        CASISEL: in std_logic;
        ADDSUB: in std_logic_vector(1 downto 0);
        CLK: in std_logic_vector(1 downto 0);
        CE: in std_logic_vector(1 downto 0);
```

```
            RESET: in std_logic_vector(1 downto 0)
        );
    end COMPONENT;
begin
    gw_gnd <= '0';

    ADDSUB_i <= gw_gnd & gw_gnd;
    CLK_i <= gw_gnd & clk;
    CE_i <= gw_gnd & ce;
    RESET_i <= gw_gnd & reset;

    multaddalu12x12_inst: MULTADDALU12X12
        GENERIC MAP(
            A0REG_CLK => "CLK0",
            A0REG_CE => "CE0",
            A0REG_RESET => "RESET0",
            A1REG_CLK => "CLK0",
            A1REG_CE => "CE0",
            A1REG_RESET => "RESET0",
            B0REG_CLK => "CLK0",
            B0REG_CE => "CE0",
            B0REG_RESET => "RESET0",
            B1REG_CLK => "CLK0",
            B1REG_CE => "CE0",
            B1REG_RESET => "RESET0",
            ACCSEL_IREG_CLK => "BYPASS",
            ACCSEL_IREG_CE => "CE0",
            ACCSEL_IREG_RESET => "RESET0",
            CASISEL_IREG_CLK => "BYPASS",
            CASISEL_IREG_CE => "CE0",
            CASISEL_IREG_RESET => "RESET0",
```

```
ADDSUB0_IREG_CLK => "BYPASS",
ADDSUB0_IREG_CE => "CE0",
ADDSUB0_IREG_RESET => "RESET0",
ADDSUB1_IREG_CLK => "BYPASS",
ADDSUB1_IREG_CE => "CE0",
ADDSUB1_IREG_RESET => "RESET0",
PREG0_CLK => "BYPASS",
PREG0_CE => "CE0",
PREG0_RESET => "RESET0",
PREG1_CLK => "BYPASS",
PREG1_CE => "CE0",
PREG1_RESET => "RESET0",
FB_PREG_EN => "FALSE",
ACCSEL_PREG_CLK => "BYPASS",
ACCSEL_PREG_CE => "CE0",
ACCSEL_PREG_RESET => "RESET0",
CASISEL_PREG_CLK => "BYPASS",
CASISEL_PREG_CE => "CE0",
CASISEL_PREG_RESET => "RESET0",
ADDSUB0_PREG_CLK => "BYPASS",
ADDSUB0_PREG_CE => "CE0",
ADDSUB0_PREG_RESET => "RESET0",
ADDSUB1_PREG_CLK => "BYPASS",
ADDSUB1_PREG_CE => "CE0",
ADDSUB1_PREG_RESET => "RESET0",
OREG_CLK => "CLK0",
OREG_CE => "CE0",
OREG_RESET => "RESET0",
MULT_RESET_MODE => "SYNC"
PRE_LOAD => X"000000000000",
DYN_ADD_SUB_0 => "FALSE",
```

```
        ADD_SUB_0 => '0',
        DYN_ADD_SUB_1 => "FALSE",
        ADD_SUB_1 => '0',
        DYN_CASI_SEL => "FALSE",
        CASI_SEL => '1',
        DYN_ACC_SEL => "FALSE",
        ACC_SEL => '0',
    )
    PORT MAP (
        DOUT => dout,
        CASO => caso,
        A0 => a0,
        B0 => b0,
        A1 => a1,
        B1 => b1,
        CASI => casi,
        ACCSEL => gw_gnd,
        CASISEL => gw_gnd,
        ADDSUB => ADDSUB_i,
        CLK => CLK_i,
        CE => CE_i,
        RESET => RESET_i
    );
```

5 IP Generation

DSP block in IP Core Generator supports three types of Gowin Primitives: MULT, MULTALU, and MULTADDALU.

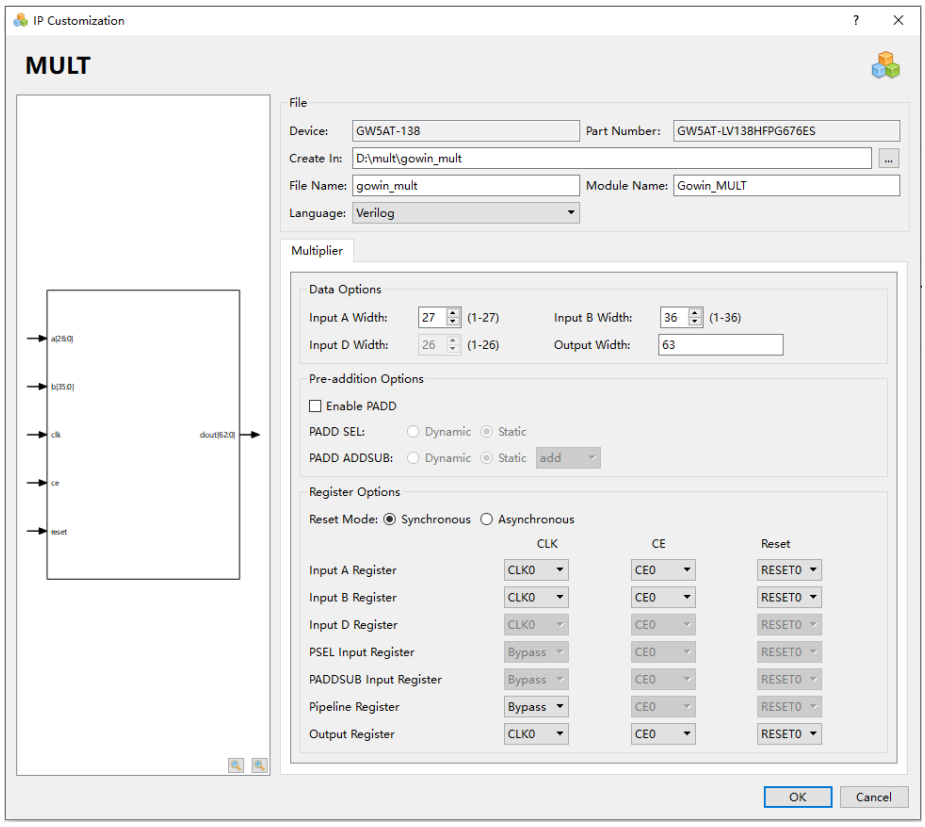
5.1 MULT

MULT implements multiplication operation based on the pre-addition and pre-subtraction of multiplication. Click "MULT" on the IP Core Generator, and a brief introduction to the MULT will be displayed.

IP Configuration

Double-click "MULT" to open the "IP Customization" window, as shown in Figure 5-1. This window includes the "File", "Multiplier", and port diagram.

Figure 5-1 IP Customization of MULT



1. The File Configuration: Configure the information about the generated IP design file.
   - Device: Display information about the configured Device.
   - Part Number: Display the configured Part Number.
   - Language: Hardware description language used to generate the IP design files. Click the drop-down list to select the language, including Verilog and VHDL.
   - Module Name: The module name of the generated IP design files. Enter the module name in the text box. Module name cannot be the same as the primitive name. If it is the same, an error will be reported.
   - File Name: The name of the generated IP design files. Enter the file name in the text box.
   - Create In: The path in which the generated IP files will be stored. Enter the target path in the box or select the target path by clicking the option.
2. The Multiplier Configuration: Configure IP by users, as shown in Figure 5-1.
   - Data Options: Configure data

- The maximum data width of the input ports (Input A Width) is 27
- The maximum data width of the input ports (Input B Width) is 36
- The maximum data width of the input ports (Input D Width) is 26
- Output width adjusts automatically according to input width and generates MULT12X12 and MULT27X36 according to the width during the instance.

- Pre-addition Options: Configure pre-addtion options
  - Enable PADD: Configure PADD
  - PADD SEL: Configure PADD to enable dynamic control or static control
  - PADD ADDSUB: Configure PADD to pre-add and pre-subtract as dynamic control or static control
  - Output width adjusts automatically according to input width and generates MULT12X12 and MULT27X36 according to the width during the instance.
- Register Options: Configure register operation mode.
  - Reset Mode: Configure MULT reset mode, supporting synchronous mode and asynchronous mode
  - Input A Register: Configure clk, ce, and reset signal source of Input A Register CLK: Configure as Bypass, CLK0, or CLK1 CE: Configure as CE0 or CE1 RESET: Configure as RESET0 or RESET1
  - Input B Register: Configure clk, ce, and reset signal source of Input B Register Configuration options are as above
  - Input D Register: Configure clk, ce, and reset signal source of Input D Register Configuration options are as above
  - PSEL Input Register: Configure clk, ce, and reset signal source of PSEL Input Register Configuration options are as above
  - PADDSUB Input Register: Configure clk, ce, and reset signal source of PADDSUB Input Register Configuration options are as above
  - Pipeline Register: Configure clk, ce, and reset signal source of Pipeline Register Configuration options are as above
  - Output Register: Configure clk, ce, and reset signal source of Output Register Configuration options are as above

3. Ports Diagram: Display current IP Core configuration. The input/output number and its bit-width update in real time based on the "Multiplier" configuration, as shown in Figure 5-1.

IP Generation Files

After configuration, it will generate three files that are named after the "File Name".

- “gowin_mult.v” file is a complete Verilog module to generate instance MULT, and it is generated according to the IP configuration;
- “gowin_mult_tmp.v” is the instance template file;
- “gowin_rpll.ipc” file is IP configuration file. You can load the file to configure the IP.

Note!

If VHDL is selected as the hardware description language, the first two files will be named with .vhd suffix.

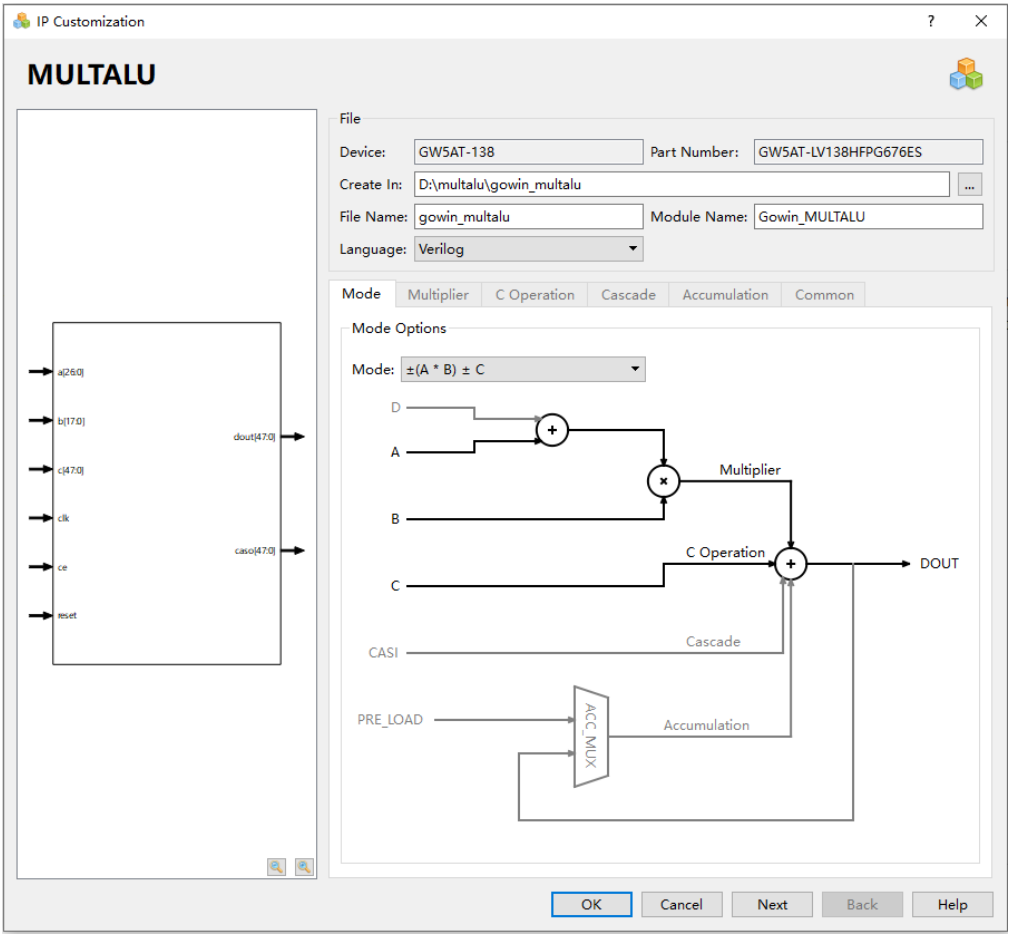
5.2 MULTALU

The MULTALU implements multiplication, multiply-add, accumulation, multiply-accumulate, shift based on multiplication/multiply-add/accumulation/multiply-accumulate, cascade based on multiplication/multiply-add/accumulation/multiply-accumulate, pre-addition and pre-subtraction based on multiplication/multiply-add/accumulation/multiply-accumulate, etc. Click "MULTALU" on the IP Core Generator, and a brief introduction to the MULTALU will be displayed.

IP Configuration

Double-click "MULTALU" to open the "IP Customization" window. This window includes “File”, “Mode”, “Multiplier”, “C Operation”, “Cascade”, “Accumulation”, “Common”, as shown in Figure 5-2.

Figure 5-2 IP Customization of MULTALU



1. File Configuration: Configure the information about the generated IP design file. The file configuration MULTALU is similar to that of MULT. For the details, refer to 5.1 MULT File Configuration.
2. Mode: Configure the operation mode of MULTALU27X18
   - Mode Options: Configure the operation mode of MULTALU27X18
     - ± (A*B)
     - ± ((A±D)*B)
     - ± (A*B) ± C
     - ± ((A±D)*B) ± C
     - Accum ± (A*B)
     - Accum ± ((A±D)*B)
     - Accum ± (A*B) ± C
     - Accum ± ((A±D)*B) ± C
     - CASI ± (A*B)

- CASI ± ((A±D)*B)
- CASI ± (A*B) ± C
- CASI ± ((A±D)*B) ± C
- Accum ± (A*B) +CASI
- Accum ± ((A±D)*B) + CASI
- Accum ± (A*B) ± C + CASI
- Accum ± ((A±D)*B) ± C + CASI

3. Multiplier: Configure multiplier, including Data Options, Pre-addition Options, ASEL Option, ADDSUB0 Option, Shift Option, and Register Options
   - The Data Options, Pre-addition Options, and Register Options configuration of MULTALU is similar to that of MULT. For the details, please refer to 5.1 MULT.
   - ASEL Option: Configure A, control mode of SIA source selection
     - Supports dynamic control “Dynamic” and static control “Static”.
     - Dynamic: AESL input port enable
     - Static: Configure “Parallel” (select A) and “Shift” (select SIA)
   - ADDSUB0 Option: Configure control mode of Addition/Subtraction selection of M0/0
     - Supports dynamic control “Dynamic” and static control “Static”.
     - Dynamic: ADDSUB0 input port enable
     - Static: Configure “add” (select Addition) and “sub” (select Subtraction)
   - Shift Option: enable shift out function
4. C Opreation: Configure input C, including Data Options, CSEL Option, ADDSUB1 Option, and Register Options
   - The Data Options and Register Options configuration of MULTALU is similar to that of MULT. For the details, please refer to 5.1 MULT.
   - CSEL Option: Configure the control mode of C/0 source selection
     - Supports dynamic control “Dynamic” and static control “Static”.
   - ADDSUB1 Option: Configure the control mode of Addition/Subtraction selection of M1/C/0
     - Supports dynamic control “Dynamic” and static control “Static”.
     - Dynamic: ADDSUB1 input port enable
     - Static: Configure “add” (select Addition) and “sub” (select

Subtraction)

5. Cascade: Configure cascading input CASI, including CASISEL Option and Register Options
   - The Register Options configuration of MULTALU is similar to that of MULT. For the details, refer to 5.1 MULT.
   - CASISEL Option: Configure the control mode of CASI/0 source selection
     - Supports dynamic control “Dynamic” and static control “Static”.
     - Dynamic: CASISEL input port enable
6. Accumulation: Configure ACCSEL and PRE_LOAD, including ACCSEL Option, Initialization Option, and Register Options
   - The Register Options configuration of MULTALU is similar to that of MULT. For the details, refer to 5.1 MULT.
   - ACCSEL Option: Configure PRE_LOAD, control mode of feedback source selection
     - Supports dynamic control “Dynamic” and static control “Static”.
     - Dynamic: ACCSEL input port enable
     - Static: Configure “PRE_LOAD” (select PRE_LOAD) and “DOUT” (select output feedback)
   - Initialization Option: set the initialization value of PRE_LOAD
     - Preload Value range: 48’h000000000000~48’hFFFFFFFFFFFF
7. Common: Configure output and reset mode, including Data Options and Register Options
   - The Data Options and Register Options configuration of MULTALU is similar to that of MULT. For the details, please refer to 5.1 MULT.
8. Ports Diagram: Display current IP Core configuration. The input/output port and its bit-width update in real time based on the "Options" configuration, as shown in Figure 5-2.

IP Generation Files

After configuration, it will generate three files that are named after the "File Name".

- “gowin_multalu.v” file is a complete Verilog module to generate instance MULTALU, and it is generated according to the IP configuration;
- “gowin_multalu_tmp.v” is the instance template file;
- “gowin_multalu.ipc” file is IP configuration file. You can load the file to

configure the IP.

Note!

If VHDL is selected as the hardware description language, the first two files will be named with .vhd suffix.

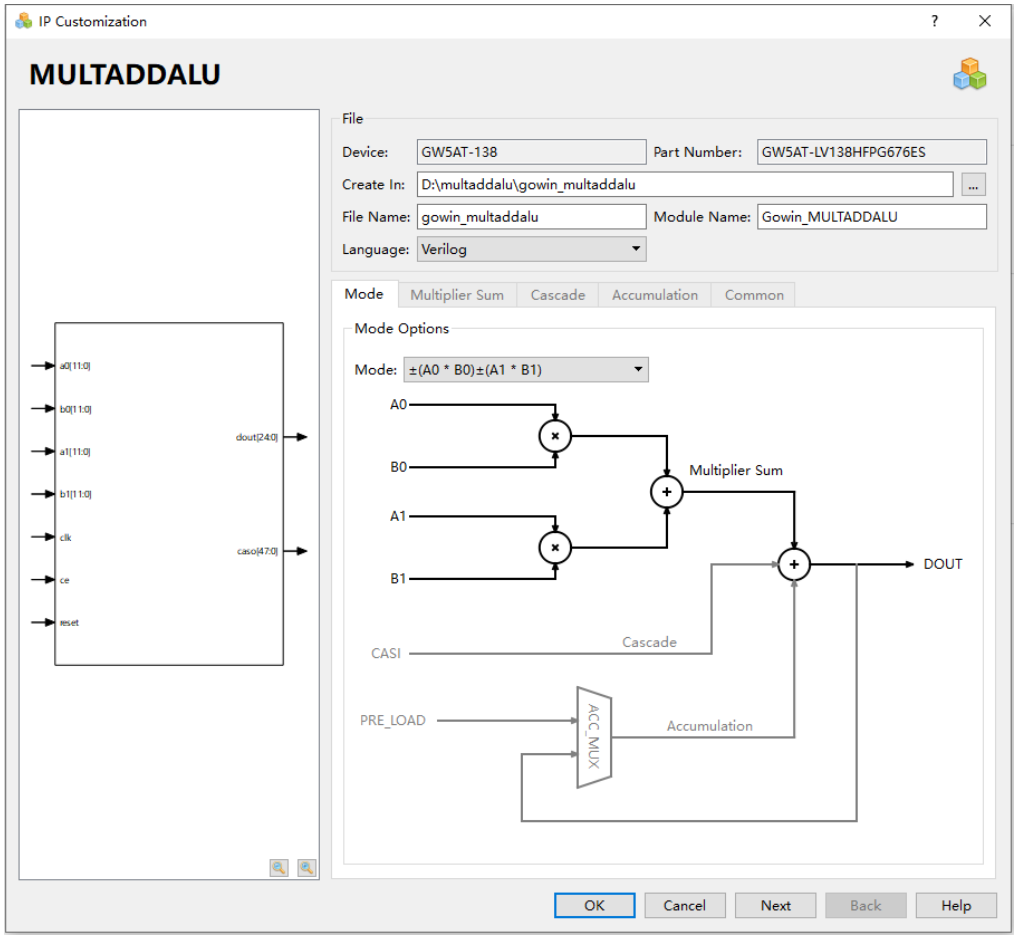
5.3 MULTADDALU

MULTADDALU implements the multiplier quadratic summation or accumulation function. Select DSP in Hard Module in IP Core Generator. Click "MULTADDALU", a brief introduction to the MULTADDALU will be displayed.

IP Configuration

Double-click the "MULTADDALU" to open the "IP Customization" window. This window includes the "File", "Options", port diagram, as shown in Figure 5-3.

Figure 5-3 IP Customization of MULTADDALU



1. File Configuration: Configure the information about the generated IP

design file. The MULTADDALU File configuration is similar to that of MULT. For the details, refer to 5.1 MULT.

2. Mode Option: Configure the operation modes of MULTADDALU.
   - ± (A0 * B0) ± (A1 * B1)
   - CASI ± (A0 * B0) ± (A1 * B1)
   - Accum ± (A0 * B0) ± (A1 * B1)
   - Accum ± CASI ± (A0 * B0) ± (A1 * B1)
3. The parameter configuration of Multiplier Sum, Cascade, Accumulation, and Common is similar to that of MULTALU. For the details, refer to 5.2 MULTALU.
4. Ports Diagram: Display current IP Core configuration. The input/output port and its bit-width update in real time based on the "Options" configuration, as shown in Figure 5-2.

IP Generation Files

After configuration, it will generate three files that are named after the "File Name".

- “gowin_multaddalu.v” file is a complete Verilog module to generate instance MULTADDALU, and it is generated according to the IP configuration;
- “gowin_multaddalu_tmp.v” is the instance template file;
- “gowin_multaddalu.ipc” file is IP configuration file. You can load the file to configure the IP.

Note!

If VHDL is selected as the hardware description language, the first two files will be named with .vhd suffix.


GOWIN
PROGRAMMING FOR THE FUTURE