



# **sysDSP Usage Guide for Nexus Platform**

## **Technical Note**

FPGA-TN-02096-1.1

June 2020

## Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

## Contents

Acronyms in This Document .....	6
1. Introduction .....	7
2. sysDSP Overview .....	7
2.1. Operating Modes and Features .....	8
3. Using sysDSP .....	9
3.1. Primitive Instantiation sysDSP .....	9
3.2. Using Lattice Radiant to Configure and Generate DSP Modules .....	9
3.2.1. IP Catalog Flow .....	9
3.3. Inferencing sysDSP Block .....	11
4. Targeting the sysDSP Block by Instantiating Primitives .....	12
4.1. MULT9X9 – 9 X 9 Multiplier with Optional Input/Output Registers .....	12
4.1.1. MULT9X9 – I/O Port Description .....	13
4.1.2. MULT9X9 – Attribute Description .....	13
4.2. MULTPREADD9X9 – 9X9 Multiplier with Pre-Adder .....	14
4.2.1. 9 X 9 Multiplier with Pre-Adder – I/O Port Description .....	15
4.2.2. 9 X 9 Multiplier with Pre-Adder – Attribute Description .....	15
4.3. MULTADDSUB9X9WIDE – 9 X 9 Wide Multiplier and Adder/Subtractor .....	16
4.3.1. 9 X 9 Wide Multiplier and Adder/Subtractor – I/O Port Description .....	17
4.3.2. 9 X 9 Wide Multiplier and Adder/Subtractor – Attribute Description .....	18
4.4. MULT18X18 – 18 X 18 Multiplier with Optional Input/Output Registers .....	19
4.4.1. MULT18X18 – I/O Port Description .....	20
4.4.2. MULT18X18 – Attribute Description .....	20
4.5. MULTPREADD18X18 – 18 X 18 Multiplier with Pre-Adder .....	21
4.5.1. 18 X 18 Multiplier with Pre-Adder – I/O Port Description .....	22
4.5.2. 18 X 18 Multiplier with Pre-Adder – Attribute Description .....	22
4.6. MULTADDSUB18X18 – 18 X 18 Multiplier and Adder/Subtractor .....	23
4.6.1. 18 X 18 Multiplier and Adder/Subtractor – I/O Port Description .....	24
4.6.2. 18 X 18 Multiplier and Adder/Subtractor – Attribute Description .....	25
4.7. MULTADDSUB18X18WIDE – 18 X 18 Wide Multiplier and Adder/Subtractor .....	26
4.7.1. 18 X 18 Wide Multiplier and Adder/Subtractor – I/O Port Description .....	28
4.7.2. 18 X 18 Wide Multiplier and Adder/Subtractor – Attribute Description .....	28
4.8. MULT18X36 – 18 X 36 Multiplier with Optional Input/Output Registers .....	29
4.8.1. MULT18X36 – I/O Port Description .....	30
4.8.2. MULT18X36 – Attribute Description .....	30
4.9. MULTADDSUB18X36 – 18 X 36 Multiplier and Adder/Subtractor .....	31
4.9.1. MULTADDSUB18X36– I/O Port Description .....	32
4.9.2. MULTADDSUB18X36– Attribute Description .....	33
4.10. MULT36X36 – 36 X 36 Multiplier with Optional Input/Output Registers .....	34
4.10.1. MULT36X36 – I/O Port Description .....	35
4.10.2. MULT36X36 – Attribute Description .....	35
4.11. MULTADDSUB36X36 – 36X36 Multiplier and Adder/Subtractor .....	36
4.11.1. MULTADDSUB36X36– I/O Port Description .....	37
4.11.2. MULTADDSUB36X36– Attribute Description .....	38
4.12. ACC54 – 54-bit Accumulator Component for DSP .....	39
4.12.1. ACC54 – I/O Port Description .....	39
4.12.2. ACC54– Attribute Description .....	40
Appendix A. Instantiating DSP Primitives in HDL .....	42
Verilog Example Showing Snippet of the MULT18X18 Instantiation .....	42
VHDL Example Showing Snippet of the MULT18X18 Instantiation .....	42
Appendix B. HDL Inference for DSP .....	44
VHDL Example to Infer Fully Pipelined Multiplier .....	44
Verilog Example to Infer Fully Pipelined Multiplier .....	45

Technical Support Assistance .....	46
Revision History .....	47

## Figures

Figure 2.1. DSP Block Diagram Overview .....	7
Figure 3.1. DSP Modules in Lattice Radiant .....	9
Figure 3.2. IP Catalog in Lattice Radiant Software .....	10
Figure 3.3. Generating the 18 x 18 Multiplier in Lattice Radiant Software .....	10
Figure 3.4. Generating the 18 x 18 Multiplier in Lattice Radiant Software .....	11
Figure 4.1. MULT9X9 Primitive .....	12
Figure 4.2. MULT9X9 Arithmetic Function .....	12
Figure 4.3. MULT9X9D Primitive .....	14
Figure 4.4. 9 X 9 Multiplier with Pre-Adder Arithmetic Function .....	14
Figure 4.5. MULT18X18C Primitive .....	16
Figure 4.6. MULT18X18C Primitive .....	17
Figure 4.7. 18 X 18 Multiplier with Optional Input/Output Registers Primitive .....	19
Figure 4.8. 18 X 18 Multiplier with Optional Input/Output Registers Arithmetic Function .....	19
Figure 4.9. 18 X 18 Multiplier with Pre-Adder Primitive .....	21
Figure 4.10. 18 X 18 Multiplier with Pre-Adder Arithmetic Function .....	21
Figure 4.11. 18 X 18 Multiplier and Adder/Subtractor Primitive .....	23
Figure 4.12. 18 X 18 Multiplier and Adder/Subtractor Arithmetic Function .....	24
Figure 4.13. 18 X 18 Wide Multiplier and Adder/Subtractor Primitive .....	26
Figure 4.14. 18 X 18 Wide Multiplier and Adder/Subtractor Arithmetic Function .....	27
Figure 4.15. MULT18X36 Primitive .....	29
Figure 4.16. 18 X 36 Multiplier with Optional Input/Output Registers Arithmetic Function .....	29
Figure 4.17. MULTADDSUB18X36 Primitive .....	31
Figure 4.18. 18 X 36 Multiplier and Adder/Subtractor Arithmetic Function .....	32
Figure 4.19. MULT36X36 Primitive .....	34
Figure 4.20. 36 X 36 Multiplier with Optional Input/Output Registers Arithmetic Function .....	34
Figure 4.21. MULTADDSUB36X36 Primitive .....	36
Figure 4.22. 36 X 36 Multiplier and Adder/Subtractor Arithmetic Function .....	37
Figure 4.23. ACC54 Primitive .....	39

## Tables

Table 4.1. MULT9X9 I/O Port Description.....	13
Table 4.2. Attribute Description for MULT9X9 .....	13
Table 4.3. 9 X 9 Multiplier with Pre-Adder I/O Port Description .....	15
Table 4.4. Attribute Description for 9 X 9 Multiplier with Pre-Adder.....	15
Table 4.5. 9 X 9 Wide Multiplier and Adder/Subtractor I/O Port Description.....	17
Table 4.6. Attribute Description for 9 X 9 Wide Multiplier and Adder/Subtractor .....	18
Table 4.7. MULT18X18 I/O Port Description.....	20
Table 4.8. Attribute Description for MULT18X18 .....	20
Table 4.9. 18 X 18 Multiplier with Pre-Adder I/O Port Description .....	22
Table 4.10. Attribute Description for 18 X 18 Multiplier with Pre-Adder.....	22
Table 4.11. 18 X 18 Multiplier and Adder/Subtractor I/O Port Description .....	24
Table 4.12. Attribute Description for 18 X 18 Multiplier and Adder/Subtractor .....	25
Table 4.13. 18 X 18 Wide Multiplier and Adder/Subtractor I/O Port Description .....	28
Table 4.14. Attribute Description for 18 X 18 Wide Multiplier and Adder/Subtractor .....	28
Table 4.15. MULT18X36 I/O Port Description .....	30
Table 4.16. Attribute Description for MULT18X36 .....	30
Table 4.17. MULTADDSUB18X36 I/O Port Description .....	32
Table 4.18. Attribute Description for MULTADDSUB18X36 .....	33
Table 4.19. MULT36X36 I/O Port Description .....	35
Table 4.20. Attribute Description for MULT36X36 .....	35
Table 4.21. MULTADDSUB36X36 I/O Port Description.....	37
Table 4.22. Attribute Description for MULTADDSUB36X36 .....	38
Table 4.23. ACC54 I/O Port Description.....	39
Table 4.24. Attribute Description for ACC54 .....	40

## Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
ALU	Arithmetic logic unit
CIB	Channel Input/Output Bus
DSP	Digital Signal Processing
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
HDL	Hardware Description Language
IIR	Infinite Impulse Response
IP	Intellectual Property
RRH	Remote Radio Head
RTL	Register transfer level

## 1. Introduction

This technical note discusses how to access the features of the Lattice Semiconductor sysDSP™ (Digital Signal Processing) block for the Nexus Platform, which includes the CrossLink™-NX and Certus™-NX devices. The sysDSP features are also described in the [CrossLink-NX Family Data Sheet \(FPGA-DS-02049\)](#) and [Certus-NX Family Data Sheet \(FPGA-DS-02078\)](#).

The Nexus devices are optimized to support high-performance DSP applications, such as wireless base station channel cards, Remote Radio Head (RRH) systems, video and imaging applications, and Fast Fourier Transform (FFT) functions.

## 2. sysDSP Overview

Figure 2.1 shows the DSP Block Diagram at a higher level. As shown, each DSP Block has eight 9-bit pre-adders, pre-adder registers, eight 9-bit multipliers, input registers, pipeline registers, two 54-bit ALU, and output registers.

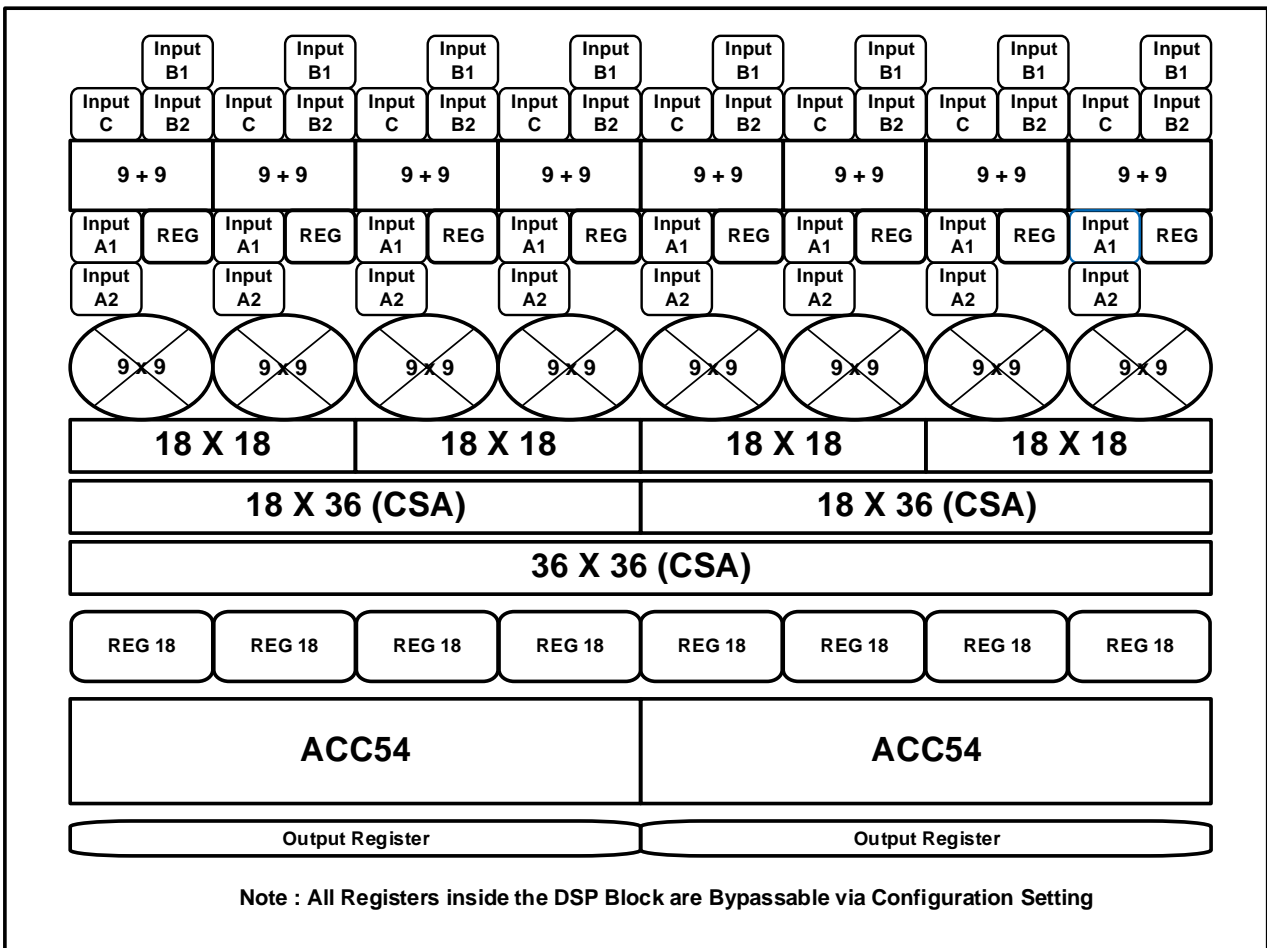


Figure 2.1. DSP Block Diagram Overview

The Nexus DSP IP has strong multiplication support and dedicated architectural features that enable efficient implementation of machine learning, wearable, mobile, video, and other DSP-intensive applications.

The most important architectural features are:

- Dedicated input shift registers, dedicated pipeline registers, and cascaded DSP to support multi-tap FIR/IIR implementation.
- Configured multipliers, which can implement 9x9, 18 X 18, 27x18, 18x36, or 36x36 multiplications.

- Dynamic control shift register with rounding option following multiplier 18 X 18, 18x36 and accumulator 54, optimized for machine learning support.
- Multiple operand accumulators to accelerate data processing speed.

The Nexus DSP IP utilizes a new architecture. While based upon previous Lattice DSP generations, it includes significant innovations in multiplication rounding and accumulation ability.

The sysDSP Blocks are located in rows throughout the device. The programmable resources in a DSP block include the pre-adders, multipliers, ALU, multiplexers, pipeline registers, shift register chain and cascade chain. If the shift out register A is selected, the cascade match register (Casc) is available. The pre-adders and the multipliers can be configured as 9 bits, 18 bits or 36 bits wide and the ALU can be configured as 54 bits or 108 bits wide. Multipliers and accumulators can be configured independently and can be used as stand-alone primitives. However, pre-adders must only be used in conjunction with the associated multiplier block.

## 2.1. Operating Modes and Features

The main features of the Nexus DSP IP are:

- Configured signed and unsigned preadder, can implement eight preadder 9+9, or four preadder 18+18, which can be used to support symmetric FIR filter implementation.
- Configured signed and unsigned multiplier, can implement eight 9 X 9 multipliers, or four 18 x 18 multipliers, or two 18 x 36 multipliers, or one 36x36 multiplier; Dynamic control shifter with rounding option after multiplier 18 x 18, 18 x 36 dedicated for machine learning support.
- Two 54-bit accumulators or one 108-bit accumulator; Dynamic control shifter with rounding option after 54-bit accumulator for machine learning support.
- Dedicated input shifter registers support FIR and IIR filter implementation.
- Input registers, pipeline registers, and output registers make easy to support multi-tap FIR and IIR filter implementation.
- Dedicated dynamic control shifter after multiplier 18 x 18, multiplier 18 x 36, or 54-bit accumulator with the rounding option: round half up, round to zero, and round to infinite.
- Multi operand accumulator for fast accumulation operation
- Flexible cascading across DSP blocks in the same row
- Long FIR/IIR support across multiple DSP
- Minimizes fabric use for common DSP
- Dynamic accumulator input path selection from accumulator output or CIB
- RTL Synthesis friendly synchronous reset on all registers
- Zero hold for all the input registers
- DSP block supports a maximum input clock Fmax of 350 MHz
- MULT36 with input and output registers
- PREADD18, MULT18, ACCU54 with input registers and output registers



## 3. Using sysDSP

The DSP blocks can be used in a number of ways in Nexus devices, as described in the following sections.

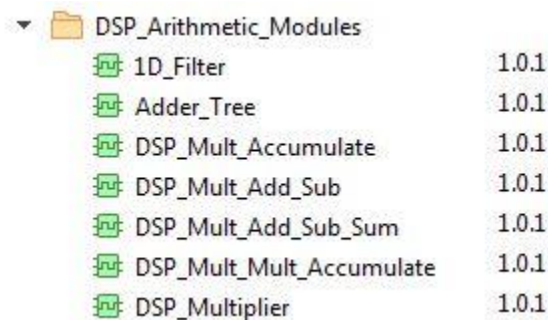
### 3.1. Primitive Instantiation sysDSP

The sysDSP primitives can be directly instantiated in the design. Each of the primitives has a fixed set of attributes that can be customized to meet the design requirements.

An example of the MULT18X18 primitive instantiation is provided in [Appendix A. Instantiating DSP Primitives in HDL](#). You can get the detailed list of the primitives from the synthesis libraries under `cae_library\synthesis` folder under Radiant® installation.

### 3.2. Using Lattice Radiant to Configure and Generate DSP Modules

You can utilize the IP Catalog's Module/IP Block Wizard of the Lattice Radiant® software to easily specify a variety of DSP modules in the design. [Figure 3.1](#) shows a screenshot of the module selection for the DSP modules under IP Catalog in Lattice Radiant software.




▼ DSP_Arithmetic_Modules	
1D_Filter	1.0.1
Adder_Tree	1.0.1
DSP_Mult_Accumulate	1.0.1
DSP_Mult_Add_Sub	1.0.1
DSP_Mult_Add_Sub_Sum	1.0.1
DSP_Mult_Mult_Accumulate	1.0.1
DSP_Multiplier	1.0.1

**Figure 3.1. DSP Modules in Lattice Radiant**

#### 3.2.1. IP Catalog Flow

IP Catalog allows you to generate, create (or open) any of the above modules for Nexus devices. From the Lattice Radiant® software, select Tools > IP Catalog.

Alternatively, you can also click on the  button in the toolbar. This opens the IP Catalog window as shown in [Figure 3.2](#).

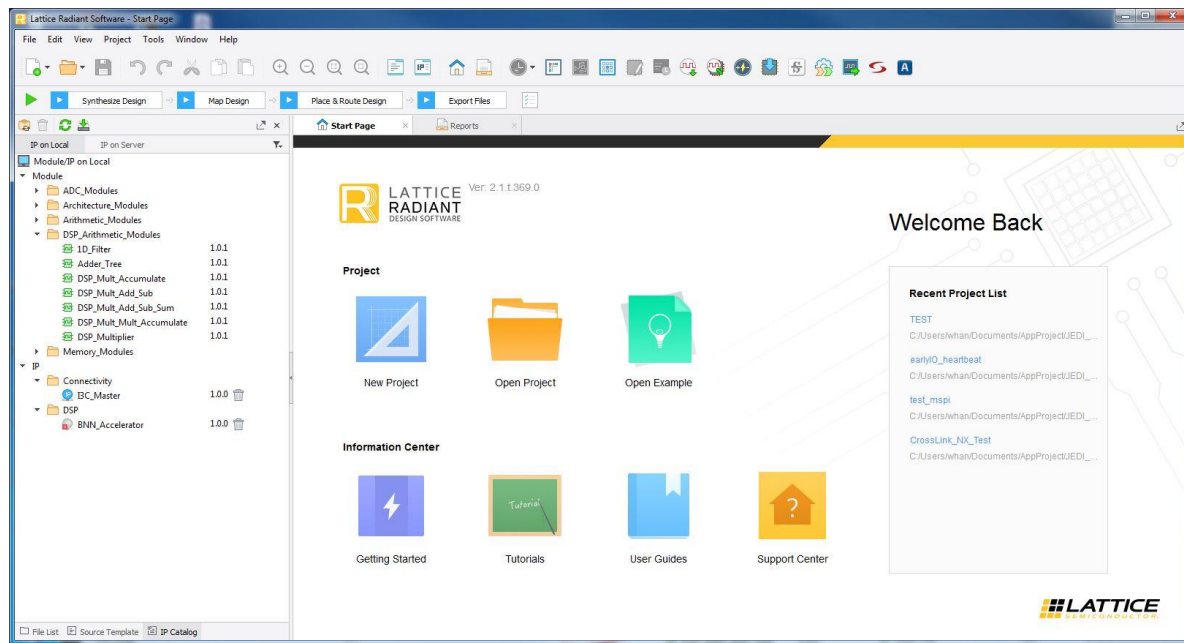


Figure 3.2. IP Catalog in Lattice Radiant Software

The left section of IP Catalog window has the Module tree, and all the sysDSP related modules are under DSP\_Arithmetic\_Modules.

Let us look at an example of generating an 18 X 18 multiplier using the IP Catalog.

To generate an 18 x 18 multiplier in IP Catalog:

1. Double-click **MULT** under the **DSP\_Arithmetic\_Modules**.
2. This opens the Lattice Radiant **Module/IP Block Wizard** window that allows you to specify file name and macro name. Fill out the form, select the preferred language (Verilog or VHDL).
3. Click **Customize**. Fill out the form. Provide **Component Name** in Figure 3.3.
4. Click **Next**.

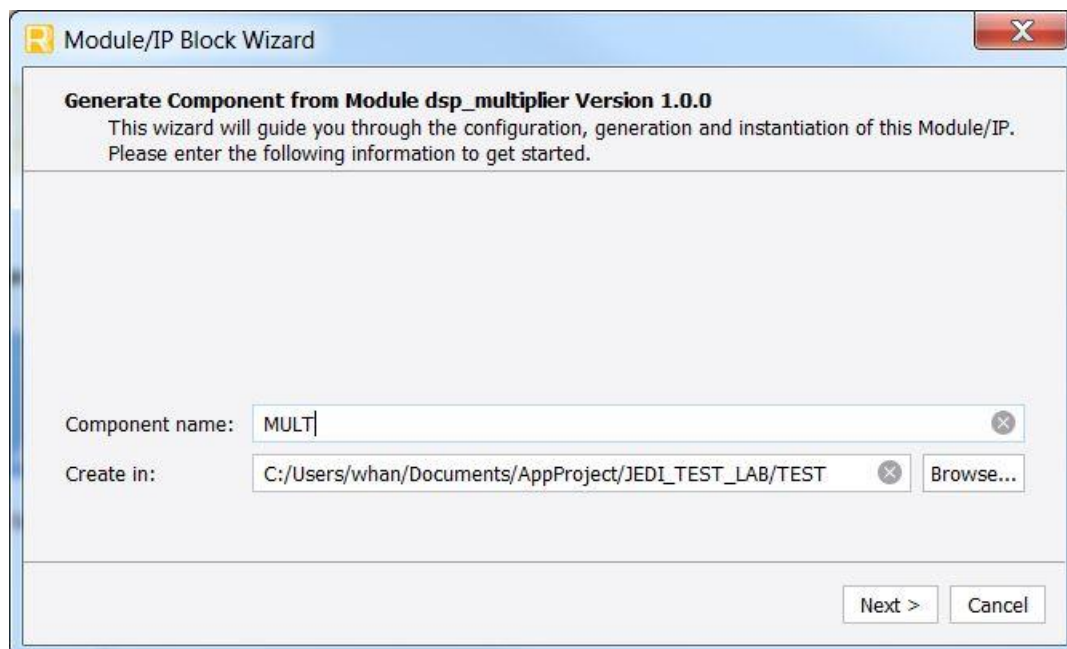
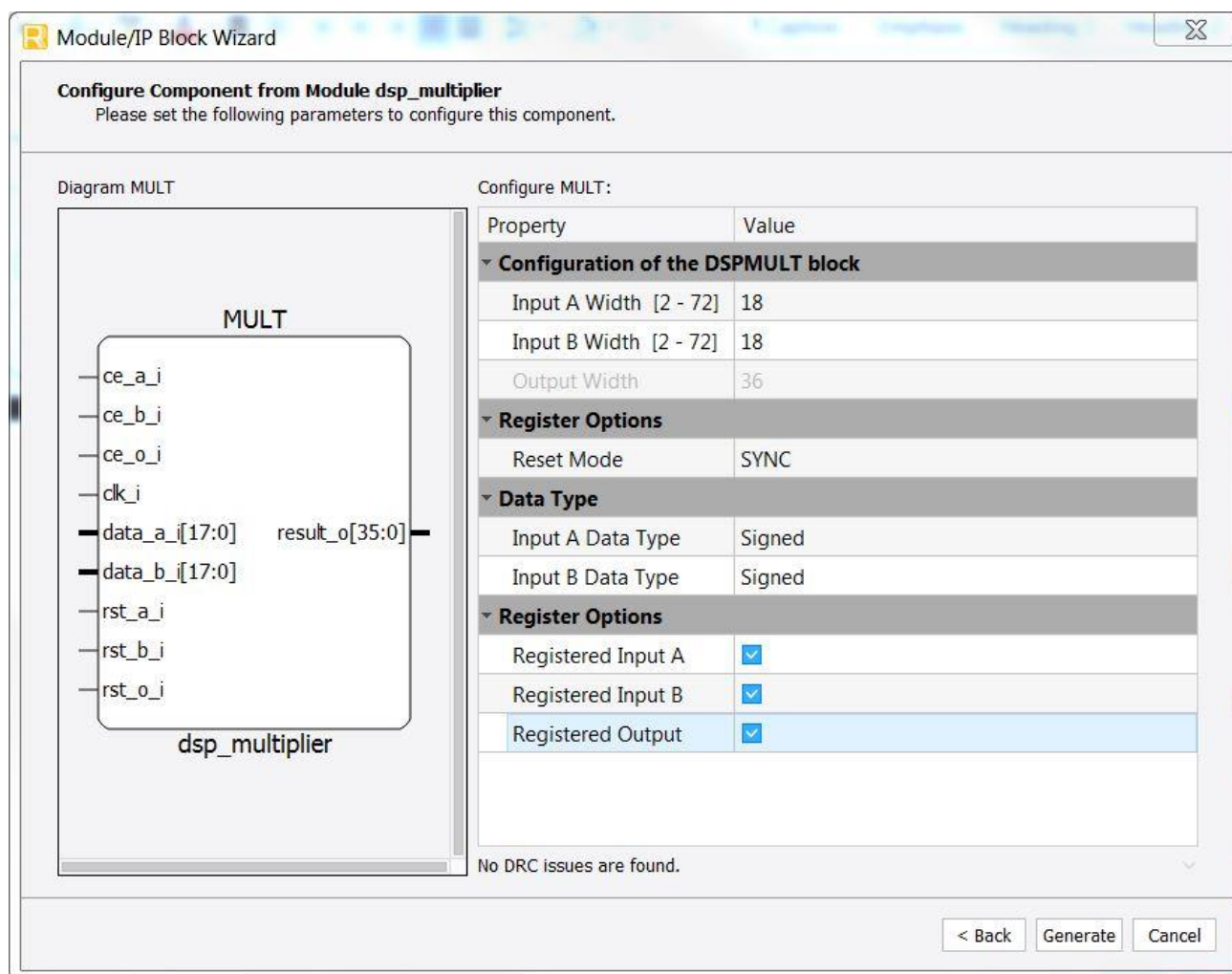


Figure 3.3. Generating the 18 x 18 Multiplier in Lattice Radiant Software

- Fill out the information of the module to generate. This is shown in [Figure 3.4](#).



**Figure 3.4. Generating the 18 x 18 Multiplier in Lattice Radiant Software**

- Once all the right options of the module being generated are filled out, click the **Generate** button. This module, once in the Lattice Radiant project, can be instantiated within other modules.

### 3.3. Inferencing sysDSP Block

You can write a behavioral code for the DSP function such as multiplier, ALU, and others, and the synthesis tool can infer the block into the Nexus sysDSP functions.

An example of the HDL inference for DSP is provided in [Appendix B. HDL Inference for DSP](#).

## 4. Targeting the sysDSP Block by Instantiating Primitives

The sysDSP can be targeted by instantiating the sysDSP primitive into a design. The advantage of instantiating primitives is that it provides access to all the available ports and parameters. The disadvantage of this flow is that the customization requires extra coding and knowledge. This section details the primitives supported by Nexus devices. See [Appendix A. Instantiating DSP Primitives in HDL](#) that shows an HDL example on how to instantiate sysDSP primitives.

The Nexus sysDSP supports all Nexus device primitives, such as MULT9X9, MULT18X18, and ACC54. In addition, several other library primitives are defined to take advantage of the features of the Nexus sysDSP.

Various primitives are available to you, along with the port definitions and attributes are discussed in the sections that follow.

### 4.1. MULT9X9 – 9 X 9 Multiplier with Optional Input/Output Registers

The 9 X 9 multiplier is a widely used module. [Figure 4.1](#) shows the MULT9X9 primitive available in the Nexus devices. The graphical view for the arithmetic function for the 9 X 9 Multiplier is demonstrated in [Figure 4.3](#).

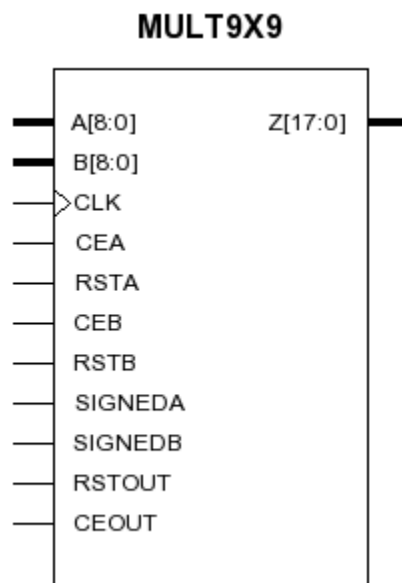


Figure 4.1. MULT9X9 Primitive

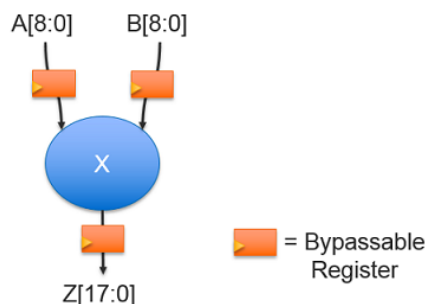


Figure 4.2. MULT9X9 Arithmetic Function

#### 4.1.1. MULT9X9 – I/O Port Description

Table 4.1 describes the list of ports available for MULT9X9 primitive.

**Table 4.1. MULT9X9 I/O Port Description**

Port	Input/Output	Description
A[8:0]	Input	Operand A
B[8:0]	Input	Operand B
CLK	Input	Clock
CEA	Input	Input A clock enable
RSTA	Input	Input A reset
CEB	Input	Input B clock enable
RSTB	Input	Input B reset
SIGNEDA	Input	Operand A signed
SIGNEDB	Input	Operand B signed
RSTOUT	Input	Output register reset
CEOUT	Input	Output clock enable
Z[17:0]	Output	Multiplication result

#### 4.1.2. MULT9X9 – Attribute Description

Table 4.2 describes the attributes for MULT9X9 primitive.

**Table 4.2. Attribute Description for MULT9X9**

Attribute Name	Values	Default Value	User Interface	Description
REGINPUTA	REGISTER, BYPASS	REGISTER	Y	Register operand A
REGINPUTB	REGISTER, BYPASS	REGISTER	Y	Register operand B
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED, DISABLED	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

## 4.2. MULTPREADD9X9 – 9X9 Multiplier with Pre-Adder

The 9X9 Multiplier with Pre-Adder is a widely used module. Figure 4.3 shows the 9 X 9 Multiplier with Pre-Adder primitive available in Nexus devices. The graphical view for the arithmetic function of the 9 X 9 Multiplier with Pre-Adder is demonstrated in Figure 4.4.

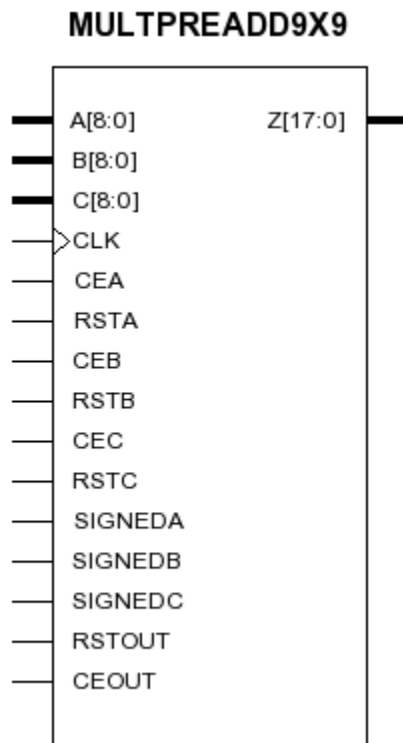


Figure 4.3. MULT9X9D Primitive

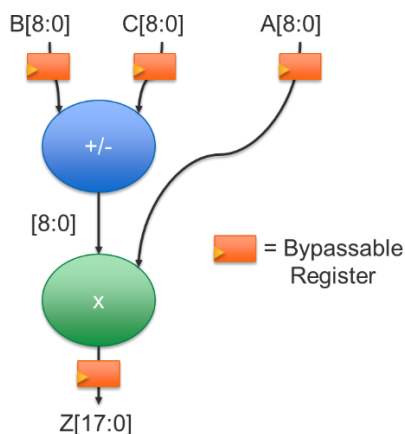


Figure 4.4. 9 X 9 Multiplier with Pre-Adder Arithmetic Function

#### 4.2.1. 9 X 9 Multiplier with Pre-Adder – I/O Port Description

Table 4.3 the list of ports available for 9 X 9 Multiplier with Pre-Adder primitive.

**Table 4.3. 9 X 9 Multiplier with Pre-Adder I/O Port Description**

Port	I/O	Description
A[8:0]	Input	Operand A
B[8:0]	Input	Operand B
C[8:0]	Input	Operand C
CLK	Input	Clock
CEA	Input	Input A clock enable
RSTA	Input	Input A reset
CEB	Input	Input B clock enable
RSTB	Input	Input B reset
CEC	Input	Input C clock enable
RSTC	Input	Input C reset
SIGNEDA	Input	Operand A signed
SIGNEDB	Input	Operand B signed
SIGNEDC	Input	Operand C signed
RSTOUT	Input	Output register reset
CEOUT	Input	Output clock enable
Z[17:0]	Output	Multiplication result

#### 4.2.2. 9 X 9 Multiplier with Pre-Adder – Attribute Description

Table 4.4 describes the attributes for 9 X 9 Multiplier with Pre-Adder primitive.

**Table 4.4. Attribute Description for 9 X 9 Multiplier with Pre-Adder**

Attribute Name	Values	Default Value	User Interface Access	Description
REGINPUTA	REGISTER, BYPASS	REGISTER	Y	Register operand A
REGINPUTB	REGISTER, BYPASS	REGISTER	Y	Register operand B
REGINPUTC	REGISTER, BYPASS	REGISTER	Y	Register operand C
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED, DISABLED	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

### 4.3. MULTADDSUB9X9WIDE – 9 X 9 Wide Multiplier and Adder/Subtractor

The 9 X 9 Wide Multiplier and Adder/Subtractor is a widely used module. Figure 4.5 shows the 9 X 9 Wide Multiplier and Adder/Subtractor primitive available in Nexus devices. The graphical view for the arithmetic function of the 9 X 9 Wide Multiplier and Adder/Subtractor is demonstrated in Figure 4.6.

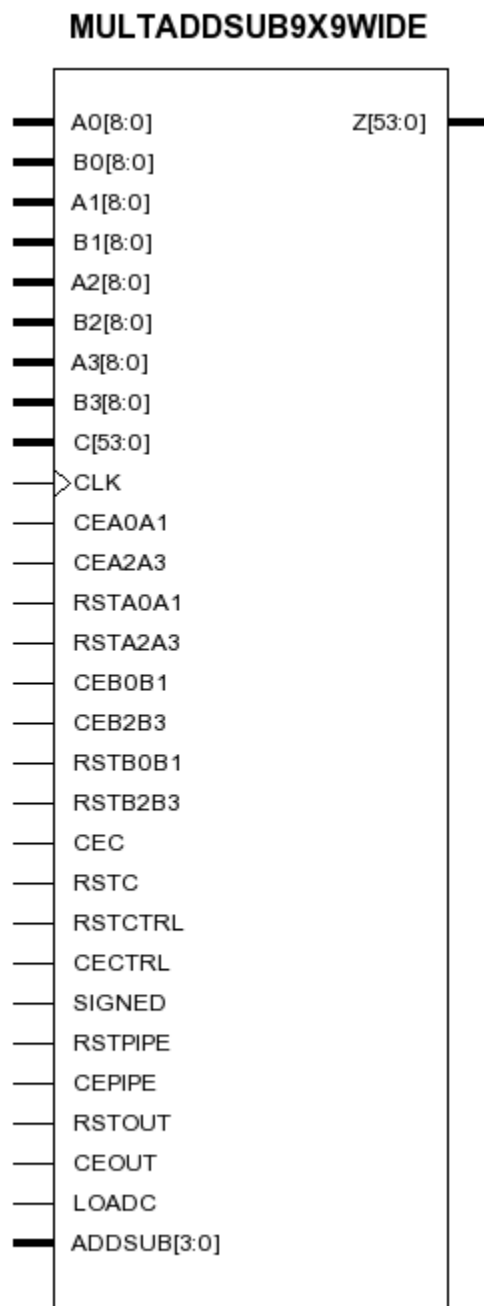


Figure 4.5. MULT18X18C Primitive



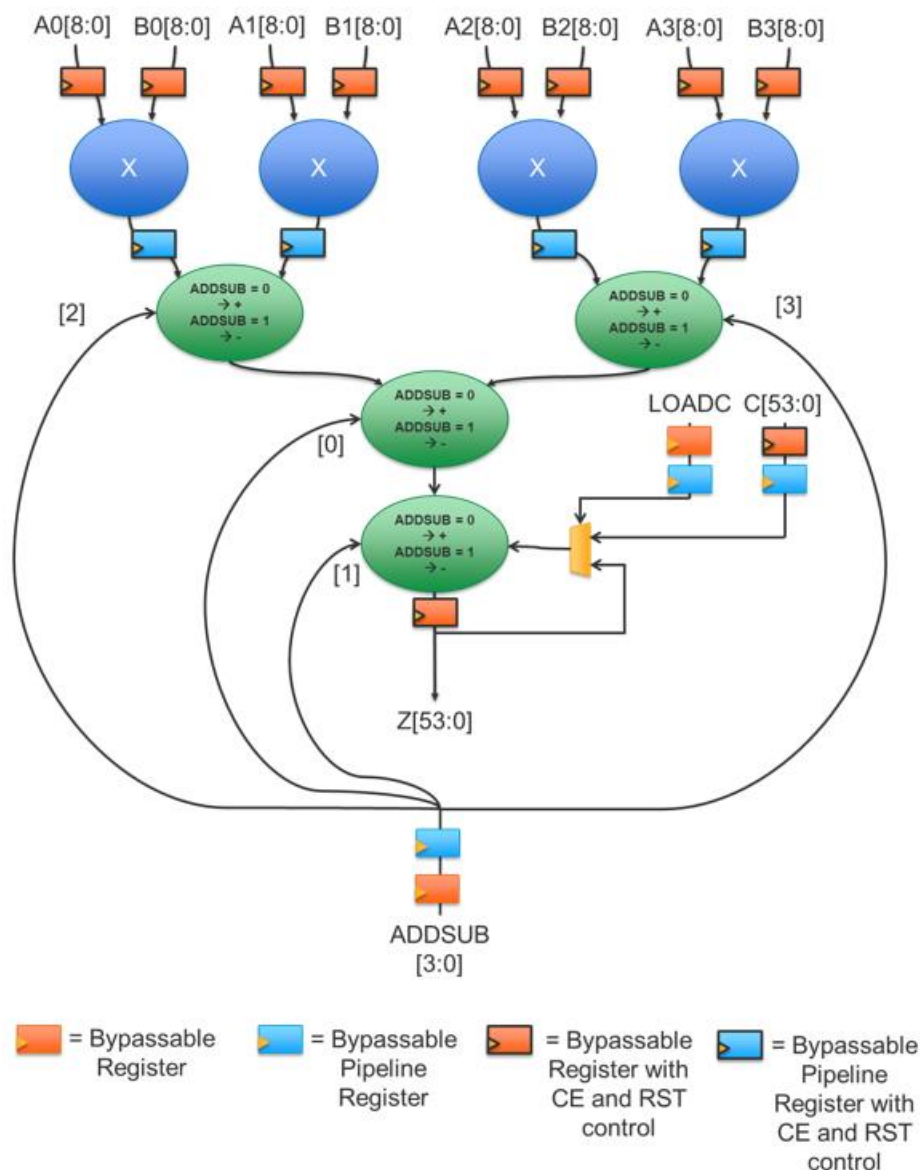


Figure 4.6. MULT18X18C Primitive

#### 4.3.1. 9 X 9 Wide Multiplier and Adder/Subtractor – I/O Port Description

Table 4.5 describes the list of ports available for 9 X 9 Wide Multiplier and Adder/Subtractor primitive.

Table 4.5. 9 X 9 Wide Multiplier and Adder/Subtractor I/O Port Description

Port	I/O	Description
A0[8:0]	Input	Operand A0
B0[8:0]	Input	Operand B0
A1[8:0]	Input	Operand A1
B1[8:0]	Input	Operand B1
A2[8:0]	Input	Operand A2
B2[8:0]	Input	Operand B2
A3[8:0]	Input	Operand A3
B3[8:0]	Input	Operand B3
C[53:0]	Input	Operand C

Port	I/O	Description
CLK	Input	Clock
CEA0A1	Input	Input A0 and A1 clock enable
CEA2A3	Input	Input A2 and A3 clock enable
RSTA0A1	Input	Input A0 and A1 reset
RSTA2A3	Input	Input A2 and A3 reset
CEB0B1	Input	Input B0 and B1 clock enable
CEB2B3	Input	Input B2 and B3 clock enable
RSTB0B1	Input	Input B0 and B1 reset
RSTB2B3	Input	Input B2 and B3 reset
CEC	Input	Input C clock enable
RSTC	Input	Input C reset
RSTCTRL	Input	Input control (addsub, m9addsub, signed, and loadc) reset
CEDTRL	Input	Input control (addsub, m9addsub, signed, and loadc) enable
SIGNED	Input	One SOGNED port for all inputs due to ACC54 HW limitation
RSTPIPE	Input	Pipeline register reset
CEPIPE	Input	Pipeline clock enable
RSTOUT	Input	Output register reset
CEOUT	Input	Output clock enable
LOADC	Input	Load value from input C (active high)
ADDSUB[3:0]	Input	Four bits ADD/SUB control (0=add, 1=subtract)
Z[53:0]	Output	Multiplication result

### 4.3.2. 9 X 9 Wide Multiplier and Adder/Subtractor – Attribute Description

Table 4.6 describes the attributes for 9 X 9 Wide Multiplier and Adder/Subtractor primitive.

**Table 4.6. Attribute Description for 9 X 9 Wide Multiplier and Adder/Subtractor**

Attribute Name	Values	Default Value	User Interface Access	Description
REGINPUTAB0	REGISTER, BYPASS	REGISTER	Y	Register operand A0 and B0
REGINPUTAB1	REGISTER, BYPASS	REGISTER	Y	Register operand A1 and B1
REGINPUTAB2	REGISTER, BYPASS	REGISTER	Y	Register operand A2 and B2
REGINPUTAB3	REGISTER, BYPASS	REGISTER	Y	Register operand A3 and B3
REGINPUTC	REGISTER, BYPASS	REGISTER	Y	Register operand C
REGADDSUB	REGISTER, BYPASS	REGISTER	Y	Register ADDSIUB
REGLOADC	REGISTER, BYPASS	REGISTER	Y	Register LOADC
REGLOADC2	REGISTER, BYPASS	REGISTER	Y	Register LOADC, Second stage
REGPIPELINE	REGISTER, BYPASS	REGISTER	Y	Register between multiplier and
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED, DISABLED	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

## 4.4. MULT18X18 – 18 X 18 Multiplier with Optional Input/Output Registers

The 18 X 18 multiplier is a widely used module. Figure 4.7 shows the MULT18X18 primitive available in Nexus devices. The graphical view for the arithmetic function of the 18 X 18 Multiplier with Optional Input/Output Registers is demonstrated in Figure 4.8.

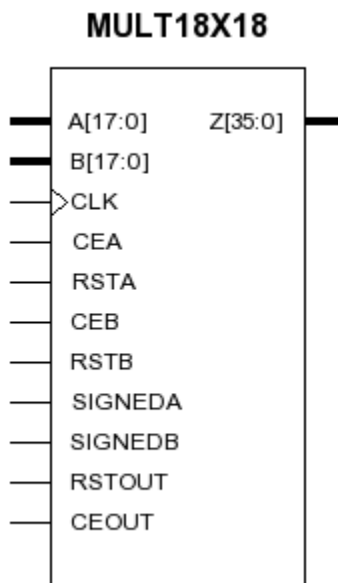


Figure 4.7. 18 X 18 Multiplier with Optional Input/Output Registers Primitive

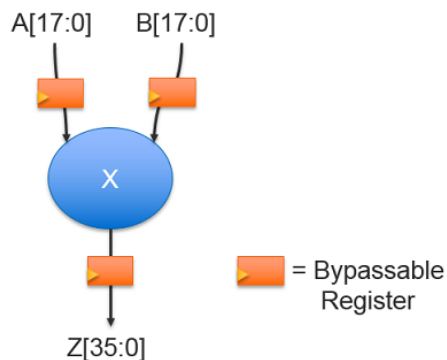


Figure 4.8. 18 X 18 Multiplier with Optional Input/Output Registers Arithmetic Function

#### 4.4.1. MULT18X18 – I/O Port Description

Table 4.7 describes the port list for MULT18X18 primitive.

**Table 4.7. MULT18X18 I/O Port Description**

Port	I/O	Description
A[17:0]	Input	Operand A
B[17:0]	Input	Operand B
CLK	Input	Clock
CEA	Input	Input A clock enable
RSTA	Input	Input A reset
CEB	Input	Input B clock enable
RSTB	Input	Input B reset
SIGNEDA	Input	Operand A signed
SIGNEDB	Input	Operand B signed
RSTOUT	Input	Output register reset
CEOUT	Input	Output clock enable
Z[35:0]	Output	Multiplication result

#### 4.4.2. MULT18X18 – Attribute Description

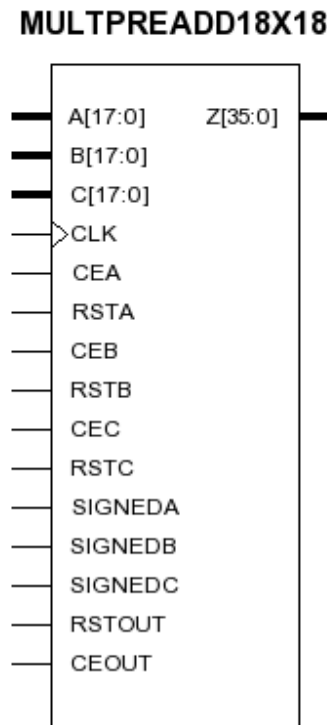
Table 4.8 describes the attributes for MULT18X18 primitive.

**Table 4.8. Attribute Description for MULT18X18**

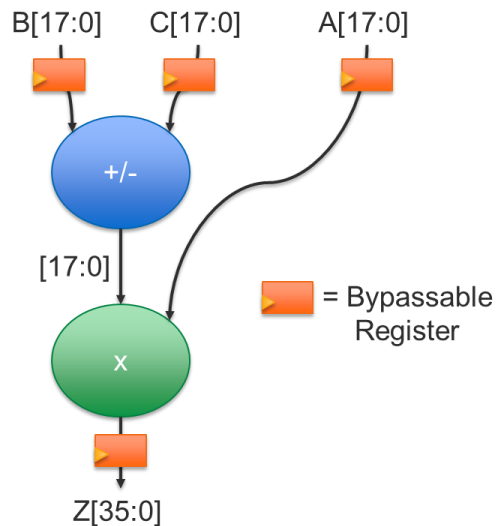
Attribute Name	Values	Default Value	User Interface Access	Description
REGINPUTA	REGISTER, BYPASS	REGISTER	Y	Register operand A
REGINPUTB	REGISTER, BYPASS	REGISTER	Y	Register operand B
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED, DISABLED	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

## 4.5. MULTPREADD18X18 – 18 X 18 Multiplier with Pre-Adder

The 18 X 18 Multiplier with Pre-Adder is a widely used module. Figure 4.9 shows the 18 X 18 Multiplier with Pre-Adder primitive available in Nexus devices. The graphical view for the arithmetic function of the 18 X 18 Multiplier with Pre-Adder is demonstrated in Figure 4.10.



**Figure 4.9. 18 X 18 Multiplier with Pre-Adder Primitive**



**Figure 4.10. 18 X 18 Multiplier with Pre-Adder Arithmetic Function**

#### 4.5.1. 18 X 18 Multiplier with Pre-Adder – I/O Port Description

Table 4.9 describes the list of ports available for 18 X 18 Multiplier with Pre-Adder primitive.

**Table 4.9. 18 X 18 Multiplier with Pre-Adder I/O Port Description**

Port	I/O	Description
A[17:0]	Input	Operand A
B[17:0]	Input	Operand B
C[17:0]	Input	Operand C
CLK	Input	Clock
CEA	Input	Input A clock enable
RSTA	Input	Input A reset
CEB	Input	Input B clock enable
RSTB	Input	Input B reset
CEC	Input	Input C clock enable
RSTC	Input	Input C reset
SIGNEDA	Input	Operand A signed
SIGNEDB	Input	Operand B signed
SIGNEDC	Input	Operand C signed
RSTOUT	Input	Output register reset
CEOUT	Input	Output clock enable
Z[35:0]	Output	Multiplication result

#### 4.5.2. 18 X 18 Multiplier with Pre-Adder – Attribute Description

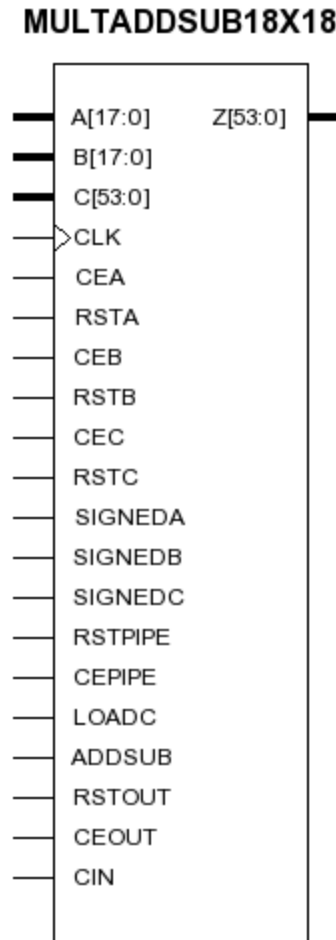
Table 4.10 describes the attributes for 18 X 18 Multiplier with Pre-Adder primitive.

**Table 4.10. Attribute Description for 18 X 18 Multiplier with Pre-Adder**

Attribute Name	Values	Default Value	User Interface Access	Description
REGINPUTA	REGISTER, BYPASS	REGISTER	Y	Register operand A
REGINPUTB	REGISTER, BYPASS	REGISTER	Y	Register operand B
REGINPUTC	REGISTER, BYPASS	REGISTER	Y	Register operand C
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED, DISABLED	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

## 4.6. MULTADDSUB18X18 – 18 X 18 Multiplier and Adder/Subtractor

The 18 X 18 Multiplier and Adder/Subtractor is a widely used module. Figure 4.11 shows the 18 X 18 Multiplier and Adder/Subtractor primitive available in Nexus devices. The graphical view for the arithmetic function of the 18 X 18 Multiplier and Adder/Subtractor is demonstrated in Figure 4.12.



**Figure 4.11. 18 X 18 Multiplier and Adder/Subtractor Primitive**

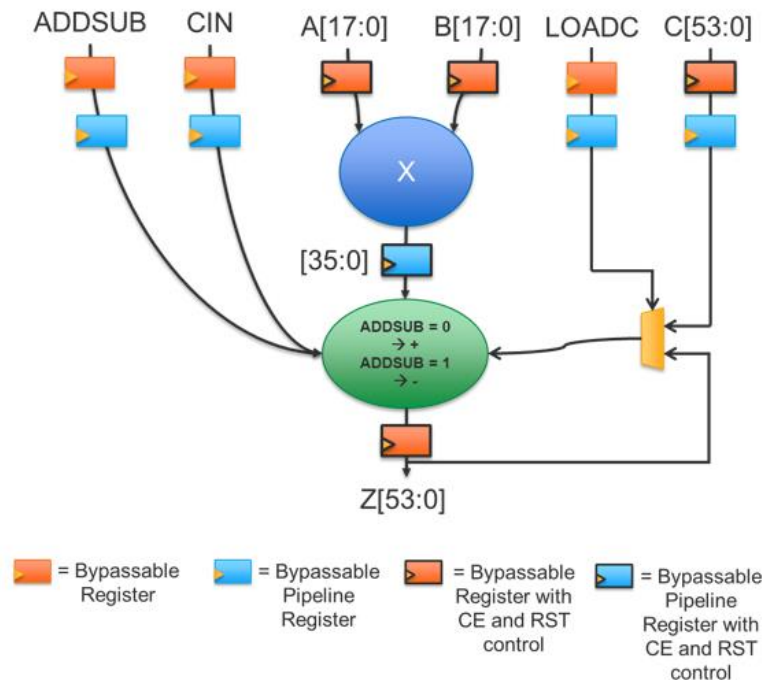


Figure 4.12. 18 X 18 Multiplier and Adder/Subtractor Arithmetic Function

#### 4.6.1. 18 X 18 Multiplier and Adder/Subtractor – I/O Port Description

Table 4.11 describes the list of ports available for 18 X 18 Multiplier and Adder/Subtractor primitive.

Table 4.11. 18 X 18 Multiplier and Adder/Subtractor I/O Port Description

Port	I/O	Description
A[17:0]	Input	Operand A
B[17:0]	Input	Operand B
C[53:0]	Input	Operand C
CLK	Input	Clock
CEA	Input	Input A clock enable
RSTA	Input	Input A register reset
CEB	Input	Input B clock enable
RSTB	Input	Input B register reset
CEC	Input	Input C clock enable
RSTC	Input	Input C reset
SIGNEDA	Input	Operand A signed
SIGNEDB	Input	Operand B signed
SIGNEDC	Input	Operand C signed
RSTPIPE	Input	Pipeline register reset
CEPIPE	Input	Pipeline clock enable
LOADC	Input	Load value from input C (active high)
ADDSUB	Input	ADD/SUB control (0=add, 1=subtract)
RSTOUT	Input	Output register reset
CEOUT	Input	Output clock enable
CIN	Input	Carry in
Z[53:0]	Output	Multiplication result



## 4.6.2. 18 X 18 Multiplier and Adder/Subtractor – Attribute Description

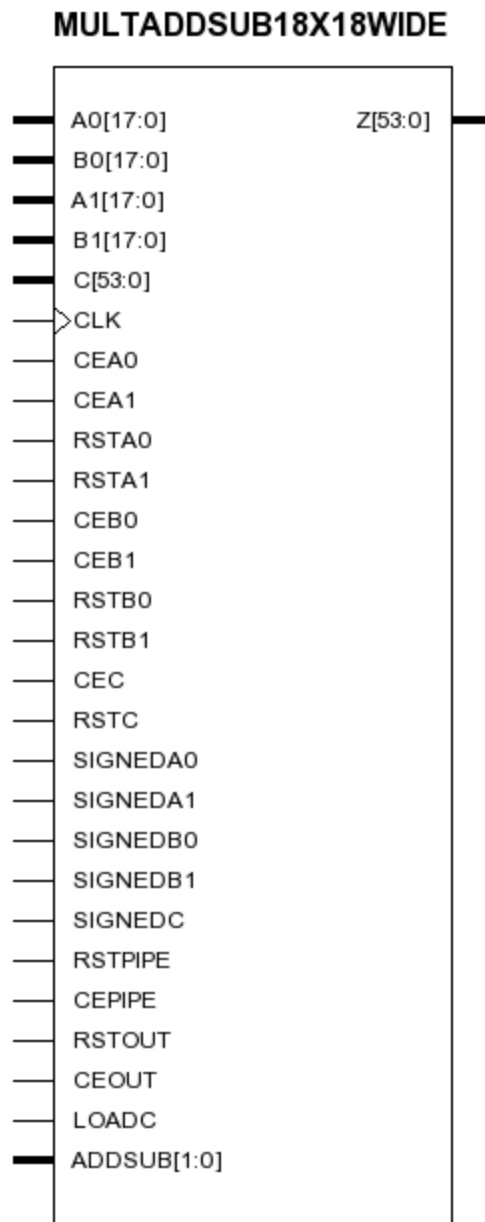
Table 4.12 describes the attributes for 18 X 18 Multiplier and Adder/Subtractor primitive.

**Table 4.12. Attribute Description for 18 X 18 Multiplier and Adder/Subtractor**

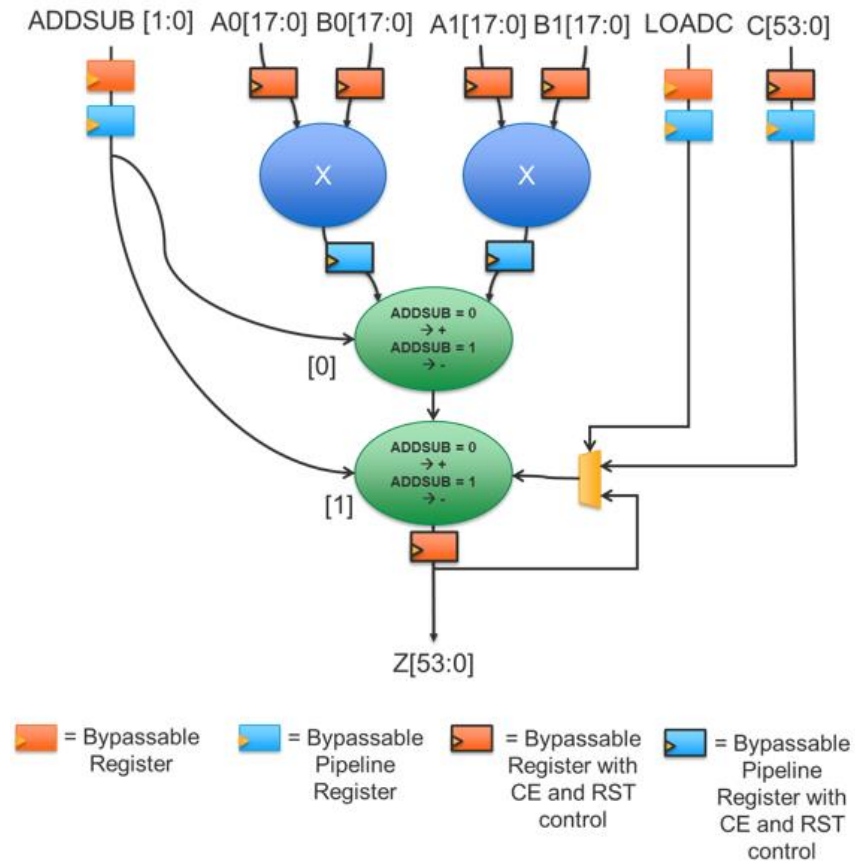
Attribute Name	Values	Default Value	User Interface Access	Description
REGINPUTA	REGISTER, BYPASS	REGISTER	Y	Register operand A
REGINPUTB	REGISTER, BYPASS	REGISTER	Y	Register operand B
REGINPUTC	REGISTER, BYPASS	REGISTER	Y	Register operand C
REGADDSUB	REGISTER, BYPASS	REGISTER	Y	Register ADDSIUB
REGLOADC	REGISTER, BYPASS	REGISTER	Y	Register LOADC
REGCIN	REGISTER, BYPASS	REGISTER	Y	Register CIN
REGPIPELINE	REGISTER, BYPASS	REGISTER	Y	Register between multiplier and
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED, DISABLED	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

## 4.7. MULTADDSUB18X18WIDE – 18 X 18 Wide Multiplier and Adder/Subtractor

The 18 X 18 Wide Multiplier and Adder/Subtractor is a widely used module. Figure 4.13 shows the 18 X 18 Wide Multiplier and Adder/Subtractor primitive available in Nexus devices. The graphical view for the arithmetic function of the 18 X 18 Wide Multiplier and Adder/Subtractor is demonstrated in Figure 4.14.



**Figure 4.13. 18 X 18 Wide Multiplier and Adder/Subtractor Primitive**



**Figure 4.14. 18 X 18 Wide Multiplier and Adder/Subtractor Arithmetic Function**

#### 4.7.1. 18 X 18 Wide Multiplier and Adder/Subtractor – I/O Port Description

Table 4.13 describes the list of ports available for 18 X 18 Wide Multiplier and Adder/Subtractor primitive.

**Table 4.13. 18 X 18 Wide Multiplier and Adder/Subtractor I/O Port Description**

Port	I/O	Description
A0[17:0]	Input	Operand A0
B0[17:0]	Input	Operand B0
A1[17:0]	Input	Operand A1
B1[17:0]	Input	Operand B1
C[53:0]	Input	Operand C
CLK	Input	Clock
CEA0	Input	Input A0 clock enable
CEA1	Input	Input A1 clock enable
RSTA0	Input	Input A0 register reset
RSTA1	Input	Input A1 register reset
CEB0	Input	Input B0 clock enable
CEB1	Input	Input B1 clock enable
RSTB0	Input	Input B0 register reset
RSTB1	Input	Input B1 register reset
CEC	Input	Input C clock enable
RSTC	Input	Input C reset
SIGNEDA0	Input	Operand A0 signed
SIGNEDA1	Input	Operand A1 signed
SIGNEDB0	Input	Operand B0 signed
SIGNEDB1	Input	Operand B1 signed
SIGNEDC	Input	Operand C signed
RSTPIPE	Input	Pipeline register reset
CEPIPE	Input	Pipeline clock enable
RSTOUT	Input	Output register reset
CEOUT	Input	Output clock enable
LOADC	Input	Load value from input C (active high)
ADDSUB[1:0]	Input	Two bits ADD/SUB control (0=add, 1=subtract)
Z[53:0]	Output	Multiplication result

#### 4.7.2. 18 X 18 Wide Multiplier and Adder/Subtractor – Attribute Description

Table 4.14 describes the attributes for 18 X 18 Wide Multiplier and Adder/Subtractor primitive.

**Table 4.14. Attribute Description for 18 X 18 Wide Multiplier and Adder/Subtractor**

Attribute Name	Values	Default Value	User Interface Access	Description
REGINPUTAB0	REGISTER, BYPASS	REGISTER	Y	Register operand A0 and B0
REGINPUTAB1	REGISTER, BYPASS	REGISTER	Y	Register operand A1 and B1
REGINPUTC	REGISTER, BYPASS	REGISTER	Y	Register operand C
REGADDSUB	REGISTER, BYPASS	REGISTER	Y	Register ADDSUB
REGLOADC	REGISTER, BYPASS	REGISTER	Y	Register LOADC
REGPIPELINE	REGISTER, BYPASS	REGISTER	Y	Register between multiplier and
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED, DISABLED	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

## 4.8. MULT18X36 – 18 X 36 Multiplier with Optional Input/Output Registers

The Nexus DSP also includes the 18 X 36 multiplier natively. Figure 4.15 shows the MULT18X36 primitive available in Nexus devices. The graphical view for the arithmetic function of the 18 X 36 Multiplier with Optional Input/Output Registers is demonstrated in Figure 4.16. .

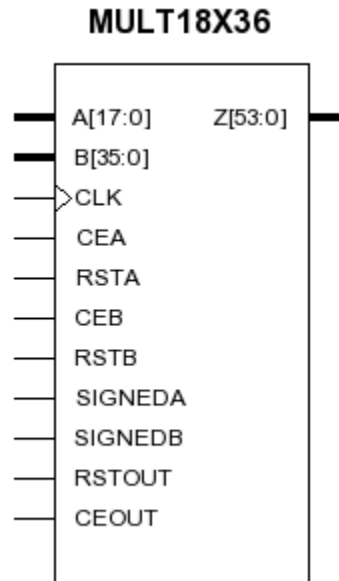


Figure 4.15. MULT18X36 Primitive

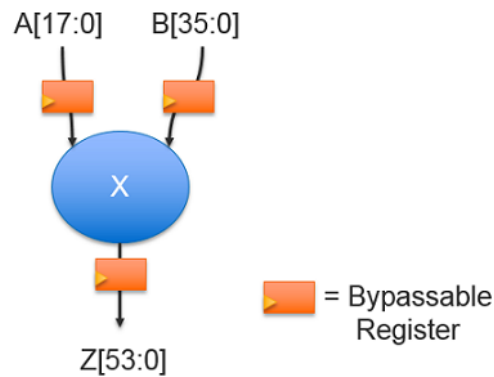


Figure 4.16. 18 X 36 Multiplier with Optional Input/Output Registers Arithmetic Function

#### 4.8.1. MULT18X36 – I/O Port Description

Table 4.15 describes the port list for MULT18X36 primitive.

**Table 4.15. MULT18X36 I/O Port Description**

Port	I/O	Description
A[17:0]	Input	Operand A
B[35:0]	Input	Operand B
CLK	Input	Clock
CEA	Input	Input A clock enable
RSTA	Input	Input A reset
CEB	Input	Input B clock enable
RSTB	Input	Input B reset
SIGNEDA	Input	Operand A signed
SIGNEDB	Input	Operand B signed
RSTOUT	Input	Output register reset
CEOUT	Input	Output clock enable
Z[53:0]	Output	Multiplication result

#### 4.8.2. MULT18X36 – Attribute Description

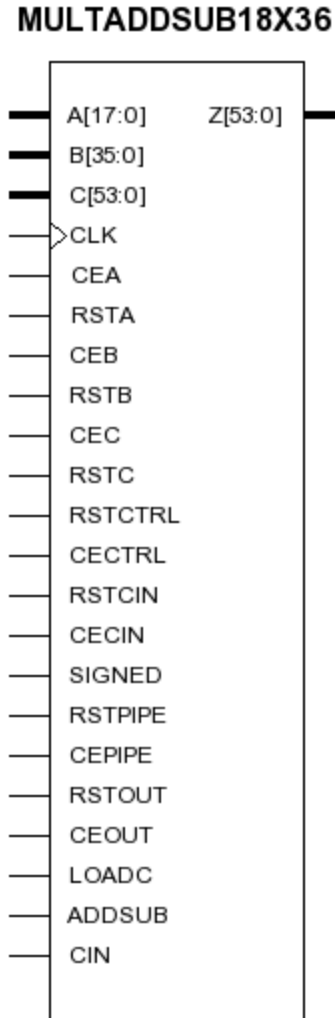
Table 4.16 describes the attributes for MULT18X36 primitive.

**Table 4.16. Attribute Description for MULT18X36**

Attribute Name	Values	Default Value	User Interface Access	Description
REGINPUTA	REGISTER, BYPASS	REGISTER	Y	Register operand A
REGINPUTB	REGISTER, BYPASS	REGISTER	Y	Register operand B
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED, DISABLED	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

## 4.9. MULTADDSUB18X36 – 18 X 36 Multiplier and Adder/Subtractor

Similar to its MULT18X36, the Nexus DSP also includes the 18 X 36 Multiplier and Adder/Subtractor - MULTADDSUB18X36. Figure 4.17 shows the MULTADDSUB18X36 primitive. The graphical view for the arithmetic function of the 18 X 36 Multiplier and Adder/Subtractor is demonstrated in Figure 4.18 shows the PRADD9A primitive.



**Figure 4.17. MULTADDSUB18X36 Primitive**

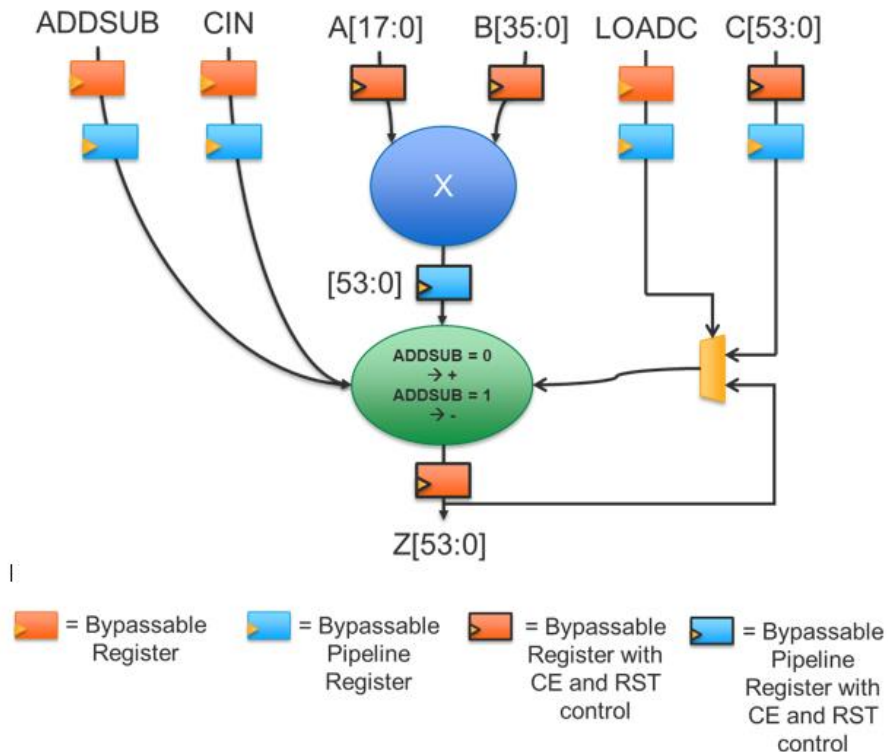


Figure 4.18. 18 X 36 Multiplier and Adder/Subtractor Arithmetic Function

#### 4.9.1. MULTADDSUB18X36– I/O Port Description

Table 4.17 describes the port list for MULTADDSUB18X36 primitive.

Table 4.17. MULTADDSUB18X36 I/O Port Description

Port	I/O	Description
A[17:0]	Input	Operand A
B[35:0]	Input	Operand B
C[53:0]	Input	Operand C
CLK	Input	Clock
CEA	Input	Input A clock enable
RSTA	Input	Input A register reset
CEB	Input	Input B clock enable
RSTB	Input	Input B register reset
CEC	Input	Input C clock enable
RSTC	Input	Input C register reset
SIGNEDA	Input	Operand A signed
SIGNEDB	Input	Operand B signed
SIGNEDC	Input	Operand C signed
RSTPIPE	Input	Pipeline register reset
CEPIPE	Input	Pipeline clock enable
RSTOUT	Input	Output register reset
CEOUT	Input	Output register enable
LOADC	Input	Load value from input C. Active high
ADDSUB	Input	ADD/SUB control (0=add, 1=subtract)
CIN	Input	Carry in
Z[53:0]	Output	Multiplication result



## 4.9.2. MULTADDSUB18X36– Attribute Description

Table 4.18 describes the attributes for MULTADDSUB18X36 primitive.

**Table 4.18. Attribute Description for MULTADDSUB18X36**

Attribute Name	Values	Default Value	User Interface Access	Description
REGINPUTA	REGISTER, BYPASS	REGISTER	Y	Register operand A
REGINPUTB	REGISTER, BYPASS	REGISTER	Y	Register operand B
REGINPUTC	REGISTER, BYPASS	REGISTER	Y	Register operand C
REGADDSUB	REGISTER, BYPASS	REGISTER	Y	Register ADDSUB
REGLOADC	REGISTER, BYPASS	REGISTER	Y	Register LOADC
REGLOADC2	REGISTER, BYPASS	REGISTER	Y	Register LOADC, Second stage
REGCIN	REGISTER, BYPASS	REGISTER	Y	Register CIN
REGPIPELINE	REGISTER, BYPASS	REGISTER	Y	Register between multiplier and
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED,	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

## 4.10. MULT36X36 – 36 X 36 Multiplier with Optional Input/Output Registers

The Nexus DSP also includes the 36 X 36 multiplier natively. Figure 4.19 shows the MULT36X36 primitive available in Nexus devices. The graphical view for the arithmetic function of the 36 X 36 Multiplier with Optional Input/Output Registers is demonstrated in Figure 4.20.

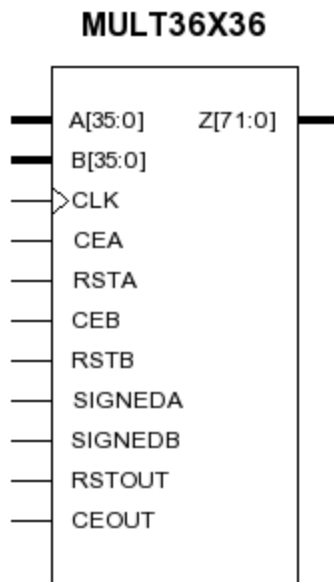


Figure 4.19. MULT36X36 Primitive

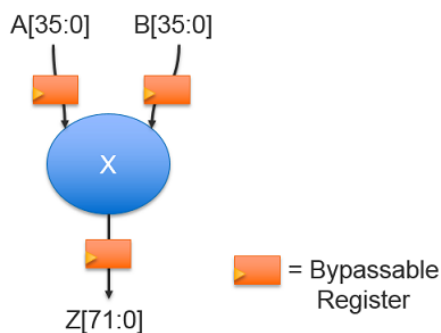


Figure 4.20. 36 X 36 Multiplier with Optional Input/Output Registers Arithmetic Function

#### 4.10.1. MULT36X36 – I/O Port Description

Table 4.19 describes the port list for MULT36X36 primitive.

**Table 4.19. MULT36X36 I/O Port Description**

Port	I/O	Description
A[35:0]	Input	Operand A
B[35:0]	Input	Operand B
CLK	Input	Clock
CEA	Input	Input A clock enable
RSTA	Input	Input A reset
CEB	Input	Input B clock enable
RSTB	Input	Input B reset
SIGNEDA	Input	Operand A signed
SIGNEDB	Input	Operand B signed
RSTOUT	Input	Output register reset
CEOUT	Input	Output clock enable
Z[71:0]	Output	Multiplication result

#### 4.10.2. MULT36X36 – Attribute Description

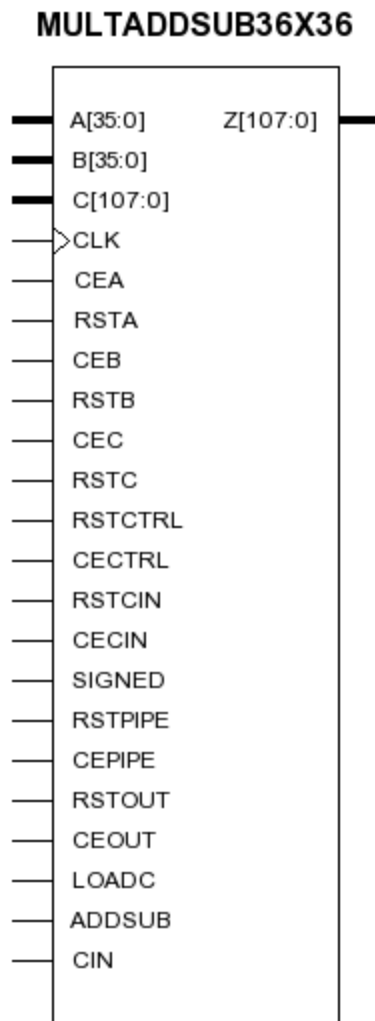
Table 4.20 describes the attributes for MULT36X36 primitive.

**Table 4.20. Attribute Description for MULT36X36**

Attribute Name	Values	Default Value	User Interface Access	Description
REGINPUTA	REGISTER, BYPASS	REGISTER	Y	Register operand A
REGINPUTB	REGISTER, BYPASS	REGISTER	Y	Register operand B
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED, DISABLED	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

## 4.11.MULTADDSUB36X36 – 36X36 Multiplier and Adder/Subtractor

Similar to its MULT36X36, the Nexus DSP also includes the 36 X 36 Multiplier and Adder/Subtractor – MULTADDSUB36X36. Figure 4.21 shows the MULTADDSUB36X36 primitive. The graphical view for the arithmetic function of the 36 X 36 Multiplier and Adder/Subtractor is demonstrated in Figure 4.22.



**Figure 4.21. MULTADDSUB36X36 Primitive**

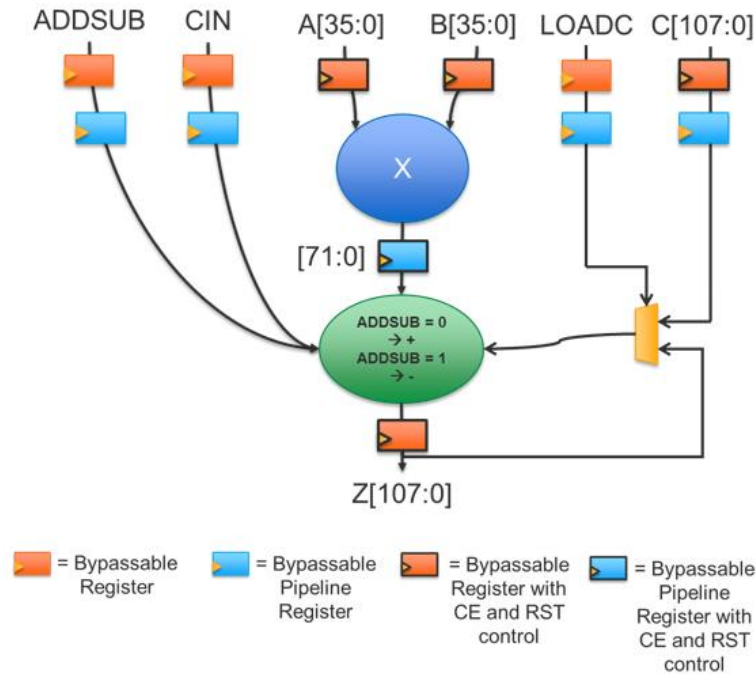


Figure 4.22. 36 X 36 Multiplier and Adder/Subtractor Arithmetic Function

#### 4.11.1. MULTADDSUB36X36– I/O Port Description

Table 4.21 describes the port list for MULTADDSUB36X36 primitive.

Table 4.21. MULTADDSUB36X36 I/O Port Description

Port	I/O	Description
A[35:0]	Input	Operand A
B[35:0]	Input	Operand B
C[107:0]	Input	Operand C
CLK	Input	Clock
CEA	Input	Input A clock enable
RSTA	Input	Input A register reset
CEB	Input	Input B clock enable
RSTB	Input	Input B register reset
CEC	Input	Input C clock enable
RSTC	Input	Input C register reset
RSTCTRL	Input	Input control (addsub, m9addsub, signed, and loadc) reset
CECTRL	Input	Input control (addsub, m9addsub, signed, and loadc) enable
RSTCIN	Input	Input carry-in reset
CECIN	Input	Input carry-in enable
SIGNED	Input	One SIGNED port for all inputs due to ACC54 HW limitation
RSTPIPE	Input	Pipeline register reset
CEPIPE	Input	Pipeline clock enable
RSTOUT	Input	Output register reset
CEOUT	Input	Output register enable
LOADC	Input	Load value from input C. Active high
ADDSUB	Input	ADD/SUB control (0=add, 1=subtract)
CIN	Input	Carry in
Z[107:0]	Output	Multiplication result

#### 4.11.2. MULTADDSUB36X36– Attribute Description

Table 4.22 describes the attributes for MULTADDSUB36X36primitive.

**Table 4.22. Attribute Description for MULTADDSUB36X36**

Attribute Name	Values	Default Value	User Interface Access	Description
REGINPUTA	REGISTER, BYPASS	REGISTER	Y	Register operand A
REGINPUTB	REGISTER, BYPASS	REGISTER	Y	Register operand B
REGINPUTC	REGISTER, BYPASS	REGISTER	Y	Register operand C
REGADDSUB	REGISTER, BYPASS	REGISTER	Y	Register ADDSUB
REGLOADC	REGISTER, BYPASS	REGISTER	Y	Register LOADC
REGLOADC2	REGISTER, BYPASS	REGISTER	Y	Register LOADC, Second stage
REGCIN	REGISTER, BYPASS	REGISTER	Y	Register CIN
REGPIPELINE	REGISTER, BYPASS	REGISTER	Y	Register between multiplier and
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED,	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

## 4.12. ACC54 – 54-bit Accumulator Component for DSP

Figure 4.23 shows the ACC54 primitive.

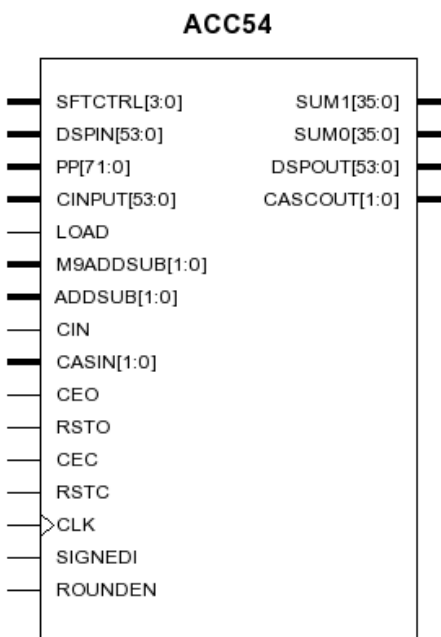


Figure 4.23. ACC54 Primitive

### 4.12.1. ACC54 – I/O Port Description

Table 4.23 describes the port list for ACC54 primitive.

Table 4.23. ACC54 I/O Port Description

Port	I/O	Description
SFTCTRL[3:0]	Input	Arithmetical right shift 0 - 15
DSPIN[53:0]	Input	Previous ACC54 data in
PP[71:0]	Input	Input form 4 groups of REG18
CINPUT[53:0]	Input	Input form 4 groups of REG18
LOAD	Input	Dynamic load control for C register
M9ADDSUB[1:0]	Input	Dynamic add/sub control for state 1 adder. 1=subtraction
ADDSUB[1:0]	Input	Dynamic add/sub control for 54-bit adder
CIN	Input	Carry in from CIB
CASIN[1:0]	Input	Carry in from previous ACC54
CEO	Input	Clock enable for output register
RSTO	Input	Synchronous reset for output register
CEC	Input	Clock enable for C register
RSTC	Input	Synchronous reset for C register
CLK	Input	Clock for output register
SIGNEDI	Input	Signed input
ROUNDEN	Input	1=ACCU54 round enable
SUM1[35:0]	Output	Results Output to CIB
SUM0[35:0]	Output	Results Output to CIB
DSPOUT[53:0]	Output	Output to next ACC54
CASCOUT[1:0]	Output	Carry output to next ACC54

#### 4.12.2. ACC54– Attribute Description

Table 4.24 describes the attributes for ACC54 primitive.

**Table 4.24. Attribute Description for ACC54**

Attribute Name	Values	Default Value	User Interface Access	Description
SIGN	DISABLED, ENABLED	DISABLED	N	Static sign operation enable for ACCU54
M9ADDSUB_CTRL	ADDITION, ADDSUB, SUBADD, SUBTRACTION	ADDITION	N	Static add/sub control for stage 1 adder
ADDSUB_CTRL	ADD_ADD_CTRL_54_BIT_ADDER, SUB_ADD_CTRL_54_BIT_ADDER, ADD_SUB_CTRL_54_BIT_ADDER, SUB_SBU_CTRL_54_BIT_ADDER	ADD_ADD_CTRL_54_BIT_ADDER	N	Static add/sub control for 54-bit adder
STATICOPCODE_EN	DISABLED, ENABLED	DISABLED	N	Static Opcode enable
OUTREGBYPS	BYPASS, REGISTER	BYPASS	N	Output register bypass
GSR	DISABLED, ENABLED	DISABLED	N	GSRN enable
PROGCONST	{54{1'b0}}	{54{1'b0}}	N	Constant value for C
CONSTSEL	BYPASS, SELECT	BYPASS	N	Select constant value for C
DSPCASCADE	DISABLED, ENABLED	DISABLED	N	DSP cascaded enable
ACC108CASCADE	BYPASSCASCADE, CASCADE2ACCU54TOFORMACCU108	BYPASSCASCADE	N	Cascade two ACCU54 to form ACCU108
ACCUMODE	MODE0, MODE1, MODE2, MODE3, MODE4, MODE5, MODE6, MODE7	MODE0	N	Accumulator mode
ACCUBYPS	USED, BYPASS	USED	N	Bypass ACC54
CREGBYPS1	REGISTER, BYPASS	REGISTER	N	Bypass C register 1
CREGBYPS2	REGISTER, BYPASS	REGISTER	N	Bypass C register 2
CREGBYPS3	REGISTER, BYPASS	REGISTER	N	Bypass C register 3
CINREGBYPS1	REGISTER, BYPASS	REGISTER	N	Bypass CIN register 1
CINREGBYPS2	REGISTER, BYPASS	REGISTER	N	Bypass CIN register 2
CINREGBYPS3	REGISTER, BYPASS	REGISTER	N	Bypass CIN register 3
LOADREGBYPS1	REGISTER, BYPASS	REGISTER	N	Bypass load register 1
LOADREGBYPS2	REGISTER, BYPASS	REGISTER	N	Bypass load register 2
LOADREGBYPS3	REGISTER, BYPASS	REGISTER	N	Bypass load register 3
M9ADDSUBREGBYPS1	REGISTER, BYPASS	REGISTER	N	Bypass M9AddSub register 1
M9ADDSUBREGBYPS2	REGISTER, BYPASS	REGISTER	N	Bypass M9AddSub register 2
M9ADDSUBREGBYPS3	REGISTER, BYPASS	REGISTER	N	Bypass M9AddSub register 3
ADDSUBSIGNREGBYPS1	REGISTER, BYPASS	REGISTER	N	Bypass AddSubSign register 1



Attribute Name	Values	Default Value	User Interface Access	Description
ADDSUBSIGNREGBYPS2	REGISTER, BYPASS	REGISTER	N	Bypass AddSubSign register 2
ADDSUBSIGNREGBYPS3	REGISTER, BYPASS	REGISTER	N	Bypass AddSubSign register 3
ROUNDHAFUP	DISABLED, ENABLED	DISABLED	N	ACCU54 round half up mode enable
ROUNDRTZI	ROUND_TO_ZERO, ROUND_TO_INFINITE	ROUND_TO_ZERO	N	ACCU54 round to zero or infinite
ROUNDBIT	ROUND_TO_BIT0, ROUND_TO_BIT1, ROUND_TO_BIT2, ROUND_TO_BIT3, ROUND_TO_BIT4, ROUND_TO_BIT5, ROUND_TO_BIT6, ROUND_TO_BIT7, ROUND_TO_BIT8, ROUND_TO_BIT9, ROUND_TO_BIT10, ROUND_TO_BIT11, ROUND_TO_BIT12, ROUND_TO_BIT13, ROUND_TO_BIT14, ROUND_TO_BIT15	ROUND_TO_BIT0	N	0000: round to bit0 ... 1111: round too bit15
CASCOUTREGBYPS	REGISTER, BYPASS	REGISTER	N	DSP cascaded enable
SFTEN	DISABLED, ENABLED	DISABLED	N	Shift enable
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous
REGINPUTA	REGISTER, BYPASS	REGISTER	Y	Register operand A
REGINPUTB	REGISTER, BYPASS	REGISTER	Y	Register operand B
REGINPUTC	REGISTER, BYPASS	REGISTER	Y	Register operand C
REGADDSUB	REGISTER, BYPASS	REGISTER	Y	Register ADDSUB
REGLOADC	REGISTER, BYPASS	REGISTER	Y	Register LOADC
REGLOADC2	REGISTER, BYPASS	REGISTER	Y	Register LOADC, Second
REGCIN	REGISTER, BYPASS	REGISTER	Y	Register CIN
REGPIPELINE	REGISTER, BYPASS	REGISTER	Y	Register between multiplier
REGOUTPUT	REGISTER, BYPASS	REGISTER	Y	Register output
GSR	ENABLED, DISABLED	ENABLED	Y	Enable global set/reset
RESETMODE	SYNC, ASYNC	SYNC	Y	Synchronous/asynchronous reset control. Reset release is always asynchronous.

## Appendix A. Instantiating DSP Primitives in HDL

This appendix illustrates how to instantiate the Nexus sysDSP primitives for both Verilog and VHDL.

### Verilog Example Showing Snippet of the MULT18X18 Instantiation

```
defparam dsp_mult_0.REGINPUTA = "REGISTER" ;
defparam dsp_mult_0.REGINPUTB = "REGISTER" ;
defparam dsp_mult_0.REGOUTPUT = "REGISTER" ;
defparam dsp_mult_0.GSR = "ENABLED" ;
defparam dsp_mult_0.RESETMODE = "SYNC" ;
MULT18X18 dsp_mult_0 (
    .A17(t5M1A_17), .A16(t5M1A_16), .A15(t5M1A_15),
    .A14(t5M1A_14), .A13(t5M1A_13), .A12(t5M1A_12), .A11(t5M1A_11),
    .A10(t5M1A_10), .A9(t5M1A_9), .A8(t5M1A_8), .A7(t5M1A_7), .A6(t5M1A_6),
    .A5(t5M1A_5), .A4(t5M1A_4), .A3(t5M1A_3), .A2(t5M1A_2), .A1(t5M1A_1),
    .A0(t5M1A_0), .B17(t5M1B_17), .B16(t5M1B_16), .B15(t5M1B_15),
    .B14(t5M1B_14), .B13(t5M1B_13), .B12(t5M1B_12), .B11(t5M1B_11),
    .B10(t5M1B_10), .B9(t5M1B_9), .B8(t5M1B_8), .B7(t5M1B_7),
    .B6(t5M1B_6), .B5(t5M1B_5), .B4(t5M1B_4), .B3(t5M1B_3),
    .B2(t5M1B_2), .B1(t5M1B_1), .B0(t5M1B_0), .CLK(Clock),
    .CEA(ClockEn), .RSTA(Reset), .CEB(ClockEn), .RSTB(Reset),
    .SIGNEDA(scuba_vhi), .SIGNEDB(scuba_vhi), .RSTOUT(Reset), CEOUT(ClockEn),
    .Z35(t5P1_35), .Z34(t5P1_34), .Z33(t5P1_33), .Z32(t5P1_32), .Z31(t5P1_31),
    .Z30(t5P1_30), .Z29(t5P1_29), .Z28(t5P1_28), .Z27(t5P1_27), .Z26(t5P1_26),
    .Z25(t5P1_25), .Z24(t5P1_24), .Z23(t5P1_23), .Z22(t5P1_22), .Z21(t5P1_21),
    .Z20(t5P1_20), .Z19(t5P1_19), .Z18(t5P1_18), .Z17(t5P1_17), .Z16(t5P1_16),
    .Z15(t5P1_15), .Z14(t5P1_14), .Z13(t5P1_13), .Z12(t5P1_12), .Z11(t5P1_11),
    .Z10(t5P1_10), .Z9(t5P1_9), .Z8(t5P1_8), .Z7(t5P1_7), .Z6(t5P1_6),
    .Z5(t5P1_5), .Z4(t5P1_4), .Z3(t5P1_3), .Z2(t5P1_2), .Z1(t5P1_1),
    .Z0(t5P1_0)
);
```

### VHDL Example Showing Snippet of the MULT18X18 Instantiation

```
dsp_mult_0: MULT18X18
    generic map (
        REGINPUTA=> "REGISTER",
        REGINPUTB=> "REGISTER",
        REGOUTPUT=> "REGISTER",
        GSR=> "ENABLED",
        RESETMODE=> "SYNC"
    )

    port map (
        A17=>t5M1A_17, A16=>t5M1A_16, A15=>t5M1A_15,
        A14=>t5M1A_14, A13=>t5M1A_13, A12=>t5M1A_12,
        A11=>t5M1A_11, A10=>t5M1A_10, A9=>t5M1A_9, A8=>t5M1A_8,
        A7=>t5M1A_7, A6=>t5M1A_6, A5=>t5M1A_5, A4=>t5M1A_4,
        A3=>t5M1A_3, A2=>t5M1A_2, A1=>t5M1A_1, A0=>t5M1A_0,
        B17=>t5M1B_17, B16=>t5M1B_16, B15=>t5M1B_15,
        B14=>t5M1B_14, B13=>t5M1B_13, B12=>t5M1B_12,
        B11=>t5M1B_11, B10=>t5M1B_10, B9=>t5M1B_9, B8=>t5M1B_8,
        B7=>t5M1B_7, B6=>t5M1B_6, B5=>t5M1B_5, B4=>t5M1B_4,
        B3=>t5M1B_3, B2=>t5M1B_2, B1=>t5M1B_1, B0=>t5M1B_0,
        CLK=>Clock, CEA=>clockEn, RSTA=>Reset, CEB=>clockEn,
        RSTB=>Reset, SIGNEDIA=>scuba_vhi, SIGNEDIB=>scuba_vhi,
        Z35=>t5P1_35, Z34=>t5P1_34, Z33=>t5P1_33, Z32=>t5P1_32,
```

```
Z31=>t5P1_31, Z30=>t5P1_30, Z29=>t5P1_29, Z28=>t5P1_28,  
Z27=>t5P1_27, Z26=>t5P1_26, Z25=>t5P1_25, Z24=>t5P1_24,  
Z23=>t5P1_23, Z22=>t5P1_22, Z21=>t5P1_21, Z20=>t5P1_20,  
Z19=>t5P1_19, Z18=>t5P1_18, Z17=>t5P1_17, Z16=>t5P1_16,  
Z15=>t5P1_15, Z14=>t5P1_14, Z13=>t5P1_13, Z12=>t5P1_12,  
Z11=>t5P1_11, Z10=>t5P1_10, Z9=>t5P1_9, Z8=>t5P1_8,  
Z7=>t5P1_7, Z6=>t5P1_6, Z5=>t5P1_5, Z4=>t5P1_4, Z3=>t5P1_3,  
Z2=>t5P1_2, Z1=>t5P1_1, Z0=>t5P1_0  
);
```

## Appendix B. HDL Inference for DSP

Synthesis inference flow enables the design tools to infer sysDSP blocks from an HDL design. It is important to note that when using the inference flow, unless the code style matches the sysDSP block, results are not optimal. You can infer the Nexus sysDSP block with Synplify Pro® from Synopsys or the Lattice Synthesis Engine (LSE) if certain coding guidelines are followed. The following are VHDL and Verilog examples. This example would not have functional simulation support. This is for example purposes only.

### VHDL Example to Infer Fully Pipelined Multiplier

```
library ieee;
use ieee.std_logic_1164.all;
--use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity mult is
    port (reset, clk : in std_logic;
          dataax, dataay : in std_logic_vector(8 downto 0);
          dataout : out std_logic_vector (17 downto 0));
end;

architecture arch of mult is
    signal dataax_reg, dataay_reg : std_logic_vector (8 downto 0);
    signal dataout_node : std_logic_vector (17 downto 0);
    signal dataout_pipeline : std_logic_vector (17 downto 0);
begin
    process (clk, reset)
    begin
        if (reset='1') then
            dataax_reg <= (others => '0');
            dataay_reg <= (others => '0');
        elsif (clk'event and clk='1') then
            dataax_reg <= dataax;
            dataay_reg <= dataay;
        end if;
    end process;

    dataout_node <= dataax_reg * dataay_reg;
    process (clk, reset)
    begin
        if (reset='1') then
            dataout_pipeline <= (others => '0');
        elsif (clk'event and clk='1') then
            dataout_pipeline <= dataout_node;
        end if;
    end process;

    process (clk, reset)
    begin
        if (reset='1') then
            dataout <= (others => '0');
        elsif (clk'event and clk='1') then
            dataout <= dataout_pipeline;
        end if;
    end process;
end arch;
```

## Verilog Example to Infer Fully Pipelined Multiplier

```
module mult (dataout, dataax, dataay, clk, reset);
output [35:0] dataout;
input [17:0] dataax, dataay;
input clk, reset;

reg [35:0] dataout;
reg [17:0] dataax_reg, dataay_reg;
wire [35:0] dataout_node;
reg [35:0] dataout_reg;

always @(posedge clk or posedge reset)
begin
    if (reset)
    begin
        dataax_reg <= 0;
        dataay_reg <= 0;
    end
    else
    begin
        dataax_reg <= dataax;
        dataay_reg <= dataay;
    end
end

assign dataout_node = dataax_reg * dataay_reg;
always @(posedge clk or posedge reset)
begin
    if (reset)
        dataout_reg <= 0;
    else
        dataout_reg <= dataout_node;
    end

always @(posedge clk or posedge reset)
begin
    if (reset)
        dataout <= 0;
    else
        dataout <= dataout_reg;
    end
end
endmodule
```

## Technical Support Assistance

Submit a technical support case through [www.latticesemi.com/techsupport](http://www.latticesemi.com/techsupport).

## Revision History

### Revision 1.1, June 2020

Section	Change Summary
All	<ul style="list-style-type: none"><li>• Changed document name to sysDSP Usage Guide for Nexus Platform.</li><li>• Changed CrossLink-NX to Nexus across the document.</li></ul>

### Revision 1.0, November 2019

Section	Change Summary
All	Initial release



[www.latticesemi.com](http://www.latticesemi.com)