7 Series DSP48E1 Slice

User Guide

UG479 (v1.10) March 27, 2018

XILINX®

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
03/01/2011	1.0	Initial Xilinx release.
03/28/2011	1.1	Added Chapter 1, Overview. Updated DSP48E1 Slice Features. In Table 2-1, removed XC7K30T and added XC7K355T, XC7K420T, and XC7K480T.
10/18/2011	1.2	Added Stacked Silicon Interconnect. Updated DSP48E1 column range and upper limit of DSP slices in DSP48E1 Tile and Interconnect. Added Artix-7 and Virtex-7 families to Table 2-1, including table notes. Added Memory-Mapped I/O Register Application.
01/30/2012	1.3	Removed top performance value of 600 MHz throughout. In Table 2-1, removed XC7A8, XC7A15, XC7A30T, and XC7A50T, and updated number of DSP48E1 slices per column for XC7K420T and XC7VX550T. Updated lower limit for number of DSP48E1 columns and DSP slices in DSP48E1 Tile and Interconnect.
10/02/2012	1.4	Updated Device Resources. Updated last sentence in first paragraph of DSP48E1 Tile and Interconnect. In Table 2-1, removed XC7A350T and XC7V1500T, and added XC7VH580T and XC7VH870T.
04/03/2013	1.5	Updated link for UG687 in Additional Documentation Resources. Updated Table 2-1 and Table 2-2. Updated first and last paragraphs in Features Relative to Prior Generations.

Date	Version	Revision
08/07/2013	1.6	Updated Additional Documentation Resources and Table 2-1.
05/10/2014	1.7	Added link to UG958 in Preface. Minor improvements to descriptions in Table 2-2 and Table 2-3. Minor changes to Figure 2-7 and Figure 2-8.
11/10/2014	1.8	Added 7A15T device to Table 2-1. Updated last paragraph in MULTSIGNOUT and CARRYCASCOUT.
09/27/2016	1.9	Added Spartan-7 family to Preface. Added 7 series FPGA data sheets to Additional Documentation Resources. In DSP48E1 Tile and Interconnect, changed smallest number of slices from 60 to 10 and added Spartan-7 devices. Added Spartan-7, 7A12T, and 7A25T devices to Table 2-1.
03/27/2018	1.10	Added UG901 to Additional Documentation Resources. Updated first paragraph in Design Recommendations.

Table of Contents

Appendix A: CARRYOUT, CARRYCASCOUT, and MULTSIGNOUT

About This Guide

Xilinx® 7 series FPGAs include four FPGA families that are all designed for lowest power to enable a common design to scale across families for optimal power, performance, and cost. The Spartan®-7 family is the lowest density with the lowest cost entry point into the 7 series portfolio. The Artix®-7 family is optimized for highest performance-per-watt and bandwidth-per-watt for cost-sensitive, high volume applications. The Kintex®-7 family is an innovative class of FPGAs optimized for the best price-performance. The Virtex®-7 family is optimized for highest system performance and capacity.

This guide serves as a technical reference describing the 7 series FPGAs DSP48E1 slice. This 7 series FPGAs DSP48E1 slice user guide is part of an overall set of documentation on the 7 series FPGAs, which is available on the Xilinx website at www.xilinx.com/documentation.

Guide Contents

This manual contains the following chapters:

- Chapter 1, Overview
- Chapter 2, DSP48E1 Description and Specifics
- Chapter 3, DSP48E1 Design Considerations
- Appendix A, CARRYOUT, CARRYCASCOUT, and MULTSIGNOUT

Additional Documentation Resources

The following documents provide useful supplemental material to this guide:

1. UG193, *Virtex-5 FPGA XtremeDSP Design Considerations User Guide*
2. DS180, *7 Series FPGAs Overview*
3. UG687, *XST User Guide for Virtex-6, Spartan-6, and 7 Series Devices*
4. UG901, *Vivado Design Suite User Guide: Synthesis*
5. UG958, *Vivado Design Suite Reference Guide: Model-Based DSP Design Using System Generator*
6. 7 Series FPGA Data Sheets:

   DS181, *Artix-7 FPGAs Data Sheet: DC and Switching Characteristics*

   DS182, *Kintex-7 FPGAs Data Sheet: DC and AC Switching Characteristics*

   DS183, *Virtex-7 T and XT FPGAs Data Sheet: DC and AC Switching Characteristics*
7. UG073, *XtremeDSP for Virtex-4 FPGAs User Guide*

Additional Support Resources

To search the database of silicon and software questions and answers or to create a technical support case in WebCase, see the Xilinx website at:

www.xilinx.com/support

XILINX

Chapter 1

Overview

DSP48E1 Slice Overview

FPGAs are efficient for digital signal processing (DSP) applications because they can implement custom, fully parallel algorithms. DSP applications use many binary multipliers and accumulators that are best implemented in dedicated DSP slices. All 7 series FPGAs have many dedicated, full-custom, low-power DSP slices, combining high speed with small size while retaining system design flexibility. The DSP slices enhance the speed and efficiency of many applications beyond digital signal processing, such as wide dynamic bus shifters, memory address generators, wide bus multiplexers, and memory-mapped I/O registers. The basic functionality of the DSP48E1 slice is shown in Figure 1-1. For complete details, refer to Figure 2-1 and Chapter 2, DSP48E1 Description and Specifics.

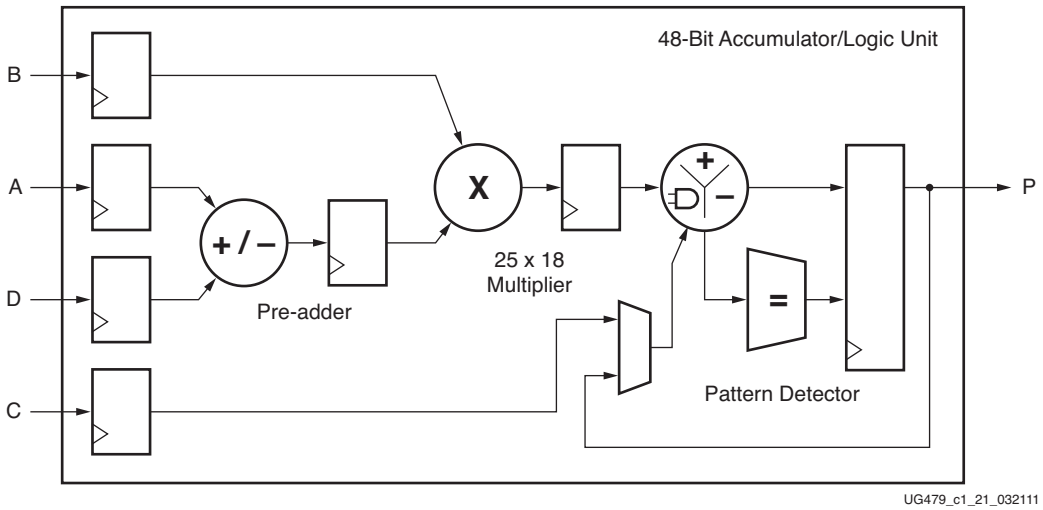


Figure 1-1: **Basic DSP48E1 Slice Functionality**

Some highlights of the DSP functionality include:

- 25 × 18 two's-complement multiplier:
  - Dynamic bypass
- 48-bit accumulator:
  - Can be used as a synchronous up/down counter
- Power saving pre-adder:
  - Optimizes symmetrical filter applications and reduces DSP slice requirements

- Single-instruction-multiple-data (SIMD) arithmetic unit:
  - Dual 24-bit or quad 12-bit add/subtract/accumulate
- Optional logic unit:
  - Can generate any one of ten different logic functions of the two operands
- Pattern detector:
  - Convergent or symmetric rounding
  - 96-bit-wide logic functions when used in conjunction with the logic unit
- Advanced features:
  - Optional pipelining and dedicated buses for cascading

Features Relative to Prior Generations

The 7 series FPGA DSP48E1 slice is functionally equivalent and fully compatible with the Virtex®-6 FPGA DSP48E1 slice, and a superset of the Virtex-5 FPGA DSP48E slice [Ref 1]. The 7 series FPGA DSP48E1 slice offers more capability than the DSP48A1 slice of the Spartan®-6 FPGA family with these differences:

- Wider functionality in DSP48E1 than DSP48A1:
  - Multiplier width is improved from 18 x 18 in the Spartan-6 family to 25 x 18 in the 7 series
  - The A register width is improved from 18 bits in the Spartan-6 family to 30 bits in the 7 series:
    - A and B registers can be concatenated in the 7 series
    - The A register feeds the pre-adder in the 7 series instead of the B register
  - Cascading capability on both pipeline paths for larger multipliers and larger post-adders
- Unique features in DSP48E1 over DSP48A1:
  - Arithmetic logic unit (ALU)
  - SIMD mode
  - Pattern detector
  - 17-bit shifter

Virtex-6 family DSP designs migrate directly to the DSP resources of the 7 series. Migration of designs with cascaded DSP slices should consider the number of DSP slices per column. Spartan-6 family DSP designs can be converted to the 7 series, but designers should examine how to take advantage of the greater capability of the DSP48E1 slice. See UG429, *7 Series FPGAs Migration User Guide* for more information.

Device Resources

The DSP resources are optimized and scalable across all the 7 series families, providing a common architecture that improves implementation efficiency, IP implementation, and design migration. The number of DSP48E1 slices and the ratio between DSP and other device resources differentiates the 7 series families. Migration between the 7 series families does not require any design changes for the DSP48E1.

See Table 2-1 for the available DSP48E1 resources for the 7 series FPGAs. Refer to *7 Series FPGAs Overview* [Ref 2] for the most up-to-date information on all the 7 series FPGAs.

Design Recommendations

Many DSP designs are well suited for the 7 series architecture. To obtain best use of the architecture, the underlying features and capabilities need to be understood so that design entry code can take advantage of these resources. DSP48E1 resources are used automatically for most DSP functions and many arithmetic functions. In most cases, DSP resources should be inferred. See your preferred synthesis tool documentation for guidelines to ensure proper inference of the DSP48E1 slice [Ref 3][Ref 4]. The synthesis tools can infer the resources. Instantiation can be used to directly access specific DSP48E1 slice features. Recommendations for using DSP48E1 slices include:

- Use signed values in HDL source
- Pipeline for performance and lower power, both in the DSP48E1 slice and fabric
- Use the configurable logic block (CLB) carry logic to implement small multipliers, adders, and counters
- Use CLB SRLs, CLB distributed RAM, and/or block RAM to store filter coefficients
- Set USE_MULT to NONE when using only the adder/logic unit to save power
- Cascade using the dedicated resources rather than fabric, keeping usage to one column for highest performance and lowest power
- Consider using time multiplexing for the design

For more information on design techniques, see Chapter 3, DSP48E1 Design Considerations.

Stacked Silicon Interconnect

The DSP slices cannot be cascaded across the interposer (super logic region (SLR) boundary). For more information on stacked silicon interconnect (SSI) technology, see WP380, *Xilinx Stacked Silicon Interconnect Technology Delivers Breakthrough FPGA Capacity, Bandwidth, and Power Efficiency*.

XILINX®

Chapter 2

DSP48E1 Description and Specifics

This chapter provides technical details of the DSP element available in 7 series FPGAs, the DSP48E1 slice.

The DSP48E1 slice provides improved flexibility and utilization, improved efficiency of applications, reduced overall power consumption, and increased maximum frequency. The high performance allows designers to implement multiple slower operations in a single DSP48E1 slice using time-multiplexing methods.

The DSP48E1 slice supports many independent functions. These functions include multiply, multiply accumulate (MACC), multiply add, three-input add, barrel shift, wide-bus multiplexing, magnitude comparator, bitwise logic functions, pattern detect, and wide counter. The architecture also supports cascading multiple DSP48E1 slices to form wide math functions, DSP filters, and complex arithmetic without the use of general FPGA logic.

This chapter contains the following sections:

- DSP48E1 Slice Features
- Architectural Highlights of the 7 Series FPGA DSP48E1 Slice
- Simplified DSP48E1 Slice Operation

The 7 series FPGA DSP48E1 slice is shown in Figure 2-1. The slice is functionally equivalent to the Virtex®-6 FPGA DSP48E1 slice and is an extension of the DSP48E slice in Virtex-5 devices, described in the *Virtex-5 FPGA XtremeDSP Design Considerations User Guide* [Ref 1].

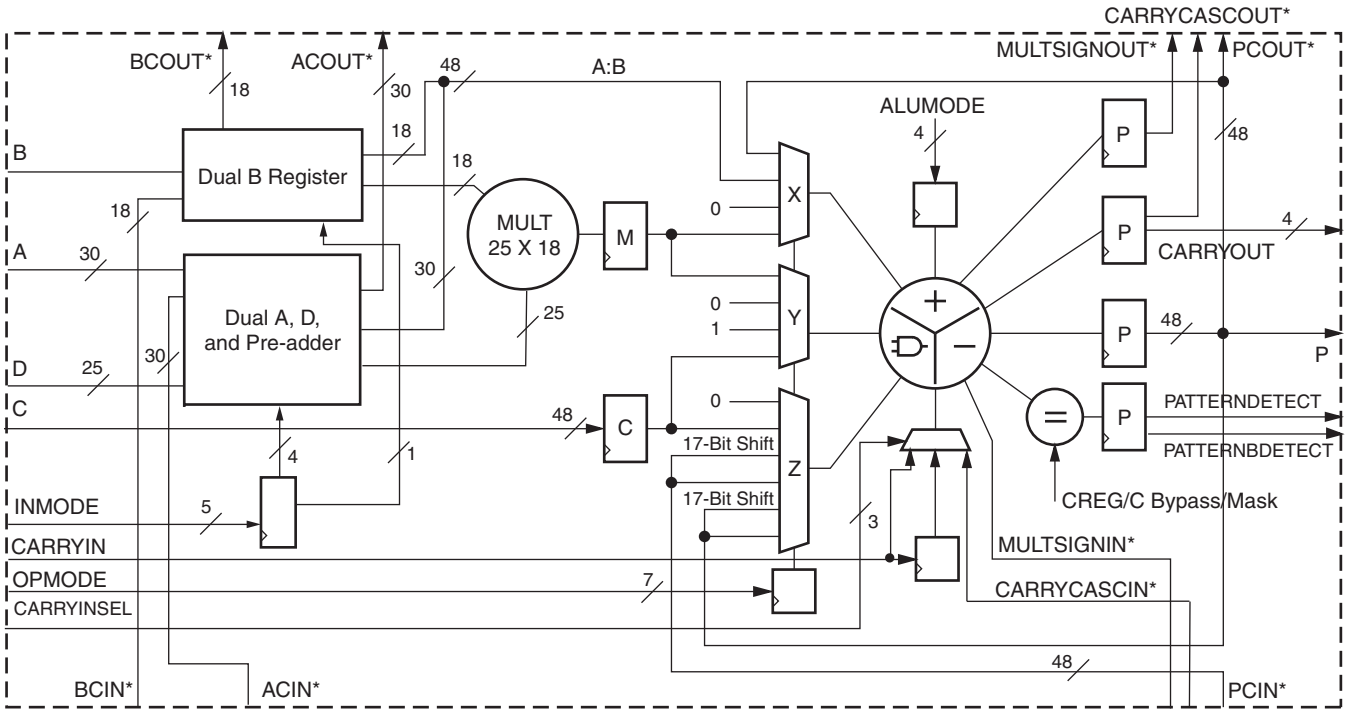


Figure 2-1: **7 Series FPGA DSP48E1 Slice**

DSP48E1 Slice Features

This section describes the 7 series FPGA DSP48E1 slice features.

The DSP slice consists of a multiplier followed by an accumulator. At least three pipeline registers are required for both multiply and multiply-accumulate operations to run at full speed. The multiply operation in the first stage generates two partial products that need to be added together in the second stage.

When only one or two registers exist in the multiplier design, the M register should always be used to save power and improve performance.

Add/Sub and Logic Unit operations require at least two pipeline registers (input, output) to run at full speed.

The cascade capabilities of the DSP slice are extremely efficient at implementing high-speed pipelined filters built on the adder cascades instead of adder trees.

Multiplexers are controlled with dynamic control signals, such as OPMODE, ALUMODE, and CARRYINSEL, enabling a great deal of flexibility. Designs using registers and dynamic opmodes are better equipped to take advantage of the DSP slice capabilities than combinatorial multiplies.

In general, the DSP slice supports both sequential and cascaded operations due to the dynamic OPMODE and cascade capabilities. Fast Fourier Transforms (FFTs), floating point, computation (multiply, add/sub, divide), counters, and large bus multiplexers are some applications of the DSP slice.

Additional capabilities of the DSP slice include synchronous resets and clock enables, dual A input pipeline registers, pattern detection, Logic Unit functionality, single

instruction/multiple data (SIMD) functionality, and MACC and Add-Acc extension to 96 bits. The DSP slice supports convergent and symmetric rounding, terminal count detection and auto-resetting for counters, and overflow/underflow detection for sequential accumulators.

ALU functions are identical in 7 series FPGA DSP48E1 slice as in the Virtex-6 FPGA DSP48E1 slice. See ALUMODE Inputs, page 35 for more information.

Architectural Highlights of the 7 Series FPGA DSP48E1 Slice

The 7 series FPGA DSP48E1 slice is functionally equivalent to the Virtex-6 FPGA DSP48E1. The 7 series FPGA DSP48E1 slice contains a pre-adder after the A register with a 25-bit input vector called D. The D register can be used either as the pre-adder register or an alternate input to the multiplier. The DSP48E1 specific features are highlighted in Figure 2-2. More detailed descriptions are found starting at Input Ports, page 29.

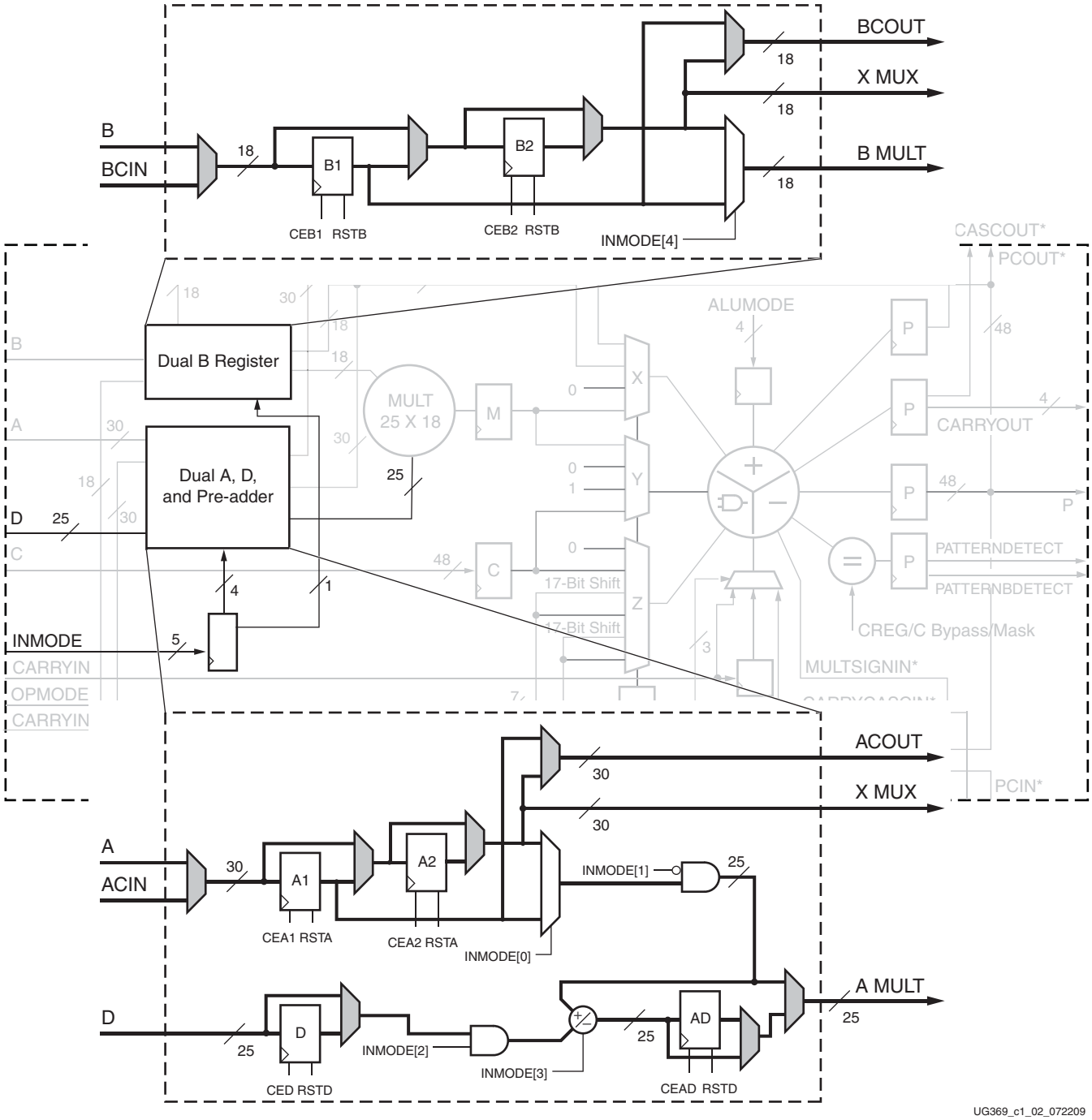


Figure 2-2: **Hierarchical View of the 7 Series DSP48E1 Slice Input Registers and Pre-adder**

The features in the 7 series FPGA DSP48E1 slice are:

- 25-bit pre-adder with D register to enhance the capabilities of the A path
- INMODE control supports balanced pipelining when dynamically switching between multiply (A*B) and add operations (A:B)
- 25 x 18 multiplier
- 30-bit A input of which the lower 25 bits feed the A input of the multiplier, and the entire 30-bit input forms the upper 30 bits of the 48-bit A:B concatenate internal bus.
- Cascading A and B input
  - Semi-independently selectable pipelining between direct and cascade paths
  - Separate clock enables two-deep A and B set of input registers
- Independent C input and C register with independent reset and clock enable.
- CARRYCASCIN and CARRYCASCOUT internal cascade signals to support 96-bit accumulators/adders/subtracters in two DSP48E1 slices
- MULTSIGNIN and MULTSIGNOUT internal cascade signals with special OPMODE setting to support a 96-bit MACC extension
- Single Instruction Multiple Data (SIMD) Mode for three-input adder/subtracter which precludes use of multiplier in first stage
  - Dual 24-bit SIMD adder/subtracter/accumulator with two separate CARRYOUT signals
  - Quad 12-bit SIMD adder/subtracter/accumulator with four separate CARRYOUT signals
- 48-bit logic unit
  - Bitwise logic operations – two-input AND, OR, NOT, NAND, NOR, XOR, and XNOR
  - Logic unit mode dynamically selectable via ALUMODE
- Pattern detector
  - Overflow/underflow support
  - Convergent rounding support
  - Terminal count detection support and auto resetting
- Cascading 48-bit P bus supports internal low-power adder cascade
  - The 48-bit P bus allows for 12-bit/QUAD or 24-bit/DUAL SIMD adder cascade support
- Optional 17-bit right shift to enable wider multiplier implementation
- Dynamic user-controlled operating modes
  - 7-bit OPMODE control bus provides X, Y, and Z multiplexer select signals
- Carry in for the second stage adder
  - Support for rounding
  - Support for wider add/subtracts
  - 3-bit CARRYINSEL multiplexer
- Carry out for the second stage adder
  - Support for wider add/subtracts
  - Available for each SIMD adder (up to four)

    - Cascaded CARRYCASCOUT and MULTSIGNOUT allows for MACC extensions up to 96 bits
- Optional input, pipeline, and output/accumulate registers
- Optional control registers for control signals (OPMODE, ALUMODE, and CARRYINSEL)
- Independent clock enable and resets for greater flexibility, with reset having priority.
- To save power when the first stage multiplier is not being used, the USE_MULT attribute allows the customer to gate off internal multiplier logic.

Each DSP48E1 slice has a two-input multiplier followed by multiplexers and a three-input adder/subtracter/accumulator. The DSP48E1 multiplier has asymmetric inputs and accepts an 18-bit two's complement operand and a 25-bit two's complement operand. The multiplier stage produces a 43-bit two's complement result in the form of two partial products. These partial products are sign-extended to 48 bits in the X multiplexer and Y multiplexer and fed into three-input adder for final summation. This results in a 43-bit multiplication output, which has been sign-extended to 48 bits. Therefore, when the multiplier is used, the adder effectively becomes a two-input adder.

The second stage adder/subtracter accepts three 48-bit, two's complement operands and produces a 48-bit, two's complement result when the multiplier is bypassed by setting USE_MULT attribute to NONE and with the appropriate OPMODE setting. In SIMD mode, the 48-bit adder/subtracter also supports dual 24-bit or quad 12-bit SIMD arithmetic operations with CARRYOUT bits. In this configuration, bitwise logic operations on two 48-bit binary numbers are also supported with dynamic ALUMODE control signals.

Higher level DSP functions are supported by cascading individual DSP48E1 slices in a DSP48E1 column. Two datapaths (ACOUT and BCOUT) and the DSP48E1 slice outputs (PCOUT, MULTSIGNOUT, and CARRYCASCOUT) provide the cascade capability. The ability to cascade datapaths is useful in filter designs. For example, a Finite Impulse Response (FIR) filter design can use the cascading inputs to arrange a series of input data samples and the cascading outputs to arrange a series of partial output results. The ability to cascade provides a high-performance and low-power implementation of DSP filter functions because the general routing in the fabric is not used.

The C input port allows the formation of many 3-input mathematical functions, such as 3-input addition or 2-input multiplication with an addition. One subset of this function is the valuable support of symmetrically rounding a multiplication toward zero or toward infinity. The C input together with the pattern detector also supports convergent rounding.

For multi-precision arithmetic, the DSP48E1 slice provides a right wire shift by 17. Thus, a partial product from one DSP48E1 slice can be right justified and added to the next partial product computed in an adjacent DSP48E1 slice. Using this technique, the DSP48E1 slices can be used to build bigger multipliers.

Programmable pipelining of input operands, intermediate products, and accumulator outputs enhances throughput. The 48-bit internal bus (PCOUT/PCIN) allows for aggregation of DSP slices in a single column. Fabric logic is needed when spanning multiple columns.

The pattern detector at the output of the DSP48E1 slice provides support for convergent rounding, overflow/underflow, block floating point, and support for accumulator terminal count (counter auto reset). The pattern detector can detect if the output of the DSP48E1 slice matches a pattern, as qualified by a mask.

DSP48E1 Tile and Interconnect

Two DSP48E1 slices and dedicated interconnect form a DSP48E1 tile (see Figure 2-3). The DSP48E1 tiles stack vertically in a DSP48E1 column. The height of a DSP48E1 tile is the same as five configurable logic blocks (CLBs) and also matches the height of one block RAM. The block RAM in 7 series devices can be split into two 18K block RAMs. Each DSP48E1 slice aligns horizontally with an 18K block RAM. The 7 series devices have up to 20 DSP48E1 columns.

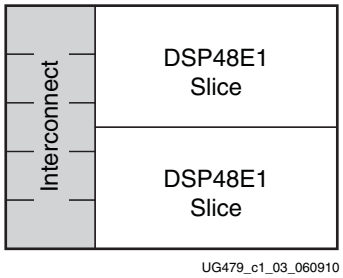


Figure 2-3: **DSP48E1 Interconnect and Relative Dedicated Element Sizes**

The 7 series devices offer from 10 to 3,600 DSP slices per device, yielding an impressive processing power in the range of tens of GMAC/s up to thousands of GMAC/s (peak), enabling very intensive DSP applications. Table 2-1 shows the number of DSP48E1 slices for each device in the 7 series. Refer to the product tables on xilinx.com for information on the Spartan-7, Artix-7, Kintex-7, and Virtex-7 families.

Table 2-1: **Number of DSP48E1 Slices in 7 Series Devices**

Device	Total DSP48E1 Slices per Device	Number of DSP48E1 Columns per Device	Number of DSP48E1 Slices per Column
7S6	10	1	20(2)
7S15	20	1	20
7S25	80	2	40
7S50	120	2	60
7S75	140	2	80(2)
7S100	160	2	80
7A12T	40	2	40(2)
7A15T	45	2	60(2)
7A25T	80	2	40
7A35T	90	2	60(2)
7A50T	120	2	60
7A75T	180	3	80(2)
7A100T	240	3	80
7A200T	740	9	100(1)
7K70T	240	3	80

Table 2-1: **Number of DSP48E1 Slices in 7 Series Devices *(Cont'd)***

Device	Total DSP48E1 Slices per Device	Number of DSP48E1 Columns per Device	Number of DSP48E1 Slices per Column
7K160T	600	6	100
7K325T	840	6	140
7K355T	1,440	12	120
7K410T	1,540	11	140
7K420T	1,680	12	160(2)
7K480T	1,920	12	160
7V585T	1,260	7	180
7V2000T	2,160	9	240(3)
7VX330T	1,120	8	140
7VX415T	2,160	18	120
7VX485T	2,800	20	140
7VX550T	2,880	18	200(2)
7VX690T	3,600	18	200
7VX980T	3,600	20	180
7VX1140T	3,360	14	240(3)
7VH580T	1,680	14	120(3)
7VH870T	2,520	14	180(3)

Notes:

1. The four center DSP columns contain GTP Quads that displace 20 DSP slices in those columns per GTP Quad.
2. The total DSP slice count is limited by tools.
3. There are 60 DSP slices per column in each SLR.

DSP48E1 Slice Primitive

Figure 2-4 shows the DSP48E1 primitive. It also shows the input and output ports of the DSP48E1 slice along with the bit width of each port. The port definitions are in Table 2-2.

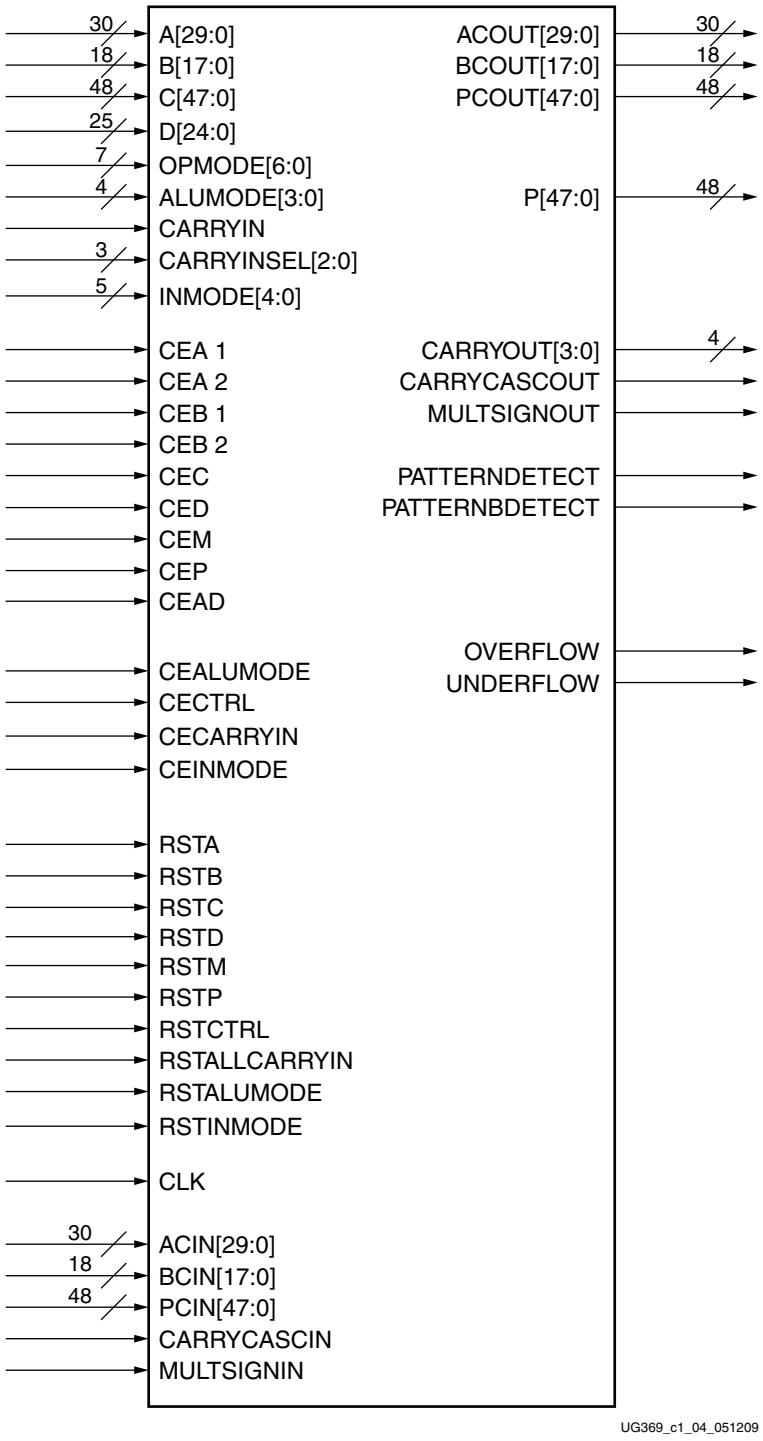


Figure 2-4: **DSP48E1 Slice Primitive**

Table 2-2: **DSP48E1 Port Descriptions**

Name	Direction	Bit Width	Description
A[1]	In	30	A[24:0] is the A input of the multiplier or the pre-adder. A[29:0] are the most significant bits (MSBs) of the A:B concatenated input to the second-stage adder/subtracter or logic function.
ACIN[2]	In	30	Cascaded data input from ACOUT of previous DSP48E1 slice (muxed with A).
ACOUT[2]	Out	30	Cascaded data output to ACIN of next DSP48E1 slice.
ALUMODE	In	4	Controls the selection of the logic function in the DSP48E1 slice (see Table 2-13, page 42).
B[1]	In	18	The B input of the multiplier. B[17:0] are the least significant bits (LSBs) of the A:B concatenated input to the second-stage adder/subtracter or logic function.
BCIN[2]	In	18	Cascaded data input from BCOUT of previous DSP48E1 slice (muxed with B).
BCOUT[2]	Out	18	Cascaded data output to BCIN of next DSP48E1 slice.
C[1]	In	48	Data input to the second-stage adder/subtracter, pattern detector, or logic function.
CARRYCASCIN[2]	In	1	Cascaded carry input from CARRYCASCOUT of previous DSP48E1 slice.
CARRYCASCOUT[2]	Out	1	Cascaded carry output to CARRYCASCIN of next DSP48E1 slice. This signal is internally fed back into the CARRYINSEL multiplexer input of the same DSP48E1 slice.
CARRYIN	In	1	Carry input from the FPGA logic.
CARRYINSEL	In	3	Selects the carry source (see Table 2-11).
CARRYOUT	Out	4	4-bit carry output from each 12-bit field of the accumulate/adder/logic unit. Normal 48-bit operation uses only CARRYOUT3. SIMD operation can use four carry out bits (CARRYOUT[3:0]).
CEA1	In	1	Clock enable for the first A (input) register. A1 is only used if AREG = 2 or INMODE[0]= 1.
CEA2	In	1	Clock enable for the second A (input) register. A2 is only used if AREG = 1 or 2 and INMODE[0]=0.
CEAD	In	1	Clock enable for the pre-adder output AD pipeline register.
CEALUMODE	In	1	Clock enable for ALUMODE (control inputs) registers.
CEB1	In	1	Clock enable for the first B (input) register. B1 is only used if BREG = 2 or INMODE[4] = 1.
CEB2	In	1	Clock enable for the second B (input) register. B2 is only used if BREG = 1 or 2 and INMODE[4]=0.
CEC	In	1	Clock enable for the C (input) register.
CECARRYIN	In	1	Clock enable for the CARRYIN (input from FPGA logic) register.

CECTRL	In	1	Clock enable for the OPMODE and CARRYINSEL (control inputs) registers.
CED	In	1	Clock enable for the D (input) register.
CEINMODE	In	1	Clock enable for the INMODE control input registers.
CEM	In	1	Clock enable for the post-multiply M (pipeline) register and the internal multiply round CARRYIN register.
CEP	In	1	Clock enable for the P (output) register.
CLK	In	1	CLK is the DSP48E1 input clock, common to all internal registers and flip-flops.
D[(1)]			

Table 2-2: **DSP48E1 Port Descriptions *(Cont'd)***

Name	Direction	Bit Width	Description
RSTINMODE	In	1	Reset for the INMODE (control input) registers.
RSTM	In	1	Reset for the M (pipeline) register.
RSTP	In	1	Reset for the P (output) register.
UNDERFLOW	Out	1	Underflow indicator when used with the appropriate setting of the pattern detector.

Notes:

1. When these data ports are not in use and to reduce leakage power dissipation, the data port input signals must be tied High, the port input register must be selected, and the CE and RST input control signals must be tied Low. An example of unused C port recommended settings would be setting C[47:0] = all ones, CREG = 1, CEC = 0, and RSTC = 0.
2. These signals are dedicated routing paths internal to the DSP48E1 column. They are not accessible via fabric routing resources.
3. All signals are active High.

Simplified DSP48E1 Slice Operation

The math portion of the DSP48E1 slice consists of a 25-bit pre-adder, a 25-bit by 18-bit two's complement multiplier followed by three 48-bit datapath multiplexers (with outputs X, Y, and Z). This is followed by a three-input adder/subtracter or two-input logic unit (see Figure 2-5). When using two-input logic unit, the multiplier cannot be used.

The data and control inputs to the DSP48E1 slice feed the arithmetic and logic stages. The A and B data inputs can optionally be registered one or two times to assist the construction of different, highly pipelined, DSP application solutions. The D path and the AD path can each be registered once. The other data inputs and the control inputs can be optionally registered once. Maximum frequency operation as specified in the data sheet is achieved by using pipeline registers. More detailed timing information is available in Chapter 3, DSP48E1 Design Considerations.

In its most basic form, the output of the adder/subtracter/logic unit is a function of its inputs. The inputs are driven by the upstream multiplexers, carry select logic, and multiplier array.

Equation 2-1 summarizes the combination of X, Y, Z, and CIN by the adder/subtracter. The CIN, X multiplexer output, and Y multiplexer output are always added together. This combined result can be selectively added to or subtracted from the Z multiplexer output. The second option is obtained by setting the ALUMODE to `0001`.

$$\text{Adder/Sub Out} = (Z \pm (X + Y + CIN)) \text{ or } (-Z + (X + Y + CIN) - 1) \qquad \text{Equation 2-1}$$

A typical use of the slice is where A and B inputs are multiplied and the result is added to or subtracted from the C register. More detailed operations based on control and data inputs are described in later sections. Selecting the multiplier function consumes both X and Y multiplexer outputs to feed the adder. The two 43-bit partial products from the multiplier are sign extended to 48 bits before being sent to the adder/subtracter.

When not using the first stage multiplier, the 48-bit, dual input, bitwise logic function implements AND, OR, NOT, NAND, NOR, XOR, and XNOR. The inputs to these functions are A:B, C, P, or PCIN selected through the X and Z multiplexers, with the Y multiplexer selecting either all 1s or all 0s depending on logic operation.

The output of the adder/subtracter or logic unit feeds the pattern detector logic. The pattern detector allows the DSP48E1 slice to support Convergent Rounding, Counter Autoreset when a count value has been reached, and Overflow/Underflow/Saturation in

accumulators. In conjunction with the logic unit, the pattern detector can be extended to perform a 48-bit dynamic comparison of two 48-bit fields. This enables functions such as A:B NAND C = = 0, or A:B (bitwise logic) C = = Pattern to be implemented.

Figure 2-5 shows the DSP48E1 slice in a very simplified form. The seven OPMODE bits control the selects of X, Y, and Z multiplexers, feeding the inputs to the adder/subtracter or logic unit. In all cases, the 43-bit partial product data from the multiplier to the X and Y multiplexers is sign extended, forming 48-bit input datapaths to the adder/subtracter. Based on 43-bit operands and a 48-bit accumulator output, the number of *guard bits* (i.e., bits available to guard against overflow) is 5. To extend the number of MACC operations, the MACC_EXTEND feature should be used, which allows the MACC to extend to 96 bits with two DSP48E1 slices. If A port is limited to 18 bits (sign-extended to 25), then there are 12 guard bits for the MACC. The CARRYOUT bits are invalid during multiply operations. Combinations of OPMODE, ALUMODE, CARRYINSEL, and CARRYIN control the function of the adder/subtracter or logic unit.

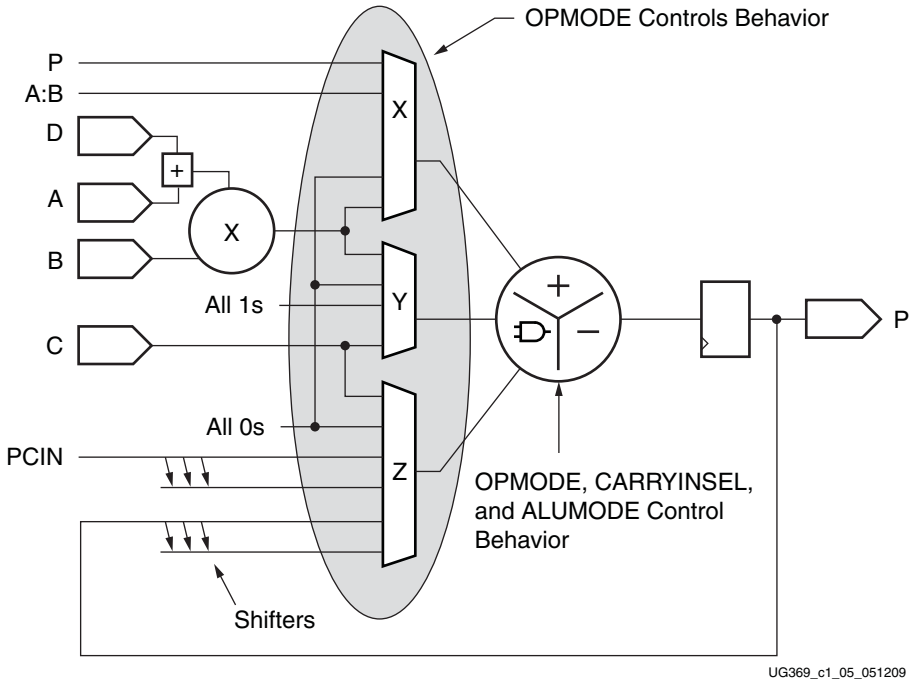


Figure 2-5: **Simplified DSP Slice Operation**

DSP48E1 Slice Attributes

The synthesis attributes for the DSP48E1 slice are described in this section. The attributes call out pipeline registers in the control and datapaths. The value of the attribute sets the number of pipeline registers. See Table 2-3.

Table 2-3: **Attribute Setting Description**

Attribute Name	Settings (Default)	Attribute Description
Register Control Attributes		
ACASCREG	0, 1, 2 (1)	In conjunction with AREG, selects the number of A input registers on the A cascade path, ACOUT. This attribute must be equal to or one less than the AREG value: AREG = 0: ACASCREG must be 0 AREG = 1: ACASCREG must be 1 AREG = 2: ACASCREG can be 1 or 2
ADREG	0, 1 (1)	Selects the number of AD pipeline registers.
ALUMODEREG	0, 1 (1)	Selects the number of ALUMODE input registers.
AREG	0, 1, 2 (1)	Selects the number of A input registers. When 1 is selected, the A2 register is used.
BCASCREG	0, 1, 2 (1)	In conjunction with BREG, selects the number of B input registers on the B cascade path, BCOUT. This attribute must be equal to or one less than the BREG value: BREG = 0: BCASCREG must be 0 BREG = 1: BCASCREG must be 1 BREG = 2: BCASCREG can be 1 or 2
BREG	0, 1, 2 (1)	Selects the number of B input registers. When 1 is selected, the B2 register is used.
CARRYINREG	0, 1 (1)	Selects the number of fabric CARRYIN input registers.
CARRYINSELREG	0, 1 (1)	Selects the number of CARRYINSEL input registers.
CREG	0, 1 (1)	Selects the number of C input registers.
DREG	0, 1 (1)	Selects the number of D input registers.
INMODEREG	0, 1 (1)	Selects the number of INMODE input registers.
MREG	0, 1 (1)	Selects the number of M pipeline registers.
OPMODEREG	0, 1 (1)	Selects the number of OPMODE input registers.
PREG	0, 1 (1)	Selects the number of P output registers (also used by CARRYOUT/ PATTERN_DETECT/ CARRYCASCOUT/ MULTSIGNOUT, etc.).
Feature Control Attributes		
A_INPUT	DIRECT, CASCADE (DIRECT)	Selects the input to the A port between parallel input (DIRECT) or the cascaded input from the previous slice (CASCADE).
B_INPUT	DIRECT, CASCADE (DIRECT)	Selects the input to the B port between parallel input (DIRECT) or the cascaded input from the previous slice (CASCADE).

Table 2-3: **Attribute Setting Description *(Cont'd)***

Attribute Name	Settings (Default)	Attribute Description
USE_DPORT	TRUE, FALSE (FALSE)	Determines whether the pre-adder and the D Port are used or not.
USE_MULT	NONE, MULTIPLY, DYNAMIC (MULTIPLY)	Selects usage of the multiplier. Set to NONE to save power when using only the Adder/Logic Unit. The DYNAMIC setting indicates that the user is switching between A*B and A:B operations on the fly and therefore needs to get the worst-case timing of the two paths.
USE_SIMD	ONE48, TWO24,FOUR12 (ONE48)	Selects the mode of operation for the adder/subtracter. The attribute setting can be one 48-bit adder mode (ONE48), two 24-bit adder mode (TWO24), or four 12-bit adder mode (FOUR12). Selecting ONE48 mode is compatible with Virtex-6 devices DSP48 operation and is not actually a true SIMD mode. Typical Multiply-Add operations are supported when the mode is set to ONE48. When either TWO24 or FOUR12 mode is selected, the multiplier must not be used, and USE_MULT must be set to NONE.
Pattern Detector Attributes		
AUTORESET_PATDET	NO_RESET, RESET_MATCH, RESET_NOT_MATCH (NO_RESET)	Automatically resets the P Register (accumulated value or counter value) on the next clock cycle, if a pattern detect event has occurred on this clock cycle. The RESET_MATCH and RESET_NOT_MATCH settings distinguish between whether the DSP48E1 slice should cause an auto reset of the P Register on the next cycle: • if the pattern is matched or • whenever the pattern is not matched on the current cycle but was matched on the previous clock cycle
MASK	48-bit field (`0011...11`)	This 48-bit value is used to mask out certain bits during a pattern detection. When a MASK bit is set to 1, the corresponding pattern bit is ignored. When a MASK bit is set to 0, the pattern bit is compared.
PATTERN	48-bit field (`00...00`)	This 48-bit value is used in the pattern detector.
SEL_MASK	MASK, C, ROUNDING_MODE1, ROUNDING_MODE2 (MASK)	Selects the mask to be used for the pattern detector. The C and MASK settings are for standard uses of the pattern detector (counter, overflow detection, etc.). ROUNDING_MODE1 (C-bar left shifted by 1) and ROUNDING_MODE2 (C-bar left shifted by 2) select special masks based off of the optionally registered C port. These rounding modes can be used to implement convergent rounding in the DSP48E1 slice using the pattern detector.

Table 2-3: **Attribute Setting Description *(Cont'd)***

Attribute Name	Settings (Default)	Attribute Description
SEL_PATTERN	PATTERN, C (PATTERN)	Selects the input source for the pattern field. The input source can either be a 48-bit dynamic "C" input or a 48-bit static attribute field.
USE_PATTERN_DETECT	NO_PATDET, PATDET (NO_PATDET)	Selects whether the pattern detector and related features are used (PATDET) or not used (NO_PATDET). This attribute is used for speed specification and Simulation Model purposes only.

Table 2-4: **Internal Register Descriptions**

Register	Description and Associated Attribute
2-Deep A Registers	These two optional registers for the A input are selected by AREG, enabled by CEA1 and CEA2, respectively, and reset synchronously by RSTA.
2-Deep B Registers	These two optional registers for the B input are selected by BREG, enabled by CEB1 and CEB2, respectively, and reset synchronously by RSTB.
AD Register	This register for the optional pre-adder result is selected by ADREG, enabled by CEAD, and reset synchronously by RSTD.
ALUMODE Register	This optional pipeline register for the ALUMODE control signal is selected by ALUMODEREG, enabled by CEALUMODE, and reset synchronously by RSTALUMODE.
C Register	This optional register for the C input is selected by CREG, enabled by CEC, and reset synchronously by RSTC.
CARRYIN Register	This optional pipeline register for the CARRYIN control signal is selected by CARRYINREG, enabled by CECARRYIN, and reset synchronously by RSTALLCARRYIN.
CARRYINSEL Register	This optional pipeline register for the CARRYINSEL control signal is selected by CARRYINSELREG, enabled by CECTRL, and reset synchronously by RSTCTRL.
D Register	This register for the optional D pre-adder input is selected by DREG, enabled by CED, and reset synchronously by RSTD.
INMODE Register	This 5-bit register selects the pre-adder and its mode, and the sign and the source of the A register feeding the multiplier. This register is selected by INMODEREG, enabled by CEINMODE, and reset synchronously by RSTINMODE.
Internal Mult Carry Register	This optional pipeline register for the Internal Carry signal (used for Multiply Symmetric Rounding Only) is enabled by CEM and reset synchronously by RSTM.
M Register	This optional pipeline register for the output of the multiplier consists of two 43-bit partial products. These two partial products are input into the X and Y multiplexers and finally into the adder/subtracter to create the product output. The M register is selected by MREG, enabled by CEM, and reset synchronously by RSTM.

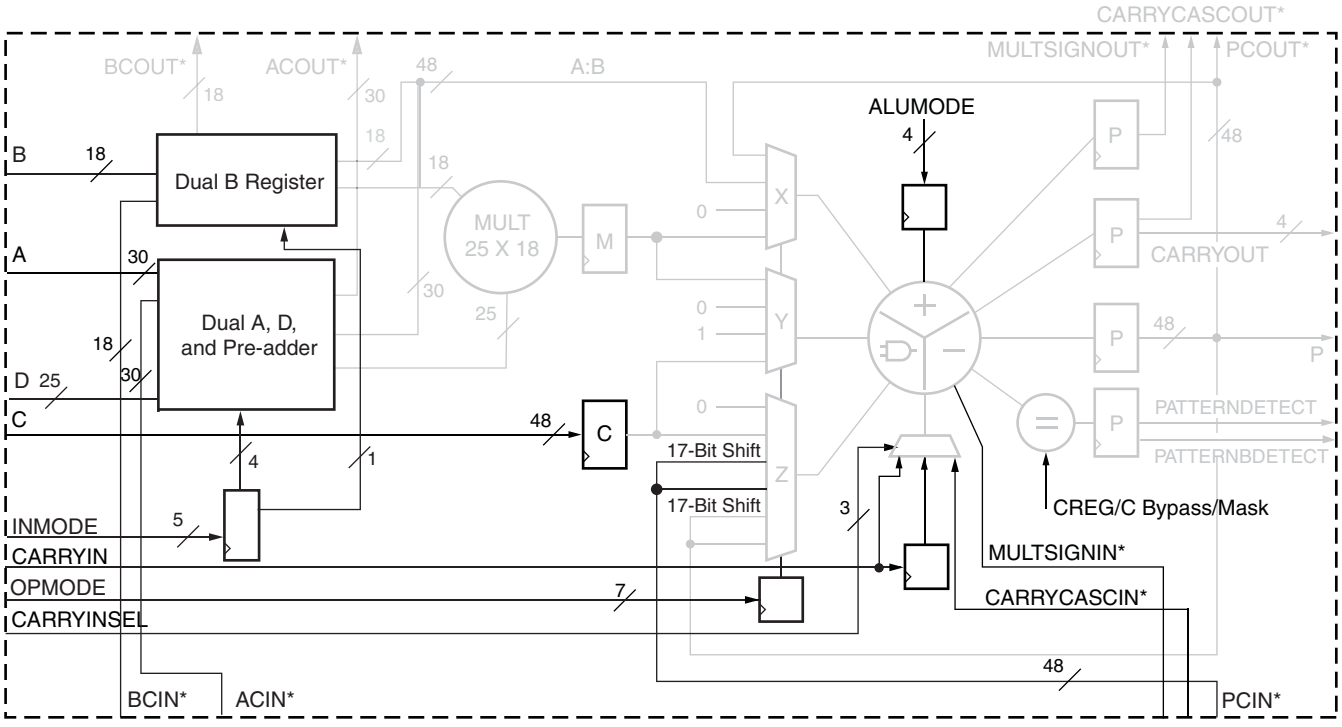
Table 2-4: **Internal Register Descriptions *(Cont'd)***

Register	Description and Associated Attribute
OPMODE Register	This optional pipeline register for the OPMODE control signal is selected by OPMODEREG, enabled by CECTRL, and reset synchronously by RSTCTRL.
Output Registers	This optional register for the P, OVERFLOW, UNDERFLOW, PATTERNDETECT, PATTERNDETECT, and CARRYOUT outputs is selected by PREG, enabled by CEP, and reset synchronously by RSTP. The same register also clocks out PCOUT, CARRYCASCOUT, and MULTSIGNOUT, which are the cascade outputs to the next DSP48E1 slice.

Input Ports

A, B, C, CARRYIN, CARRYINSEL, OPMODE, BCIN, PCIN, ACIN, ALUMODE, CARRYCASCIN, MULTSIGNIN along with the corresponding clock enable inputs and reset inputs, are legacy ports. The D and INMODE ports are unique to the DSP48E1 slice.

This section describes the input ports of the DSP48E1 slice in detail. The input ports of the DSP48E1 slice are highlighted in Figure 2-6.



*These signals are dedicated routing paths internal to the DSP48E1 column. They are not accessible via fabric routing resources.

UG369_c1_06_052109

Figure 2-6: **Input Ports in the DSP48E1 Slice**

A, B, C, and D Ports

The DSP48E1 slice input data ports support many common DSP and math algorithms. The DSP48E1 slice has four direct input data ports labeled A, B, C, and D. The A data port is 30 bits wide, the B data port is 18 bits wide, the C data port is 48 bits wide, and the pre-adder D data port is 25 bits wide.

The 25-bit A (A[24:0]) and 18-bit B ports supply input data to the 25-bit by 18-bit, two's complement multiplier. With independent C port, each DSP48E1 slice is capable of Multiply-Add, Multiply-Subtract, and Multiply-Round operations.

Concatenated A and B ports (A:B) bypass the multiplier and feed the X multiplexer input. The 30-bit A input port forms the upper 30 bits of A:B concatenated datapath, and the 18-bit B input port forms the lower 18 bits of the A:B datapath. The A:B datapath, together with the C input port, enables each DSP48E1 slice to implement a full 48-bit adder/subtracter provided the multiplier is not used, which is achieved by setting USE_MULT to NONE (or DYNAMIC).

Each DSP48E1 slice also has two cascaded input datapaths (ACIN and BCIN), providing a cascaded input stream between adjacent DSP48E1 slices. The cascaded path is 30 bits wide for the A input and 18 bits wide for the B input. Applications benefiting from this feature include FIR filters, complex multiplication, multi-precision multiplication and complex MACCs.

The A and B input port and the ACIN and BCIN cascade port can have 0, 1, or 2 pipeline stages in its datapath. The dual A, D, and pre-adder port logic is shown in Figure 2-7. The dual B register port logic is shown in Figure 2-8. The different pipestages are set using attributes. Attributes AREG and BREG are used to select the number of pipeline stages for A and B direct inputs. Attributes ACASCREG and BCASCREG select the number of pipeline stages in the ACOUT and BCOUT cascade datapaths. The allowed attribute settings are shown in Table 2-3. Multiplexers controlled by configuration bits select flow through paths, optional registers, or cascaded inputs. The data port registers allow users to typically trade off increased clock frequency (i.e., higher performance) versus data latency.

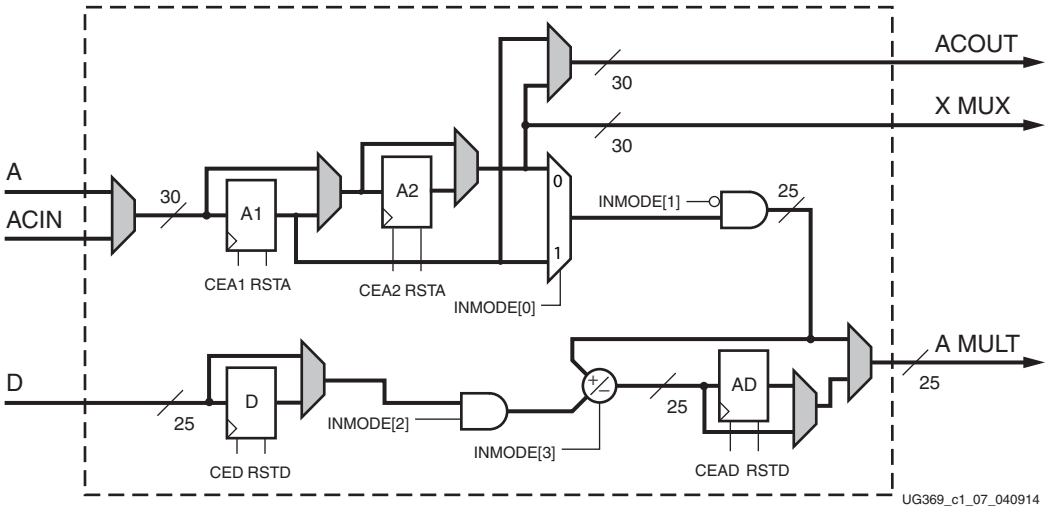


Figure 2-7: **Dual A, D, and Pre-adder Logic**

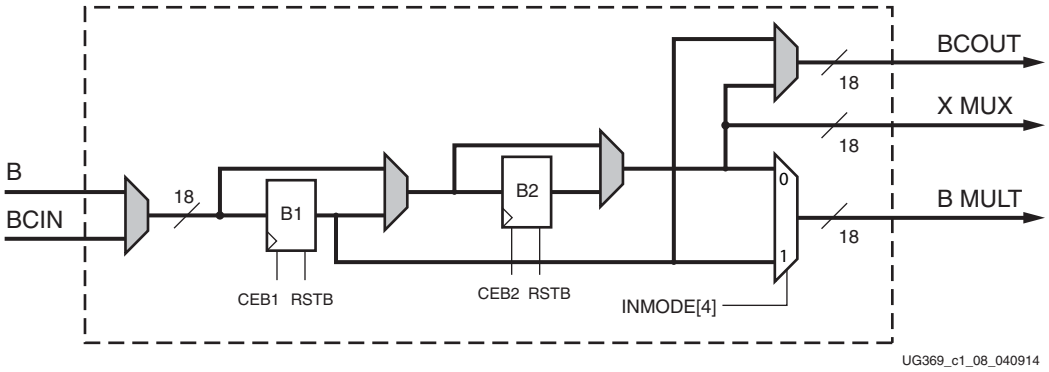


Figure 2-8: **Dual B Register Logic**

Table 2-5 shows the encoding for the INMODE[3:0] control bits. These bits select the functionality of the pre-adder, the A and D input registers. The USE_DPORT attribute must be set to TRUE to enable the pre-adder functions described in Table 2-5. INMODE[4], which selects between B1 and B2, is shown in Table 2-6.

In summary, the INMODE and USE_DPORT attributes control the pre-adder functionality and A, B, and D register bus multiplexers that precede the multiplier. If the pre-adder is not used, the default of USE_DPORT is FALSE.

Table 2-5: **INMODE[3:0] Functions (when AREG = 1 or 2)**

INMODE[3]	INMODE[2]	INMODE[1]	INMODE[0]	USE_DPORT	Multiplier A Port
0	0	0	0	FALSE	A2
0	0	0	1	FALSE	A1
0	0	1	0	FALSE	Zero
0	0	1	1	FALSE	Zero
0	0	0	0	TRUE	A2
0	0	0	1	TRUE	A1
0	0	1	0	TRUE	Zero
0	0	1	1	TRUE	Zero
0	1	0	0	TRUE	D + A2(1)
0	1	0	1	TRUE	D + A1(1)
0	1	1	0	TRUE	D
0	1	1	1	TRUE	D
1	0	0	0	TRUE	-A2
1	0	0	1	TRUE	-A1
1	0	1	0	TRUE	Zero
1	0	1	1	TRUE	Zero
1	1	0	0	TRUE	D – A2(1)
1	1	0	1	TRUE	D – A1(1)

Table 2-5: **INMODE[3:0] Functions (when AREG = 1 or 2) *(Cont'd)***

INMODE[3]	INMODE[2]	INMODE[1]	INMODE[0]	USE_DPORT	Multiplier A Port
1	1	1	0	TRUE	D
1	1	1	1	TRUE	D

Notes:

1. Set the data on the D and the A1/A2 ports so the pre-adder, which does not support saturation, does not overflow or underflow. See Pre-adder, page 40.

INMODE[0] selects between A1 (INMODE[0] = 1) and A2 (INMODE[0] = 0).

INMODE[1] =1 forces the A input to the pre-adder to 0.

INMODE[2] = 0 forces the D input to the pre-adder to 0.

INMODE[3] provides pre-adder subtract control, where INMODE[3] = 1 indicates subtract and INMODE[3] = 0 indicates add.

INMODE[4] selects the Multiplier B port as shown in Table 2-6.

Table 2-6: **INMODE[4] Encoding (when BREG = 1 or 2)**

INMODE[4]	Multiplier B Port
0	B2
1	B1

The 48-bit C port is used as a general input to the Y and Z multiplexers to perform add, subtract, three-input add/subtract, and logic functions. The C input is also connected to the pattern detector for rounding function implementations. The C port logic is shown in Figure 2-9. The CREG attribute selects the number of pipestages for the C input datapath.

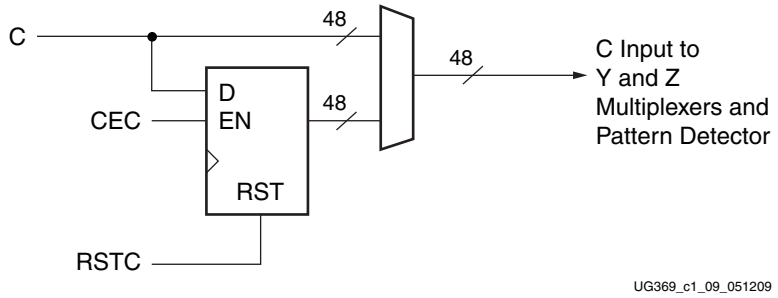


Figure 2-9: **C Port Logic**

OPMODE, ALUMODE, and CARRYINSEL Port Logic

The OPMODE, ALUMODE, and CARRYINSEL port logic supports flow through or registered input control signals. Multiplexers controlled by configuration bits select flow through or optional registers. The control port registers allow users to trade off increased clock frequency (that is, higher performance) versus data latency. The registers have independent clock enables and resets. The OPMODE and CARRYINSEL registers are reset by RSTCTRL. The ALUMODE is reset by RSTALUMODE. The clock enables and the OPMODE, ALUMODE, and CARRYINSEL port logic are shown in Figure 2-10.

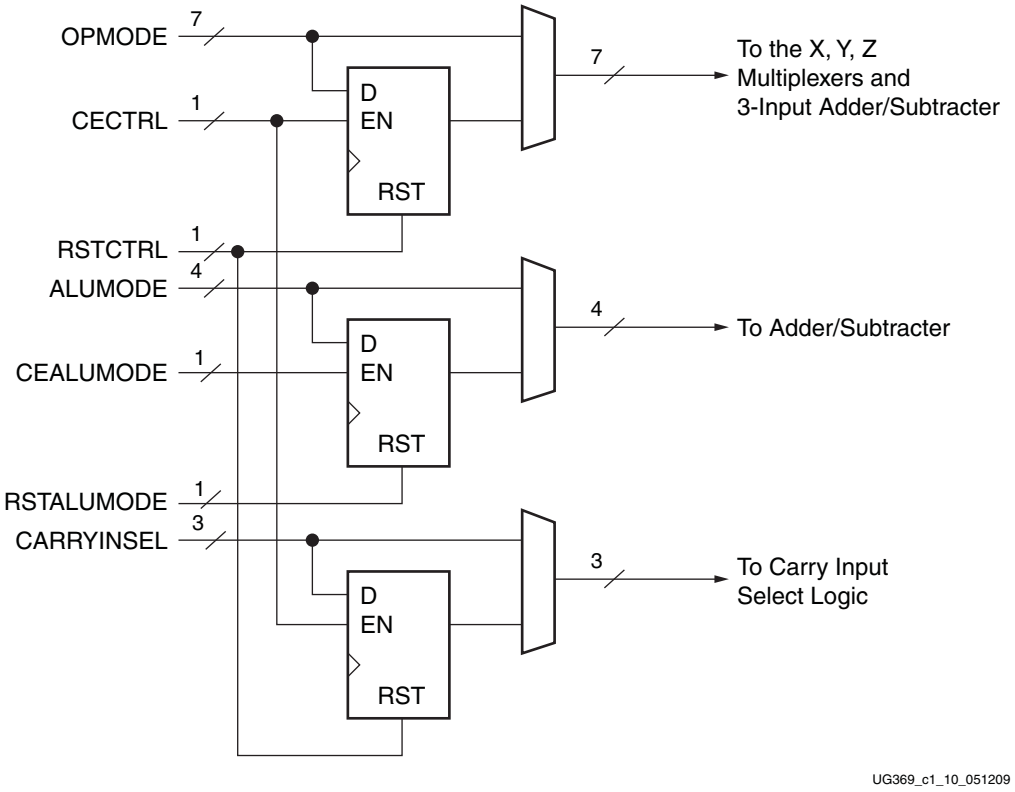


Figure 2-10: **OPMODE, ALUMODE, and CARRYINSEL Port Logic**

X, Y, and Z Multiplexers

The OPMODE (Operating Mode) control input contains fields for X, Y, and Z multiplexer selects.

The OPMODE input provides a way for you to dynamically change DSP48E1 functionality from clock cycle to clock cycle (for example, altering the internal datapath configuration of the DSP48E1 slice relative to a given calculation sequence).

The OPMODE bits can be optionally registered using the OPMODEREG attribute (as noted in Table 2-3).

Table 2-7, Table 2-8, and Table 2-9 list the possible values of OPMODE and the resulting function at the outputs of the three multiplexers (X, Y, and Z multiplexers). The multiplexer outputs supply three operands to the following adder/subtracter. Not all possible combinations for the multiplexer select bits are allowed. Some are marked in the tables as "illegal selection" and give undefined results. If the multiplier output is selected, then both the X and Y multiplexers are used to supply the multiplier partial products to the adder/subtracter.

Table 2-7: **OPMODE Control Bits Select X Multiplexer Outputs**

Z OPMODE[6:4]	Y OPMODE[3:2]	X OPMODE[1:0]	X Multiplexer Output	Notes
xxx	xx	00	0	Default
xxx	01	01	M	Must select with OPMODE[3:2] = 01
xxx	xx	10	P	Must select with PREG = 1
xxx	xx	11	A:B	48 bits wide

Table 2-8: **OPMODE Control Bits Select Y Multiplexer Outputs**

Z OPMODE[6:4]	Y OPMODE[3:2]	X OPMODE[1:0]	Y Multiplexer Output	Notes
xxx	00	xx	0	Default
xxx	01	01	M	Must select with OPMODE[1:0] = 01
xxx	10	xx	48'FFFFFFFFFFFF	Used mainly for logic unit bitwise operations on the X and Z multiplexers
xxx	11	xx	C	

Table 2-9: **OPMODE Control Bits Select Z Multiplexer Outputs**

Z OPMODE[6:4]	Y OPMODE[3:2]	X OPMODE[1:0]	Z Multiplexer Output	Notes
000	xx	xx	0	Default
001	xx	xx	PCIN	
010	xx	xx	P	Must select with PREG = 1
011	xx	xx	C	
100	10	00	P	Use for MACC extend only. Must select with PREG = 1
101	xx	xx	17-bit Shift (PCIN)	
110	xx	xx	17-bit Shift (P)	Must select with PREG = 1
111	xx	xx	xx	Illegal selection

ALUMODE Inputs

The 4-bit ALUMODE controls the behavior of the second stage add/sub/logic unit. ALUMODE = 0000 selects add operations of the form Z + (X + Y + CIN). CIN is the output of the CARRYIN mux (see Figure 2-11). ALUMODE = 0011 selects subtract operations of the form Z – (X + Y + CIN). ALUMODE = 0001 can implement –Z + (X + Y + CIN) – 1. ALUMODE = 0010 can implement –(Z + X + Y + CIN) – 1, which is equivalent to not (Z + X + Y + CIN). The negative of a two's complement number is obtained by performing a bitwise inversion and adding one, for example, –k = not (k) + 1. Other subtract and logic operations can also be implemented with the enhanced add/sub/logic unit. See Table 2-10.

Table 2-10: **Three-Input ALUMODE Operations**

DSP Operation	OPMODE[6:0]	ALUMODE[3:0]			
		3	2	1	0
Z + X + Y + CIN	Any legal OPMODE	0	0	0	0
Z – (X + Y + CIN)	Any legal OPMODE	0	0	1	1
–Z + (X + Y + CIN) – 1 = not (Z) + X + Y + CIN	Any legal OPMODE	0	0	0	1
not (Z + X + Y + CIN) = –Z – X – Y – CIN - 1	Any legal OPMODE	0	0	1	0

Notes:

1. In two's complement: –Z = not (Z) + 1

See Table 2-13, page 42 for two-input ALUMODE operations and Figure A-3, page 56.

Carry Input Logic

In 7 series devices, the carry input logic result is a function of a 3-bit CARRYINSEL signal. The inputs to the carry input logic appear in Figure 2-11. Carry inputs used to form results for adders and subtracters are always in the critical path. High performance is achieved by implementing this logic in silicon. The possible carry inputs to the carry logic are "gathered" prior to the outputs of the X, Y, and Z multiplexers. In 7 series devices, CARRYIN has no dependency on the OPMODE selection.

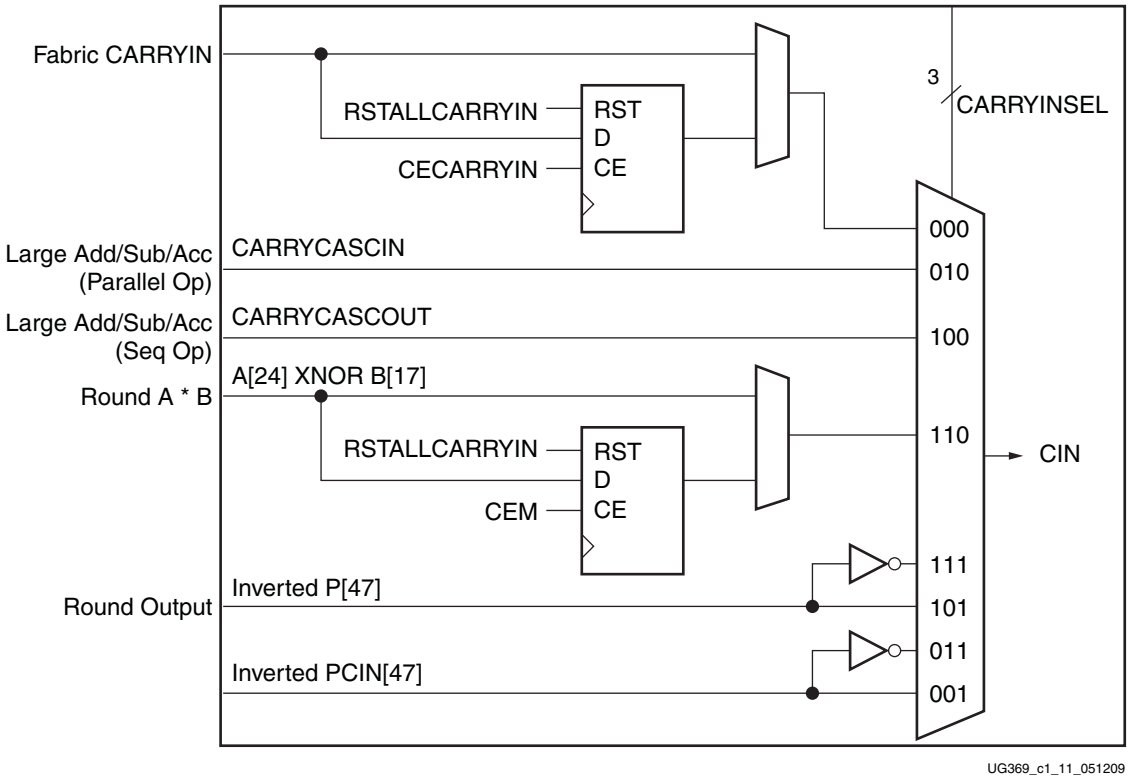


Figure 2-11: **CARRYINSEL Port Logic**

Figure 2-11 shows eight inputs selected by the 3-bit CARRYINSEL control. The first input, CARRYIN (CARRYINSEL set to binary 000), is driven from general logic. This option allows implementation of a carry function based on user logic. CARRYIN can be optionally registered. The next input, (CARRYINSEL is equal to binary 010) is the CARRYCASCIN input from an adjacent DSP48E1 slice. The third input (CARRYINSEL is equal to binary 100) is the CARRYCASCOUT from the same DSP48E1 slice, fed back to itself. Refer to Table 2-4 for internal register definitions pertaining to carry logic.

The fourth input (CARRYINSEL is equal to binary 110) is A[24] XNOR B[17] for symmetrically rounding multiplier outputs. This signal can be optionally registered to match the MREG pipeline delay. The fifth and sixth inputs (CARRYINSEL is equal to binary 111 and 101) selects the true or inverted P output MSB P[47] for symmetrically rounding the P output. The seventh and eight inputs (CARRYINSEL is equal to binary 011 and 001) selects the true or inverted cascaded P input MSB PCIN[47] for symmetrically rounding the cascaded P input.

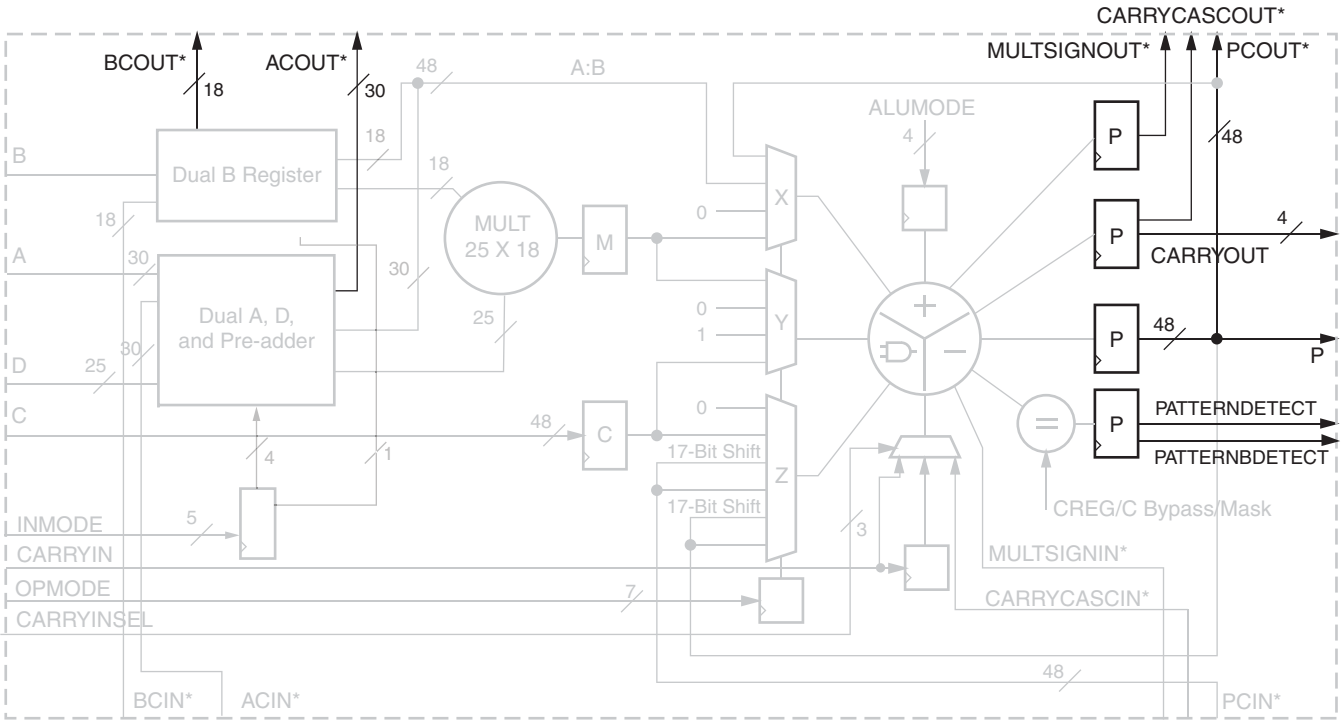
Table 2-11 lists the possible values of the three carry input select bits (CARRYINSEL) and the resulting carry inputs or sources.

Table 2-11: **CARRYINSEL Control Carry Source**

CARRYINSEL			Select	Notes
2	1	0		
0	0	0	CARRYIN	General interconnect
0	0	1	~PCIN[47]	Rounding PCIN (round towards infinity)
0	1	0	CARRYCASCIN	Larger add/sub/acc (parallel operation)
0	1	1	PCIN[47]	Rounding PCIN (round towards zero)
1	0	0	CARRYCASCOUT	For larger add/sub/acc (sequential operation via internal feedback). Must select with PREG = 1
1	0	1	~P[47]	Rounding P (round towards infinity). Must select with PREG = 1
1	1	0	A[24] XNOR B[17]	Rounding A x B
1	1	1	P[47]	For rounding P (round towards zero). Must select with PREG = 1

Output Ports

This section describes the output ports of the 7 series FPGA DSP48E1 slice in detail. The output ports of the DSP48E1 slice are shown in Figure 2-12.



*These signals are dedicated routing paths internal to the DSP48E1 column. They are not accessible via fabric routing resources.

UG369_c1_12_052109

Figure 2-12: **Output Ports in the DSP48E1 Slice**

All the output ports except ACOUT and BCOUT are reset by RSTP and enabled by CEP (see Figure 2-13). ACOUT and BCOUT are reset by RSTA and RSTB, respectively (shown in Figure 2-7 and Figure 2-8).

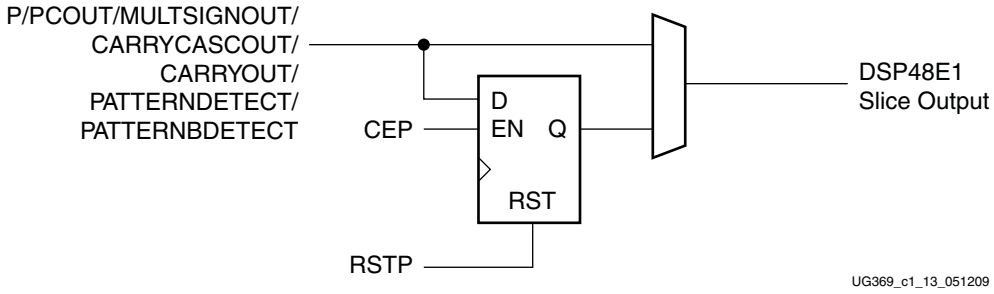


Figure 2-13: **Output Port Logic**

P Port

Each DSP48E1 slice has a 48-bit output port P. This output can be connected (cascaded connection) to the adjacent DSP48E1 slice internally through the PCOUT path. The PCOUT connects to the input of the Z multiplexer (PCIN) in the adjacent DSP48E1 slice. This path provides an output cascade stream between adjacent DSP48E1 slices.

CARRYCASCOUT and CARRYOUT Ports

The carry out from each DSP48E1 slice can be sent to the FPGA logic using the CARRYOUT port. This port is 4 bits wide. CARRYOUT[3] is the valid carry output for a two-input 48-bit adder/subtracter or one-input accumulator. In this case, USE_SIMD = ONE48 is the default setting and represents a non-SIMD configuration. When a two-input adder/subtracter or one-input accumulator is used in SIMD mode, such as TWO24 or FOUR12, the valid CARRYOUT signals are listed in Table 2-12. The CARRYOUT signals are not valid if three-input adder/subtracter (for example, A:B + C + PCIN) or two-input accumulator (for example, A:B + C + P) configurations are used or if the multiplier is used.

Table 2-12: **CARRYOUT Bit Associated with Different SIMD Mode**

SIMD Mode	Adder Bit Width	Corresponding CARRYOUT
FOUR12	P[11:0]	CARRYOUT[0]
	P[23:12]	CARRYOUT[1]
	P[35:24]	CARRYOUT[2]
	P[47:36]	CARRYOUT[3]
TWO24	P[23:0]	CARRYOUT[1]
	P[47:24]	CARRYOUT[3]
ONE48	P[47:0]	CARRYOUT[3]

See also Table 2-10, page 35 for 3-input ALUMODE operations.

The CARRYOUT signal is cascaded to the next adjacent DSP48E1 slice using the CARRYCASCOUT port. Larger add, subtract, ACC, and MACC functions can be implemented in the DSP48E1 slice using the CARRYCASCOUT output port. The 1-bit CARRYCASCOUT signal corresponds to CARRYOUT[3], but is not identical. The

CARRYCASCOUT signal is also fed back into the same DSP48E1 slice via the CARRYINSEL multiplexer.

The CARRYOUT[3] signal should be ignored when the multiplier or a ternary add/subtract operation is used. Because a MACC operation includes a three-input adder in the accumulator stage, the combination of MULTSIGNOUT and CARRYCASCOUT signals is required to perform a 96-bit MACC, spanning two DSP48E1 slices. The second DSP48E1 slice OPMODE must be MACC_EXTEND (`1001000`) to use both CARRYCASCOUT and MULTSIGNOUT, thereby eliminating the ternary adder carry restriction for the upper DSP48E1 slice. The actual hardware implementation of CARRYOUT/CARRYCASCOUT and the differences between them are described in Appendix A, CARRYOUT, CARRYCASCOUT, and MULTSIGNOUT.

MULTSIGNOUT Port Logic

MULTSIGNOUT is a software abstraction of the hardware signal. It is modeled as the MSB of the multiplier output and used only in MACC extension applications to build a 96-bit MACC. The actual hardware implementation of MULTSIGNOUT is described in Appendix A, CARRYOUT, CARRYCASCOUT, and MULTSIGNOUT.

The MSB of a multiplier output is cascaded to the next DSP48E1 slice using the MULTSIGNIN port and can be used only in MACC extension applications to build a 96-bit accumulator. The actual hardware implementation of MULTSIGNOUT is described in Appendix A, CARRYOUT, CARRYCASCOUT, and MULTSIGNOUT.

PATTERNDETECT and PATTERNBDETECT Port Logic

A pattern detector has been added on the output of the DSP48E1 slice to detect if the P bus matches a specified pattern or if it exactly matches the complement of the pattern. The PATTERNDETECT (PD) output goes High if the output of the adder matches a set pattern. The PATTERNBDETECT (PBD) output goes High if the output of the adder matches the complement of the set pattern.

A mask field can also be used to hide certain bit locations in the pattern detector. PATTERNDETECT computes ((P == pattern)||mask) on a bitwise basis and then ANDs the results to a single output bit. Similarly, PATTERNBDETECT can detect if ((P == ~pattern)||mask). The pattern and the mask fields can each come from a distinct 48-bit configuration field or from the (registered) C input. When the C input is used as the PATTERN, the OPMODE should be set to select a 0 at the input of the Z multiplexer. If all the registers are reset, PATTERNDETECT is High for one clock cycle immediately after the RESET is deasserted.

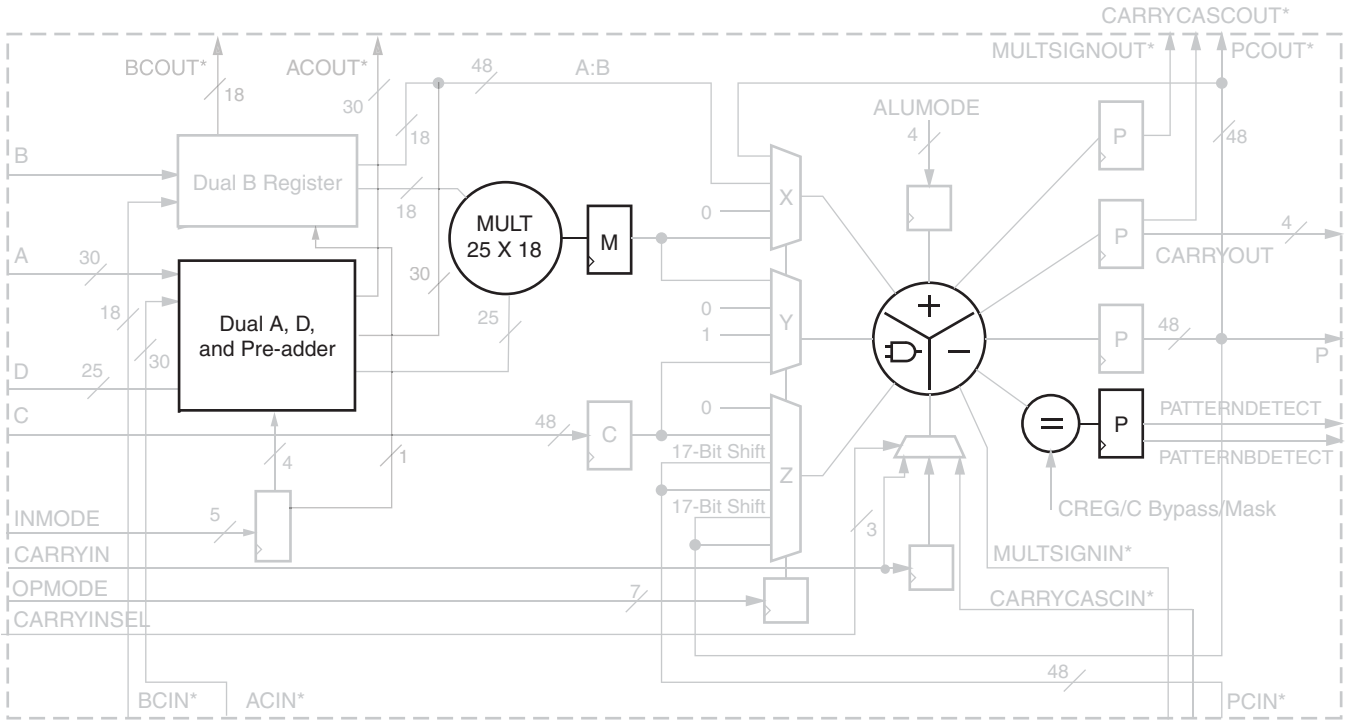
The pattern detector allows the DSP48E1 slice to support convergent rounding and counter auto reset when a count value has been reached as well as support overflow, underflow, and saturation in accumulators.

Overflow and Underflow Port Logic

The dedicated OVERFLOW and UNDERFLOW outputs of the DSP48E1 slice use the pattern detector to determine if the operation in the DSP48E1 slice has overflowed beyond the P[N] bit where N is between 1 and 46. The P register must be enabled while using overflow and underflow ports. This is further described in the Embedded Functions section.

Embedded Functions

The embedded functions in 7 series devices include a 25 x 18 multiplier, adder/subtracter/logic unit, and pattern detector logic (see Figure 2-14).



*These signals are dedicated routing paths internal to the DSP48E1 column. They are not accessible via fabric routing resources.

UG369_c1_14_052109

Figure 2-14: **Embedded Functions in a DSP48E1 Slice**

Pre-adder

The 7 series FPGA DSP slice has a 25-bit pre-adder, which is inserted in the A register path (shown in Figure 2-14 with an expanded view in Figure 2-7, page 30). With the pre-adder, pre-additions or pre-subtractions are possible prior to feeding the multiplier. Since the pre-adder does not contain saturation logic, designers should limit input operands to 24-bit two's complement sign-extended data to avoid overflow or underflow during arithmetic operations. Optionally, the pre-adder can be bypassed, making D the new input path to the multiplier. When the D path is not used, the output of the A pipeline can be negated prior to driving the multiplier. There are up to 10 operating modes, making this pre-adder block very flexible.

In Equation 2-2, A and D are added initially through the pre-adder/subtracter. The result of the pre-adder is then multiplied against B, with the result of the multiplication being added to the C input. This equation facilitates efficient symmetric filters.

$$\text{Adder/Subtracter Output} = C \pm (B \times (D \pm A) + C_{IN}) \qquad \textit{Equation 2-2}$$

Two's Complement Multiplier

The two's complement multiplier in the DSP48E1 slice in Figure 2-14 accepts a 25-bit two's complement input and an 18-bit two's complement input. The multiplier produces two 43-bit partial products. The two partial products together give an 86-bit result at the output of the multiplier, as shown in Figure 2-15. Cascading of multipliers to achieve larger products is supported with a 17-bit, right-shifted, cascaded output bus. The right shift is used to right justify the partial products by the correct number of bits. This cascade path feeds into the Z multiplexer, which is connected to the adder/subtracter of an adjacent DSP48E1 slice. The multiplier can emulate unsigned math by setting the MSB of an input operand to zero.

Figure 2-15 shows an optional pipeline register (MREG) for the output of the multiplier. Using the register provides increased performance with an increase of one clock latency.

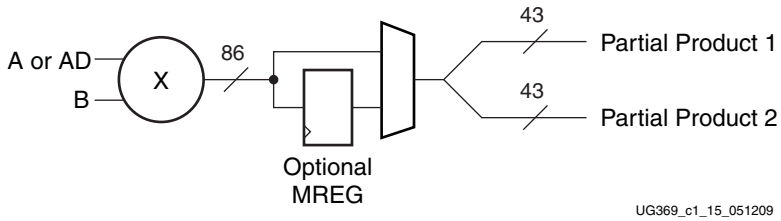


Figure 2-15: **Two's Complement Multiplier Followed by Optional MREG**

Adder/Subtracter or Logic Unit

The adder/subtracter or logic unit output is a function of control and data inputs (see Figure 2-16). The data inputs to the adder/subtracter are selected by the OPMODE and the CARRYINSEL signals. The ALUMODE signals choose the function implemented in the adder/subtracter. Thus, the OPMODE, ALUMODE, and CARRYINSEL signals together determine the functionality of the embedded adder/subtracter/logic unit. When using the logic unit, the multiplier must not be used. The values of OPMODEREG and CARRYINSELREG must be identical.

As with the input multiplexers, the OPMODE bits specify a portion of this function. The symbol ± in the table means either add or subtract and is specified by the state of the ALUMODE control signal. The symbol ":" in the table means concatenation. The outputs of the X and Y multiplexer and CIN are always added together. Refer to ALUMODE Inputs, page 35.

Two-Input Logic Unit

In 7 series devices, the capability to perform an addition, subtraction, and simple logic functions in the DSP48E1 slice exists through use of a second-stage, three-input adder.

Table 2-13 lists the logic functions that can be implemented in the second stage of the three input adder/subtracter/logic unit. The table also lists the settings of the control signals, namely OPMODE and ALUMODE.

Setting OPMODE[3:2] to "`00`" selects the default "`0`" value at the Y multiplexer output. OPMODE[3:2] set to "`10`" selects "`all 1s`" at the Y multiplexer output. OPMODE[1:0] selects the output of the X multiplexer, and OPMODE[6:4] selects the output of the Z multiplexer.

Table 2-13: **OPMODE and ALUMODE Control Bits Select Logic Unit Outputs**

Logic Unit Mode	OPMODE[3:2]		ALUMODE[3:0]			
	3	2	3	2	1	0
X XOR Z	0	0	0	1	0	0
X XNOR Z	0	0	0	1	0	1
X XNOR Z	0	0	0	1	1	0
X XOR Z	0	0	0	1	1	1
X AND Z	0	0	1	1	0	0
X AND (NOT Z)	0	0	1	1	0	1
X NAND Z	0	0	1	1	1	0
(NOT X) OR Z	0	0	1	1	1	1
X XNOR Z	1	0	0	1	0	0
X XOR Z	1	0	0	1	0	1
X XOR Z	1	0	0	1	1	0
X XNOR Z	1	0	0	1	1	1
X OR Z	1	0	1	1	0	0
X OR (NOT Z)	1	0	1	1	0	1
X NOR Z	1	0	1	1	1	0
(NOT X) AND Z	1	0	1	1	1	1

Single Instruction, Multiple Data (SIMD) Mode

The 48-bit adder/subtracter/accumulator can be split into smaller data segments where the internal carry propagation between segments is blocked to ensure independent operation for all segments. The adder/subtracter/accumulator can be split into four 12-bit adder/subtracter/accumulators or two 24-bit adder/subtracter/accumulators with carry out signal per segment. The SIMD mode segmentation is a static configuration as opposed to dynamic OPMODE type control (see Figure 2-16).

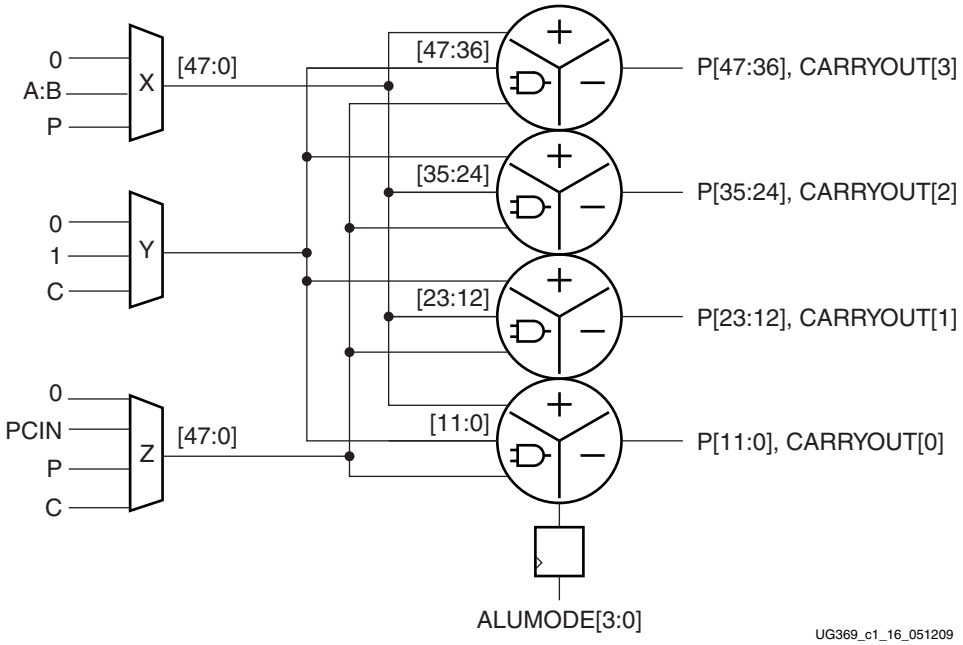


Figure 2-16: **Four 12-Bit SIMD Adder Configuration**

- Four segments of dual or ternary adders with 12-bit inputs, a 12-bit output, and a carry output for each segment
- Function controlled dynamically by ALUMODE[3:0], and operand source by OPMODE[6:0]
- All four adder/subtracter/accumulators perform same function
- Two segments of dual or ternary adders with 24-bit inputs, a 24-bit output, and a carry output for each segment is also available (not pictured).

The SIMD feature, shown in Figure 2-16, allows the 48-bit logic unit to be split into multiple smaller logic units. Each smaller logic unit performs the same function. This function can also be changed dynamically through the ALUMODE[3:0] and opmode control inputs.

Pattern Detect Logic

The pattern detector is connected to the output of the add/subtract/logic unit in the DSP48E1 slice (see Figure 2-14).

The pattern detector is best described as an equality check on the output of the adder/subtracter/logic unit that produces its result on the same cycle as the P output. There is no extra latency between the pattern detect output and the P output of the DSP48E1 slice. The use of the pattern detector leads to a moderate speed reduction due to the extra logic on the pattern detect path (see Figure 2-17).

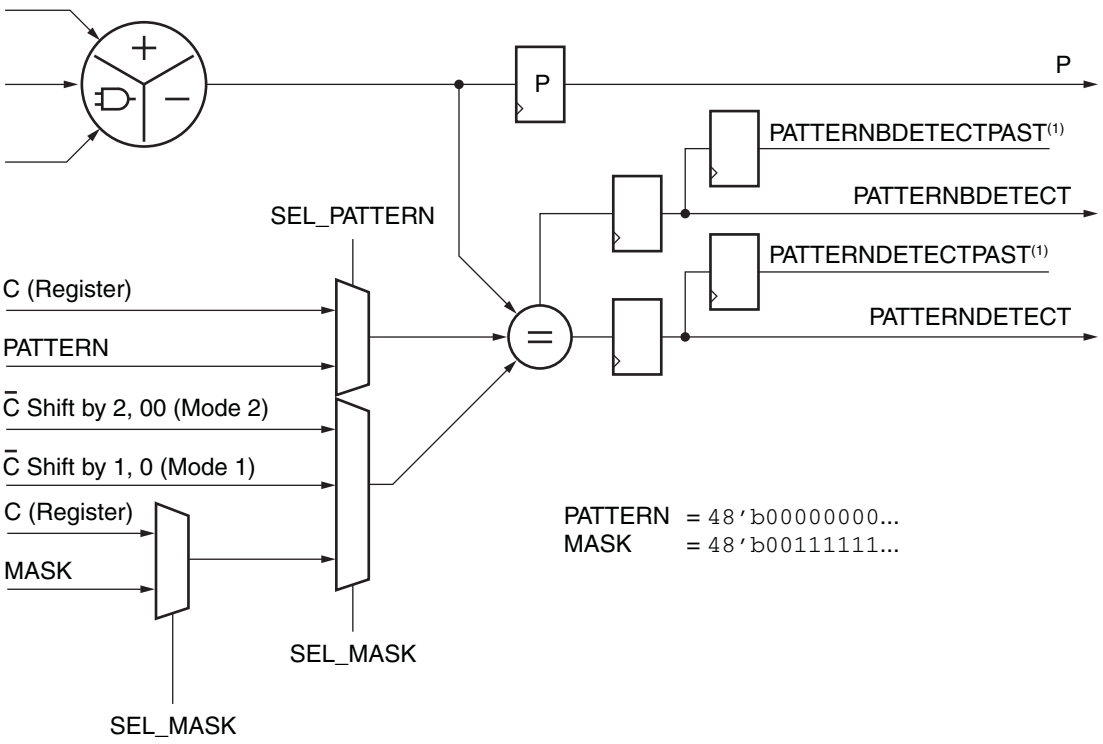


Figure 2-17: **Pattern Detector Logic**

Some of the applications that can be implemented using the pattern detector are:

- Pattern detect with optional mask
- Dynamic C input pattern match with A x B
- Overflow/underflow/saturation past P[46]
- A:B == C and dynamic pattern match, for example, A:B OR C == 0, A:B AND C == 1
- A:B {function} C == 0
- 48-bit counter auto reset (terminal count detection)
- Detecting mid points for rounding operations

If the pattern detector is not being employed, it can be used for other creative design implementations. These include:

- Duplicating a pin (for example, the sign bit) to reduce fanout and thus increase speed.
- Implementing a built-in inverter on one bit (for example, the sign bit) without having to route out to the fabric.
- Checking for sticky bits in floating point, handling special cases, or monitoring the DSP48E1 slice outputs.
- Raising a flag if a certain condition is met or if a certain condition is no longer met.

A mask field can also be used to mask out certain bit locations in the pattern detector. The pattern field and the mask field can each come from a distinct 48-bit memory cell field or from the (registered) C input.

Overflow and Underflow Logic

The discussion of overflow and underflow below applies to sequential accumulators (MACC or Adder-Accumulator) implemented in a single DSP48E1 slice. The accumulator should have at least one guard bit. When the pattern detector is set to detect a pattern equal to `00000…0` with a mask of `0011111 …1` (default settings), the DSP48E1 slice flags overflow beyond `00111 … 1` or underflow beyond `11000… 0`. The USE_PATTERN_DETECT attribute is set to PATDET to enable the use of the pattern detect logic. This overflow/underflow implementation uses a redundant sign bit and reduces the output bit width to 47 bits.

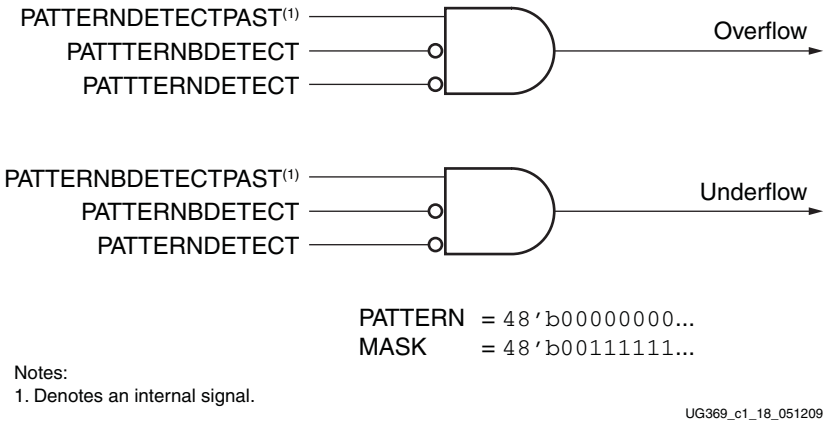


Figure 2-18: **Overflow/Underflow Logic in Pattern Detect**

By setting the mask to other values like `0000111 …1`, the bit value P[N] at which overflow is detected can be changed. This logic supports saturation to a positive number of $2^N - 1$ and a negative number of 2^N in two's complement where N is the number of 1s in the mask field.

To check overflow/underflow condition for N = 2, the following example is used:

- Mask is set to `0...11`.
- The (N) LSB bits are not considered for the comparison.
- For N = 2, the legal values (patterns) are 2^2-1 to -2^2 or 3 to –4.

See Figure 2-19 and Figure 2-20 for overflow and underflow examples, respectively.

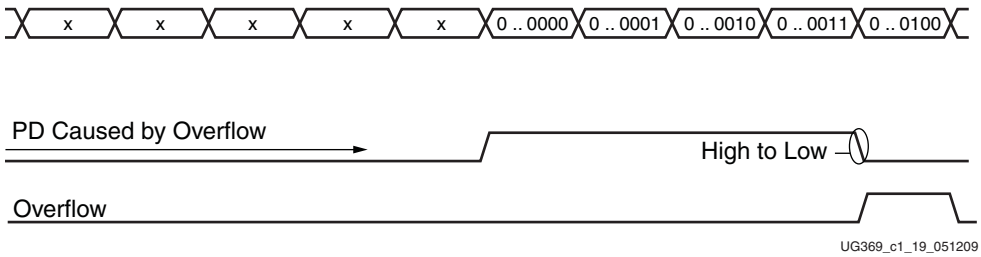


Figure 2-19: **Overflow Condition in the Pattern Detector**

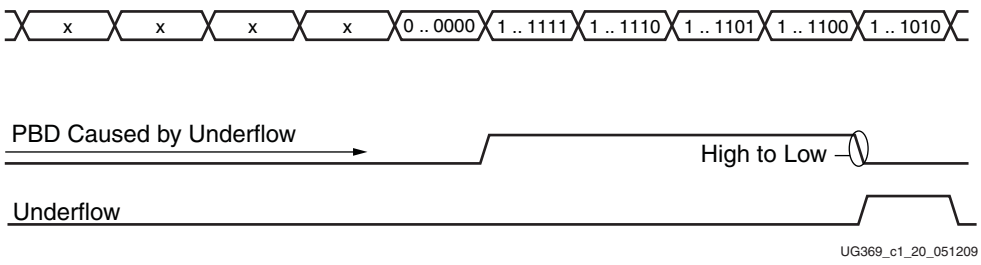


Figure 2-20: **Underflow Condition in the Pattern Detector**

- PD is 1 if P == pattern or mask
- PBD is a 1 if P == patternb or mask

Overflow is caused by addition when the value at the output of the adder/subtracter/logic unit goes over 3. Adding 1 to the final value of `0..0011` gives `0..0100` as the result, which causes the PD output to go to 0. When the PD output goes from 1 to 0, an overflow is flagged.

Underflow is caused by subtraction when the value goes below –4. Subtracting 1 from `1..1100` yields `1..1010` (–5), which causes the PBD output to go to 0. When the PBD output goes from 1 to 0, an underflow is flagged.

XILINX

Chapter 3

DSP48E1 Design Considerations

This chapter describes some design features and techniques to use to achieve higher performance, lower power, and lower resources in a particular design.

This chapter contains the following sections:

Designing for Performance

To achieve maximum performance when using the DSP48E1 slice, the design needs to be fully pipelined. For multiplier-based designs, the DSP48E1 slice requires a three-stage pipeline. For non-multiplier-based designs, a two-stage pipeline should be used. Also see the 7 series FPGA data sheets [Ref 6]. If latency is important in the design and only one or two registers can be used within the DSP48E1 slice, always use the M register.

Designing for Power

The USE_MULT attribute selects usage of the multiplier. This attribute should be set to NONE to save power when using only the Adder/Logic Unit. Functions implemented in the DSP48E1 slice use less power than those implemented in fabric. Using the cascade paths within the DSP48E1 slice instead of fabric routing is another way to reduce power. A multiplier with the M register turned on uses less power than one where the M register is not used. For operands less than 25 x 18, fabric power can be reduced by placing operands into the MSBs and zero padding unused LSBs.

Adder Tree Versus Adder Cascade

Adder Tree

In typical direct form FIR filters, an input stream of samples is presented to one input of the multipliers in the DSP48E1 slices. The coefficients supply the other input to the multipliers. An adder tree is used to combine the outputs from many multipliers as shown in Figure 3-1.

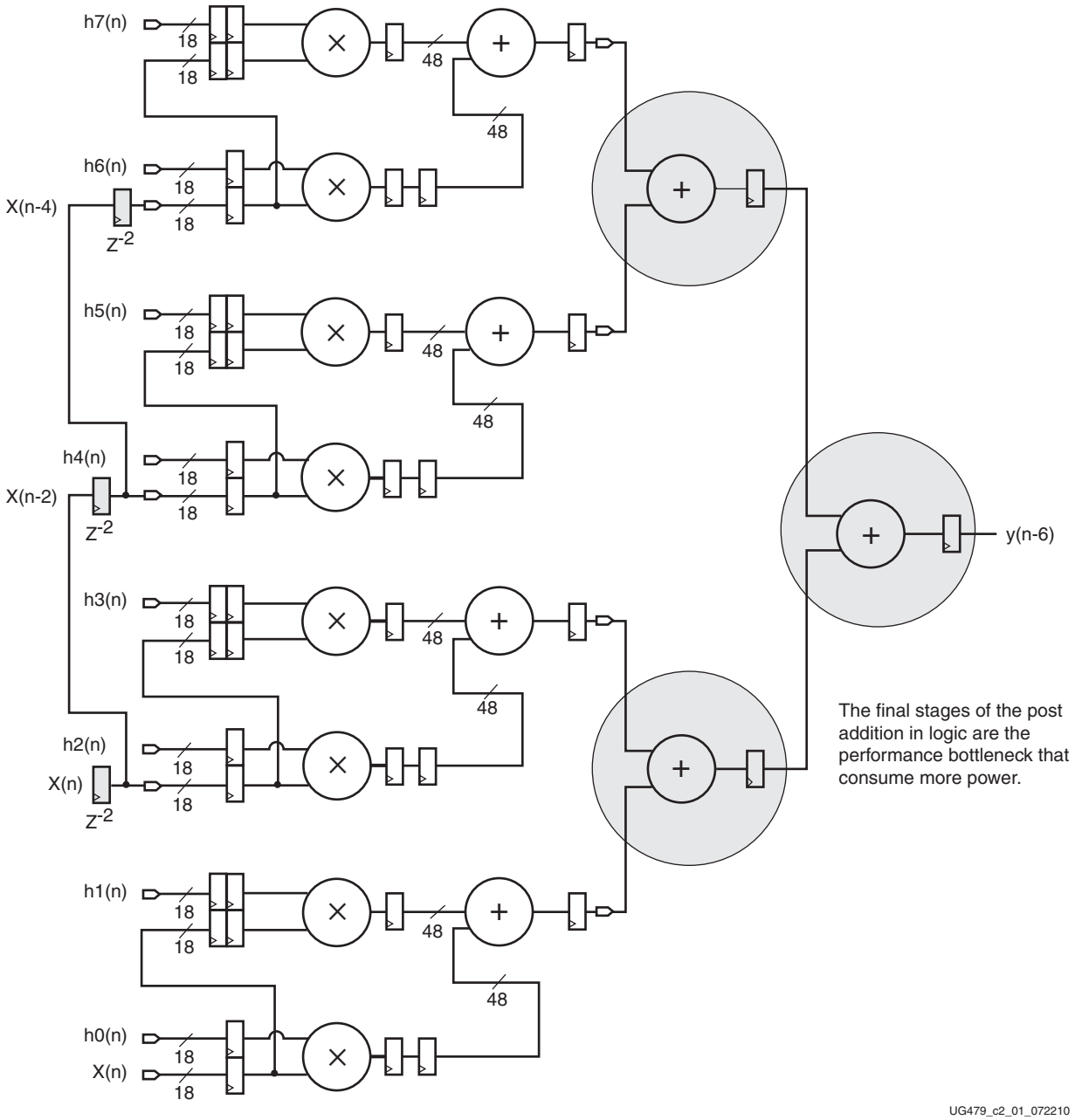
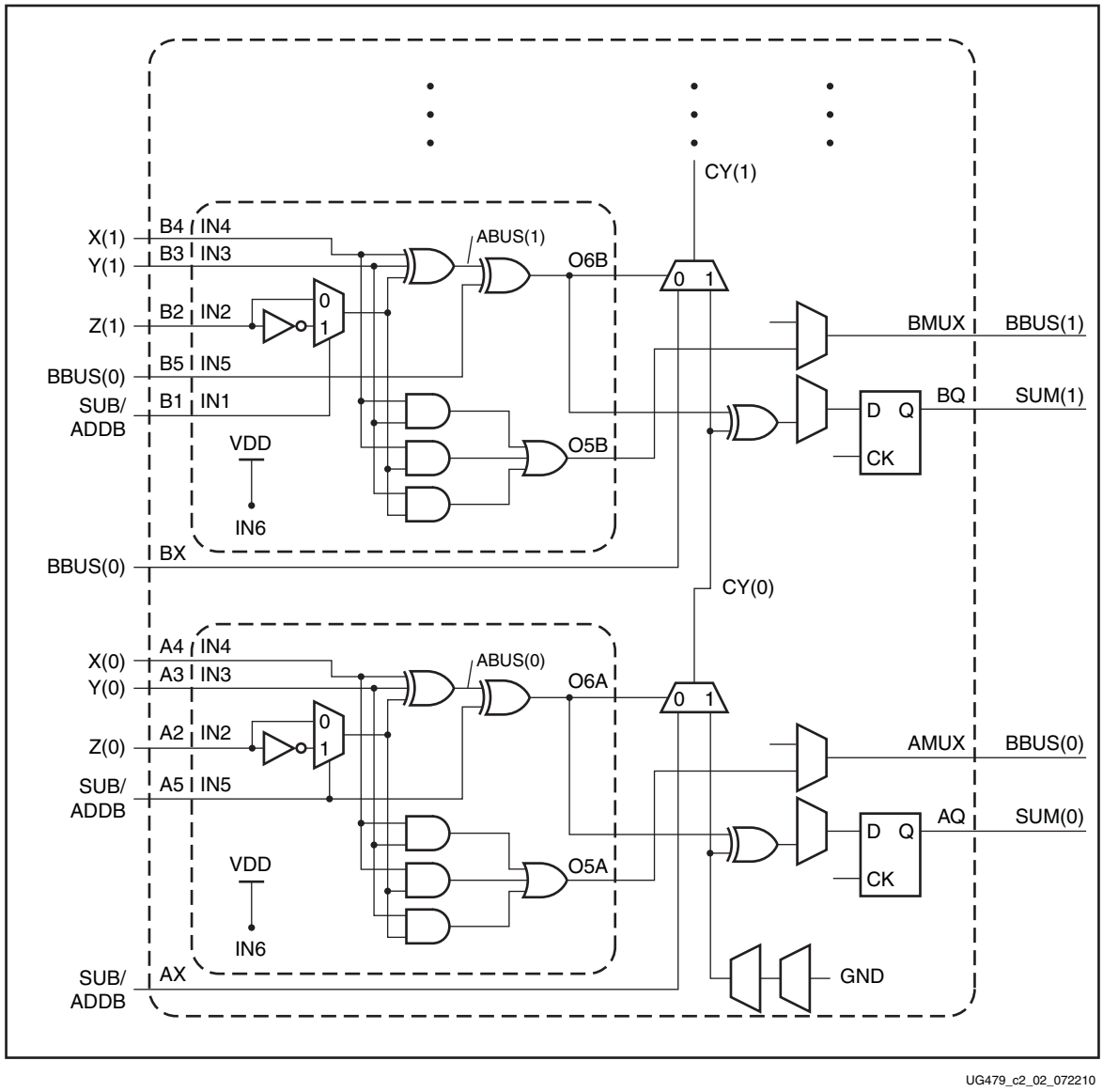


Figure 3-1: **Traditional FIR Filter Adder Tree**

In traditional FPGAs, the fabric adders are usually the performance bottleneck. The number of adders needed and the associated routing depends on the size of the filter. The depth of the adder tree scales as the $\log_2$ of the number of taps in the filter. Using the adder tree structure shown in Figure 3-1 could also increase the cost, logic resources, and power.

The 7 series FPGA CLB allows the use of both the 6LUT and the carry chain in the same slice to build an efficient ternary adder. The 6LUT in the CLB functions as a dual 5LUT. The 5LUT is used as a 3:2 compressor to add three input values to produce two output values. The 3:2 compressor is shown in Figure 3-2.



UG479_c2_02_072210

Figure 3-2: **Ternary Add/Sub with 3:2 Compressor**

The dual 5LUT configured as a 3:2 compressor in combination with the 2-input carry cascade adder adds three N-bit numbers to produce one N+2 bit output, as shown in the Figure 3-3, by vertically stacking the required number of slices.

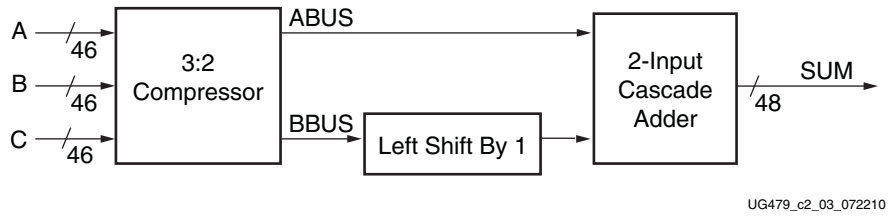


Figure 3-3: **Three-Input Adder**

The 3:1 adder shown in Figure 3-3 is used as a building block for larger adder trees. Depending on the number of inputs to be added, a 5:3 or a 6:3 compressor is also built in fabric logic using multiple 5LUTs or 6LUTs. The serial combination of 6:3 compressor, along with two DSP48E1 slices, adds six operands together to produce one output, as shown in Figure 3-4. The LSB bits of the first DSP48E1 slice that are left open due to left shift of the Y and Z buses should be tied to zero. The last DSP48E1 slice uses 2-deep A:B input registers to align (pipeline matching) the X bus to the output of the first DSP48E1 slice. Multiple levels of 6:3 compressors can be used to expand the number of input buses.

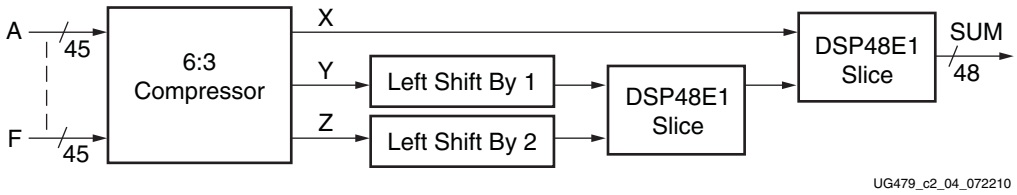


Figure 3-4: **Six-Input Adder**

The logic equations for the X, Y, and Z buses in Figure 3-4 are listed here:

$$X(n) = A(n)\ \text{XOR}\ B(n)\ \text{XOR}\ C(n)\ \text{XOR}\ D(n)\ \text{XOR}\ E(n)\ \text{XOR}\ F(n) \quad \textit{Equation 3-1}$$

$$\begin{aligned} Y(n) = {} & A(n)B(n)\ \text{XOR}\ A(n)C(n)\ \text{XOR}\ A(n)D(n)\ \text{XOR}\ A(n)E(n) \\ & \text{XOR}\ A(n)F(n)\ \text{XOR}\ B(n)C(n)\ \text{XOR}\ B(n)D(n)\ \text{XOR}\ B(n)E(n) \\ & \text{XOR}\ B(n)F(n)\ \text{XOR}\ C(n)D(n)\ \text{XOR}\ C(n)E(n)\ \text{XOR}\ C(n)F(n) \\ & \text{XOR}\ D(n)E(n)\ \text{XOR}\ D(n)F(n)\ \text{XOR}\ E(n)F(n) \end{aligned} \quad \textit{Equation 3-2}$$

$$\begin{aligned} Z(n) = {} & A(n)B(n)C(n)D(n)\ \text{OR}\ A(n)B(n)C(n)E(n)\ \text{OR}\ A(n)B(n)C(n)F(n) \\ & \text{OR}\ A(n)B(n)D(n)E(n)\ \text{OR}\ A(n)B(n)D(n)F(n)\ \text{OR}\ A(n)B(n)E(n)F(n) \\ & \text{OR}\ A(n)C(n)D(n)E(n)\ \text{OR}\ A(n)C(n)D(n)F(n)\ \text{OR}\ A(n)C(n)E(n)F(n) \\ & \text{OR}\ A(n)D(n)E(n)F(n)\ \text{OR}\ B(n)C(n)D(n)E(n)\ \text{OR}\ B(n)C(n)D(n)F(n) \\ & \text{OR}\ B(n)C(n)E(n)F(n)\ \text{OR}\ B(n)D(n)E(n)F(n)\ \text{OR}\ C(n)D(n)E(n)F(n) \end{aligned} \quad \textit{Equation 3-3}$$

The compressor elements and cascade adder can be arranged like a tree to build larger adders. The last add stage should be implemented in the DSP48E1 slice. Pipeline registers should be added as needed to meet timing requirements of the design. These adders can have higher area and/or power than the adder cascade.

Adder Cascade

The adder cascade implementation accomplishes the post addition process with minimal silicon resources by using the cascade path within the DSP48E1 slice. This involves computing the additive result incrementally, utilizing a cascaded approach as illustrated in Figure 3-5.

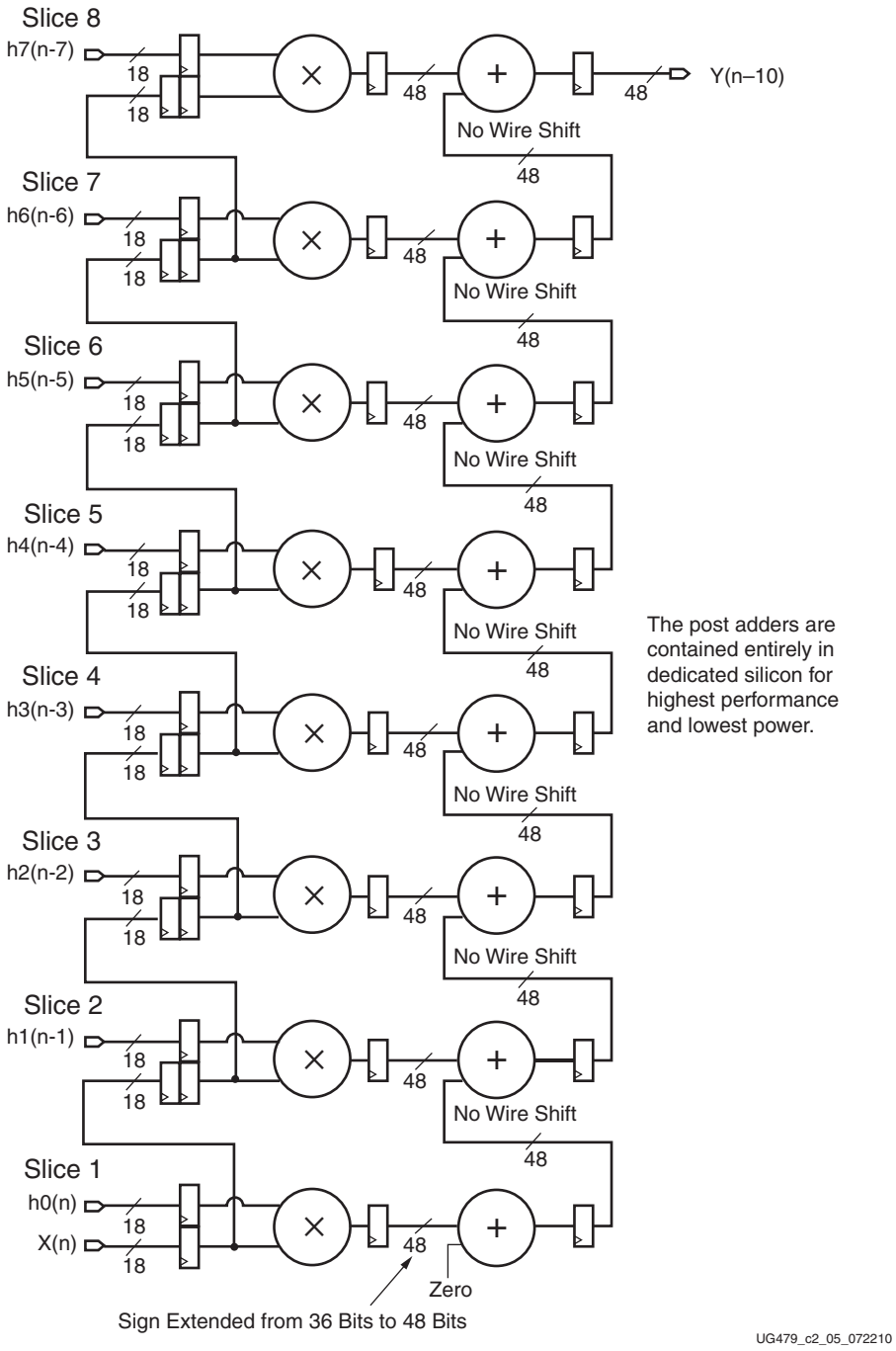


Figure 3-5: **Adder Cascade**

It is important to balance the delay of the input sample and the coefficients in the cascaded adder to achieve the correct results. The coefficients are staggered in time (wave coefficients).

Connecting DSP48E1 Slices across Columns

Using the cascade paths to implement adders significantly improves power consumption and speed. The maximum number of cascades in a path is limited only by the total number of DSP48E1 slices in one column in the chip.

The height of the DSP column can differ between the Virtex-5, Virtex-6, and 7 series devices and should be considered while porting designs between the devices. Spanning columns is possible by taking P bus output from the top of one DSP column and adding fabric pipeline registers to route this bus to the C port of the bottom DSP48E1 slice of the adjacent DSP column. Alignment of input operands is also necessary to span multiple DSP columns.

Time Multiplexing the DSP48E1 Slice

The high-speed math elements in the DSP48E1 slice enable designers to use time multiplexing in their DSP designs. Time multiplexing is the process of implementing more than one function within a single DSP48E1 slice at different instances of time. Time multiplexing can be done for designs with low sample rates. The calculation to determine the number of functions (N) that can be implemented in one single DSP48E1 slice is shown in Equation 3-4:

$$N * \text{channel frequency} \leq \text{maximum frequency of the DSP48E1 slice} \qquad \text{Equation 3-4}$$

These time-multiplexed DSP designs have optional pipelining that permits aggregate multichannel sample rates of up to 500 million samples per second. Implementing a time-multiplexed design using the DSP48E1 slice results in reduced resource utilization and reduced power.

The DSP48E1 slice contains the basic elements of classic FIR filters: a multiplier followed by an adder, delay or pipeline registers, and the ability to cascade an input stream (B bus) and an output stream (P bus) without exiting to a general slice fabric.

Multichannel filtering can be viewed as time-multiplexed, single-channel filters. In a typical multichannel filtering scenario, multiple input channels are filtered using a separate digital filter for each channel. Due to the high performance of the DSP48E1 slice within the 7 series device, a single digital filter can be used to filter all eight input channels by clocking the single filter with an 8x clock. This implementation uses 1/8th of the total FPGA resource as compared to implementing each channel separately.

Miscellaneous Notes and Suggestions

- Small multiplies (for example, 4 x 4 multiplies) and small bit width adders and counters should be implemented using the interconnect logic LUTs and carry chain. If you have a large number of small add operations and/or counters, you should take advantage of the SIMD mode and implement the operation in the DSP48E1 slice. Factor of 2x area and power savings occur, when compared to using interconnect logic, whenever input registers are also folded into the DSP48E1 slice for SIMD mode functions.
- Always sign extend the input operands when implementing smaller bit width functions. For lower fabric power, push operands into MSBs and ground (GND) LSBs.
- While cascading different DSP48E1 slices, the pipestages of the different signal paths should be matched.

- A count-up-by-one counter should be implemented within the DSP48E1 slice using the CARRYIN input. A count-by-N or variable-bit counter can use the C or A:B inputs.
- DSP48E1 counters can be used to implement control logic that runs at maximum speed.
- SRL16s/SRL32s in the CLB and block RAM should be used to store filter coefficients or act as a register file or memory elements in conjunction with the DSP48E1 slice. The bit pitch of the input bits (4 bits per interconnect) is designed to pitch match the CLB and block RAM.
- The block RAM can also be used as a fast, finite state machine to drive the control logic for the DSP design.
- The DSP48E1 slice can also be used in conjunction with a processor, for example, MicroBlaze™ or PicoBlaze™ processors, for hardware acceleration of processor functions.
- A pipeline register should be used at the output of an SRL16 or block RAM before connecting it to the input of the DSP48E1 slice. This ensures the best performance of input operands feeding the DSP48E1 slice.
- In 7 series devices, the register at the output of the SRL16 in the slice has a reset pin and a clock-enable pin. To reset the SRL16, a zero is input into the SRL16 for 16 clock cycles while holding the reset of the output register High. This capability is particularly useful in implementing filters where the SRL16s are used to store the data inputs.

DSP48E1 Design Resources

Refer to the *XST Hardware Description Language (HDL) Coding Techniques* chapter in the *XST User Guide for Virtex-6, Spartan-6, and 7 Series Devices* [Ref 3], which includes VHDL and Verilog inference coding templates for various DSP filters and DSP functions.

Pre-Adder Block Applications

DSP48E1 slice users can benefit from the pre-adder in several applications ranging from wireless applications (for example, in algorithms as in the Long-Term Evolution specification), in generic filtering (FIR and IIR), in video processing (for example, alpha blending), and many others.

Figure 3-6 illustrates how the pre-adder (shaded in gray) can be used in an 8-tap even symmetric systolic FIR design.

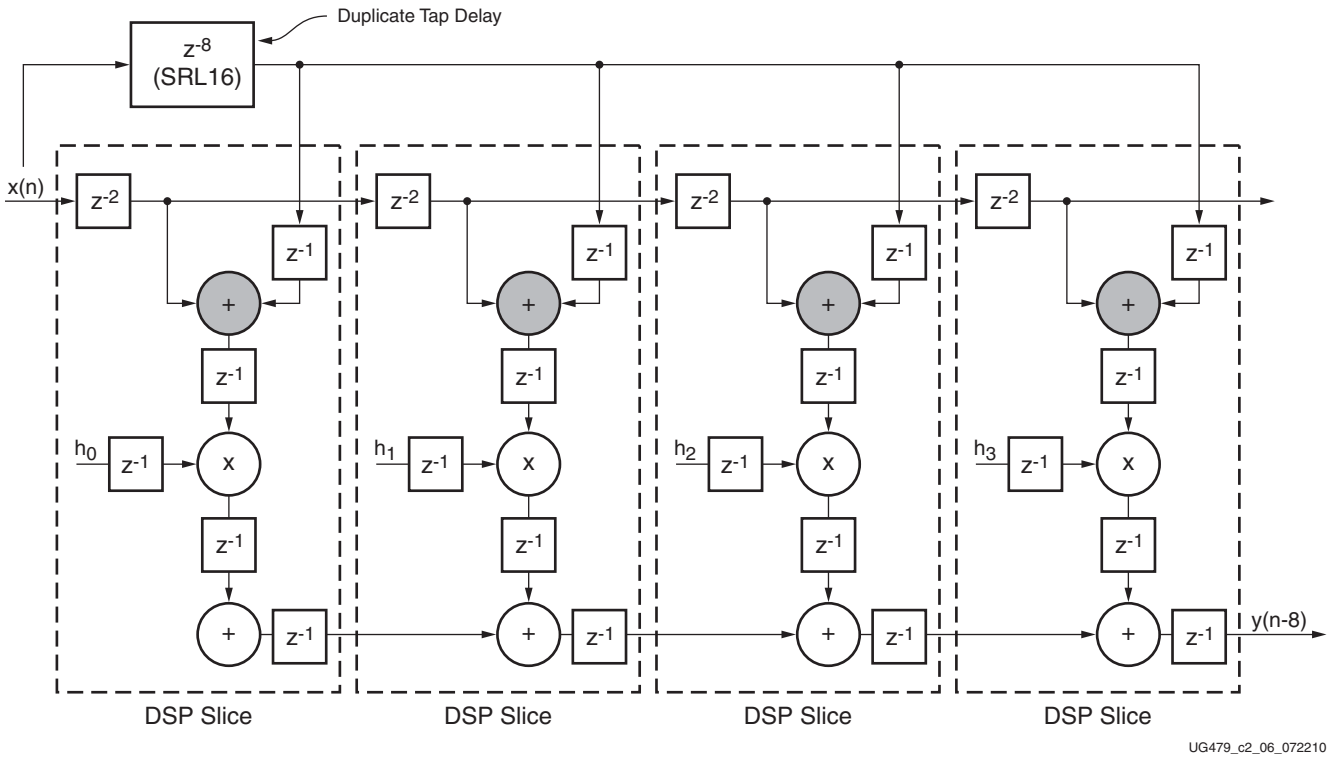


Figure 3-6: **8-Tap Even Symmetric Systolic FIR**

Memory-Mapped I/O Register Application

To use DSP48E1 slices as memory-mapped I/O registers, you must broadcast the write data bus feeding to all the DSP48E1 slices to be used in this fashion. To have random read access, a wide multiplexer is needed. Additional DSP48E1 slices can be configured as a wide bus multiplexer to help reduce routing congestion. An address decoder should be implemented in fabric logic to control individual PREG CEs to load the appropriate DSP output register from the write data bus.

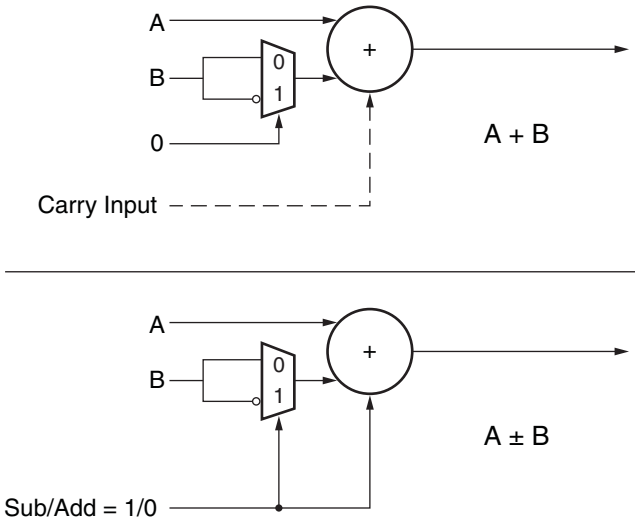
Appendix A

CARRYOUT, CARRYCASCOUT, and MULTSIGNOUT

CARRYOUT/CARRYCASCOUT

The DSP48E1 slice and the fabric carry chain use a different implementation style for subtract functions. The carry chain implementation in the CLB slices requires the fabric carry input pin to be connected to a constant `1` during subtract operations. The standard subtract operation with ALUMODE = `0011` in the DSP48E1 slice does not require the CARRYIN pin to be set to `1`.

In two's complement notation, negative values are obtained by performing a bitwise inversion and adding one, for example, –B = ((not B) + 1). The CLB implements A-B as (A + (not B) + 1), as shown in Figure A-1. An alternate implementation of a two-input subtracter is [not (B + (not A))], as shown in Figure A-2. The alternate implementation allows the carry inputs to the adder/subtracter to still be available for normal use.



(Carry input must be 1 for a subtract operation, so it is not available for other uses.)

UG369_A_01_111208

Figure A-1: **FPGA Logic Interconnect Based Adder/Subtracter**

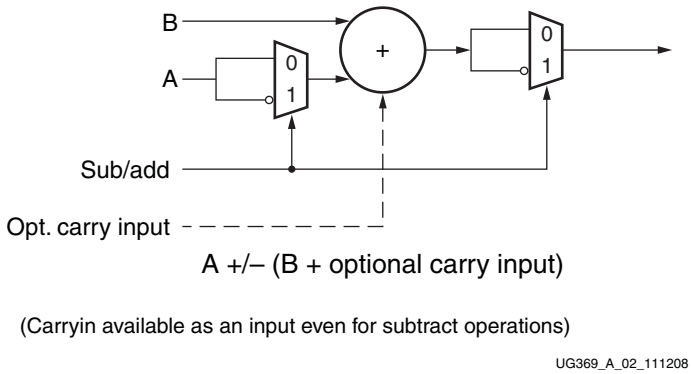


Figure A-2: **Adder/Subtracter with Optional Carry Inputs**

The DSP48E1 slice uses the second implementation extended to 3-input adders with a CARRYIN input as shown in Figure A-3. This allows DSP48E1 SIMD operations to perform subtract operations without a CARRYIN for each smaller add/subtract unit.

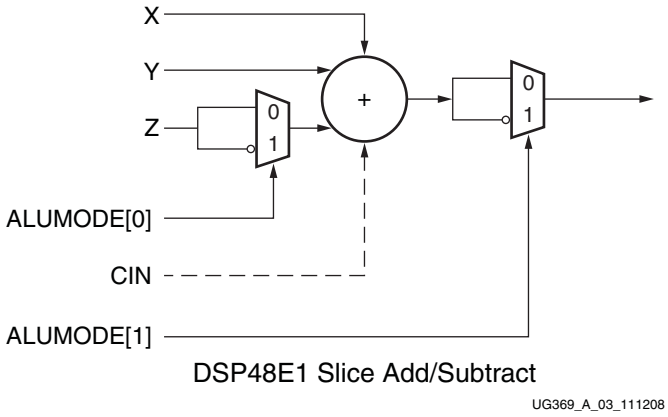


Figure A-3: **DSP48E1 Slice Three-Input Adder**

Virtex-6 and 7 series FPGA ALUMODE supports additional subtract operations:

ALUMODE = `0001` implements (-Z + (X + Y + CIN) – `1`).

- For most uses, CIN is set to `1` to cancel out the -`1`.

ALUMODE = `0010` actually implements -(Z + X + Y + CIN) – `1`.

- This inversion of the P output obtained by using ALUMODE `0010` can be cascaded to another slice to implement a two's complement subtract.

For add operations, CARRYOUT[3] and CARRYCASCOUT are identical, but the convention used to indicate a borrow (for subtract operations) is different. CARRYOUT[3] matches the convention for fabric subtractions. The matched convention allows a fabric add-sub function to use the CARRYOUT[3] pin directly to extend two-input DSP48E1 slice subtract operations in the fabric. The CARRYCASCOUT, however, follows the subtract convention of DSP slices and is, therefore, ideal for cascading up vertically to another DSP slice.

These CARRYOUT[3] and CARRYCASCOUT signals enable the construction of higher precision add/sub functions using multiple DSP48E1 slices or using a hybrid approach with both a DSP48E1 slice and a fabric adder/subtracter.

MULTSIGNOUT and CARRYCASCOUT

CARRYOUT[3] should not be used for multiply operations because the first stage multiplier generates two partial products, which are added together in the second stage of the DSP48E1 slice.

All DSP three-input add operations (including Multiply-Add and Multiply Accumulate) produce two CARRYOUT bits for retaining full precision. This is shown in Figure A-4.

MULTSIGNOUT and CARRYCASCOUT serve as the two carry bits for MACC_EXTEND operations. If MULTSIGNOUT is the multiplier sign bit and CARRYCASCOUT is the cascaded carryout bit, the result is the software/UNISIM model abstraction, shown in Figure A-4.

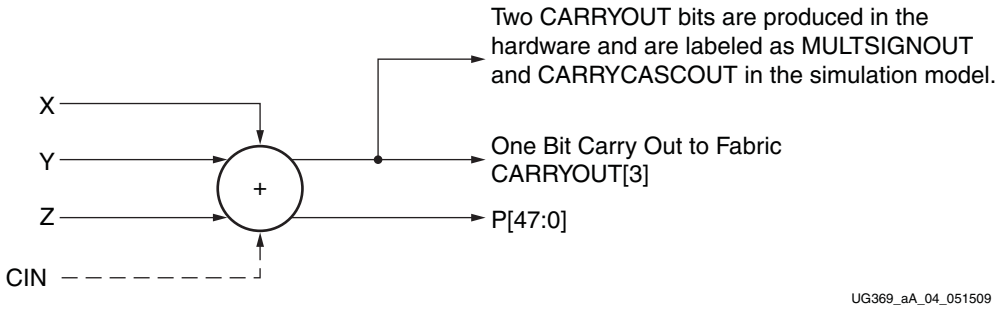


Figure A-4: **DSP48E1 Three-Input Adder**

MULTSIGNOUT and CARRYCASCOUT in the simulation model do not match the hardware, but the output P bits do match for supported functions, such as MACC_EXTEND. For example, routing CARRYCASCOUT to the FPGA logic by using all zeros in the upper DSP slice does not match the CARRYOUT[3] of the lower DSP slice. Similarly, MULTSIGNOUT routed out to fabric is not actually the sign of the multiply result.

These MULTSIGNOUT and CARRYCASCOUT signals enable the construction of higher precision multiply-accumulate (MACC_EXTEND) functions, such as a 25x18 multiply accumulated for up to 96 bits of accumulator precision and running at the maximum DSP48E1 slice frequency.

It is also necessary to set the OPMODEREG and CARRYINSELREG to 1 when building large accumulators such as the 96-bit Multiply Accumulate. This prevents the simulation model from propagating unknowns to the upper DSP48E1 slice when a reset occurs.

Summary

Adder/Subtracter-only Operation

CARRYOUT[3]: Hardware and software match.

CARRYCASCOUT: Hardware and software match when ALUMODE = `0000` and inverted when ALUMODE = `0011`. The mismatch happens because the DSP48E1 slice performs the subtract operation using a different algorithm from the FPGA logic; thus, the DSP48E1 slice requires an inverted CARRYOUT from the FPGA logic.

MULTSIGNOUT is invalid in adder-only operation.

MACC Operation

CARRYOUT[3] is invalid in the MACC operation.

CARRYCASCOUT and MULTSIGNOUT: Hardware and software do not match due to modeling difference. The software simulation model is an abstraction of the hardware model. The software views of CARRYCASCOUT and MULTSIGNOUT enable higher-precision MACC functions to be built in the UNISIM model. They are not logically equivalent to hardware CARRYCASCOUT and MULTSIGNOUT. Only the hardware and software results (P output) are logically equivalent. The internal signals (CARRYCASCOUT and MULTSIGNOUT) are not. See Figure A-5.

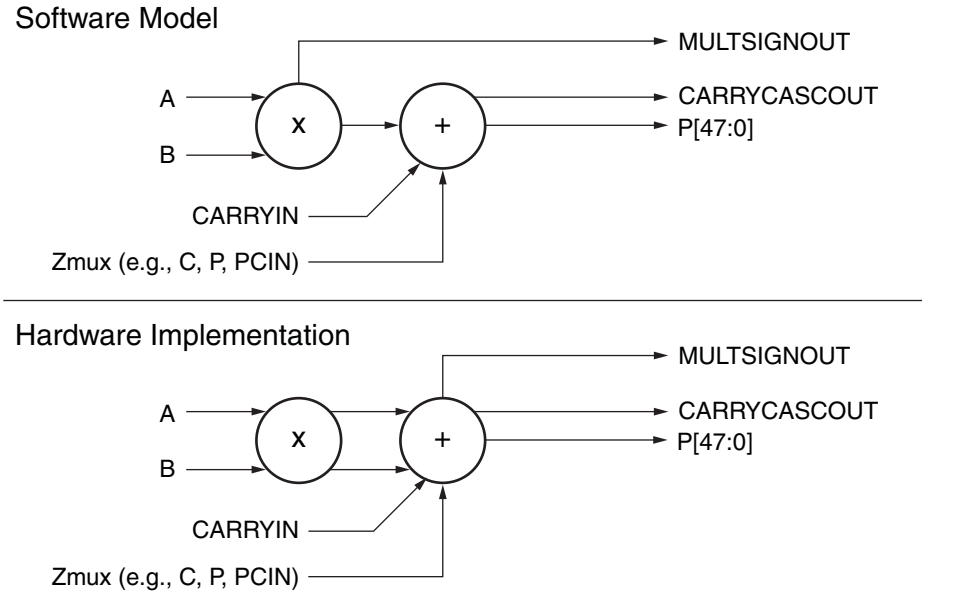


Figure A-5: **MACC Software and Hardware Models**