Complex Multiplier/Mixer Megafunction

Target Applications:

Communications
Digital Signal Processing

Family: FLEX 10K and FLEX 8000

Vendor:

NOVA
E N G I N E E R I N G

Nova Engineering, Inc.
5 Circle Freeway Drive
Cincinnati, OH 45246
Tel.   (513) 860-3456
Fax   (513) 860-3535
E-mail   sales@nova-eng.com
WWW   http://www.nova-eng.com

Features

- Parallel implementation for maximum speed
- Parameterized data width
- Optimized for the Altera® FLEX® 10K and FLEX 8000 device architectures
- Ideal for the following applications:
    - Digital mixers
    - Discrete Fourier transforms
    - Modulators/demodulators
    - Complex arithmetic
    - Wireless telecommunication systems

General Description

The complex multiplier/mixer megafunction from Nova Engineering can multiply two complex numbers or mix two complex signals. This function can be used for vector cross products, vector dot products, up/down frequency conversion, differential phase detection, digital amplitude modulation (AM), and quadrature amplitude modulation (QAM). Nova Engineering will customize the input width, output width, and processing latency of the complex multiplier/mixer megafunction at no additional cost. See Figure 1.

Figure 1. Complex Multiplier/Mixer Megafunction Block Diagram
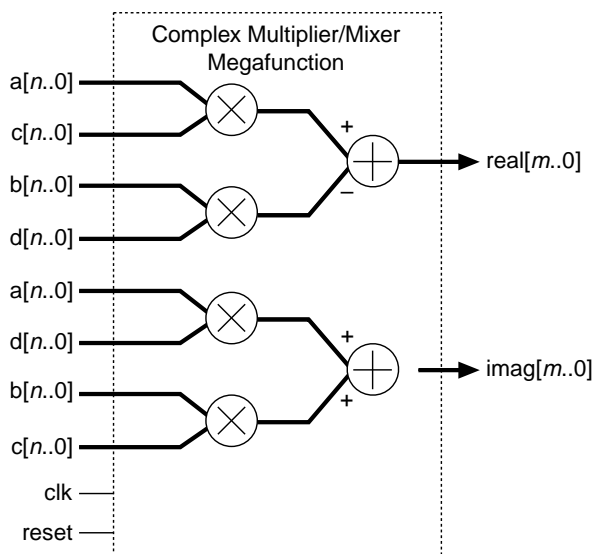
Functional Description

The complex multiplier/mixer megafunction is a parallel and pipelined architecture that provides maximum speed. The function's internal and external operations are synchronized to the rising clock edge, and an asynchronous `reset` input is provided for initializing all internal registers.

The complex multiplier/mixer megafunction computes the following function:

$$\texttt{real[}m..0\texttt{]} + j\texttt{imag[}m..0\texttt{]} = (\texttt{a[}n..0\texttt{]} + j\texttt{b[}n..0\texttt{]}) \times (\texttt{c[}n..0\texttt{]} + j\texttt{d[}n..0\texttt{]})$$

where: $\texttt{real[}m..0\texttt{]} = (\texttt{a[}n..0\texttt{]} \times \texttt{c[}n..0\texttt{]}) - (\texttt{b[}n..0\texttt{]} \times \texttt{d[}n..0\texttt{]})$
$\texttt{imag[}m..0\texttt{]} = (\texttt{a[}n..0\texttt{]} \times \texttt{d[}n..0\texttt{]}) + (\texttt{b[}n..0\texttt{]} \times \texttt{c[}n..0\texttt{]})$

Because $\texttt{b[}n..0\texttt{]}$ and $\texttt{d[}n..0\texttt{]}$ are both imaginary (i.e., preceded by $j = \sqrt{(-1)}$), the $\texttt{real[}m..0\texttt{]}$ output is equal to $(\texttt{a[}n..0\texttt{]} \times \texttt{c[}n..0\texttt{]}) - (\texttt{b[}n..0\texttt{]} \times \texttt{d[}n..0\texttt{]})$. The $\texttt{imag[}m..0\texttt{]}$ output is implicitly imaginary (i.e., preceded by j).

Ports

This function has input ports for two complex numbers or data streams. Table 1 describes the ports for the complex multiplier/mixer megafunction.

Table 1. Complex Multiplier/Mixer Megafunction Ports

Name	Type	Size	Description
a[n..0]	Input	4 to 12 bits	Real part of complex vector 1. Two's complement format.
b[n..0]	Input	4 to 12 bits	Imaginary part of complex vector 1. Two's complement format.
c[n..0]	Input	4 to 12 bits	Real part of complex vector 2. Two's complement format.
d[n..0]	Input	4 to 12 bits	Imaginary part of complex vector 2. Two's complement format.
clk	Input	–	System clock.
reset	Input	–	Asynchronous reset, active high.
real[m..0]	Output	4 to 25 bits	Real part of complex result. Two's complement format.
imag[m..0]	Output	4 to 25 bits	Imaginary part of complex result. Two's complement format.

Parameters

Nova Engineering can customize the width of the input and output data parameters to eliminate unnecessary logic and precision, and to optimize speed and area. In addition, the multiplier latency parameter can be increased to improve execution speed or reduced to conserve logic cells. See Table 2. The total latency from input to output is equal to the specified multiplier latency plus one. The output of the adders is latched to increase the speed without increasing the logic used.

Table 2. Complex Multiplier/Mixer Megafunction Parameters

Parameter	Typical Value	Description
Input data width	4 to 12 bits	Specifies the width of inputs a[n..0], b[n..0], c[n..0], and d[n..0].
Output data width	4 to 25 bits	Specifies the width of the output data. Truncation retains the most significant bits (MSBs) of the result and discards the least significant bits (LSBs).
Multiplier latency	0 to 4 clocks	Specifies the number of pipeline registers added to the design. Pipelining increases the operating speed at the expense of additional logic cells.

Performance

The megafunction does not require embedded array blocks (EABs), and thus performs efficiently in both FLEX 10K and FLEX 8000 device families. Because this function does not use EABs, it can be combined with a sine/cosine look-up table (LUT) for frequency down conversion or discrete Fourier transform (DFT) functions within a single device.

Typically, the complex multiplier/mixer megafunction is used as a coprocessor to a digital signal processing (DSP) function. The efficiency of the DSP processor design can be improved by implementing both the megafunction and a FIFO function into the same device. For example, the input/output process loading on the DSP processor can be reduced if the megafunction's inputs and outputs are accessed as blocks of data through a FIFO function.

Table 3 shows the device utilization and maximum clock frequency for a typical implementation of the complex multiplier/mixer megafunction.

Table 3. Typical Device Utilization			
Implementation	**Clock (Max)**	**Logic Cells**	**EABs**
Input data width = 8 bits Multiplier output width = 16 bits Output data width = 8 bits Multiplier latency = 3 clocks	50 MHz	600	0

Contact Nova Engineering for more information on the complex multiplier/mixer megafunction.

Applications

The complex multiplier/mixer megafunction is useful for a variety of applications, including vector arithmetic, digital AM and QAM, and frequency down conversion.

Vector Arithmetic

The complex multiplier/mixer megafunction can calculate vector cross products and vector dot products. For example, to calculate the vector cross product of vectors $\mathbf{R} = r\mathbf{x} + r\mathbf{y}$ and $\mathbf{S} = s\mathbf{x} + s\mathbf{y}$, each variable is assigned to an input of the megafunction (i.e., a[n..0] = $r\mathbf{x}$, c[n..0] = $s\mathbf{y}$, b[n..0] = $r\mathbf{x}$, and d[n..0] = $s\mathbf{y}$). The megafunction performs the calculation and provides the result on the real[m..0] output.

A vector dot product can be calculated in a similar manner. To calculate the vector dot product of vectors $\mathbf{R} = r\mathbf{x} + r\mathbf{y}$ and $\mathbf{S} = s\mathbf{x} + s\mathbf{y}$, each variable is assigned to an input of the megafunction (i.e., a[n..0] = $r\mathbf{x}$, c[n..0] = $s\mathbf{y}$, b[n..0] = $r\mathbf{y}$, and d[n..0] = $s\mathbf{x}$). The complex multiplier/mixer megafunction performs the calculation and provides the result on the imag[m..0] output.

Digital AM & QAM

Amplitude modulation (AM) and quadrature amplitude modulation (QAM) require the source data to vary the amplitude of a carrier frequency over time. For example, to create an AM wave with the complex multiplier/mixer megafunction, assign the carrier frequency to the c[n..0] input and the information signal to the a[n..0] input. Set the b[n..0] and d[n..0] inputs to zero. The megafunction produces an AM wave on the real[m..0] output.

Frequency Down Conversion

Communications systems often transmit and receive data at frequencies significantly faster than the data rate of the information. The megafunction can recover this information or convert it to a lower frequency by mixing a modulated signal with a carrier frequency. For example, to mix a modulated signal with a carrier frequency generated from a numerically controlled oscillator (NCO), apply the NCO outputs to the c[n..0] and d[n..0] inputs of the megafunction. Apply the received signal to the a[n..0] input.

For example, if y(t) = s(t) × cos(wt) is the received signal, then mixing with the carrier frequency cos(wt) produces:

z(t) = s(t) × cos(wt) × cos(wt)

Where: s(t) = Information signal

Using the trigonometric identity cos(a) × cos(b) = (1/2) × cos($a - b$) + (1/2) × cos($a + b$) and substituting cos(a) = cos(b) = cos(wt) produces the following:

z(t) = s(t) × (1/2) × [cos($a - b$) + cos($a + b$)]

Low-pass filtering can be accomplished via a low-pass, finite impulse response (FIR) filter, effectively discarding the cos($a + b$) element because it is twice the carrier frequency. Because $a = b$, the following equation results:

z(t) = (1/2) × s(t)

The carrier frequency is removed to show the information signal.

Altera Corporation