Introduction

The Stratix® III family of devices have dedicated high-performance digital signal processing (DSP) blocks optimized for DSP applications. These DSP blocks of the Altera® Stratix device family are the third generation of hardwired, fixed function silicon blocks dedicated to maximizing signal processing capability, ease of use, and lowest silicon cost.

Stratix III DSP Blocks Architecture Overview

The Stratix III DSP block is a superset of previous generations of embedded multipliers and offers significant improvements. Some of the key features include the following benefits:

- Native support for 9-, 12-, 18-, and 36-bit multiplication
- Signed and unsigned input support
- Built-in addition, subtraction, and accumulation units
- Cascaded 18-bit input bus for filter tap-delay line
- Included chain out adder for efficient cascaded adder structure implementation
- Built-in rounding and saturation units
- Efficient barrel shifter implementation
- Loopback capability to support adaptive filtering

For more information, refer to the *DSP Blocks in Stratix III Devices* chapter in volume 1 of the *Stratix III Device Handbook*.

Resource Distribution

The Stratix III device family has two variants:

- Logic devices, which focus on logic rich applications
- Enhanced devices, which focus on digital signal processing and memory intensive applications

Each DSP block in Stratix III occupies four Logic Array Blocks (LABs) in height and can be divided further into two half-blocks that share some common clock signals.
In most circumstances, these half-blocks are identical in functionality. Each half DSP block can be used to implement four 9 x 9 multipliers, three 12 x 12 multipliers, two 18 x 18 multipliers, or one 36 x 36 multiplier. In multiply and add mode, each half DSP block can implement four multipliers with input bit-width up to 18 bits and sum the results.

For detailed information about DSP block resource distribution across different devices, refer to the *DSP Blocks in Stratix III Devices* chapter in volume 1 of the *Stratix III Device Handbook*.

**Megafunction Overview**

To utilize the DSP block efficiently in custom designs, Altera provides a number of megafunctions in the Quartus® II software to facilitate the design process.

**Accessing the DSP Block Using the Megafunction MegaWizard**

The easiest way to access the DSP block resource in a Stratix III device is to use the MegaWizard® Plug-In Manager. Four megafunctions access the DSP blocks directly:

- LPM_MULT, a single multiplier function
- ALTMULT_ADD, a multiply and add function
- ALTMULT_ACCUM, a multiply and accumulate function
- ALTMULT_COMPLEX, a complex multiplication function

*Figure 1* shows how you can select each of these megafunctions while using the MegaWizard Plug-In Manager.
For more information about these megafuctions, including design examples and detailed descriptions, refer to the following User Guides:

- **lpm_mult Megafuction User Guide**
- **altmult_add Megafuction User Guide**
- **altmult_accum Megafuction User Guide**
- **altfp_mult Megafuction User Guide**
Optimizing for Performance Using Megafunctions

These megafunctions offer a flexible way to instantiate an arithmetic function block for multiplication. This level of flexibility is beneficial for shortening the development cycle. However, to achieve the highest performance, the designer must understand the limitation of the megafunctions and work within these limits. This section explains how to maximize DSP performance using the megafunctions.

**LPM_MULT Megafunction**

The LPM_MULT megafunction is the simplest of the megafunction multipliers. It is limited to implementing a single multiplier, with only a subset of control signals. To achieve the highest $f_{\text{MAX}}$ possible, restrict the input bus width to 36 bits or fewer, and a pipeline latency of at least 2. For multipliers with a larger bus width, the ALTMULT_ADD megafunction or a custom implementation is better suited.

For more information about the LPM_MULT megafuction, including design examples and detailed descriptions, refer to the lpm_mult Megafunction User Guide.

**ALTMULT_ADD Megafunction**

The ALTMULT_ADD megafunction is a flexible unit responsible for instantiating multiply and add modules. It is also capable of instantiating a single multiplier and is a superset of LPM_MULT functions. It also includes a resource usage estimator in the MegaWizard Plug-In Manager.

The following guidelines must be taken into consideration:

- In the single multiplier mode, if high performance is the top priority of the design, the input bus width must be restricted to 54 bits or fewer.
- For more than one multiply and add instances, high performance can be achieved if the input bus width for all multipliers is restricted to 18 bits or fewer.
- If larger multiply and add units are required, Altera recommends implementing each multiplier as a single multiplier and custom designing a pipelined adder for the summation unit.

If the only resource required to implement the desired function is DSP blocks, the instance is likely to achieve the highest possible performance. If the soft logic resources, such as ALUTs, are used in addition to DSP blocks, further optimization may be possible. The exception to this rule is
when a single multiplier has a bus width between 37 and 54 bits. The megafun-
c tion is optimized for this mode to ensure high level of performance.

For more information about the ALTMULT_ADD megafunction, includ-
ing design examples and detailed descriptions, refer to the
altmult_add Megafunction User Guide.

ALTMULT_ACCUM Megafunction

The accumulator unit in the DSP block can be accessed via the
ALTMULT_ACCUM megafunction. For a high-performance design, the
input bus width for all multipliers must be 18 bits or fewer.

For more information about the ALTMULT_ACCUM megafunction, includ-
ing design examples and detailed descriptions, refer to the
altmult_accum Megafunction User Guide.

ALTMULT_COMPLEX Megafunction

A quick and easy way to instantiate a complex multiplier is to use the
ALTMULT_COMPLEX megafunction. To maintain the highest level of
performance while using this megafunction, the input bus width must be
18 bits or fewer. For bit-widths greater than 18 bits, implementing each of
the multipliers separately and combining them with custom pipelined
adders may produce better performance.

For more information about the ALTMULT_COMPLEX megafunction, refer to the Quartus II Help.

ALTFP_MULT Megafunction

The ALTFP_MULT megafunction is a floating point multiplication unit
that makes use of DSP blocks in Stratix III devices. It requires a single
half-DSP block to implement a single precision floating point
multiplication function; it requires two-and-a-half DSP blocks to
implement a double precision floating point multiplication.

For more information about the ALTFP_MULT megafunction, including
design examples and detailed descriptions, refer to the altfp_mult
Megafunction User Guide.
**Design Examples**

Altera has constructed design examples to demonstrate how Stratix III DSP blocks can be used efficiently in DSP-related applications. In this section, four design examples are described, showcasing different features and design considerations.

**Direct-Form Parallel FIR**

A finite impulse response (FIR) filter is one of the most popular filters in digital signal processing, because of its simple structure, high performance, and ease of implementation in most technologies. Equation 1 shows the basic equation for the FIR filter:

\[
y_n = \sum_{i=0}^{N-1} x_{n-i}c_i
\]

A direct-form parallel FIR filter means that the filter is implemented to match the equation, and every coefficient has a dedicated multiplier. Therefore, the entire equation can be evaluated in a single clock cycle. Properly pipelined, a direct-form parallel FIR filter can stream output samples continuously after a predetermined latency, as long as there is input data, to ensure a high level of performance. The structure of a four-tapped, direct-form parallel FIR filter is shown in Figure 2.

![Figure 2. Direct Form FIR Filter Structure](image)

The adder tree structure in the FIR filter can be implemented in the FPGA using soft logic. However, using soft logic introduces an additional resource overhead, and could result in reduced system performance.

The DSP block in the Stratix III device accomplishes the post-addition in two steps.
Each half DSP block can implement four 18-bit multipliers and have a built-in adder tree to combine the results. For designs that span multiple DSP blocks, a dedicated chain out adder at the last stage of each half DSP block can be used to sum the results, without wasting silicon resource or sacrificing performance.

The chain out adder can be used only in the four multiplier add modes or the multiply and accumulate modes of the Stratix III DSP block, and the input width of the multipliers must be 18 bits or fewer.

To implement the filter shown in Figure 2 efficiently using the chain out adder feature, divide the multipliers into groups of four and contain each group in a single half DSP block. The results from each half DSP block can then be added together using the chain out adder.

The input tap-delay pipeline in the direct form parallel FIR filter can also be optimized in a Stratix III DSP block implementation. Each input of the multiplier can be registered, and the input registers can be linked together to form a shift-register chain for the delay pipeline. This tap delay line can also propagate from DSP block to DSP block, which makes the system easily scalable.

Figure 3 shows a Stratix III DSP block implementation of an $n$ tap direct form parallel FIR filter with 18-bit input and coefficient quantization. If $n$ is not a multiple of four, the final half DSP block is used incompletely.

**Figure 3. Direct-Form Parallel FIR Filter Using DSP Blocks**
Configure Chain Out Adder

To implement this filter in a Stratix III device, use the ALTMULT_ADD megafunction to instantiate each DSP block. Arrange the multipliers in groups of four and ensure that their inputs are restricted to 18 bits or fewer. Figure 4 shows the General tab, and indicates how to configure the first multiplier when using the MegaWizard Plug-In Manager.

The chain out adder of the first four-multiplier section must be enabled to achieve the correct latency of the output, but the input port to the chain out adder must be disabled. This is because the chain out adder can only take its input from the output of a DSP block, and cannot have any soft logic in between or connected to a constant, such as zero. For the rest of
the DSP block stages, the chain in port must be enabled. This configuration can be made on the Chainout/Shift/Loopback tab of the MegaWizard Plug-In Manager, shown in Figure 5.

Figure 5. Chain Out Configuration
**Configure Input Tap Delay**

To take advantage of the input shift register feature of the Stratix III DSP block, one set of the input registers must be configured to take input from previous multiplier inputs. This can be performed on the Multipliers tab of the MegaWizard Plug-In Manager, shown in Figure 6.

*Figure 6. Input Tap Delay Configuration*
Note the shiftout A port must be enabled if the current DSP block is not the last one in the filter, and an extra register must be added to account for the extra latency if the chain out adder output is registered. This can be done in the **Extra Modes** tab of the MegaWizard Plug-In Manager, shown in Figure 7.

**Figure 7. Shift Out Configuration**

![Shift Out Configuration](image)

**Configure Rounding and Saturation**

The internal word length for each chain out adder is 44 bits, more than sufficient for most FIR implementations. However, the signal processing stage after the FIR may not be able to handle such high precision. The simple solution is to truncate the result to the desired word length, but that may introduce asymmetrical inaccuracies into the result.
The best way to resolve this issue is to round and saturate the result to the desired word length. To facilitate this operation, the Stratix III DSP block is equipped with dedicated rounding and saturation units at the second-stage adder and the chain out adder.

To enable this feature, on the General tab (see Figure 4 on page 8), select Add support for hardware saturation and rounding. Then, on the Saturation tab, select the desired saturation bits, as shown in Figure 8. (If the number of saturation bits you require doesn’t appear in the drop-down list, you can enter the number manually.) The saturation fractional width indicates how many bits from the MSB will be saturated to sign bits. You can also select whether to employ symmetrical or asymmetrical saturation.

In the Rounding tab, Rounding fractional width represents the number of bits that remain valid starting from the MSB. The parameter can be adjusted as shown in Figure 9. The rounded-off bits remain unchanged and do not become zeros.
Implementation Results

The Stratix III DSP blocks are well-suited for FIR filter implementation. The hardware resource usage and $f_{\text{MAX}}$ of a 16-tap, direct form FIR example is shown in Table 1. An example implementation of a 5-channel, 16-tap FIR can be downloaded from the Altera website, and the implementation results are shown in Table 2 on page 16. Note that this information is specific to the Quartus II software version 7.2, using a Stratix III device with a speed grade of –2. Both data and coefficient widths are 18 bits for this example.

<table>
<thead>
<tr>
<th>Table 1. Direct Form Parallel FIR Filter Resource Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Half DSP Blocks</strong></td>
</tr>
<tr>
<td>----------------------</td>
</tr>
<tr>
<td>16-tap FIR</td>
</tr>
</tbody>
</table>
Multi-Channel Direct Form FIR

Just as an FPGA offers significant performance advantage over a DSP processor, a FIR filter implemented in an FPGA often greatly exceeds the throughput requirement of a single channel. Therefore, sharing a single filter between multiple channels is an efficient use of resources. This can be accomplished by increasing the delay cycles between each multiplier tap and sending different channels in an interleaved format.

The MLAB memory, as part of the Stratix III device family, is a memory structure constructed out of ALUTs and distributed throughout the device. These MLAB memory units greatly facilitate the implementation of the tap delay elements. Figure 10 demonstrates this concept.

For more information, refer to the TriMatrix Embedded Memory Blocks in Stratix III Devices chapter in volume 1 of the Stratix III Device Handbook.

---

**Figure 10. Multi-Channel FIR Filter Note (1)**

In the Stratix III family of FPGAs, half of the LABs serve two purposes: either as soft logic or as memory blocks. Each MLAB memory block contains 640 bits of simple dual-port SRAM, making it a good choice for shift registers and small FIFOs. An implementation of a 16-tap multi-channel, direct-form FIR filter is shown in Figure 11.
Figure 11. Multi-Channel FIR Implementation Using MLABs

Configure the MLAB Memory

To implement this filter, the MLAB memory modules must be configured as shift registers, and the distance between taps must match the number of channels. You have options to create a multiple-taps shift register, one tap for each coefficient. This can be done using the SHIFT_REG megafuction, as shown in Figure 12.
Implementation Results

The dedicated DSP blocks and the distributed MLAB memory structure make Stratix III devices an ideal choice for multi-channel FIR filters. An example implementation of a 5-channel, 16-tap FIR can be downloaded from the Altera website. The implementation results are shown in Table 2. Note that this information is specific to the Quartus II software version 7.2, using a Stratix III device with a speed grade of –2. Both data and coefficient widths are 18 bits for this example.

| Table 2. Multi-Channel Direct Form Parallel FIR Filter Resource Usage |
|-------------------------|---------------------|-----------------|----------------|----------------|
|                         | MLABs | Half DSP Blocks | ALUTs | Registers | f_MAX (MHz) |
| 5-channel, 16-tap FIR    | 15    | 4              | 170   | 870        | 489.95       |
Multiply-Accumulate FIR

A fully parallel implementation of the direct-form FIR filter often far exceeds the performance requirement of the given application. Therefore, you may choose to reduce the logic resource usage at the expense of performance.

Using the accumulator unit in the Stratix III DSP block, you can build a multi-tap multiply-accumulate (MAC) FIR filter. This filter takes multiple clock cycles to compute an output, sharing the available resources.

Each half DSP block is equipped with one accumulator unit. If that accumulator block is used, no other resource in that particular half DSP block can be routed to the output. Therefore, it is resource-efficient to utilize all available resources in the half DSP block before sending the result to the accumulator.

If the input and coefficient word length is restricted to 18 bits or fewer, each half DSP block can implement four multipliers with an adder tree to sum the results. The accumulator unit can then accumulate the result of the four multiplier adders to finish the post-addition operations required by the filter algorithm. By dividing the number of taps into groups of four and restricting the input width to 18 bits or fewer, all available resources can be used efficiently. A sample multi-tap MAC FIR is shown in Figure 13.
Configure the MLAB Memory

Stratix III devices offer a TriMatrix™ memory structure for efficient and flexible implementation of the data and coefficient memory, depending on the number of taps. In most cases, the small and distributed nature of MLAB memory cells is ideally suited for this purpose.

To configure an MLAB as a simple dual-port RAM, from page 3 of the MegaWizard Plug-In Manager (the Memory Compiler section), select **RAM: 2-PORT**. On the same page, select the **With one read port and one write port** option because independent read and write addresses are required. These selections are shown in Figure 14.
For an $N$-tap filter with $M$ multipliers, it takes $N/M$ clock cycles to calculate each output. The appropriate sample shifted value must be presented to the multipliers at each clock cycle, and the accumulator must be reset after each output is produced. A controller should be designed to control the read and write address of the memory as well as loading the output register.
Implementation Results

With the high $f_{\text{MAX}}$ of Stratix III devices, even a single half DSP block can achieve the performance required by many applications. An example implementation of a 100-tap, single-channel MAC FIR filter with four 18-bit multipliers can be downloaded from the Altera website. Because it takes 25 clock cycles to produce each output, the throughput is $1/25$ of the $f_{\text{MAX}}$. The resource usage and performance results are shown in Table 3. This information is specific to the Quartus II software version 7.2, using a Stratix III device with a speed grade of –2 and that both data and coefficient widths are 18 bits for this example.

<table>
<thead>
<tr>
<th>Table 3. MAC FIR Filter Resource Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLABs</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>100-tap MAC FIR</td>
</tr>
</tbody>
</table>

Large Multiply-Adder

Multiply and add units can be implemented efficiently in the Stratix III DSP blocks if they can be contained within a single half DSP block, or constructed with multiple DSP blocks with their chain out adders. However, if the input word length is greater than 18 bits, the internal adders are used to implement the multiplier, and the chain out adders can no longer be used. In this situation, the ALTMULT_ADD megafunction may not provide the most efficient implementation of the adder tree structure, and custom designed pipelined adders may produce better results. With appropriate pipeline stages, it is possible to achieve a system level $f_{\text{MAX}}$ equivalent to the $f_{\text{MAX}}$ of the DSP blocks. A performance-optimized implementation of four $32 \times 32$ multiply and add units is shown in Figure 15.
The carry chain is often the critical path in a large word-length adder. However, if throughput is more important than latency, which often is the case in non-recursive signal processing systems, the carry chain can be pipelined to reduce the signal propagation delay from register to register. The LPM_ADD_SUB megafunction offers a simple way to insert pipeline stages into an adder. To create a 64-bit adder with a four-stage pipeline, first specify the input bus width to be 64-bit, and then specify the latency to be four clock cycles in the pipelining tap. This is demonstrated in Figures 16 and 17.

For more information about the LPM_ADD_SUB megafuction, refer to the lpm_add_sub Megafuction User Guide.
Figure 16. LPM_ADD_SUB Megafunction, Page 1 of 6

Figure 17. LPM_ADD_SUB Megafunction, Page 4 of 6
Implementation Results

With proper pipelining, it is possible to create large multiply and add units that can operate at a very high \( f_{\text{MAX}} \). An example of a four 32-bits multiply and add unit can be downloaded from the Altera website. Its resource utilization and performance is shown in Table 4. This information is specific to the Quartus II software version 7.2, using a Stratix III device with a speed grade of –2.

<table>
<thead>
<tr>
<th></th>
<th>Half DSP Blocks</th>
<th>ALUTs</th>
<th>Registers</th>
<th>( f_{\text{MAX}} ) (Mhz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 32 x 32-bit Multiply-Add</td>
<td>4</td>
<td>804</td>
<td>1106</td>
<td>440.12</td>
</tr>
</tbody>
</table>

Conclusion

The DSP block in the Stratix III DSP block offers numerous features to facilitate the implementation of a digital signal processing system. When used correctly, they can achieve a high performance that is capable of handling multiple communication channels, video image streams, or other high-speed, real-time inputs, making Stratix III devices a cost-effective solution for many applications.

Referenced Documents

- altfp_mult Megafuction User Guide
- ALTMEMPHY Megafuction User Guide
- altmult_accum Megafuction User Guide
- altmult_add Megafuction User Guide
- Design Example 1: Parallel FIR
- Design Example 2: Multi-Channel FIR
- Design Example 3: MAC_FIR (vhdl)
- Design Example 4: Large Mult_Add
- DSP Blocks in Stratix III Devices chapter in volume 1 of the Stratix III Device Handbook
- lpm_mult Megafuction User Guide
- lpm_add_sub Megafuction User Guide
Revision History  

Table 5 shows the revision history for this application note.

<table>
<thead>
<tr>
<th>Date and Document Version</th>
<th>Changes Made</th>
<th>Summary of Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>February 2008 v1.0</td>
<td>Initial Release.</td>
<td>—</td>
</tr>
</tbody>
</table>