

Parallel FIR Filters

This chapter describes the implementation of high-performance, parallel, full-precision FIR filters using the DSP48 slice in a Virtex-4 device. Because the Virtex-4 architecture is flexible, it is practical to construct custom FIR filters to meet the requirements of a specific application. Creating optimized, parallel filters saves either resources and potential clock cycles.

This chapter demonstrates two parallel filter architectures: the Transposed and Systolic Parallel FIR filters. The reference design files in VHDL and Verilog permit filter parameter changes including coefficients and the number of taps.

Overview

There are many filtering techniques available to signal processing engineers. A common filter implementation for high-performance applications is the fully parallel FIR filter. Implementing this structure in the Virtex-II architecture uses the embedded multipliers and slice based arithmetic logic. The Virtex-4 DSP48 slice introduces higher performance multiplication and arithmetic capabilities specifically designed to enhance the use of parallel FIR filters in FPGA-based DSP.

Parallel FIR Filters

A wide variety of filter architectures are available to FPGA designers due to the “liquid hardware” nature of FPGAs. The type of architecture chosen is typically determined by the amount of processing required in the available number of clock cycles. The two most important factors are:

- Sample Rate (F_s)
- Number of Coefficients (N)

In Figure 5-1, as the sample rate increases and the number of coefficients increase, the architecture selected for a desired FIR filter becomes a more parallel structure involving more multiply and add elements. Chapter 4, “MAC FIR Filters” addresses the details of the sequential processing FIR filters including the single and dual MAC FIR filter. This chapter investigates the other extreme of the fully parallel FIR filter as required to filter the fastest data streams.

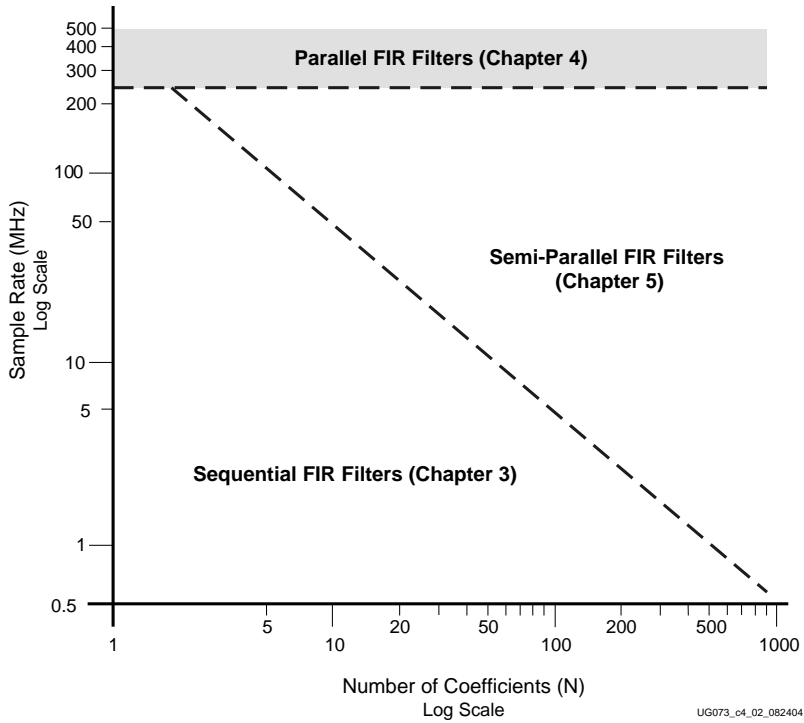


Figure 5-1: Selecting Filter Architectures

The basic parallel architecture, shown in Figure 5-2, is referred to as the Direct Form Type 1.

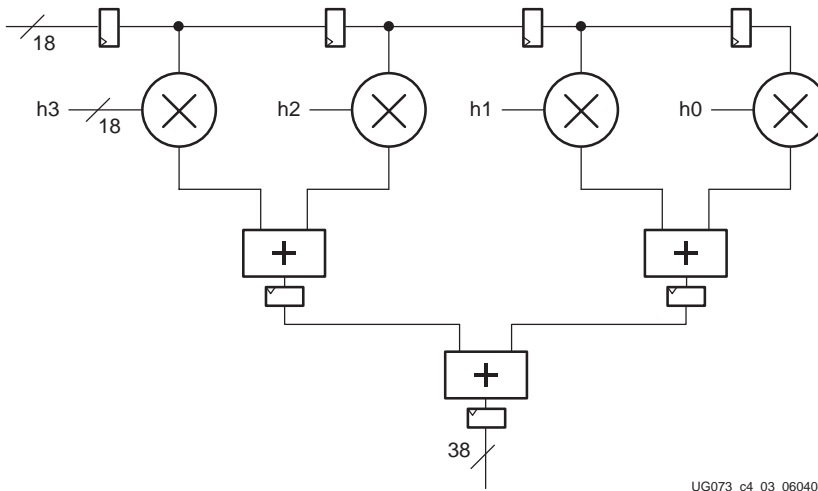


Figure 5-2: Direct Form Type 1 FIR Filter

This structure implements the general FIR filter equation of a summation of products as defined in Equation 5-1.

$$y_n = \sum_{i=0}^{N-1} x_{n-i} h_i \tag{Equation 5-1}$$

In Equation 5-1, a set of N coefficients is multiplied by N respective data samples. The results are summed together to form an individual result. The values of the coefficients determine the characteristics of the filter (e.g., a low-pass filter).

The history of data is stored in the individual registers chained together across the top of the architecture. Each clock cycle yields a new complete result and all multiplication and arithmetic required occurs simultaneously. In sequential FIR filter architectures, the data buffer is created using Virtex-4 dedicated block RAMs or distributed RAMs. This demonstrates a trend; as algorithms become faster, the memory requirement is reduced. However, the memory bandwidth increases dramatically since all N coefficients must be processed at the same time.

The performance of the Parallel FIR filter is calculated in Equation 5-2.

$$\text{Maximum Input Sample Rate} = \text{Clock Speed} \tag{Equation 5-2}$$

The bit growth through the filter is the same for all FIR filters and is explained in the section “Bit Growth” in Chapter 4.

Transposed FIR Filter

The DSP48 arithmetic units are designed to be easily and efficiently chained together using dedicated routing between slices. The Direct Form Type I uses an adder tree structure. This makes it difficult to chain the blocks together. The Transposed FIR filter structure (Figure 5-3) is more optimal for use with the DSP48 Slice.

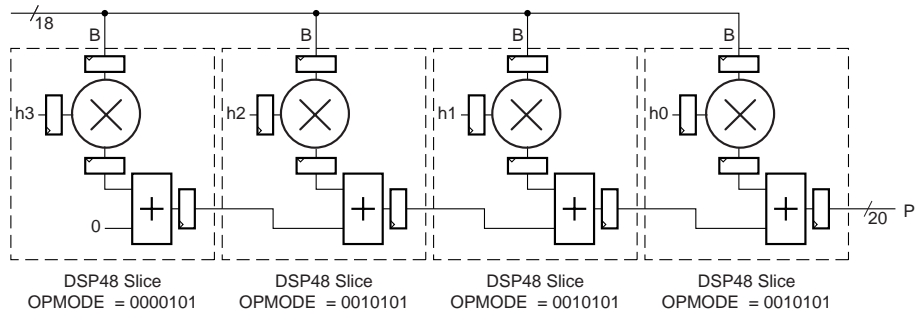


Figure 5-3: Transposed FIR Filter

The input data is broadcast across all the multipliers simultaneously, and the coefficients are ordered from right to left with the first coefficient, h_0 , on the right. These results are fed into the pipelined adder chain acting as a data buffer to store previously calculated inner products in the adder chain. The rearranged structure yields identical results to the Direct Form structure, but gains from the use of an adder chain. This different structure is easily mapped to the DSP48 slice without

additional external logic. If more coefficients are required, then more DSP48 slices are required to be added to the chain.

The configuration of the DSP48 slice for each segment of the Transposed FIR filter is shown in Figure 5-4. Apart from the very first segment, all processing elements are to be configured as in Figure 5-4. OPMODE is set to multiply mode with the adder combining the results from the multiplier and from the previous DSP48 slice through the dedicated cascade input (PCIN). OPMODE is set to binary 0010101.

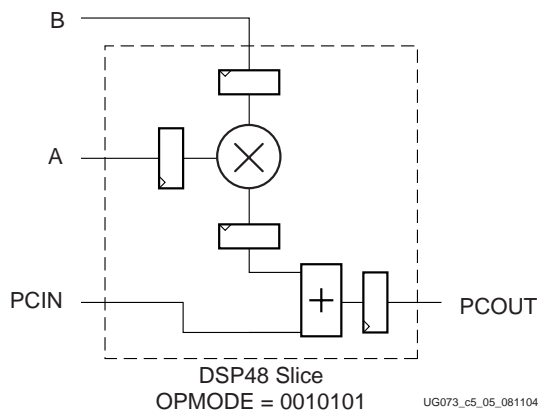


Figure 5-4: Transpose Multiply-Add Processing Element

Advantages and Disadvantages

The advantages to using the Transposed FIR filter are:

- **Low Latency:** The maximum latency never exceeds the pipelining time through the slice containing the first coefficient. Typically, this is three clock cycles between the data input and the result appearing.
- **Efficient Mapping to the DSP48 Slice:** Mapping is enabled by the adder chain structure of the Transposed FIR filter. This extendable structure supports both large and small FIR filters.
- **No External Logic:** No external FPGA fabric is required, enabling the highest possible performance to be achieved.

The disadvantage to using the Transposed FIR filter is:

- **Limited Performance:** Performance may be limited by a high fanout input signal if there are a large number of taps.

Resource Utilization

An N coefficient filter uses “N” DSP48 slices. A design cannot use symmetry to reduce the number of DSP48 slices when using the Transposed FIR filter structure.

Systolic FIR Filter

The systolic FIR filter is considered an optimal solution for parallel filter architectures. The systolic FIR filter also uses adder chains to fully utilize the DSP48 slice architecture (Figure 5-5).

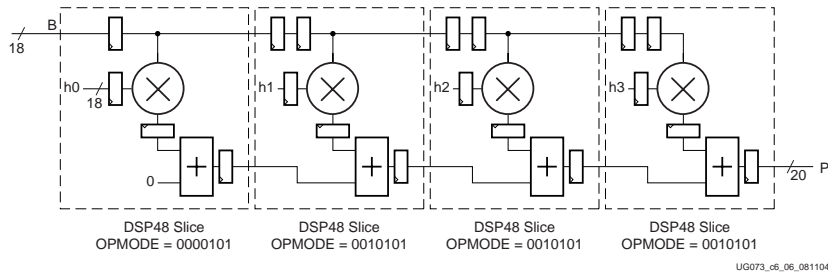


Figure 5-5: Systolic FIR Filter

The input data is fed into a cascade of registers acting as a data buffer. Each register delivers a sample to a multiplier where it is multiplied by the respective coefficient. In contrast to the Transposed FIR filter, the coefficients are aligned from left to right with the first coefficients on the left side of the structure. The adder chain stores the gradually combined inner products to form the final result. As with the Transposed FIR filter, no external logic is required to support the filter and the structure is extendable to support any number of coefficients.

The configuration of the DSP48 slice for each segment of the Systolic FIR filter is shown in Figure 5-6. Apart from the very first segment, all processing elements are to be configured as shown in Figure 5-6. OPMODE is set to multiply mode with the adder combining the results from the multiplier and from the previous DSP48 slice through the dedicated cascade input (PCIN). OPMODE is set to binary 0010101. The dedicated cascade input (BCIN) and dedicated cascade output (BCOUT) are used to create the necessary input data buffer cascade.

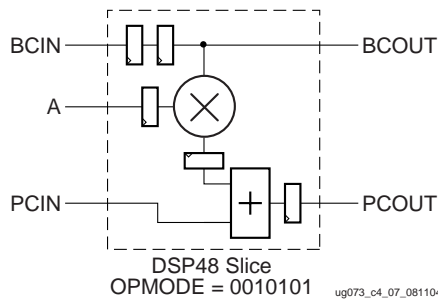


Figure 5-6: Systolic Multiply-Add Processing Element

Advantages and Disadvantages

The advantages to using the Systolic FIR filter are:

- **Highest Performance:** Maximum performance can be achieved with this structure because there is no high fanout input signal. Larger filters can be routing-limited if the number of coefficients exceeds the number of DSP slices in a column on a device.
- **Efficient Mapping to the DSP48 Slice:** Mapping is enabled by the adder chain structure of the Systolic FIR Filter. This extendable structure supports large and small FIR filters.
- **No External Logic:** No external FPGA fabric is required, enabling the highest possible performance.

The disadvantage to using the Systolic FIR filter is:

- **Higher Latency:** The latency of the filter is a function of how many coefficients are in the filter. The larger the filter, the higher the latency.

Resource Utilization

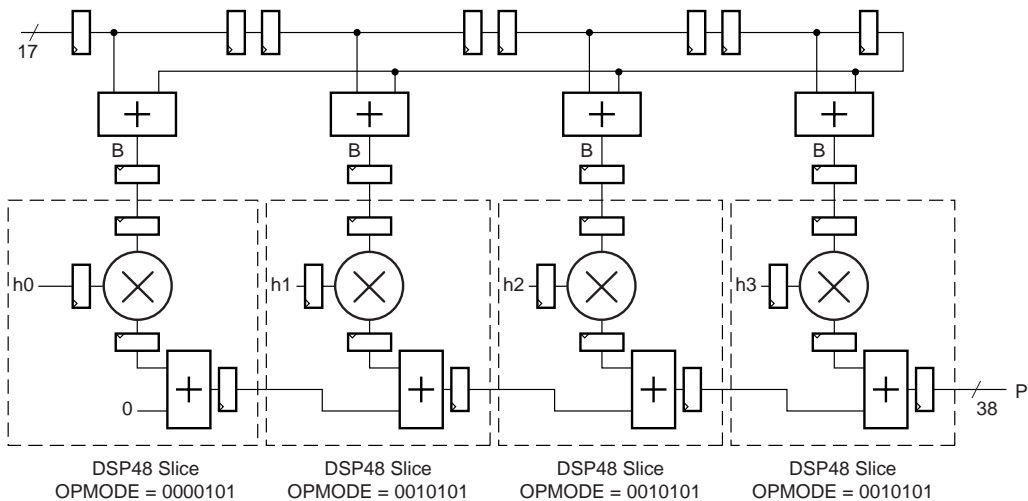
An N coefficient filter uses “N” DSP48 slices.

Symmetric Systolic FIR Filter

In Chapter 4, “MAC FIR Filters,” symmetry was examined and an implementation was illustrated to exploit this symmetric nature of the coefficients. Exploiting symmetry is extremely powerful in Parallel FIR filters because it halves the required number of multipliers, which is advantageous due to the finite number of DSP48 slices. Equation 5-3 demonstrates how the data is pre-added before being multiplied by the single coefficient.

$$(X_0 \times C_0) + (X_n \times C_n) \dots \setminus (X_0 + X_n) \times C_0 \quad (\text{if } C_0 = C_n) \quad \text{Equation 5-3}$$

Figure 5-7 shows the implementation of this type of Systolic FIR Filter structure.



UG073_c6_10_082404

Figure 5-7: Symmetric Systolic FIR

In this structure, DSP48 slices have been traded off for Fabric slices. From a performance viewpoint, to achieve the full speed of the DSP48 slice, the fabric 18-bit adder has to run at the same speed. To achieve this, register duplication can be performed on the output from the last tap that feeds all the other multipliers.

The two register delay in the input buffer time series is implemented as an SRL16E and a register output to save on logic area. A further benefit of the symmetric implementation is the reduction in latency, due to the adder chain being half the length.

Figure 5-8 shows the configuration of the DSP48 slice for each segment of the Symmetric Systolic FIR filter. Apart from the very first segment, all processing elements are to be configured as in Figure 5-8. OPMODE is set to multiply mode with the adder combining results from the multiplier and from the previous DSP48 slice via the dedicated cascade input (PCIN). OPMODE is set to binary 0010101.

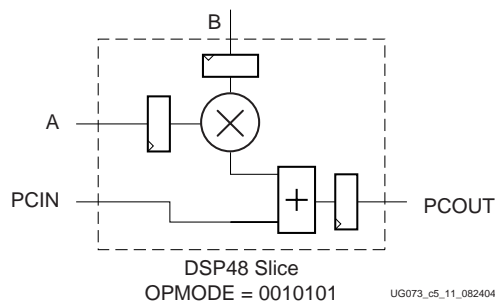


Figure 5-8: Symmetric Systolic Multiply-Add (MADD) Processing Element

Resource Utilization

An N symmetric coefficient filter uses N DSP48 slices. The slice count for the pre-adder and input buffer time series is a factor of the input bit width (n) and N . The equation for the size in slices is:

$$((n+1) * (N/2)) + (n/2) \quad \text{Equation 5-4}$$

For the example illustrated in Figure 5-7, the size is $(17+1) * 8/2 + 17/2 = 81$ slices.

Rounding

The number of bits on the output of the filter is much larger than the input and must be reduced to a manageable width. The output can be truncated by simply selecting the MSBs required from the filter. However, truncation introduces an undesirable DC data shift. Due to the nature of two's complement numbers, negative numbers become more negative and positive numbers also become more negative. The DC shift can be improved with the use of symmetric rounding, where positive numbers are rounded up and negative numbers are rounded down.

The rounding capability in the DSP48 slice maintains performance and minimizes the use of the FPGA fabric. This is implemented in the DSP48 slice using the C input port and the Carry In port. Rounding is achieved by:

For positive numbers: Binary Data Value + 0.10000... and then truncate

For negative numbers: Binary Data Value + 0.01111... and then truncate

The actual implementation always adds 0.0111... to the data value through the C port input as in the negative case, and then adds the extra carry in required to adjust for positive numbers. Table 5-1 illustrates some examples of symmetric rounding.

Table 5-1: Symmetric Rounding Examples

Decimal Value	Binary Value	Add Round	Truncate: Finish	Rounded Value
2.4375	0010.0111	0010.1111	0010	2
2.5	0010.1000	0011.0000	0011	3
2.5625	0010.1001	0011.0001	0011	3
-2.4375	1101.1001	1110.0000	1110	-2
-2.5	1101.1000	1101.1111	1101	-3
-2.5625	1101.0111	1101.1110	1101	-3

For both the Transposed and Systolic Parallel FIR filters, the C input is used at the beginning of the adder chain to drive the carry value into the accumulated result. The final segment uses the MSB of the PCIN as the carry-in value to determine if the accumulated product is positive or negative. CARRYINSEL is used to select the appropriate carry-in value. If positive, the carry-in value is used, and if negative, the result is kept the same (see Figure 5-9).

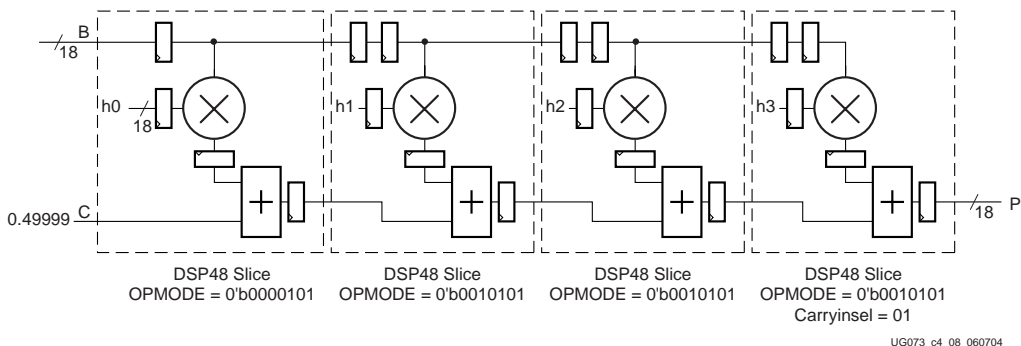


Figure 5-9: Systolic FIR Filter with Rounding

The one problem with this solution occurs when the final accumulated inner product input to the final DSP48 slice is very close to zero. If the value is positive and the final inner product makes the result negative (leading to a rounding down), then an incorrect result occurs because the rounding function assumes a positive number instead of a negative. The last coefficient in typical FIR filters is very small, so this situation rarely occurs. However, if absolute certainty is required, an extra DSP48

slice can perform the rounding function (see Figure 5-10). A Transposed FIR filter can have exactly the same problem as the Systolic FIR filter.

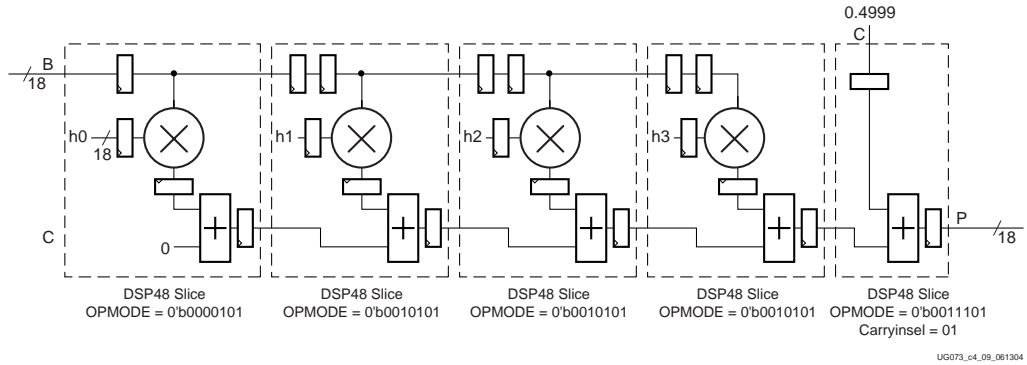


Figure 5-10: Systolic FIR Filter with Separate Rounding Function

Performance

When examining the performance of a Virtex-4 Parallel FIR filter, a Virtex-II Pro design is a valuable reference. Table 5-2 illustrates the ability of the Virtex-4 DSP48 slice to greatly reduce logic fabric requirements while improving the speed of the design and reducing the power utilization of the filter.

Table 5-2: Performance Analysis

Filter Type	Device Family	Size	Performance	Power (Watts)
18 x 18 Parallel Transposed FIR Filter (51 Tap Symmetric)	Virtex-II Pro FPGA	1860 Slices 26 Embedded Multipliers	300 MHz Clock Speed 300 MSPS	TBD
18 x 18 Parallel Systolic FIR Filter (51 Tap Symmetric)	Virtex-II Pro FPGA	2958 Slices 26 Embedded Multipliers	300 MHz Clock Speed 300 MSPS	TBD
18 x 18 Parallel Transposed FIR Filter (51 Tap Symmetric)	Virtex-4 FPGA	0 Slices 51 DSP48 Slices	400 MHz Clock Speed 400 MSPS	TBD

Table 5-2: Performance Analysis (*Continued*)

Filter Type	Device Family	Size	Performance	Power (Watts)
17 x 18 Systolic FIR Filter (51 Tap Non-symmetric)	Virtex-4 FPGA	0 Slices 51 DSP48 Slices	450 MHz Clock Speed 450 MSPS	TBD
17 x 18 Systolic FIR Filter (51 Tap Symmetric)	Virtex-4 FPGA	477 Slices 26 DSP48 Slices	400 MHz Clock Speed 400 MSPS	TBD

Conclusion

Parallel FIR filters are commonly used in high-performance DSP applications. With the introduction of the Virtex-4 DSP48 slice, DSPs can be achieved in a smaller area, thereby producing higher performance with less power penalty.

Designers have tremendous flexibility in determining the desired implementation, and also have the ability to change the implementation parameters. The ability to “tune” a filter in an existing system or to have multiple filter settings is a distinct advantage. By making the necessary coefficient changes in the synthesizable HDL code, the reconfigurable nature of the FPGA is fully exploited. The coefficients can be either hardwired to the A input of the DSP48 slices or stored in small memories and selected to change the filter characteristics. The HDL and System Generator for DSP reference designs are easily modified to achieve specific requirements.