CoreFIR v8.6

Handbook

Microsemi

Table of Contents

Introduction

Core Overview

The finite impulse response (FIR) filter is one of the most essential building blocks in digital signal processing (DSP) systems. Digital filters have been used in many systems to remove unwanted noise, improve signal quality, or shape signal spectrum.

CoreFIR provides a configurable high performance multiplier-accumulator (MAC)-based FIR filter. The core is available as a register transfer level (RTL) code of the filter, both in Verilog and VHDL languages.

The core implements a range of filter types:

- Single-rate
  - Fully enumerated (parallel)
  - Folded (semi-parallel)
- Multi-rate
  - Polyphase interpolation
  - Polyphase decimation

The N-tap single rate FIR filter computes its sum-of-products output *y(k)* as explained in EQ 1:

$$y(k) = \sum_{j=0}^{N-1} x(k-j)c(j) \ = x(k)c(0) + x(k-1)c(1) + \cdots + x(k-N+1)c(N-1)$$

EQ 1

A series of coefficients *c(0), c(1), c(2), …, c(N-1)* are the filter impulse response. The coefficient values define whether the filter is a low-pass, band-pass, or high-pass filter. The fully enumerated filter type has as many physical MACs as filter taps. Such architecture provides the fastest input sample rate. The folded filter utilizes fewer physical MACs, taking advantage of a ratio between high clock rate and a slower input sample rate. It reuses the MACs over vacant clock intervals to process the input samples.

Figure 1 shows the single rate filter functional block diagram. The figure presents only a general notion of a digital filter computational component. An actual realization can be quite different. This handbook contains chapters dedicated to every filter type. Refer to the specific chapter for more information on each particular type.
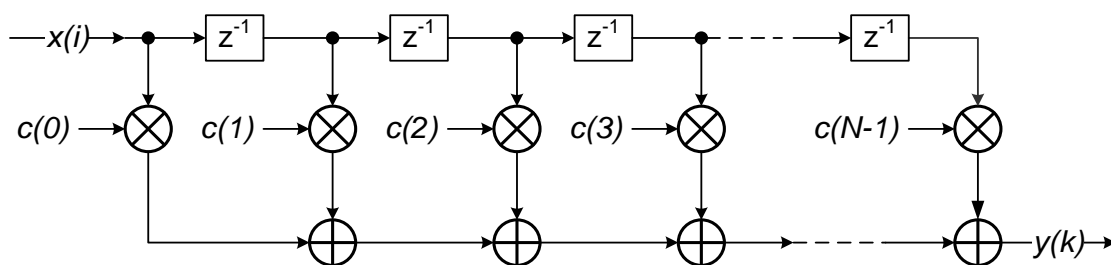

Figure 1 · Single-Rate FIR Filter Functional Block Diagram

CoreFIR supports the following three filter coefficient modes:

- Constant coefficient
- Multiple constant coefficient sets
- Reloadable coefficient

In Constant Coefficient mode, a single coefficient set can be programmed into field programmable gate array (FPGA) fabric. In Multiple Constant Coefficient mode, the multiple constant coefficient sets can be programmed, and the FPGA stores all of them. It is possible to switch between the pre-loaded sets at any moment during runtime, thus changing the filter impulse response. In Reloadable Coefficient mode, the coefficients can be reloaded at the runtime.

Internal filter processing takes place at full precision to reduce truncation or rounding noise and to avoid a risk of overflow. The filtered results are presented in full precision as well.

Figure 2 shows an example of using a FIR filter. Digital samples to be filtered enter the filter input and filtered samples appear at the filter output.
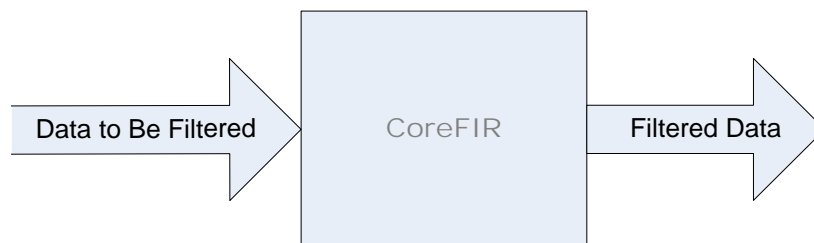
Figure 2 · CoreFIR Example Application

Key Features

CoreFIR supports various filter types: Fully Enumerated, folded, and polyphase interpolator. The key features for each type are listed in Table 1.

Table 1 Key Feature Support

Feature	Fully Enumerated	Folded	Interpolation
Number of filter coefficients	2 to 2N, where N is a number of physically available MAC's	4 to 1024	2 – 1,024
Input data bit width	2 – 18	2 – 18	2 – 18
Coefficient bit width	2 – 18	2 – 18	2 – 18
Signed and unsigned data coefficients	Yes	Yes	Yes
Full precision output	Yes	Yes	Yes
Coefficient symmetry optimization	Yes	No	No
Constant coefficients and constant coefficient sets	Yes	Yes	Yes
Run-time reloadable coefficients	Yes	Yes	Yes
RAM-based coefficient storage	No	Yes	Yes
RAM-based data storage	No	Yes	Yes

Supported Families

The core supports RTG4™, SmartFusion®2, and IGLOO®2 FPGA families.

Core Version

This handbook applies to CoreFIR v8.6.

Utilization and Performance

CoreFIR has been implemented in the SmartFusion2 devices using speed grade-1. A summary of the implementation data is listed in Table 2.

Fully Enumerated Filter

Table 2 · Fully Enumerated CoreFIR M2S050 Device Utilization and Performance

| Taps | Symmetry | Architecture | Configuration | Bit Width | | Mathblocks | Resource Usage | | | Max. Data Rate (Msps) |
				Coefficient	Data		Sequential	Combinatorial	Total	
24	No	Transposed	1	18	18	24	28	168	196	310
32	No	Transposed	1	18	18	32	40	385	425	310
48	No	Transposed	1	18	18	48	56	403	459	310
72	No	Transposed	1	18	18	72	84	637	721	286
16	No	Systolic	1	18	18	16	35	714	749	423
24	No	Systolic	1	18	18	24	51	1,020	1,071	423
48	No	Systolic	1	18	18	48	103	2,144	2,247	412
72	No	Systolic	1	18	18	72	155	3275	3,430	371

Note: Data in this table were achieved using typical synthesis settings. Synplify Frequency (MHz) was set to be 400. CoreFIR was configured for signed coefficients and data, a single set of constant coefficients, and RSTN pin was tied to Vcc
Layout settings were set as follows:
- Block creation enabled
- Timing-driven High Effort Layout

The maximum data rate shown also reflects the maximum clock rate. For example, the rate of 100 Msamples per second corresponds to the maximum clock rate of 100 MHz.

Folded Filter

The filter was realized on M2S150 device.

Table 3 Folded CoreFIR Device Utilization and Performance

Coefficients						Use RAM			Utilization					Maximal Clock Rate, MHz
Type	Number of Taps	Width	Number of Sets	Data Width	Folding Factor	For Coefficients	For Data	Micro RAM Depth	4LUT	DFF	R64X18	RAM1K18	MACC	
Constant	29	18	1	18	2	No	No	0	2271	3061	-	-	15	267
Constant	29	18	1	18	3	No	No	0	1776	2392	-	-	10	258
Constant	29	18	1	18	10	No	No	0	1102	1454	-	-	3	254
Constant	60	18	1	18	60	Yes	Yes	64	404	438	2	-	1	250
Constant	60	18	3	18	60	Yes	Yes	64	427	345	1	1	1	250
Reloadable	60	18	1	18	60	Yes	Yes	64	385	382	1	1	1	250
Constant	1024	18	1	18	100	Yes	Yes	0	1733	1778	-	2	11	250
Reloadable	1024	18	1	18	100	Yes	Yes	0	1547	1869	-	3	11	273

Note: Data in this table were achieved using typical synthesis settings. Synplify Frequency (MHz) was set to be 300. CoreFIR was configured for signed coefficients and data, RSTN pin was tied to Vcc and REF pin was grounded
Layout settings were set as follows:
- Block creation enabled
-Timing-driven High Effort Layout

Interpolation Filter

The filter was realized on M2S150 device

Table 4 Interpolation CoreFIR Device Utilization and Performance

Coefficients								Utilization					Maximal Clock Rate, MHz
Type	Number of Taps	Width	Number of Sets	Data Width	Interpolation Factor	Use RAM For Coefficients	Micro RAM Depth	4LUT	DFF	R64X18	RAM1K18	MACC	
Constant	16	18	1	18	4	No	0	359	700	-	-	4	420
Constant	16	18	1	18	8	No	0	305	477	-	-	2	418
Constant	16	18	4	18	4	No	0	600	916	-	-	4	385
Constant	16	18	4	18	8	No	0	529	708		-	2	348
Reloadable	16	18		18	4	No	0	671	1285			4	257
Reloadable	16	18		18	8	No	0	605	1036			2	267
Constant	128	18	4	18	16	Yes	0	863	1205	-	8	8	367
Constant	128	18	4	18	32	Yes	0	574	690	-	4	4	374
Constant	128	18	1	18	16	Yes	64	782	1178	8		8	250
Constant	128	18	4	18	16	Yes	64	863	1205	8		8	250
Reloadable	128	18		18	16	Yes	64	727	1236	8		8	250
Constant	1024	18	1	18	256	Yes	0	684	920		4	4	351
Constant	1024	18	4	18	256	Yes	0	766	935		4	4	321
Reloadable	1024	18		18	256	Yes	0	459	963		4	4	343

Note: Data in this table were achieved using typical synthesis settings. Synplify Frequency (MHz) was set to be 400. CoreFIR was configured for signed coefficients and data, RSTN pin was tied to Vcc and REF pin was grounded
Layout settings were set as follows:
- Block creation enabled
-Timing-driven High Effort Layout

Decimation Filter

The filter was realized on M2S150 device

Table 5 Decimation CoreFIR Device Utilization and Performance

| Coefficients | | | | | | Use RAM | | | Utilization | | | | | Maximal Clock Rate, MHz |
Type	Number of Taps	Width	Number of Sets	Data Width	Decimation Factor	For Coefficients	For Data	Micro RAM Depth	4LUT	DFF	R64X18	RAM1K18	MACC	
Constant	16	18	1	18	4	No	No	0	388	977			5	425
Constant	16	18	4	18	4	No	No	0	591	1186			5	413
Reloadable	16	18		18	4	No	No	0	694	1524			5	294
Constant	128	18	1	18	16	Yes	Yes	64	1058	1390	15		9	250
Constant	128	18	4	18	16	Yes	Yes	64	1077	1404	15		9	250
Reloadable	128	18		18	16	Yes	Yes	64	1036	1440	15		9	250
Constant	1024	18	1	16	256	Yes	Yes	0	727	12936		4	5	319
Constant	1024	18	4	16	256	Yes	Yes	0	729	12942		4	5	319
Reloadable	1024	18		16	64	Yes	Yes	0	484	12962		4	5	330
Constant	1000	18	1	16	250	Yes	Yes	0	717	12645		4	5	313
Constant	1020	18	2	16	60	Yes	Yes	64	2291	2718	16	17	18	250
Constant	1024	18	3	16	256	Yes	Yes	0	729	12942		4	5	351
Reloadable	1000	18		16	250	Yes	Yes	0	488	12647		4	5	338

Note: *Data in this table were achieved using typical synthesis settings. Synplify Frequency (MHz) was set to be 400. CoreFIR was configured for signed coefficients and data, RSTN pin was tied to Vcc and REF pin was grounded*
Layout settings were set as follows:
- Block creation enabled
-Timing-driven High Effort Layout

Filter Types

Depending on a user configuration, CoreFIR generates one of the supported filter types.

Fully Enumerated Filter

The single rate parallel FIR filter provides the highest input sampling rate processing. The performance of the parallel filter expressed as the maximum input sample frequency equals the maximum clock rate, that is, the filter can process a sample per clock interval. Figure 1 on page 5 shows an accurate functional model of the fully enumerated filter. The filter utilizes as many MAC blocks as the number of coefficients also called the number of taps. The largest filter (in terms of number of coefficients) a particular FPGA device can carry is limited by the number of available MAC blocks.

Folded Filter

This one is a single-rate semi-parallel filter. In many practical cases, the filter input sample rate equals only a fraction of the FPGA clock rate. Then the processing power of the MAC block can be re-used to process more than a single filter tap. At every clock interval, the semi-parallel filter computes and accumulates several products of EQ 1 on page 5. In a few clocks before a next input sample comes in, all N products are accumulated.

The number PHY_TAPS of the products the filter engine computes at every clock interval is determined based on required number of taps N, input sample frequency, and the FPGA clock frequency as follows:

$$PHY_{TAPS} \geq N * \frac{\text{Sample frequency}}{\text{Clock frequency}}$$

EQ 2

Figure 3 on page 14 shows the folded filter simplified functional block diagram. Input data samples come to a Data FIFO that collects a number of samples sufficient to compute PHY_TAPS products each clock interval. The Coefficient ROM stores N coefficients. The MAC engine has PHY_TAPS MACs, a delay line comprised of Z^{-1} blocks, and an output multiplexer. After filling in a delay line with necessary input samples, the MAC engine reads N data samples simultaneously with N coefficients to compute PHY_TAPS filtered results. After the computation is over, the PHY_TAPS results are collected in PHY_TAPS accumulators. The multiplexer (MUX) puts them out one by one.
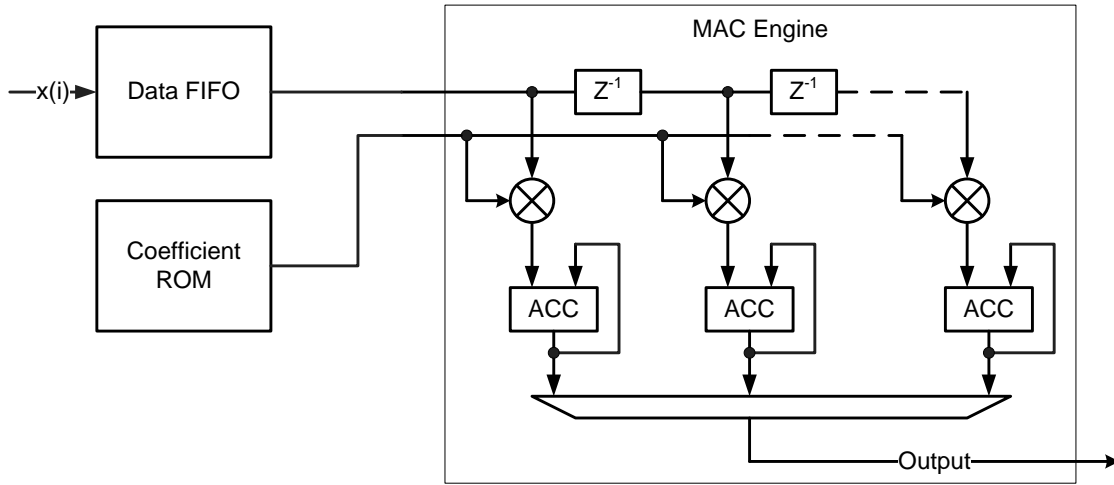
Figure 3 · Semi-Parallel FIR Filter Functional Block Diagram

$$\text{Maximum sample frequency} = \text{Clock frequency} * (\text{PHY\_TAPS})/N$$

EQ 3

There is no need to indicate whether the single rate filter is to be fully enumerated or folded. It is only required to indicate the clock, and input sample frequencies. If the clock to sample rate ratio is not less than 2, CoreFIR automatically generates the folded type. If the fully enumerated type is needed, the same value for the clock and sample frequencies must be entered.

Polyphase Interpolation Filter

The primary reason for interpolation is to increase the sampling rate at the output of one system so that another system operating at a higher sampling rate can input the signal.

Through calculations on existing data, interpolation fills in missing information between the samples of a signal. Interpolation increases a sample rate by an integer factor L.

The architecture calculates output using N/L multipliers, where N is a number of filter coefficients and L is an Interpolation factor, to get an overall computational saving of (N - N/L) compared to the straightforward implementation.

At every clock interval the architecture computes and accumulates several products, as shown in EQ 4.

$$Yp(k) = \sum_{j}^{N} X(k-j) * h(j),$$

EQ 4

Where, P = 0 to (L-1) and j = P, P+ (L-1), P+ 2*L – 1, 3*L – 1,…N

Figure 4 on page 13 provides an example of the polyphase interpolation filter architecture for a TAP of 16 and Interpolation factor of 4. The interpolator contains distributed coefficient ROM, one storage per physical filter tap. In Figure 4 on page 13, the first storage keeps coefficients h_0 to h_3, the second storage keeps coefficients h_4 to h_7, and so on.
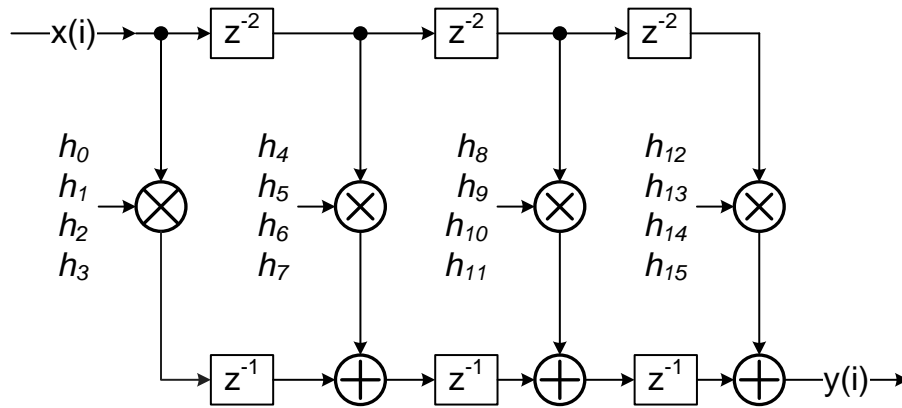
Figure 4 · 16-Tap Polyphase Interpolation Filter, L = 4

Polyphase Decimation Filter

The motivation for decimation is to reduce the cost of processing: the calculation to implement a DSP system, generally is proportional to the sampling rate, so the use of a lower sampling rate usually results in a cheaper implementation.

The decimation factor is simply the ratio of the input rate to the output rate. It is usually symbolized by "M", so input rate/output rate = M.

If a signal is defined by N samples, M:1 decimation is achieved by throwing away M − 1 samples after every sample kept In an M:1 decimator, the output data rate is 1/M times the input data rate, and M is the decimation factor.

The output Y is given by EQ 5.

$$Y(n) = \sum_{i}^{N} X(i) * h[nM - i]$$

EQ 5

Y (n) is discarded for n != 0, M, 2*M, 3*M etc.
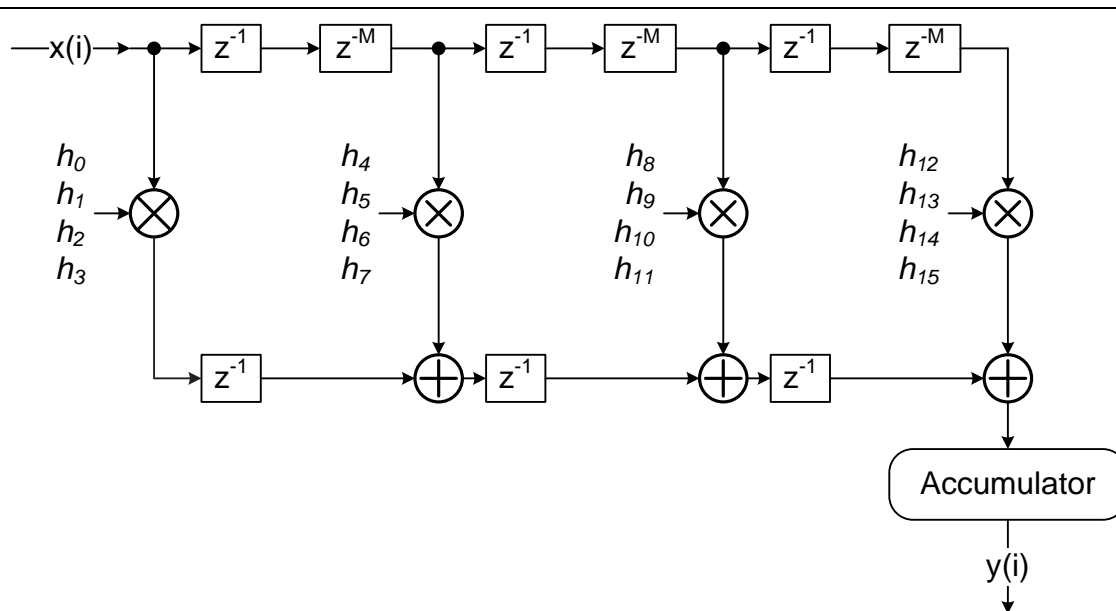
Figure 5 · 16 TAP Polyphase Decimation Filter, M = 4

Figure 5 shows an example of the polyphase decimation filter structure for a number of taps TAPS = 16 and a decimation factor M = 4. A decimated output sample is obtained at the accumulator. A valid output is available only at every M[th] clock. The decimator contains distributed coefficient ROM, one storage per physical filter tap. In Figure 5, the first storage keeps coefficients h_0 to h_3, the second storage keeps coefficients h_4 to h_7, and so on.

Fully Enumerated Filter

Filter Description

This section describes how the fully enumerated CoreFIR filter implements hardware (HW) architectures. All the architectures realize EQ 1 on page 5.

Transposed Architecture

Figure 6 depicts the transposed HW structure. The architecture is functionally equivalent to the structure shown in Figure 1, but eliminates the need for a multi-input adder. Instead, it uses the pipelined two-input adder chain, which makes it highly beneficial for the HW implementation. It also offers low latency implementation.
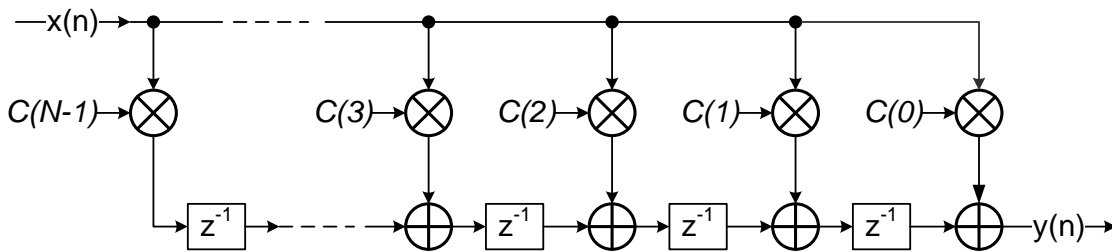
Figure 6 · Transposed FIR Filter Architecture

Systolic Architecture

The systolic architecture adds more pipelines to the Direct Form I structure of Figure 1. Depending on particular filter parameters, the systolic structure can offer better performance due to extensive pipelining. The structure is always used to implement FIR filters with symmetric impulse response. The systolic architecture is shown in Figure 7.
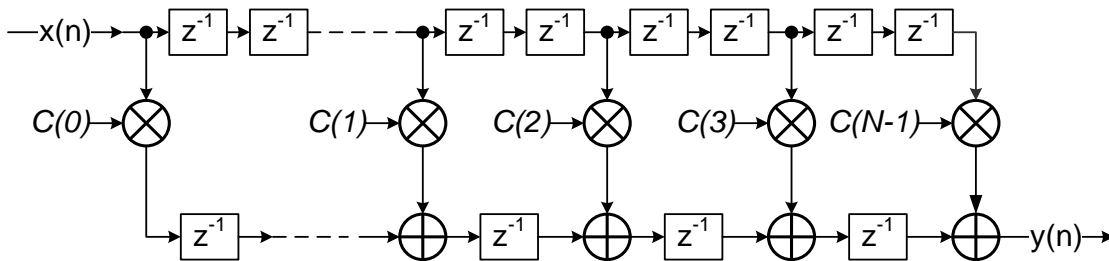

Figure 7 · Systolic FIR Filter Architecture

Fully Enumerated Filter Symmetry

Many FIR filters are symmetric. The fully enumerated filter exploits the filter symmetry to minimize the number of physical MACs required for the implementation. Figure 8 shows the impulse response for a 9-tap symmetric filter. In this filter $c(0) = c(8)$, $c(1) = c(7)$, $c(2) = c(6)$, and $c(3) = c(5)$. When exploited, the symmetry can substantially reduce the number of multipliers. The symmetric FIR filter implementation first adds together the data samples that engage equal coefficients. Figure 9 shows the corresponding functional block diagram.
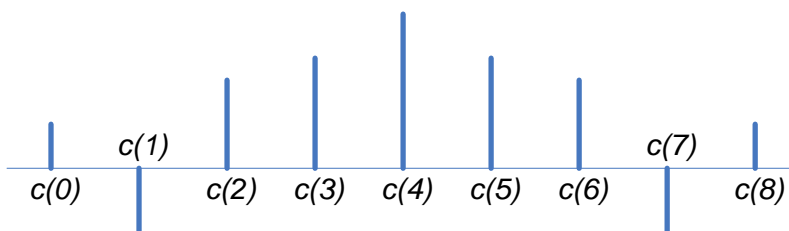
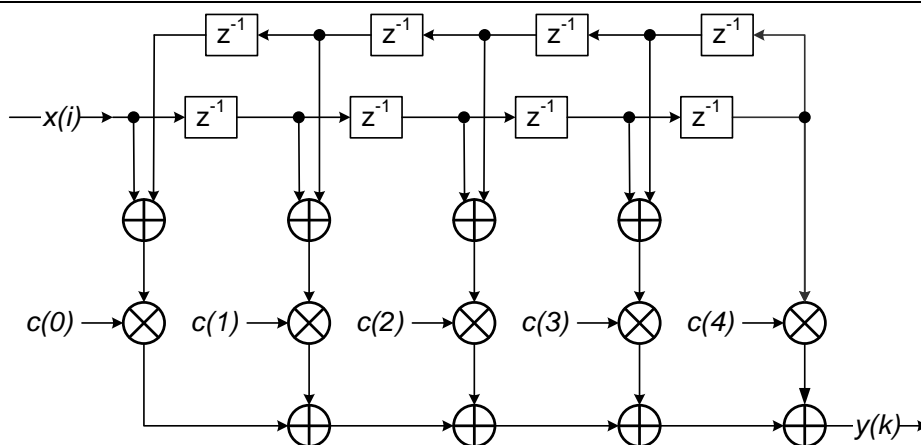**Figure 8** · Symmetric FIR Filter Impulse Response

Figure 9 · Symmetric FIR Filter

There are a few types of symmetry. Figure 8 shows an example of an odd symmetry, where the number of taps is odd, and a central tap is unique. Even symmetry is applied to a filter with an even number of taps. The even symmetric filter does not have a unique center tap.

Figure 10 shows an 8-tap anti-symmetric filter impulse response. In this filter $c(0) = -c(7)$, $c(1) = -c(6)$, $c(2) = -c(5)$, and $c(3) = -c(4)$. Similarly, odd anti-symmetric filters exist. The core exploits all four symmetry types to nearly double the tap number, given the same number of mathblocks utilized.
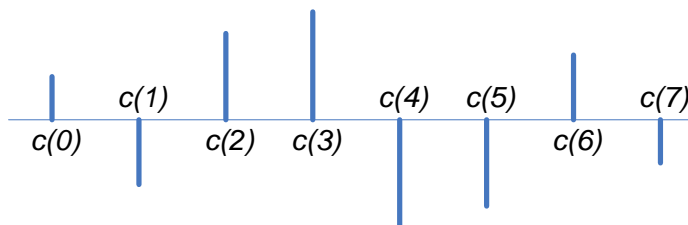

Figure 10 · Even Anti-Symmetric FIR Filter Impulse Response

All symmetric/anti-symmetric filters utilize the systolic architecture in Figure 7.

Reloadable Coefficient Mode

Figure 11 shows a functional block diagram for Reloadable Coefficient mode. This mode uses two memory pages: active and auxiliary. The active page comprises Storage registers connected to the multiplier coefficient inputs. The auxiliary page implements a multi-bit Shift register. Once a new coefficient vector is loaded in the auxiliary page, it can be copied in the active page within a single clock period. Then the auxiliary page is ready to accept another coefficient vector. Thus, the filter operation is not disturbed during coefficient reload.

New loadable coefficients arrive at COEFI input to be shifted in the multi-bit Shift register. Every new coefficient shifts the register contents one step to the left. The coefficients come in natural order, c(0), c(1),…,c(N-1). After the last coefficient arrives, the coefficients in the auxiliary page are distributed, as shown in Figure 11. On the COEF_ON signal the recently entered coefficients are loaded into the active page. From this time on, they are used as the filter coefficients.
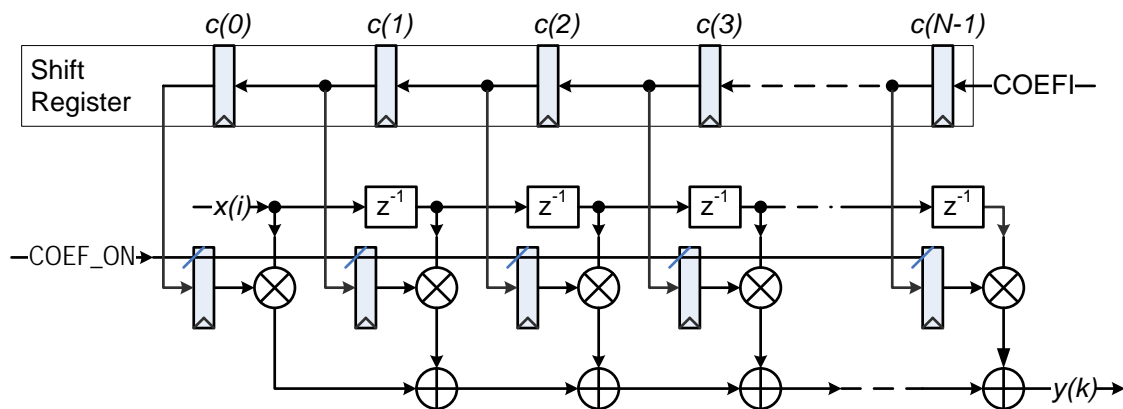

Figure 11 · Reloadable Coefficient Mode.

Fully Enumerated Interface Description

Parameters/Generics

The parallel fully enumerated CoreFIR RTL has parameters (Verilog) or generics (VHDL), described in Table 6. All the parameters and generics are positive integer types. The table shows the superset of the parameters used by all FIR filter types, while indicating which parameters the fully enumerated type does not utilize.

Table 6 · Fully Enumerated Filter Parameter/Generic Descriptions

Parameter Name	Valid Range	Default	Description
CFG_ARCH	1-4	1	Filter type: 1: Fully Enumerated 2: Folded 3: Polyphase Interpolator 4: Polyphase Decimator
TAPS	2-2N[1]	16	Number of taps. For a symmetric filter the valid range is 2 to 2*N (2 to 2*N-1 for odd symmetry filters), for the other filters the range is 2 to N.
COEF_TYPE	0-1	0	0: Constant coefficients (including multiple coefficient sets) 1: Reloadable coefficients
COEF_SETS	1-16	1	1: Single coefficient set 2-16: Multiple coefficient sets Valid when constant coefficient type is selected (COEF_TYPE==0)
COEF_SYMM	0-2	0	0: Not symmetric coefficients 1: Symmetric coefficients 2: Anti-symmetric coefficients
COEF_UNSIGN	0-1	0	0: Signed coefficients 1: Unsigned coefficients
COEF_WIDTH	2-18	12	Coefficient bit width. Signed coefficient width ranges from 2 to 18 bits; unsigned from 2 to 17 bits.
DATA_UNSIGN	0-1	0	0: Signed input data 1: Unsigned input data
DATA_WIDTH	2-18	12	Data bit width. Signed data width ranges from 2 to 18 bits; unsigned from 2 to 17 bits.
SYSTOLIC	0-1	0	• 0: Transposed architecture • 1: Systolic architecture Valid when non-symmetric filter is being implemented (COEF_SYMM==1). Otherwise the Systolic architecture is enforced.
INP_REG	0-1	1	Disable (0) or enable (1) input registers. Enabling the registers helps improving the filter speed.
FPGA_FAMILY	19, 24, 25	19	The target FPGA family: SmartFusion2 (19), IGLOO2 (24) or RTG4 (25)
DIE_SIZE	5 - 50	20	Die size. The parameter is automatically derived from FPGA device name set through the Libero® project settings dialog. If the Libero device selection changes, the core configuration interface must be invoked and the core needs to be regenerated.

Parameter Name	Valid Range	Default	Description
1. N is a number of physically available MAC's.			
COEF_RAM	0-1	0	Fully enumerated filter does not use the parameter
DATA_RAM	0-1	0	Fully enumerated filter does not use the parameter
SAMPLEID	0-1	0	Fully enumerated filter does not use the parameter
ID_WIDTH	1-10	5	Fully enumerated filter does not use the parameter
URAM_MAXDEPTH	4, 8, 16, 32, 64, 128, 256, 512	0	Fully enumerated filter does not use the parameter
L	2-512	2	Fully enumerated filter does not use the parameter
M	2-512	2	Fully enumerated filter does not use the parameter

Ports
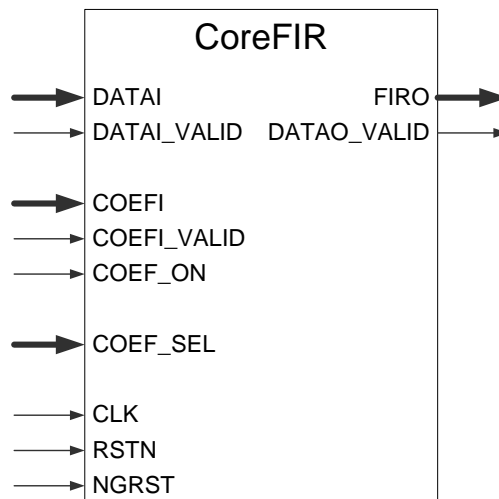
The parallel FIR filter symbol is shown in Figure 12.

Figure 12 · Fully Enumerated Filter I/O Posts

The pinout of Figure 12 is a superset of all possible ports. In every configuration only a subset of these is used.

Table 7 Fully Enumerated Filter In/Out Signals

Signal	In/Out	Port Width Bits	Description
DATAI	In	Input data width, DATA_WIDTH	Input data to be filtered.
DATAI_VALID	In	1	Input data valid. Active high. When the signal is active, the input data sample is loaded into the FIR Filter.
COEFI	In	Coefficient bit width, COEF_WIDTH	Coefficient input. The coefficients are to be loaded sequentially, one by one. The coefficients are loaded in a temporary storage. They replace the current coefficients all at once on COEF_ON signal. This port is enabled only when reloadable coefficient mode is selected.
COEFI_VALID	In	1	Coefficient valid. Active high. When the signal is active, a coefficient is loaded into the MAC FIR filter. This port is enabled only when reloadable coefficient mode is selected.
COEF_ON	In	1	Coefficients on. Active high. Updates the filter coefficients. In Constant Coefficient mode (COEF_TYPE = 0), the signal replaces the existing filter coefficients with the new set of coefficients pointed by the input COEF_SEL. In Reloadable Coefficient mode (COEF_TYPE = 1), the signal makes the filter start using coefficients recently loaded in the auxiliary page.
COEF_SEL	In	4	Coefficient set selector. Identifies the pre-programmed fixed coefficient set to be activated and used as the filter coefficients. This port is enabled only when Multiple Coefficients mode is selected.
CLK	In	1	Clock. Rising edge active. The core master clock.
NGRST	In	1	Asynchronous reset. Active low. Resets all internal registers. The signal is expected to follow the FPGA power-on.
RSTN	In	1	Optional synchronous reset. Active low. Resets all internal registers.
FIRO	Out	DATA_WIDTH + COEF_WIDTH + ceiling($\log_2$TAPS)	Data output. The filtered data appear on this port. It is a full precision output. For example, at 12-bit data, 15-bit coefficients, and 150 taps, the output width = 12 + 15 + ceil(log2150) = 12 + 15 + 8 = 35 bits.
DATAO_VALID	Out	1	Output data valid. Active high. Indicates that a new output data sample is present at the FIRO port

Figure 13 and Figure 14 show examples of the core in Constant and Reloadable Coefficient modes. Figure 15 shows Multiple Coefficients mode.
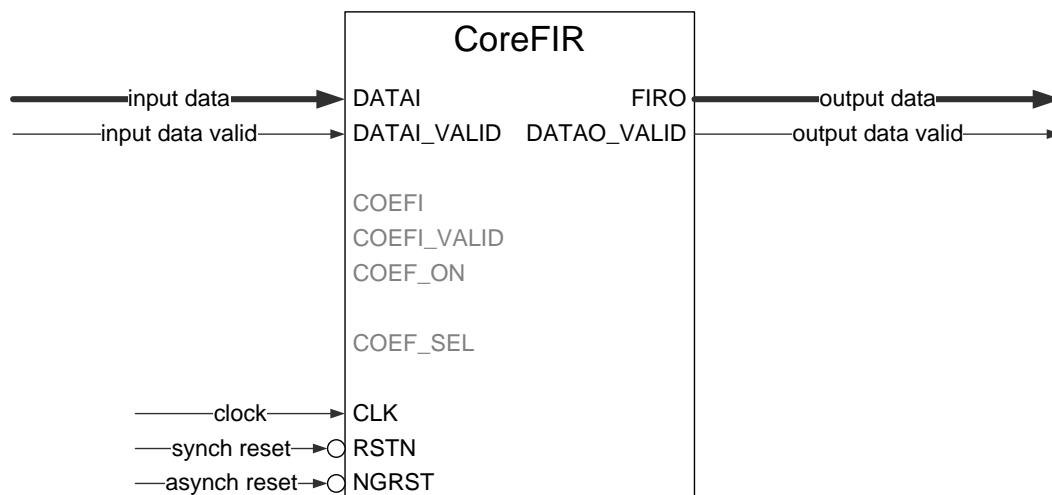

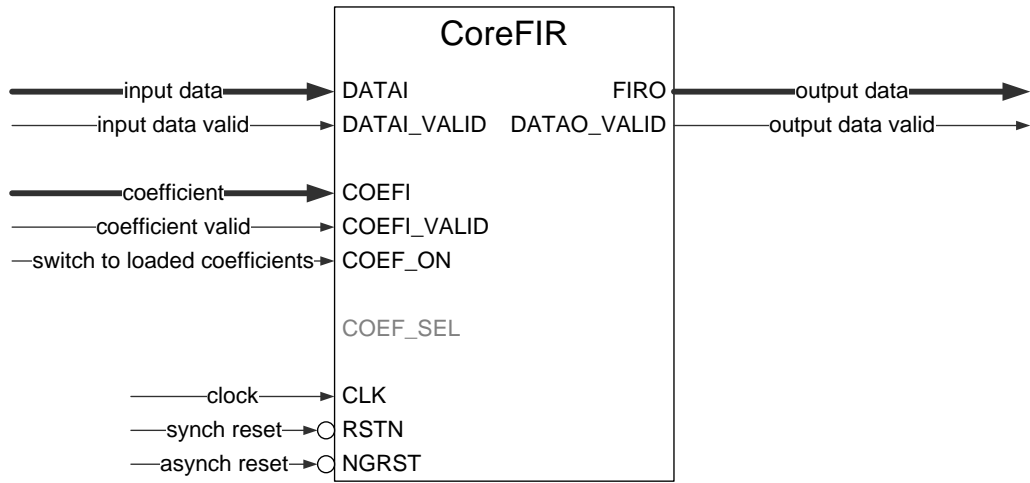**Figure 13** · Parallel Filter Constant Coefficient Configuration

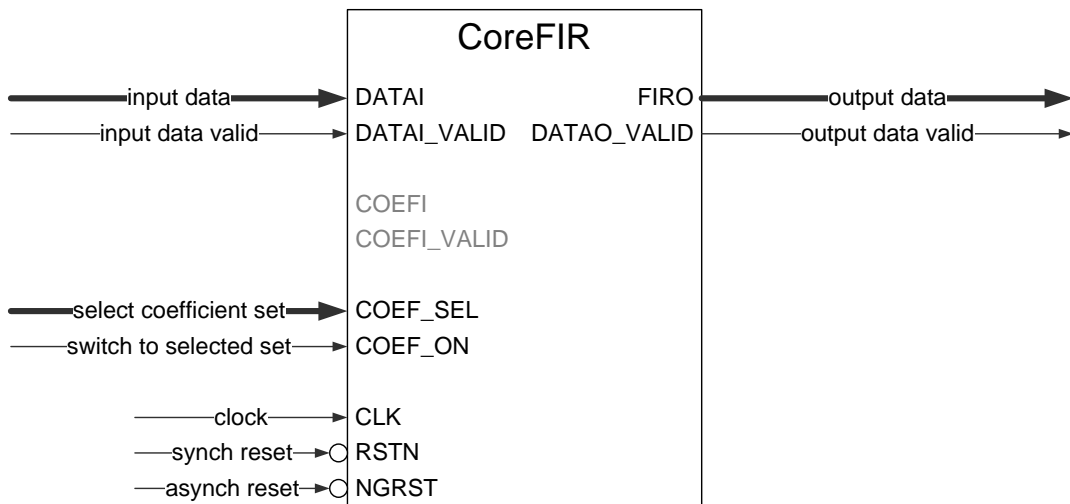**Figure 14** · Parallel Filter Reloadable Coefficient Configuration

Figure 15 · Parallel Filter Multiple Coefficients Configuration

Fully Enumerated Filter Implementation Details

Data Path Bit Width

The core supports signed and unsigned data and coefficients. The core can be configured accordingly. The output data is unsigned if both input data and coefficients are unsigned. Otherwise the output data is signed. Input and output data, as well as the coefficients are integers in two's complement format.

The core supports signed data and coefficients of 2 to 18 bits. For the unsigned data and coefficients, the width is limited to 17 bits. With symmetric filter implementation, the maximum data width is reduced to 17 bits for signed data, and to 16 bits for unsigned data.

Internal filter processing takes place at full precision to reduce truncation/rounding noise and avoid risk of overflow. The filter output data are presented in full precision as well. If the data and coefficient bit widths are DATA_WIDTH and COEF_WIDTH, and the number of filter taps = TAPS, the full precision output bit width is given by EQ 6:

Output Bit Width = DATA_WIDTH + COEF_WIDTH + ceiling ($\log_2$TAPS)

EQ 6

Maximum Number of Taps

As the fully enumerated filter utilizes as many MACs as the number of taps, the maximum number of taps depends on the number of available mathblocks, which in turn is defined by your selection of a particular FPGA device. Symmetric and anti-symmetric filters can implement twice as many taps as the number of physically available MACs. Note, in the odd-symmetry filters, the maximum number of taps is one less. The core configuration window automatically limits the number of taps depending on the device selection.

Multiple Coefficients Mode

In this mode, the filter can switch between *k* pre-configured coefficient sets. Figure 16 shows a single filter tap in this mode. Other taps are organized and behave similarly. The COEF_SEL input controls a MUX, that is, selects one of the coefficient sets, but the coefficients are not propagated to the filter yet. This only takes place when the COEF_ON signal is issued, which loads the newly selected coefficients in the Pipeline registers.

To improve switching characteristics, issue the COEF_ON signal at least four clock cycles later after changing the COEF_SEL signal.
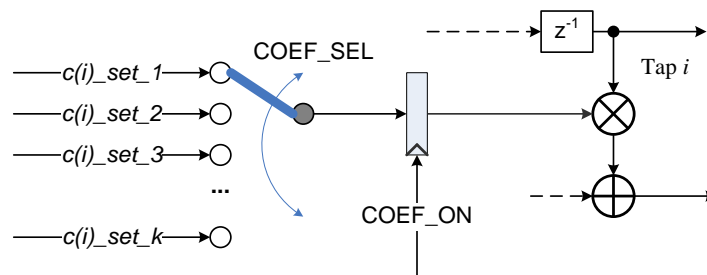

Figure 16 · A Parallel Filter Tap in Multiple Coefficients Mode

Input Registers

The core inputs that present extensive load for the input signal sources are optionally registered, so that the user circuitry does not face extensive fan-out. When the parameter INP_REG is set as 1, the core infers a pair of registers on the following inputs: DATAI, DATAI_VALID, COEFI, COEFI_VALID, COEF_SEL, and COEF_ON. Figure 17 shows an example of the input register inference. The pairs of registers are used to enable the synthesis tool (Synplify) to infer replicated register instances and to contain the user input fanout within the optimal limit. To achieve the best timing results, the global syn_replicate attribute of Synplify should be used.
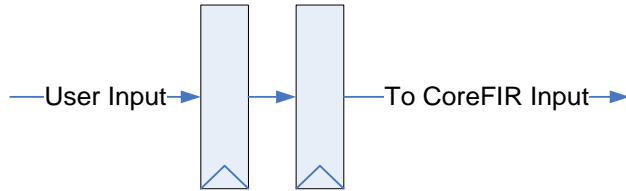

Figure 17 · Optional Input Registers

Inter-Column/Row Pipelines

CoreFIR implements several HW architectures, depending on the user configuration. All of them utilize the transposed architecture shown in Figure 6.

The hard MACs on a chip are organized into physical columns or rows. Within a column or row, the adder chain runs on a dedicated resource thus providing excellent performance characteristics. When a filter utilizes more than one hard MAC physical column or row, the long data path between the columns introduces an extended propagation delay. To eliminate this critical path, CoreFIR automatically infers optimal number of fabric pipeline registers in the inter-column or row sections of the adder chain. Simultaneously, it infers fabric registers in other data paths, which are necessary to preserve the correct functionality of the filter.

Figure 18 on page 23 presents an example of the two fabric inter-column or row registers in the adder chain balanced by a pair of the data bus registers. The added registers are shaded in Figure 18.
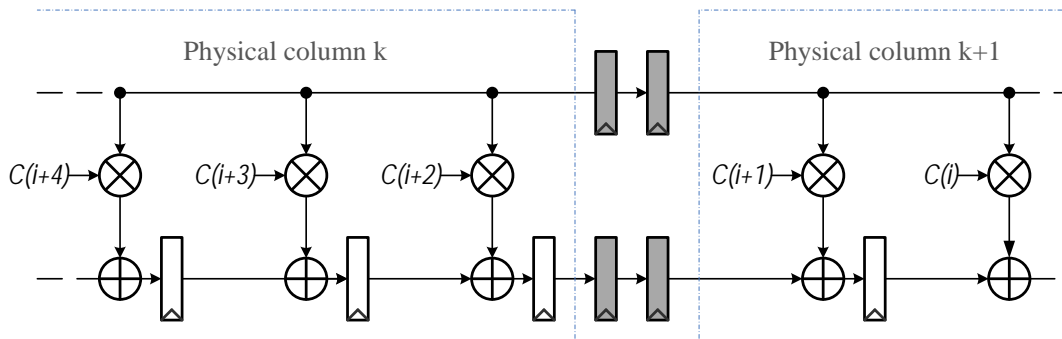

Figure 18 · Transposed Architecture with Inter-Column Fabric Registers

Fully Enumerated Filter Latencies

The filter imposes the following two latency types:

- Pipeline Latency
- Transition Latency, which is proportionate to the number of filter taps

The overall latency is a sum of these two latency types.

Pipeline Latency

This latency accounts for a time period between a valid input and valid output samples. Figure 19 shows the latency when the input registers are disabled. If these are enabled, the pipeline latency adds up to two clock cycles. The DATAO_VALID flag marks the valid output samples.
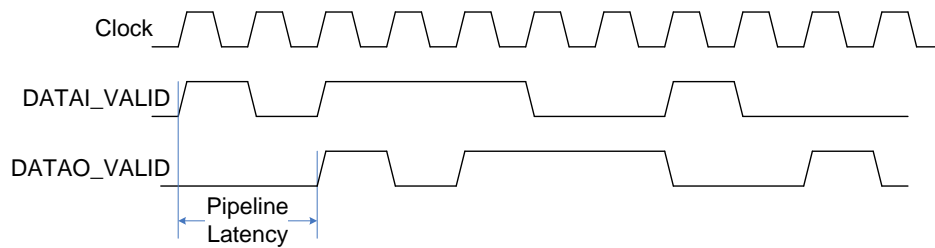

Figure 19 · Pipeline Latency

The flag is not of particular use, when the valid data to be filtered are coming at every clock period (Figure 20). It quickly becomes permanently active.
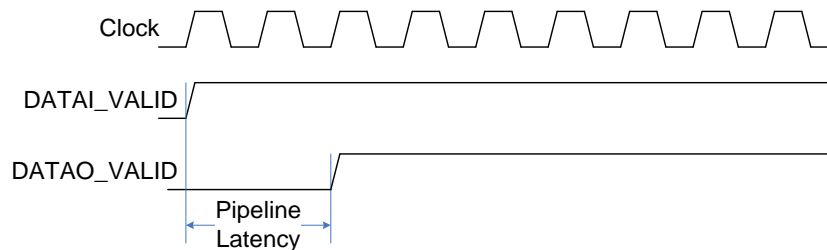

Figure 20 · Pipeline Latency with No Breaks in the Input Data

Transition Latency

The FIR filter starts producing valid output samples after its delay line is filled with input data samples. Until then, there is not enough data to be entered in EQ 1. In other words, after reset, when the filter delay line becomes empty, the first valid output will be available only when the filter receives N data samples. After that, every subsequent input sample will cause the filter to generate a fresh output sample. The reset is not the only event causing the transitional delay. The same applies to the filter coefficient modification or update that takes place on the COEF_ON signal.

Figure 21 shows an example of a 20-tap transposed filter. This filter must collect 20 input samples to satisfy EQ 1. The transition latency depends on the number of taps (TAPS), filter architecture, and symmetry.
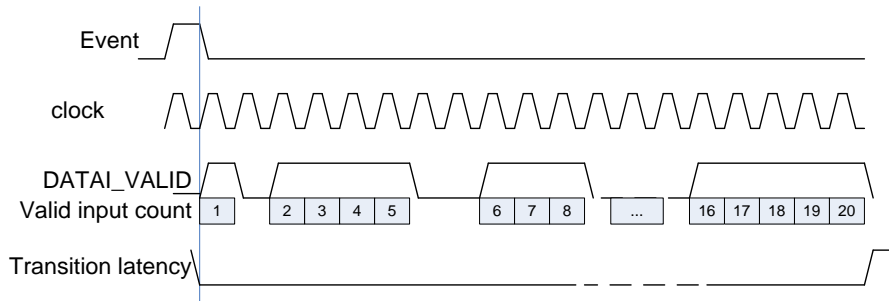

Figure 21 · Transition Latency of a 20-Tap Filter

CoreFIR generates the DATAO_VALID flag that accounts for the pipeline and transition latencies.

In many cases there is no need for the filtered sample recipient to know precisely when the valid filtered data starts. It is sufficient to know that after some initial warm-up time the filter generates the valid data. Table 8 reflects the maximum transition latency values expressed in number of valid input samples required to fill in the filter delay line.

Table 8 Maximum Transition Latency Values

Transposed	Systolic	Symmetric
TAPS + 12	2 * TAPS + 12	ceiling(1.5 * TAPS) + 12

If the precise transition latency is not of concern, it it might be convenient to let the filter run for the time indicated in Table 8 plus the pipeline latency. Every DATAO_VALID flag marks the valid filtered data sample.

Folded Filter

Folded Filter Description

The folded (semi-parallel) single rate CoreFIR type implements the general FIR filter EQ 1. The folded filter utilizes a minimal number of MAC blocks that are sufficient to keep up with an average input sample rate. CoreFIR automatically generates the semi-parallel filter type, if the FPGA clock frequency is at least two times bigger than the input sample frequency. Figure 3 shows the simplified block diagram of the folded filter.

If the clock to sample rate ratio is equal or more than the number N of filter coefficients, the folded filter utilizes a single MAC block.

Folded Filter Interface

Parameters or Generics

Folded CoreFIR RTL has parameters (Verilog) or generics (VHDL), described in Table 9. All the parameters and generics are positive integer types. The table shows a superset of the parameters used by all FIR filter types while indicating which parameters the folded type utilizes.

Table 9 · Semi-Parallel CoreFIR Parameter or Generic Descriptions

Parameter Name	Valid Range	Default	Description
CFG_ARCH	1-4	1	Filter type: 1: Fully Enumerated 2: Folded 3: Polyphase Interpolator 4 : Polyphase Decimator
TAPS	2-1024	16	Number of taps
COEF_TYPE	0-1	0	0: Constant coefficients (including multiple coefficient sets). 1: Reloadable coefficients.
COEF_SETS	1-16	1	1: Single coefficient set. 2-16: Multiple coefficient sets. Valid when constant coefficient type is selected (COEF_TYPE==0).
COEF_SYMM	0-2	0	Folded filter does not use the parameter.
COEF_UNSIGN	0-1	0	0: Signed coefficients. 1: Unsigned coefficients.
COEF_WIDTH	2-18	12	Coefficient bit width. Signed coefficient width ranges from 2 to 18 bits; unsigned from 2 to 17 bits.
DATA_UNSIGN	0-1	0	0: Signed input data. 1: Unsigned input data.
DATA_WIDTH	2-18	12	Data bit width. Signed data width ranges from 2 to 18 bits; unsigned from 2 to 17 bits.
SYSTOLIC	0-1	0	Folded filter does not use the parameter.

Parameter Name	Valid Range	Default	Description
INP_REG	0-1	1	Folded filter does not use the parameter.
FPGA_FAMILY	19, 24, 25	25	The target FPGA family: RTG4 (25), SmartFusion2 (19) or IGLOO2 (24). The parameter is set through the Libero SoC project settings dialog and automatically transfers to the core.
DIE_SIZE	5-50	20	Die size. The parameter is automatically derived from FPGA device name set through Libero project settings dialog. If the Libero device selection changes, you must invoke the core configuration interface and regenerate RTL.
COEF_RAM	0-1	0	0: Build Coefficient ROM out of fabric resources. 1: Build Coefficient ROM using RAM blocks available on a chip.
DATA_RAM	0-1	0	0: Build Data FIFO out of fabric resources. 1: Build Data FIFO using RAM blocks available on a chip.
SAMPLEID	0-1	0	Disable (0) or enable (1) optional support for attaching numerical ID to input and output samples.
ID_WIDTH	1-10	5	Numerical ID bit width. Valid when the ID support is enabled (SAMPLEID = 1).
URAM_MAXDEPTH	4, 8 ,16, 32, 64, 128, 256, 512	0	Micro RAM depth upper limit. The parameter limits the depth of uRAM to be taken for data and/or coefficient storage. Once either or both COEF_RAM or DATA_RAM parameters are set, the core uses on-chip RAM blocks to implement appropriate storage. If the required storage depth does not exceed the URAM_MAXDEPTH value, the Micro RAM is used, otherwise the core utilizes the Large RAM blocks.
L	2-512	2	Folded filter does not use the parameter.
M	2-512	2	Folded filter does not use the parameter.

Other User Parameters

Table 10 lists two more parameters which must be configured for the core. They are not limited to integer numbers, as they are not used by the RTL directly. The user interface converts two user parameters, Input sample frequency and clock frequency, to the single RTL parameter PHY_TAPS.

Table 10 · Other CoreFIR User Parameters

Parameter Name	Valid Range	Default	Description
SAMPLE_RATE	0.001 – 75.0	1.0	Input data sample rate, Msamples/sec. A real number.
CLOCK_RATE	0.01 – 150.0	10.0	Clock frequency, MHz. A real number.

Ports

The semi-parallel FIR filter symbol is shown in Figure 22. Table 11 provides the port definitions for the core.
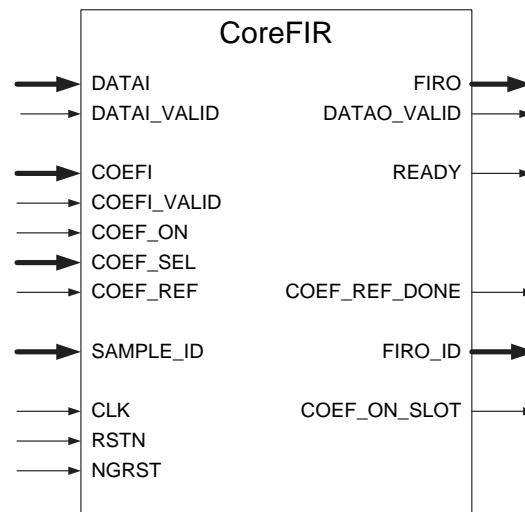

Figure 22 · CoreFIR I/O Ports

The pinout of Figure 22 is a superset of all possible folded filter ports. In every configuration, only a subset of these is used.

Table 11 Folded CoreFIR Filter In/Out Signals

Signal	In/Out	Port width, bits	Description
DATAI	In	Input data width, DATA_WIDTH	Input data to be filtered.
DATAI_VALID	In	1	Input data valid. Active high. When the signal is active, the input data sample is loaded into the FIR Filter. DATAI_VALID should not be active, if READY signal is inactive. Input data samples coming while the READY signal is inactive are ignored.
COEFI	In	Coefficient bit width, COEF_WIDTH	Coefficient input. The coefficients are to be loaded sequentially, one by one. The coefficients are loaded on an Auxiliary memory page. They replace the current coefficients all at once on COEF_ON signal. This port is enabled only when Reloadable coefficient mode is selected.
COEFI_VALID	In	1	Coefficient valid. Active high. When the signal is active, a coefficient is loaded into the FIR Filter. This port is enabled only when Reloadable coefficient mode is selected.
COEF_ON	In	1	Coefficients on. Active high. Swaps Active and Auxiliary pages of the coefficient memory. In Multiple constant set mode (COEF_TYPE==0, COEF_SETS>1), the signal replaces the current coefficient set with the one loaded in the auxiliary page. In Reloadable coefficient mode (COEF_TYPE == 1), the signal makes the filter start using coefficients recently loaded in the auxiliary page. The port is enabled in Multiple set and Reloadable modes. In the Constant coefficient mode (COEF_TYPE==0, COEF_SETS==0), CoreFIR starts using the coefficients shortly after FPGA is powered.

Signal	In/Out	Port width, bits	Description
COEF_REF	In	1	Refresh coefficients. Active high. When asserted must last at least one clock interval. In Constant coefficient mode, the signal initiates refreshing the coefficient set stored on the active coefficient memory page. In Reloadable coefficient mode, the signal initiates reloading coefficients into the auxiliary page of the coefficient storage. In Multiple constant Set mode, the signal starts loading into the Auxiliary page another set of coefficients pointed to by the input COEF_SEL.
COEF_REF_DONE	Out	1	Done refreshing coefficients. Active high. The flag notifies a user that refreshing the constant coefficient set on the active memory page, or reloading coefficients or loading another multiple set into the auxiliary page is completed. Now the auxiliary page is ready to become the active one.
COEF_SEL	In	4	Coefficient set selector. Identifies the pre-programmed fixed coefficient set to be loaded on the auxiliary page. This port is enabled only when Multiple coefficients mode is selected.
CLK	In	1	Clock. Rising edge active. The core master clock.
NGRST	In	1	Asynchronous reset. Active low. Resets the filter and initializes internal coefficient storage. The signal is expected to follow the FPGA power-on.
RSTN	In	1	Optional synchronous reset. Active low. Being asserted along with CLK signal, resets all internal fabric registers.
DATAO	Out	DATA_WIDTH + COEF_WIDTH + ceiling(log2TAPS)	Data output. The filtered data appear on this port. It is a full precision output. For example, at 12-bit data, 15-bit coefficients, and 150 taps, the output width = 12 + 15 + ceil(log2150) = 12 +15 + 8 = 35 bits.
DATAO_VALID	Out	1	Output data valid. Active high. Indicates that a new output data sample is present at the FIRO port.
READY	Out	1	Active high. The core is ready to accept a fresh input data sample.
SAMPLE_ID	In	Numerical ID width ID_WIDTH	Optional numerical ID input. The optional ID is provided by user synchronously with the DATAI and DATAI_VALID signals. The port is enabled when support for the ID is enabled (SAMPLEID = 1).
FIRO_ID	Out	Numerical ID width ID_WIDTH	Optional numerical ID output. CoreFIR accompanies an output sample with the optional numerical ID that matches the corresponding input sample ID. The output is valid when support for the ID is enabled (SAMPLEID = 1) and DATAO_VALID signal is asserted.
COEF_ON_SLOT	Out	1	Optimized time slot for issuing COEF_ON signal. Active high. The core generates this optional signal at the times when asserting the COEF_ON signal does not cause any data or result loss. Once user circuitry is ready to issue the COEF_ON signal, it should wait for the next COEF_ON_SLOT pulse and loop it back to the core as the COEF_ON.

Figure 23 and Figure 24 show examples of the semi-parallel filter active ports in Constant and Reloadable coefficient modes. Figure 25 shows Multiple coefficient set mode.
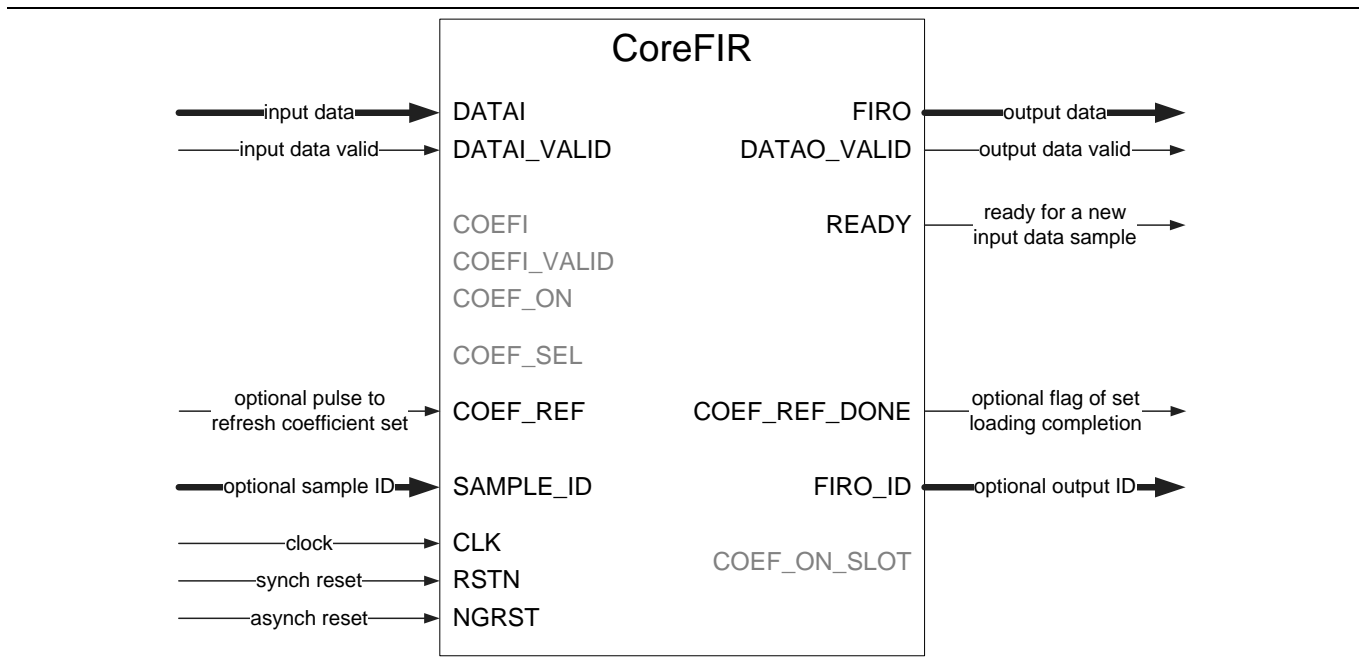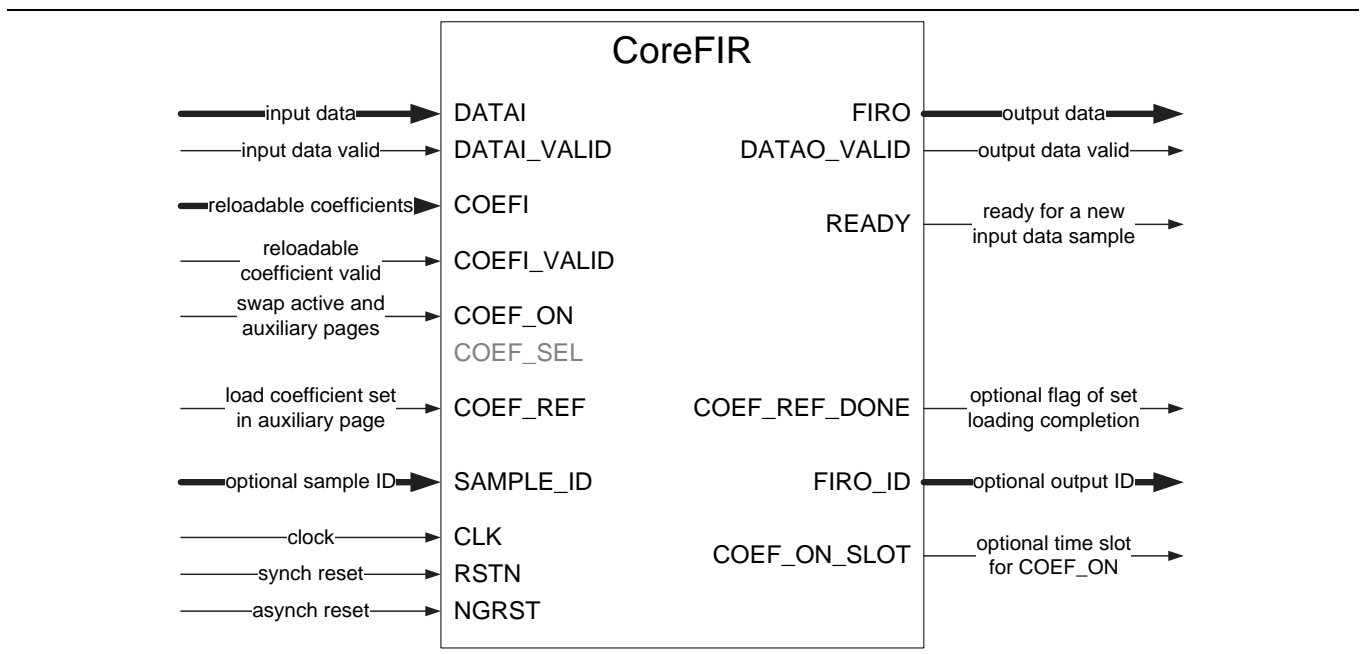

Figure 23 · Folded Constant Coefficient Configuration

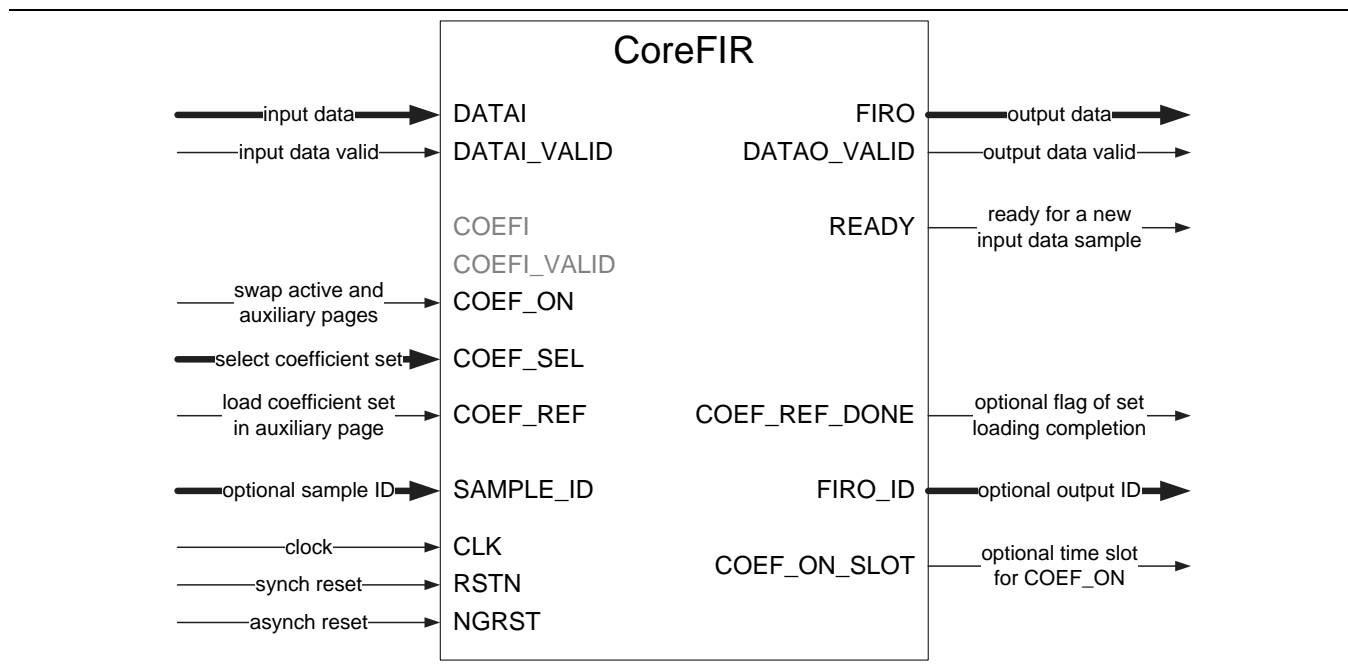**Figure 24** · Folded Reloadable Coefficient Configuration

Figure 25 · Folded Multiple Coefficient Set Configuration

Folded Filter Implementation Details

Data Path Bit Width

The core supports signed and unsigned data and coefficients. The core can be configured accordingly. The output data is unsigned if both input data and coefficients are unsigned. Otherwise the output data is signed. Input and output data, as well as the coefficients, are integers in two's complement format.

The core supports signed data and coefficient of 2 to 18 bits. For the unsigned data and coefficients, the width is limited to 17 bits.

Internal filter processing takes place at full precision to reduce truncation or rounding noise and avoid risk of overflow. The filter output data are presented at full precision as well. If the data and coefficient bit widths are DATA_WIDTH and COEF_WIDTH, and the number of filter taps = TAPS, then the full precision output bit width is given by EQ 7:

$$\text{Output Bit Width} = \text{DATA\_WIDTH} + \text{COEF\_WIDTH} + \text{ceiling}(\log_2 \text{TAPS})$$

EQ 7

Coefficient Modes

Constant Coefficient Mode

In the single set Constant coefficient mode, the coefficients entered at configuration time are copied into the coefficient ROM. This normally happens once, on asynchronous reset signal NGRST, which is expected to follow powering on an FPGA device. A mechanism built into the core runs the process automatically. No action is required.

If necessary, refreshing the contents of the ROM can be done by asserting and deasserting the COEF_REF signal. Then the core launches the copying sequence again. The core generates the optional flag, CORE_REF_DONE, once the initial or secondary copying is completed. The copying takes approximately 4 * TAPS clock intervals. The core keeps the signal DATAO_VALID deasserted while running the copying sequence.

Multiple Constant Coefficient Sets

In this mode the filter can switch between *k* pre-configured coefficient sets. Figure 26 shows the coefficient ROM in this mode. The core maintains two "cache"-style pages where it is convenient to store the current set and the set to be used next. Then the switch from a current set to the next one takes a single clock interval. After the switch, the page that is used to store the former current set gets vacant. This can be used to fill-in with the yet next coefficient set.

CoreFIR automatically configures a coefficient ROM to use dual-page memory in Multiple set or Reloadable mode. The COEF_SEL input controls a MUX that selects one of the constant coefficient sets. On asserting and deasserting the COEF_REF signal, the core starts copying the selected set on the auxiliary page of the ROM. In the meantime, the filter engine keeps using the current coefficient set stored on the active page. Once the copying is completed, the core issues the optional signal COEF_REF_DONE, but the coefficients are not propagated to the filter engine yet. This only takes place when the COEF_ON signal is issued, which swaps active and auxiliary pages.

After initial power-up, the active page does not contain any coefficient set. Therefore, the core does not produce a meaningful result until the auxiliary page is filled with a valid set and becomes the active page after the COEF_ON signal comes in.

When the signal COEF_ON swaps the active and auxiliary pages, generally the PHY_TAPS filtered results immediately following the COEF_ON, are incorrect and required to be discarded. If this is of concern, loop back the COEF_ON_SLOT signal as the COEF_ON.
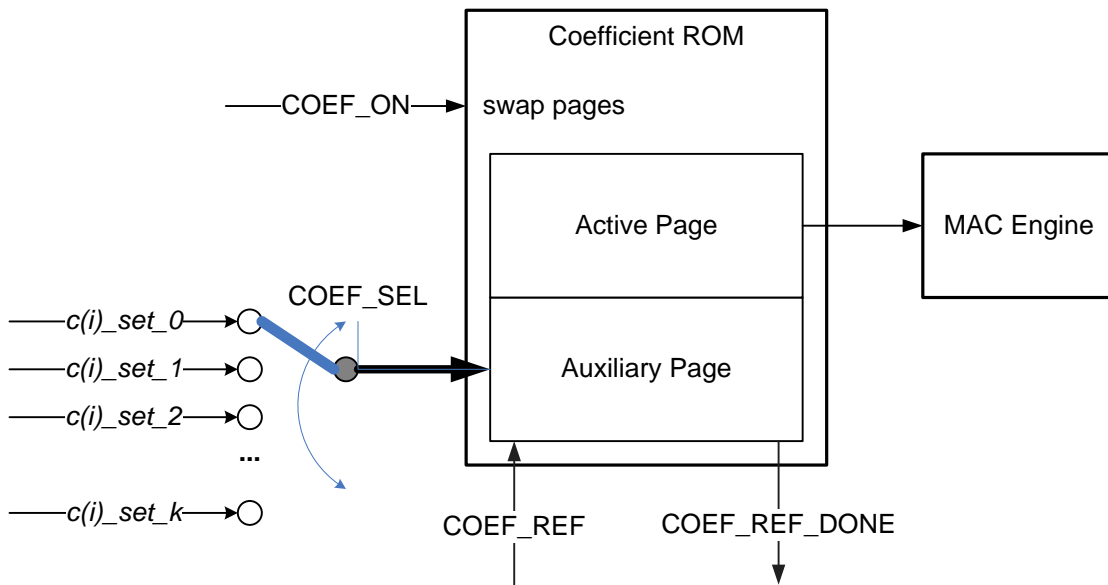

Figure 26 · Two-Page Coefficient Storage in Multiple Set Mode

Reloadable Coefficients

Similar to ultiple constant set mode, the coefficient ROM contains two pages, Active and Auxiliary. While the filter engine keeps using coefficients stored in the active page, a new reloadable set of coefficients can be downloaded by the user circuitry on the auxiliary page. To do so, assert and deassert the COEF_REF signal and supply the new reloadable coefficients and validity bits to the COEFI and COEFI_VALID ports. The coefficients must be supplied in reverse order: the coefficient $c(N-1)$ first, followed by $c(N-2)$, and so on until the last coefficient $c(0)$ is supplied. Once TAPS number of coefficients is loaded, the core issues the optional signal COEF_REF_DONE. For the filter MAC engine to switch to the just loaded coefficients, assert the COEF_ON signal, which swaps the pages.

After initial power-up, the active page does not contain any meaningful coefficients. Therefore, the core does not produce a valid result until the auxiliary page is filled with a valid set of reloadable coefficients and becomes the active page after the COEF_ON signal comes in.

When the signal COEF_ON swaps the active and auxiliary pages, generally the PHY_TAPS filtered results immediately following the COEF_ON are incorrect and need to be discarded. If this is of concern, loop back the COEF_ON_SLOT signal as COEF_ON.

Data Control and Timing

Input Rate Limitations

Semi-parallel architecture expects the filter input sample rate to be a fraction of the clock rate (EQ 2). Once the number of physical MACs PHY_TAPS is defined, the filter can handle the input data rates up to the maximum sample frequency of EQ 3. This does not mean that every sample interval needs to satisfy EQ 3. The core accepts non-periodic input samples coming with instantaneous frequency up to the clock frequency.

To relax requirements for the input sample periodicity, the core features the Data FIFO and the READY signal. A data source can supply data at any frequency as long as the average rate does not exceed the maximum sample frequency. The FIFO collects PHY_TAPS new input samples, which is sufficient to compute PHY_TAPS filtered results[1]. Then it deasserts the READY signal and waits for the MAC engine to start processing to collect data. Depending on the actual input rate the waiting period may vary from a fraction of the input sample interval to several intervals. For example, if the actual input rate never exceeds the sample frequency, the waiting period is less than the input sample period. In this case the READY signal is always active by the time a fresh input sample comes in. In case the data source attempts to supply data, for example at clock frequency, the READY signal is deasserted for significant interval. The data source should only assert DATAI_VALID when the READY signal is asserted.

Consider a 12-tap FIR filter with the clock to sample frequency ratio of 4. According to EQ 2, the semi-parallel filter has PHY_TAPS = 12/4 = 3 physical MACs. Figure 27 shows a case where the input samples are periodic at the fixed timing interval of 4 clocks that satisfies the EQ 3.

Note: The READY signal is always active by the time a fresh input sample comes in and thus, can be neglected in such a case.
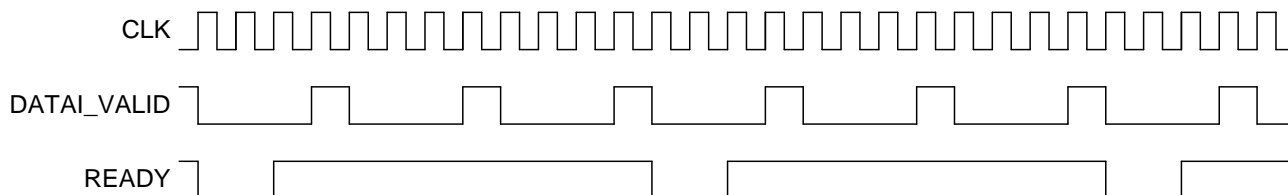

Figure 27 · Fixed Timing Interval Between Input Samples

For the same filter, Figure 28 shows the input samples coming at an instantaneous frequency equal to the clock frequency. The data comes in bursts of PHY_TAPS = 3 samples each. The data source supplies the samples only when the READY signal is Active.

---

[1] *This does not refer to initial warm-up time, which takes more input samples to compute PHY results. Refer to the Warm-Up Time section on page 39 for details.*
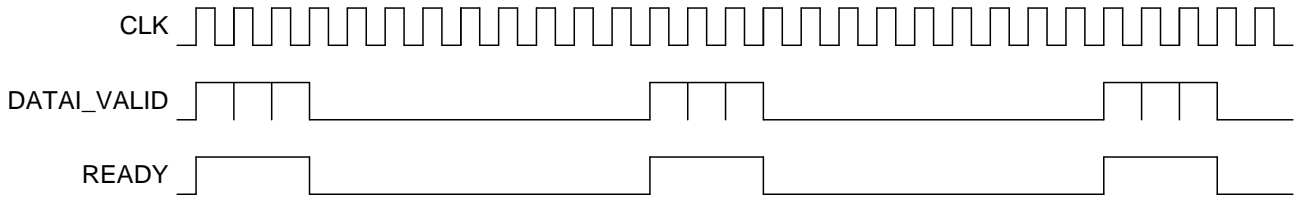
Figure 28 · Bursts of the Input Samples

Input and Output Time Domains

The core accepts input samples synchronized with the core CLK signal. The average data input rate must satisfy EQ 3. The filtered results come out of the FIRO port in bursts of PHY_TAPS samples each, one output sample per clock interval. The signal DATAO_VALID accompanies the valid output samples. Figure 29 shows the filter output bursts for the filter example described in the section "Input Rate Limitations".
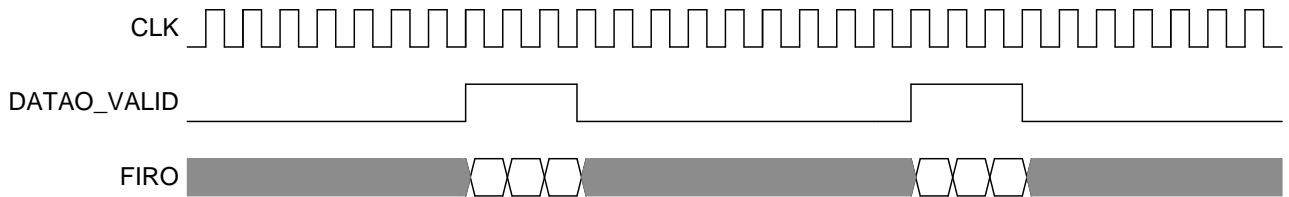

Figure 29 · Filtered Results

Filter Latency

Once the FIFO collected PHY_TAPS new input samples and the MAC engine is ready for processing, the FIFO bursts out all the samples necessary to compute PHY_TAPS filtered results. It takes the MAC engine N + 7 clock intervals of processing time for the filter to start generating PHY_TAPS results, one per clock interval. Refer to the "Input Rate Limitations" section on page 34 as an example of a fixed input rate. The rising edge of the COEF_ON_SLOT output signal marks the beginning of the processing time (Figure 30), and the falling edge of the READY signal marks the moment when FIFO finishes collecting PHY_TAPS fresh data samples. The MAC engine is ready to process fresh PHY_TAPS input samples as soon as the Data FIFO finishes collecting them. The overall latency from the moment when FIFO started collecting the fresh PHY_TAPS data samples to the first result of the output burst is shown in EQ 8.

$$\frac{\text{PHY\_TAPS}}{\text{Sample frequency}} + \frac{N + 7}{\text{Clock frequency}}$$
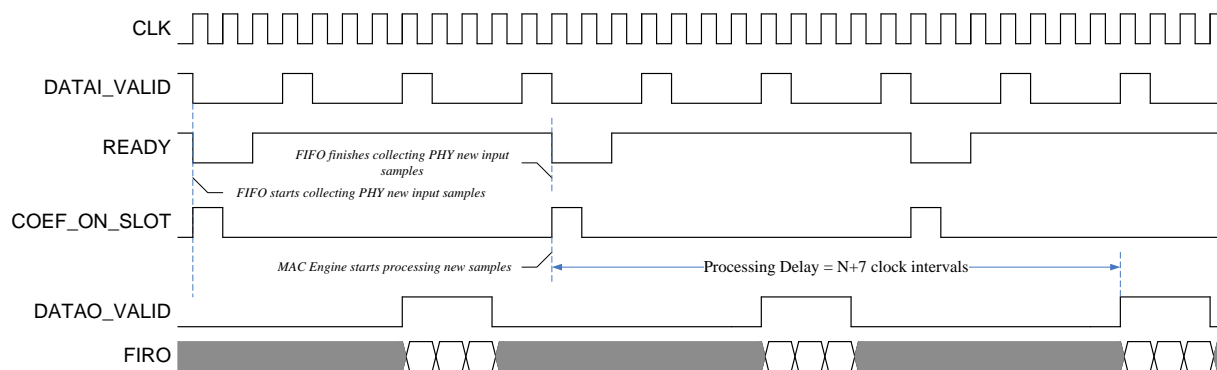
EQ 8

Figure 30 · Filter Latency Example at Fixed Input Rate

Latency calculation for the case when momentary input rate may exceed maximum sample frequency of EQ 3 is more complicated; as by the time the FIFO is filled with new samples, the MAC engine can still be busy processing the previous burst (Figure 31). Therefore, processing starts on the rising edge of a COEF_ON_SLOT pulse, following a FIFO collecting period.
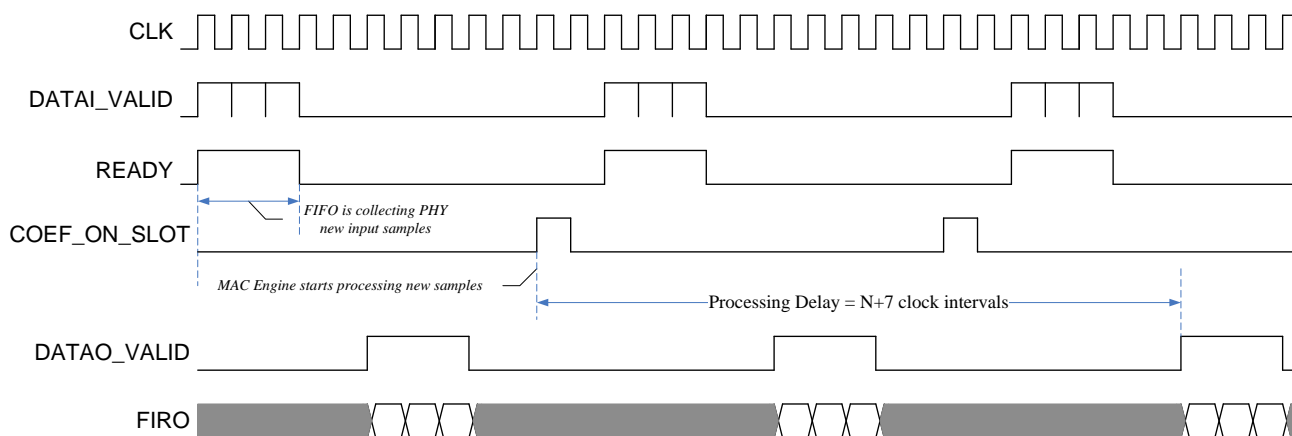

Figure 31 · Filter Latency Example at High Momentary Input Rate

Optional Numeric ID

The core provides a mechanism for attaching numerical IDs to the input and output samples to know which filtered sample corresponds to certain portion of the input data. From EQ 1, it is seen that every filtered output *y(k)* is a result of convolution between the filter coefficients *c(0)* to *c(N-1)* and a range of data samples from *x(k-N+1)* to *x(k)*. The ID helps to associate output index with the input data samples from *k-N+1* to *k*. The output ID carries the input ID attached to the first input sample of the range, namely k-N+1 (to make output indices match the formula of the EQ 1, N-1 must be added to the actual output index). A user is required to supply the ID to the SAMPLE_ID port simultaneously with the DATAI_VALID and DATAI signals (Figure 32).
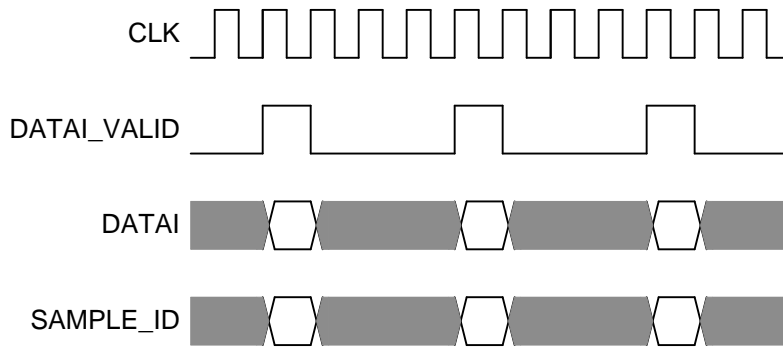
Figure 32 · SAMPLE_ID Timing Diagram

The ID must be generated by a regular incremental binary counter. The bit width of the ID is configurable. With the 5-bit ID, consider the example in the Input Rate Limitations section on page 34. Figure 33 shows the SAMPLE_ID attached to the input data and the output FIRO_ID. The IDs are shown in decimal format for convenience. The output sample with the ID 16 is the filtered result produced by the input samples from 16 to 27; the output ID 17 is the filtered input sequence from 17 to 28, etc. Due to the limited bit width, the ID rolls over to 0 when the ID counter overflows. The FIRO_ID 22 marks the filtered output of the input samples 22-31 and 0-1, as shown in Figure 33 on page 37.
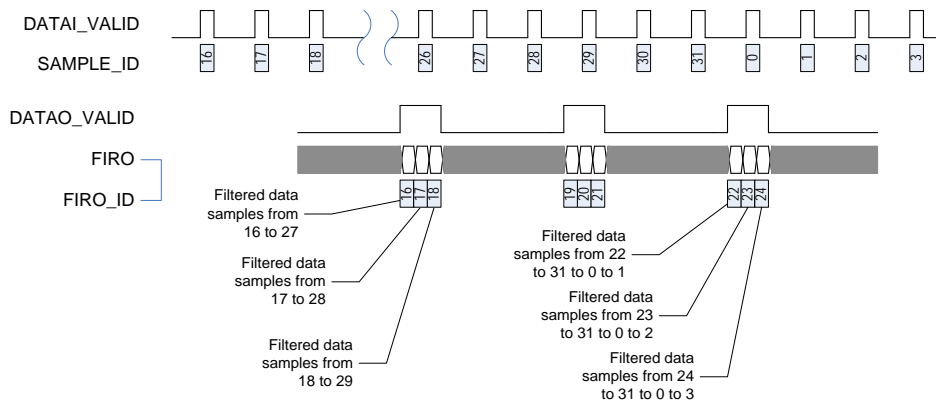

Figure 33 · Numeric ID Use Example

Warm-Up Time

In addition to initial coefficient copying time described in the "Coefficient Modes" section on page 32, the filter takes certain warm-up time. Theoretically, an FIR filter starts producing valid output samples after it collects enough data to be entered in EQ 1 on page 5, N data samples. After initial reset when the filter delay line is empty, the first valid output can possibly be available only after the filter receives N data samples. CoreFIR requires PHY_TAPS more input samples to start generating valid results. It also discards the first result as it is not accurate. The core deasserts the DATAO_VALID signal while initial inaccurate results are generated.

After the warm-up, every subsequent PHY_TAPS input samples causes the filter to generate PHY_TAPS valid output samples.

Data and Coefficient Storages

The core can implement data or coefficient storages, or both using on-chip RAM blocks. Check RAM for coefficients and/or RAM for data boxes on the folded filter UI to direct the core accordingly. Otherwise the core utilizes fabric flip-flops to build the storage.

SmartFusion2 devices provide two types of the on-chip RAM blocks: Micro RAM and Large RAM. The core enables the user to influence selection of a particular RAM type by using the Max micro RAM drop-down menu. CoreFIR will utilize micro RAM if the coefficient or data storage depth does not exceed a value selected form the menu. Otherwise it uses the Large RAM blocks. This happens only if RAM for coefficients and/or RAM for data are checked. A recommendation on setting the Max micro RAM (parameter URAM_MAXDEPTH) value is as follows: if Micro RAM is expected to be a scarce resource in overall design, set a lower value. If the Large RAM is taken by other FPGA design components, select a larger value on the Max micro RAM (uRAM) menu.

Polyphase Interpolation Filter

Description

Polyphase interpolation combines up sampling and subsequent filtering. The result is similar to the common meaning of the interpolation that creates additional output samples in between the original ones. The output sample rate is always an integer multiple of the input rate. The interpolated samples are calculated using the hardware structure depicted in .

Interface

Parameters or Generics

Interpolation CoreFIR RTL has parameters (Verilog) or generics (VHDL), as described in . All the parameters and generics are positive integer types. The table shows the superset of parameters used by all FIR filter types while indicating which parameters the polyphase interpolation type does not utilize.

Table 12 Semi-Parallel CoreFIR Parameter or Generic Descriptions

Parameter Name	Valid Range	Default	Description
CFG_ARCH	1-4	1	Filter type: 1: Fully Enumerated 2: Folded 3: Polyphase Interpolator 4: Polyphase Decimator.
TAPS	2-1024	16	Number of taps.
COEF_TYPE	0-1	0	0: Constant coefficients (including multiple coefficient sets). 1:Reloadable coefficients.
COEF_SETS	1-16	1	1: Single coefficient set. 2: 16 - Multiple coefficient sets. Valid when constant coefficient type is selected (COEF_TYPE==0).
COEF_SYMM	0-2	0	Interpolation filter does not use the parameter.
COEF_UNSIGN	0-1	0	0: Signed coefficients. 1: Unsigned coefficients.
COEF_WIDTH	2-18	12	Coefficient bit width. Signed coefficient width ranges from 2 to 18 bits; unsigned from 2 to 17 bits.
DATA_UNSIGN	0-1	0	0: Signed input data. 1: Unsigned input data.
DATA_WIDTH	2-18	12	Data bit width. Signed data width ranges from 2 to 18 bits; unsigned from 2 to 17 bits.
SYSTOLIC	0-1	0	Interpolation filter does not use the parameter.
INP_REG	0-1	1	Interpolation filter does not use the parameter.
FPGA_FAMILY	19, 24, 25	19	The target FPGA family: SmartFusion2 (19), IGLOO2 (24) or RTG4 (25). The parameter is set through the Libero SoC project settings dialog and

Parameter Name	Valid Range	Default	Description
			automatically transfers to the core.
DIE_SIZE	5-50	20	Die size. The parameter is automatically derived from FPGA device name set through the Libero SoC project settings dialog. If the Libero device selection changes, the core configuration interface must be invoked and the core needs to be regenerated.
COEF_RAM	0-1	0	0: Build Coefficient ROM out of fabric resources. 1: Build Coefficient ROM using RAM blocks available on a chip.
DATA_RAM	0-1	0	Interpolation filter does not use the parameter.
SAMPLEID	0-1	0	Interpolation filter does not use the parameter.
ID_WIDTH	1-10	5	Interpolation filter does not use the parameter.
URAM_MAXDEPTH	4, 8, 16, 32, 64, 128, 256, 512	0	Micro RAM upper limit. Limits the depth of uRAM to be taken for data and/or coefficient storage. Once either one or both COEF_RAM or /and DATA_RAM parameters are set, the core uses on-chip RAM blocks to implement appropriate storage. If the required storage depth does not exceed the URAM_MAXDEPTH value, the uRAM should be used, otherwise the core utilizes LRAM blocks.
L	2-512	2	Interpolation factor L.
M	2-512	2	Interpolation filter does not use the parameter.

Ports

The interpolation FIR filter symbol is shown on Figure 34. Table 13 provides the port definitions for the core.
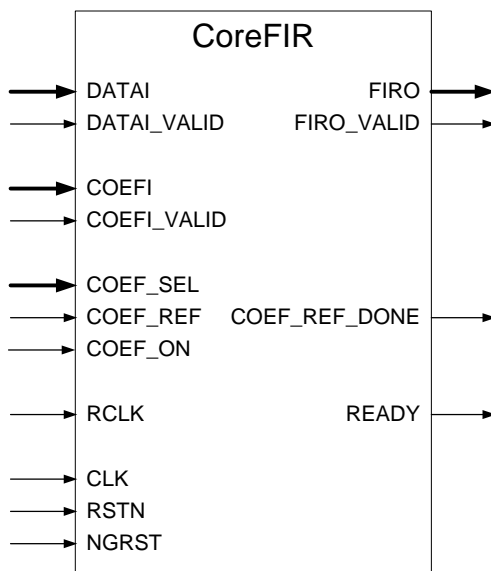

Figure 34 · CoreFIR I/O Ports

The pinout of Figure 34 is a superset of all possible ports. In every configuration only a subset of these is used.

Table 13 FIR In/Out Signals

Signal	In/Out	Port Width, Bits	Description
DATAI	In	Input data width, DATA_WIDTH	Input data to be filtered.
DATAI_VALID	In	1	Input data valid. Active high. When the signal is active, the input data sample is loaded into the FIR filter. DATAI_VALID should not be active, if READY signal is inactive. Input data samples coming while the READY signal is inactive are ignored.
COEFI	In	Coefficient bit width, COEF_WIDTH	Coefficient input. The coefficients are to be loaded sequentially, one by one. This port is enabled only when reloadable coefficient mode is selected.
COEFI_VALID	In	1	Coefficient valid. Active high. When the signal is active, a coefficient is loaded into the FIR filter. This port is enabled only when reloadable coefficient mode is selected.
COEF_REF	In	1	Refresh coefficients. Active high. In Reloadable coefficient mode, the signal initiates reloading coefficients into auxiliary page of the coefficient storage. In Multiple constant set mode, the signal starts loading into the auxiliary page another set of coefficients pointed to by the input COEF_SEL.
COEF_REF_DONE	Out	1	Done refreshing coefficients. Active high. Notifies that reloading coefficients or loading another Multiple set into the auxiliary page is completed. Now the auxiliary page is ready to become the active one.
COEF_SEL	In	4	Coefficient set selector. Identifies the pre-programmed fixed coefficient set to be loaded on the auxiliary page. This port is enabled only when Multiple coefficients mode is selected.
COEF_ON	In	1	Coefficients on. Active high. Swaps active and auxiliary pages of the coefficient memory. In Multiple constant set mode (COEF_TYPE==0, COEF_SETS>1), the signal replaces the current coefficient set with the one loaded in the auxiliary page. In Reloadable coefficient mode (COEF_TYPE == 1), the signal makes the filter start using coefficients recently loaded in the auxiliary page. The port is enabled in Multiple set and Reloadable modes. In Constant coefficient mode (COEF_TYPE==0, COEF_SETS==0), CoreFIR starts using the coefficients shortly after FPGA is powered.
CLK	In	1	Clock. Rising edge active. The core master clock.
NGRST	In	1	Asynchronous reset. Active low. Resets the filter and initializes internal coefficient storage. The signal is expected to follow the FPGA power-on.
RSTN	In	1	Optional synchronous reset. Active low. Being asserted along with CLK signal, resets all internal fabric registers.

Signal	In/Out	Port Width, Bits	Description
FIRO	Out	DATA_WIDTH + COEF_WIDTH + ceiling(log$_2$(TAPS/INTRP_FACTOR))	Interpolated data output. The filtered data appears on this port. It is a full precision output. For example, at 12-bit data, 15-bit coefficients, and 150 taps, L = 10 the output width = 12 + 15 + ceil(log$_2$15) = 12 + 15 + 4 = 31 bits.
DATAO_VALID	Out	1	Output data valid. Active high. Indicates that a new output data sample is present at the FIRO port.
READY	Out	1	Active high. The core is ready to accept a fresh input data sample. Can optionally be used to simplify interface between a data source and the filter.
RCLK	In	1	A downstream device is ready to accept another interpolated sample from the filter. The optional signal helps to lower CoreFIR output rate to match the interpolated sample recipient throughput. In case the recipient is not ready to accept another filtered sample, it deactivates the RCLK signal until it is ready. If the recipient is always ready to accept the filter output sample rate, the signal must be activated. This happens automatically if the RCLK pin is not connected to user circuitry.

Figure 35 and Figure 36 show examples of the core pinout in Constant and Reloadable coefficient modes and Figure 37 shows Multiple coefficient set mode.
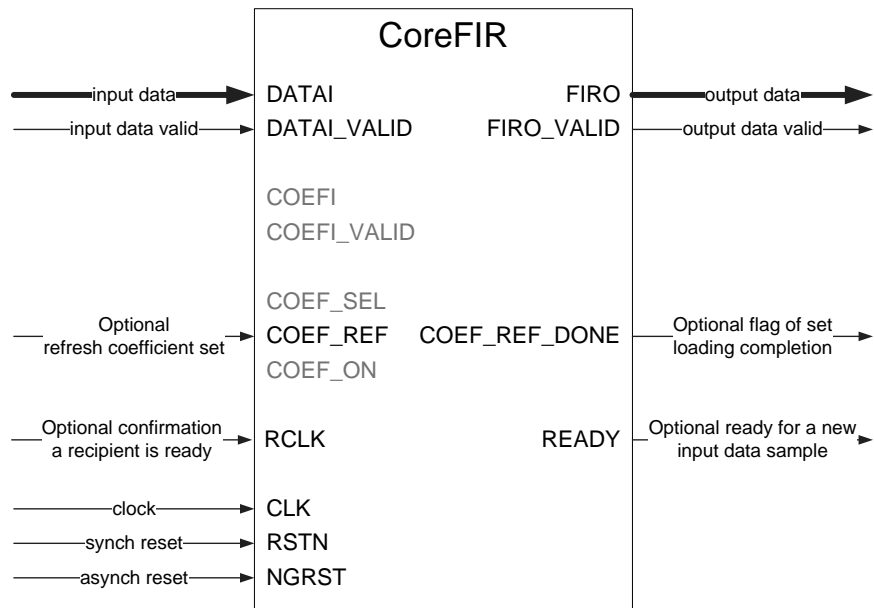

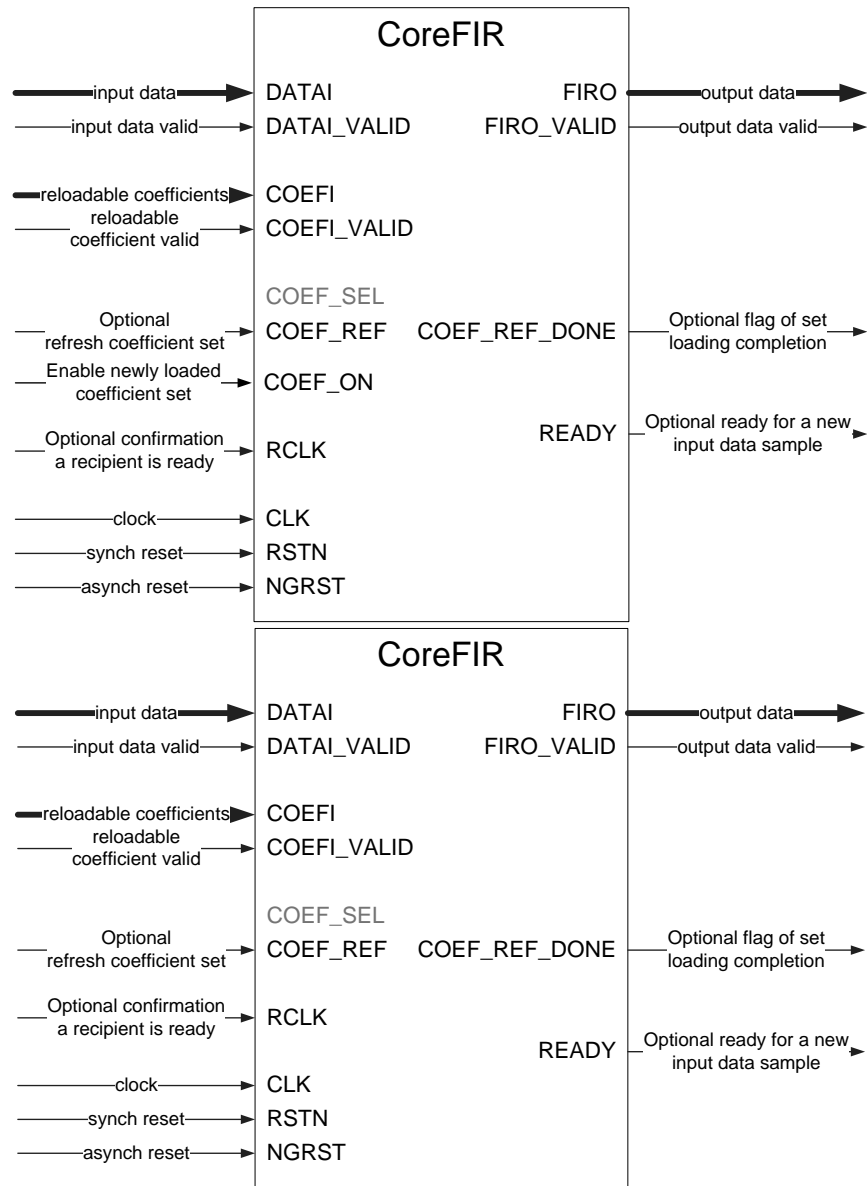**Figure 35** · Constant Coefficient Configuration

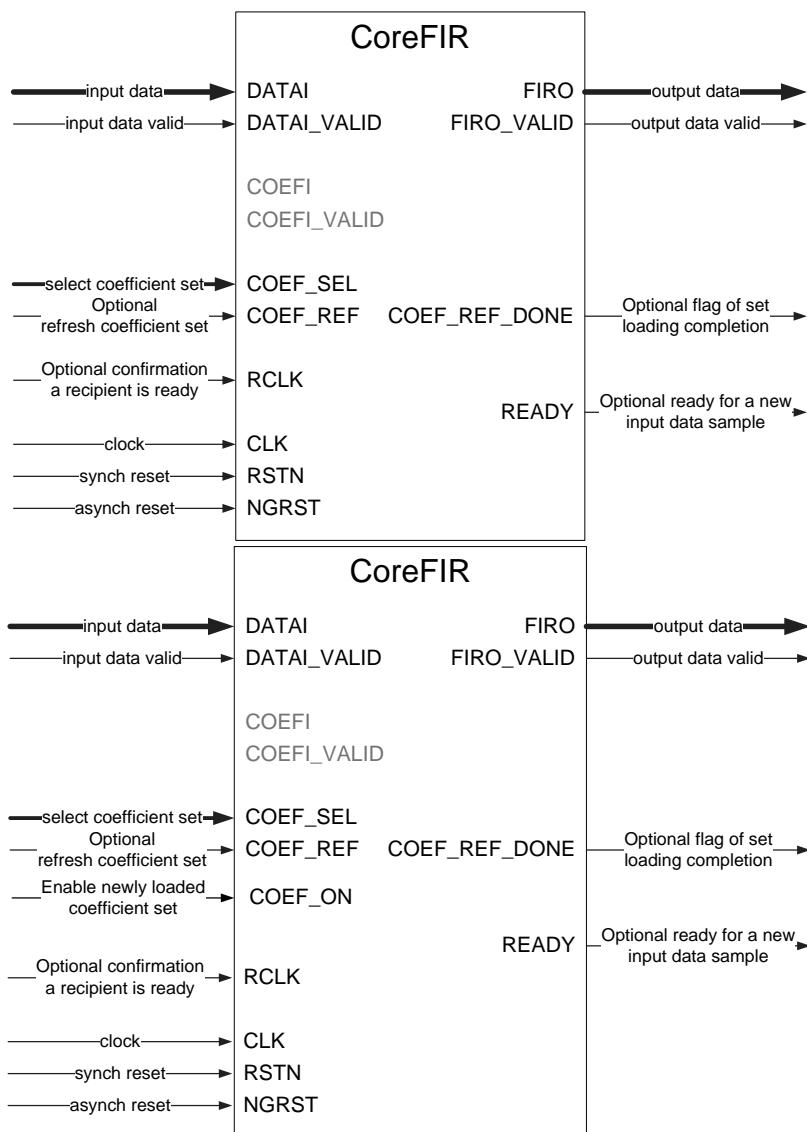**Figure 36** · Reloadable Coefficient Configuration

Figure 37 · Multiple Coefficient Set Configuration

Data Path Bit Width

The core supports signed and unsigned data and coefficients. The core can be configured accordingly. The output data is unsigned, if both input data and coefficients are unsigned. Otherwise the output data is signed. Input and output data, as well as the coefficients, are integers in two's complement format.

The core supports signed data and coefficient of 2 to 18 bits. For the unsigned data and coefficients, the bit width is limited to 17 bits.

Internal filter processing takes place at full precision to reduce truncation or rounding noise and avoid risk of overflow. The filter output data are presented in full precision as well. If the data and coefficient bit widths are DATA_WIDTH and COEF_WIDTH, and the number of filter taps = TAPS, then the full precision output bit width is given by EQ 9.

Output Bit Width = DATA_WIDTH + COEF_WIDTH + ceiling($\log_2$(TAPS/L))

Coefficient Modes

Constant Coefficient Mode

In the single set Constant coefficient mode, the coefficients entered at the configuration time are copied onto the coefficient ROM. This initial copying happens once, on asynchronous reset signal NGRST that is normally asserted upon powering-on the FPGA device. A mechanism built into the core runs the process automatically. No action is required.

It is possible to refresh the ROM contents by asserting the clock-wide COEF_REF signal, if required. Then the core runs the copying sequence again. The core generates the optional flag CORE_REF_DONE, once the initial or subsequent copying is completed. The copying takes approximately 8*TAPS clock intervals. The core keeps the signal DATAO_VALID deasserted until copying is completed.

Multiple Constant Coefficient Sets

In this mode, the filter can switch between *k* pre-configured coefficient sets. Figure 38 shows the coefficient ROM in this mode. The core maintains two cache style pages where it is convenient to store the current set and the set the filter is expected to use next. Then the switch from a current set to the next one takes a few clock intervals. After the switch, the page that used to store the active set is vacant. It can be used for the next coefficient set.

CoreFIR automatically configures a coefficient ROM to use dual-page memory in Multiple set or Reloadable mode. The COEF_SEL 4-bit input controls a MUX that selects one of the constant coefficient sets. On asserting and deasserting the COEF_REF signal, the core starts copying the selected set on the auxiliary page of the ROM. In the meantime, the filter engine keeps using current coefficient set stored on the active page. Once the copying is completed, the core issues the optional signal COEF_REF_DONE but the coefficients are not propagated to the filtered engine yet. This only takes place when the COEF_ON signal is issued, which swaps active and auxiliary pages. The swapping takes up to four clock intervals. The input data samples coming after that time will be properly processed.

After initial power-up, the Active page does not contain any coefficient set. Therefore the core does not produce a meaningful result until the auxiliary page gets filled with a valid set and becomes the active page after the COEF_ON signal comes in. The core downloads the coefficient set 0 (COEF_SEL = 0) on power-on. As a result, the interpolation filter uses the coefficient set 0 upon power-on.
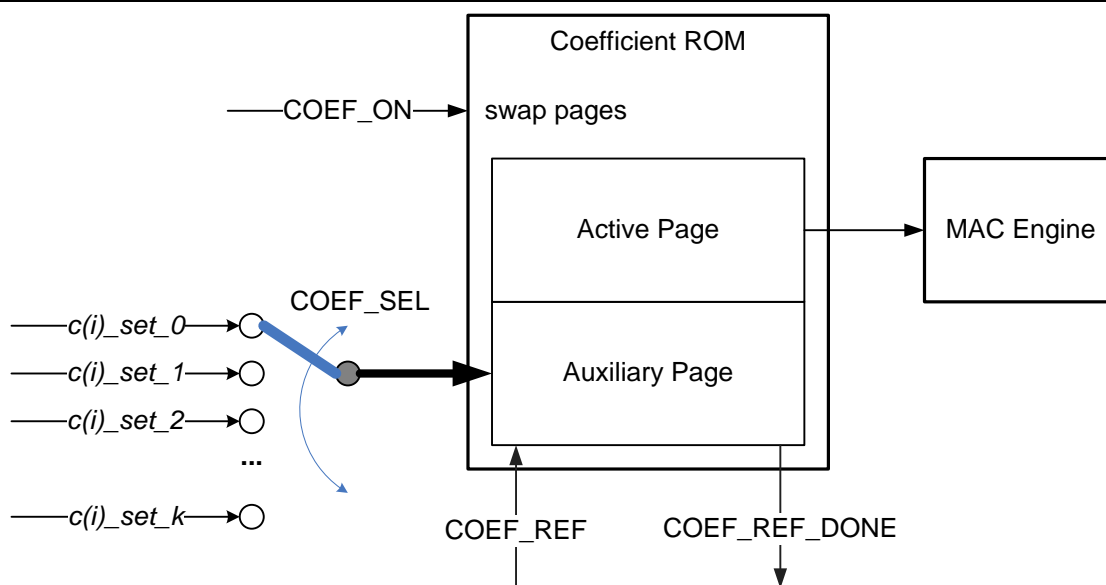

Figure 38 · Two-Page Coefficient Storage in Multiple Set Mode

Reloadable Coefficients

Similarly to Multiple constant set mode, the coefficient ROM contains two pages, Active and Auxiliary. While the filter engine keeps using coefficients stored in the Active page, user circuitry can download a new reloadable set of coefficients on the auxiliary page. To do so, assert and deassert COEF_REF signal and supply the new reloadable coefficients and validity bits to the ports COEFI and COEFI_VALID, respectively. The coefficients must be supplied in the natural order: h_0, h_1, and so on. Once TAPS number of coefficients is loaded, the core issues the optional signal COEF_REF_DONE. For the filter MAC engine to switch to the just loaded coefficients, pulse the COEF_ON signal that swaps the pages. The swapping takes up to four clock intervals. The input data samples coming after that time will be properly processed.

After initial power-up, the active page does not contain any meaningful coefficients. Therefore, the core does not produce a valid result until the auxiliary page gets filled with a valid set of reloadable coefficients and becomes the active page after the COEF_ON signal comes in Timing and Controls.

Data Rate Control

By definition, the input sample rate of the interpolation filter is lower than its output rate. Often the output rate equals the FPGA clock rate used. Provided Fclk is the clock frequency, the input sample frequency is Fclk/L. The relation between input and output samples is simple: upon receiving an input sample, the filter generates L output samples. These appear at the filter FIRO output after certain latency discussed below. Once the filter receives another input sample, it lowers the READY signal to let a signal source know it is going to be busy computing interpolation samples. In L-1 clock cycles, the filter asserts the READY signal back to let the data source know it is ready to receive the next input sample. Figure 39 and Figure 40 show the examples of the READY signal for a filter with interpolation factor L = 3. The filter requires L = 3 clock intervals to output the interpolated samples. In other words, the interpolation cycle here equals three clock cycles.

The data source issues a fresh sample once per four clock intervals that is the data interval is larger than the interpolation cycle factor L, as shown in Figure 39. The signal READY goes low at the next clock cycle after the active DATAI_VALID signal and stays Low for two clock cycles marked busy. Then it goes High, signaling to the data source it is ready to accept another valid data sample. The data source can take as much time as it needs to generate the fresh data sample.
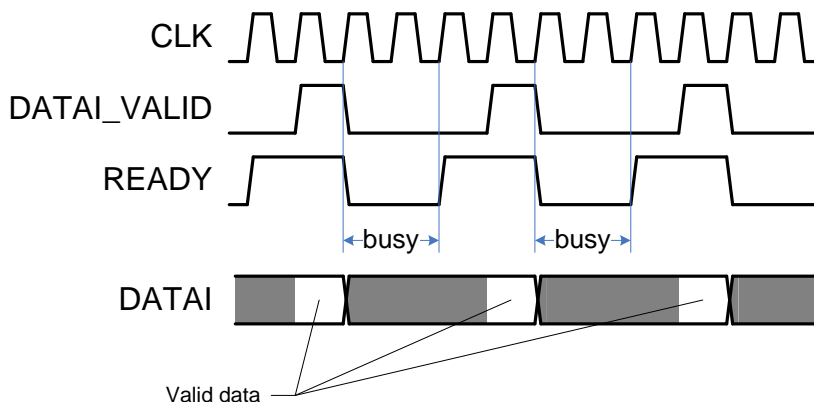

Figure 39 · Input Sample Interval Exceeds Interpolation Cycle

Figure 40 shows an example where the data source is permanently ready after initialization. Once the DATAI_VALID goes High, the READY signal lets the first data sample to get in, and goes low on the front edge of the next clock pulse. It stays low for two clock cycles marked busy and then goes High for one clock interval. In such a simple case, the DATAI_VALID can be connected to VCC, and the data source only needs to refresh data simultaneously with the negative edge of the READY signal.

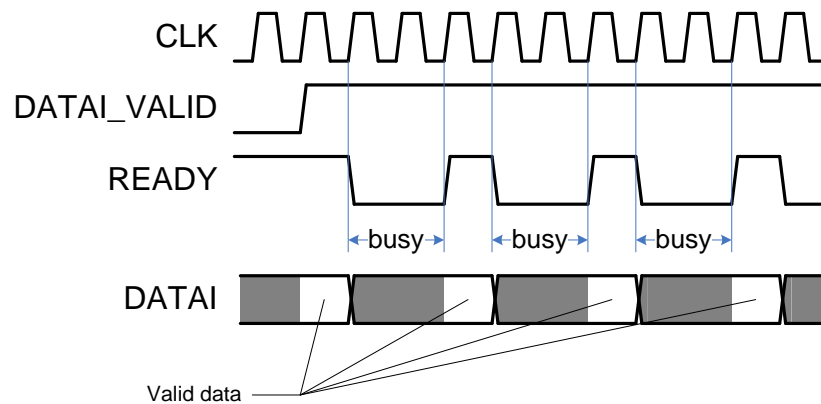The core ignores input data samples coming when the READY signal is Low.

Figure 40 · Data Source Is Always Ready

Output Rate Control

As the output sample rate of the Interpolation filter is higher than the data rate, there might be a need for a slow downstream device to control the filter throughput. You can do this using the RCLK signal by deasserting it for a number of clock cycles. The core expects the RCLK signal to be active for a full clock cycle. If a recipient of the interpolated samples asserts the RCLK every other clock cycle, the sample rate diminishes with a factor of two. The RCLK active every third clock cycle reduces the output sample speed by three times, etc. The core automatically adjusts the READY signal for the data source to drop rate as well.

Figure 41 presents the case depicted on Figure 40 but the output sample rate equals one third of the Figure 40 rate due to lowering the RCLK signal for two clock cycles out of every three. It is seen the READY signal got slower, too.
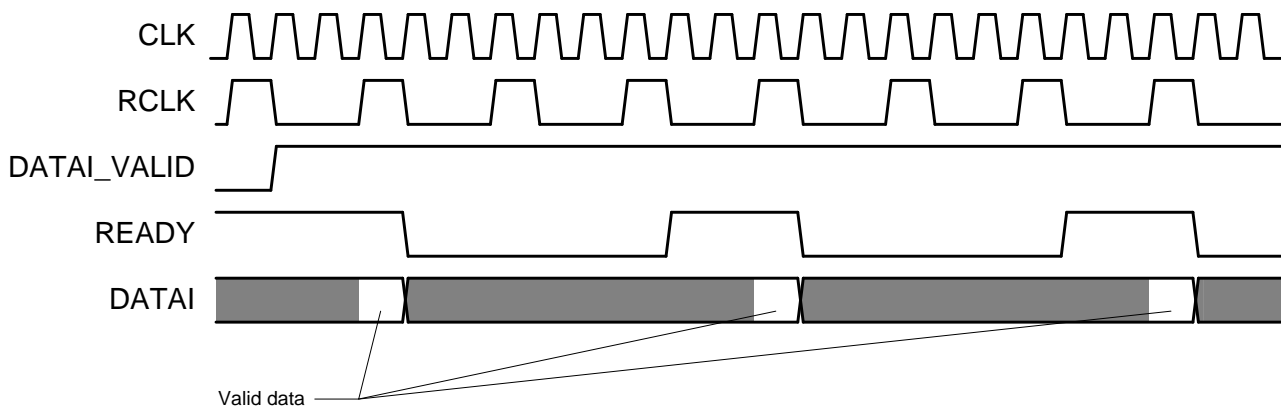

Figure 41 · RCLK Used to Reduce the Output Rate by Three Times

If the downstream device can handle interpolated samples at clock rate, the signal RCLK should be connected to VCC or left unconnected.

Interpolator Latency

The latency primarily depends on the filter size; that is number of physical taps equal TAPS/L. The following formula describes the latency expressed in the number of clock cycles:

$$\text{Latency} = (\text{TAPS/L} + 2) \text{ clock cycles}$$

If RCLK is used to diminish the output sample rate, the formula stays the same but the latency now is measured in RCLK cycles:

$$\text{Latency} = (\text{TAPS/L} + 2) \text{ RCLK cycles}$$

The latency also depends on how many hard MAC rows (columns) the filter implementation takes and a part-dependent number of pipeline registers used to cross inter-row or column space. For convenience the core calculates the overall latency and prints it out during simulation. Look for the line "LATENCY = x CLOCK or RCLK cycles".

Any FIR filter convolves a series of data samples with a set of filter coefficients. Thus, the filter starts generating valid results only after getting enough data samples and coefficients. Upon power-on or swapping coefficient pages, CoreFIR holds the FIRO_VALID signal inactive until it gets TAPS valid data samples. Here "valid data samples" means the data coming in response to the active READY signal.

Polyphase Decimation Filter

Description

The primary reason for using Polyphase decimation is to decrease the sampling rate at the output so that another system operating at a lower sampling rate can use the filtered signal. Another reason is reducing the processing cost. In any case, the decimation filter performs low-pass filtering and down sampling.

Interface

Parameters or Generics

Decimation CoreFIR RTL has parameters (Verilog) or generics (VHDL), which are described in Table 14. All parameters and generics are positive integer types. The table shows the superset of the parameters used by all FIR filter types, while indicating which parameters the Polyphase Decimation type does not utilize.

Table 14 · Semi-Parallel CoreFIR Parameter or Generic Descriptions

Parameter Name	Valid Range	Default	Description
CFG_ARCH	1-4	1	Filter type: 1: Fully Enumerated 2: Folded 3: Polyphase Interpolator 4: Polyphase Decimator
TAPS	2-1024	16	Number of taps.
COEF_TYPE	0-1	0	0: Constant coefficients (including multiple coefficient sets). 1: Reloadable coefficients.
COEF_SETS	1-16	1	1: Single coefficient set. 2-16 : Multiple coefficient sets. Valid when constant coefficient type is selected (COEF_TYPE==0).
COEF_SYMM	0-2	0	0: Not symmetric coefficients. 1: Symmetric coefficients. 2: Anti-symmetric coefficients.
COEF_UNSIGN	0-1	0	0: Signed coefficients. 1: Unsigned coefficients.
COEF_WIDTH	2-18	12	Coefficient bit width. Signed coefficient width ranges from 2 to 18 bits; unsigned from 2 to 17 bits.
DATA_UNSIGN	0-1	0	0: Signed input data. 1: Unsigned input data.
DATA_WIDTH	2-18	12	Data bit width. Signed data width ranges from 2 to 18 bits; unsigned from 2 to 17 bits.
SYSTOLIC	0-1	0	Decimation filter does not use the parameter.
INP_REG	0-1	1	Decimation filter does not use the parameter.

Parameter Name	Valid Range	Default	Description
FPGA_FAMILY	19, 24, 25	19	The target FPGA family: SmartFusion2 (19), IGLOO2 (24) or RTG4 (25). The parameter is set through the Libero SoC project settings dialog and automatically transfers to the core.
DIE_SIZE	5 - 50	20	Die size. The parameter is automatically derived from FPGA device name set through the Libero SoC project settings dialog and automatically transfers to the core. If the Libero SoC device selection changes, the core configuration interface must be invoked and the core regenerated.
COEF_RAM	0-1	0	0: Build Coefficient ROM out of fabric resources. 1: Build Coefficient ROM using RAM blocks available on a chip.
DATA_RAM	0-1	0	0: Build Data RAM out of fabric resources. 1: Build Data RAM using RAM blocks available on a chip.
SAMPLEID	0-1	0	Decimation filter does not use the parameter.
ID_WIDTH	1-10	5	Decimation filter does not use the parameter.
URAM_MAXDEPTH	4, 8, 16, 32, 64, 128, 256, 512	0	Micro RAM upper limit. The parameter limits the depth of uRAM to be taken for data and/or coefficient storage. Once either one or both COEF_RAM or/and DATA_RAM parameters are set, the core will use on-chip RAM blocks to implement appropriate storage. If the required storage depth does not exceed the URAM_MAXDEPTH value, the uRAM is used, otherwise the core utilizes Large RAM blocks.
L	2-512	2	Decimation filter does not use the parameter.
M	2-512	2	Decimation Factor.

Ports

The decimation FIR filter symbol is shown in Figure 42. Table 15 provides the port definitions for the core.
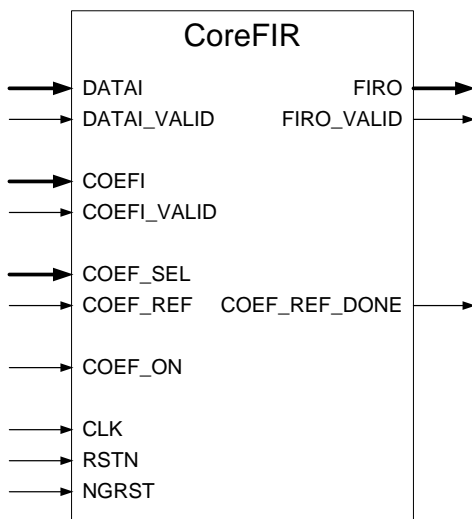

Figure 42 · CoreFIR I/O Ports

The pinout of Figure 42 is a superset of all possible ports. In every configuration only a subset of these is used.

Table 15 CoreFIR I/O Signals

Signal	In/Out	Port width expressed in Bits	Description
DATAI	In	Input data width, DATA_WIDTH	Input data to be filtered.
DATAI_VALID	In	1	Input data valid. Active high. When the signal is active, the input data sample is loaded into the FIR filter. DATAI_VALID should not be active, if READY signal is inactive. Input data samples coming while the READY signal is inactive are ignored.
COEFI	In	Coefficient bit width, COEF_WIDTH	Coefficient input. The coefficients are to be loaded sequentially, one by one. This port is enabled only when Reloadable coefficient mode is selected.
COEFI_VALID	In	1	Coefficient valid. Active high. When the signal is active, a coefficient is loaded into the FIR Filter. This port is enabled only when Reloadable coefficient mode is selected.
COEF_REF	In	1	Refresh coefficients. Active high. In Reloadable coefficient mode, the signal initiates reloading coefficients into Auxiliary page of the coefficient storage. In Multiple constant set mode, the signal starts loading into the Auxiliary page another set of coefficients pointed to by the input COEF_SEL.
COEF_REF_DONE	Out	1	Done refreshing coefficients. Active high. Notifies that reloading coefficients or loading another multiple set into the auxiliary page is completed. Now the auxiliary page is ready to become the active one.
COEF_SEL	In	4	Coefficient set selector. Identifies the pre-programmed fixed coefficient set to be loaded on the Auxiliary page. This port is enabled only when Multiple coefficients mode is selected.
CLK	In	1	Clock. Rising edge active. The core master clock.
NGRST	In	1	Asynchronous reset. Active low. Resets all internal fabric registers. The signal is expected to follow the FPGA power-on.
RSTN	In	1	Optional synchronous reset. Active low. Being asserted along with CLK signal, resets all internal fabric registers.
FIRO	Out	DATA_WIDTH + COEF_WIDTH + ceiling($\log_2$TAPS)	Data output. The filtered data appear on this port. It is a full precision output. For example, at 12-bit data, 15-bit coefficients, and 150 taps, M = 10 the output width = 12 + 15 + ceil($\log_2$150) = 12 + 15 + 8 = 31 bits.
FIRO_VALID	Out	1	Output data valid. Active high. Indicates that a new output data sample is present at the FIRO port.
COEF_ON	In	1	Coefficients on. Active high. Swaps active and auxiliary pages of the coefficient memory. In Multiple constant set mode (COEF_TYPE==0, COEF_SETS>1), the signal replaces the current coefficient set with the one loaded in the Auxiliary page. In Reloadable coefficient mode (COEF_TYPE == 1) the signal makes the filter start using coefficients recently loaded in the auxiliary page. The port is enabled in Multiple set and Reloadable modes. In Constant coefficient mode (COEF_TYPE==0, COEF_SETS==0), CoreFIR starts using the coefficients shortly after the FPGA is powered.

Figure 43 and Figure 44 show examples of the core pinout in Constant and Reloadable coefficient modes and Figure 45 shows Multiple coefficient set mode.
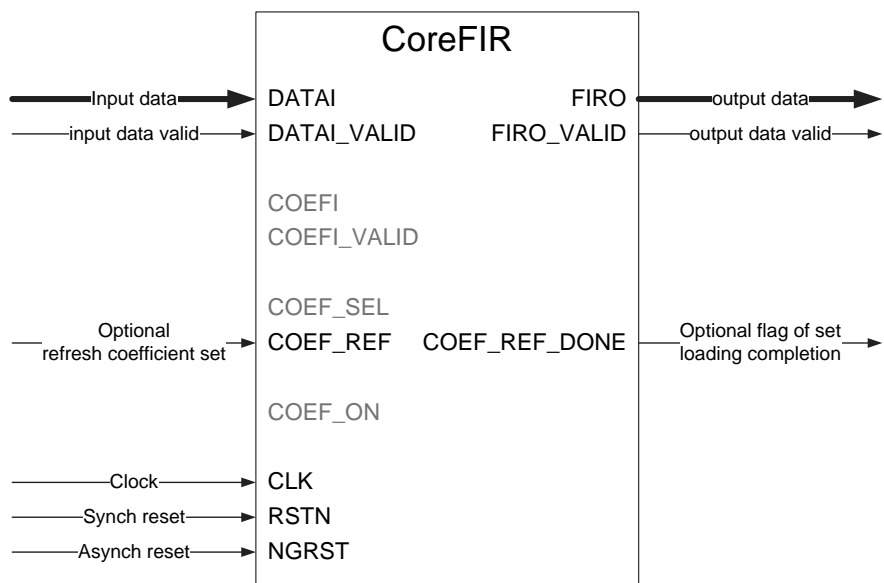

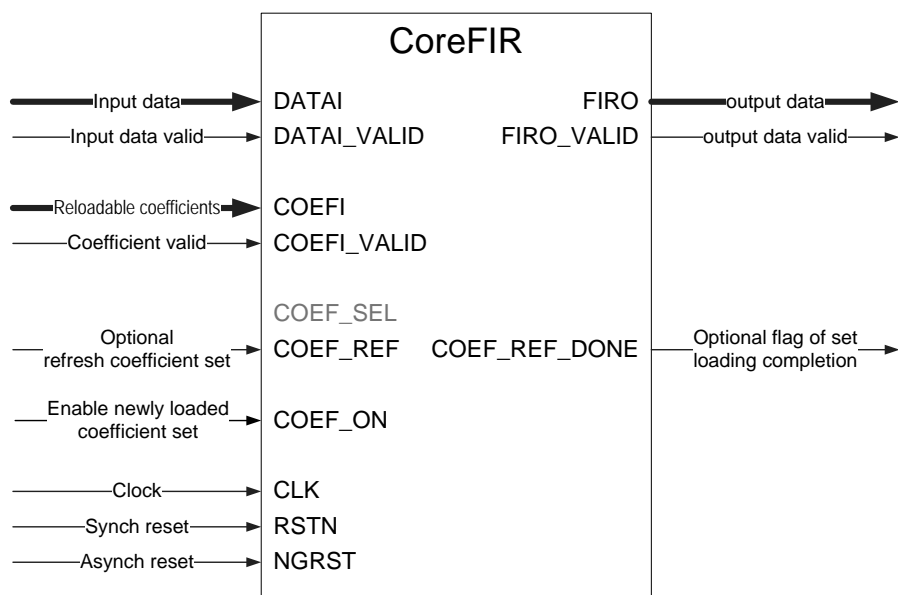**Figure 43 ·** Constant Coefficient Configuration

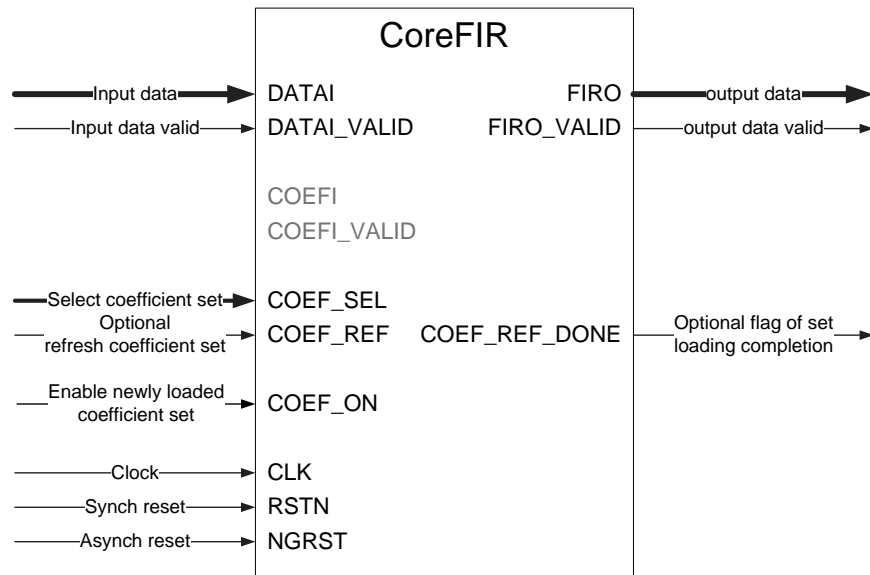**Figure 44 ·** Reloadable Coefficient Configuration

Figure 45 · Multiple Coefficient Set Configuration

Data Path Bit Width

The core supports signed and unsigned data and coefficients. The core can be configured accordingly. The output data is unsigned, if both input data and coefficients are unsigned. Otherwise the output data is signed. Input and output data, as well as the coefficients, are integers in two's complement format.

The core supports signed data and coefficient of 2 to 18 bits. For the unsigned data and coefficients, the width is limited to 17 bits. With symmetric filter implementation, the maximum data width is reduced to 17 bits for signed data, and 16 bits for unsigned data.

Internal filter processing takes place at full precision to reduce truncation or rounding noise and avoid risk of overflow. The filter output data are presented in full precision as well. If the data and coefficient bit widths are DATA_WIDTH and COEF_WIDTH, and the number of filter taps = TAPS, then the full precision output bit width is given by EQ 10:

$$\text{Output Bit Width} = \text{DATA\_WIDTH} + \text{COEF\_WIDTH} + \text{ceiling}(\log_2 \text{TAPS})$$

EQ 10

Coefficient Modes

Constant Coefficient Mode

In single set Constant coefficient mode, the coefficients entered at configuration time are copied into the coefficient ROM. The initial copying happens once on asynchronous reset signal NGRST, which normally is asserted upon powering on the FPGA device. A mechanism built into the core runs the process automatically. No action is required.

Assert the COEF_REF signal to refresh the contents of the ROM, if required. The core runs the copying sequence again. The core generates the optional flag, CORE_REF_DONE, once the initial or subsequent copying is completed. The copying takes approximately 8 * TAPS clock intervals. The core keeps the signal DATAO_VALID deasserted until copying is completed.

Multiple Constant Coefficient Sets

In this mode, the filter can switch between *k* pre-configured coefficient sets. Figure 46 shows the coefficient ROM in this mode. The core maintains two cache-style pages where it is convenient to store the current set and the set that is expected to be used next. Then the switch from the current to the next set takes a single clock interval. After the switch, the page that used to store the active set is vacant. It can be used for the yet next coefficient set.

CoreFIR automatically configures a coefficient ROM to use dual-page memory in Multiple set or Reloadable mode. The COEF_SEL 4-bit input controls a MUX that selects one of the constant coefficient sets. On asserting and deasserting the COEF_REF signal, the core starts copying the selected set on the Auxiliary page of the ROM. In the meantime, the filter engine keeps using current coefficient set stored on the Active page. Once the copying is completed, the core issues the optional signal COEF_REF_DONE but the coefficients are not propagated to the filter engine yet. This only takes place when the COEF_ON signal is issued, which swaps Active and Auxiliary pages. The swapping takes up to four clock intervals. The input data samples coming after that time will be properly processed.

After initial power-up, the active page does not contain any coefficient set. Therefore, the core does not produce a meaningful result until the auxiliary page gets filled with a valid set and becomes the active page after the COEF_ON signal comes in. The core always downloads the coefficient set 0 (COEF_SEL = 0) on power-on. As a result, the filter uses the coefficient set 0 upon power-on.
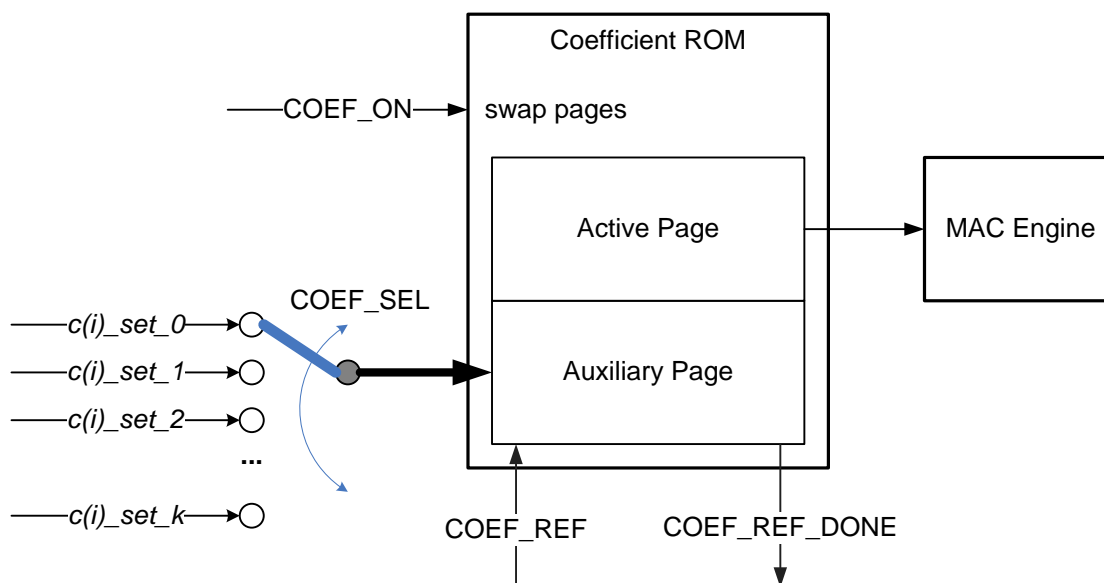

Figure 46 · Coefficient Storage in Multiple Set Mode

Reloadable Coefficients

Similar to Multiple constant set mode, the coefficient ROM contains two pages, Active and Auxiliary. While the filter engine keeps using coefficients stored in the active page, user circuitry can download a new reloadable set of coefficients on the auxiliary page. To do so, COEF_REF signal should be asserted and deasserted and supply the new reloadable coefficients and validity bits to the ports COEFI and COEFI_VALID, respectively. The coefficients must be supplied in natural order: h_0, h_1, and so on. Once TAPS number of coefficients is loaded, the core issues the optional signal COEF_REF_DONE. For the filter MAC engine to switch to the just loaded coefficients, pulse the COEF_ON signal that swaps the pages.

After initial power-up, the active page does not contain any meaningful coefficients. Therefore, the core does not produce a valid result until the auxiliary page is filled with a valid set of reloadable coefficients and becomes the active page after the COEF_ON signal comes in.

Filter Latency

The input data samples can come at clock or lower rate. Every input sample is accompanied by the DATAI_VALID signal.

The relation between input data samples and output decimated samples is simple: upon receiving M data samples, the filter generates an output sample. In other words the filter must obtain M fresh input samples in order to generate another output sample. The latency is defined here as the delay between the last sample of the group of M data samples, and the output sample, which was calculated based on the TAPS input samples including the group. The latency is measured in input data intervals whether those are coming at clock or lower rate, and in clock cycles:

$$\text{Latency} = (\text{TAPS}/M - 1)\ \text{DATAI\_VALID intervals} + 9\ \text{clock cycles}$$

EQ 11

When the input samples are coming at clock rate, EQ 11 simplifies:

$$\text{Latency} = (\text{TAPS}/M + 8)\ \text{clock cycles}$$

The latency also depends on how many hard MAC rows (columns) the filter implementation takes and a part-dependent number of pipeline registers used to cross inter-row or column space. For convenience, the core calculates the overall latency and prints it out during simulation. Look for the line "LATENCY = x DATAI_VALID intervals + 8 CLK intervals".

Any FIR filter convolves a series of data samples with a set of filter coefficients. Thus, the filter starts generating valid results only after getting enough data samples and coefficients. Upon power-on or swapping coefficient pages, CoreFIR holds the FIRO_VALID signal inactive until it receives TAPS data samples.

Coefficient Specification

The core enables specification of the FIR filter constant coefficients and constant coefficient sets as an ASCII text file (* .txt). Create the coefficient file using a text editor. Figure 47 and Figure 48 show the file formats for the multiple and single constant coefficient sets. N and m signify numbers of filter taps and coefficient sets, respectively. Coefficient values must be entered as integer numbers. For a symmetric or anti-symmetric filter, only half of the coefficients must be listed in the file (applies to the Fully Enumerated type only).

```
coefficient_set_1
C(1)(0)
C(1)(1)
...
C(1)(N-1)

coefficient_set_2
C(2)(0)
C(2)(1)
...
C(2)(N-1)


...
coefficient_set_m
C(m)(0)
C(m)(1)
...
C(m)(N-1)
```

Figure 47 · Multiple Coefficient Sets File Format

```
coefficient_set_1
c(0)
c(1)
...
c(N-1)
```

Figure 48 Constant Coefficient File Format

The file format is as follows:

1. The filter coefficients can be presented in decimal, hexadecimal, or binary formats. In decimal format (radix-10) negative values are accepted; for example, −25. In hexadecimal and binary formats (radix-16, radix-2), present negative values in two's complement format; for example, 7-bit −25 = 0 x 67 = 110 0111.
2. Only one coefficient value per line is permitted.

3. An extra empty line must be placed after the last coefficient of the last set.
4. The file must match the core parameters entered through the GUI:
   - Coefficient bit width (COEF_WIDTH)
   - Number of coefficient sets (COEF_SETS)
   - Symmetry of the filter (COEF_SYMM) – applies to the Fully Enumerated type only
   - Coefficient radix: decimal, hexadecimal, or binary

A few examples of the filter coefficient file are given in Figure 49, Figure 50, and Figure 54.

```
coefficient_set_1
5
6
10
25
63
-1
-11
-32
-63
```

Figure 49 · Coefficient File Example – 9 Taps, 7 Bits, Radix-10

Figure 49 shows a non-symmetric 9-tap 7-bit constant coefficient filter. Coefficients are entered using decimal (Radix-10) entries.

```
coefficient_set_1        coefficient_set_1
5                        0000101
6                        0000110
A                        0001010
19                       0011001
3F                       0111111
7F                       1111111
75                       1110101
60                       1100000
41                       1000001
```

Figure 50 · Coefficient File Examples – 9 Taps, 7 Bits, Radix-16, and Radix-2

Figure 50 shows the same set of constant coefficients in hexadecimal and binary formats. All coefficient files shown in Figure 49 and Figure 50 will produce the same filter. The filter impulse response is shown in Figure 51.

If the COEF_SYMM core parameter is set to generate a symmetric or anti-symmetric filter, any of the coefficient files of Figure 49 and Figure 50 can identify such filters. The impulse response examples of the symmetric 17-tap filter and anti-symmetric 18-tap filter are depicted in Figure 52 and Figure 53, respectively. This applies to the Fully Enumerated type only.
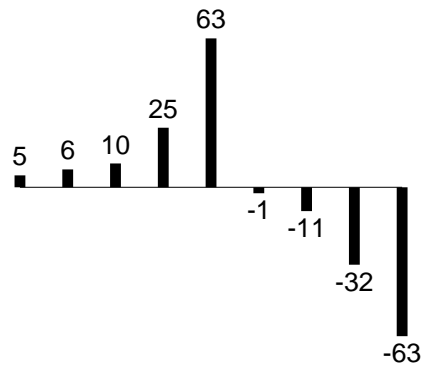
Figure 51 · Constant Coefficient Filter Impulse Response
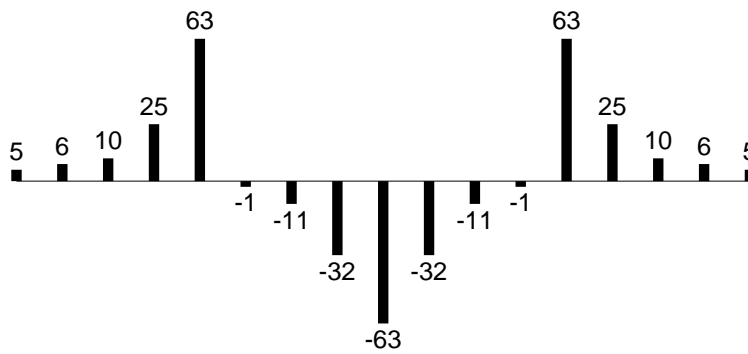
Figure 52 · Symmetric Constant Coefficient Filter Impulse Response (applies to the Fully Enumerated type only)
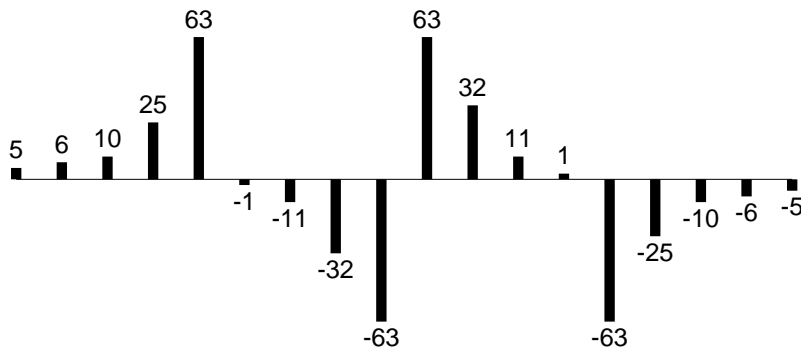
Figure 53 · Anti-Symmetric Constant Coefficient Filter Impulse Response (applies to the Fully Enumerated type only)

Figure 54 presents an example of a 2-set, 9-tap, 7-bit FIR filter coefficient file.

The core verifies whether the coefficient file is valid and issues detailed warnings if it is not. The entered coefficient and validation warnings on the Coefficients page of the core user interface can be viewed. The page also allows correction of the entry.

```
coefficient_set_1
5
6
10
25
63
-1
-11
-32
-63

coefficient_set_2
21
12
-10
-25
63
-11
-64
-32
18
```

Figure 54 Coefficient File Example - 2 Sets, 9 Taps, 7 Bits, Radix-10

Tool Flows

License

CoreFIR requires an RTL license to be used and instantiated. Complete source code and a testbench are provided for the core.

SmartDesign

CoreFIR is available for download to the Libero SoC IP Catalog through the web repository. Once it is listed on the catalog, the core can be instantiated using SmartDesign flow. The core can be configured using the configuration GUI within SmartDesign. An example of the GUI for the SmartFusion2 family is shown in Figure 55.
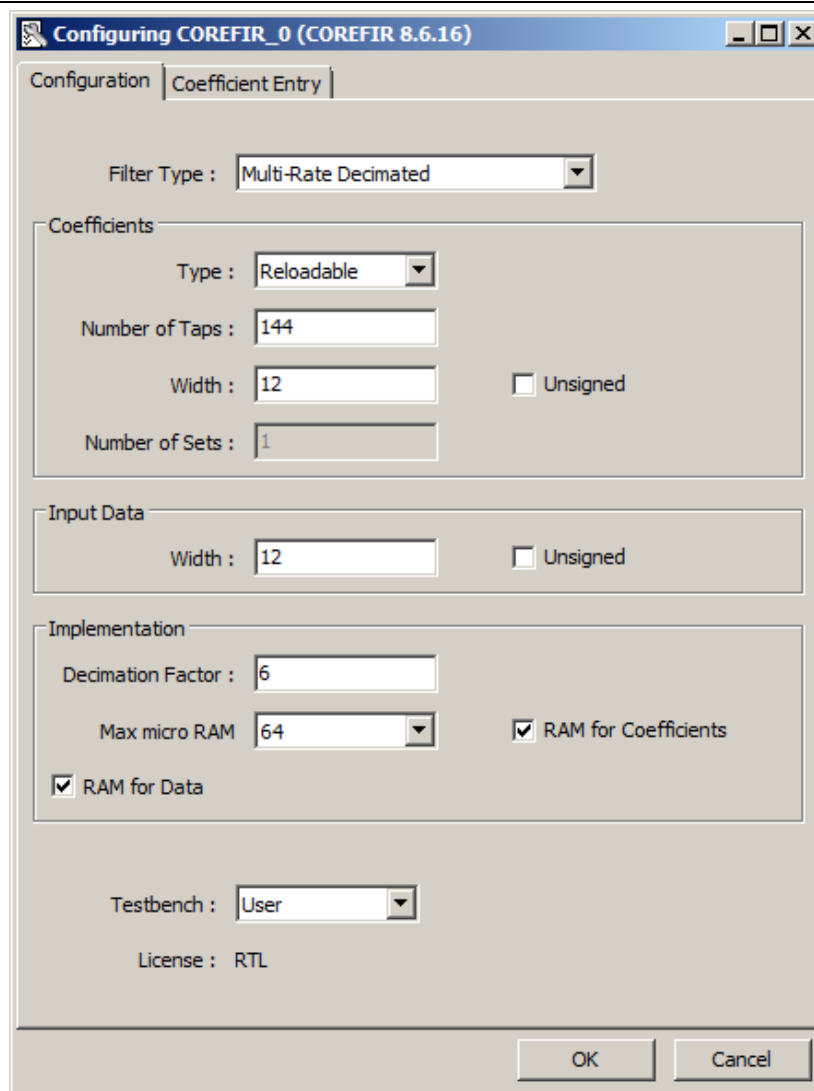

Figure 55 · Configuring CoreFIR

For information on using SmartDesign to configure, connect and generate cores, refer to the Libero SoC Online Help.

Note: A full path to a Libero project that instantiates CoreFIR including the project name must not contain spaces.

Simulation Flows

The User Testbench for CoreFIR is included in the release.

To run simulations, select the User Testbench flow within SmartDesign. The User Testbench is selected through the Core Configuration GUI.

When SmartDesign generates the core, it will install the user testbench files.

To run the user testbench, set the design root to the CoreFIR instantiation in the Libero design hierarchy pane and run Pre-Synthesis design simulation.

Note: When simulating the VHDL version of the core you might want to get rid of the IEEE.NUMERIC_STD library warnings. To do so add the following two lines to the automatically generated run.do file:

- set NumericStdNoWarnings 1
- set StdArithNoWarnings 1

User Testbench

CoreFIR comes with a user testbench, which can be invoked using the core user interface. Figure 56 depicts the testbench block diagram. The golden behavioral FIR filter directly implements the appropriate filter equation for a selected filter type. Both the golden behavioral filter and CoreFIR are configured identically and receive the same test signal. The testbench compares the output signals of the Golden filter and the UUT.
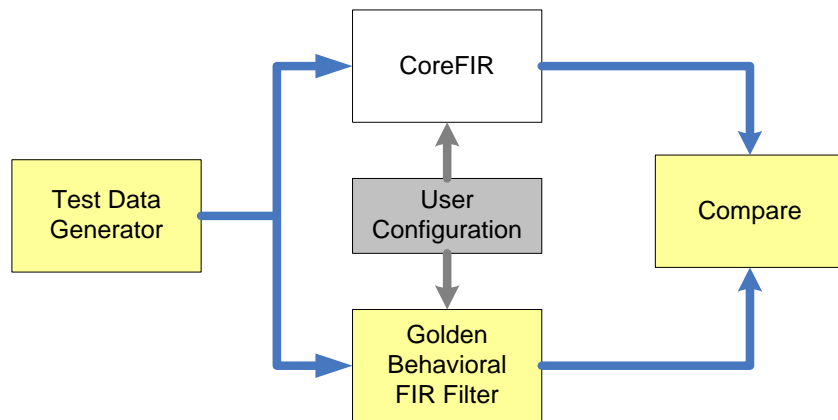

Figure 56 · CoreFIR User Testbench

The testbench provides examples of how to use generated FIR filter. The testbench can be modified as needed.

Fully Enumerated User Testbench

The testbench for the Fully Enumerated filter in Reloadable coefficient mode uses the constant coefficient file to configure the golden behavioral Filter and simulate Reloadable coefficients. To run the user testbench in Reloadable mode, follow the steps listed below.

1. Enter the desired core configuration. Set Constant Coefficient type.
2. Create and load constant coefficient file matching the core configuration.
3. Generate design.
4. Set Reloadable Coefficient type, generate design, and run simulation.

Synthesis in Libero SoC

Having set the design root appropriately, run synthesis tool from the Libero SoC Design Flow pane. Set Synplify to use the Verilog 2001 standard if Verilog is being used.

Place-and-Route in Libero SoC

After the design has been synthesized, run **Compile** and then **Place-and-Route** tools. CoreFIR requires no special place-and-route settings.

List of Changes

The following table shows important changes made in this document for each revision.

Date	Change	Page
March 2015	Updated the document for v8.6.	N/A
January 2014	Updated the document for v8.5.	N/A
May 2013	Updated Supported Families section (SAR 47945).	6
	Updated Table 12, Table 13, Table 18, Table 20, and Table 22 (SAR 47945).	N/A
	Updated Data Path Bit Width section (SAR 47945)	53
March 2013	Updated the document for v8.3.	N/A

Product Support

Microsemi SoC Products Group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, electronic mail, and worldwide sales offices. This appendix contains information about contacting Microsemi SoC Products Group and using these support services.

Customer Service

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

From North America, call **800.262.1060**
From the rest of the world, call **650.318.4460**
Fax, from anywhere in the world **650. 318.8044**

Customer Technical Support Center

Microsemi SoC Products Group staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions about Microsemi SoC Products. The Customer Technical Support Center spends a great deal of time creating application notes, answers to common design cycle questions, documentation of known issues and various FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

Technical Support

For Microsemi SoC Products Support, visit http://www.microsemi.com/products/fpga-soc/design-support/fpga-soc-support.

Website

You can browse a variety of technical and non-technical information on the Microsemi SoC Products Group home page, at http://www.microsemi.com.

Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center. The Technical Support Center can be contacted by email or through the Microsemi SoC Products Group website.

Email

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. We constantly monitor the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support email address is soc_tech@microsemi.com.

My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to My Cases.

Outside the U.S.

Customers needing assistance outside the US time zones can either contact technical support via email (soc_tech@microsemi.com) or contact a local sales office. Sales office listings can be found at www.microsemi.com/company/contact/default.aspx.

ITAR Technical Support

For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), contact us via soc_tech_itar@microsemi.com. Alternatively, within My Cases, select **Yes** in the ITAR drop-down list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the ITAR web page.

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; security technologies and scalable anti-tamper products; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 3,400 employees globally. Learn more at **www.microsemi.com**.

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996

E-mail: sales.support@microsemi.com

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided "as is, where is" and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.

50200174-7/03.15