Technical notes on using Analog Devices products and development tools
Visit our Web resources http://www.analog.com/ee-notes and http://www.analog.com/processors or e-mail processor.support@analog.com or processor.tools.support@analog.com for technical support.

Using ADSP-SC83x/2183x High Performance FIR/IIR Accelerators

Contributed by Sanket Nayak and Mitesh Moonat *Rev 1 – July 8, 2024*

Introduction

The ADSP-SC83x/2183x processors (hereby referred to as ADSP-SC83x processors) incorporate high-performance hardware accelerators dedicated to performing two widely used signal processing operations: FIR (Finite Impulse Response) and IIR (Infinite Impulse response), hereby referred to as FIRA and IIRA, respectively.

As shown in Figure 1, an ADSP-SC83x processor has a SHARC-FX core equipped with two FIR and four IIR accelerators. The SHARC-FX core and accelerators together can provide an overall performance of up to **20 Giga 32-bit Floating Point MAC (Multiply and Accumulate) Operations Per Second** when running at 1 GHz.

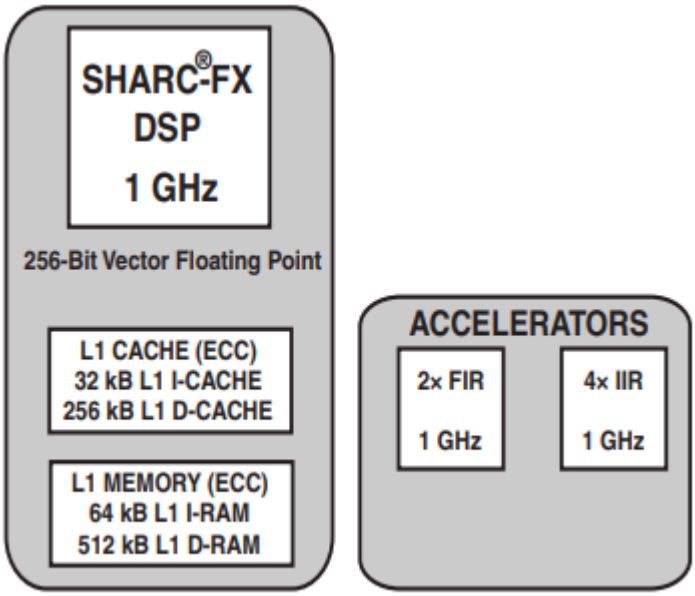


Figure 1: SHARC-FX Core and Accelerators on ADSP-SC83x Processors

As illustrated in Figure 2, these accelerators can run in tandem with the core to perform dedicated fixed-function computations (FIR and IIR), thereby increasing the overall processing capability of the processor.

This application note provides a brief overview of the functional and performance improvements achieved with the ADSP-SC83x FIR/IIR accelerators (hereby referred to as *accelerators*) when compared with their predecessors.

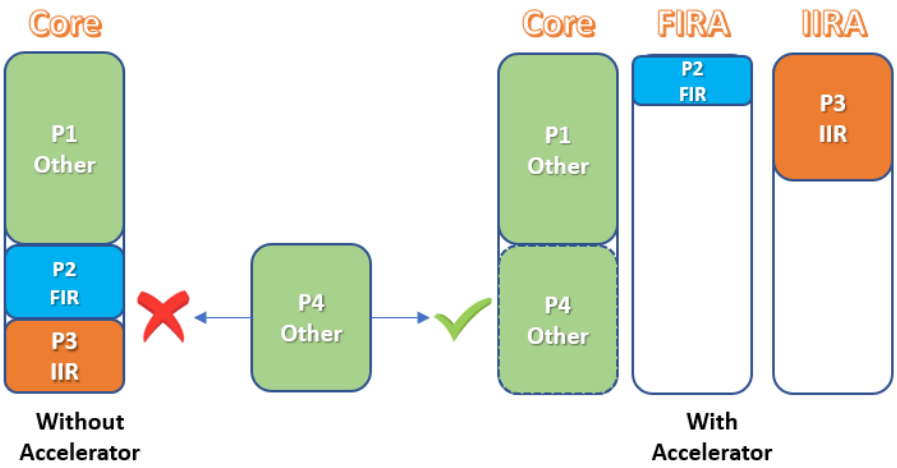


Figure 2: Processing Capability with and without FIR/IIR Accelerators

This note discusses the following topics related to:

- Accelerator performance under various configurations and contributing factors
- Accelerator drivers from CrossCore® Embedded Studio (CCES), including usage, examples, and benchmark details
- Accelerator usage models for optimum performance in various application scenarios

For more details about the accelerator architecture and programming model, refer to the *ADSP-2183x/ADSP-SC83x SHARC-FX Processor Hardware Reference* [1].

Most of the generic concepts from *Using ADSP-SC59x/2159x High Performance FIR/IIR Accelerators (EE-436)*[5] still apply here.. This application note includes the feature enhancements, figures, tables, and data specific to the ADSP-SC83x/2183x family of processors.

ADSP-SC83x Accelerator Enhancements

Accelerator enhancements include performance and functional improvements.

Performance Improvements

- There are no specific performance enhancements from accelerator module in ADSP-SC83x processors when compared to ADSP-SC59x processors. However, since the fabric and SHARC-FX core architecture is different, the overall performance may vary due to different fabric latencies.

Functional Improvements

The functional improvements of ADSP-SC83x accelerator include:

IIRA Coefficient Slewing

A dedicated hardware engine to slew the IIR coefficients in parallel with IIR filtering. This feature offloads the core to perform other tasks and speeds up the slewing process. The slewing engine is placed inside IIR filter to avoid any DMA overheads while loading the slewed coefficients. Setting a bit in IIR_CTL1 register enables the slewing mode of operation.

For more information on coefficient slewing, refer to *ADSP-SC83x/ADSP-2183x SHARC-FX Processor Hardware Reference* [1].

Accelerator Performance

The processing cycles of the accelerators consist of two components: DMA cycles and compute (MAC) cycles.

For more information, refer to *Using ADSP-SC59x/2159x High Performance FIR/IIR Accelerators (EE-436)*[5]

FIR/IIR Accelerators Performance

This section illustrates the FIR/IIR accelerators performance with different filter parameters and in different processors.

1. ADSP-SC83x FIRA/IIRA @ maximum ACLK (=CCLK) = 1 GHz
2. ADSP-SC59x FIRA/IIRA @ maximum ACLK (=CCLK) = 1 GHz
3. ADSP-SC83x(SHARC-FX) core @ maximum CCLK = 1 GHz
4. ADSP-2156x (SHARC+) core @ maximum CCLK = 1 GHz

For other applicable comparisons with different processors, refer to *Using ADSP-SC59x/2159x High Performance FIR/IIR Accelerators (EE-436)*[5].

All the buffers are placed in L1 memory during the measurements. Core FIR/IIR cycles are measured by the CCES library functions.

Additional measurement and analysis are provided to discuss how the various factors affect the MAC Utilization Efficiency (MUE) of the accelerator.

FIR Performance

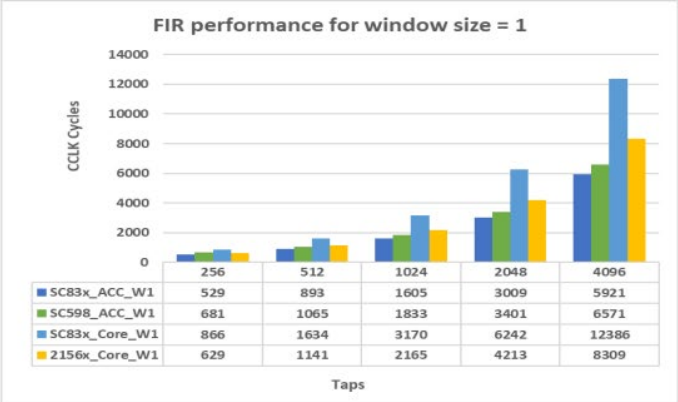
This section contains the following information:

- Measured performance of the ADSP-SC83x FIRA for different filter parameters and comparison with other cases
- Measured FIRA Compute Efficiency (CE) and contributing factors
- Effect of buffer placement (L1/L2/L3) on FIRA performance

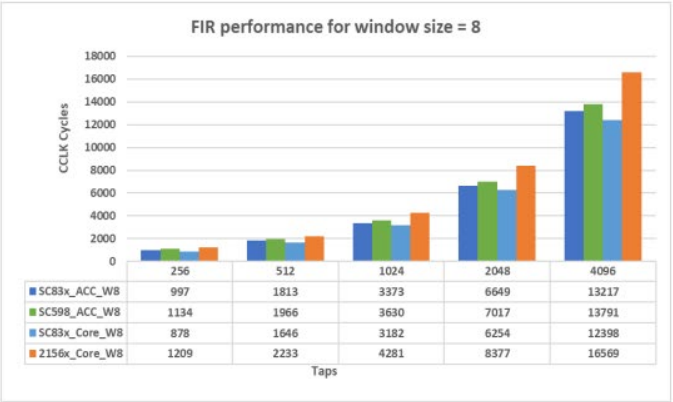
Comparison with ADSP-SC59x Accelerators/SHARC-FX (measured on ADSP-SC83x) and SHARC+ (measured on ADSP-2156x) Core

Figure 3 and Figure 4 show the measured FIR performance in core cycles and time units (µs), respectively, for the five cases. Figure 5 shows the performance improvement factor of one ADSP-SC83x FIR accelerator instance at different window sizes and a tap length of 512 in comparison with the other cases.

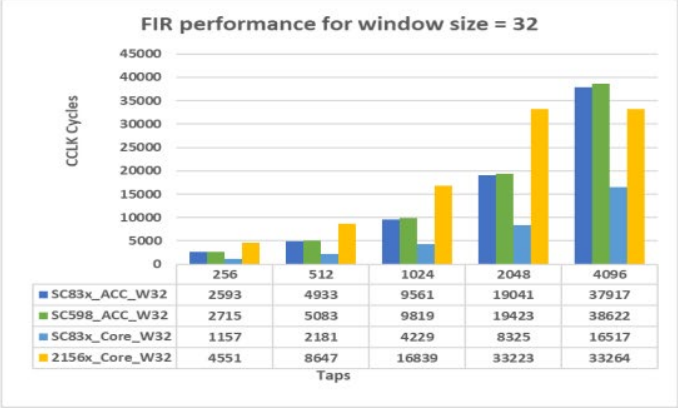
The following numbers compare the performance of a single instance of the accelerator on different processors



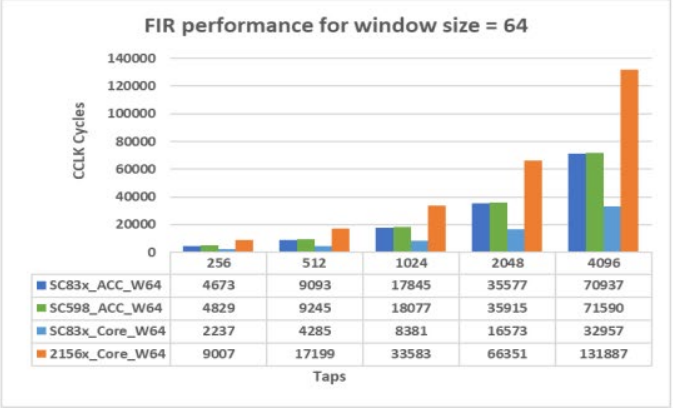
	256	512	1024	2048	4096
SC83x_ACC_W1	529	893	1605	3009	5921
SC598_ACC_W1	681	1065	1833	3401	6571
SC83x_Core_W1	866	1634	3170	6242	12386
2156x_Core_W1	629	1141	2165	4213	8309



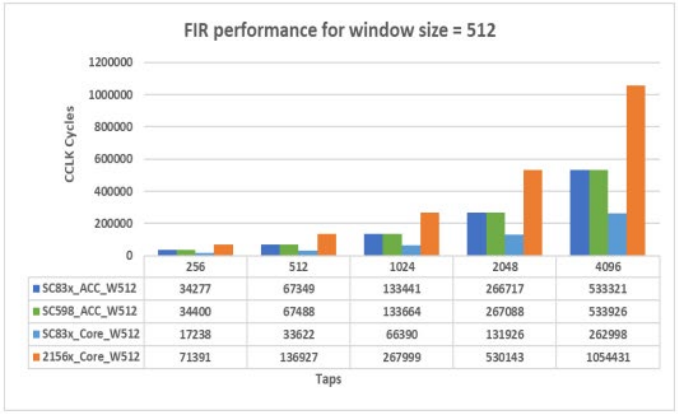
	256	512	1024	2048	4096
SC83x_ACC_W8	997	1813	3373	6649	13217
SC598_ACC_W8	1134	1966	3630	7017	13791
SC83x_Core_W8	878	1646	3182	6254	12398
2156x_Core_W8	1209	2233	4281	8377	16569



	256	512	1024	2048	4096
SC83x_ACC_W32	2593	4933	9561	19041	37917
SC598_ACC_W32	2715	5083	9819	19423	38622
SC83x_Core_W32	1157	2181	4229	8325	16517
2156x_Core_W32	4551	8647	16839	33223	33264



	256	512	1024	2048	4096
SC83x_ACC_W64	4673	9093	17845	35577	70937
SC598_ACC_W64	4829	9245	18077	35915	71590
SC83x_Core_W64	2237	4285	8381	16573	32957
2156x_Core_W64	9007	17199	33583	66351	131887



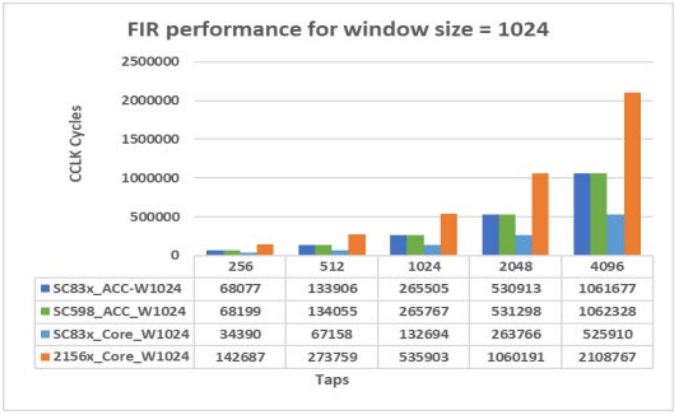
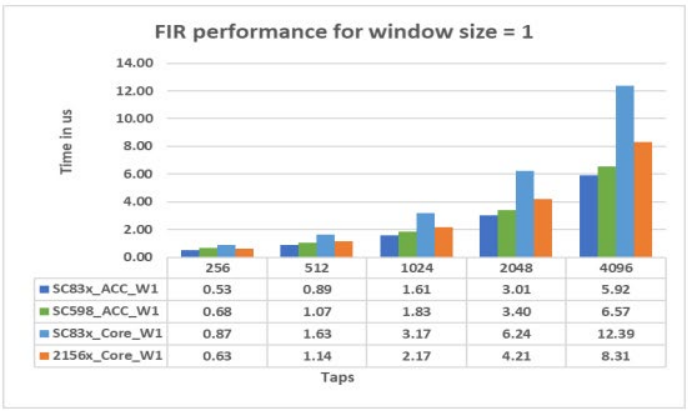
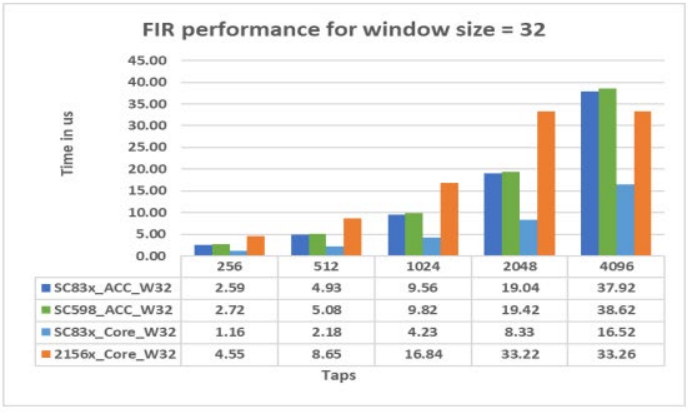
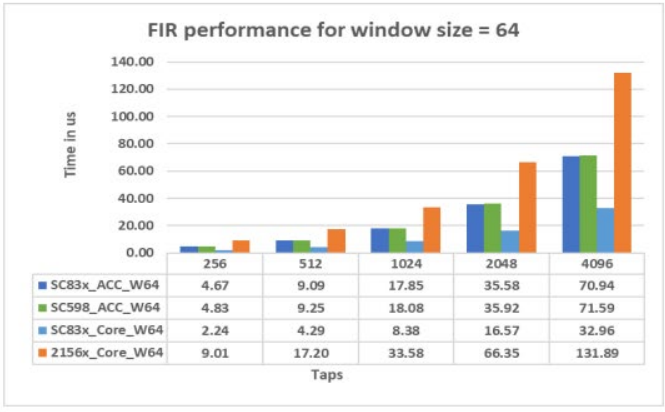


Figure 3: FIR Performance in Core Cycles Across Window Size and Tap Length







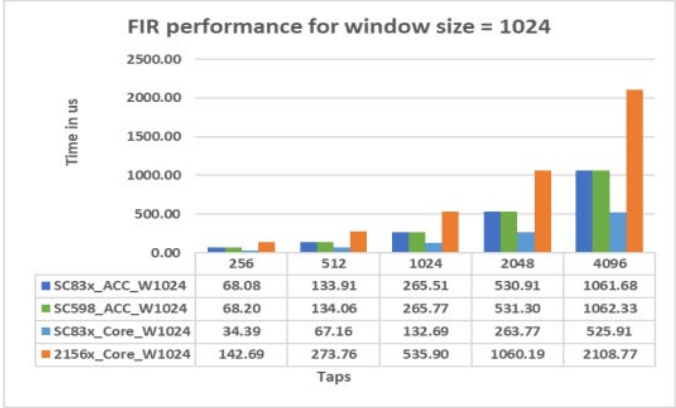


Figure 4: FIR Performance in Time (µs) Across Window Size and Tap Length

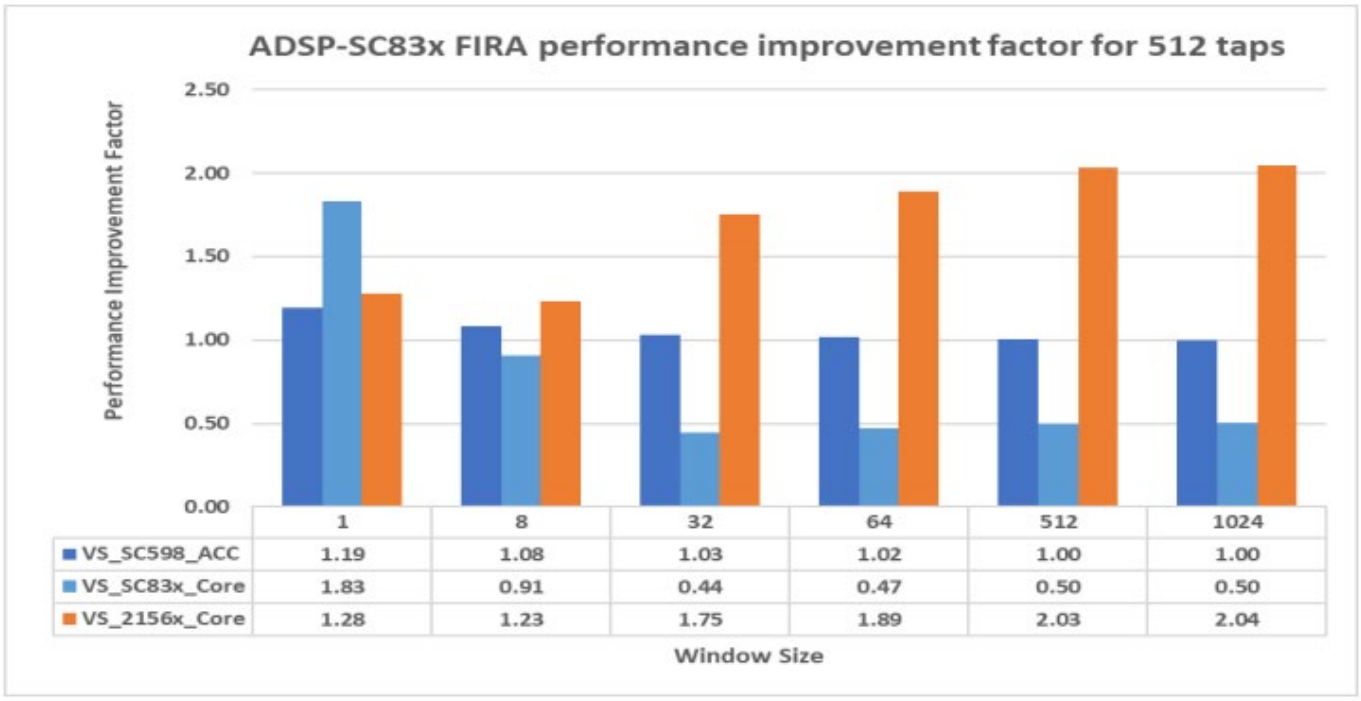


Figure 5: ADSP-SC83x FIR Performance Improvement Factor for 512 Taps Considering Time

The following observations are related to the FIR performance numbers shown in Figure 5:

- The ADSP-SC83x FIRA performance is similar to ADSP-SC59x/2159x FIRA in the cases of large window sizes. Both ADSP-SC83x FIRA and ADSP-SC59x/2159x FIRA operate at 1 GHz.
- **The ADSP-SC83x SHARC-FX core's performance is around 2x ADSP-SC83x FIRA for large window sizes**. The improvement is because SHARC-FX core has eight MAC units, whereas the ADSP-SC83x FIRA has four MAC units. An exception to this conclusion is lower block sizes where implementation-specific software overheads of the SHARC-FX core dominate the MAC computation cycles.

- **The ADSP-SC83x FIRA performance is around 2x the ADSP-2156x SHARC+ core**. The improvement is because the **ADSP-SC83x FIRA** has four MAC units, whereas the ADSP-2156x SHARC+ core has two MAC units. An exception to this conclusion applies to smaller block sizes, where the DMA overhead dominates the compute cycles.

Compute Efficiency (CE)

For more information, refer to *Using ADSP-SC59x/2159x High Performance FIR/IIR Accelerators (EE-436)*[5]

Buffer Placement

For more information, refer to *Using ADSP-SC59x/2159x High Performance FIR/IIR Accelerators (EE-436)*[5]

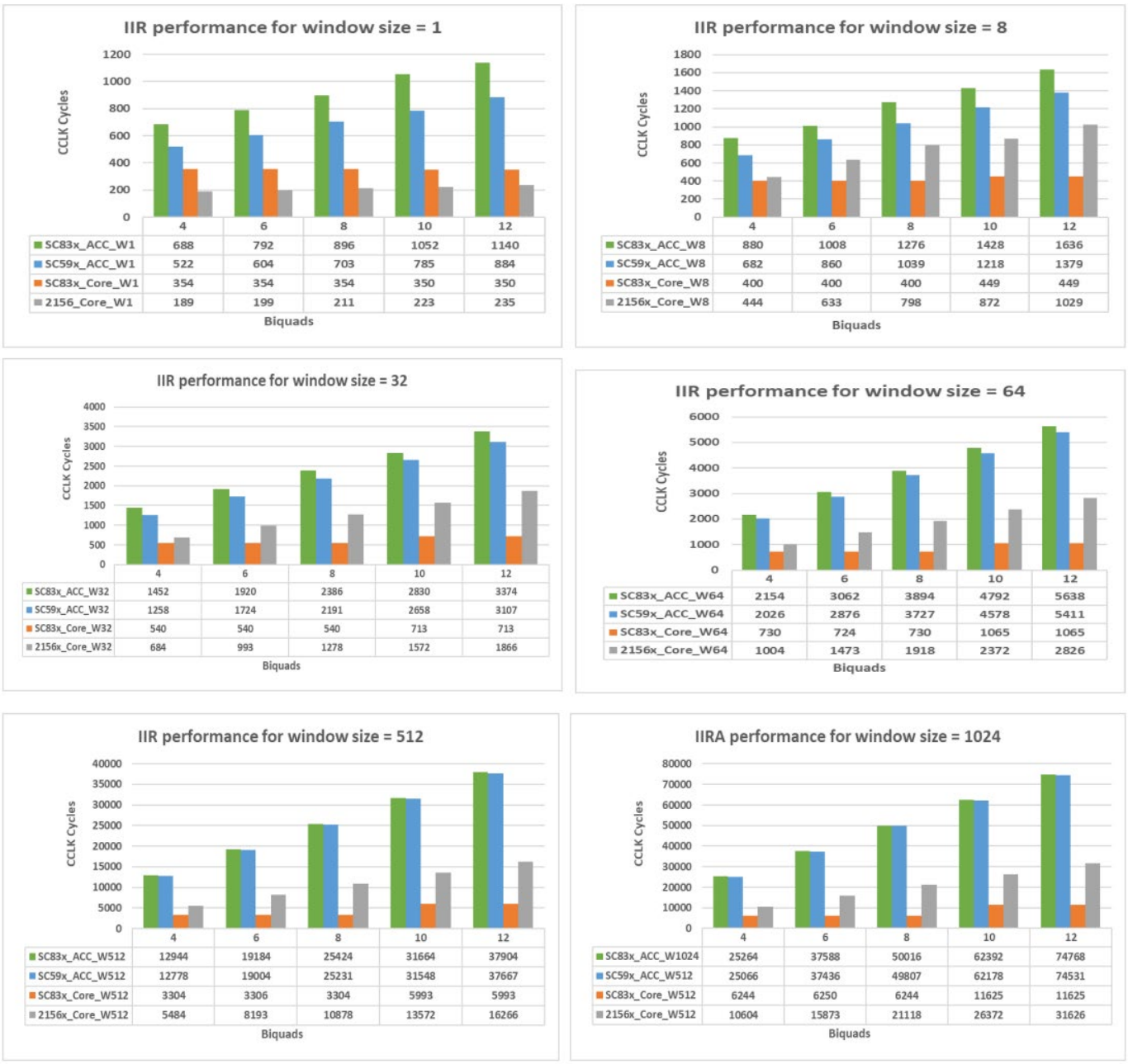
IIR Performance

This section contains the following information:

- Measured performance of the ADSP-SC83x IIRA for different filter parameters and comparison with other cases
- Measured IIRA Compute Efficiency (CE) and contributing factors
- Effect of buffer placement (L1/L2/L3) on IIRA performance

Comparison with ADSP-SC59x Accelerators/SHARC-FX (measured on ADSP-SC83x) Core and SHARC+ (measured on ADSP-2156x) Core

Figure 6 and Figure 7 show the measured IIR performance of one IIRA instance in core cycles and time units (µs), respectively, for all the cases. Figure 8 shows the performance factor of the ADSP-SC83x IIRA as compared with other cases for different window sizes and six biquads.



	4	6	8	10	12
SC83x_ACC_W1	688	792	896	1052	1140
SC59x_ACC_W1	522	604	703	785	884
SC83x_Core_W1	354	354	354	350	350
2156_Core_W1	189	199	211	223	235

	4	6	8	10	12
SC83x_ACC_W8	880	1008	1276	1428	1636
SC59x_ACC_W8	682	860	1039	1218	1379
SC83x_Core_W8	400	400	400	449	449
2156x_Core_W8	444	633	798	872	1029

	4	6	8	10	12
SC83x_ACC_W32	1452	1920	2386	2830	3374
SC59x_ACC_W32	1258	1724	2191	2658	3107
SC83x_Core_W32	540	540	540	713	713
2156x_Core_W32	684	993	1278	1572	1866

	4	6	8	10	12
SC83x_ACC_W64	2154	3062	3894	4792	5638
SC59x_ACC_W64	2026	2876	3727	4578	5411
SC83x_Core_W64	730	724	730	1065	1065
2156x_Core_W64	1004	1473	1918	2372	2826

	4	6	8	10	12
SC83x_ACC_W512	12944	19184	25424	31664	37904
SC59x_ACC_W512	12778	19004	25231	31548	37667
SC83x_Core_W512	3304	3306	3304	5993	5993
2156x_Core_W512	5484	8193	10878	13572	16266

	4	6	8	10	12
SC83x_ACC_W1024	25264	37588	50016	62392	74768
SC59x_ACC_W512	25066	37436	49807	62178	74531
SC83x_Core_W512	6244	6250	6244	11625	11625
2156x_Core_W512	10604	15873	21118	26372	31626

Figure 6: IIR Performance in Core Cycles Across Window Size and Biquad Values

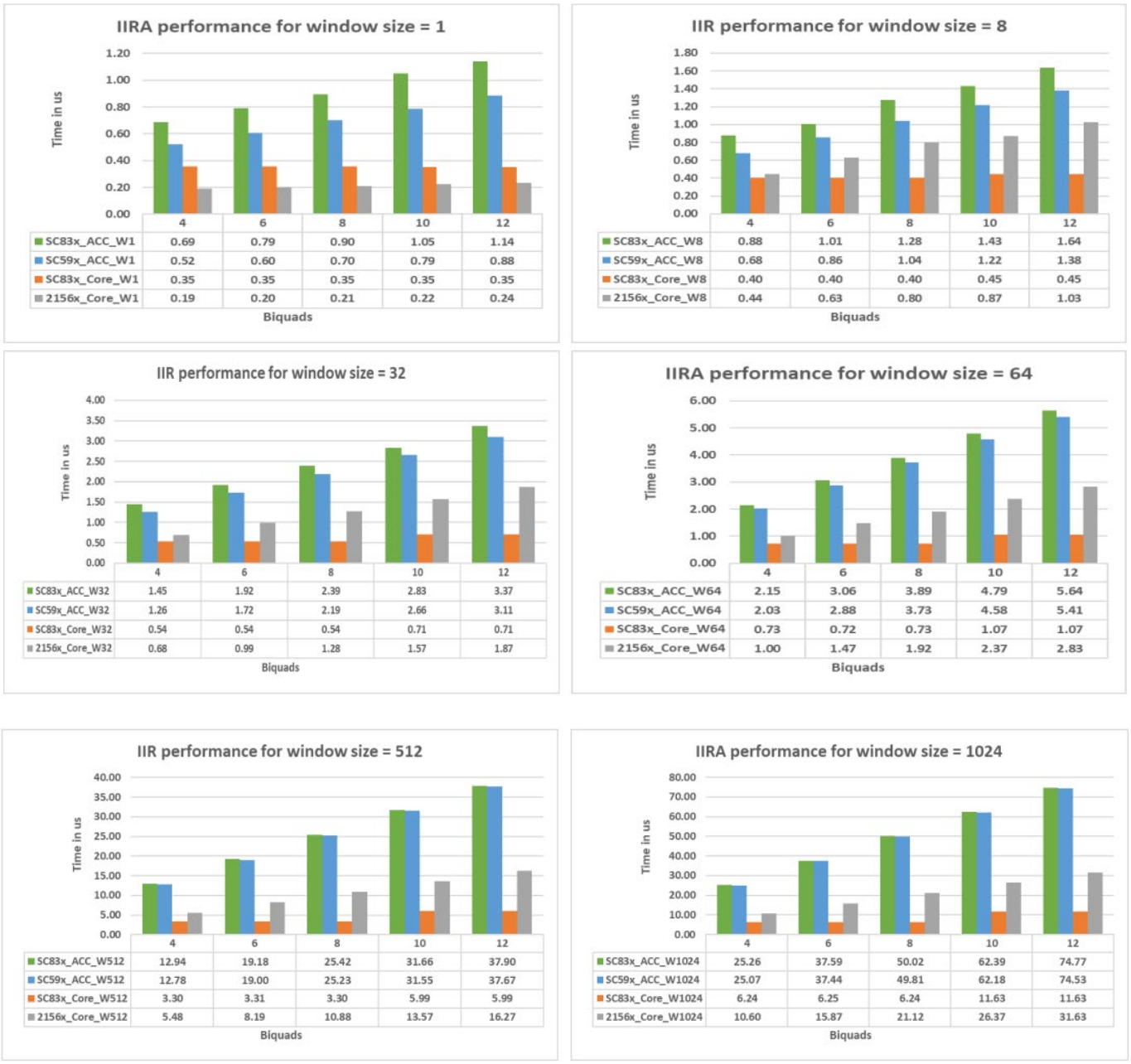
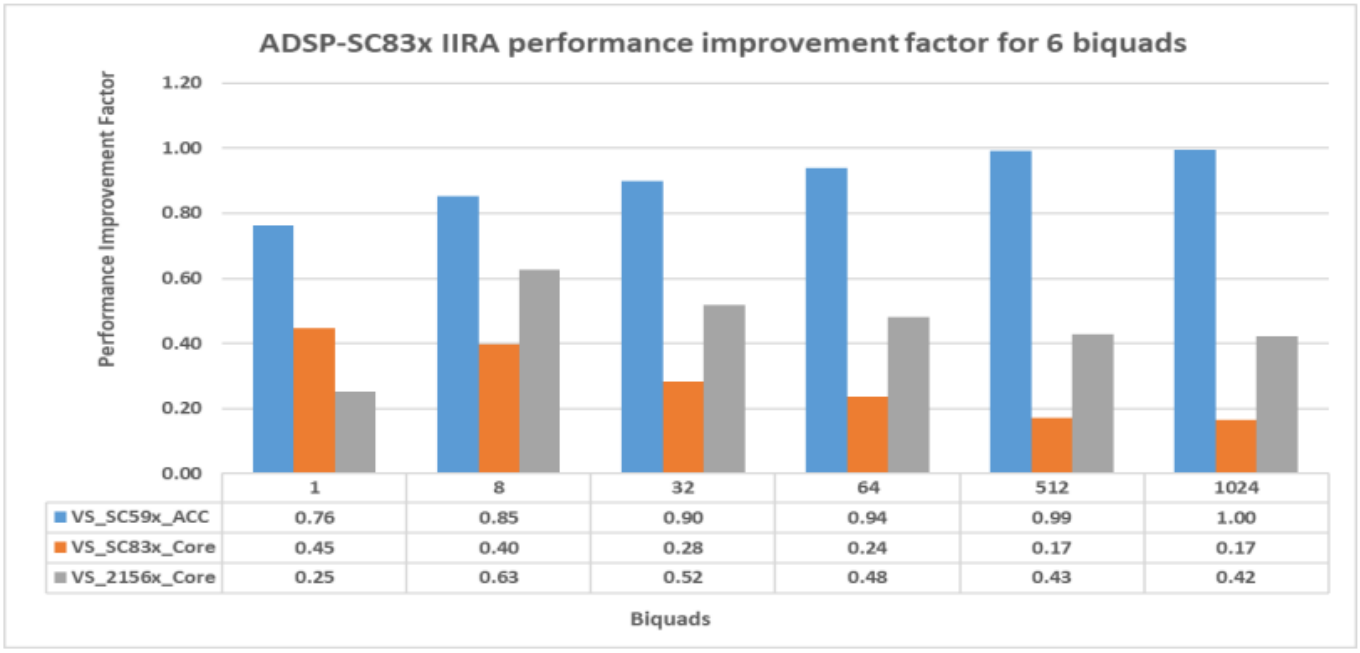


Figure 7: IIR Performance in Time (µs) Across Window Size and Biquad Values

The above numbers compare the performance of a single accelerator instance on different processors.



	1	8	32	64	512	1024
VS_SC59x_ACC	0.76	0.85	0.90	0.94	0.99	1.00
VS_SC83x_Core	0.45	0.40	0.28	0.24	0.17	0.17
VS_2156x_Core	0.25	0.63	0.52	0.48	0.43	0.42

Figure 8: ADSP-SC83x IIR Performance Improvement Factor for 6 Biquads Considering Time

The following observations relate to the IIR performance numbers shown in Figure 8:

- The ADSP-SC83x IIRA performance is close to ADSP-SC59x IIRA performance at large window sizes. Both ADSP-SC59x IIRA and ADSP-2156x IIRA run at CCLK (1 GHz).
- The ADSP-SC83x IIRA performance is around 0.17x the ADSP-SC83x SHARC-FX core's execution of IIR code.
- **The ADSP-SC83x IIRA performance is around 0.42x the ADSP-2156x SHARC+ core's execution of IIR code**. Each ADSP-SC83x IIRA instance has one MAC unit, whereas each SHARC+ core has two MAC units. Additionally, the IIRA compute unit takes six cycles per biquad, whereas the SHARC+ core takes 2.5 cycles.

Compute Efficiency (CE)

For more information, refer to *Using ADSP-SC59x/2159x High Performance FIR/IIR Accelerators (EE-436)*[5]

Buffer Placement

For more information, refer to *Using ADSP-SC59x/2159x High Performance FIR/IIR Accelerators (EE-436)*[5]

Programming the Accelerators

The ADSP-SC83x accelerators can be programmed using the device drivers available with the CCES installation package.

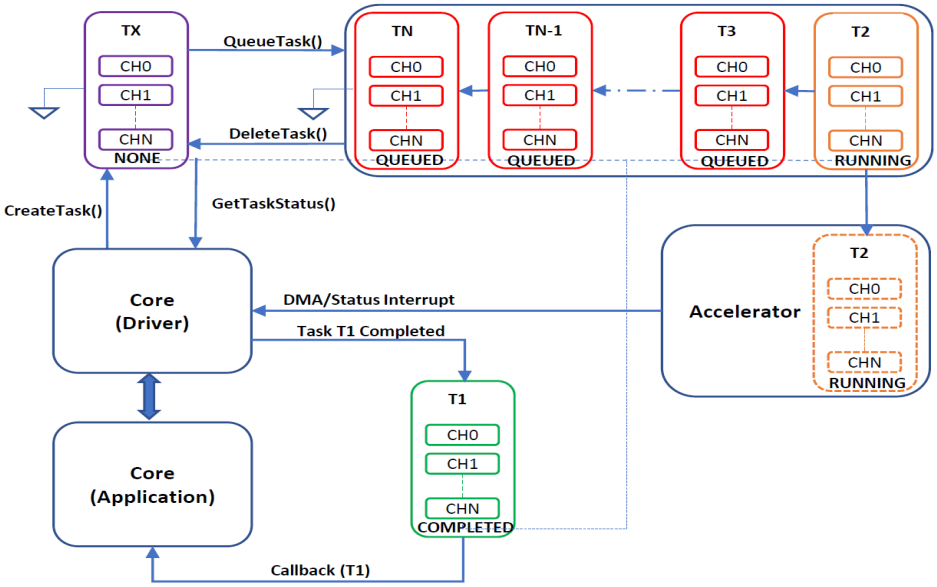


Figure 9: ADSP-SC83x Accelerator Device Driver Structure

The programming model is the same for FIR and IIR accelerators. As shown in Figure 9, the accelerator drivers follow a queueing programming model. Abstractions of task and queue are used in the model. A task contains a set of channels which, when given to the accelerator, are processed sequentially. A queue is a set of tasks maintained on a first-in-first-processed basis which the accelerator processes in sequentially. Applications can create tasks for the accelerator using the `adi_fir(/iir)_CreateTask()` and queue the same when required using the `adi_fir(/iir)_QueueTask()`. The driver maintains a processing queue of the tasks given/queued by the application for the accelerator and sequentially schedules the same to the accelerator. The applications can register a user-defined callback using `adi_fir(/iir)_RegisterCallback()`. The driver uses the callback to notify the application on completion of the tasks/channels in a task. The driver also maintains the status of all the tasks the application has queued. The application can also query the status of the task as required using the `adi_fir(/iir)_GetFir(/Iir)TaskStatus()`.Once the task which was queued is completed by the accelerator, the task is removed from the processing queue. The driver maintains the information of all the tasks which are completed/created. Hence, the application can reuse the tasks which were created by simply queuing it again using `adi_fir(/iir)_QueueTask()`.

The accelerator can be configured to operate in legacy mode or ACM using static configuration. In legacy mode, the driver maintains a software queue (linked list) of the tasks queued by the application and sequentially schedules the same to the accelerator. In ACM, the driver utilizes the halt feature in hardware to implement a hardware queuing mechanism. There are additional channel-customizable features available in ACM like interrupts/triggers and more. Refer to `ADI_FIR(/IIR)_CHANNEL_INFO` in the *ADSP-SC83x SSDD API Reference Manual for SHARC-FX Core*[2] in the CCES help for details. The programming model of the driver is the same for both legacy and ACM operation.

The `FIR_Multi_Channel_Processing` and `IIR_Multi_Channel_Processing` code examples provided with this application note[3] (also available with the ADSP-SC83x EZ-Kit *Board Support Package*) can be used as a reference to understand how to use the ADSP-SC83x accelerator device drivers.

Table 1 summarizes the accelerator driver benchmarks measured on the ADSP-SC83x EZ-Kit evaluation board. The `ADSP_SC83x_FIRA_Driver_Benchmark` and `ADSP_SC83x_IIRA_Driver_Benchmark` code examples supplied with this application note[3] were used to obtain these measurements. Note that these numbers were measured with all buffers in L1 memory.

The driver provides a NOQUEUE mode, where only a single task can be assigned to the accelerator for processing; it can be run for multiple iterations (Windows) without needing to reload the coefficients into accelerator memory for every iteration. The `ADI_IIR_CFG_NOQUEUE_MODE` macro can be set to 1 in static configuration to enable this mode.

Applications can call `adi_iir_StartTask()` API to assign the task to the accelerator and start processing. The same API can be called multiple times to process multiple iterations (Windows) without the need to reload the coefficients for every iteration. Applications can call `adi_iir_UnassignTask()` API to unassign the task from the accelerator.

The `ADI_IIR_CFG_SLEW_EN` macro can be set to 1 in static configuration to enable slewing feature. Note that in order to use slewing feature, NOQUEUE mode of the driver should be enabled.

Slewing parameters can be configured in the task while creating the task using `adi_fir_CreateTask()` API. The slewing parameters of the already created task can be updated by using `adi_iir_UpdateTask()` API.

The following is an example of how the application can use the slewing feature :

Initially, when the application does not require slewing, it can create a task with the `nNn` parameter in `ADI_IIR_CHANNEL_INFO` set to 1. Setting `nNn` to 1 disables slewing. It can use `adi_iir_StartTask()` API to process the input. When slewing needs to be enabled during processing, the application can use `adi_iir_UpdateTask()` API to update the task with the slewing parameters. When the application calls `adi_iir_StartTask()` further, the coefficients are slewed according to the slewing parameters. The application can continue calling this API even after the coefficient slewing is complete. Note that the save state should be disabled during coefficient slewing.

Operation	**Description**	**Measured Core Cycles**			
		FIRA		**IIRA**	
		Legacy	**ACM**	**Legacy**	**ACM**
CreateTask()	Constant Overhead	104	55	250	68
	Per Channel Overhead	707	782	661	728
QueueTask()	Pushing a task to the empty queue	780	1138	982	1220
	Pushing a task to the non-empty queue	324	612	344	720
Interrupt Overhead (Round trip cycles including SEC interrupt dispatcher latency)	For a single task in queue	788	872	820	928
	For more than one task in queue	920	1105	956	1126

Table 1: ADSP-SC83x Accelerator Driver Benchmarks

Additional overhead for cache flushing/invalidation can occur when the buffers are placed in L2/L3 memories. It is recommended to place the buffers in L1 memory to avoid cache overhead.

Accelerator Usage Models

For more information, refer to *Using ADSP-SC59x/2159x High Performance FIR/IIR Accelerators (EE-436)*[5]

Real-Time Implementation of an Example FIR/IIR Use Case

Figure 10 shows an example of FIR/IIR real-time processing of 12-channel audio data.

- Each channel passes through a 1024-tap FIR filter followed by a five-band IIR equalizer.
- For simplification, the FIR filter is an all-pass filter (the `b0` coefficient is one, and all other coefficients are zero).

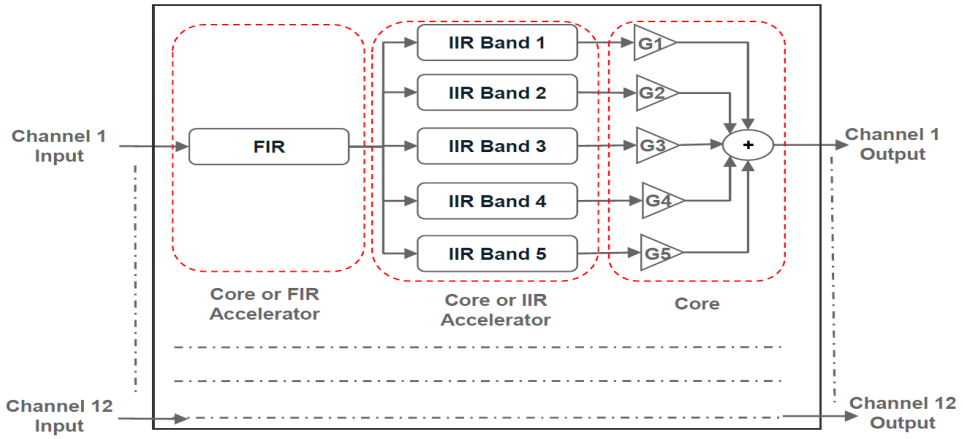


Figure 10: FIR/IIR Real-Time Use Case

- Each IIR band is an 8th order IIR filter implemented with four cascaded biquad stages. The IIR band details are as shown in Table 2. The output of each band is multiplied with the respective gain (from 0.0 min to 1.0 max) and added together to generate the final output. The ***IIRGains*** flag can be set or cleared dynamically using the CCES memory browser to respectively bypass or enable processing.

Band No.	Lower Cutoff in Hz	Higher Cutoff in Hz	Filter Type
1	-	250	Low Pass
2	250	500	Band Pass
3	500	2000	Band Pass
4	2000	4000	Band Pass
5	4000	-	High Pass

Table 2: IIR Bands Details

- The block size is 256 samples per channel.
- All buffers are placed in internal (L1) memory.

The core performs Ogg Vorbis decoding of .ogg music file.

The `codes under 'OggVorbis' folder` supplied with the application note [3] implements this case on the ADSP-SC83x evaluation board for the different usage models.

The core and accelerator performance numbers are measured for the *core only* case and for different accelerator usage model along with the resultant core MIPS savings for each model. Table 3 shows the MIPS of the core and accelerator for different usage models along with the core MIPS savings. The numbers measured on ADSP-SC594 are mentioned in braces for comparison.

From a performance perspective, the *Data Pipelining* model is the best, resulting in saving approximately 85 core MIPS, followed by the *Split Task* model (saving 29 core MIPS) and then the *Direct Replacement* model (saving -5 core MIPS). These results align with the previous discussion about the accelerator usage models.

Usage Model	MIPS Usage			Core MIPS Saving
	Core	FIRA	IIRA	
Core Only	104 (333)	0.00	0.00	-
Direct Replacement	109 (114)	72 (76)	19 (11)	-5 (219)
Split Task	75 (102)	44 (64)	12 (11)	29 (231)
Data Pipelining	19 (10)	72 (76)	18 (11)	85 (323)

Table 3: MIPS Summary for Different Accelerator Usage Models

References

[1] *ADSP-2183x/ADSP-SC83x SHARC-FX Processor Hardware Reference,* Rev 0.1, March 2024. Analog Devices, Inc.

[2] *ADSP-SC83x SSDD API Reference Manual for SHARC-FX Core*, Version 1.0, CrossCore® Embedded Studio Help.

[3] *Associated ZIP file for EE-460: Using ADSP-SC83x/2183x High Performance FIR/IIR Accelerators,* July 2024. Analog Devices, Inc.

[4] *ADSP-21591/21593/21594/ADSP-SC591/SC592/SC594 Silicon Anomaly List (Rev. E).*

[5] *EE-436: Using ADSP-SC59x/2159x High Performance FIR/IIR Accelerators,* September 2022. Analog Devices, Inc.

Document History

Revision	Description
Rev 1 – July 8, 2024 *by Sanket Nayak and Mitesh Moonat*	Initial Release