Technical notes on using Analog Devices products and development tools
Visit our Web resources http://www.analcom/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-SC59x/2159x High Performance FIR/IIR Accelerators

Contributed by Sanket Nayak and Mitesh Moonat *Rev 2 – September 23, 2022*

Introduction

ADSP-SC59x/2159x processors (referred in this EE Note as ADSP-SC59x processors) incorporate high-performance hardware accelerators dedicated to performing two widely used signal processing operations, Finite Impulse Response (FIR) and Infinite Impulse response (IIR). These accelerators are hereby referred to FIRA and IIRA respectively.

Figure 1 shows an ADSP-SC59x processor that can have one or two SHARC+ cores, each equipped with one FIR and four IIR accelerators. The SHARC+ cores and accelerators together can provide an overall performance of up to *20 Giga Floating Point MAC (Multiply and Accumulate) Operations Per Second* when running at 1 GHz.

Figure 1 SHARC+ Cores and Accelerators on ADSP-SC59x Processors

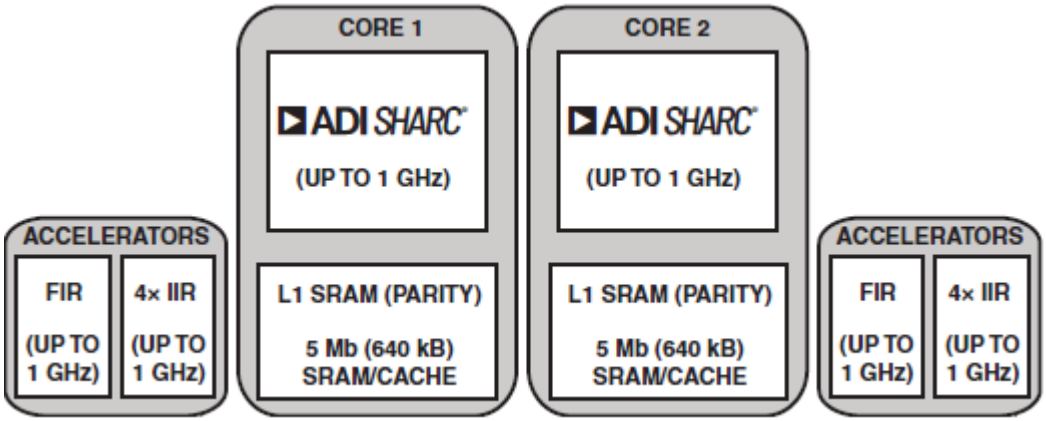
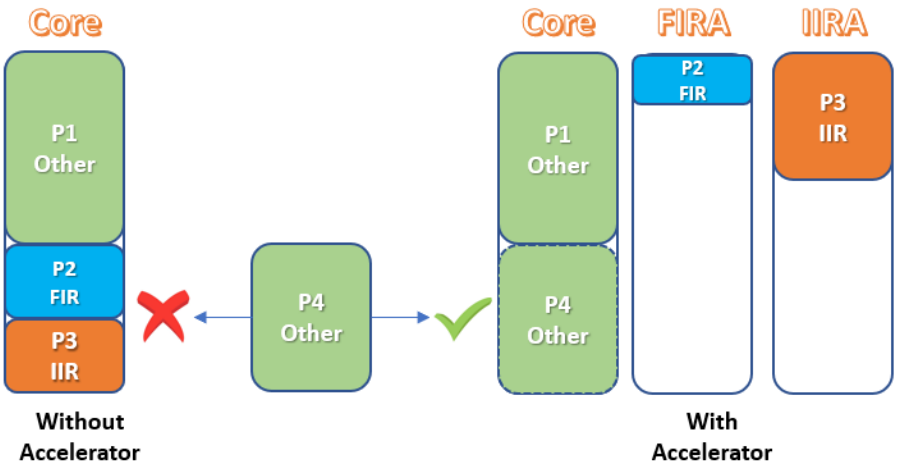


Figure 2 Processing Capability with and without FIR/IIR Accelerators on page 2, shows these accelerators can run in tandem with the core performing the dedicated fixed-function computations (FIR and IIR). This increases the overall processing capability of the processor.

This application note provides a brief overview of the functional and performance improvements achieved with the ADSP-SC59x FIR and IIR accelerators. For this EE Note, FIRs and IIRs are referred to as *accelerators,* when compared with their predecessors.

Figure 2 Processing Capability with and without FIR/IIR Accelerators



This Engineer-to-Engineer Note discusses the following topics:

- Accelerator performance under different configurations and contributing factors
- Accelerator drivers from the CrossCore® Embedded Studio (CCES) software, that include usage directions, examples, and benchmark details
- Accelerator usage models for optimum performance in different application scenarios

For more details about the accelerator architecture and programming model, refer to the *ADSP-SC59x SHARC+ Processor Hardware Reference* [1].

A majority of the content of this application note is the same as in *Using ADSP-2156x High Performance FIR/IIR Accelerators* (EE-408) [5] EE Note. This application note includes the feature enhancements, figures, tables, and data specific to the ADSP-SC59x/2159x family of processors.

The content of this application note supports both ADSP-SC594 and ADSP-SC598 family of processors. In the following sections, ADSP-SC59x/2159x refers to the *ADSP-SC594 family*. The details applicable to the ADSP-SC598 family are explicitly mentioned.

ADSP-SC59x Accelerator Enhancements

Accelerator enhancements include both performance and functional improvements.

Performance Improvements

- More FIRA and IIRA instances exist in the ADSP-SC59x processors, compared to previous processors. Each SC59x SHARC+ core has one FIRA and four IIRA instances, all running at CCLK frequency. Both SHARC+ cores can access all the FIRA and IIRA instances, and all the instances can run in parallel. Eight IIRA instances can support a total of 2304 biquads [6] (288*8) and 256 channels (32*8). The ADSP-2156x processors can only support 288 biquads and 32 channels.

When a SHARC+ core accesses the accelerator instance associated with another SHARC+ core, there is additional access latency, compared to accessing its own accelerator instance.

- Finite Impulse Response Accelerator (FIRA) Direct Memory Access (DMA) overhead optimization.
    - Maximum Burst Transfer size is increased from INCR8 to INCR16 to reduce the Transfer Control Block (TCB) load time. This feature can be disabled by setting `FIRCTL1.BURST16_DIS` bit. However, a burst length of 16 requires the end address of the Input Buffers must be 4-KB aligned. (Refer to Anomaly 20000114 in the *ADSP-21591-21593-21594-SC591-SC592-SC594 Anomaly List* [4].)
    - The Data and Coefficient load now happens in parallel as compared to sequential loading in previous processors thereby reducing the DMA load time. This feature can be disabled by setting the `FIR_CTL1.DCP_DIS` bit.
    - The use of Pre-Fetch Buffer reduces the access latency in first data arrival of a Burst Transfer. This feature can be set by the `FIR_CTL1.PFB_EN` bit. Two target SHARC+ Core ports (S1 and S2) can be used to balance the DMA load. One constraint is that this feature can only be used when the Output Buffer and Input/Coefficient Buffers of the respective channels are separated by a minimum size of 12 bytes. (Refer to Anomaly 20000118 in *ADSP-21591-21593-21594-SC591-SC592-SC594 Anomaly List* [4].)

Pre-Fetch Buffer can be enabled only when the Maximum Burst Transfer size of 16 is enabled in the ADSP-SC59x/2159x.

In the ADSP-SC598, the Pre-Fetch Buffer and the Maximum Burst Transfer size of 16 can be enabled or disabled, independent of each other.

Functional Improvements

The functional improvements of the ADSP-SC59x accelerator include:

- *Legacy Mode Index Update Scheme in Auto Configuration Mode*–In Legacy Mode, after processing a window of data, the FIRA and IIRA updates the TCB Input and Output Buffer Index using a Circular Buffer Scheme. This feature can be enabled in the Auto Configuration Mode (ACM) by setting the `FIR_CTL1.SMQ_LIUPS_EN` bit. When disabled, the Input Buffer Index updates to `0x00000000` and the Output Buffer Index updates to `0xFFFFFFFF`.
- *IIRA Selective Interrupt Generation in Legacy Mode*–In Legacy Mode, an interrupt can be generated either after processing all TCBs or after processing each TCB, by the `CTL1.CCINTR` bit. Interrupts can be customized in the ACM on a per-channel basis by setting the `IIR_CTL1.L_TIMASK_EN` bit.
- *IIRA Trigger Generation and Trigger Wait in Legacy Mode*–In ACM, each channel can generate a controller trigger after its completion. This is then routed to a trigger target by the Trigger Routing Unit (TRU). Each channel can be configured to wait for a controller trigger after loading the TCB. The above trigger mechanism is now available in Legacy Mode and can be enabled using the `IIR_CTL1.L_TIMASK_EN` and `IIR_CTL1.L_TWAIT_EN` bits.

Accelerator Performance

Accelerator processing cycles consist of two components, DMA cycles, plus Multiply and Compute (MAC) cycles.

DMA Cycles

Two factors significantly impact the DMA cycles, Filter Parameters and Buffer Placement.

Filter Parameters–DMA cycles consist of two operations:

- Initial preload of the coefficients and delay line (FIRA) or state variables (IIRA)–for IIRA, the coefficient load and state variable load can be skipped in Legacy Mode to save these cycles. The initial preload data size depends on the **Tap Length** (FIRA) or the **Number of Biquads** (IIRA).
- Fetch and Store of the Input/Output buffers–these operations are influenced by **Window Size** and take place in parallel with the next compute operation. As such, they do not add any overhead for a larger tap length or number of biquads. For a smaller tap length and biquad values, these cycles *can dominate* the total processing cycles. The crossover point (N) in terms of tap length (FIRA) and biquads (IIRA) depends on the memory access latency. The value of the crossover point increases as buffers are moved from L1 to L2 memory, and further to L3 memory.

Buffer Placement–The DMA cycles depend on the location of the input, output, and coefficient buffers in DSP memory, as each memory type, L1 or L2, or L3, has different access latencies.

For optimal performance in FIRA, Analog Devices recommends that the Input/Coefficient Buffers are 4-KB aligned and the TCB is 64-Byte aligned.

Compute (MAC) Cycles

Unlike previous SHARC processors, the MAC units in the ADSP-SC59x/ADSP-2156x accelerators are derived from the SHARC+ core MAC units, where a multiplication operation takes place in two stages. This design implies that a minimum of N+1 number of cycles are required to perform N MAC operations. For the FIRA, the tap length size is generally significantly higher; consequently, the theoretical number of cycles per sample per tap is approximately 0.25 (1/4). For the IIRA, processing each biquad requires 6 cycles (for 5 MAC operations).

In an ideal scenario, to make maximum use of the accelerator computing power, the percent contribution of the compute and MAC cycles, called the **Compute Efficiency (CE)**, must be 100%.

CE can be measured using the expression:

CE = (M/N)*100

where,

M = Theoretical Cycles per Tap (FIRA = 0.25) or per biquad (IIRA = 6)

N = Measured Cycles per Tap (FIRA) or per biquad (IIRA)

FIR/IIR Accelerators Performance

This section illustrates FIR and IIR accelerator performance with different filter parameters and in different processors:

- ADSP-SC59x FIRA/IIRA at maximum ACLK (=CCLK) = 1 GHz
- ADSP-2156x FIRA/IIRA at maximum ACLK (=CCLK) = 1 GHz
- ADSP--SC57x FIRA/IIRA at maximum ACLK (=SCLK0) = 125 MHz and at maximum CCLK = 500 MHz
- ADSP-214xx FIRA/IIRA at maximum ACLK(=PCLK=CCLK/2) = 225 MHz and at maximum CCLK = 450 MHz
- ADSP-2156x (SHARC+) core at maximum CCLK = 1 GHz

The ADSP-SC58x accelerator performance is the same as the ADSP-SC57x accelerator, except for a slight performance loss due to the lack of a DMA burst mode.

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

Additional measurements and analysis discuss how the various factors affect the accelerator MAC Utilization Efficiency (MUE).

FIR Performance

This section contains the following information:

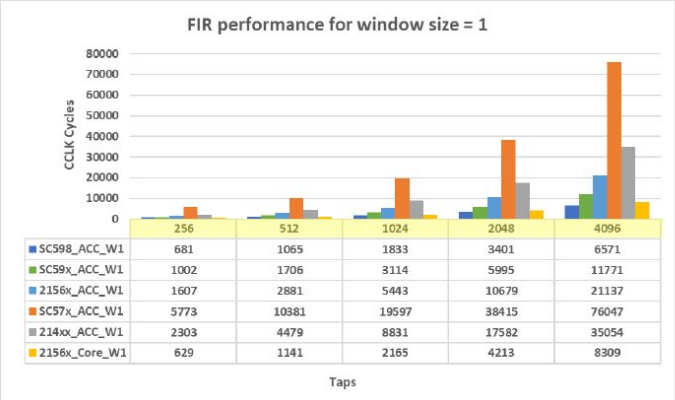
- Measured performance of the ADSP-SC59x FIRA for different filter parameters and comparison with other cases
- Measured FIRA Compute Efficiency (CE) and contributing factors
- The effect of buffer placement (L1, L2, and L3 memory) on FIRA performance

Comparison with ADSP-2156x/SC57x/214xx Accelerators and SHARC+(measured on ADSP-2156x) Core

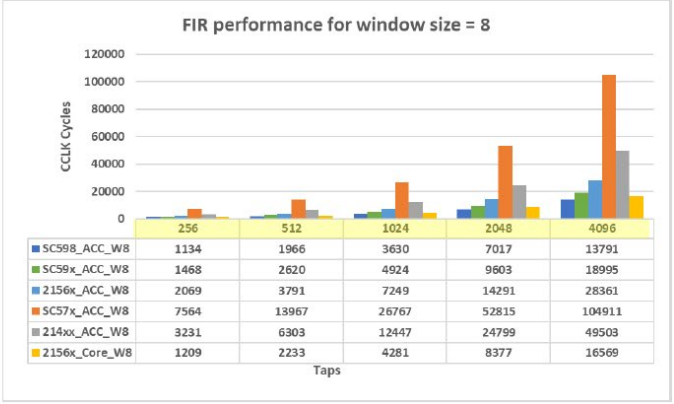
Figure 3 FIR Performance in Core Cycles Across Window Size and Tap Length on page 6 and Figure 4 FIR Performance in Time (µs) Across Window Size and Tap Length on page 7 show the measured FIR performance in core cycles and time units (µs) for the six cases. Figure 5 ADSP-SC59x FIR Performance Improvement Factor for 512 Taps on page 8 shows the performance improvement factor of one ADSP-SC598 FIR accelerator instance at different window sizes and a tap length of 512, in comparison with other cases.

The following figures (Figure 3, Figure 4, and Figure 5 on the next 3 pages) compare the performance of a single accelerator instance on different processors.

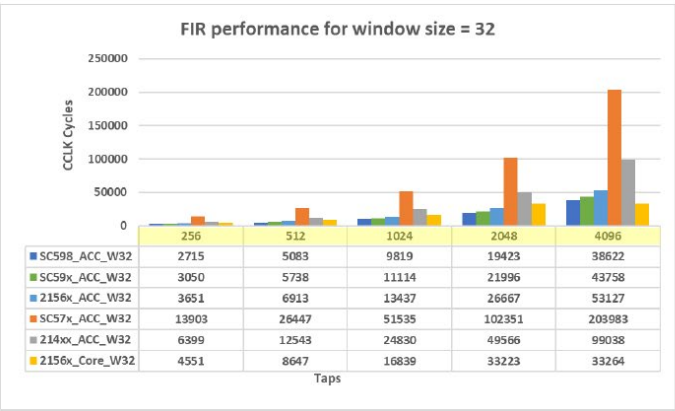
Figure 3 FIR Performance in Core Cycles Across Window Size and Tap Length



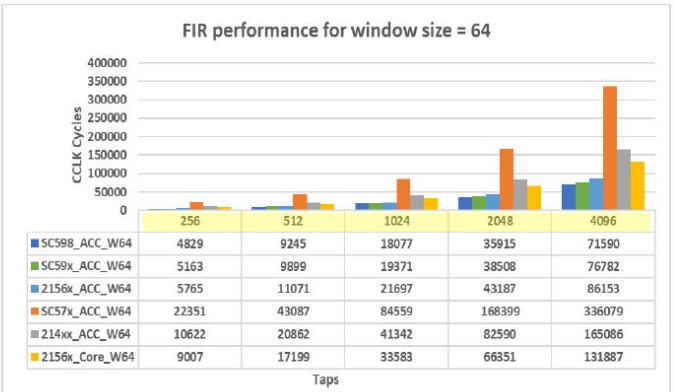
	256	512	1024	2048	4096
SC598_ACC_W1	681	1065	1833	3401	6571
SC59x_ACC_W1	1002	1706	3114	5995	11771
2156x_ACC_W1	1607	2881	5443	10679	21137
SC57x_ACC_W1	5773	10381	19597	38415	76047
214xx_ACC_W1	2303	4479	8831	17582	35054
2156x_Core_W1	629	1141	2165	4213	8309



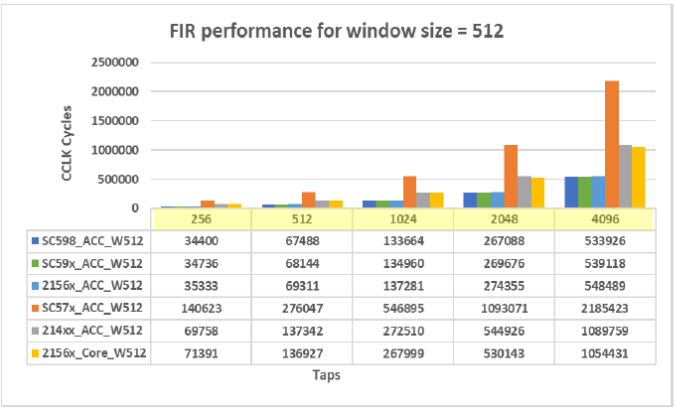
	256	512	1024	2048	4096
SC598_ACC_W8	1134	1966	3630	7017	13791
SC59x_ACC_W8	1468	2620	4924	9603	18995
2156x_ACC_W8	2069	3791	7249	14291	28361
SC57x_ACC_W8	7564	13967	26767	52815	104911
214xx_ACC_W8	3231	6303	12447	24799	49503
2156x_Core_W8	1209	2233	4281	8377	16569



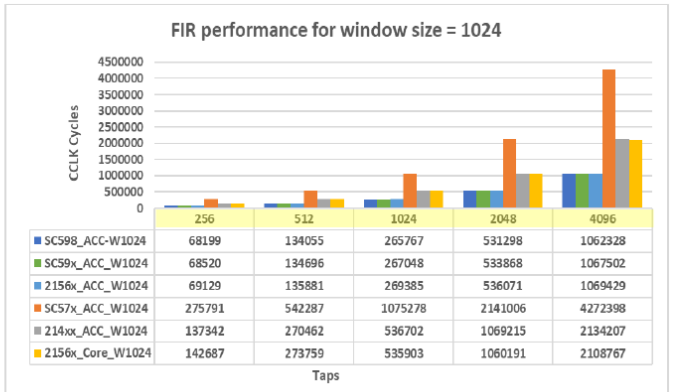
	256	512	1024	2048	4096
SC598_ACC_W32	2715	5083	9819	19423	38622
SC59x_ACC_W32	3050	5738	11114	21996	43758
2156x_ACC_W32	3651	6913	13437	26667	53127
SC57x_ACC_W32	13903	26447	51535	102351	203983
214xx_ACC_W32	6399	12543	24830	49566	99038
2156x_Core_W32	4551	8647	16839	33223	33264



	256	512	1024	2048	4096
SC598_ACC_W64	4829	9245	18077	35915	71590
SC59x_ACC_W64	5163	9899	19371	38508	76782
2156x_ACC_W64	5765	11071	21697	43187	86153
SC57x_ACC_W64	22351	43087	84559	168399	336079
214xx_ACC_W64	10622	20862	41342	82590	165086
2156x_Core_W64	9007	17199	33583	66351	131887

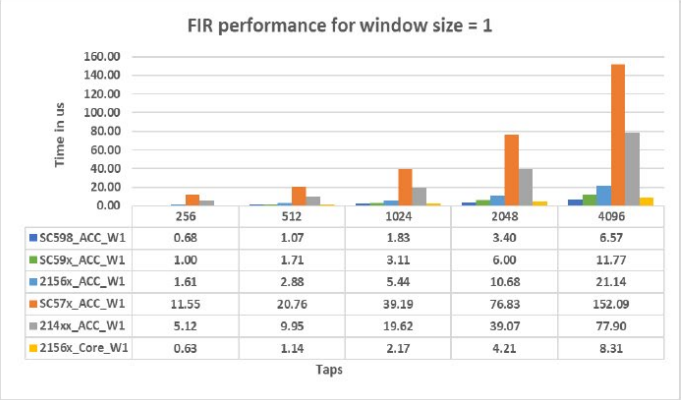


	256	512	1024	2048	4096
SC598_ACC_W512	34400	67488	133664	267088	533926
SC59x_ACC_W512	34736	68144	134960	269676	539118
2156x_ACC_W512	35333	69311	137281	274355	548489
SC57x_ACC_W512	140623	276047	546895	1093071	2185423
214xx_ACC_W512	69758	137342	272510	544926	1089759
2156x_Core_W512	71391	136927	267999	530143	1054431



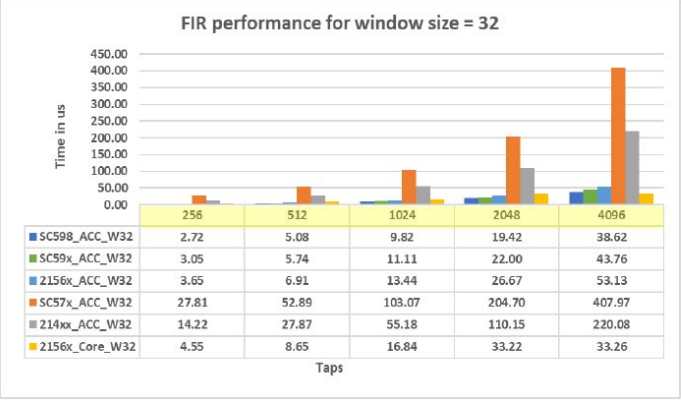
	256	512	1024	2048	4096
SC598_ACC-W1024	68199	134055	265767	531298	1062328
SC59x_ACC_W1024	68520	134696	267048	533868	1067502
2156x_ACC_W1024	69129	135881	269385	536071	1069429
SC57x_ACC_W1024	275791	542287	1075278	2141006	4272398
214xx_ACC_W1024	137342	270462	536702	1069215	2134207
2156x_Core_W1024	142687	273759	535903	1060191	2108767

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

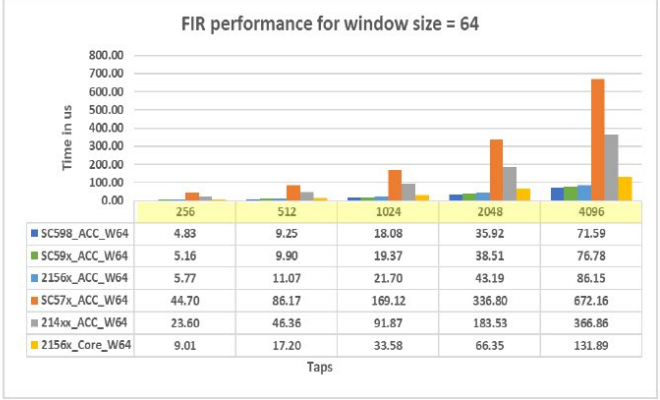


	256	512	1024	2048	4096
SC598_ACC_W1	0.68	1.07	1.83	3.40	6.57
SC59x_ACC_W1	1.00	1.71	3.11	6.00	11.77
2156x_ACC_W1	1.61	2.88	5.44	10.68	21.14
SC57x_ACC_W1	11.55	20.76	39.19	76.83	152.09
214xx_ACC_W1	5.12	9.95	19.62	39.07	77.90
2156x_Core_W1	0.63	1.14	2.17	4.21	8.31

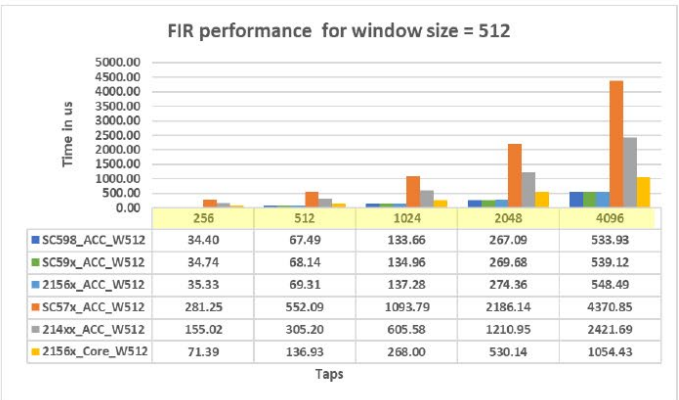
	256	512	1024	2048	4096
SC598_ACC_W8	1.13	1.97	3.63	7.02	13.79
SC59x_ACC_W8	1.47	2.62	4.92	9.60	19.00
2156x_ACC_W8	2.07	3.79	7.25	14.29	28.36
SC57x_ACC_W8	15.13	27.93	53.53	105.63	209.82
214xx_ACC_W8	7.18	14.01	27.66	55.11	110.01
2156x_Core_W8	1.21	2.23	4.28	8.38	16.57



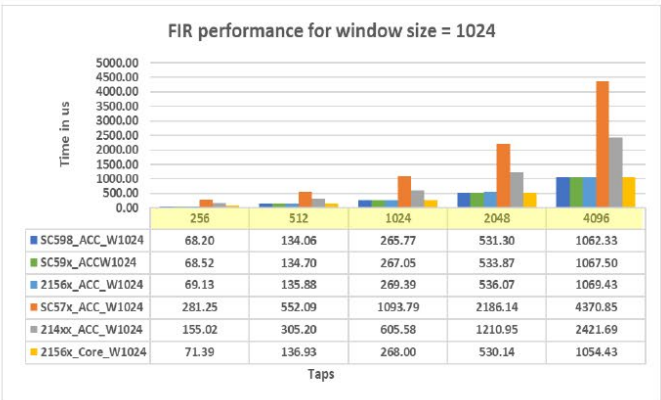
	256	512	1024	2048	4096
SC598_ACC_W32	2.72	5.08	9.82	19.42	38.62
SC59x_ACC_W32	3.05	5.74	11.11	22.00	43.76
2156x_ACC_W32	3.65	6.91	13.44	26.67	53.13
SC57x_ACC_W32	27.81	52.89	103.07	204.70	407.97
214xx_ACC_W32	14.22	27.87	55.18	110.15	220.08
2156x_Core_W32	4.55	8.65	16.84	33.22	33.26



	256	512	1024	2048	4096
SC598_ACC_W64	4.83	9.25	18.08	35.92	71.59
SC59x_ACC_W64	5.16	9.90	19.37	38.51	76.78
2156x_ACC_W64	5.77	11.07	21.70	43.19	86.15
SC57x_ACC_W64	44.70	86.17	169.12	336.80	672.16
214xx_ACC_W64	23.60	46.36	91.87	183.53	366.86
2156x_Core_W64	9.01	17.20	33.58	66.35	131.89

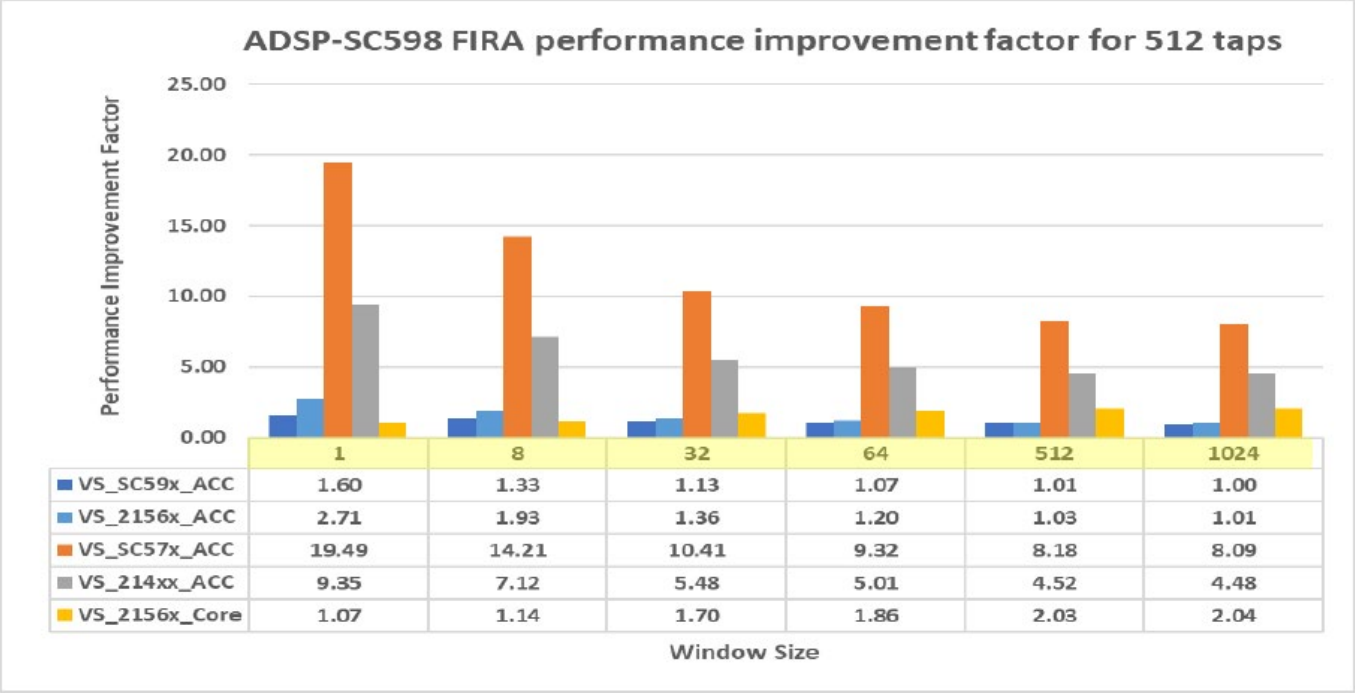


	256	512	1024	2048	4096
SC598_ACC_W512	34.40	67.49	133.66	267.09	533.93
SC59x_ACC_W512	34.74	68.14	134.96	269.68	539.12
2156x_ACC_W512	35.33	69.31	137.28	274.36	548.49
SC57x_ACC_W512	281.25	552.09	1093.79	2186.14	4370.85
214xx_ACC_W512	155.02	305.20	605.58	1210.95	2421.69
2156x_Core_W512	71.39	136.93	268.00	530.14	1054.43



	256	512	1024	2048	4096
SC598_ACC_W1024	68.20	134.06	265.77	531.30	1062.33
SC59x_ACCW1024	68.52	134.70	267.05	533.87	1067.50
2156x_ACC_W1024	69.13	135.88	269.39	536.07	1069.43
SC57x_ACC_W1024	281.25	552.09	1093.79	2186.14	4370.85
214xx_ACC_W1024	155.02	305.20	605.58	1210.95	2421.69
2156x_Core_W1024	71.39	136.93	268.00	530.14	1054.43

Figure 5 ADSP-SC59x FIR Performance Improvement Factor for 512 Taps for Different Window Sizes



	1	8	32	64	512	1024
VS_SC59x_ACC	1.60	1.33	1.13	1.07	1.01	1.00
VS_2156x_ACC	2.71	1.93	1.36	1.20	1.03	1.01
VS_SC57x_ACC	19.49	14.21	10.41	9.32	8.18	8.09
VS_214xx_ACC	9.35	7.12	5.48	5.01	4.52	4.48
VS_2156x_Core	1.07	1.14	1.70	1.86	2.03	2.04

The following observations are derived from Figure 5:

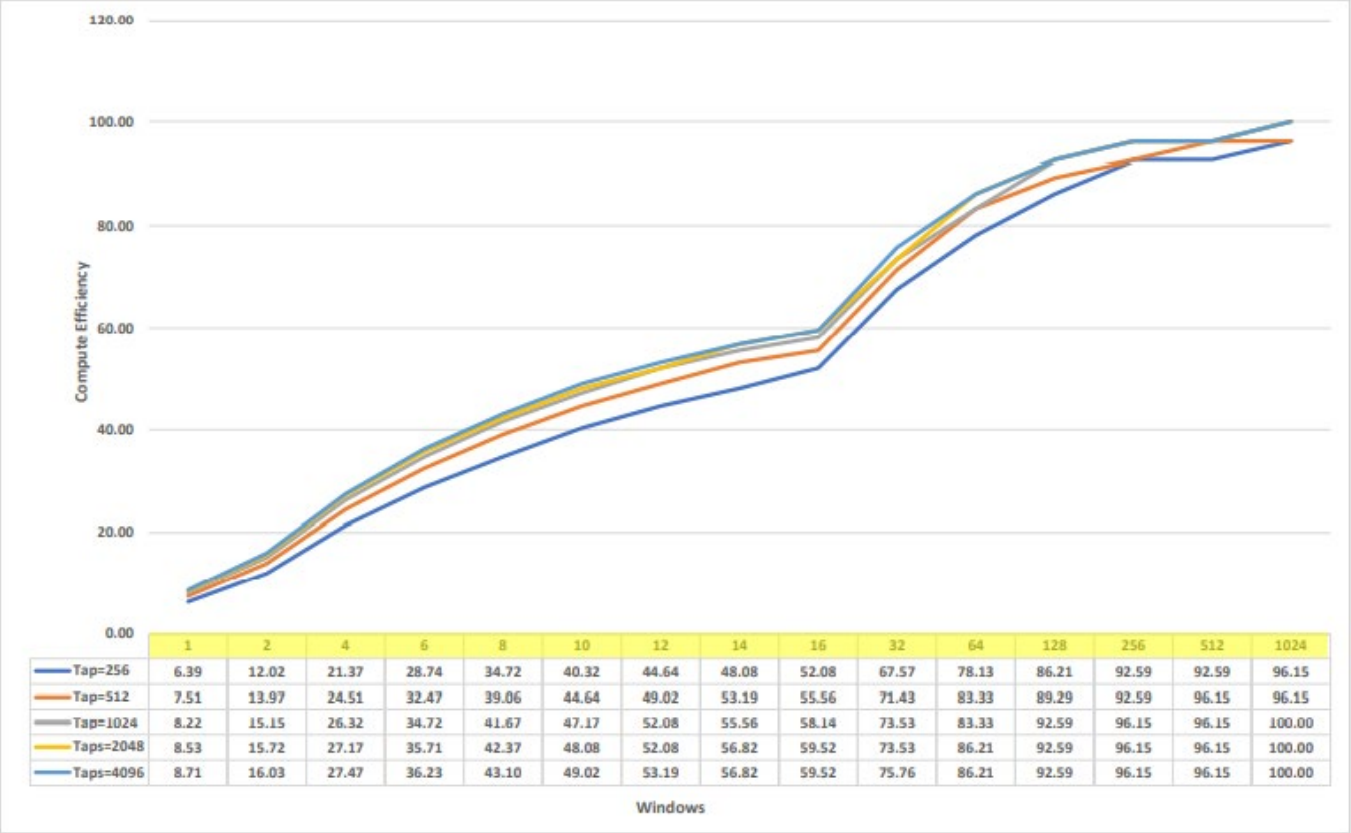
- **The ADSP-SC598 FIRA outperforms the ADSP-SC59x/2159x for small window sizes**. The improvement primarily results from the reduced ADSP-SC598 FIRA DMA overhead. The DMA load time accounts for a large portion of the total processing time in the small window cases, whereas the DMA load time is insignificant in the large window cases.
- **The ADSP-SC598 FIRA outperforms the ADSP-2156x for small window sizes**. The improvement primarily results from the reduced ADSP-SC598 FIRA DMA overhead. The DMA load time accounts for a large portion of the total processing time in the small window cases whereas the DMA load time is insignificant in the large window cases.
- **The ADSP-SC598 FIRA performance is approximately 8 times that of ADSP-SC57x FIRA**. The improvement is because the ADSP-SC598 FIRA runs at CCLK (1 GHz), whereas the ADSP-SC57x FIRA runs at SCLK, which is a maximum of CCLK/4 (125 MHz). For small window sizes, the performance factor increases due to the enhancements to reduce the DMA load time on ADSP-SC598 FIRA.
- **The ADSP-SC598 FIRA performance is approximately 4.46 times that of the ADSP-214xx FIRA.** The improvement is because the ADSP-SC598 FIRA runs at CCLK (1 GHz), whereas the ADSP-214xx FIRA runs at PCLK, which is a maximum of CCLK/2 (225 MHz). For small window sizes, the performance factor increases because of the enhancements that reduce the DMA load time for the ADSP-SC598 FIRA.
- **The ADSP-SC598 FIRA performance is approximately two times that of the ADSP-2156x SHARC+ core's FIR code execution**. The improvement is because the **ADSP-SC598 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)

Figure 6 shows how the Compute Efficiency (CE) percentage for the FIRA varies with window size and tap length. The CE approximates 100% for large (>128) window sizes. The CE significantly decreases for smaller window sizes. Compared to previous processors, the CE for smaller window sizes are significantly better because of the DMA load time improvement in ADSP-SC59x FIRA. For a constant window size, the CE percentage does not vary significantly for different tap lengths.

Figure 6 FIRA Compute Efficiency (CE) for Different Tap Lengths and Window Sizes

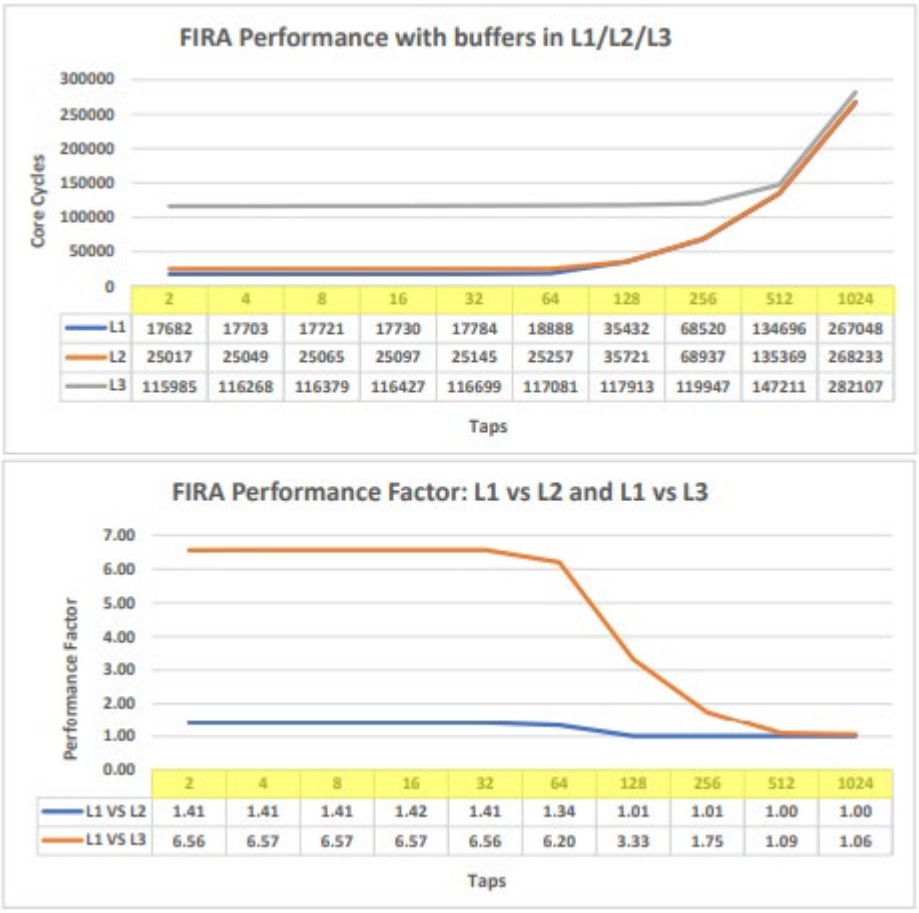


	1	2	4	6	8	10	12	14	16	32	64	128	256	512	1024
Tap=256	6.39	12.02	21.37	28.74	34.72	40.32	44.64	48.08	52.08	67.57	78.13	86.21	92.59	92.59	96.15
Tap=512	7.51	13.97	24.51	32.47	39.06	44.64	49.02	53.19	55.56	71.43	83.33	89.29	92.59	96.15	96.15
Tap=1024	8.22	15.15	26.32	34.72	41.67	47.17	52.08	55.56	58.14	73.53	83.33	92.59	96.15	96.15	100.00
Taps=2048	8.53	15.72	27.17	35.71	42.37	48.08	52.08	56.82	59.52	73.53	86.21	92.59	96.15	96.15	100.00
Taps=4096	8.71	16.03	27.47	36.23	43.10	49.02	53.19	56.82	59.52	75.76	86.21	92.59	96.15	96.15	100.00

Buffer Placement

Figure 7 ADSP-SC59x FIRA Performance Comparison for Buffers in L1, L2, and L3 Memory on page 10 shows how the core cycles change with increased tap lengths when the input/output/coefficient buffers are placed in L1/L2/L3 memory for a window size of 1024. It also shows a plot of the performance factor when the buffers are in L1 memory, versus in L2 and L3 memory.

Figure 7 ADSP-SC59x FIRA Performance Comparison for Buffers in L1, L2, and L3 Memory



	2	4	8	16	32	64	128	256	512	1024
L1	17682	17703	17721	17730	17784	18888	35432	68520	134696	267048
L2	25017	25049	25065	25097	25145	25257	35721	68937	135369	268233
L3	115985	116268	116379	116427	116699	117081	117913	119947	147211	282107

	2	4	8	16	32	64	128	256	512	1024
L1 VS L2	1.41	1.41	1.41	1.42	1.41	1.34	1.01	1.01	1.00	1.00
L1 VS L3	6.56	6.57	6.57	6.57	6.56	6.20	3.33	1.75	1.09	1.06

The following observations are derived from the Figure 7 performance factor graphs:

- As the tap length increases for a constant window size, the core cycles remain constant until a certain point (N), then increase proportionally. For small tap lengths, the input/output DMA cycles dominate the compute cycles.
- As the memory access latency increases from L1 to L2 memory and then further to L3 memory, the value of certain point (N) increases, and the performance factor approaches 1.0. Placing buffers in L2 and L3 memory instead of L1 memory might be expensive for tap lengths less than certain point (N) and comparable for tap lengths greater than or equal to certain point(N).

IIR Performance

This section contains the following topics:

- Measured performance of the ADSP-SC59x IIRA for different filter parameters and comparison with other cases
- Measured IIRA Compute Efficiency (CE) and contributing factors
- The effect of buffer placement (L1, L2, and L3 memory) on IIRA performance

Comparison with ADSP-2156x/SC57x/214xx Accelerators and SHARC+ (measured on ADSP-2156x) Core

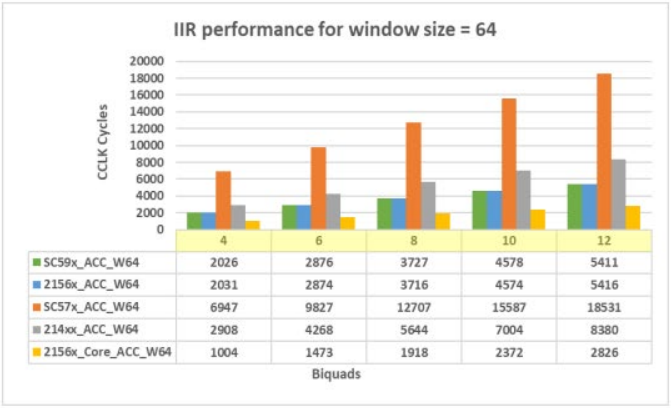
Figure 8 and Figure 9 show the measured IIR performance of one IIRA instance in core cycles and time units (µs), respectively, for all the cases. Figure 10 ADSP-SC59x IIR Performance Improvement Factor for 6 Biquads Considering Time on page 13 shows the performance factor of the ADSP-SC9x IIRA, as compared to the other cases for different window sizes and a constant value of six biquads.

Figure 8 IIR Performance in Core Cycles Across Window Size and Biquad Values

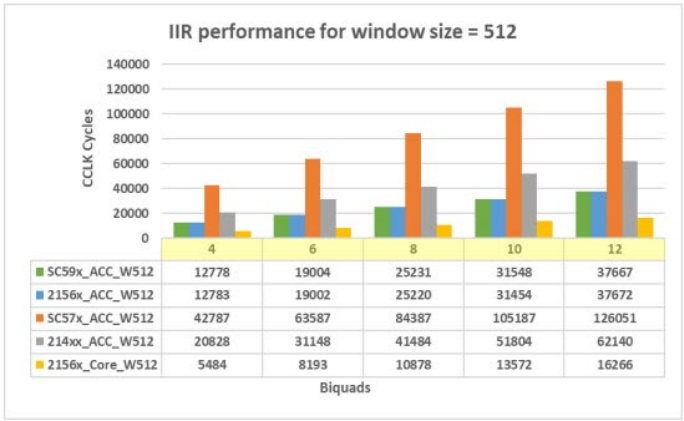
	4	6	8	10	12
SC59x_ACC_W1	522	604	703	785	884
2156x_ACC_W1	519	606	692	794	880
SC57x_ACC_W1	1891	2275	2659	3043	3363
214xx_ACC_W1	380	492	604	716	812
2156x_Core_W1	189	199	211	223	235

	4	6	8	10	12
SC59x_ACC_W8	682	860	1039	1218	1379
2156x_ACC_W8	687	858	1028	1214	1384
SC57x_ACC_W8	2467	3107	3747	4387	5091
214xx_ACC_W8	668	908	1164	1404	1660
2156x_Core_W8	444	633	798	872	1029

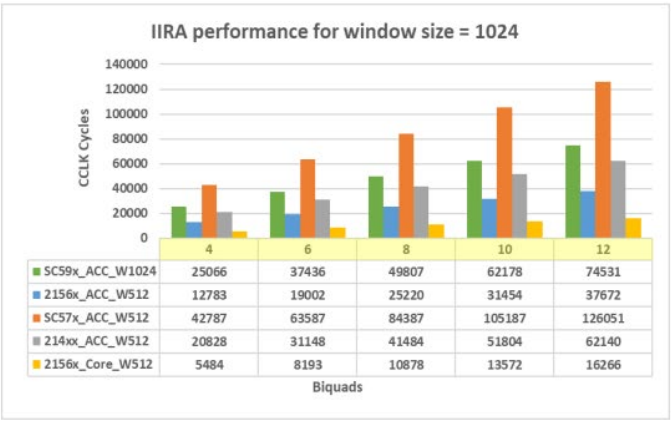
	4	6	8	10	12
SC59x_ACC_W32	1261	1726	2191	2641	3106
2156x_ACC_W32	1261	1726	2191	2641	3106
SC57x_ACC_W32	4387	5987	7587	9187	10851
214xx_ACC_W32	1628	2348	3084	3804	4540
2156x_Core_W32	684	993	1278	1572	1866



	4	6	8	10	12
SC59x_ACC_W64	2026	2876	3727	4578	5411
2156x_ACC_W64	2031	2874	3716	4574	5416
SC57x_ACC_W64	6947	9827	12707	15587	18531
214xx_ACC_W64	2908	4268	5644	7004	8380
2156x_Core_ACC_W64	1004	1473	1918	2372	2826



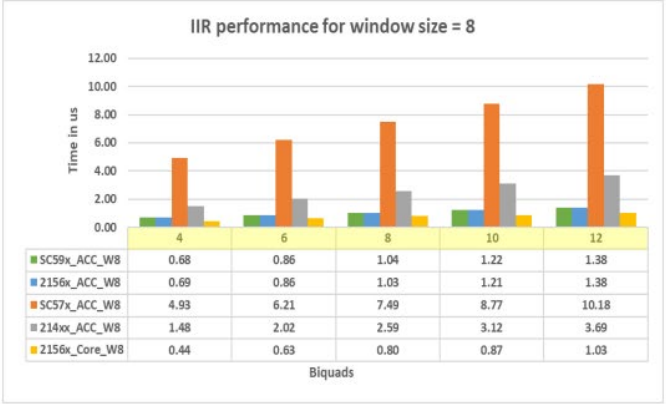
	4	6	8	10	12
SC59x_ACC_W512	12778	19004	25231	31548	37667
2156x_ACC_W512	12783	19002	25220	31454	37672
SC57x_ACC_W512	42787	63587	84387	105187	126051
214xx_ACC_W512	20828	31148	41484	51804	62140
2156x_Core_W512	5484	8193	10878	13572	16266



	4	6	8	10	12
SC59x_ACC_W1024	25066	37436	49807	62178	74531
2156x_ACC_W512	12783	19002	25220	31454	37672
SC57x_ACC_W512	42787	63587	84387	105187	126051
214xx_ACC_W512	20828	31148	41484	51804	62140
2156x_Core_W512	5484	8193	10878	13572	16266

Figure 9 IIR Performance in Time (µs) Across Window Sizes and Number of Biquads

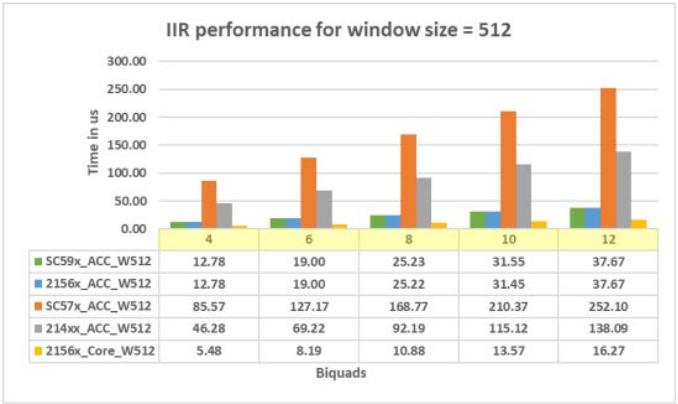
	4	6	8	10	12
SC59x_ACC_W1	0.52	0.60	0.70	0.79	0.88
2156x_ACC_W1	0.52	0.61	0.69	0.79	0.88
SC57x_ACC_W1	3.78	4.55	5.32	6.09	6.73
214xx_ACC_W1	0.84	1.09	1.34	1.59	1.80
2156x_Core_W1	0.19	0.20	0.21	0.22	0.24



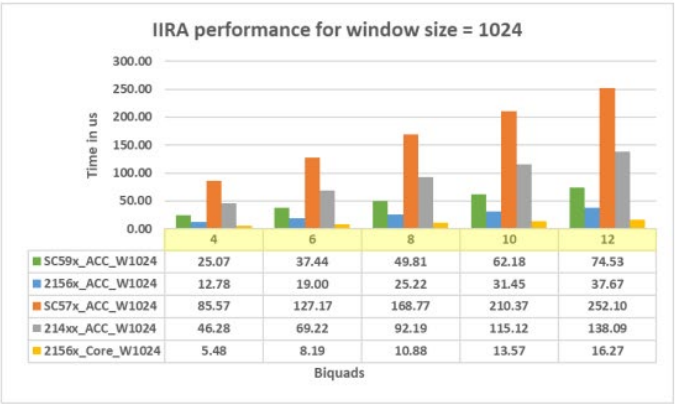
	4	6	8	10	12
SC59x_ACC_W8	0.68	0.86	1.04	1.22	1.38
2156x_ACC_W8	0.69	0.86	1.03	1.21	1.38
SC57x_ACC_W8	4.93	6.21	7.49	8.77	10.18
214xx_ACC_W8	1.48	2.02	2.59	3.12	3.69
2156x_Core_W8	0.44	0.63	0.80	0.87	1.03

	4	6	8	10	12
SC59x_ACC_W32	1.26	1.72	2.19	2.66	3.11
2156x_ACC_W32	1.26	1.73	2.19	2.64	3.11
SC57x_ACC_W32	8.77	11.97	15.17	18.37	21.70
214xx_ACC_W32	3.62	5.22	6.85	8.45	10.09
2156x_Core_W32	0.68	0.99	1.28	1.57	1.87

	4	6	8	10	12
SC59x_ACC_W64	2.03	2.88	3.73	4.58	5.41
2156x_ACC_W64	2.03	2.87	3.72	4.57	5.42
SC57x_ACC_W64	13.89	19.65	25.41	31.17	37.06
214xx_ACC_W64	6.46	9.48	12.54	15.56	18.62
2156x_Core_W64	1.00	1.47	1.92	2.37	2.83



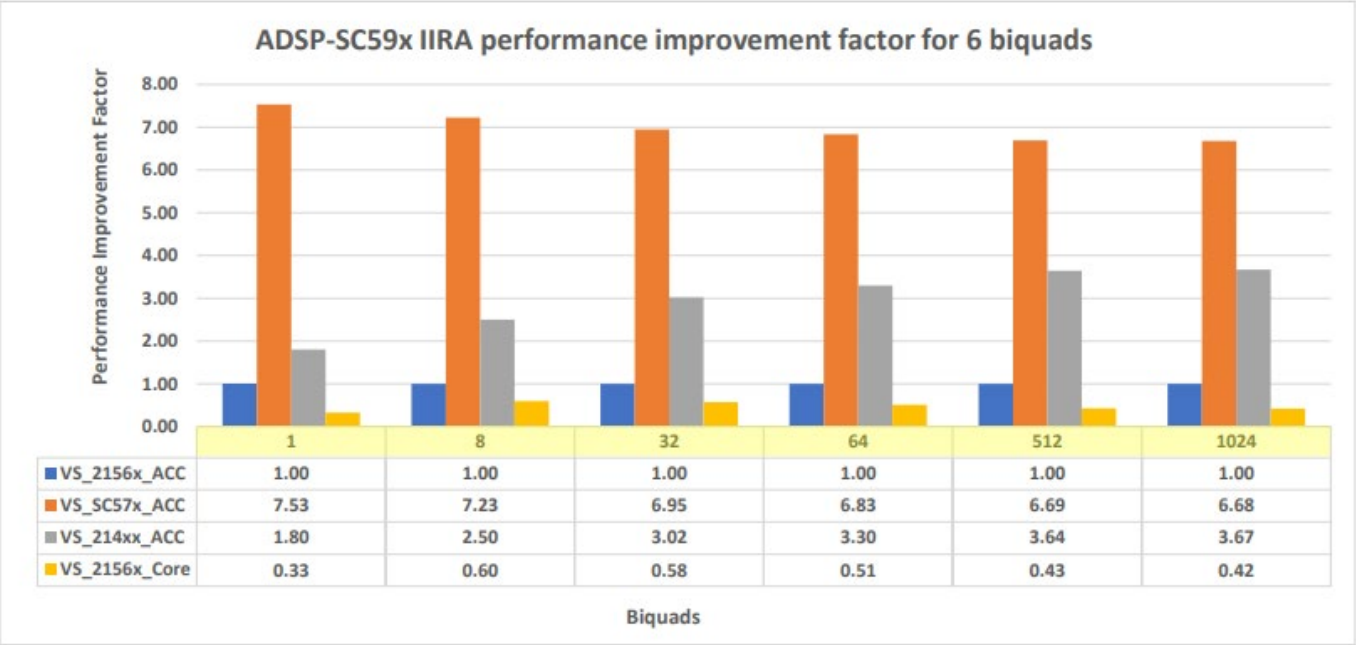
	4	6	8	10	12
SC59x_ACC_W512	12.78	19.00	25.23	31.55	37.67
2156x_ACC_W512	12.78	19.00	25.22	31.45	37.67
SC57x_ACC_W512	85.57	127.17	168.77	210.37	252.10
214xx_ACC_W512	46.28	69.22	92.19	115.12	138.09
2156x_Core_W512	5.48	8.19	10.88	13.57	16.27



	4	6	8	10	12
SC59x_ACC_W1024	25.07	37.44	49.81	62.18	74.53
2156x_ACC_W1024	12.78	19.00	25.22	31.45	37.67
SC57x_ACC_W1024	85.57	127.17	168.77	210.37	252.10
214xx_ACC_W1024	46.28	69.22	92.19	115.12	138.09
2156x_Core_W1024	5.48	8.19	10.88	13.57	16.27

The Figure 9 values compare the performance of a single accelerator instance on different processors. The performance of ADSP-SC598 IIRA is similar to the ADSP-SC59x/2159x IIRA performance. Hence, the ADSP-SC59x/2159x IIRA values shown are also applicable to the ADSP-SC598 IIRA.

Figure 10 ADSP-SC59x IIR Performance Improvement Factor for 6 Biquads Considering Time



	1	8	32	64	512	1024
VS_2156x_ACC	1.00	1.00	1.00	1.00	1.00	1.00
VS_SC57x_ACC	7.53	7.23	6.95	6.83	6.69	6.68
VS_214xx_ACC	1.80	2.50	3.02	3.30	3.64	3.67
VS_2156x_Core	0.33	0.60	0.58	0.51	0.43	0.42

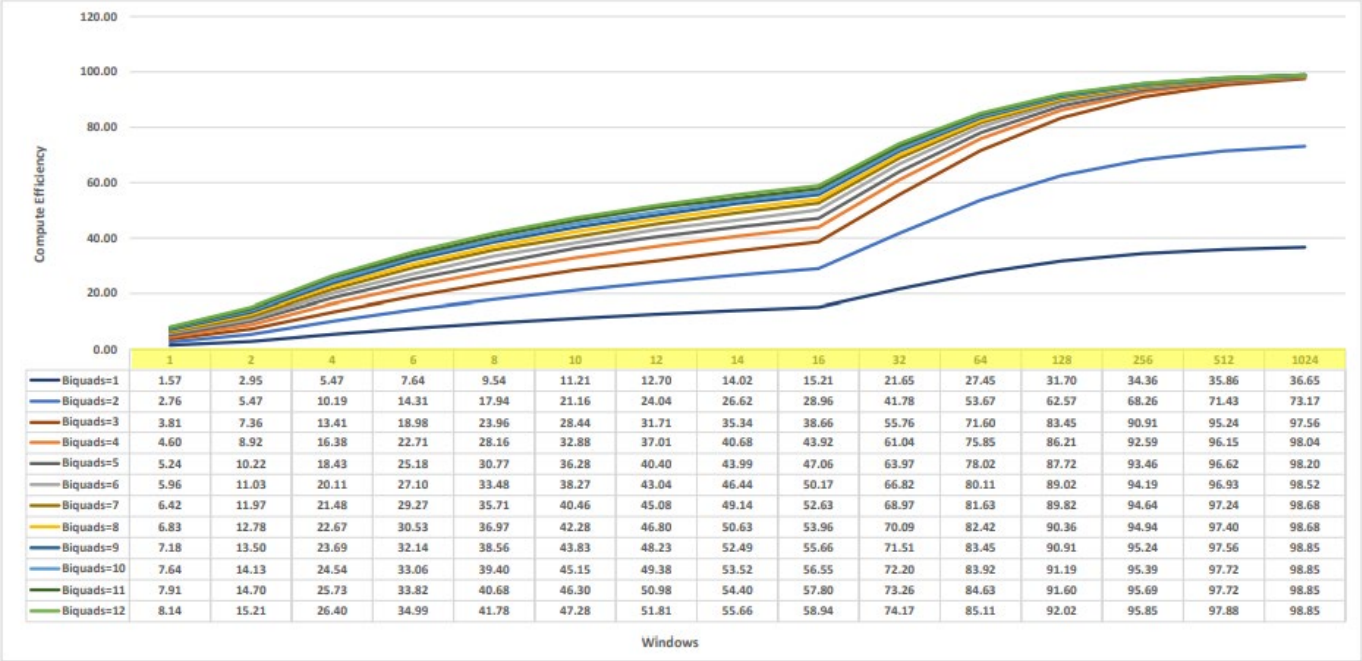
The following observations are derived from the IIR performance values shown in Figure 10:

- The ADSP-SC59x IIRA performance is similar to the ADSP-2156x IIRA performance. Both ADSP-SC59x IIRA and ADSP-2156x IIRA run at CCLK (1 GHz).
- **The ADSP-SC59x IIRA performance is approximately 6.7 times the ADSP-SC57x IIRA performance.** The ADSP-SC59x IIRA runs at CCLK (1 GHz), whereas the ADSP-SC57x IIRA runs at SCLK, which is a maximum of CCLK/4 (125 MHz). The reduction in the improvement factor from 8 to 6.7, compared to the FIRA, is because the ADSP-SC59x IIRA compute unit takes 6 cycles per biquad. The ADSP-SC57x IIRA takes 5 cycles per biquad.
- **The ADSP-SC59x IIRA performance is approximately 3.7 times the ADSP-214xx IIRA performance.** The ADSP-2156x IIRA runs at CCLK (1 GHz), whereas the ADSP-214xx IIRA runs at PCLK, which is a maximum of CCLK/2 (225 MHz). The reduction in the improvement factor from 4.44 to 3.7, compared to the FIRA, is because the ADSP-2156x IIRA compute unit takes 6 cycles per biquad. The ADSP-214xx IIRA takes 5 cycles per biquad.
- **The ADSP-SC59x IIRA performance is approximately 0.42 times the ADSP-2156x SHARC+ core's execution of IIR code**. Each ADSP-SC59x IIRA instance has one MAC unit, whereas each SHARC+ core has two MAC units. Additionally, the IIRA compute unit takes 6 cycles per biquad. The SHARC+ core takes 2.5 cycles.

Compute Efficiency

Figure 11 shows how the Compute Efficiency (CE) percentage for IIRA varies with the window size and number of biquads. As can be seen in the plot, the CE approximates 100% for larger window sizes, and it significantly reduces for small window sizes. For a constant window size, the CE percentage does not vary significantly with the number of biquads, except when the biquad count is one or two. This is because the input and output DMA cycles dominate the total processing cycles for a low biquad count.

Figure 11 IIRA Compute Efficiency (CE) for Different Filter Parameters



	1	2	4	6	8	10	12	14	16	32	64	128	256	512	1024
Biquads=1	1.57	2.95	5.47	7.64	9.54	11.21	12.70	14.02	15.21	21.65	27.45	31.70	34.36	35.86	36.65
Biquads=2	2.76	5.47	10.19	14.31	17.94	21.16	24.04	26.62	28.96	41.78	53.67	62.57	68.26	71.43	73.17
Biquads=3	3.81	7.36	13.41	18.98	23.96	28.44	31.71	35.34	38.66	55.76	71.60	83.45	90.91	95.24	97.56
Biquads=4	4.60	8.92	16.38	22.71	28.16	32.88	37.01	40.68	43.92	61.04	75.85	86.21	92.59	96.15	98.04
Biquads=5	5.24	10.22	18.43	25.18	30.77	36.28	40.40	43.99	47.06	63.97	78.02	87.72	93.46	96.62	98.20
Biquads=6	5.96	11.03	20.11	27.10	33.48	38.27	43.04	46.44	50.17	66.82	80.11	89.02	94.19	96.93	98.52
Biquads=7	6.42	11.97	21.48	29.27	35.71	40.46	45.08	49.14	52.63	68.97	81.63	89.82	94.64	97.24	98.68
Biquads=8	6.83	12.78	22.67	30.53	36.97	42.28	46.80	50.63	53.96	70.09	82.42	90.36	94.94	97.40	98.68
Biquads=9	7.18	13.50	23.69	32.14	38.56	43.83	48.23	52.49	55.66	71.51	83.45	90.91	95.24	97.56	98.85
Biquads=10	7.64	14.13	24.54	33.06	39.40	45.15	49.38	53.52	56.55	72.20	83.92	91.19	95.39	97.72	98.85
Biquads=11	7.91	14.70	25.73	33.82	40.68	46.30	50.98	54.40	57.80	73.26	84.63	91.60	95.69	97.72	98.85
Biquads=12	8.14	15.21	26.40	34.99	41.78	47.28	51.81	55.66	58.94	74.17	85.11	92.02	95.85	97.88	98.85

e

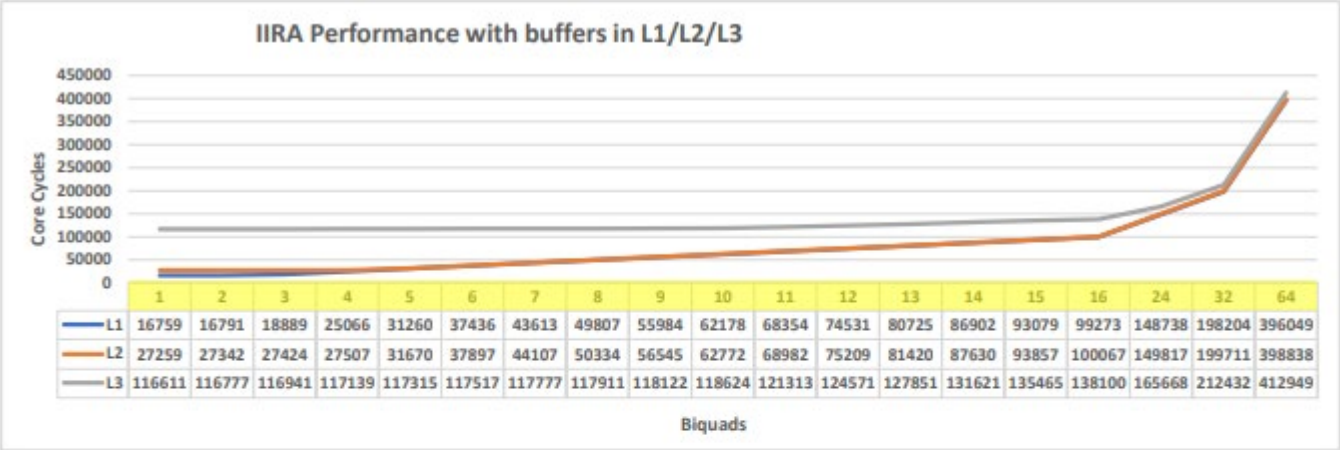
Buffer Placement

Figure 12 ADSP-SC59x IIRA Performance Comparison for Buffers in L1, L2, and L3 Memory on page 15 shows how the performance measured in terms of core cycles change when the input/output/coefficient buffers are placed in L1/L2/L3 memory with a window size equal to 1024 and the number of biquads is increased. It also shows a plot of the performance factor when the buffers are in L1 memory, compared to when they are in L2 or L3 memory.

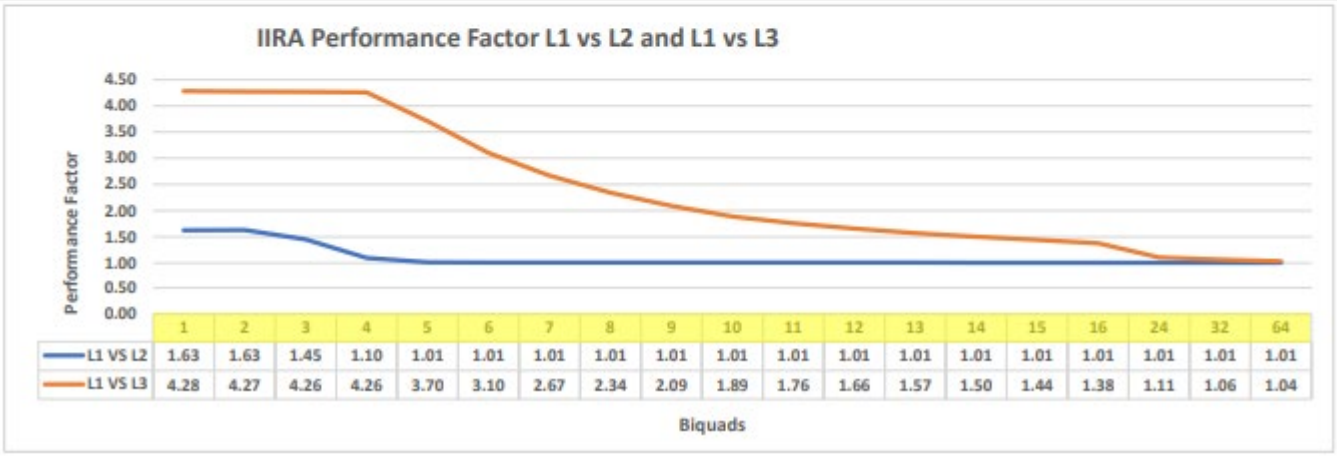
The following observations are derived from the Figure 12 CE graphs:

- As the number of biquads increases, the core cycles do not increase significantly until a point (N), at which the core cycles start increasing proportionally. This change happens because the input and output DMA cycles dominate the compute cycles when the biquad count is low.
- As the memory access latency increases from L1 to L2 memory, and then further to L3 memory, the N value increases, and the performance factor approximates 1.0, as the number of biquads becomes large. Placing buffers in L2 and L3 memory, instead of L1 memory, might be expensive for biquad counts less than N; and comparable for biquad counts greater than or equal to N.

Figure 12 ADSP-SC59x IIRA Performance Comparison for Buffers in L1, L2, and L3 Memory



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	24	32	64
L1	16759	16791	18889	25066	31260	37436	43613	49807	55984	62178	68354	74531	80725	86902	93079	99273	148738	198204	396049
L2	27259	27342	27424	27507	31670	37897	44107	50334	56545	62772	68982	75209	81420	87630	93857	100067	149817	199711	398838
L3	116611	116777	116941	117139	117315	117517	117777	117911	118122	118624	121313	124571	127851	131621	135465	138100	165668	212432	412949

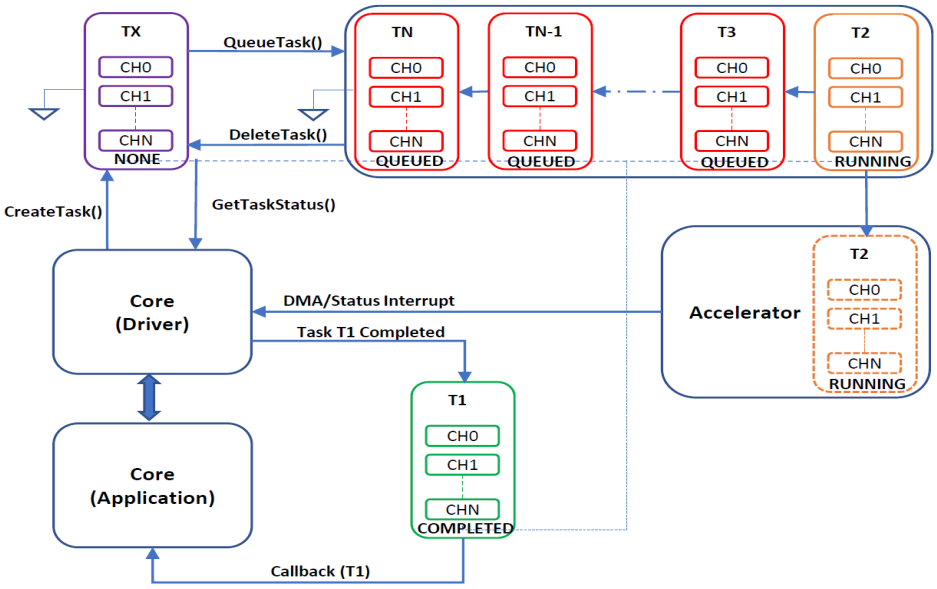


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	24	32	64
L1 VS L2	1.63	1.63	1.45	1.10	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01
L1 VS L3	4.28	4.27	4.26	4.26	3.70	3.10	2.67	2.34	2.09	1.89	1.76	1.66	1.57	1.50	1.44	1.38	1.11	1.06	1.04

Programming the Accelerators

The ADSP-SC59x accelerators can be programmed using the device drivers available with the CrossCore® Embedded Studio (CCES) software.

Figure 13 ADSP-SC59x Accelerator Device Driver Structure



Each ADSP SC59x processor has multiple instances of a FIRA(2) and IIRA(8). The SHARC+ cores can access all FIRA/IIRA instances. To minimize the software complexity of accessing those accelerator instances in parallel by the SHARC+ cores, a static configuration scheme can be employed. In this scheme, all accelerator instances are assigned to one SHARC+ core or divided into two groups for the cores. With this organization, both SHARC+ cores can operate independently in parallel using the assigned accelerator instances for processing.

The programming model is the same for the FIR and IIR accelerators. Figure 13 shows the accelerator drivers adhere to a queueing programming model. The model uses abstractions of task and queues. A *task* contains a set of *channels* that are processed sequentially when given to the accelerator,. A *queue* is a set of tasks the accelerator processes sequentially and maintains on a first-in, first-processed basis. Applications can create tasks using the `adi_fir(/iir)_CreateTask()` call and queue those tasks using the `adi_fir(/iir)_QueueTask()` call.

The driver maintains a processing queue of the tasks given or queued by the application for the accelerator and sequentially schedules the tasks to the accelerator. The applications can register a user-defined callback with a `adi_fir(/iir)_RegisterCallback()` call. The driver uses the callback to notify the application when the tasks and channels complete their work.

The driver maintains the status of all the application tasks it has queued. The application can query the status of the task (when necessary) using the `adi_fir(/iir)_GetFir(/Iir)TaskStatus()` call. Once the accelerator completes the queued task, it is removed from the processing queue. The driver maintains all task information for the tasks which are completed or created. The application can later reuse the created tasks by re-queuing them with the `adi_fir(/iir)_QueueTask()` call.

The accelerator can be configured using a static configuration to operate in legacy mode or Auto Configuration Mode (ACM). In legacy mode, the driver maintains a software queue (linked list) of the tasks queued by the application and sequentially schedules the tasks to the accelerator. In ACM, the driver software uses the hardware halt feature to implement a hardware queuing mechanism. Additional channel-customized features exist in the ACM, such as interrupts and triggers. For more feature details, see the `ADI_FIR(/IIR)_CHANNEL_INFO` topic in the *ADSP-SC59x SSDD API Reference Manual for SHARC+ Core* [2] of the CCES. The driver programming model is the same for legacy and ACM operation.

The `FIR_Multi_Channel_Processing` and `IIR_Multi_Channel_Processing` source code examples provided with this application note [3] (also available with the ADSP-SC594 EZ-Kit *Board Support Package*), can be used as a reference for the ADSP-SC59x accelerator device drivers.

Table 1 summarizes the accelerator driver benchmarks captured using the ADSP-SC594 EZ-Kit evaluation board. The source code examples used to obtain these performance measurements, `ADSP_SC59x_FIRA_Driver_Benchmark` and `ADSP_SC59x_IIRA_Driver_Benchmark`, can be found in the ZIP file associated with the EE436 application note [3]. The measured values obtained were done with all buffers in L1 memory.

Table 1: ADSP-SC59x Accelerator Driver Benchmarks

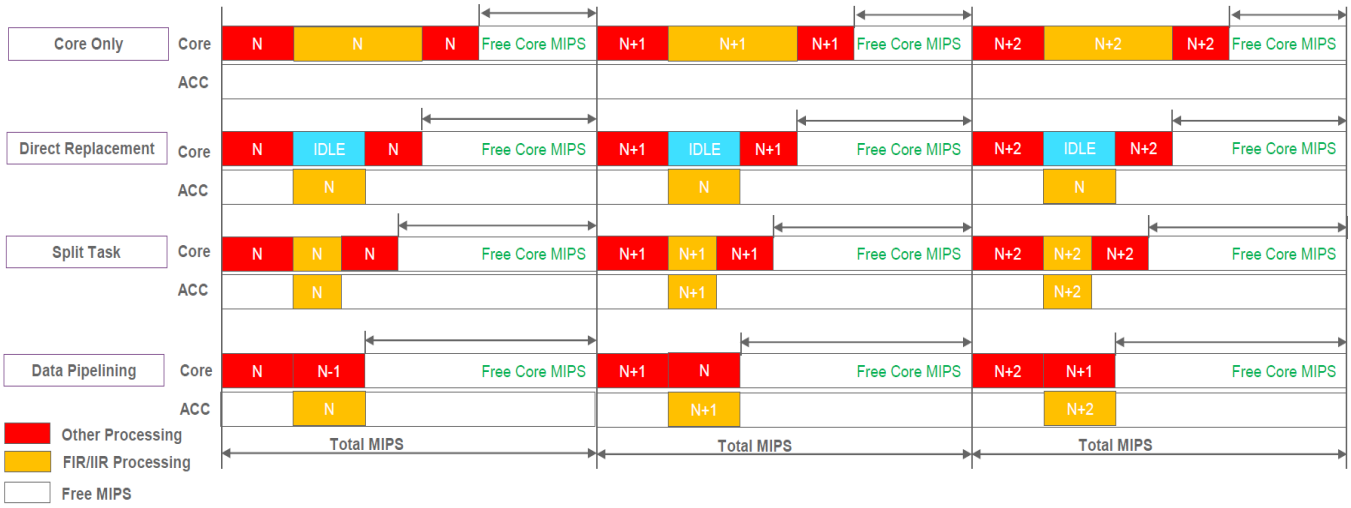
Operation	**Description**	**Measured Core Cycles**			
		FIRA		**IIRA**	
		Legacy	**ACM**	**Legacy**	**ACM**
CreateTask()	Constant Overhead	89	149	151	125
	Per Channel Overhead	335	347	332	353
QueueTask()	Pushing a task to the empty queue	350	536	355	537
	Pushing a task to the non-empty queue	194	445	189	436
Interrupt Overhead (Round trip cycles include the SEC interrupt dispatcher latency.)	For a single task in queue	495	456	500	458
	For more than one task in queue	623	646	645	622

Additional overhead for cache flushing and invalidation can occur when the buffers are placed in L2 and L3 memories. Analog Devices recommends placing the buffers in L1 memory to avoid cache overhead. The Table 1 numbers are also applicable to the ADSP-SC598 FIRA/IIRA accelerators.

Accelerator Usage Models

This section describes models that optimally use the accelerator for different application scenarios. The optimum solution is to offload all FIR and IIR tasks from the SHARC+ cores to the accelerators and allow the cores to do other parallel processing. This solution may not be feasible in some scenarios, particularly when the core needs to use the output from the accelerator for additional processing and has no other independent tasks to finish in parallel. The EE-436 application note discusses the four accelerator usage models (Core Only, Direct Replacement, Split Task, and Data Pipelining) shown in Figure 14.

Figure 14 Accelerator Usage Models



Direct Replacement

Direct Replacement is the most straightforward and easy-to-use method. The core FIR and IIR processing is directly replaced by the accelerator. The core simply waits for the accelerator to finish the job. This model is useful only when the FIRA is employed, because it is faster than the core.

Split Task

Split Task divides the overall FIR and IIR processing between the core and the accelerator. This model is useful when there are multiple channels available for parallel processing. Based on a timing estimation, Split Task divides the total number of channels between the core and the accelerator so that both finish processing at approximately the same time. In some scenarios, this approach yields better results than the Direct Replacement approach. The Split Task model uses the core idle time to finish the overall FIR and IIR processing tasks faster than the accelerator working alone.

Data Pipelining

In this model, data flow between the core and accelerator can be pipelined in such a way that both can work in parallel on different data frames. As shown in Figure 14 Accelerator Usage Models on page 18, the core processes the Nth frame and then initiates the accelerator's processing of this frame. The core continues in parallel to further process the $(N-1)^{th}$ frame output produced by the accelerator in the previous frame. This sequence provides for the complete offloading of the FIR/IIR processing task to the accelerator, at the cost of additional output latency. The pipeline stages and consequently the output latency, can increase depending upon the number of such FIR/IIR processing stages in the processing chain.

Table 2 compares the model approaches and comments on the performance gains and constraints.

Table 2: Accelerator Usage Approach Comparison

Approach	Description	Performance Advantage	Performance Constraints
Direct Replacement	Core processing is replaced by the accelerator. Core waits until the accelerator completes the job.	+	Suitable only for FIRA, as it is faster than the core.
Split Task	FIR/IIR processing of multiple channels is split between the core and the accelerator.	++	Cannot be used for single-channel processing.
Data Pipelining	Data between the core and accelerator is pipelined to allow the core and accelerator to run in parallel	+++	Costs additional latency of at least one frame.

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

Figure 15 FIR/IIR Real-Time Use Case on page 20 provides 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).
- Each IIR band is an 8^{th} order IIR filter, implemented with four-cascaded biquad stages. Table 3: IIR Band Specifications on page 20 describes the IIR band details. The output of each band is multiplied with the respective gain (from 0.0 minimum through 1.0 maximum) and added together to generate the final output. The *IIRGains* flag can be set (bypass) or cleared (enable processing) dynamically using the CCES memory browser.

Figure 15 FIR/IIR Real-Time Use Case

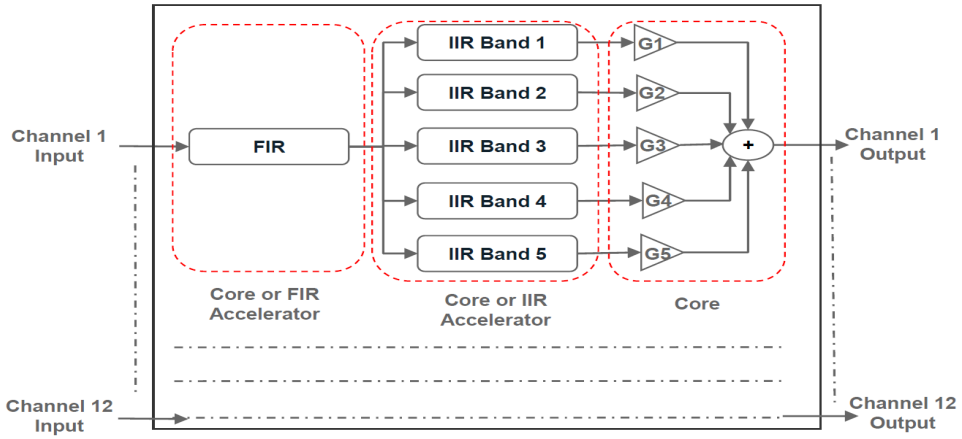


Table 3: IIR Band Specifications

IIR Band Number	Lower Cutoff in Hz	Higher Cutoff in Hz	Filter Type
1	-	250	Low Pass
2	250	500	Band Pass
3	500	2,000	Band Pass
4	2,000	4,000	Band Pass
5	4,000	-	High Pass

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

The `Direct_Replacement`, `Pipelined`, and `Split_Task example code` is provided with the application note EE436 ZIP file [3], that implements these models on the ADSP-21593 EZ-Kit evaluation board.

A similar set of code examples is provided for the ADSP-21569 EZ-Kit evaluation board. The core and accelerator performance numbers are measured for the *core only* case and for the single accelerator usage model. The resultant core MIPS savings for each model are displayed. Table 4 MIPS Summary for Different Accelerator Usage Models on page 21 shows the MIPS of the core and accelerator for different usage models, plus the Core MIPS savings. *The numbers measured on the ADSP-21569 are mentioned in parentheses for comparison*.

From a performance perspective, the *Data Pipelining* model is the best, resulting in saving approximately 333 core MIPS. Next best is the *Split Task* model (saving 252 core MIPS), and followed the *Direct Replacement* model (saving 239 core MIPS). These results align with the previous discussion about the accelerator usage models.

Table 4 MIPS Summary for Different Accelerator Usage Models

Usage Model	MIPS Usage			Core MIPS Saving
	Core	FIRA	IIRA	
Core Only	339 (339)	0.00		-
Direct Replacement	100 (241)	82 (161)	12 (75)	239 (98)
Split Task	87 (165)	69 (110)	12 (50)	252 (174)
Data Pipelining	6 (5)	82 (161)	12 (75)	333 (334)

The numbers were measured on ADSP-SC59x/2159x and Pre-Fetch Buffer and Maximum Burst Transfer size of 16 were enabled for FIRA.

Key Take-aways

The following observations derive from the comparison of Table 4 MIPS usage for the ADSP-SC59x/2159x processors and the ADSP-2156x:

- The core MIPS saving achieved on the ADSP-SC59x/2159x is significantly better than that on ADSP-2156x. This is because multiple instances of FIRA / IIRA(2x/8x) operate in parallel on the ADSP-SC59x/2159x, compared to a single instance of FIRA/IIRA on ADSP-2156x.
- As we move from the Core Only to Direct Replacement model, we observe a significant amount of core MIPS saving on the ADSP-SC59x/2159x. This means for the customer that with a minimal amount of code changes, that is, replacing the *Core* function calls with *Accelerator* function calls, a significant amount of core MIPS can be saved.

References

[1] *ADSP-SC59x SHARC+ Processor Hardware Reference,* Rev 0.1, October 2018. Analog Devices, Inc.

[2] *ADSP-SC59x SSDD API Reference Manual for SHARC+ Core*, Version 2.0, CrossCore® Embedded Studio Help.

[3] *Associated ZIP file for EE-436: Using ADSP-SC59x/2159x High Performance FIR/IIR Accelerators,* March 2022. Analog Devices, Inc.

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

[5] *EE-408: Using ADSP-2156x High Performance FIR/IIR Accelerators,* August 2019. Analog Devices, Inc.

[6] ***Biquad**–any block that realizes a biquadratic transfer function (that is, the numerator and denominator are quadratic polynomials in s [continuous time] or in z to the -1 [discrete time]). The Biquad Filter block independently filters each channel of the input signal with the specified biquadratic infinite impulse response (IIR) filter.*

Document History

Revision	Description
Rev 2– September 23, 2022 *by Sanket Nayak and Mitesh Moonat*	Added changes to include the details and graphs for ADSP-SC598.
Rev 1– June 14, 2022 *by Sanket Nayak and Mitesh Moonat*	Initial Release