Vol 54 No 2, April 2020

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

AnalogDialogue

Leveraging the On-Chip FIR and IIR Hardware Accelerators on a Digital Signal Processor

Mitesh Moonat, Applications Engineer and
Sanket Nayak, Applications Engineer

Abstract

Finite impulse response (FIR) and infinite impulse response (IIR) filters are the most frequently used digital signal processing algorithms—especially for audio processing applications. Thus, a significant portion of a processor core's time is consumed for FIR and IIR filtering in a typical audio system. The on-chip FIR and IIR hardware accelerators, also referred to as FIRA and IIRA, respectively, on a digital signal processor can be used to offload the FIR and IIR processing tasks, thus freeing up the core for other processing. In this article, we will discuss how to make use of these accelerators in practice with the help of different usage models illustrated with tested real-time examples.

Introduction

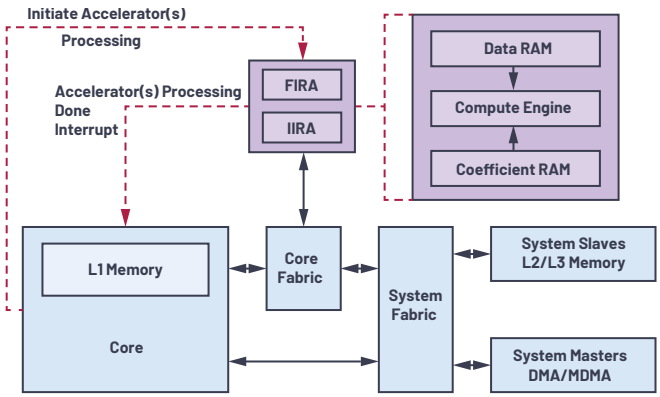

Figure 1. FIRA and IIRA system block diagram.

Figure 1 shows a simplified block diagram of FIRA and IIRA and how they interact with the rest of the processor system and resources.

- Both the FIRA and IIRA blocks mainly consist of a compute engine—multiply and accumulate (MAC) units—along with a small local data and coefficient RAM.
- To start the FIRA/IIRA processing, the core initializes a chain of DMA transfer control blocks (TCBs) in processor memory with channel-specific information. The core then writes the FIRA/IIRA chain pointer register with the start address of this TCB chain and then configures the FIRA/IIRA control register to start the accelerator processing. Once processing of all the channels is complete, an interrupt is sent to the core so that it can use the processed output for further operations.
- Theoretically, the best approach is to offload all the FIR and/or IIR tasks from the core to the accelerator(s) and allow the core to do something else in parallel. But in practice, this may not always be feasible, particularly when the core needs to use the output from the accelerator(s) for further processing and has no other independent tasks to finish in parallel. In such cases, we need to choose the appropriate accelerator usage model to achieve the best results.

In this article, we'll discuss various models to use these accelerators optimally for different application scenarios.

Using FIRA and IIRA in Real Time

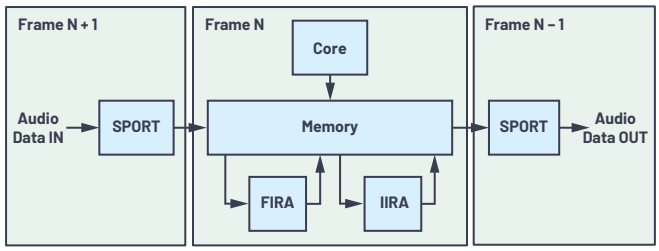

Figure 2. Typical real-time audio data flow.

Figure 2 shows a typical real-time PCM audio data flow diagram. One frame of digitized PCM audio data is received over a synchronous serial port (SPORT) and sent to memory via direct memory access (DMA). While reception of frame N+1 is going on, frame N is processed by the core and/or the accelerator(s), and the output of the previously processed frame (N-1) is sent out via SPORT to the DAC for digital-to-analog conversion.

Accelerator Usage Models

As discussed earlier, depending upon the application, the accelerators may need to be used in different ways to offload the maximum FIR and/or IIR processing tasks and to save the most possible core cycles for other operations. At a high level, the accelerator usage models can be divided into three categories: direct replacement, split task, and data pipelining.

Direct Replacement

- The core FIR and/or IIR processing is directly replaced by the accelerator(s) and the core simply waits for the accelerator(s) to finish the job.
- This model is effective only when the accelerator can process faster than the core; that is, using the FIRA block.

Split Task

- The FIR and/or IIR processing tasks are divided between the core and the accelerator(s).
- This model is especially useful when multiple channels are available to be processed in parallel.
- Based on a rough timing estimation, the total number of channels can be divided between the core and the accelerator(s) in such a way that both finish at approximately the same time.
- As shown in Figure 3, this usage model results in more core-cycle savings than the direct replacement model.

Data Pipelining

- The data flow between the core and accelerator(s) can be pipelined in such a way that both can work in parallel on different data frames.
- As shown in Figure 3, the core processes the N^{th} frame and then initiates the accelerator's processing of this frame. The core then continues in parallel to further process the $N\text{-}1^{th}$ frame output produced by the accelerator(s) in the previous iteration. This sequence allows the complete offloading of the FIR and/or IIR processing task to the accelerator(s) at the cost of additional output latency.
- The pipeline stages and, consequently, the output latency can increase depending upon the number of such FIR and/or IIR processing stages in the complete processing chain.

Figure 3 illustrates how the audio data frames flow between three stages—DMA IN, core/accelerator processing, and DMA OUT—for various accelerator usage models. It also shows how free core cycles increase compared to the core only model by fully or partially offloading the FIR/IIR processing to the accelerator across different accelerator usage models.

FIRA and IIRA on SHARC Processors

The following Analog Devices SHARC® processor families support on-chip FIRA and IIRA (oldest to the newest).

- ADSP-214xx (for example, ADSP-21489)
- ADSP-SC58x
- ADSP-SC57x/ADSP-2157x
- ADSP-2156x

Across processor families:

- Compute speed varies.
- The basic programming model remains the same except for the auto configuration mode (ACM) on ADSP-2156x processors.
- FIRA has four MAC units while IIRA has a single MAC unit.

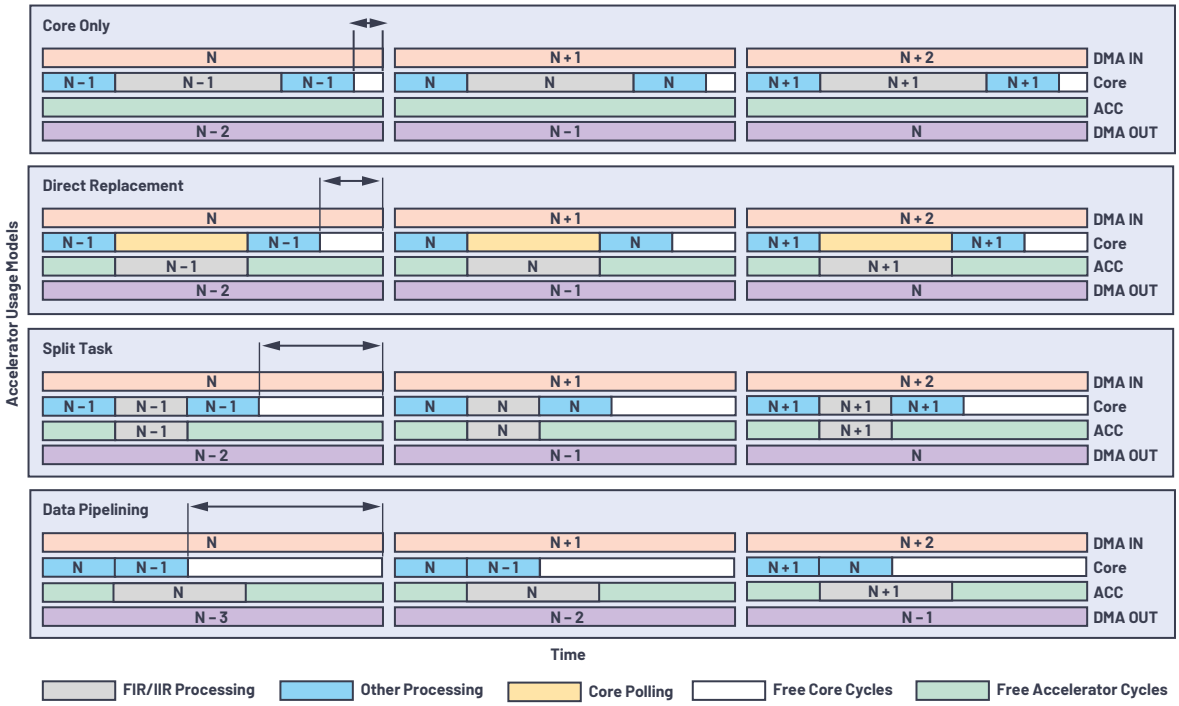

Figure 3. Accelerator usage models comparison.

FIRA/IIRA Improvements on ADSP-2156x Processors

ADSP-2156x is the latest addition to the SHARC processor family. It is the first 1 GHz single-core SHARC processor with FIRA and IIRA also capable of running at 1 GHz. FIRA and IIRA on ADSP-2156x processors offer various improvements over its predecessors, the ADSP-SC58x/ADSP-SC57x processors.

Performance Improvements

- Compute speed increased by eight times (SCLK-125 MHz to CCLK-1 GHz).
- Lesser data and MMR access latency between the core and the accelerators is possible due to closer integration of the core and the accelerators with the help of a dedicated core fabric.

Functional Improvements

Support for ACM was added to minimize core intervention required to handle accelerator processing. This mode comes with the following new, main features:

- Allows halting of the accelerator for dynamic task queuing.
- No channel number limitation.
- Trigger generation (master) and trigger wait (slave) support.
- Selective interrupt generation for each channel.

Experimental Results

In this section, we'll discuss the results of two real-time multichannel FIR/IIR use cases implemented with the help of different accelerator usage models on an ADSP-2156x evaluation board.

Use Case 1

Figure 4 shows the block diagram of use case 1. The sample rate is 48 kHz, the block size is 256 samples, and the ratio of core to accelerator channels used in the split task model is 5:7.

Table 1 shows the measured core and FIRA MIPS numbers along with the resultant core MIPS savings compared to the core only model. The table also shows additional output latency added by the corresponding usage model. As we can see, with accelerator usage, up to 335 core MIPS could be saved with a data pipelining usage model, at the cost of 1 block (5.33 ms) of output latency. Direct replacement and split task usage models also result in 98 MIPS and 189 MIPS savings, respectively, without any additional output latency.

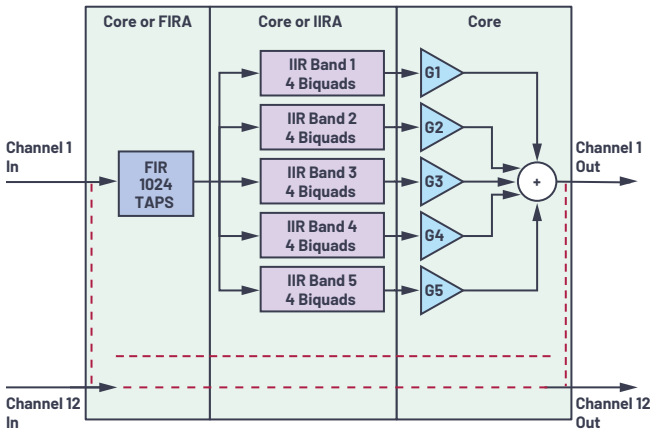

Figure 4. Use case 1 block diagram.

Table 1. Core and FIR/IIRA MIPS Summary for Use Case 1

Usage Model	Core MIPS	FIRA MIPS	IIRA MIPS	Core MIPS Saving	Usage Model Latency (ms)
Core Only	337				0
Direct Replacement	239	162	75	98	0
Split Task	148	96	44	189	0
Data Pipelining	2	161	75	335	5.33 (1 frame)

Use Case 2

Figure 5 shows the block diagram of use case 2. The sample rate is 48 kHz, the block size is 128 samples, and the ratio of core to accelerator channels used in the split task model is 1:1.

Like Table 1, Table 2 shows the results for this use case. As we can see, with accelerator usage, up to 490 core MIPS could be saved with a data pipelining usage model at the cost of 1 block (2.67 ms) of output latency. A split task usage model results in 234 core MIPS savings without any additional output latency. Note that unlike in use case 1, frequency-domain (fast convolution) processing is used for the core instead of time-domain processing. This is the reason why the core MIPS taken to process one channel is less than the FIRA MIPS taken, which results in a negative core MIPS savings for the direct replacement usage model.

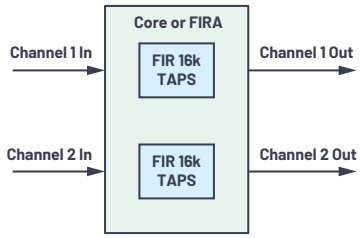

Figure 5. Use case 2 block diagram.

Table 2. Core and FIR/IIRA MIPS Summary for Use Case 2

Usage Model	Core MIPS	FIRA MIPS	Core MIPS Saving	Usage Model Latency (ms)
Core Only	493			0
Direct Replacement	515	511	–22	0
Split Task	259	257	234	0
Data Pipelining	3	511	490	2.67 (1 frame)

Conclusion

In this article, we have seen how significant core MIPS can be offloaded to FIRA and IIRA accelerators on ADSP-2156x processors, utilizing different accelerator usage models to achieve desired MIPS and processing profiles.

Further Reading

"Graphical Demonstration of ADSP-2156x FIR/IIR Accelerator Performance and Real Time Usage." Analog Devices, Inc.

Nayak, Sanket and Mitesh Moonat. "Engineer-to-Engineer Note EE-408: Using ADSP-2156x High Performance FIR/IIR Accelerators." Analog Devices, Inc., August 2019.

About the Author

Mitesh Moonat is currently working as an applications engineer in processor applications team at Bangalore (ADBL), India. He works on pre-/postsilicon validation, peripheral driver development, and support for SHARC® processors. He also worked on Blackfin® and ADSP-21xx processors in his career at ADI. His interests include processor architecture, digital signal processing algorithm optimization, module, and system-level debug of embedded systems. Mitesh joined Analog Devices in 2006. He graduated with a bachelor's degree in electronics and communication engineering from National Institute of Technology, Warangal, India. He can be reached at *mitesh.moonat@analog.com*.

About the Author

Sanket Nayak is a product applications engineer in the Processor Applications Team in Bangalore (ADBL), India. He joined ADI in 2016 and has worked on pre-/postsilicon validation of automotive DSPs, drivers/FuSa ROM design, development, and testing. He earned his bachelor's degree in electronics and communication engineering from P.E.S Institute of Technology, Bangalore. He can be reached at *sanket.nayak@analog.com*.