

SHARC Bites Back

From the Editor's Files:

This is the 2nd in a series of articles that explores the complex processor tradeoffs and evaluations required to choose the most effective processor for continuous real-time signal processing applications as typified by the 1024-point complex Fast Fourier Transform (cFFT). Part I appeared in the December 2002 issue of *COTS Journal*, and is available on www.cotsjournalonline.com.

Part II: Continuous Real-Time Signal Processing—Comparing TigerSHARC and PowerPC Via Continuous cFFTs

Continuous, measured benchmarks provide the only real insight into real-world DSP performance. An analysis using continuous cFFTs sheds light on the real performance differences between TigerSHARC and PowerPC.

Jeffrey Milrod, President, BittWare
Chuck Millet, TigerSHARC Product
Manager, Analog Devices

Although often quoted, peak performance benchmarks are misleading when used to depict DSP performance. Any real-world view into DSP performance should center on sustained or continuous, algorithm performance. Part I of this article series illustrated that fact. It also proposed the continuous 1024-point cFFT algorithm as an excellent indicator of real-world performance. The Analog Devices TigerSHARC (ADSP-TS101) and the Motorola G4 PowerPC with AltiVec (MPC7410 and MPC7455) processors were compared along those lines, with predictions made for their performance.

Here, that analysis is taken a step further. BittWare engineers took continuous 1024-point cFFT algorithm benchmarks and coded, optimized and measured them on the ADSP-TS101 at 250 MHz and on the MPC7410 at 400 MHz. Unlike some other “system-level” benchmarks, these implementations of the continuous cFFT benchmarks took advantage of all the features of each processor that could improve performance—since this approach is more like what would be implemented by an engineer designing a real system. The results are reported in detail, along with extrapolations for other variants and speeds, as well as board-level implications.

TigerSHARC Implementation

The continuous cFFT algorithm was implemented on a BittWare Tiger-PCI board (TSPC) that features a cluster of four ADSP-TS101 TigerSHARCs running at 250 MHz. The benchmark code was written in C and was developed using Analog Devices' VisualDSP++.

Part I of the article series predicted that running continuous cFFTs on the TigerSHARC would be processor-limited, implying that the continuous benchmark performance would be driven by the performance of the cFFT algorithm itself. That drove the engineers to choose a highly optimized cFFT routine for the benchmark. With that in mind, the

Special Feature

SHARC Bites Back

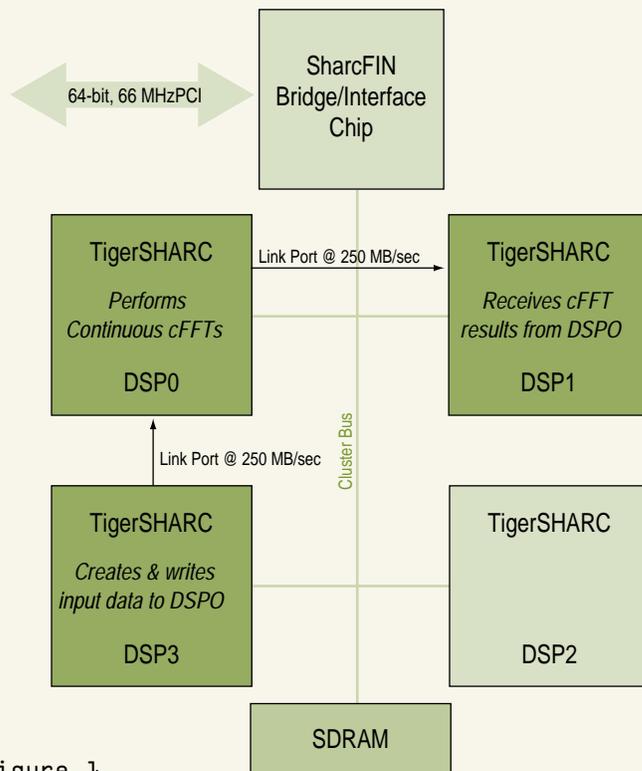


Figure 1

The link ports on the TigerSHARC provide a useful way to move data on and off the processor. A single TigerSHARC (DSP0) was used to continuously perform cFFTs. Two additional TigerSHARCs were used as test instrumentation to generate the input data (DSP3) and receive the results (DSP1) via link ports.

Processor Specifications and Performance Predictions from Part I

Parameter	TigerSHARC ADSP-TS101S	PowerPC MPC7410	PowerPC MPC7455
Core Clock	250 MHz	500 MHz	1,000 MHz
Peak Floating-pt Performance	1,500 MFLOPS	4,000 MFLOPS	8,000 MFLOPS
Memory Bus Size/Speed	64-bit/100 MHz	64-bit/125 MHz	64-bit/133 MHz
External Link Ports	4@250 MB/Sec	None	None
I/O Bandwidth (inc. memory)	1,800 MB/Sec	1,000 MB/Sec	1,064 MB/sec
Bandwidth-to-Processing Ratio	1.20 Bytes/FLOP	0.25 Bytes/FLOP	0.13 Bytes/FLOP
1024-pt cFFT Benchmark	39 μ sec	22 μ sec	13 μ sec (est.)
Approx Cycles for 1024-pt cFFT	9,750 cycles	11,000 cycles	13,000 cycles
Predicted 1024-pt cFFTs/chip	25,641 per Sec	26,053* per Sec	64,941* per Sec

* Assumes 100% of peak I/O used for continuous cFFTs, and neglects cache & data movement overheads due to inability to predict – real-world performance could be much less.

Table 1

Compared here are TigerSHARC and PowerPC specs and the predictions made for the performance of the continuous 1024-point cFFT implementation on the TigerSHARC and the PowerPC.

benchmark implementation calls the 1024-point cFFT function from the EZ-DSP TSLibs function library, which is handcrafted and fully optimized in assembly language.

The link ports on the TigerSHARC were the obvious way to move data on and off the processor. As Figure 1 shows, a single TigerSHARC was used to continuously perform cFFTs. Two additional TigerSHARCs were used as test instrumentation to generate the input data and receive the results via link ports. Those link ports could also be used to move data from/to other I/O devices such as FPGAs—using the other TigerSHARCs on the TSPC board was simply a convenience.

Since the TigerSHARC can support I/O DMAs to and from internal memory in background, dual input and output buffers were used to implement a ping-pong scheme (Figure 2). Using this ping-pong scheme, the internal memory of the TigerSHARC must hold the benchmark code, as well as dual input and output buffers. It was verified that how the buffers are placed in memory could dramatically impact the benchmark performance. As shown in Figure 3, the TigerSHARC's internal memory consists of three banks of 2 Mbits each.

PowerPC Implementation

Turning to the PowerPC benchmark analysis, the continuous cFFT algorithm was implemented on a Motorola PrPMC800 PMC that has a single MPC7410 running at 400 MHz with a 100 MHz front-side bus; the PMC board was placed on a PCI carrier card for ease of use. The benchmark code was written in C and assembly, and was compiled using the GNU C compiler version 2.96; it was loaded on to the board using the PPCBUG monitor via a serial port.

Part I of this article series predicted that running continuous cFFTs on the G4 PowerPC with AltiVec would be bandwidth-limited. That implies that continuous benchmark performance is limited by I/O, not by the performance of the cFFT algorithm itself. That means that the optimization of the cFFT routine used for the benchmark is not critical since it does not drive the performance. A public domain 1024-point cFFT algorithm implementation was used that offered good, but not stellar performance.

As with most PowerPC boards, the primary mechanism for data movement on the PrPMC800 is through the external node controller and PCI bridge. The PrPMC800 uses Motorola's Harrier chip that features a 64-bit, 66 MHz PCI interface, as shown in Figure 4. Since the cFFT algorithm operates "in-place"—the input data buffer is overwritten with the results—only two buffers are required to implement a ping-pong scheme. These buffers were placed in the main memory.

While a cFFT was performed on buffer A, data output from the previous cFFT calculation is written out via a DMA from buffer B to the PCI bus; data for the next cFFT calculation is then read in via a DMA from the PCI bus back into buffer B. As with the TigerSHARC implementation, the buffers simply switch when the whole process is complete. The two DMAs were combined using a chained DMA, which allows a sequence of DMA operations to be combined into a single operation using a linked list of DMA descriptors.

Most programmers use an operating system when programming the PowerPC due to the complexity and difficulty of creating stand-alone programs. Since the overhead of an operating system could reduce the benchmark performance, the engineers

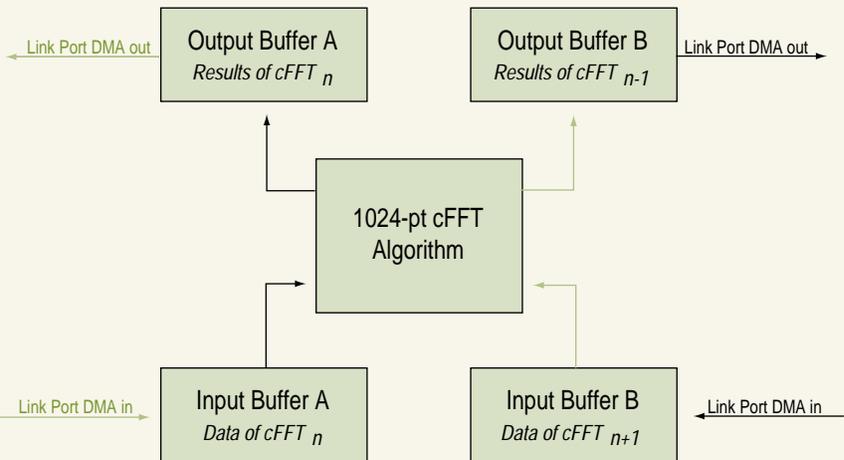


Figure 2

For the TigerSHARC benchmark analysis dual input and output buffers were used to implement a ping-pong scheme. While the cFFT routine is processing data from input buffer A and writing the results to output buffer A, the DMA engines are doing two things: moving the data for the next cFFT into input buffer B from a link port while the results from the previous cFFT are being written out the other link port from output buffer B. After the cFFTs and both DMAs complete, the ping-pong buffers are swapped; the cFFT now processes out of/into buffer B while the DMAs operate on buffer A.

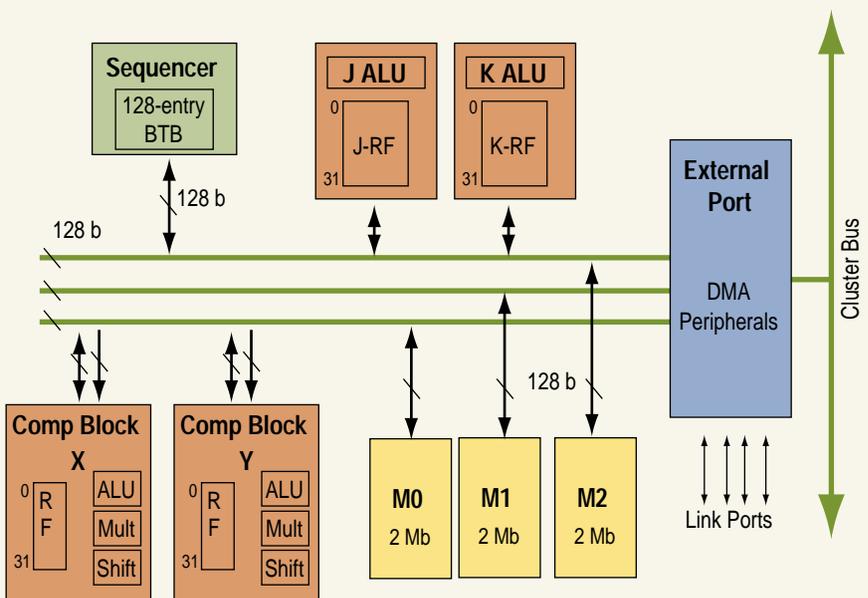


Figure 3

With an internal memory consisting of three banks of 2 Mbits each, there was ample room on the TigerSHARC to implement the buffers used in the benchmark analysis. Bank 1 is used for the program code—with plenty of room to spare. Even though both data buffers would easily fit into a single bank, it was found that for optimal performance each input/output buffer set needed to be placed in a separate bank—buffers A placed in Bank 2, and buffers B placed in Bank 3.

Special Feature SHARC Bites Back

configured the PowerPC manually, including managing the cache, MMU and DMA engine on the external node controller.

Bandwidth Limited

Because the PowerPC is bandwidth-limited and uses cache to speed access to data, cache management was considered critical in the algorithm

implementation. The cache is not addressable memory, but merely a fast buffer holding a copy of portions of the memory and is controlled by the cache manager. Once a memory address range—in this case, an I/O buffer—is accessed by the processor, it was “cached” for quicker subsequent accesses. As a consequence, input data that is read in from the PCI bus into

memory is not necessarily the same data that the processor sees when it attempts to access the data. That’s because the PowerPC will look to the copy of data that is stored in cache rather than the copy that is in memory.

One way to eliminate this problem is to enable “snooping”, which causes the hardware to keep the contents of cache and main memory coherent. But because snooping can severely hinder performance, the cache was managed manually by “invalidating” the address range containing the data received via DMA. An “invalidate” operation forces the cache to reload the data in the invalidated address range from memory the next time it is accessed, causing the data in cache to be consistent with the data in memory.

Likewise, when the cFFT writes its results into an output buffer, this buffer is located in cache rather than in memory. It must then be “flushed” from cache to memory before a DMA is performed; otherwise the DMA engine will move stale data located in memory rather than the results of the calculations that are stored in cache.

Results Compared

A comparison of the processors’ specs and the predictions made for the performance of the continuous 1024-point cFFT implementation on the TigerSHARC and the PowerPC are shown in Table 1. Table 2 shows the actual results of the benchmarks tests. While neither processor did as well as predicted, the TigerSHARC dramatically outperformed the PowerPC.

Further examination of the TigerSHARC results and implementations revealed that the prediction made in Part I neglected to allow for any DMA management overhead. Setting up the link port DMAs, handling the DMA-done interrupts and checking for DMA completion adds an overhead of approximately 10% that

Results of Continuous 1024-pt cFFT Benchmark Implementation

Parameter	TigerSHARC ADSP-TS101S	PowerPC MPC7410
Core Clock	250 MHz	400 MHz
Actual 1024-pt cFFTs/chip	22,923 per sec	7,899 per sec

Table 2

Shown here are actual results of the benchmarks tests. While neither processor did as well as predicted, the TigerSHARC dramatically outperformed the PowerPC.

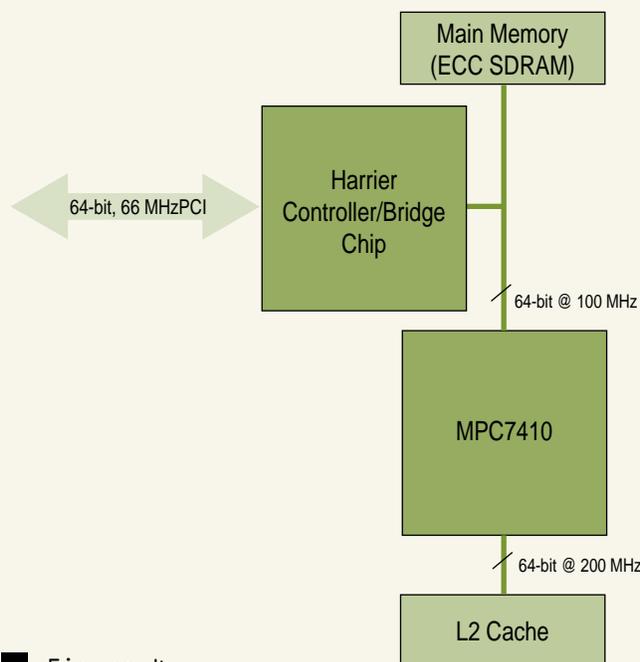


Figure 4

The PrPMC800, a Motorola Processor PMC board, provided the platform for the PowerPC benchmark analysis. The card’s primary mechanism for data movement is via the external node controller and PCI bridge. Since the cFFT algorithm operates “in-place”, only two buffers are required to implement a ping-pong scheme. These buffers were placed in the main memory.

accounts for the performance difference. For the sake of convenience the DMA management code was written in straightforward C, and since the improvement of the overall benchmark would be small (10% at best), no attempt was made to minimize this overhead. The results reveal that the PowerPC's overhead is much more significant. Although the benchmark tests were only performed on a 400 MHz processor, the results are approximately a factor of three less than predicted.

To ensure accurate and optimized results, several variants of the full benchmark implementation were run on the PowerPC. As expected, with the cache disabled, performance decreased by about an order of magnitude. Benchmarks were also run with the cFFT disabled, and it was discovered that just moving the data in and out, as well as managing the cache and MMU, resulted in no performance improvement. The upshot is that even if a considerably faster cFFT algorithm were used, there would be no performance improvement of the continuous cFFT benchmark. Along similar lines, eliminating the chained DMAs and forcing all data movement over PCI to be writes, could possibly result in a small, but relatively insignificant improvement.

Operating System Impact

One complication is the possible impact of an operating system to manage the cache and MMU. It is assumed that the overhead would be increased; however, this may be a false assumption—it's possible that commercial operating systems could provide an improvement over this manual implementation, but it seems unlikely due to the inefficiencies and cache overhead associated with the OS services and context switches.

Arguments could be made that this benchmark implementation on the PowerPC is not truly indicative of

Board-level Implications and Extrapolations

Parameter	TigerSHARC	TigerSHARC	PowerPC
	ADSP-TS101S	ADSP-TS201S	MPC7410
Core Clock	250 MHz	500 MHz	500 MHz
Typical # Processors/Board	8	8	4
Peak FLOPS/Board	12 GFLOPS	24 GFLOPS	16 GFLOPS
Memory Bus Size/Speed	64-bit/83.3 MHz	64-bit/100 MHz	64-bit/100 MHz
Typical Off-board I/O	2 PMC + 16 Links	2 PMC + 16 Links	2 PMCs
Peak Off-board I/O (not backplane)	5,056 MB/Sec	9,056 MB/Sec	1,056 MB/Sec
Bandwidth-to-Processing Ratio	0.42 Bytes/FLOP	0.38 Bytes/FLOP	0.07 Bytes/FLOP
Projected 1024-pt cFFT/Board	180,000* per Sec	360,000* per Sec	24,000* per Sec

* Estimated based on previous results

Table 3

Board level performance is projected to simply be the number of continuous cFFTs per processor multiplied by the number of processors per board. It was originally projected that an Octal TigerSHARC board with an 250 MHz ADSP-TS101S could perform three times as many continuous cFFTs as a quad 500 MHz PowerPC MPC7410 board. The results of this study indicate that the TigerSHARC board will actually outperform the PowerPC board by more than seven times, and that the new TigerSHARC (ADSP-TS201S) will outperform the PowerPC boards by a factor of 15.

the real-time signal processing capabilities of the PowerPC. But the conclusions of the benchmark analysis suggest that the TigerSHARC is far superior at processing continuous cFFTs and is, therefore, a better real-time signal processor.

As discussed in Part I of this article series, multi-processor COTS boards are readily available with TigerSHARCs and PowerPCs. The original board-level predictions accounted for the I/O bandwidth scalability of the TigerSHARC, and further bandwidth limits imposed for the PowerPC boards. However, the reduced processor benchmark performance of the PowerPC indicates that the boards should no longer further limit the I/O required to keep all the PowerPCs fed when running continuous cFFTs. As shown in Table 3, the board level performance is projected to simply be the number of continuous cFFTs per processor multiplied by the number of processors per board.

Original Conclusions Supported

The benchmark implementations and testing supported the conclusions from Part I that the TigerSHARC is a superior real-time signal processor. In fact, the results make them even more emphatic. In summary, the PowerPC is more suited for applications that require a lot of number crunching with little data movement, typical of so-called back-end data processing. For continuous real-time signal processing such as imaging, radar, sonar, SIGINT and other applications that require high data flow or throughput, however, the TigerSHARC can dramatically outperform the PowerPC. ■■

BittWare
Concord, NH.
(603)226-0404.
[www.bittware.com].

