Applying a High Performance Tiled Rad-Hard Digital Signal Processor to Spaceborne Applications

Joseph Marshall
BAE Systems
9300 Wellington Road
Manassas, VA 20110
703-367-1326
Joe.Marshall@BAESystems.com

Richard Berger
BAE Systems
9300 Wellington Road
Manassas, VA 20110
703-367-1809
Richard.W.Berger@BAESystems.com

Michael Bear
BAE Systems
9300 Wellington Road
Manassas, VA 20110
703-367-4199
Micahel.Bear@BAESystems.com

Lisa Hollinden
BAE Systems
9300 Wellington Road
Manassas, VA 20110
703-367-2447
Lisa.Hollinden@BAESystems.com

Jeffrey Robertson
BAE Systems
9300 Wellington Road
Manassas, VA 20110
703-367-3476
Jeff.Robertson@BAESystems.com

Dale Rickard
BAE Systems
9300 Wellington Road
Manassas, VA 20110
703-367-5619
Dale.Rickard@BAESystems.com

Abstract— **This paper discusses the architecture and reviews the development of the RA*DSP*EED™ DSP. It illustrates planned board solutions and briefly highlights the other critical components needed such as regulators, bridges to the rest of the spacecraft and high performance memory. The paper describes the various algorithm elements that may apply to the application classes and compiles information on RADSPEED algorithm elements. Lessons learned from development and translation of algorithms from single string to multi-processing elements using the supporting tools are given. For the many spaceborne processing applications that fit onto this architecture, the RADSPEED DSP provides a very high performance / power solution that will scale with the needs of the application.**

TABLE OF CONTENTS

1. INTRODUCTION 1
2. RADSPEED DSP ARCHITECTURE.....2
4. RADSPEED DSP BOARDS 3
5. CRITICAL COMPONENTS – BRIDGE.....4
6. CRITICAL COMPONENTS – HIGH SPEED MEMORY.....5
7. ALGORITHM ELEMENTS AND APPLICATION CLASSES.....5
8. CAF ALGORITHM DISCUSSION 6
9. TOOLSET AND APPLICATION CONVERSION FROM PC TO PPC.....6
10. SINGLE TO MULTI-CORE CONVERSION OF ALGORITHMS.....7
11. SUMMARY 9
REFERENCES.....9
BIOGRAPHIES.....10

1. INTRODUCTION

The interest in expanded onboard programmable signal processing in spacecraft is growing. Driven by large increases in available data to process from improved sensors, large solid state storage systems, high speed intra-spacecraft interfaces, the continuing shrinkage of silicon required to perform an operation and the growth of cloud computing terrestrially, the desires and requirements for more onboard processing are showing up in new and upgraded spacecraft concepts.

On-board processing may be performed in a variety of ways including 1) specialized radiation hardened application specific integrated circuits (ASIC) with their highest performance/watt but limited functional flexibility, 2) reprogrammable field programmable gate arrays (FPGA) with higher power requirements and additional challenges in radiation mitigation and configuration storage; 3) multi-core general purpose processors (GPP) with floating point units supported by COTS infrastructure but with lower overall performance and 4) multi-core digital signal processors (DSP) optimized for signal processing applications that provide programmability and high performance at a reasonable power.

The RA***DSP***EED™ DSP - a radiation hardened version of the ClearSpeed™ CSX700 digital signal processor – is being developed by BAE Systems to provide a high performance per watt digital signal processor for these emerging spacecraft applications. The CSX700 is a best in class device in the commercial world and is used in a variety of graphics, financial, server and other signal processing applications supported by a full set of development tools. In hardening the RADSPEED DSP, over 70% of the performance of the commercial device was retained (70 GFLOPS peak performance) with a modest decrease in processing speed (8%) and the number of processing element cores (160 from 200). Interfaces were adjusted to scalable direct connections between RADSPEED DSPs enabling the support of dual and quad devices. All commercial development tools supporting the CSX700 may be used in either the native or RADSPEED DSP configurations. A virtual private network (VPN) capability providing RADSPEED DSP operation using CSX700 hardware access is available for remote benchmarking by interested parties.

978-1-4577-0557-1/12/$26.00 ©2012 IEEE

The RADSPEED DSP opens up a programmable world for specialized payload applications. This includes 1) various types of RF processing, 2) radar processing with the high throughput required for both Space-Time Adaptive Processing (STAP) and Synthetic Aperture Radar (SAR) algorithms, 3) hyperspectral imaging with the need for simultaneous processing of images across a number of frequency bands, fusing data for analysis, spectral analysis for the simultaneous assessment of data across a number of frequency bands and 4) image processing including edge and object detections and the efficient distribution of a high resolution image across a series of parallel processing elements.

2. RADSPEED DSP ARCHITECTURE

A block diagram of the RADSPEED DSP is shown in Figure 1. The RADSPEED DSP is made up of 160 processing elements (PE) (arrayed in two groups of 76 processing elements) with eight spares. Each group of PEs is organized as a multi-threaded array processor (MTAP), controlled by a "mono processor" core. Each PE incorporates double precision floating point hardware as well as integer processing. Each MTAP operates as a single instruction multiple data (SIMD) architecture with a dedicated 30 Gbps DMA-controlled DDR2 memory interface to avoid bottlenecks when supplying or unloading data. Two ClearConnect™ Bridges (CCBR) provide the external data interface to the MTAPs using a wide parallel DDR-like interface capable of moving 30 Gbps each. Each CCBR may be connected between RADSPEED DSPs or between a RADSPEED DSP and an external bridging device. The RADSPEED DSP bridge device is described in section 5. The dual CCBR structure is an embedded processor scalability improvement over the ClearSpeed CSX700 which has one CCBR and one PCI Express interface for external data transfers. A JTAG diagnostic interface and a Host Debug Port (HDP) interface rounds out the functions of the RADSPEED DSP.

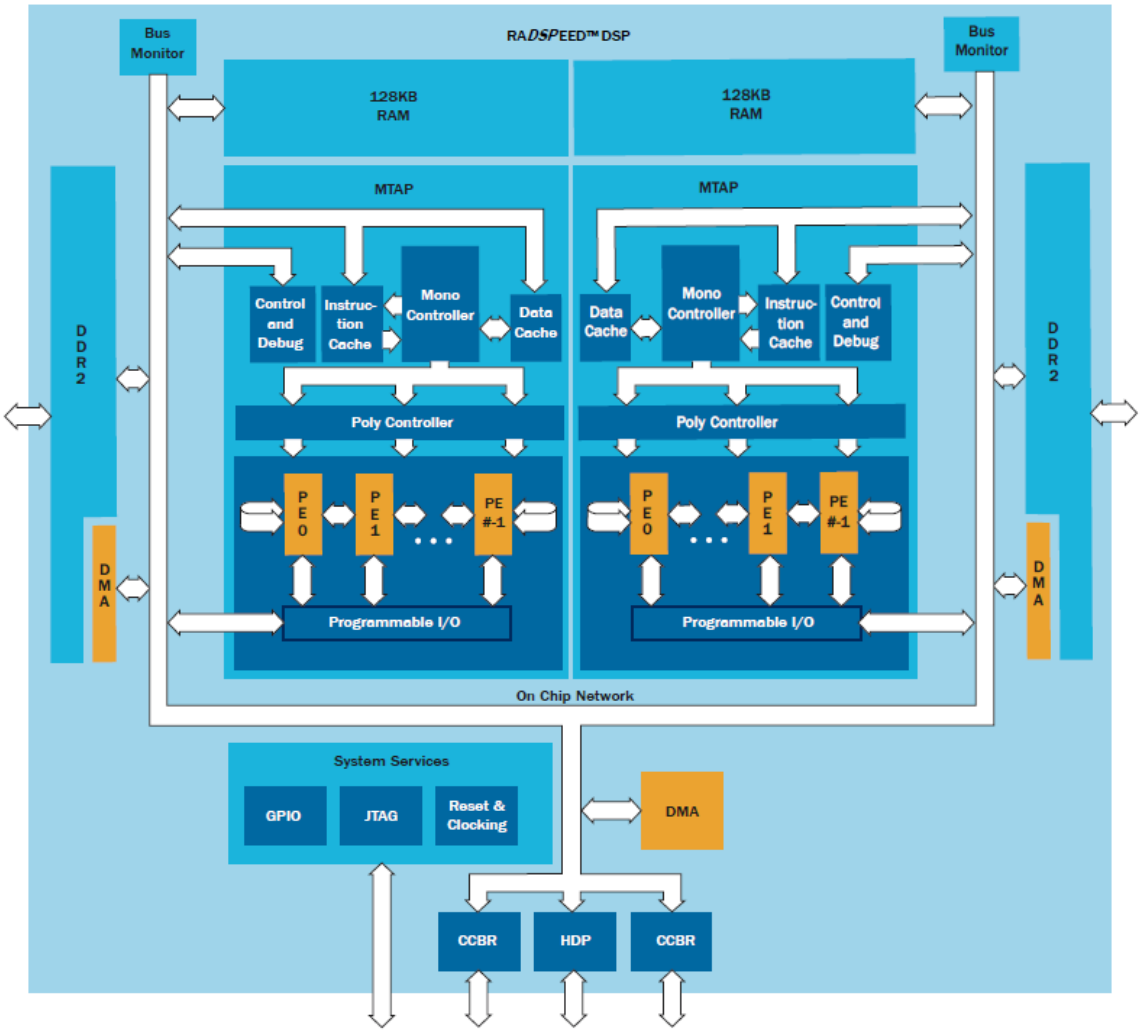


Figure 1 - RADSPEED DSP Block Diagram

3. RADSPEED DSP DEVELOPMENT

The RADSPEED DSP is a variant of the ClearSpeed™ CSX700 [5] DSP. BAE Systems and ClearSpeed Technology modified the device for the space environment and for multi-device intercommunication. While the commercial ClearSpeed CSX700 includes a PCI Express interface and a single proprietary ClearConnect bridge (CCBR) interface, the RADSPEED DSP variant replaces the PCI Express port with a second CCBR. The inclusion of two CCBR buses allows "daisy-chaining" of the DSPs for increased processing performance. BAE Systems performed radiation hardening of the device through circuit and physical design modifications combined with the process hardening features available in BAE Systems' RH90 90nm CMOS technology. The modifications required to achieve radiation hardening resulted in the decision to decrease the number of processing elements from 192 to 152 active processing cores, and the speed changed from 250 MHz to 233 MHz. These changes reduced the peak processing performance from 96 GFLOPS to 70 GFLOPs while keeping the power dissipation to 15W. The high performance bandwidth of the individual buses is maintained at the ~4GB/s. Figure 2 shows the completed device layout.

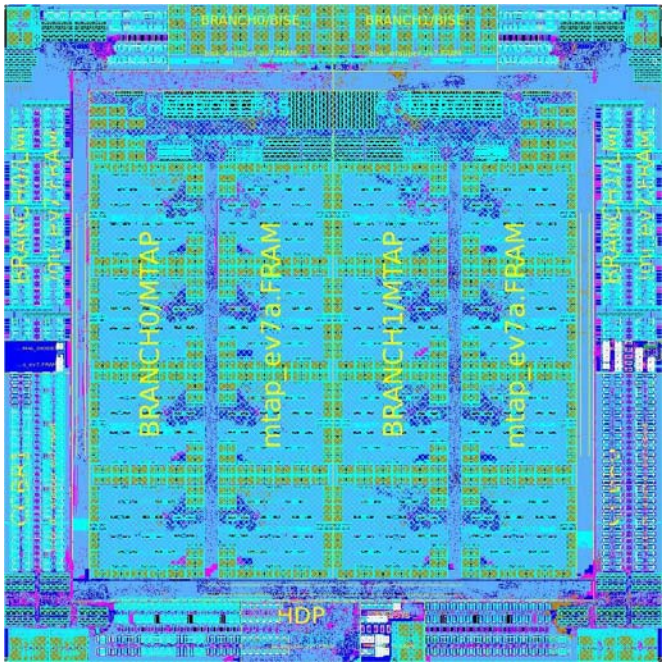


Figure 2 - RADSPEED DSP Physical Design Layout

4. RADSPEED DSP BOARDS

A block diagram of a dual RADSPEED DSP implementation is shown in Figure 3. Two RADSPEED DSPs are connected together via their CCBRs allowing movement of data directly between the processing arrays. Each RADSPEED DSP has two banks of DDR2 memory. Providing access to the external application and the remainder of the instrument is the function of the RADSPEED host/bridge (RADSPEED-HB™) ASIC. Each RADSPEED DSP is connected through one CCBR to the host/bridge. The host/bridge then provides connectivity to the rest of the system via multiple serial Rapid IO interfaces as well as directs data transfers into and out of the RADSPEED DSPs. Additional DDR2 memory for the RADSPEED-HB is attached. The DDR2 memory and the RADSPEED-HB ASIC enable the RADSPEED DSP to be inserted into an instrument or processing system and are described in more detail below. The host debug port provides control access from the RADSPEED-HB to to the RADSPEED DSPs.

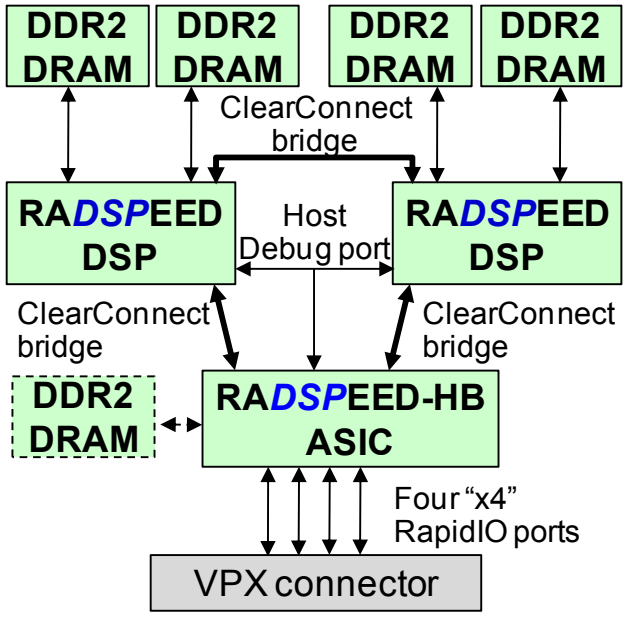


Figure 3 - Dual RADSPEED DSP Board Block Diagram

An early mock-up of a dual RADSPEED DSP board is shown in Figure 4. Passives and termination devices are not shown; many would be on the back side of this board. We have baselined a 6U-160 VPX module as the likely first form factor to house this device.

Power entering the board would typically be at a voltage of 3.3V or higher. Since the RADSPEED DSP and RADSPEED-HB have core voltages of 1V and I/O voltages of 1.8V and 2.5V, switching regulators will be used to drop the voltage to those level. Six regulator daughter boards are shown in the mockup possibly supplemented by linear regulators for very low current requirements in other voltages.

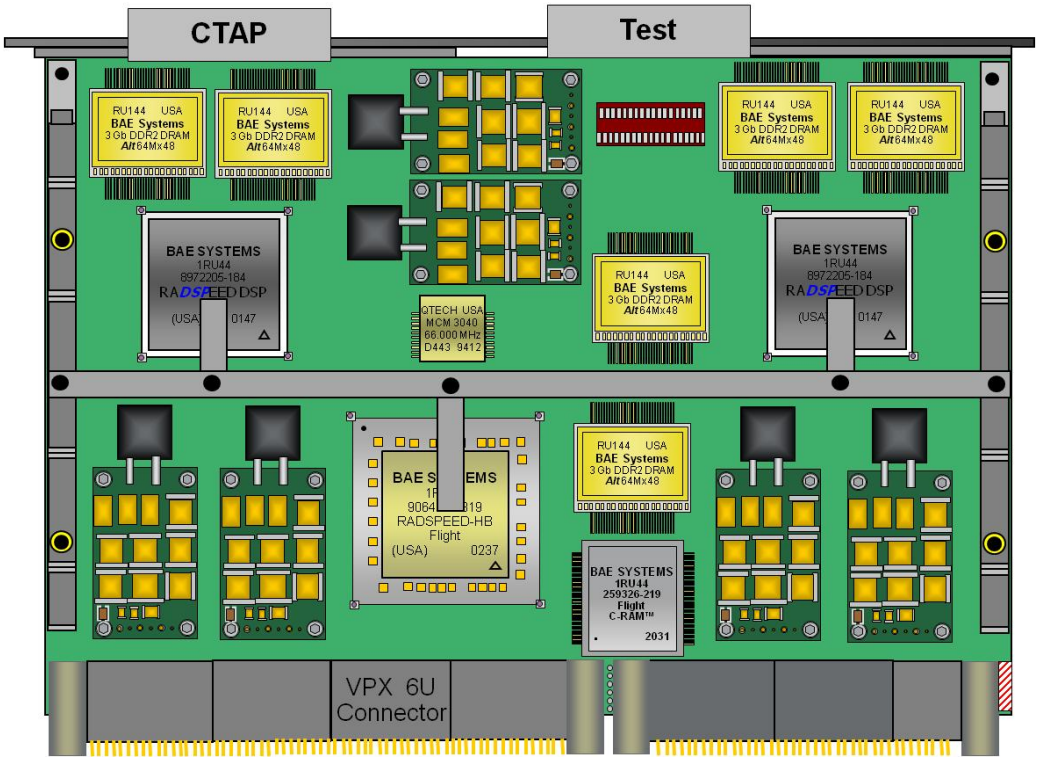


Figure 4 - Dual RADSPEED DSP Board Mock-up

5. CRITICAL COMPONENTS – BRIDGE

The RADSPEED DSP is designed as an accelerator to a host processor. In its ClearSpeed CSX700 commercial implementation, the host processor is typically a personal computer with an Intel-architecture processor and the interface to the host processor is a PCI Express Gen1 port that is programmable up to 16 lanes in width. For space applications, a different implementation is being developed, consistent both with embedded processing and the existing infrastructure within the space community. As such, the host processor is being transitioned to Power Architecture® with CPU cores embedded within the bridge ASIC designed to attach the RADSPEED DSP to a RapidIO backplane.

The RADSPEED-HB host/bridge ASIC connects to the DSP via the CCBR described in section 3. The CCBR physical interface is full-duplex Double Data Rate (DDR) for data transfers and also includes a host debug port for control. Because the RADSPEED-HB will be capable of supporting up to four RADSPEED DSPs, it includes two independent CCBR interfaces. Each of these interfaces can be used for transfer to and from the RapidIO backplane as well as for direct transfers between the two interfaces for communication between RADSPEED DSPs located on the same card.

To support the high performance applications planned, the RapidIO backplane interface architecture consists of up to four ports each with four 3.125-5 Gb/s serializer/deserializer (SerDes) lanes per the RapidIO version 2.1 specification [2]. This provides for simultaneous backplane throughput of up to 64 Gb/s of data transfer after accounting for the 8B/10B overhead associated with the physical layer. In addition to the active RapidIO ports, there is an additional SerDes interface available as a spare, with multiplexing provided internally attach any of the RapidIO link layers to any of the sets of SerDes lanes for fault tolerance purposes. Beyond that, each of the RapidIO ports is designed to support graceful degradation by supporting decrease of interface width from four lanes to either two lanes or one lane in the event of an unrecoverable failure.

A control plane interface is provided via the SpaceWire protocol, eliminating any need to intersperse data and control operations on the RapidIO interfaces. The SpaceWire interface is compatible with various SpaceWire network topologies. Multiple SpaceWire endpoint interfaces are provided along with an embedded router so that no central router is required in the system.

The 64-bit Power Architecture processor cores each include a large dedicated L2 cache and the bridge ASIC architecture provides for full cache coherency within the core and between the cores. The processor cores serve several purposes in the RADSPEED DSP system architecture. First, they are an integral part of setting up RADSPEED operations. They are also used to support transfers to and from the backplane. When a DSP sends data out to the backplane, data is stored within the bridge ASIC in a temporary SRAM buffer. A processor core commands the RapidIO interface to pull the data from that temporary buffer. It also sends messages to the host processor of the receiving DSP that data is ready for transfer as well as when the transfer has been completed. Finally, the processor acts

to perform data processing in some cases. When executing algorithms where the parallelism naturally decreases to a low level as the algorithm near completion, it is more efficient to transfer the final calculations from the DSP into the host processor, thereby freeing up the DSP itself to begin the next calculation where the parallelism is much higher. This increases not only total DSP throughput, it more effectively utilizes both the DSP and CPU processing resources. For algorithms with this characteristic, the point of transfer is programmable by the user. Once defined, no further interaction is required.

The RADSPEED-HB also includes multiple memory interfaces. It provides a DDR2 / DDR3 memory interface that provides main storage for the processor cores as well as shared memory for the DSPs on the RADSPEED DSP card at the user's discretion. This interface includes single bit error correction and double bit error detection (SECDED). A NAND FLASH memory controller, also with SECDED, is provided to support non-volatile memory. A PROM interface is provided for Start-up ROM (SuROM). A preliminary block diagram of the RADSPEED-HB ASIC is shown in Figure 5. Some elements of the architecture are not yet finalized.

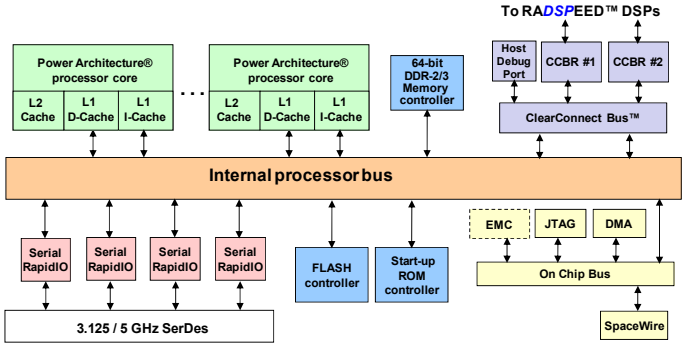


Figure 5: Preliminary RADSPEED-HB ASIC block diagram

6. Critical Components – High Speed Memory

The RADSPEED DSP and RADSPEED-HB devices use advanced external memory architectures to support the processing speed and volume of data that the devices can process. The use of DDR2 memories, while not the state-of-the-art in commercial designs, is leading edge in space applications. The incorporation of standardized memory controller IP allows the use of standardized (JEDEC) external memory modules along with the verified functionality needed to reduce risk in complex system-on-chip (SoC) ASICs. The availability of DDR2 memory modules from various vendors would, on its face, allow us to have a wide selection of vendors to choose from. Since the memory manufacturers are targeting a commercial base, many of the memories are not suited for the space radiation environment. This increases the part selection cost, reduces vendor selection, and incurs the requirement to maintain a stockpile of devices due to lot variability in the radiation performance. Because DDR2 memories have a very tight timing budget for board trace length and data skew, the task is made even harder due to the board constraints needed for space applications. The overall path from the memory controller inside the SoC to the DDR2 devices must be considered during SoC package design through board layout in order to meet the timing constraints. Finally, most DDR2 memory modules are sold in JEDEC standard packages for use directly on boards or for DIMM/SIMM manufacturing. These plastic packages pose another challenge for the space environment and mitigation techniques must be applied.

7. Algorithm Elements and Application Classes

Several application studies have been completed using the ClearSpeed processor on commercial boards. These include a 2009 SAR processing application [3] that explored the efficient operation of two dimensional Fast Fourier Transforms (FFTs) and an image processing application [4] that compared processing performance between the ClearSpeed CSX700 and state of the art FPGA, ASIC and GPU devices. To rapidly make the RADSPEED DSP applicable to the widest group of applications and algorithms, we are collecting, optimizing and benchmarking elements that can be reused by the space community.

Algorithms with real life application which had elements of parallelization that could be benchmarked on the commercial ClearSpeed processor were initially identified as candidates for optimization. By benchmarking on the commercial processor, it becomes possible to extrapolate an estimated value for the RADSPEED DSP prior to the availability of hardware.

Individual algorithms are the building blocks of applications; therefore several applications were broken out into their individual components. The following are the criteria used to determine which application/algorithms would be initially implemented:

- Applications that are computationally intensive with elements of parallelization
- Applications which made use of existing library elements on the commercial ClearSpeed processor
- Applications which had previously been implemented and for which the code was readily available
- Applications which had been used to benchmark other parallel processors for ease of comparison

The Complex Ambiguity Function (CAF) application met all of the criteria that had been identified. Figure 6 shows a list of applications/algorithms and their type that have been identified for use on the RADSPEED DSP.

8. CAF ALGORITHM DISCUSSION

The CAF application[1][6] was considered an excellent candidate for initial benchmarking efforts. The CAF is an algorithm that has elements of parallelization and higher levels of complexity. Its heavy use of Discrete Fourier Transforms (DFTs) makes it an ideal candidate for parallelism. A major advantage is that the ClearSpeed FFT Library can be utilized and benchmarked within an application. Serial code had already been developed for the CAF which offered the advantage of starting with existing code. Most users of the RADSPEED DSP will likely begin with existing code with the goal to optimize it for SIMD processing within a fairly short amount of time and minimal effort. BAE Systems determined that developing in-house experience with optimization would provide benefit to future users. Finally, prior parallelization of the CAF had been performed on other parallel processors. This demonstrated the ability to successfully achieve parallelization and offered an opportunity to develop direct comparisons. These comparisons are underway.

Algorithm /Benchmark	Algorithm Type
Complex ambiguity function (CAF) Surface processing / detection	two-dimensional function of time delay and doppler frequency showing
Edge detection	canny
Matrix manipulation	DGEMM, parallel matrix transpose
Fast fourier transform (FFT)	one dimension FFT, two dimension FFT
Convolution	signal multiplication
Integer math	various
Floating point math	various
Hilbert transform	linear operator used in fourier analysis
FIR filter	low pass finite filter
Data compression	Huffman coding (lossless compression)
Image compression	JPEG
Eigen value / vector	Jacobi
Wiener filter	reduce the amount of noise present in a
Principle component	
Anomaly detection	
Data extraction	
Rock segmentation through edge	rock and target detection
Image / video processing	image filters, video stabalization
Integer sort	bucket
String sort	qSort
Encryption	blowfish
Data fitting	least squares optimization
Harris corner detection (HCD)	feature detection
SSD like algorithms	stereo vision
RANSAC	model estimation
Histogram of oriented gradient	object detection

Figure 6 - Algorithms for RADSPEED DSP

Computation of the Complex Ambiguity Function is the correlation of 2 signals collected from 2 different detectors from an emitter over a range of time and frequency offsets that have been contaminated by noise. The correlation produces CAF planes which are used for determining Time Difference of Arrival (TDOA) and Frequency Difference of Arrival (FDOA) of the emitter. There are several methods that can be used to perform the correlation. .

The coarse-mode CAF processing utilizing the Fast Fourier Transform (FFT) method was selected for implementation. The RADSPEED processor comes with an FFT library function. Because most of the processing burden of the CAF lies in the FFT calculations, having a library that handles the parallelization of the FFT greatly facilitated the effort. The block diagram of the CAF application is shown in Figure 7. Currently the pre-processing, which consists of the Hilbert Transform and the Finite Impulse Response (FIR) filter is being performed on the signal generator. The Hilbert Transform applies the correct format to the sampled vectors and the FIR filter produces the proper delay spacing. When this portion is implemented into the application it will add a 5% processing burden to the current results.

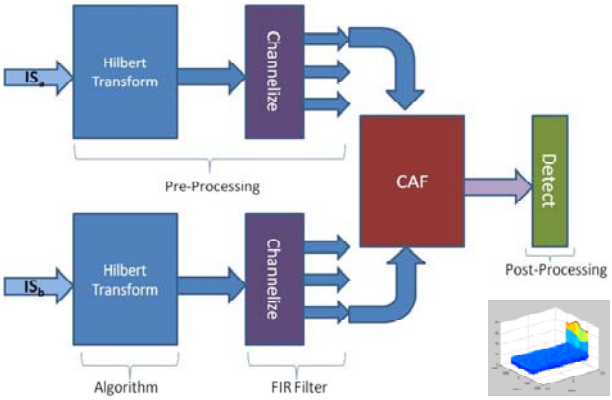


Figure 7 - CAF Algorithm Flow

Figure 7 depicts the algorithm flow. Once the shift and merge has been implemented the signals are broken down into sub-blocks to ease processing burden. FFT calculations performed on these blocks to obtain the CAF planes could be run parallel to each other. There are no dependencies for initial optimization implementation. Optimization was also gained by moving more of the algorithm into the DSP to minimize the host to DSP transfer of data. Further optimization was obtained by taking advantage of the SIMD structure to reduce the transfers from single processing on the Physical Elements (PEs) to multi-threaded processing. These techniques are described in the next section.

9. TOOLSET AND APPLICATION CONVERSION FROM PC TO PPC

The RADSPEED DSP Software Development Kit consists of the following major components.

- A Linux based development environment consisting of a compiler, assembler, linker, simulator, debugger and profiler. The development environment was converted to work with the RADSPEED DSP by ClearSpeed Technology Ltd., the developers of the CSX700.
- A RADSPEED DSP runtime. The ClearSpeed runtime was converted to work with the RADSPEED DSP by ClearSpeed Technology Ltd.
- A host (Windows (Intel), Linux (Intel), vxWorks (PowerPC) based runtime. The host runtime was

converted to work on a PowerPC under vxWorks by BAE Systems.

The development environment is built around the C^n compiler. C^n is a superset of C that has additional features to allow the definition of code which is executed simultaneously on the RADSPEED DSP's Processing Elements (PEs).

Figure 8shows a function which will simultaneously sum a vector on all of the RADSPEED DSP's PEs. The attribute "poly" tells the compiler that elements a, b and c point to are in the local memory of each PE. The attribute "mono" (which is the default, and could be omitted) identifies "d", "cnt" and "i" as variables which reside in "mono" memory.

```
// Vector math, c[] = (a[] + b[])*d
void Vector_Sum(poly float* c,
                poly float* a,
                poly float* b,
                mono float  d,
                mono int    cnt)
{
      int i;
      for (i=0; i<cnt; i++)
      {
         c[i] = (a[i] + b[i])*d;
      }
}
```

Figure 8 - Cn code which sums to vectors simultaneously on all PEs.

Operations which are applied to "mono" variables (such as i++) are executed by the mono processor. In general these are control or indexing operations such as the "for loop" and array indexing in the example.

Operations which are applied to "poly" variables or a combination of "poly" and "mono" variables are executed simultaneously on each PE. In the example the operations in "c[i]=a[i]+b[i]*d" are performed on each PE. The elements "a[i]", "b[i]", and "c[i]" reside in each PE's memory and can have different values. The variables "i" and "d" reside in mono memory and have only one value.

In support of the compiler the development environment contains a linker and an assembler. A cycle accurate simulator can be used when actual hardware is not available. A host based "gdb" debugger can test code running on the RADSPEED DSP or the simulator. The debugger has be extended to support poly variables and the RADSPEED DSP profiler. The profiler allows the identification of bottlenecks in the execution of RADSPEED DSP applications.

The RADSPEED DSP runtime is a collection of standard C libraries (stdlib, mathlib, etc.) augmented with libraries which support PE processing (poly versions of mathlib routines, transfer of data between mono and poly memory, etc.).

The runtime and hardware supports 8 hardware threads. One of these threads is dedicated to the mono/poly memory transfer functions supplied by the runtime while another is dedicated to the main execution thread. The other 6 threads can be used by the applications to create specialized mono/poly memory transfer functions when the generic ones supplied by the runtime are not efficient enough.

10. SINGLE TO MULTI-CORE CONVERSION OF ALGORITHMS

The conversion of the CAF algorithm from a single thread algorithm on a General Purpose processor to a parallel SIMD algorithm was done in several steps. Each step increased the amount of parallelism. Initial development was done on a Windows PC and a CSX600. The more advanced algorithms were then tested on a vxWorks PPC and a CSX700.

When designing an algorithm for the RADSPEED DSP the optimizations should take the following rules into account:

1. Minimize the amount of data transfer between the host and RADSPEED DSP memory.

2. Minimize the amount of data transfer between mono and PE memory.

3. Optimize the algorithm for transferring data between mono and poly memory. This includes minimizing the overhead while doing the transfer and allowing the overlap of PE operations with the memory transfer.

4. Optimize the PE's arithmetic operations by using PE vector math. Vectors math allows simultaneous operations to be performed on a 4 point vector. For example if a, b and c are all 4 point vectors in poly memory then the operation a = b + c is performed on all vectors with one add operation.

Figure 9 is the high level pseudo code for the CAF algorithm. For purposes of data movement the CAF algorithm can be divided into the following steps (the arithmetic operations performed are described in the "CAF Algorithm Discussion"):

1. The "merge_func" function combines two complex data streams (s1 and s2) into a merged fft_size*dt_steps*blocks 3d array. In order to do this s1 and s2 must have (fft_size + dt_steps)*blocks data points. The data points in s1, s2 and the merged array are all complex values.

2. The "ffts" function = perform dt_steps*blocks FFTs of size fft_size on the output of the "Merge" function. This results in a complex fft_size*dt_steps*blocks 3d array.

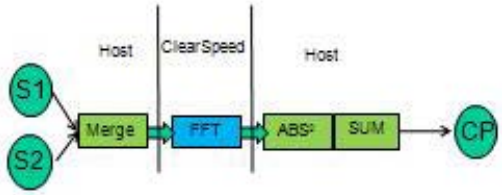
3. The "abs_square" function calculate the abs^2 on each point output by the FFT (this converts each complex point into a real point). This results in a real fft_size*dt_steps*blocks 3d array.

4. The "sum" function perform a element summation of all of the fft_size*dt_steps. This results in a fft_size*dt_steps 2d array.

```
void CAF(
  in  complex
      s1[(fft_size + dt_steps)*blocks],
  in  complex
      s2[(fft_size + dt_steps)*blocks],
  out real
      caf_plane[fft_size][steps])
{
  complex
    merged[fft_size][dt_steps][blocks]
        = merge_func(s1,s2)
  complex
    fft_o[fft_size][dt_steps][blocks]
        = ffts(merged);
  real
    abs_sqr[fft_size][dt_steps][blocks]
        = abs_square(fft_o);
  caf_plane = sum(abs_sqr);
}
```

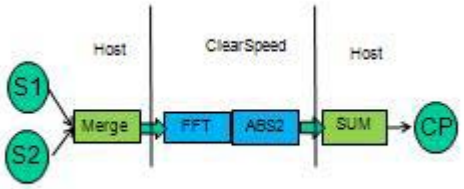
Figure 9 - High Level Pseudo-Code for CAF algorithm

The first optimization was replacing the host fft implementation with the ClearSpeed DFFT host library. The DFFT host library allows the host application to perform FFTs on the ClearSpeed board. Although faster than the host based FFT approach, this optimization was hampered by the fft_size*dt_steps*blocks complex points that had to be transferred between host and ClearSpeed memory.



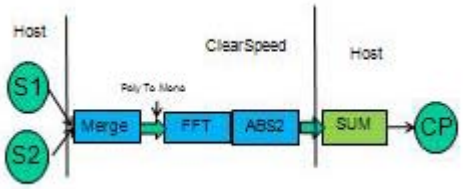
Optimization 1 - Perform FFT on ClearSpeed

The second optimization was to write a custom ClearSpeed program which performed the FFT's and calculated the abs^2 on their output. Normally the ClearSpeed FFT library reads the fft input data from mono memory to poly memoy, performs the fft and writes the data from poly to mono memory. The ClearSpeed FFT library allows the insertion of additional operations before and after FFT execution. For this optimization the abs^2 function was performed before the poly to mono transfer. This decreases the amount of data being written from poly to mono memory by a 2x (wrote real vs complex data). The same 2x reduction was realized for the ClearSpeed processor to host processor transfer.



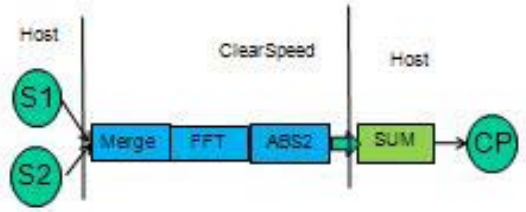
Optimization 2 - Add Abs^2 of FFT Output to ClearSpeed-

The third optimization performed the "merge_func" on the ClearSpeed CSX700. This reduced the transfer of data from the host to the ClearSpeed processor from fft_size*dt_steps*blocks to (fft_size + dt_steps)*blocks (the amount of data transferred to the host was unchanged). As the merged data was calculated it was transferred from poly to mono. When the ffts were executed the output of the merged function was transferred from mono to poly.



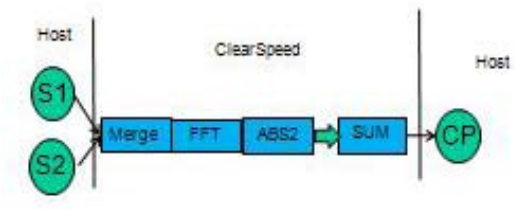
Optimization 3 - Add Merge Function to ClearSpeed

The fourth optimization eliminates the transfer of Merge output from Poly to Mono Memory (and the subsequent read from Mono to Poly Memory by the FFT function). In this optimization the output of the Merge function is immediately processed by the FFT/Abs^2 function.



Optimization 4 - Remove Poly to/from Mono Memory Transfer after Merge.

The fifth optimization performed the "sum" function on the ClearSpeed processor. This reduced the data being transferred to the host from fft_size*dt_steps*blocks real points to fft_size*dt_steps real points. This algorithm writes the data from poly to mono memory after the FFT/Abs^2 functions and then reads them to poly memory for the Sum function.



Optimization 5 - Add Sum to the ClearSpeed

on one core of a 2.8 GHz Intel Pentium processor. The ClearSpeed optimizations use one side (1 mono processor and 96 PE processors) of a CSX600 or CSX700 processor. The RADSPEED DSP estimate scales the CSX700 results to take into account the smaller number of PE's and lower clock frequency. Multiply the CSX600, CSX700 and RADSPEED DSP estimates by 2 for the performance of a complete chip.

The result of the optimizations is shown in Figure 10 The baseline uses the FFTW library for FFT processing and runs

Optimization	iterations/sec 512x1024x72	Improvement vs. baseline	FLOPs/SEC
FFTW (1x2.8Ghz Intel Core)(baseline)	0.358	(base line)	743,272,611.84
Opt 1 (FFTs on CSX600)	0.400	1.1	830,472,192.00
Opt 2 (FFTs and ABS on CSX600)	0.591	1.7	1,227,022,663.68
Opt3 (Merge <-> FFT, ABS on CSX600)	0.929	2.6	1,928,771,665.92
Opt4 (Merge FFT, ABS on CSX600)	1.089	3.0	2,260,960,542.72
Opt5 (Merge FFT, ABS->SUM on CSX600)	2.053	5.7	4,262,398,525.44
Opt5 CSX700 vxWorks(192pes 250Mhz)	4.940	13.8	10,256,331,571.20
Opt5 RADSPEED (152 pes 233Mhz) (estimate	3.645	10.2	7,567,463,310.95

Figure 10 - CAF Algorithm Performance Improvement when using ClearSpeed and RADSPEED DSP

11. SUMMARY

The RADSPEED DSP and RADSPEED-HB ASICs and their critical components have been described. Development of these devices continues to progress toward fabrication. In parallel, efforts to realize these on the advanced printed circuit board assemblies are underway.

We have described efforts to move, optimize and benchmark applications and algorithms that may be used on the RADSPEED DSP, starting with the efforts that have been completed on the ClearSpeed CSX700. We continue to seek potential users for benchmarking their applications in preparation for applying the RADSPEED DSP to their needs.

The RADSPEED DSP represents a major step forward for highly power efficient signal processing in spacecraft applications.

REFERENCES

[1] Robertson, Jeffrey E., "Power Aware CAF Algorithm," 2005 IEEE Aerospace Conference, March 2005.
[2] RapidIO version 2.1 specification, September 2009, http://www.rapidio.org/specs/current
[3] ClearSpeed Technology Ltd, "Synthetic Aperture Radar on CSX700" technical notes, May 2009, http://www.clearspeed.com/applications/syntheticaperturerera dar/index.php
[4] Fijany, A. and Hosseini, F., "Image processing applications on a low power highly parallel SIMD architecture", 2011 IEEE Aerospace Conference", March 2011
[5] McIntosh-Smith, S., "A next-generation many-core processor with reliability, fault tolerance and adaptive power management features optimized for embedded and high performance computing applications", 2008 High Performance Embedded Computing conference, September 2008
[6] D. P. Enright, et al., "Multicore Acceleration for the Complex Ambiguity Function," HPEC 2008, the High Performance Embedded Computing Workshop (poster), 2008.

BIOGRAPHIES

Joe Marshall *is a Senior Principal Systems Engineer working on embedded processor systems at BAE Systems in Manassas, Virginia. He has developed and / or led development of hardware, software and subsystems at GTE, IBM, Loral, Lockheed Martin and BAE Systems with the last 22 years focused on Spacecraft Processing Systems and their applications. He was chair of the AIAA Computer Systems Technical Committee from 2001 to 2006 and specializes in dependable systems, reconfigurable systems, processor architectures, plug and play, high speed interconnects and systems engineering. Joe has a BSEE and MSEE from Purdue University and has been certified twice as a System Architect by Lockheed Martin.*

Dale Rickard *is a BAE Systems Engineering Fellow and Technical Director for the Space Products and Systems product line headquartered in Manassas, Virginia. He has 33 years experience in developing and applying advanced technologies in a variety of aerospace applications at IBM, Loral, Lockheed Martin and BAE Systems. Dale has a BSEE and PhD in electrical engineering from the University of Tennessee.*

Lisa M. Hollinden *is a Principal Systems Engineer, currently working in the development of Space Applications for the RADSPEED DSP. She has worked at BAE Systems for the past seven years where she has been involved in multiple projects working in the fields of Systems Engineering, Signal Processing, Reliability Engineering, Statistical Analysis, and Test and Integration. Lisa is the lead Systems Engineer on the RADSPEED DSP Development program and she was awarded the Bronze Chairman's Award for her work on this program. Lisa earned her BEE and MEE from Catholic University with concentrations in Signal Processing and Communications.*

Jeff Robertson *is a Senior Principal Systems Engineer at BAE Systems in Manassas, Virginia. He has 30 years of experience in embedded systems for both space and terrestrial systems. Currently he is developing applications for the RADSPEED DSP processor and SpaceWire Network Management. He received his BS in Computer Science from the University of Dayton.*

Richard Berger *is a Senior Principal Systems Engineer responsible for the development of advanced products and the insertion of state of the art technologies for aerospace applications. He has over thirty years of circuit, logic, and system architecture design experience in high performance commercial and military semiconductors and processing systems at IBM, Loral, Lockheed Martin, and BAE Systems. Richard has recently been driving the development of complex mixed-signal and digital system-on-chip products. He has also provided management consulting services to a number of companies for the development of complex high technology products in a variety of industries. Richard has published numerous technical papers related to spaceborne processors and radiation effects. Richard received a BSEE degree from Rutgers University in 1978.*

Michael Bear*-is a Senior Principal Systems Engineer at BAE Systems in Manassas, Virginia. He over 25 years of experience in the entire design cycle from advanced device reliability and ASIC design through systems architecture. He is currently the engineering lead on the RADSPEED DSP program and the lead for advanced technology ASIC design. Michael has a PhD in computational physics from George Mason University.*