Application Report
SPRABB8–June 2010

Using TI's Embedded Processor Software Toolkit for Medical Imaging (STK-MED)

Dan Thomas

ABSTRACT

Advances in digital signal processors (DSPs) are driving both high-computational performance and power efficiency into ever increasing application areas. The medical diagnostic ultrasound industry can leverage these advances, particularly as these devices are incorporated into portable and hand carried application spaces. This document includes a collection of standard ultrasound algorithms optimized for TI's TMS320C64x+™ DSP core. These optimized building blocks showcase how ultrasound processing functions can leverage the C64x+™ DSP architecture for efficient performance and power consumption. The goal of the this application report is to shorten customer development time by providing optimized implementations of processing blocks commonly used in medical imaging. It provides an overview of the STK-MED, its benefits and how to begin using it.

Contents

List of Figures

List of Tables

1 Introduction

Due to their high-performance, real-time processing capabilities, their flexible programmable nature, and their power efficiency, DSPs are increasingly being used in medical diagnostic ultrasound machines. With improvements in power efficiency and system integration, these DSPs are helping to drive ultrasound into portable and even hand carried units where computational efficiency is critical for performance and

maximizing battery life. As DSP performance continues to advance, DSPs are proving to be a good solution for more and more of the ultrasound processing chain. The flexibility and programmability of DSPs allows for both field upgradeable products and efficient implementation of new adaptive algorithms not easily realizable in other technologies. These features are making DSPs a valuable processor in ultrasound systems.

In an effort to support this application segment, TI has developed the TI Embedded Processor Software Toolkit for Medical Imaging (STK-MED). STK-MED includes a collection of standard ultrasound algorithms optimized for TI's C64x+ DSP core. These algorithms showcase how ultrasound processing functions can leverage the C64x+ architecture for efficient performance and power consumption. The goal of the STK-MED is to shorten customer development time by providing optimized implementations of processing blocks commonly used in a diagnostic ultrasound system. The source code in the STK-MED can easily be extended or modified to create customized versions for development.

2 Benefits of STK-MED

This toolkit provides many benefits for the medical diagnostic ultrasound system developer.

2.1 Highly Optimized Implementations

The STK-MED provides highly optimized DSP code, taking full advantage of the sophisticated processing capabilities of the TI C64x+ core. These modules can be used by developers to improve their system performance – either through use of the included modules or by using the modules as templates on how to optimize code for the DSP. As such, these modules can be used as building blocks for an ultrasound system, with developers capable of easily adding to or modifying the code as needed for their particular implementation.

2.2 Ease of Software Integration

The STK-MED provides well defined APIs for several common medical diagnostic ultrasound functions. When used as-is, the functional level APIs abstract the implementation specific details from the user. As such the modules can easily be integrated into an existing system to help shorten development time and costs. APIs and header files are annotated in Doxygen [1] format to provide self documenting interfaces in html. Each module is documented with a full algorithmic description, parameterized cycle performance information, memory configuration and requirements, and algorithm references.

2.3 Optimal Implementation References

The STK-MED provides several common functional blocks found in medical diagnostic ultrasound systems. The well structured and documented software allows for function customization and provides coding illustrations of computational optimization methods for the C64x+ core which can be modeled in custom processing functions.

2.4 Ease of Profiling and Evaluation

Each software module includes an easy to use unit test bench as an evaluation and development platform. A set of functional test vectors are provided along with automatic function cycle profiling using TI's Code Composer Studio™ [2]. Custom test vectors can be easily added to the test bench to check functional and cycle performance for any input. The test benches can be run either on a Code Composer Studio device simulator or on the target device. The *TMS320C6455 DSP Starter Kit for Medical Imaging* (SPRW190) [3] is an ideal choice to evaluate and develop with the STK-MED.

3 STK-MED Functional Overview

The basic functional building blocks of a diagnostic ultrasound imaging system are illustrated in the Figure 1 through Figure 3 and separated into front-end, mid-end, and back-end signal processing regions. The STK-MED components are highlighted in orange in the following diagrams.

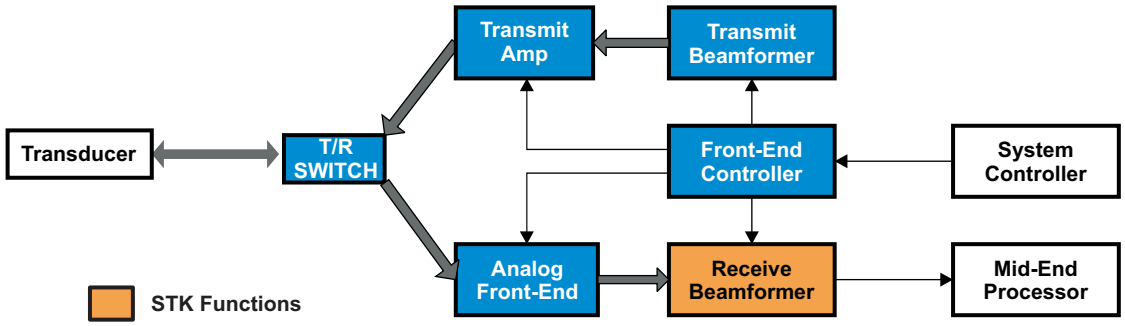

Figure 1. Front End Processing

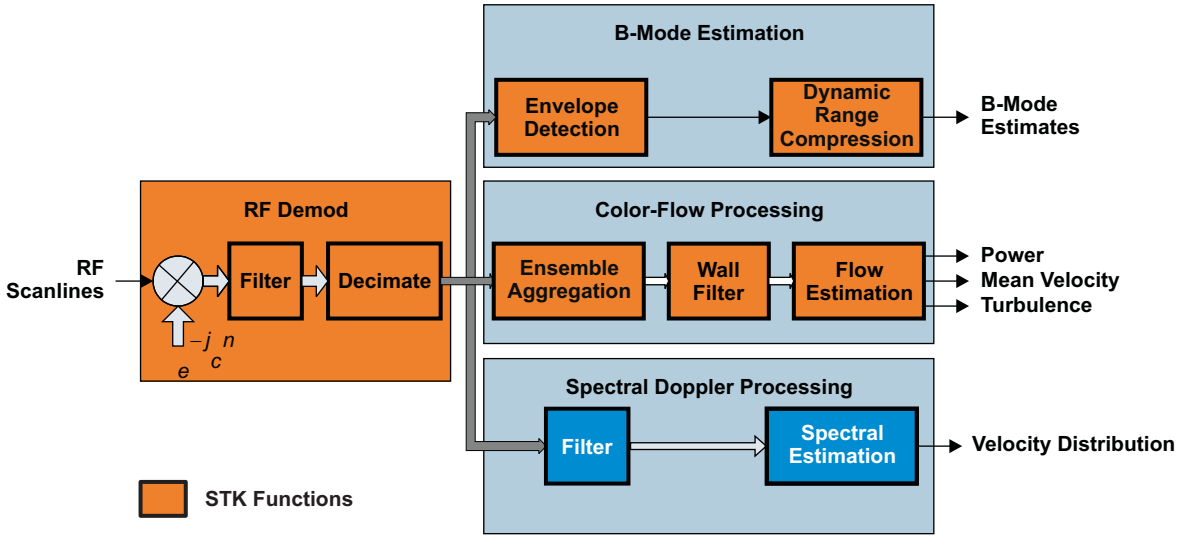

Figure 2. Mid to Back-End Processing

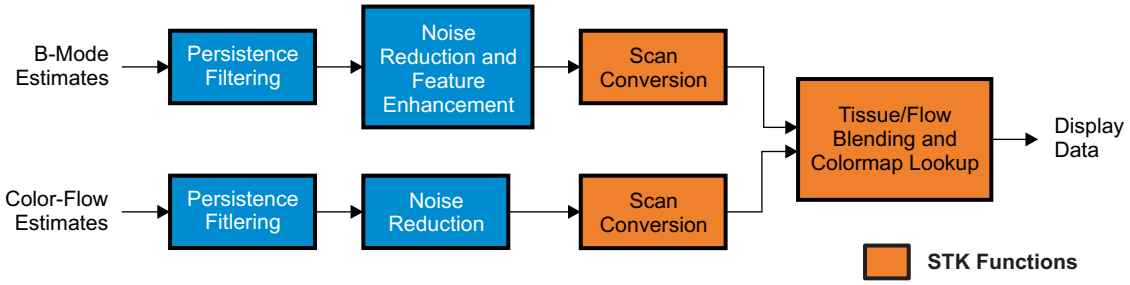

Figure 3. Back-End Processing

TI's STK-MED provides the following functional modules that are typical of a medical diagnostic ultrasound system:

- Optimized kernel implementations and optimized system implementation of B-mode Processing functions for Envelope Detection and Dynamic Range Compression on TI's C64x+ DSP core of the C6472, C6455, and OMAP3530 SoC.
- Optimized kernel implementations and optimized system implementation of Doppler Processing functions for 1D Color Flow, 2D Color Flow, Wall Filter and Power Estimator on TI's C64x+ DSP core of the C6472, C6455, and OMAP3530 SoC.
- Optimized kernel implementations of RF demodulation and decimation
- Optimized kernel implementations of DAS (Delay And Sum) receive beamforming

- Optimized system implementation of Scan Conversion on TI's C64x+ DSP core of the C6472, C6455, DM6437, DM6446 and OMAP3530 SoC.
- Optimized math utilities.

Each software module is described briefly in the following sections.

3.1 B-mode Estimators

The B-mode Processing Unit (BPU) implements two main processing B-mode functions, Envelope Detection and Dynamic Range Compression. The Envelope Detection is a magnitude calculation on the complex input, the complex input values being (16-bit real, 16-bit imaginary) with a 16-bit output. The complex magnitude function is included in the UTIL directory of the STK.

The Dynamic Range Compression operation is defined either by a general compression table which can be supplied by the caller of the BPU process API, or via two parameters which define the shape of logarithmic function used to build the compression table during the BPU config API. Since the compressed dynamic range output is 8-bit, a Binary Search algorithm of a 256 item table is employed as an efficient means to implement the compression. There are two APIs for the Binary Search function; one that takes 16-bit input and produces 8-bit output and another that takes 32-bit input and produces 8-bit output. The Binary Search function is included in the UTIL directory of the STK.

3.2 Doppler Color Flow Estimators

The purpose of Doppler color flow estimators are to provide the necessary tools to estimate blood flow parameters like flow velocity, turbulence and power, using the decimated RF demodulated data. There are a large variety of techniques to achieve this basic goal and the underlying algorithms lend themselves to a large degree of customization. Nonetheless, an efficient processing chain example, including DMA operation to allow for efficient internal memory processing, is provided in the toolkit.

3.2.1 Doppler Color Flow Estimator 1 Dimension

This module includes functions to compute the complex autocorrelation and 1D power estimates from the ensembles of input data. It also includes functions to use these complex autocorrelation and power results to compute the flow velocity (1D) and flow turbulence (1D) as explained in [4]. The 1D estimators operate at a significantly lower computational complexity than the 2D estimators.

3.2.2 Doppler Color Flow Estimator 2 Dimension

This module estimates the blood flow velocity, turbulence and power using 2D auto-correlation based technique. The advantages of 2D-autocorrelation technique over 1D-autocorrelation technique are:

- Reduced bias caused by narrow-band assumptions
- Higher velocity precision
- Accurate detection of low turbulence estimate

3.3 Wall Filter for Color Flow

This module filters the echoes reflected from tissue and vessel walls, so that velocity is estimated only from the moving red blood cells. This module implements the infinite impulse response (IIR) filtering operation. However, since it needs to filter very short sequences of ensemble sizes, its transient performance is of primary importance. To gain greater control over its transient performance, a state-space formulation of the IIR filter [5] is used. This filter can be used with different types of initialization, the most common forms being: zero, step, and projection initialization schemes. The basic operation done for filtering for each scan-line set is a multiplication of a complex input data matrix with a real coefficient matrix.

3.4 RF Demodulator

This module removes the carrier signal from the echoes produced by the tissue and vessel walls. In this module, each scan line is processed independently by mixing it with sinusoids to produce in-phase and quadrature components, followed by low pass filtering using finite impulse response (FIR) filter to prevent aliasing, and then decimating the filtered output by the decimation factor.

3.5 Scan Conversion

The scan converter translates the scanned echo data or color flow data (velocity, turbulence or power) into a representation suitable for displaying the image on a monitor such as an LCD screen. Following features are supported:

- B-mode sector shape of up to 100 degrees sector angle and linear shape
- Color-mode sector shape of up to 100 degrees sector angle and linear shape
- 2x2 bilinear interpolation
- Tissue and flow decision
- Color mapping for tissue and flow
- 422 video format output

3.6 Receive Beamformer

The purpose of the receive beamformer unit is to provide the necessary software modules needed to perform receive beamforming on a DSP. In receive mode, the beamformer produces a signal that represents the echo characteristics of the tissue along a given scan line for a time duration that corresponds to the desired depth of imaging. Since multiple transducer elements are needed and they are not collocated with one another, their spatial position with respect to the scan line must be well understood so that the elements are triggered appropriately in transmit mode and the echo signals are properly processed in receive mode. A common beamformer implementation in the receiver is to delay and sum the signals from the transducer elements in such a way as to produce a signal that has been focused at each sample point along the given scan line .

3.7 Math Utilities for STK-MED

The software toolkit includes several optimal precision and low complexity implementations of utility math functions used within the functional modules. The math utilities include test benches and profiling. The functions provided used in the STK for the diagnostic ultrasound functions include:

- Four quadrant inverse tangent function
- Complex magnitude function
- Division function
- Binary Search function

4 Using the Software Toolkit

The STK-MED is designed for ease of use and rapid integration. As such, it can be used for quick prototyping into an existing TI DSP system using well defined public APIs. The highly optimized functions can be used as-is or used as optimal processing models for user function adaptations. The STK-MED can also be used as a reference for newly developing systems. The parameterized functions can easily be benchmarked based on existing test cases in the STK-MED, or evaluated on customized user test cases easily added to the module test bench.

4.1 Software Toolkit Directory Organization and APIs

At the top level the STK-MED consist of several directories, with a directory for each module:

- BPU – B-mode Processing Unit
- DPU – Doppler Processing Unit
- RF DEMOD – RF Demodulator
- RXBF – Receive Beamformer
- SCU – Scan Conversion Unit

Each module contains a consistent directory structure and file organization as shown in Figure 4.

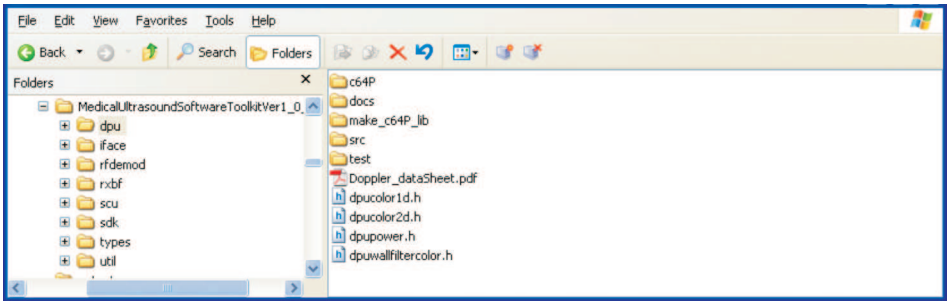

Figure 4. Software Toolkit Directories

All the top directory level, all public APIs are published along with the module datasheet, as can be seen above for DPU.

Each top level API is Doxygen annotated which produce HTLM based CHM files. These files are linked to in the Release Notes and providing HTML linked navigation between APIs, API parameters, and header files. Figure 5 shows a screen shot from the DPU documentation.

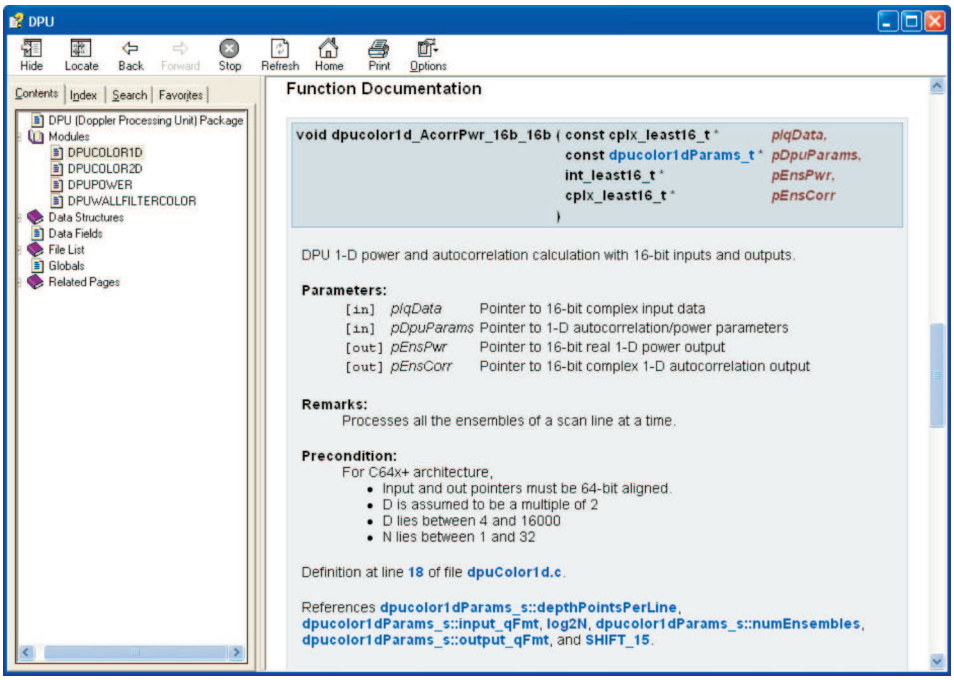

Figure 5. Doxygen Output Example

The well defined and concise APIs allow for:

- Easy integration into an existing system
- Fast benchmarking of module functions with user specific parameters
- Jump-start for porting legacy implementations into DSP

4.2 Using the Software Toolkit Test Benches

Each module includes a comprehensive unit test bench for ease of evaluation and development that runs using Code Composer Studio software. The test bench program for each module is located at:

`<module>\test\src\test<module>.c`

The Code Composer Studio project file for each test bench is located at:

`<module>\test\ccsProj_c64p\<module>\test<module>.pjt`

Each module also includes a representative set of unit test vectors organized as shown in Figure 6:

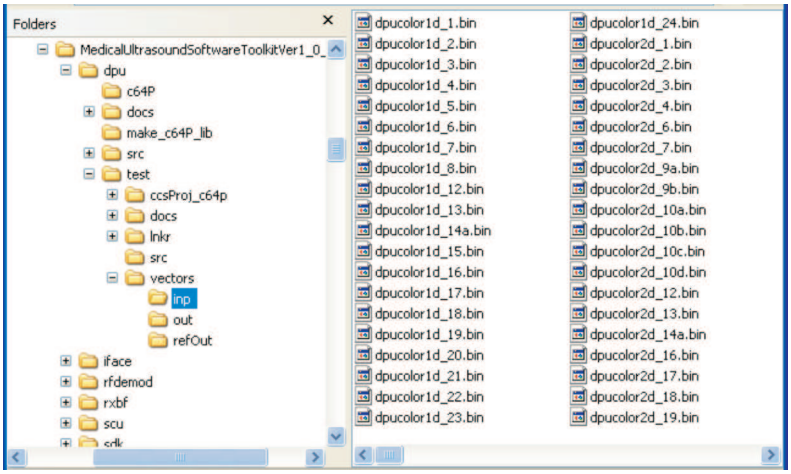

Figure 6. Test Vectors Example

<module>\test\vectors\inp - Contains binary input test vectors

<module>\test\vectors\out - Contains binary output vectors generated by the test bench

<module>\test\vectors\refOut - Contains binary reference output test vectors

All vectors are fixed point input/output of size defined in the API.

The tests are controlled and configured in a straightforward manner by input text files:

<module>\test\vectors\inp\<module>TestList.txt - List of active numbered test cases

<module>\test\vectors\inp\<module>_param<N>.txt - Test case N input parameters

As an example, see file `dpupower_param1.txt` that provides the input parameters for dpupower test case #1:

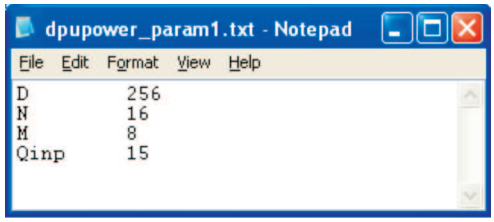

Figure 7. Test Case Parameters Example

The test bench, in this case `testdpupower.c`, first reads the file `dpupowerTestList.txt`. As "1" is the first datum in the file, representing test case #1, the test bench then accesses `dpupower_param1.txt`. From this file the input parameterization for the function of this test case is read. The test bench accesses the input file for the test case #1, `dpupower_1.bin`. Based on the input parameterization the input data arrays are read into the test program. The test program calls the function API to process the test case input data set. The output data is then written by the test program into the appropriate output test vector of the vectors\out directory, in this case `out\dpupower_1.bin`. The test programs continues to process all test cases present in the *Test List* file. The output bin files for each test case can be compared for bit-exactness with the reference output files, located in `vectors\refOut`.

Cycle profiling is automatically generated for each API call for each test case as well, written into the file `<module>_cycles.txt`.

The output pane of Code Composer Studio also prints the test progress (see Figure 8):

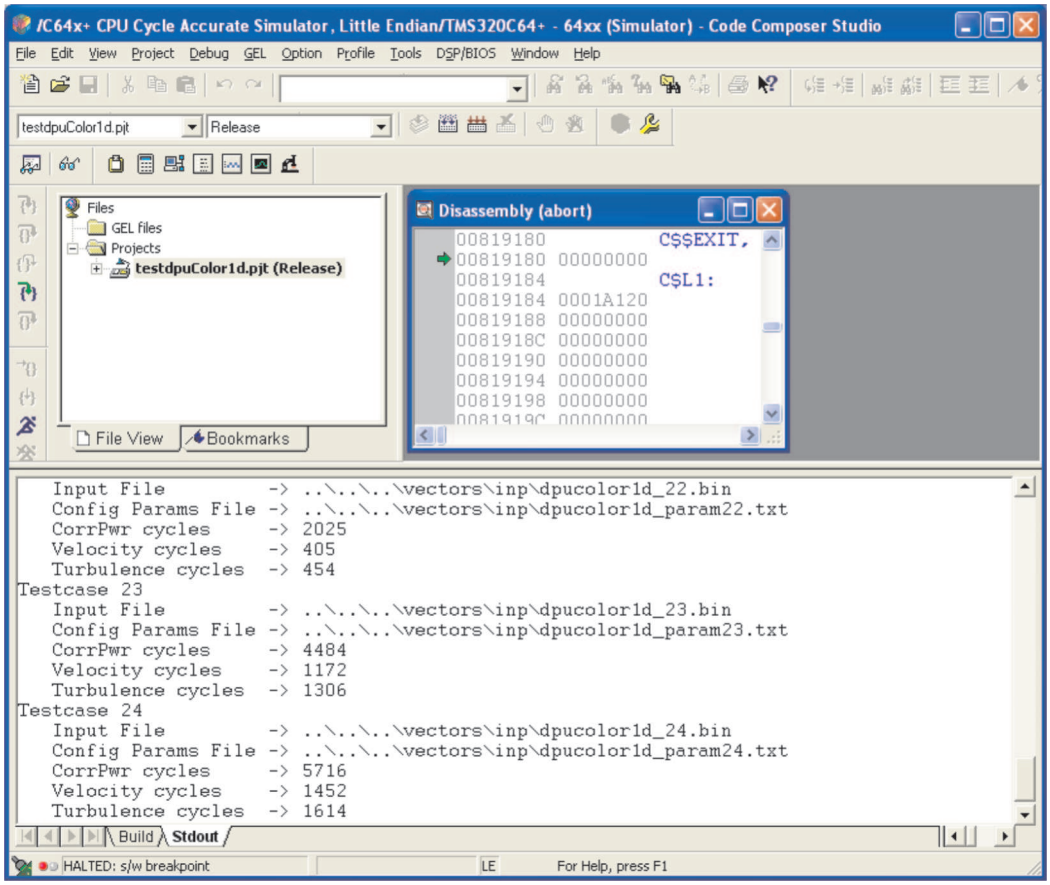

Figure 8. Code Composer Studio Running a Test Case

Figure 9 shows an example cycle count listing output for dpucolor1D. Check the STK-MED release data sheets for the latest cycle performance for each module.

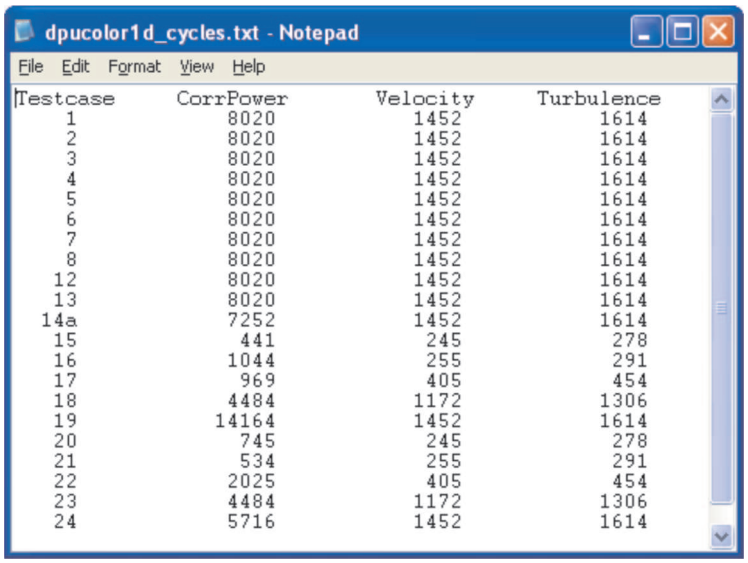

Figure 9. Module Cycle Profiling Example

For the software toolkit user the module test bench provides a readymade framework for:

- The addition of new test cases – Using the existing test cases as examples, the user has complete freedom to create new test case input parameterization. Likewise input and reference output can be user generated to validate functional and cycle performance at any applicable data set.
- Regression testing – as implementations are modified by you, quick regression validation is possible.

4.2.1 Test Bench Optimization on the Evaluation Module (EVM)

The memory interfaced to the C64x+ core on all TI devices containing the C64x+ core are composed of a memory hierarchy. The first level consists of data (L1D) and program (L1P) caches, “level 1”. The second level in the memory hierarchy is the “level 2” or L2 memory. The third level is usually the external SDRAM memory. In general, different TI devices have differently sized L1D, L1P and L2 memories. The C6455 and similar high performance DSPs have 32 KB of L1D, 32 KB of L1P and generally 1 MB to 2 MB of L2, with a maximum of 256 KB of L2 configurable as cache and the rest as flat memory or SRAM (Static RAM). When accessing the level 1 caches, the execution of program on the C64x+ results is no cycle penalty beyond the c64x+ core cycles, except for bank conflicts within the L1 memories themselves depending on algorithm access patterns. The bank conflicts in data memory are in general rare for typical signal processing algorithms. It is desirable to exercise the module APIs in the test bench environment without incurring any cache misses of L1D and L1P during the API execution itself after an initial loading in the cache for optimal execution of the modules on devices like C6455.

To this end, the test bench has been structured to (a) divide the execution of long test vectors into smaller chunks of data of suitable size that ensure that all active data during the module’s execution fits within 32 KB L1D cache and is contiguous in the address space, which prevents L1D cache misses and (b) the placement of code in memory is such that there are no L1P cache misses during the module execution. These optimizations can be observed in the test bench C code (see `.. / test/src/<module>test.c`) and in the linker command files (`../test/lnkr/c64p/*.cmd`) for the test.

When integrating the modules in a system, if the module performance is degraded below that of the toolkit, then attention to cache misses will be required, as illustrated by the test bench programs. The device cycle accurate simulator in CCS can be used to profile cache misses and other statistics to help improve performance. When CCS is configured to use this simulator upon bring-up, the menu of Tools->Simulator Analysis will be visible. Related context sensitive help (F1) can be looked-up to learn how to configure the statistics of interest and how to profile.

4.3 Cycle Performance Optimization in the Software Toolkit

The C64x+ DSP core has the processing power to implement sophisticated medical diagnostic ultrasound functions. During every clock cycle in the TI C64x+ platform, the instruction fetch, dispatch, and decode units deliver instructions to the eight functional units.

The C64x+ includes these features and more:

- Advanced VLIW CPU with eight functional units, including two multipliers and six arithmetic units
- Executes up to eight instructions per cycle
- Each multiplier can perform two 16x16 bit or four 8x8 bit multiplies every clock cycle.
- Quad 8-bit and dual 16-bit instruction set extensions with data flow support
- Support for non-aligned 32-bit (word) and 64-bit (double word) memory accesses

The STK_MED takes full advantage of the advanced signal processing capabilities of the C64x+ to efficiently implement the ultrasound algorithms. Users can use these optimal implementations as is, or leverage them as reference models for optimal implementation methods.

Efficient Implementation of Ultrasound Color Doppler Algorithms on Texas Instruments' C64x™ Platforms (SPRAB11) [7] illustrates in detail the methodology of mapping an algorithm efficiently on to the C64x+ architecture. It then goes on to address specific functions used in medical ultrasound diagnostic doppler processing.

This approach attempts to optimize the loop kernels that dominate performance and are expected to take the maximum cycles of these algorithms. The most efficient data flow and CPU instructions are estimated for each of the algorithmic operations. Then, the various DSP core processing units where these instructions could be mapped to are considered and, finally, the cycles on the unit that are most loaded are used as the estimate of the algorithm's complexity.

As an example, consider the dpuColor1D. This set of functions computes the complex autocorrelation and 1D power estimates from the ensembles of input data. It also includes functions to use these complex autocorrelation and power results to compute the flow velocity (1D) and flow turbulence (1D). The Flow Parameter Estimator section of *Efficient Implementation of Ultrasound Color Doppler Algorithms on Texas Instruments' C64x™ Platforms* (SPRAB11) [7] describes in detail the mapping of the correlation and power computations to the C64x+ architecture.

The STK-MED kernel cycle performance for dpuColor1D, then is characterized as:

Flow Power and complex correlation: 1.5 * D * N + 7 * D + 27

Flow Velocity computation: 5 * D + 92

Flow Turbulence computation: 11/2 * D + 94

where,

D = Number of scan-lines

N = Number of ensembles

For a typical operating point, the computational loading of a 1GHz DSP can be estimated as follows:

D	N	Total Cycles per Scan Line	Total Cycles per Frame (1)	Total Cycles per Second (2)	Percentage of 1.0 GHz DSP
256	16	10837	1387136	41614080	4.16%

(1) Assuming 128 scan lines per frame

(2) Assuming 30 frames per second

4.4 Scan Conversion – An Example of an Efficient Implementation on a Memory Constrained Device

Many algorithms require the processing of large amounts of data per frame. Interpolation is required in Scan Conversion because all points of specified input geometry do not map exactly on the raster output grid of pixels and there are typically more points in the output than in the input. Interpolation is the core processing work in scan conversion, so its computation cost is important. Most DSP devices contain only a limited amount (32 KB) of locally cacheable memory, L1D memory. The STK-MED Scan Conversion uses an efficient DMA scheme for transferring input samples, interpolation coefficients, and outputs between external memory and internal L1 and L2 SRAM.

The interpolation coefficients and addresses required for all the pixels are stored in the external memory. During scan conversion operation, the coefficients and addresses for a horizontal line are transferred from external memory into L1 SRAM. A ping-pong buffering scheme is then used for DMA transfer. During scan conversion operation, the output data for a horizontal line are transferred from L1 SRAM to external memory using DMA.

Pre-scan samples need to be brought into internal memory in order to achieve high performance for the scan conversion calculations. Because the scan conversion is implemented to operate along the horizontal line and the natural sequence of samples in memory is along scanlines, access of a wide range of memory for input samples is required to process every horizontal line. Most of the today's DSP devices have limited on-chip internal memory and the amount of internal memory may not be enough to hold an entire frame of pre-scanned ultrasound image.

The smart DMA scheme utilized in the STK-MED is summarized below. This scheme keeps track of the required inputs for every horizontal line of scan converter outputs and only accesses the necessary samples from the external memory to internal memory, thus reducing the overall memory requirements.

1. Divide sector image into sub-blocks along both the angle and scan line direction.
2. Determine the sub-blocks of inputs that are required for processing each horizontal line.
3. Allocate one DMA channel for every sub-block that is required. The maximum number of DMA channels required is the maximum number of sub-blocks required for line processing.
4. Pre-compute EDMA tables, including the source and destination addresses of all the sub-blocks.
5. Scan converter checks for every horizontal line if any new sub-blocks are required to bring into the internal memory. If yes, the scan converter configures the DMA channels and chains all the DMAs together, then trigger the first DMA to start data transfer.

This method is addressed in detail in *Ultrasound Scan Conversion on TI's C64x+ DSPs* (SPRAB32) [8].

5 Summary

This application report serves as a brief introduction to the TI Embedded Processor Software Toolkit for Medical Imaging. The STK-MED is available as a source code distribution for no charge at http://www.ti.com/medicalimaging [9]. It can shorten customer development time by providing optimized, easy to use implementations of signal processing blocks commonly used in a diagnostic ultrasound system on the highly efficient TI C64x+ architecture.

6 References

1. http://www.doxygen.org/
2. http://focus.ti.com/docs/toolsw/folders/print/ccstudio.html
3. *TMS320C6455 DSP Starter Kit for Medical Imaging* (SPRW190)
4. C. Kasai, K. Namekawa, A. Koyano, and R. Omoto, *Real-Time Two Dimensional Blood Flow Imaging Using an Autocorrelation Technique*, IEEE Trans. Sonics Ultrasonics, vol. SU-32, pp. 458- 464, 1985.
5. Steinar Bjaxum, Hans Torp and Kjell Kristoffersen, *Clutter Filter Design for Ultrasound Color Flow Imaging*, IEEE Trans. on ultrasonics, Ferroelectrics, and Frequency Control, Vol. 49, No. 2, pp. 204-216, Feb. 2002.
6. Thomenius, Kai E., *Evolution of Ultrasound Beamformers*, Proceedings from the 1996 IEEE Ultrasonics Symposium, Vol. 2, November 3-6, 1996, pp.1615-1622.
7. *Efficient Implementation of Ultrasound Color Doppler Algorithms on Texas Instruments' C64x™ Platforms* (SPRAB11)

8. *Ultrasound Scan Conversion on TI's C64x+ DSPs* (SPRAB32)
9. http://www.ti.com/medicalimaging

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Communications and Telecom	www.ti.com/communications
DSP	dsp.ti.com	Computers and Peripherals	www.ti.com/computers
Clocks and Timers	www.ti.com/clocks	Consumer Electronics	www.ti.com/consumer-apps
Interface	interface.ti.com	Energy	www.ti.com/energy
Logic	logic.ti.com	Industrial	www.ti.com/industrial
Power Mgmt	power.ti.com	Medical	www.ti.com/medical
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Space, Avionics & Defense	www.ti.com/space-avionics-defense
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video and Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless-apps

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265