VisualDSP++ Development and Debugging Environment

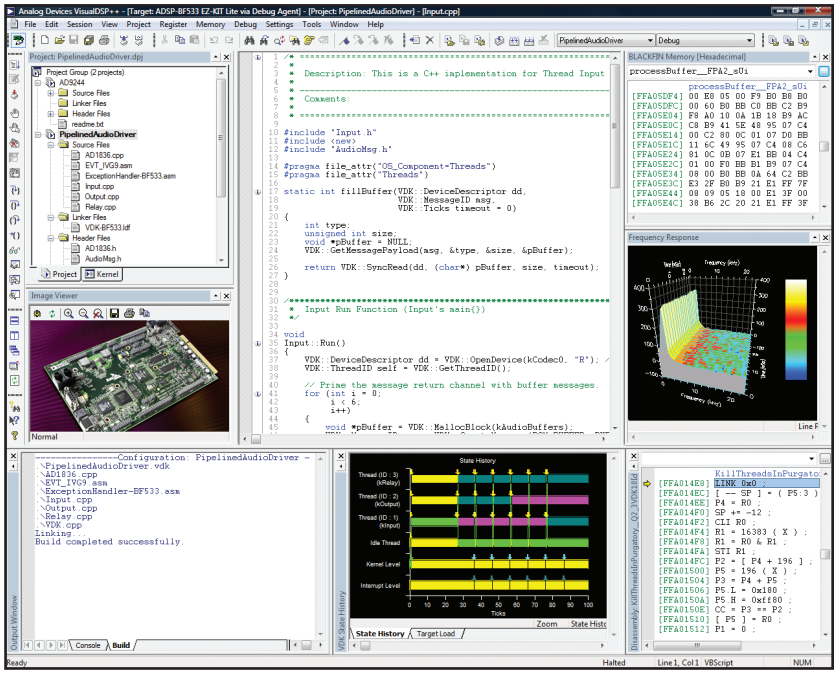
Features

Integrated Development and Debugging Environment

- Multiple project management
- Profiling and tracing of instruction execution
- Automation API and automation aware scripting engine
- Multiple processor (MP) support
- Background telemetry channel (BTC) support with data streaming capability
- Statistical profiling
- Graphical plotting capabilities
- Cache visualization
- Execution pipeline viewer
- Compiled simulation

Efficient Application Code Generation

- Native C/C++ compiler and enhanced assembler
- Profile-guided optimization (PGO)
- Expert linker with profiling capability
- Integrated source code control
- TCP/IP and USB support for Blackfin Processors
- Processor configuration/start-up code wizard for Blackfin Processors
- VisualDSP++ kernel (VDK) with multiprocessor messaging capability
- System services and device driver support for Blackfin Processors
- File system support for Blackfin Processors



Overview

VisualDSP++® 5.0 is a state-of-the-art software development environment targeting the Analog Devices embedded processor portfolio. With the embedded software engineer and signal processing-intensive applications specifically in mind, VisualDSP++, coupled with Analog Devices in-circuit emulator (ICE) and EZ-KIT Lite® evaluation products, provides best-in-class capabilities for developing demanding real-time applications.

Platforms and Processor Support

VisualDSP++ supports Analog Devices' Blackfin® Processors, SHARC® Processors, and TigerSHARC® Processors. Windows XP,® Windows 2000,® and Windows Vista® are supported.

Develop High Performance Applications Quickly

At the heart of VisualDSP++ is a robust and powerful C/C++ compiler. The compiler consistently delivers industry-leading performance on standard benchmarks, ensuring that all but the most performance-demanding applications can be written entirely in the C language, accelerating development time while maintaining a portable code base. The compiler is backed by a rich library of signal-processing routines, allowing easy access to hand-coded, optimized implementations of FFTs, FIRs, etc. The Blackfin and SHARC compilers support MISRA-C:2004 for safety-critical embedded systems (*www.misra-c.com*).

The ANSI-C compiler is also augmented with popular language extensions and enhancements to make it more amiable to existing code bases. Examples include GNU GCC extensions, ETSI fractional libraries, and multiple heap support.

A compiler's overriding mission is to produce correct code, so there are occasions when the compiler must take a conservative approach to a code sequence when a more aggressive approach could have been taken if certain constraints could be guaranteed by the programmer. The VisualDSP++ compiler supports a broad range of pragma that allow the programmer to better exploit the compiler while maintaining C language neutrality. Just as important, the compiler has the ability to feed back advisory information to the programmer, offering further improvements to a code sequence should the programmer be able to make certain guarantees about it. This information is displayed seamlessly in the VisualDSP++ main editor window. This "lifts the veil" off the "black box" that compilers are often, and accurately, accused of being.

Backing the compiler is a powerful assembler and linker technology. Analog Devices' processors are noted for their intuitive algebraic assembly language syntax, and the VisualDSP++ assembler extends that ease of use with the ability to import C header files, allowing for symbolic references into arbitrarily complex C data structures. Binary data can be "included" directly into assembly source files, creating an easy way to add blocks of static data (such as audio samples and bitmaps) to an application. The VisualDSP++ linker is fully multicore and multiprocessor (MP) aware, allowing for the creation of cross-linked, multiexecutable applications in a single pass. Other powerful capabilities of the linker include dead code and data elimination, code and data overlays, section spilling (i.e., automatic overflow from internal to external memory), and automatic short-to-long call expansion.

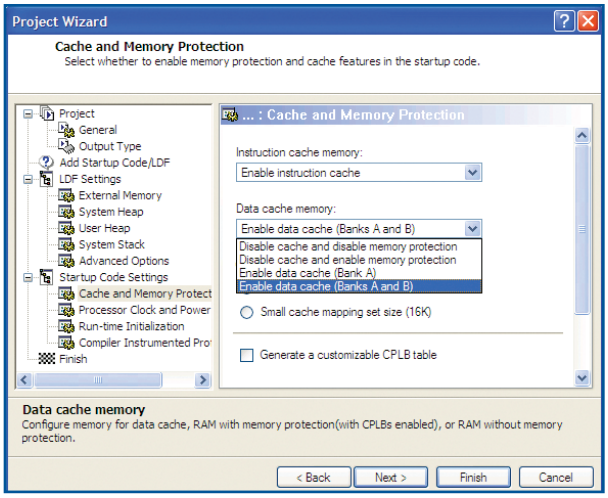
Leverage Proven Application Infrastructure

VisualDSP++ goes beyond robust code generation tools, providing considerable application infrastructure and middleware out of the box to speed application development. The VisualDSP++ kernel (VDK) is a robust, royalty-free, real-time operating system (RTOS) kernel. It provides essential kernel features in a minimal footprint. Features include a fully preemptive scheduler (time slicing and cooperative scheduling are also supported), thread creation, semaphores, interrupt management, inter-thread messaging, events, and memory management (memory pools and multiple heaps). In MP environments, MP messaging is also provided. Configuration of these elements is done graphically with code wizards to speed the creation of new threads and interrupt handlers. VDK has been available for multiple releases of VisualDSP++ and is now a key component of products shipping from a number of high volume vendors. Several commercial RTOSs are also available from select Analog Devices third parties.

Blackfin Processors can take advantage of the system service library (SSL), which provides consistent, easy C language access to Blackfin features such as the interrupt manager, direct memory access (DMA), and power management units. Clock frequency and voltage can be changed easily at run time through a set of simple APIs. Interrupt handling can be live, fired at the time of the event, or deferred to a later time of the application's choosing. A device manager integrates device drivers for on- and off-chip peripherals. VisualDSP++ includes ever expanding device driver support for all on-chip peripherals and off-chip devices found on Analog Devices EZ-KIT Lite and EZ-Extender® products. The SSL is OS-neutral and can be run as a standalone or in conjunction with an RTOS.

Built upon the system service library, the file system service (FSS) provides a portable and extensible means of accessing mass storage media from the Blackfin Processor. Support for the ADSP-BF548 EZ-KIT Lite development board is provided with VisualDSP++ 5.0 for FAT file systems on the attached hard disk drive, supplied SD card, and USB flash.

As embedded applications become increasingly part of the connected world, the ability to rapidly add reliable Ethernet or USB connectivity to an application can often make or break a development schedule. For Blackfin Processors, VisualDSP++ includes a tuned port of the open source LwIP TCP/IP stack. An example application showcasing an embedded Web server is among the highlights of this support. For Blackfin Processors and SHARC Processors, USB 2.0 device connectivity is provided. Bulk and asynchronous transfer modes are supported out of the box, with USB-IF logo certified embedded and host applications provided with full source code.



Source code generation.

Wrapping all of these powerful tools and libraries together is the VisualDSP++ state-of-the-art integrated development and debugging environment (IDDE). The IDDE includes full-featured editing and project manage-ment tools with incremental builds, multiple build configurations ("Debug" and "Release," for example), syntax-coloring editor, and many other code editing features. Makefiles can be imported and exported freely. For Blackfin Processors, many application attributes can be configured graphically, enabling point-and-click access to SDRAM setup, stack and heap placement, power management, clock speed, cache setup, and more.

Debug and Tune Your Application with Ease

The ability to develop a high performance application is often gated by the visibility into your running system that your debugger provides. VisualDSP++ excels in this regard, with best-in-class debugging and inspection support. Robust fundamental C language source debugging (source-level stepping and breakpoints, stack unwinds, local variable and C expression support, memory and register windows) serves as a foundation upon which multiple innovative and unique tools rest.

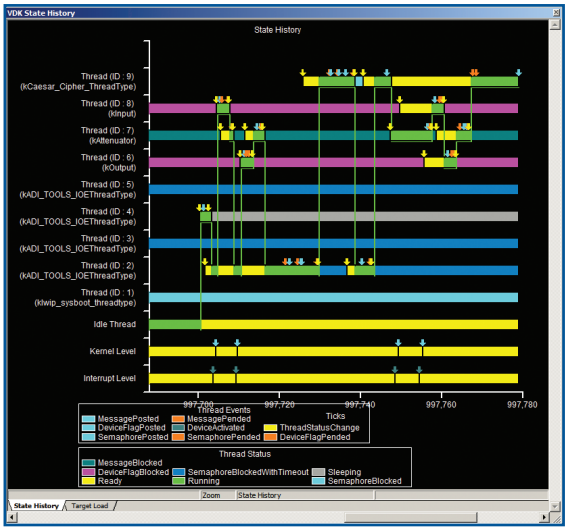
VisualDSP++ supports a variety of debugging targets. Most common is a JTAG connection to an EZ-KIT Lite board or to a custom target board by means of Analog Devices emulator products. However, there will be

occasions where closer inspection in a simulated environment may be required. VisualDSP++ provides core cycle-accurate simulators, allowing inspection of every nuance of activity within the processor, including visualization of the processor's pipeline and cache. These simulators are robust and highly accurate, so much so that they are used by Analog Devices' own silicon designers for validation. A second simulator is available to Blackfin Processor users—a high speed functional simulator. Using proprietary just-in-time (JIT) technology, the simulators have the ability to simulate millions of cycles per second on the most modest of host PCs. Effectively, this means that what used to be an overnight run is now a 10-minute coffee break, and what was once a coffee break is now a near-instantaneous simulation.

As many of the most performance-demanding applications process a signal of some sort, comprehensive memory plotting is a cornerstone of VisualDSP++ debugger support. VisualDSP++ provides multiple views, from basic (line plots) to sophisticated (eye diagrams and waterfalls) to pinpoint anomalous data sequences in your application. Image viewing in a number of data formats is also available.

Users of the VDK get unparalleled visibility into the internals of the kernel. Status on a per-thread basis is available, as is a comprehensive pictorial history of kernel events and CPU loading. Thread changes, posted and pended semaphores, and other kernel events are captured in this display.

For Blackfin Processors and SHARC Processors, inspection, or even



Kernel event history.

application stimulation, from the debugger at run time is possible through the use of the processor's background telemetry channel (BTC). BTC allows for an arbitrary number of communication channels to be established between the host debugger and the application. Channels may go in either direction, so BTC can be used to read and write data as the processor runs. Scalar values or entire arrays may be serviced by a channel. Arrays read from the target can even be plotted in real time.

MP users get the same compelling set of debugging features across all processors, unified into a single debugging interface. Individual windows can be made to "float" their focus to whichever processor currently is the debugger's focus, or they can be "pinned" to a specific processor so their contents do not follow the debugger's focus. To further aid MP debug, synchronous run, step, halt, and reset are also provided.

The Analog Devices patented statistical profiler offers unprecedented and unique visibility into a running application. Operating completely nonintrusively to the application, the application is polled thousands of times per second and a statistical view of where an application is spending the majority of its time is quickly assembled. This tool can be used to easily inspect an application for unexpected hotspots (suggesting the need to move a key routine from external to internal memory, for example). Simulator targets provide a completely linear profiling view. For Blackfin Processors, traditional instrumented profiling is also available.

Going even further, the VisualDSP++ compiler is able to act upon profiling information. Profile-guided optimization (PGO) is a technique that allows the compiler to instrument an application, run the application, and then make a second pass compilation, exploiting the information that was gathered during the run of the application. This gives the compiler unique insight on a block-by-block basis, allowing it to optimize with a level of granularity that is not possible with a tool that operates only on a file-by-file basis.

Integrate into Your Existing Environment

A development tool suite is always a part of an organization's larger software engineering environment. VisualDSP++ has been designed to operate in a larger environment.

Since an embedded engineer is often developing on a new platform while maintaining existing products that were likely developed with earlier versions of the tools, VisualDSP++ can be installed discretely an arbitrary number of times at a variety of release levels, allowing engineering to easily switch between current and legacy versions of VisualDSP++.

To better integrate to source code control (SCC) systems, VisualDSP++ is able to connect to any SCC provider that supports the Microsoft® common source code control (MCSCC) interface. This interface is supported by all leading SCC vendors. VisualDSP++ goes one step further by supporting the control of VisualDSP++ itself within a source code control system.

The ability to robustly test an embedded application is enabled through a comprehensive automation application programmers interface (API). Using Microsoft's language-neutral automation technology, nearly every feature of the graphical environment is available to script authors. Applications can be rebuilt, downloaded, and run from a simple script executed from the command line or from within a custom test harness framework. The automation API is supported by C++ and VBScript examples for all API calls, though any automation-aware language can be used.

For prototype runs and/or small volume deployment, an Analog Devices emulator can be used to flash a program onto your custom system. Accessible through the Automation API, the flash programmer can be scripted, making it possible to develop a turnkey user interface for use by a production floor technician or other individual not familiar with VisualDSP++. Device drivers are provided for all flash devices found on EZ-KIT Lite products, and these drivers can be easily adjusted to support an arbitrary flash device. The Stand-Alone Flash Programmer enables the development engineer to script or automate this process with a license-