Better Digital Signal Processing Performance;
Lower Costs With Innovative IntervalZero RTX Real-time Platform

I. Overview

Digital Signal Processors (DSPs) have specialized architectures that are optimized for heavy math computations and they have dominated the real-time digital signal processing market as a cost-effective solution for many complex designs.

Although DSPs remain a viable choice for many systems, developers are finding that proprietary DSPs are no longer required to perform real-time digital signal processing. Instead, developers are migrating to the IntervalZero RTX Real-time Platform, which comprises multi-core x86 general purpose processors (GPPs), the Windows operating system and RTX real-time software to outperform DSPs, to significantly reduce costs, and to streamline development cycles.

This paper compares DSP-based systems with systems based on the RTX Real-time Platform and provides an overview of the economic and operational benefits that are catalysts for the increasing migration from DSPs. IntervalZero's symmetric multiprocessing-enabled RTX software provides hard real-time functionality for Windows operating systems, including the breakthrough Windows 7 with its touch and gesture capabilities.

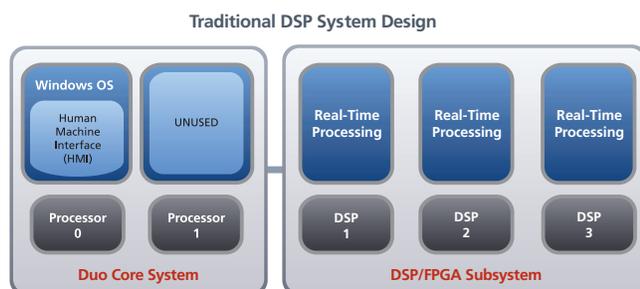## II. System Diagrams
a. DSP-Based Platform

Diagram 1 - DSP

DSPs rely on specialized architectures designed for heavy math computations, but not for general processing. Because of DSPs' computational focus, GPPs are used for human machine interface (HMI) and general purpose functions such as input and output (Diagram 1). Additionally, it should be noted that DSPs require separate memory devices for each device and separate buses are needed for all inter-processor communications (i.e. PCIe or Serial).
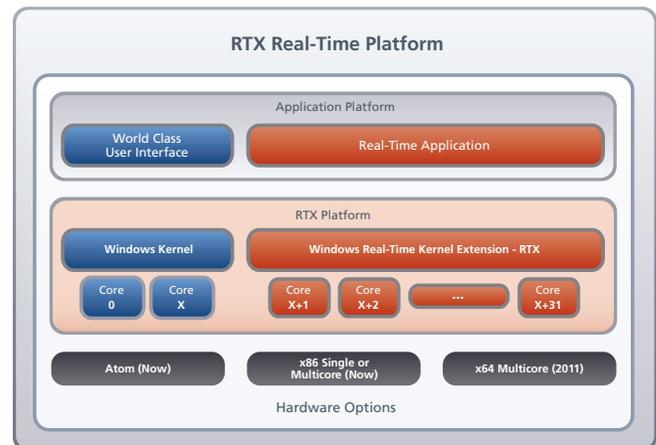
b. RTX Real-time Platform

Diagram 2 - RTX

The RTX Real-time Platform takes a different approach, with the multiple cores divided up for dedicated real-time processing and general purpose processing. Diagram 2 illustrates a multi-core x86 device with cores dedicated for Windows and cores dedicated to RTX for real-time signal processing functions. Unlike the DSP design, memory is shared using the same high-speed bus, allowing for data sharing and inter-processor communications.

The following chart shows a comparison of the two architectures and in the sections that follow, this paper will discuss the merits of the architectures in detail.

III. Comparison Chart: DSP vs. RTX Real-time Platform

Category	DSP Platform	RTX Real-time Platform	RTX Benefit
Hardware Design	• Custom board design • Long development cycle • Difficult to modify hardware • Requires In Circuit Emulators (ICE)	• (COTS) • Short development cycle • Easy to change hardware • Local Debug – no ICE needed	• Reduced Cost • Reduced Risk • Faster Time To Market
Drivers / BSP	• Custom Drivers – device specific	• Standard Drivers	• Cost • Productivity
System Communication	• Custom protocol • Custom buses	• Shared memory • Standard APIs	• Ease of use • Future proof
Memory	• Chip Down – hard to change • Limited selection • Limited memory reach (1GB) • No MMU	• PnP/COTS Modules – easy to change • Large selection • Large memory reach (>4GB) • MMU – Virtual memory	• Reduced system size, cost and complexity
RTOS	• Proprietary	• winAPI	• Familiar API
Development Environment	• Proprietary Tools • Manufacture specific • Proprietary Compilers	• Standard x86 Tools • Microsoft Visual Studio • Intel and Microsoft Compilers	• Common tool chain throughout the product
Code Base	• Assembly and C • Architecture specific	• C, C++, C# • Universal / Portable	• Uses other higher level languages for non real-time
DSP Libraries	• Processing libraries from OEM	• Intel supplied IPP library • Other 3rd party libraries	• No requirement to use OEM proprietary libraries.
Benchmarks	• 1.25 GHz - Max core speed • 20 GFLOPS / Device Max	• 3.0 GHz - Max core speed • >50 GFLOPS / Device Max	• Faster • Higher performing

IV. Hardware Design:

The x86/x64 multi-core advances from Intel and AMD are changing the way systems developers meet real-time signal processing needs. New GPPs not only have multiple cores with extremely high clock speeds, but also perform complex math efficiently. They deliver several times the performance of standard DSPs, as can be seen from the benchmarks in the comparison chart.

Additionally, multi-core x86 devices are delivered with Commercial off the Shelf (COTS) hardware, which is not usually an option for DSP users. COTS hardware rather than custom hardware translates into reduced costs, shorter time to market, and less risk.

a) Processor Selection

When replacing DSPs with x86 multi-core, attention should be focused on the number of cores needed, as well as which x86 processor family best fits the system requirements. As a rough guideline, there can be a 1:1 relationship between DSPs and x86 cores. Because of the extremely high clock speeds with the x86 cores, the design will likely require fewer cores, but the 1:1 ratio is a conservative start, with room for optimization.

Both Intel and AMD have different processor families based on cost and power. For example, Intel's Atom and AMD's Fusion devices are focused on embedded systems requiring low power and cost minimization.

b) Board Design

In DSP-based systems, engineers spend a great deal of time selecting the right processor so that the board can be designed and built in time for the software. Having to make hardware commitments early in the design often creates challenges and any miscalculation can lead to a hardware redesign, which will have a negative impact on the delivery schedule.

When using the RTX Real-time Platform there are no strict deadlines for choosing and designing the hardware. Because RTX is x86 based, standard COTS hardware is available for both development and production. With no custom boards or drivers required for development, the final hardware selection is performed later in the design cycle. This reduces risk and greatly improves the chances of releasing products on time.

Also, because RTX is based on an x86 Windows architecture, engineers can use PCs as both development and target machines. No separate target system with specialized in circuit emulators is needed to develop and debug code. As the comparison chart shows, RTX's use of COTS hardware reduces both engineering costs and risks.

c) Communication
DSP-based systems rely on custom hardware and software interfaces when communicating between devices. Serial lines and sometimes PCI buses are used for inter-processor communications. These custom interfaces are troublesome to design on custom hardware and can be difficult to upgrade as requirements change.

The RTX Real-time Platform uses shared memory and formalized APIs to communicate between the processes. Because everything is running on a single device, it is easy to share data and messages between threads running on different cores. This communication architecture makes programming easy and scalable.

d) Memory Selection
While DSPs require separate memory devices, with the RTX Platform memory is consolidated into a single memory device. Also, the x86-based platform can use standard off-the-shelf memory modules (i.e. 4GB SDRAM DIMM) and memory modules can be easily changed as system requirements dictate. DSPs require on-board memory chips because of strict timing and routing requirements. Switching out memory chips is difficult and requires board changes to add or remove memory. The DSP-based approach is more costly, complex and less flexible than the RTX Platform.

V. Software Design:
Software advances have also had significant impact on DSP-based systems. The demand for standardized tooling in lieu of the proprietary tool sets is a recent change for many DSP developers. Programming DSPs requires proprietary tools and lower-level languages such as C and Assembly. Many engineering teams find that proprietary DSP tools and low-level languages require specialized expertise, which is often very difficult to find and expensive to acquire. The use of standardized development tools and higher-level languages, such as Visual Studio and C++, not only increases productivity but also greatly reduces engineering costs.

a) General Purpose OS:
The demand for complex graphical user interfaces (GUIs) has also impacted DSP- based systems. Customers are increasingly seeking elaborate touch-and-gesture-based user interfaces on top of a real-time subsystem. DSPs have never had the GUI and I/O support to satisfy most complex system requirements. That is why GPP's running general purpose operating systems are typically found next to DSPs to handle all of the I/O and the complex user interfaces.

As a result of the strong demand for powerful GUIs, Windows is becoming the preferred operating system because of its abundant tools support and its standardization. The use of standardized tools like Visual Studio and the large support structure of Microsoft's Developers Network make transitioning from DSPs to RTX straight forward.

b) Real-time Operating System:
DSPs use proprietary RTOSs from device manufacturers such as DSP/BIOS from Texas Instruments (TI) or VDK from Analog Devices (ADI). These DSP-based

RTOSs are quite capable, but have significant limitations because of the lack of communications among the different DSP cores. Each DSP runs its own application and very little inter-processor communication is used.

The RTX Real-time Platform implements a single symmetric multiprocessing scheduler across all of the cores in the real-time subsystem (Diagram 3). Instead of a separate RTOS and application running on each real-time core, the RTX Platform uses a single real-time scheduler to schedule threads across a number of different cores. The flexible SMP scheduling model makes synchronization

and communications among the different cores/threads simple and easy to implement.

Load balancing can be easily implemented from the APIs provided by the RTX Subsystem. The RTX APIs enable threads to be moved between processors during runtime.
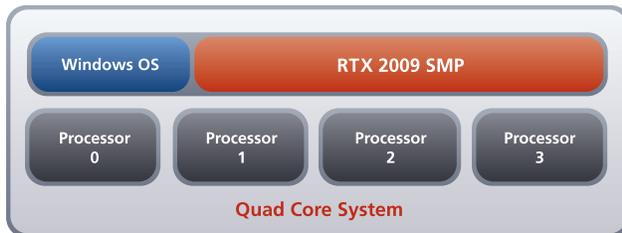

Diagram 3 – RTX SMP-enabled Scheduler

This scalability translates well when moving the application to systems with different core configurations. When cores are either added or removed from a system, the user can easily use logic within their application to load balance threads as needed.

c) Development Tools

DSP manufactures provide proprietary tools such as Code Composer Studio from TI and VisualDSP++ from ADI. While these tools are very useful for developing DSP-based applications, market demand and cost pressures are providing impetus for more standardized tooling. Additionally, DSPs' low-level tooling and proprietary coding practices do not promote code reuse and portability. This translates into longer development times and higher risks. Many developers are finding that working with proprietary DSP tools is not only challenging, but also costly to maintain.

RTX uses Microsoft's Visual Studio which is the industry standard tool for x86/x64- based programming. Finding engineering talent for Microsoft tools is easier and more cost effective than is the case with DSPs. Leveraging the support and robustness of Visual Studio and the Microsoft Developers Network (MSDN) makes for increased productivity and reduced costs. Most engineering teams prefer the use of standardized tools for their robustness and their support.

d) Code Base

Migrating code base to the RTX Platform is straight forward.

When considering a port, the developer needs to be aware that there are two parts to an RTX application. First, there is the general purpose, or non real-time processing and then there is the DSP/real-time processing. The non real-time/general purpose code will directly port over using a standard Visual Studio project and will run on the Windows operating system. The DSP/real-time code will also be built using Visual Studio, but it will be running on the RTX controlled cores (real-time subsystem).

DSPs are usually programmed in both C and Assembly. While the Assembly code is not portable and will need to be written in C or C++, the C code can be ported over using Visual Studio. The RTX Platform enables real-time users to code efficiently in C++ rather than having to start with C and Assembly programming. While DSPs do have some C++ support, object-oriented languages lose too much efficiency to be useful when compiled for DSP architectures.

The RTX Platform's strong support for C++ is a distinct advantage. Through RTX's use of powerful x86 devices, programmers can use C++ to increase productivity.

By keeping the code base in higher level languages like C/C++, portability and code reuse can be leveraged to reduce costs and risks.

e) DSP Libraries

Texas Instruments and Analog Devices both provide optimized DSP libraries to help developers with performance and time to market. To take the place of these DSP libraries Intel created the Integrated Performance Primitives (IPP). The IPP library is essentially a collection of optimized functions (i.e. DSP, imaging, video, etc.) for the multi-core x86 architectures. The IPP library helps to increase productivity and performance by providing optimized routines for the most common DSP-based functions.

VI. Benchmarks:

There are a number of ways to measure performance between processors. Because the focus is on DSPs, measuring the number of floating point operations per second (GFLOPS) is used in this comparison.

Benchmarks	DSP	Intel Multi-core
	• 1.25 GHz - Max core speed • 20 GFLOPS / Core	• 3.0 GHz - Max core speed • 32 GFLOPS / Core

The comparison above is between aC66xx multi-core DSP from Texas Instruments and an Intel i7 Sandy Bridge device.

The TI C66xxx @ 1.25 GHz outputs 20 Single Precision GFLOPS per core. The Intel i7 Sandy Bridge processor outputs 32 Double Precision GFLOPS per core. Although the GFLOPS performance is very strong with the TI DSP, it does not match the raw performance and speed of the Intel processors.

VII. Summary:

The RTX Real-time Platform marries the powerful Windows 7 interface with the real-time signal processing capabilities of Intel's and AMDs multi-core architectures. Using standardized COTS hardware and standardized tooling greatly simplifies the engineering effort and reduces cost. The RTX Platform increases innovation, portability and scalability while also reducing costs.

There will always be a need for digital signal processing, but because of the many hardware and software advances, dedicated DSPs are an option, but not a requirement. For complex systems requiring powerful graphical user interfaces along with real-time signal processing, there are higher-performing, more-scalable, less costly options.

IntervalZero