WHITE PAPER

Eric Biscondi,
Manager,
System & Architecture

Tom Flanagan,
Director,
Technical Strategy

Frank Fruth,
Director,
Software Development

Zhihong Lin,
Strategic Marketing Manager

Filip Moerman,
Senior System Architect,
Communications Infrastructure and Multicore
Texas Instruments

TEXAS INSTRUMENTS

Maximizing Multicore Efficiency with Navigator Runtime

Introduction

Multicore processors bring new challenges to programmers. More than half of the cost of a multicore project comes from software development. Multicore programming is particularly challenging with Asymmetric Multicore Processors (AMP) where different types of processing elements, such as RISC and DSP cores, reside in the same device. This is primarily due to minimal or no operating system (OS) support for resource management and load balancing, which results in poor scalability and inefficient resource utilization. Texas Instruments' innovative KeyStone II multicore architecture is designed to simplify multicore programmability by providing specialized hardware to facilitate scheduling and load balancing. Through these facilities, KeyStone II has achieved a performance breakthrough in multicore programming.

The Challenges of AMP Programming

As multicore technology evolves, more and more SoCs are designed to have Asymmetric Multicore architecture to achieve cost efficiency as well as higher performance. A typical AMP has heterogeneous cores that run on different operating systems, hardware accelerators and distributed memory that may not be shared by all cores. In the case of Symmetric Multicore Processors (SMP), where the cores are identical and run the same OS with the same shared memory architecture, using inter-core communications, scheduling and load balancing provided by the OS is relatively straightforward. Programming AMP devices requires a new level of parallel programming skills to control and coordinate different cores and OSs to achieve the level of stability and performance that comes with single core or SMP programming.

Traditional asymmetric multicore processing requires static partitioning of the multicore resource at compile time. This is often challenging because the software loading at runtime cannot be determined in advance, especially for cutting-edge technologies, such as 4G LTE, LTE advanced and cloud computing. A typical solution is to leave excess headroom to ensure the system will function correctly in the worst-case scenario. The impact of over allocating resources is inefficient resource utilization, which results in increased product cost. On the other hand, manually re-partitioning and optimizing the software can induce tremendous software effort when initially undertaken or when new features are introduced, field enhancements are requested or requirements change.

The efficiency of synchronization and Inter-Processor Communication (IPC) is essential in multicore programming. The lack of proper hardware support for various synchronization and IPC mechanisms results in poor multicore utilization and reduced system performance due to excessive software overhead.

This is even more true in AMP multicore systems, where software scalability and flexibility are difficult to achieve.

Innovating with Multicore Navigator

Multicore Navigator is an innovative packet-based infrastructure that facilitates data movement and multicore control. TI's heterogeneous KeyStone architecture combines DSP CorePacs, ARM® CorePacs, hardware AccelerationPacs and I/O peripherals. They are physically interconnected with TeraNet and logically interconnected through Multicore Navigator. In TI's

KeyStone II architecture, Multicore Navigator includes queue manager with 16K hardware queues which typically hold pointers to packets that consists of descriptors and data payload, eight uRISC cores with 3,200 MIPS, packet DMA for data movement and a hardware data structure that supports 1 million descriptors. PacketDMA is also built into the AcclerlationPacs and I/O subsystems, so that Multicore Navigator can move data from any element to any end point without core intervention.

Multicore Navigator provides a unified interface for CorePacs, AccelerationPacs and I/O using hardware queues for different system endpoints as shown in Figure 1. This provides an AMP system with a common communication methodology for all IP blocks. Traffic routing, IPC, resource management, scheduling and load balancing are managed by Multicore Navigator hardware by leveraging built-in uRISC cores in the queue manager which are designed to optimize and expedite data flow. Tasks are dispatched and distributed by the queue manager to the least loaded cores or IP subsystems on demand.

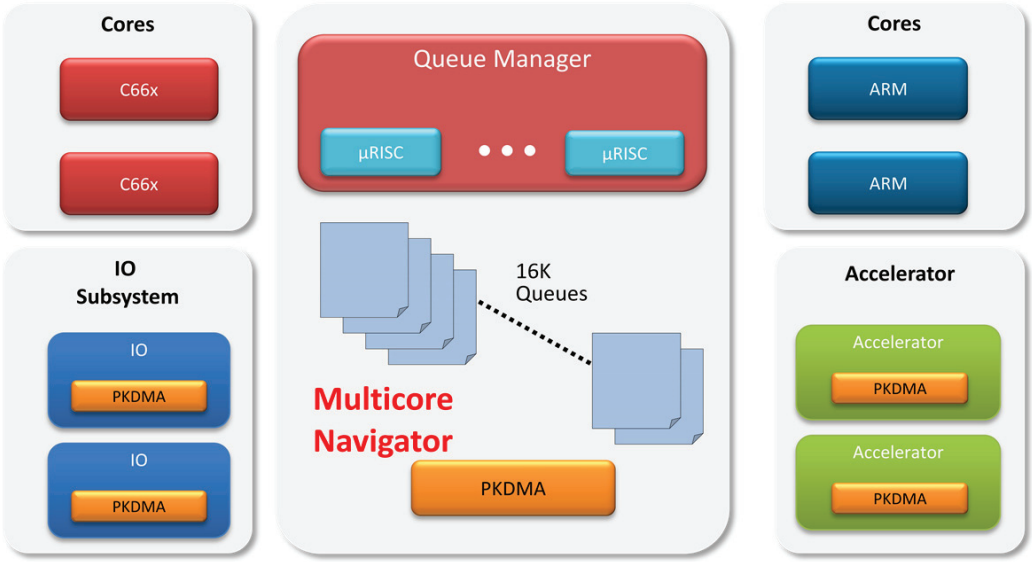

Figure 1. Multicore Navigator subsystem

Multicore Navigator provides a highly efficient inter-core communication mechanism. Hardware queues and packet DMA are the basic building blocks for IPC. Certain queues are designed to be able to generate interrupts to cores for IPC. The uRISC core inside Multicore Navigator enables autonomous queue monitoring and management with programmable interrupt notification. Multicore Navigator substantially eliminates software overhead, reduces synchronization latencies and boosts IPC throughput. In addition, it enables lock-free programming models. Figure 2 illustrates IPC using Multicore Navigator.

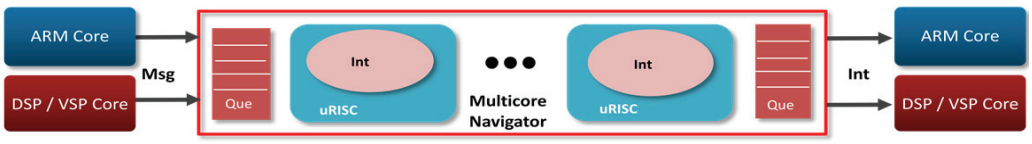

Figure 2. Inter-Processor Communications with Multicore Navigator

Maximizing AMP Multicore Efficiency through Navigator Runtime

While addressing the challenges of multicore in TI's KeyStone II silicon architecture, the benefits of the hardware architecture are only truly achieved by application software developers bringing out the full multicore performance. On the software front, TI is investing in standard programming methodologies which will make the benefits of KeyStone II with Multicore Navigator transparent to the application. There are numerous multicore programming trends evolving as standards in the industry and in academia. A common theme to all of these approaches focuses on application software developers using certain means to describe their application parallelism through language expressions which are later mapped to the underlying runtime. The runtime handles the mapping of the processing to the underlying hardware architecture.

Navigator Runtime is a thin and scalable software layer that enables Multicore Navigator to achieve a higher level of parallel programming performance providing increased scalability, portability and efficiency. The combination of Multicore Navigator and Navigator Runtime is a powerful, unique solution for the challenges of AMP programming.

The main function of the Navigator Runtime is to distribute jobs over multiple cores. Jobs are put into virtual queues waiting to be executed. Central scheduling is performed by the uRISC cores embedded in Multicore Navigator hardware. Jobs are selected based on priority, atomicity and locality by the scheduler then distributed to a software dispatcher which is the essential part of Navigator Runtime residing in each of the cores. The dispatcher sends each job for execution by a processing element which could be threads in a core, an AccelerationPac or an I/O endpoint.

Taking advantage of Multicore Navigator, the abstraction of job producer and consumer is done through Navigator Runtime. By leveraging embedded uRISC cores for centralized scheduling (without consuming MIPS on the primary DSP or ARM® cores), low overhead, low latency and high throughput of 250K jobs per core can be achieved for unmatched parallel-programming performance.

Figure 3 depicts the Navigator Runtime concept and its interaction with Multicore Navigator.

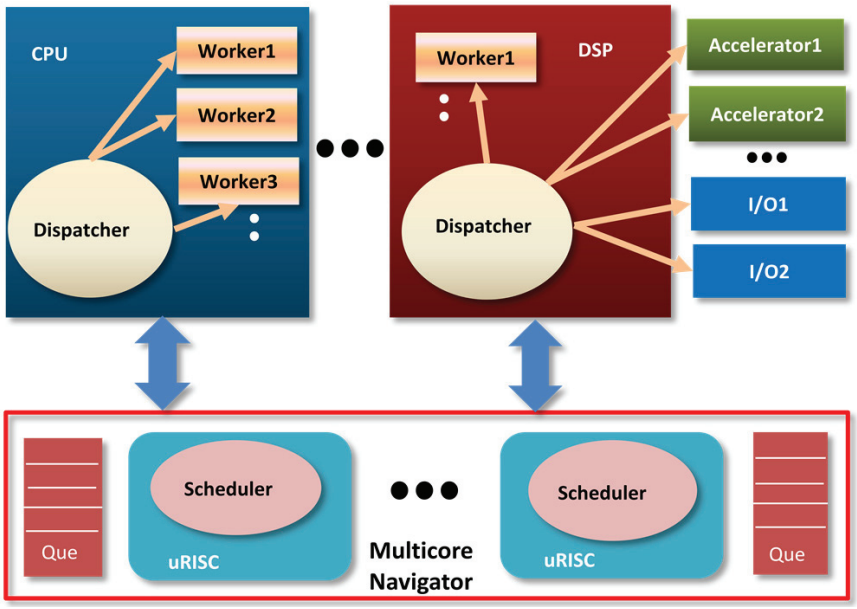

Figure 3. Navigator Runtime interacting with Multicore Navigator

Multicore performance can be measured using speedup which is defined as sequential execution time in a single core divided by multicore execution time. In an ideal scenario, the speedup of an eight-core system will equal 8. In reality, and due to multicore overhead such as multicore bus arbitration, memory access latencies, cache coherency management, synchronization and IPC, the typical speedup is far from ideal. Navigator Runtime consumes very little overhead aiming to achieve close to ideal speedup to maximize multicore performance.

In an example of LTE uplink physical layer processing, a sequential code is broken into 1,024 jobs for antenna data processing, channel estimation, equalization, etc. An average job has 4K input data and 2K output data that resides in shared memory. Navigator Runtime will be used to schedule and dispatch those jobs into eight different cores, so the speedup is measured as the following:

$$\textit{8-core speed up} = \frac{\textit{Sequential code execution time in a single core with data in local L2 memory}}{\textit{Parallel 8-core execution time with data in shared DDR3 memory}}$$

In a parallel eight-core execution case, several Navigator packet DMA channels can be allocated to preload the data from DDR3 into local L2 memory before processing and posting data from L2 into DDR3 after the processing, acting like the CPU cache operation to reduce memory access latencies. As a result, the measured speedup in a KeyStone device shows that with a job size of 32K cycles, the benchmark data measured from an eight-core KeyStone device can achieve a speedup of 7.8. With a job size of 16K cycles, it can achieve a speedup of 7.7, which is very close to an ideal speedup of 8. Navigator has significantly improved in KeyStone II, as compared to KeyStone I, with 4 times uRISC engines to enable more resource for scheduling, double the packet DMA channels and double the hardware queues and descriptors to provide better execution throughput.

Figure 4 demonstrates KeyStone Navigator Runtime speedup with different job sizes comparing the ideal speedup measured from 2–8 cores.

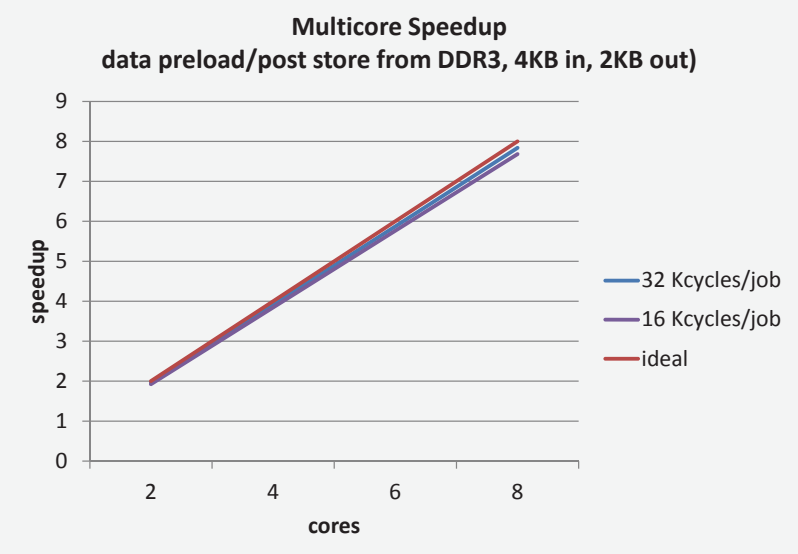

Figure 4. Multicore speedup with Navigator Runtime

TI's KeyStone II architecture also features 6MB of on-chip shared memory (MSMC memory) across all heterogeneous cores. Memory access performance with MSMC is very close to L2 memory access. When data is put in the MSMC, preloading and post storing of data using Navigator is not needed and can achieve similar speedup as above. Large on-chip share memory provides unique advantages for multicore performance with low system latency compared to alternative solutions.

Navigator Runtime can support a variety of system applications, or integrate with high-level multicore programming paradigms such as OpenMP.

OpenMP is an Application Programming Interface (API) that supports multi-platform shared memory multiprocessing programming. It consists of compiler directives, runtime library routines and environment variables. In OpenMP the user identifies parallelism in their software by using language directives (e.g., pragma), and potentially using tools to help. OpenMP pragmas are read by an OpenMP-compatible compiler which transforms the sequential code annotated with pragmas into parallel code and inserts calls into the OpenMP runtime. The runtime latency and overhead performance of a multicore programming methodology will limit the parallelization that can be achieved for an application on a specific device. Lower latencies and overheads provide greater opportunities for parallelization in an application and, in turn, achieve greater multicore efficiency.

Initially, OpenMP was specified for shared memory architectures. Support for distributed memory and heterogeneous processor architectures is currently under discussion. TI's Navigator Runtime can serve as the runtime system for OpenMP. Multicore synchronization and IPC can be efficiently handled by queues and packet DMA engines in Navigator. Early benchmarks demonstrate that using Navigator Runtime as OpenMP runtime can significantly reduce the pragma construct overhead. This can significantly increase the level of parallelization inside a multicore system, so that the programmer can focus on identifying the parallel tasks. The scheduling and load balancing is automatically managed by Navigator Runtime with simplified programming and maximized multicore efficiency.

Figure 5 on the following page illustrates a KeyStone-based AMP programming flow using Navigator Runtime and Open MP. TI's Code Gen Tool v7.4, which comes with Code Composer Studio™ Integrated Development Environment, is an OpenMP-ready compiler. This programming flow is generic so it can be applied to wide variety of multicore applications and scale to a different number of cores and system dimensions. The goal is to enable multicore performance with the simplicity of single-core programming.

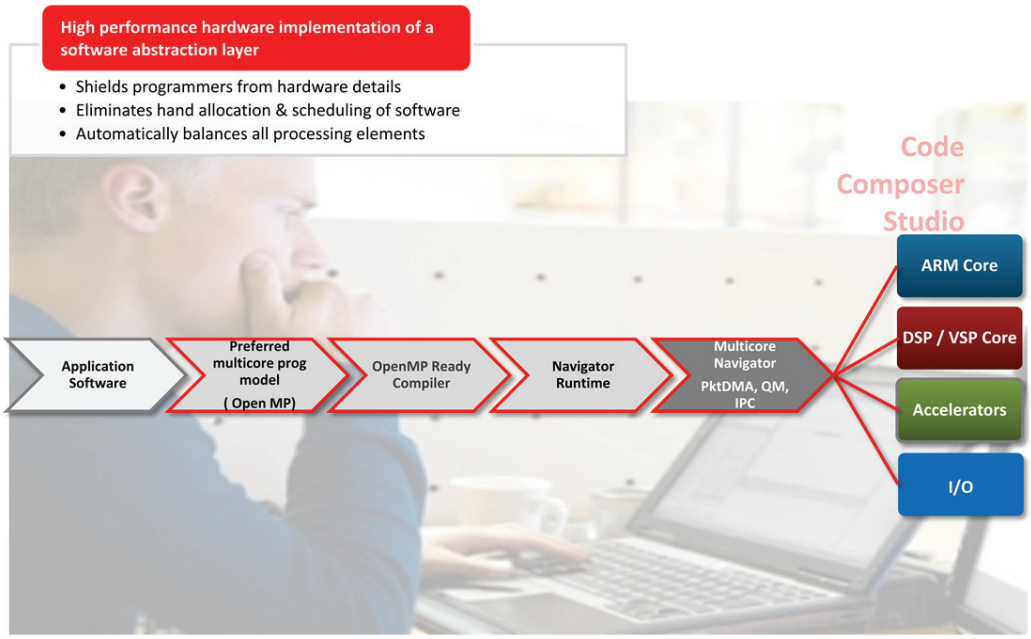

Figure 5. KeyStone AMP programming flow using Navigator Runtime

Conclusion

Multicore Navigator is an innovative and intelligent chip infrastructure for KeyStone devices that facilitates multicore programming. Multicore Navigator and Navigator Runtime provide an abstraction of each hardware component enabling platform virtualization with resource management, scheduling and dynamic load balancing to maximize multicore efficiency and achieve higher hardware utilization with lowered system cost. By combining it with high-level multicore programming paradigms, such as OpenMP, Navigator Runtime simplifies AMP multicore software design, increases multicore program scalability and flexibility, and enables rapid deployment of new technologies at reduced cost.

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**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Mobile Processors	www.ti.com/omap		
Wireless Connectivity	www.ti.com/wirelessconnectivity		
	TI E2E Community Home Page		e2e.ti.com

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