Reed Electronics Group

MICROPROCESSOR REPORT

www.MPRonline.com

THE INSIDER'S GUIDE TO MICROPROCESSOR HARDWARE

JUPITER'S TWIN CORES

Advanced Technologies Converge in a Powerful Multipurpose Chip

By Max Baron {11/17/03-02}

Jupiter is the largest planet in the solar system. Its mass is about twice that of the rest of the planetary bodies circling Sol. Jupiter's powerful gravity and internal pressure are believed to envelop a fiery core at a searing temperature of 20,000 to 30,000 degrees centigrade, a core which may not even be solid. The cores in Motorola's new Jupiter chip are reassuringly solid silicon, designed to spend minute quantities of power rather than fling it out into space as the chip's namesake does. But a closer look at the system design and the chip's expected performance may furnish some insight into the reason the company's engineering team decided to call it Jupiter.

On October 15, 2003, at In-Stat/MDR's **Microprocessor Forum** (MPF 2003), John Vaglica, manager of the Wireless System Division's Advanced Architecture Team, presented Motorola's new chip, Jupiter. With Jupiter, Motorola SPS is launching a new architecture, a new approach to SoC design, and a business strategy to match. Motorola SPS is calling its new brainchild the Mobile Extreme Convergence (MXC) Architecture. We'll refer to it as the MXC architecture to differentiate it from Motorola's line of cellphone chip sets known as Innovative Convergence Platforms. Although primarily aimed at air communications such as cellular, Wi-Fi, and Bluetooth, Motorola's approach may find additional markets in the consumer, automotive, and industrial segments.

Design Strategy

Jupiter's architecture represents the first use of two cores—an ARM1136JF-S, employed as applications processor, and a StarCore SC140e DSP engine entrusted with communications. The design strategy that has prompted the collaboration of two design teams, one in the U.S. and the other in Israel, so far apart in the world, involves more than just one chip aimed at 2.5G cellphone workloads, and also possibly at 3G. According to Motorola SPS, Jupiter will deliver sufficient performance and integration to reduce the number of chips required to build air communications for handheld devices, simplifying the software required for their operation. Motorola SPS views Jupiter as a "chassis" built around an architecture suitable for a wide range of performance requirements and applications. Jupiter's underlying architecture is targeting Motorola's 90nm low-power process, which will allow the cores to function at fairly high frequencies, up to 600MHz.

Motorola SPS expects Jupiter's cores to function at modest frequencies: 208MHz for the four MAC/cycle SC140e and 400MHz for the ARM1136JF-S core. Performance, however, cannot be obtained simply by high frequency of operation and by four MAC operations in parallel, especially when it must be delivered by two cores sharing one chip and one common port to memory. Motorola spent most of its engineering efforts in simulating and designing on-chip bus structures and memory hierarchies—to support high performance by reducing the contention between the two cores to on- and off-chip resources. System design, at last!

From the viewpoint of software, the MXC architecture combines the two cores, their interaction, and the system-resource model they offer to OS and application writers—all aimed at introducing a simpler, faster way for software designers to create, reuse, and move code to cellphone and wireless platforms based on the MXC architecture.

Among the claimed benefits, we note that the architecture is intended to offer cellular, Wi-Fi, and Bluetooth modem support on a single DSP core. Today's cellular modems require both a DSP and an MCU core for a complete protocol stack, whereas Wi-Fi and Bluetooth modems require independent ARM MCUs. The implementation will improve the efficiency in communication between baseband and application processors, since the protocol stack is implemented on a single processor. The two cores are designed to enable separation and independence of baseband and applications processes from one another. The SC140e's capability of delivering both DSP and microcontroller functions allows it to function as a complete baseband processor by itself, without involving an additional ARM core or the services of the resident ARM1136JF-S. With the two cores operating independently on one chip, it follows that separate operating systems can be supported: one controlling baseband and system resources and the other serving and running on the applications processor. The physical separation of modem and application code is also expected to provide improved security.

SC140e: DSP and Control in One

Owners of DSP and MCU architectures have been trying to blend ISAs into one another for quite a while. Semiconductor vendors have been adding instructions to their ISA—some to improve the DSP capability of a processor designed for general-purpose computing and control, and some to improve a DSP's chances of executing system software. But the execution of a multiply accumulate (MAC) does not a DSP make, nor do bit-wise and semaphore instructions make a processor that can run an operating system.

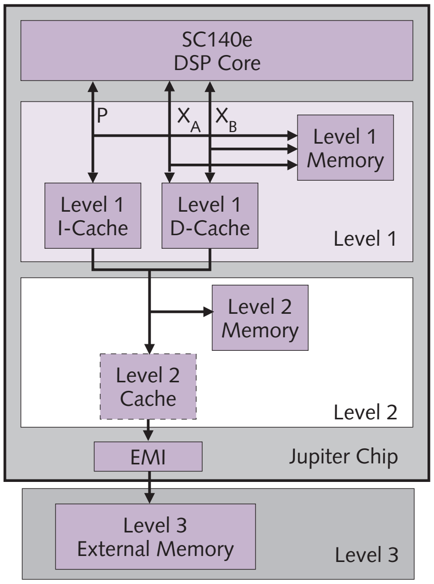

Figure 1. SC140e block diagram shows on-chip two-level cache and SRAM memory hierarchy. The EMI block is the external memory interface–arbitrated controller. The DSP engine can directly access its L1 cache and M1 SRAM. L2 memory and L2 cache are designed to reduce the frequency of external memory accesses that may occur upon a miss in either the L1 I-cache or D-cache The DSP in Jupiter uses a two-level on-chip memory hierarchy, combining at each level, caches and static RAM, and at L2 adding nonvolatile memory (NVM), which, most probably, will be implemented by Motorola SPS via on-chip ROM.

General-purpose processors (GPP), as they seem to be defined today, were intended to be very efficient at running instructions-intensive workloads: large code operating on relatively minute data. DSP engines differ by processing massive amounts of data and using small amounts of code. The key to providing not just an enhanced MCU or DSP instruction set but to making it deliver efficient performance is in the memory hierarchy isolating the high-frequency engine from the lower-speed external memory.

DSP and GPP workloads exhibit different temporal and spatial locality. They must be supported by different on-chip SRAM and cache structures that aim at making best use of the characteristics of the intended workloads. DSP and GPP workloads must be supported by appropriate addressing of data and instructions and must use memory management to protect processes from harming one another and/or from corrupting the system. These were the basic requirements facing the designers of the SC140e—a DSP that must also be able to perform system control functions.

The SC140e is tasked to function as a single-core modem, performing all the signaling protocol layers (L1, L2, and L3) for 2.5G, 2.75G, and 3G (W-CDMA/UMTS) standards. Figure 1 shows the SC140e's memory hierarchy configured to provide efficient processing for both DSP and MCU workloads. Following StarCore's basic architecture implementation, the SC140e employs four Data ALUs (DALU), each of which can deliver a one-cycle MAC, an ALU operation, or a bit-field-oriented operation (BFU). Its 16- × 40-bit data registers can be accessed for data representations of 8, 16, 32, and 40 bits. StarCore SC140's architecture defines two address-generation units (AGU). The address units can be employed to form addresses for incoming operands or for storing results in cache memory. With only two address units, operands must be laid out at contiguous addresses in memory. At its top frequency of 208MHz (a multiple of 13MHz used in GSM), the SC140e operating on 16-bit data must be fed 128 bits at a time, if both data and coefficients are to be read from the XA and XB sides of the data cache. The maximum required bandwidth is 3.32GB/sec. Returning 16-bit results would require only half the bandwidth, a maximum of approximately 1.66GB/sec.

The distribution of code and data for DSP and MCU workloads reflects the system's support for high-performance digital-signal processing and for system requirements to execute hard real-time deterministic functions. The L1's separate 16KB I- and 32KB D-cache can be accessed in one cycle and are designed to support high throughput by holding most-frequently-used code and data. The instruction cache and data caches are virtually addressed. Note that although StarCore's instructions and data can reside in the same memory space, the implementation of SC140e employs separate spaces for instructions and data, providing a better organization from the viewpoint of the DSP, which may have to run fairly short, repetitive code on streams of incoming data that should not be mixed with instructions.

The recently added MMU to the SC140e provides memory protection with support for virtual memory. An interesting addition to an MMU otherwise intended exclusively for system code is a user mode. The user mode suggests that Motorola envisions a business model where third parties may create portions of the operating system and/or resource drivers that will become part of the OS. It may also become a parachute for applications programmers needing to "steal performance" from the modem side of the chip—a practice Motorola worked hard to avoid by providing the ARM core with enough frequency and bus bandwidth to execute the most demanding general-purpose applications.

SC140e's L1 memory is virtual and can be accessed in one cycle and protected by the MMU. L1 memory stores critical code and data. It will be used most often for interrupt service routines and related data requiring deterministic memory latency to execute hard real-time processes. Most cache designs aimed at hard real-time operation provide mechanisms for programmers to ensure SRAM-like deterministic operation. Cache sections and/or lines can be locked to prevent them from being overwritten by fresh data or code. Motorola's DSP L1 cache-locking configuration is unique; Figure 2 shows how the SC140e's cache may be controlled but requires some explanation of its intended use.

The SC140e's L1 cache can be locked in 1KB increments (one way for a 16KB cache) at way boundaries to provide arbitrarily long locked sections for critical processes and their data. The interesting aspects of this cache are the method in which a locked cache portion is initialized and the way processes are able to use the cache to communicate with one another. Initializing a locked portion of the cache is accomplished by locking the whole cache, except the section about to be dedicated to a process, invalidating its entries, and running the process. By comparison, some designs define a section to be locked and, to initialize it, execute a speculative run of the process. Communication between processes is supported by allowing them to share one or more overlapping ways. One is reminded of register windows architectures like Sun's SPARC, which overlaps register sections for passing parameters. Quite a few additional details must be ironed out, such as the algorithm used to invalidate entries in the cache ways that become available and how ways allocated to a process are made available to it when it becomes active, possibly via tagging them with the process's context identifier. Interpreted as an algorithm that removes from the cache entries that have not been used recently, the L1 cache management of ways can be very useful in delivering complete, warm caches to a process that is returned to the active state.

It is unlikely that several processes will be permitted to lock the cache, preventing all other processes from taking advantage of the memory hierarchy. One can imagine that a system software directive could prohibit system programmers from locking the whole cache, leaving a portion of it always available to act as a smaller L1 mirror of memory for non-critical processes. (See PowerPC G2 cache locking *http://e-www.motorola.com/files/netcomm/doc/app_note/AN2129.pdf.*)

The L2 cache is a superset of L1 holding both data and instructions. It is designed to improve performance, reduce contention with the ARM1136JF-S core, and lower overall power consumption by mirroring external memory. The slower and less-energy-consuming L2 memory is expected to store modem Layers 1 and 2, the RTOS, and code for vocoders. (The SC140e is responsible for the microphone.)

Motorola's introduction of an L2 cache for handheld devices brings the design into the realm of high-performance system-oriented configurations seen before mostly in microprocessors powering systems such as desktops and servers. Motorola's Jupiter presentation at MPF and its recently released white paper seem to disagree on the existence of an L2 cache for the SC140e; however, both agree on the existence of the L2 SRAM and NVRAM. The apparent disagreement indicates that the MXC architecture is already in the process of being scaled across a range of performance levels. *MPR* thinks the confusion comes from the fact that Motorola has more than one product in development, some with the L2 cache and some without. Jupiter is a high-end product; it has been designed to incorporate an L2 cache intended to support its performance. At the lower performance levels, the L2 cache is optional and will not be implemented. Motorola's white paper describes a lower-tier product.

Motorola's simulations are averaging a hit rate of 95% for the L1 I-cache, 90% for the D-cache, and 70–80% for the L2 instruction and data caches. A purely arithmetic calculation,

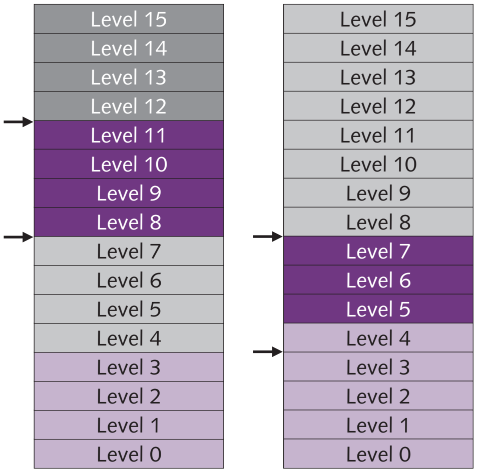

Figure 2. The DSP's L1 cache, a 16KB 16-way associative resource, can be partitioned for locking on one of its 16-way boundaries to form arbitrarily long "frozen" sections (4KB and 3KB, shown above). Cache management can also assign to processes an equal cache size (4KB for example).

assuming 90% data-cache hits at the L1 level and 75% hits at L2, would yield a maximum incoming operand bandwidth of 41.6MB/sec from external memory (L3 memory), an easy task for a 32-bit 133MHz memory that can deliver 532MB/sec and appropriate for flash memory functioning at 66MHz (266MB/sec). L3 memory will hold modem Layers 2 and 3, GPS, and other applications.

ARM1136JF-S: A Powerful Apps Processor

ARM Holdings PLC's first implementation of the ARMv6 architecture is the ARM1136JF-S, announced in spring 2002. The ARMv6 architecture, announced in 2001, features support for SIMD functions such as dual 16 × 16 MAC per cycle. The new features are claimed to boost performance by up to four times. Other key features incorporated in ARMv6 are improvements in memory management, data sharing, and synchronization for multiprocessing, bi-endian data handling, and exceptions and interrupts. ARMv6 supports all of ARM's previous enhancements and extensions, such as Thumb, DSP, Jazelle (ARM's Java bytecode interpreter-accelerator), and media instructions. Aside from general-purpose program execution, ARM's targeted applications for ARM1136JF-S are Java and DSP intensive, such as Internet-related workloads, streaming media, MPEG4 encode/decode, and voice and handwriting recognition.

The "-S" suffix in ARM1136JF-S stands for "synthesizable." Motorola's choice of this core may indicate the design methodology used in implementing Jupiter. With "over two years spent in design by over one hundred people" according to Motorola SPS, the chip may be waiting for the company's 90nm technology. The team's best use of the remaining time may be deeper verification and more simulations mixing applications and baseband workloads.

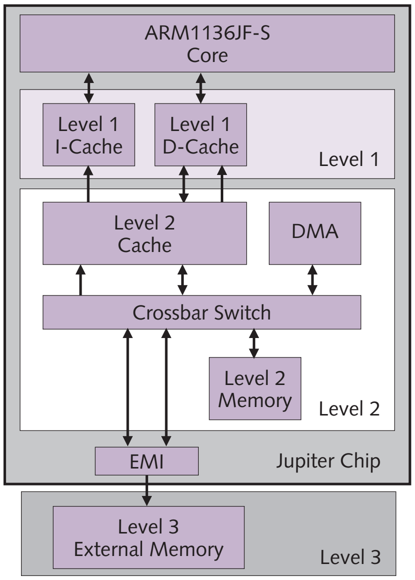

Figure 3. The memory hierarchy supporting the ARM11 core provides, in addition to L1 I- and D-caches, a hefty 128KB L2 cache and L2 SRAM to help isolate the ARM from its bandwidth-hungry modem powered by the SC140e.

The targeted 90nm process is puzzling: The original SC140 was designed to execute at 300MHz in 180nm processes, yet in Jupiter it is required to go up to only 208MHz. The eight-pipestage-deep ARM1136JF-S is specified by ARM to execute at 400MHz, using TSMC's 130nm process. Motorola may have in mind more-demanding applications than it is willing to make public at this time.

Figure 3 shows the ARM core's three-level memory hierarchy incorporating L1 I- and D-caches of 16KB each. The ARM 11 core can communicate with its L1 caches using 64-bit bus widths. The peripheral bus is 32 bits wide.

Motorola is not implementing ARM's tightly coupled I and D memories (TCM). At L2, the ARM core is supported by a 128KB cache common to instructions and data. The ARM's L3 support is the external memory itself, in an arrangement that copies the configuration supporting the DSP engine.

The System: 1.6GMAC/sec Plus Accelerators

A simplified block diagram of Jupiter appears in Figure 4. Compared with a system configuration using separate chips for applications processing and for each medium of air communication (Wi-Fi, Bluetooth, GPS, and cellular), the MXC architecture saves four processor cores, three RAMs, three ROMs, and associated logic. Its predominant attributes are thus price reduction, smaller size, lower power consumption, and the inherent flexibility of being scaled up or down in system performance to match the price-performance requirements of different applications.

Having seen the memory hierarchies and communications assigned to the ARM1136JF-S and the SC140e, the chip's block diagram is not surprising. The chip really looks like two separate engines integrated onto one die. Indeed, the only connections one can see between the modem engine, shown in the upper half of the block diagram, and the application processor (AP) are a block of registers used for interprocessor communications (IPC), shared peripherals, and the arbiter and memory interface responsible for communication with external resources. Two crossbar switches, one on each section, ensure that the processor core, DMA, accelerators, and other dedicated logic will not be in contention with each other, trying to access local memory. All on-chip buses are AHB except the specially designed buses connecting the crossbars to peripherals and security hardware.

Jupiter's external memory interface is designed to support connection to mobile SDR/DDR SDRAM, cellular RAM, NAND Flash, synchronous NOR Flash, and SRAM. Accesses to external memory are arbitrated in favor of the modem, the better to help it execute time-critical functions. The applications processor L1 cache and the generous L2 cache are expected by the design team to help the ARM core operate with little impact from the modem's bus activity. The chart in Figure 5 was generated by the design team, following our request to provide quantitative data. The chart provides an estimate of the applications processor (AP) average performance as a function of its L1 I-cache hit rate in the presence of continuous modem activity.

The workload executed on the application processor represents a mix of expected applications code. Results were generated by using execution traces and appropriate wait cycles for accesses to external memory. One notes that when the AP experiences 100% I-cache hits, the modem, AP L1 D-cache misses, and L2 misses are the only users of the external bus, requiring approximately 33% of its available bandwidth. Between 99% and 95% hit rates, the applications processor may lose as much as 30% of its performance but still deliver enough throughput for most of the workloads it is expected to execute. The L2 I and D 128KB cache contributes significantly to these numbers. Its removal will reduce performance to approximately 50% of its previous value with L2 cache active.

The ARM1136JF-S frequency of operation simulated was 400MHz (actually 399MHz) using a bus and memory clock ratio of 1:3 (133MHz). Note that the interface between the memory bus and the modem is asynchronous. The design team stated that it has taken the necessary steps to ensure the highest reliability of asynchronous communications.

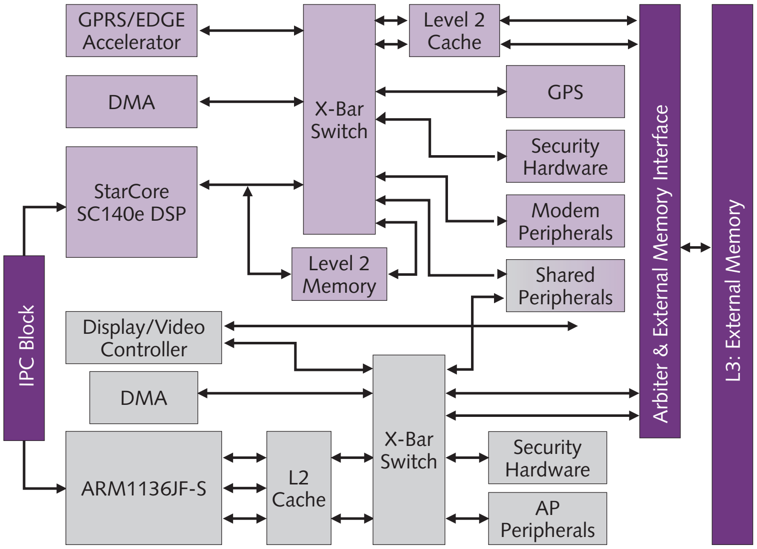

Figure 4. Jupiter's simplified block diagram shows similarities in memory hierarchy between the single-core modem (upper half) and the applications processor below it. Local efficiency is further enhanced by two crossbar switches.

Power Management to Lengthen Battery Life

One of the more obvious advantages that the MXC architecture can claim is its inherent reduction of power consumption, obtained by using fewer processing cores, keeping the communications on-chip between the remaining two, and providing on-chip memory mirroring to reduce the number of external accesses to a minimum. The design team at Motorola SPS has incorporated into Jupiter most of the usual well-known methods of saving energy, plus a few harder-to-implement power-management techniques involving control over voltage—an absolute must considering that the design targets 90nm silicon and must deal with increased leakage current.

Jupiter was implemented using power-optimized synthesis. The chip's designed-in power reduction employs dual-V_T to ensure that higher leakage is used only where it's necessary to deliver fast propagation at high frequency and low V_{DD}^2. The design incorporates clock gating to peripherals and also inside to the processing cores. The extent of clock-gated flip-flops has been estimated by Motorola to be 90%. Motorola has also provided 1.8V I/O levels to external memory, to further reduce the consumption of power.

Voltage-control power-management techniques were used to help reduce dynamic and static power consumption. The design team used dynamic frequency and voltage scaling, active well bias, and power-gated memory arrays with and without state retention.

Table 1 provides an estimate of voltages required by the cores for operation at various frequencies. Assuming that SC140e has been assigned to an individual power island, its requirement for 1.2V at a lower frequency (208MHz) is an indication of longer propagation delays associated with ALU

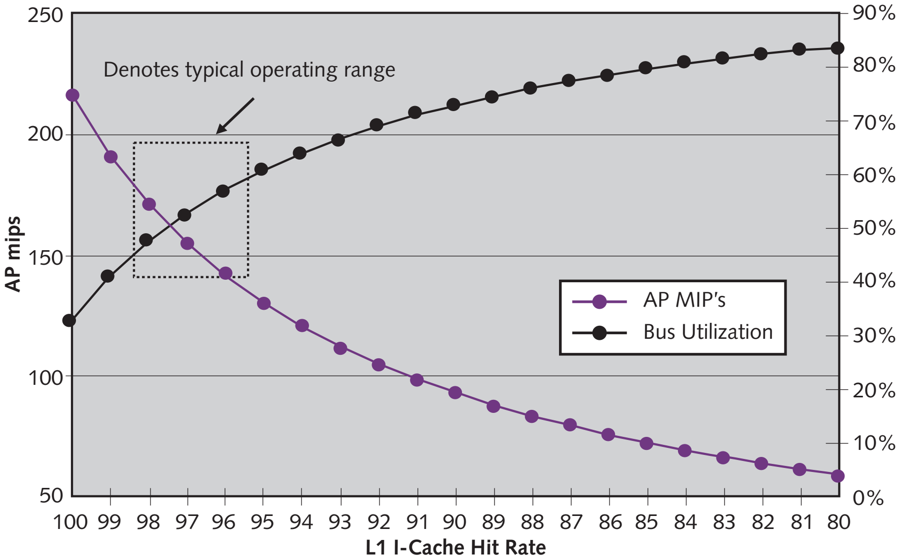

Figure 5. Applications processor performance as a function of L1-cache on-chip hit rate in the presence of continuous modem activity. Misses in both L1 and L2 caches require accesses to external memory.

Price & Availability

Motorola SPS expects Jupiter samples to be available during mid-2004. For additional information, please visit www.motorola.com/MXC.

operations and/or with data fetching for the 40-bit datapath architecture.

Note the relatively high voltages employed for ensuring operating frequencies for the ARM core. The need for these values probably results from a combination of using soft cores and a requirement to reduce 90nm leakage current via higher V_T.

The design team has recognized the value of minimizing the number of cycles required to execute recurrent workloads and has provided preliminary estimates for different call scenarios: the SC140e processing GSM Voice (EFR) will require 39MHz; GPRS class 8 will execute at 78MHz; and EDGE class 12 can be supported at 104MHz. (Note: the number of cycles required to execute a workload are an indication of energy consumption; in Jupiter's case, the number of total cycles is determined by the product of frequency and time of utilization.)

The Speed of Security

No modern architecture or chip configuration can expect to be accepted unless it has dedicated significant resources to maintaining the security of its end-user content and the security of data visited or employed by the end user. The MXC architecture, as embodied in Jupiter, has provided special secure regions and a configuration to enable separation of system and applications software plus dedicated hardware to speed up encrypt-decrypt algorithms.

Jupiter's modem memory and peripherals are physically separated from the application processor to ensure that the integrity of communications cannot be easily corrupted by applications software or by scripts an end user may unknowingly download while browsing the Internet.

Jupiter provides two secure on-chip boot ROMs performing code image validation. The ROMs are used to establish a point of trust for the application processor operating system before beginning the activity of applications or communications. At boot time, secure boot code running from on-chip ROM validates the code image. Each processor has its own secure boot image. Once validated, a secure operating mode is entered and monitored by hardware for potential security breaches (e.g., entering debug mode). Any breach immediately removes the system from the secure state.

Processor	Low	Medium	High
ARM1136	1.2V @ 399MHz	1.4V @ 500MHz	1.6V @ 532MHz
StarCore DSP	1.2V @ 208MHz	1.4V @ 225MHz	1.6V @ 250MHz

Table 1. V_{DD} required by the cores at three frequencies. Note StarCore requirement of 1.2V at 208MHz.

Code and data images are validated using public-key digital signatures based on the SHA-1 hash algorithm and RSA public-key cryptography. SHA-1 is an algorithm that creates a 160-bit signature for code and data objects. Data validation is performed by running the algorithm and comparing the newly generated digest with a separately stored original signature. Jupiter provides a dedicated hardware accelerator for hashing code and data images before they are trusted to be virus- or Trojan-free. The SHA-1 algorithm is used to support verification of authenticity (validation of identity of sending party), and integrity (validation that the message has not been altered or replaced).

The security-control module includes a secure RAM module and a security monitor. The secure RAM module contains a 168-bit secret key and 3DES encryption, used to prepare sensitive data, such as passwords and credit card numbers, for off-processor storage. The security monitor delivers system-monitor functions and security-support tasks.

Jupiter's boundary of the system memory shared by the two cores is protected by a hardware firewall, and a unique on-chip hardware ID number is provided to support Digital Rights Management (DRM).

A Test Drive, Soon

Motorola expects to ship first samples of Jupiter by mid-2004, the actual silicon probably being gated by the availability of the production 90nm process. Voltage/frequency graphs shown by Motorola at MPF 2003 go beyond 550MHz, promising a chip that can be scaled in performance to address applications requirements going beyond air communications.

Good software development tools for the applications processor are available from ARM and others. The SC140e is still at the beginning of its life cycle. Motorola encourages developers to use tools available for the SC140 until the development environment supporting all of the SC140e's enhancements is released. Motorola expects the new tools to be released in conjunction with the first product samples.

Motorola's available DSP software for the SC140 includes an RTOS optimized for StarCore and for the needs of real-time signal-processing environments. A C compiler with provisions for in-line assembly is also available. Motorola expects to offer a communications stack covering GSM, GPRS, EDGE, and 3G (W-CDMA).

The technical and business strategy that has created the MXC architecture and Jupiter, its first instantiation as announced by Motorola, may well be heralding an era when processor design must share the limelight with system design and for complex SoCs—even becoming less important than the system. ◇