MICROPROCESSOR REPORT

www.MPRonline.com

THE INSIDER'S GUIDE TO MICROPROCESSOR HARDWARE

The Single-chip Cloud Computer

Intel Networks 48 Pentiums on a Chip

By Max Baron {4/26/10-01}

Intel's presentation of the 80-Core Tera-Scale Research Processor at ISSCC 2007 should have prepared us for the Single-Chip Cloud Computer, but it didn't. Intel's TRP—the Tera-FLOPS Research Processor, AKA Tera-Scale Research Processor—was built by the company for use by hardware and software designers as a learning chip. The PE, its processing engine, was a relatively simple high-speed dual-SIMD floating-point CPU. It was easy to envision this CPU's integration along with the other 79 identical cores in the company's experimental chip, because multiple-core chips employing simple CPUs were already in the market.

But with no more than four, or soon—six, of the present complex cores implementing Intel's Architecture (IA) in today's PCs on one hand, and the simple CPUs employed in existing multiple-core chips on the other, it was difficult to imagine an on-chip integration of 48 dual integer superscalar floating point Pentium5s, or more precisely P54Cs—the processors that were almost the equivalent to 96 486 CPUs. Almost, since the P54C's two superscalar pipes are not identical: the V-pipe has limited functionality; only the U-pipe can execute all integer functions.

Considering the achievement in integration, it became even more interesting to understand the reasons why the SCC, the Single-chip Cloud Computer, should be introduced by Intel, in December 2009, as yet another experimental configuration.

The ISA is Given, But That's Not the Exercise

Before taking a detailed tour of the Single-chip Cloud Computer's internals it may be useful to speculate a bit at the "black box" level about Intel's and a handful of its selected developers' experience with the 80-core TRP, and the conclusions the company may have drawn from programming the previous experimental chip.

We know that the TRP was functional in 2007 because we've seen demos of applications that could take advantage of the TRP cores' simple ISA. (See *MPR 4/09/10-01*, "Low-Key Intel 80-Core Intro: The Tip Of The Iceberg.")

We know that the architecture of the 80 routers was adequate for the applications shown and that one of the ports implemented on each router allowed tuning the available on-chip storage by adding distributed memory that could be accessed from a stacked memory die.

We also know, or we think that we do, that the communication among the 80 tiles was to be defined by the application software employing vectors travelling a route that would connect an individual PE to the other program-relevant PEs and to local and off-chip memory.

Compared with the TRP, the Single-chip Cloud Computer, as we shall see from its description, indicates a shift of focus in several categories (see Table 1):

- If the simple floating point CPU employed in the 80-core TRP was intended to process scientific workloads instead of being just a test processor—the SCC, whose origins were already seen in FPGA test platforms in Germany, could represent an Intel focus shift toward general-purpose, networked, server-like applications.

  Intel's research leaders are pointing out, however, that TRP was always an experimental processor, and one of its aims was to examine the feasibility of putting intensive floating point on die within a realistic power budget. The TRP proved the feasibility of the workload and the power budget. The team was also interested in investigating a tile-based design approach in the context of intensive power management.

The SCC, according to Intel's architects, represents a different focus within the company's broad future multicore research agenda. They want the SCC to be a software development vehicle (SDV) supporting message-passing and software power-management features.

Intel has not abandoned the type of workload exemplified by the 80-core TRP. Lessons learned from both TRP and SCC may surface in future chips.

- As most multiple-core architectures have already shown, the 80-core TRP was difficult to program, with preset communication routes among processors encountering excessive latency due to packet contention on buses and routers. The TRP was capable only of executing very limited kernels due to ISA, I/O, and memory limitations. For example, programs had to be entered slowly over JTAG.

  The software experimentation turned out to be so interesting that the SCC was configured as a fully programmable processor capable of supporting software experimentation with a full-stack of OS, runtimes, and applications. The SCC relaxed the software constraints on communicating processes and on the predictability of routes.
- The SCC may execute workloads slightly slower than it could were it to employ a TRP-like implementation supporting 48 Pentiums, but its gains in efficiency and scalability will more than offset the reduction in speed.
- The most important survivor that has migrated from the 80-core TRP to the SCC is the router. It was improved and so were the buses / links it controls.

With router improvements, new link definitions, and the software to best take advantage of the 48-core chip, Intel's real exercise is not about a new ISA or about its own IA's behavior in a many-core chip.

Intel is not designing a processor core. It's designing a network on a chip; it's investigating power management and, it's researching message passing.

Pentium5: The Near-Forgotten Core

According to a presentation made by Intel this February at its SCC developer symposium, the core employed in the SCC is the Pentium P54C. A quick search of Intel's website yielded zero overview information about the well hidden, or forgotten processor, but some brief information was found on a few of the websites that provide a history of processors. Manuals, mainly software, can be found at *www.intel.com/info/scc*.

The first Pentium5 (see Figure 1) was launched by Intel in 1993 following a decision to switch to names of chips that, contrary to numbers such as 486—can be trademarked. Pentium5 was a 5-pipe-stage two-wide integer superscalar implementation of the IA minus the MMX extensions, but it included floating point instructions. Its five pipe stages were: Instruction fetch (IF), Decode 1 (D1), Decode 2 (D2), Execute (EX), and Write Back into registers (WB).

Compared with the 486, the Pentium5 had a wider 64-bit data bus supporting memory accesses, but internally it continued to employ 32-bit data. Its execution was supported by a split L1 two-way associative cache providing 8KB for data and 8KB for instructions, it was equipped with dual instructions and data TLBs, and could use an off-chip L2 unified cache sized between 256KB and 512KB. Its CPU clock frequency, in March 1993, was 60MHz. The chip was implemented in 0.8 micron semiconductor technology, it used 3.1 million transistors occupying a die size of 295 mm^2 and it required a V_{DD} of 5V.

Pentium5 is simple, lacks out-of-order execution—a plus—it implements branch prediction and built-in power management and, as P54C, it has the necessary hooks to execute in a dual SMP configuration—all good features that must have made it attractive to the architects of Larrabee who, for the graphics project, extended the P54C's internal instructions and data widths to 64 bits. The dual SMP hooks, however, were not used by the architects of the SCC.

Pentium P54C was introduced 12 months later, in March 1994, at CPU clock frequencies of 90MHz and 100MHz. It was implemented in Intel's 0.6 micron process; it numbered between 3.2 and 3.3 million transistors. The transistor number increase vs. Pentium5 was dedicated to additional clock control, an on-chip Advanced Programmable Interrupt Controller (APIC) and a dual-processor interface. The P54C occupied a die size of 147 mm^2; it was powered by a 3.3V V_{DD}, but, driven by external memory technology, its I/O bandwidth remained at approximately 460MB/s.

2006: Teraflops Research Processor	2009: Single-chip Cloud Computer
Many simple FP cores	Many fully-functional IA cores
Validated tiled-design concept	Prototypes a tiled-design microprocessor
Tested HW limits of a mesh network	Improved mesh with 3x performance/watt
Sleep capabilities at core and circuit level	Dynamic voltage & frequency scaling
Light weight message passing	Message passing & controlled memory sharing
Limited programmability for basic benchmarks	Full programmability for application research
Primarily a circuit experiment	Circuit & software research vehicle

Table 1. Intel's comparison of research directions for TRP vs. SCC

Configuring the Experimental SCC

The Single-chip Cloud Computer continues Intel's, and the whole Industry's, trend toward employing higher core counts to replace the increases in clock frequency that have resulted in high power requirements, difficult to control high temperature, associated lower reliability, and chip design difficulties. At *MPR*, we estimated the work on the SCC to have taken at least three years, but as it turns out it took only two and a half years for a small group of 50–60 people from Intel, Hillsboro, OR; Intel, Bangalore, India; Intel, Braunschweig, Germany; Intel, Santa Clara, CA; Intel, DuPont, WA.

As we look at and appreciate the work that has been done, we remind ourselves that today's use of high core counts has its own limits set by similar parameters, such as dynamic and leakage power consumption, software development tools, and, again, at present, the nature of the specific workloads that can profit from multiple core architectures. The spectrum of single workloads that can be parallelized on multiple core architectures extends from a granularity that numbers just a handful of instructions—to threads that can be as long as tens of thousands of instructions and beyond. In workloads of this kind, some of which may be associated with one type of cloud computing, the size of code that can be parallelized depends on the overhead imposed by the executing architecture and software. In datacenters, many hundreds to thousands of cores are applied to tasks using a different programming model. Consider for instance, a large number of cores that cooperate in a search whose results are ready when the last cooperating core has completed its task. Intel's researchers are interested in finding out if they can bring support for the datacenter model on-die for better efficiency and if they can bring that programming model to the workstation and client with better hardware support.

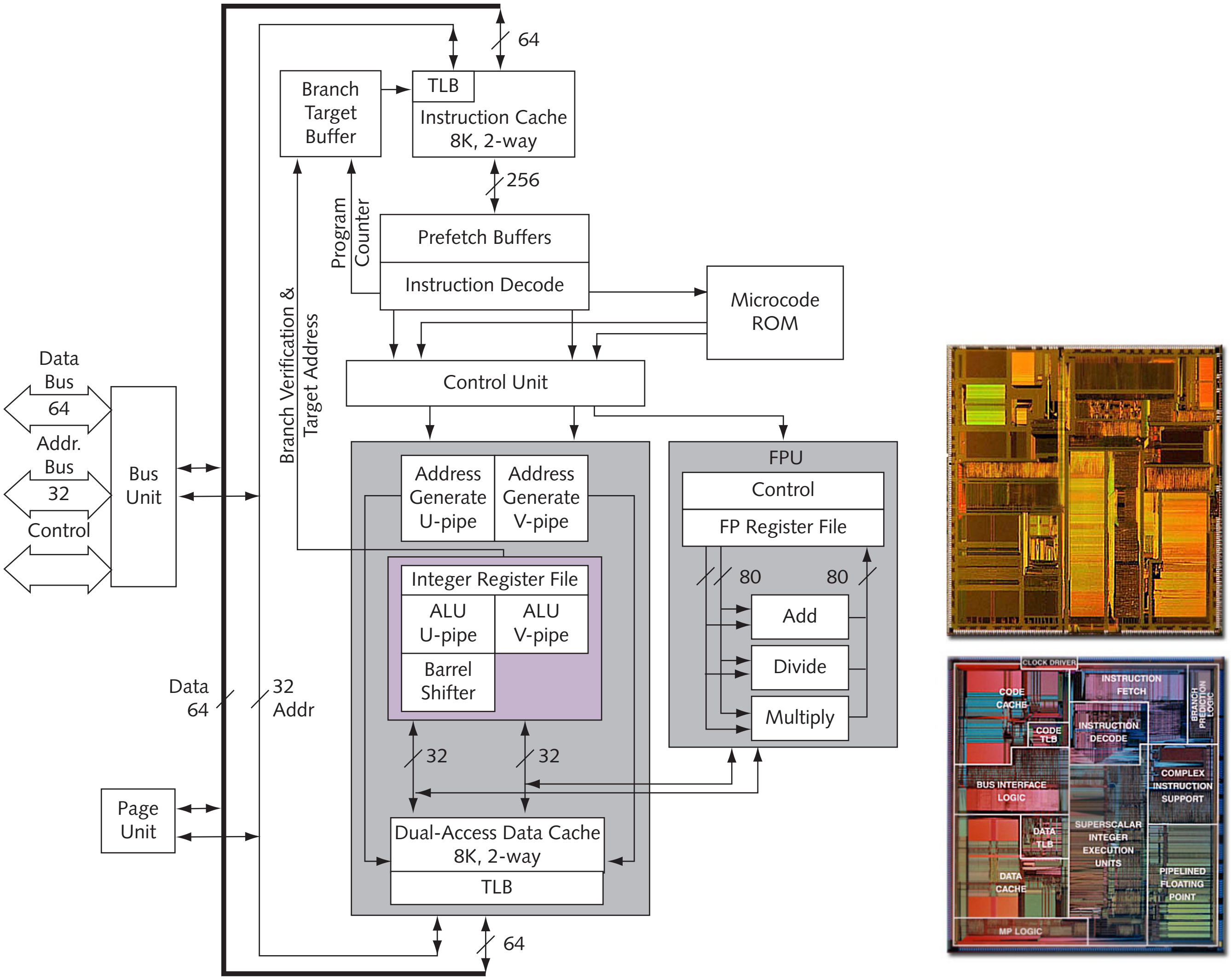

Figure 1. The block diagram of the very first Pentium5 shows its simple microarchitecture, its separate L1 data and instructions caches each providing a modest 8KB and its two integer pipes, U and V, of which only the U-pipe was designed to execute the complete integer functions included in the IA ISA. The photomicrographs at the right show the Pentium5 (top) minus clock control, MP logic, and APIC, and the P54C with functional blocks identified.

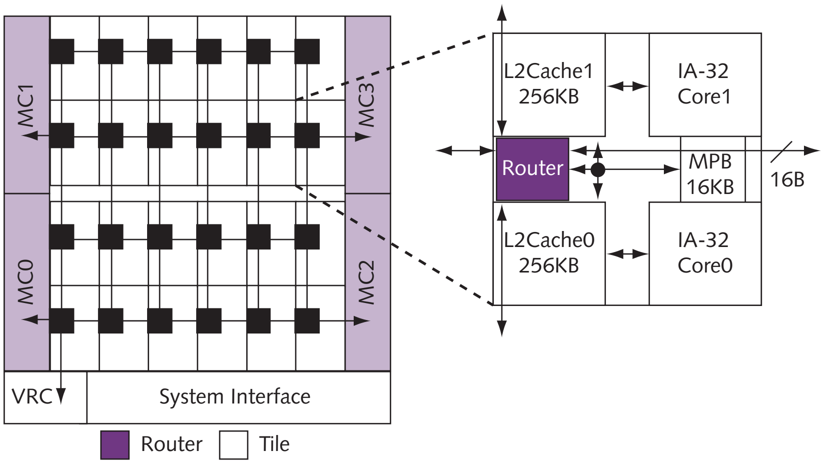

Figure 2. Block diagram shows that Intel's Single-chip Cloud Computer configuration is almost identical in principle to the 80-core TRP—a Manhattan-style network, tiles and packet communications, but there have been many improvements based on lessons learned, most of which were introduced in the routers and ports. The implementations of P54C as the processing core will attract many more software experts to work with the experimental SCC.

At first glance, it looks like Intel took the easy way out by using a well-tested processor configuration that it has employed in many of its chips (see Figure 2) and replacing the older arbitrated buses that are no longer efficient in multiple-core architecture with routers, crossbar links, and communication via packets. But, for a designer thinking about Intel's development, the adoption of older cores and existing configuration makes sense, since one must focus on the real difficulties that arise in controlling communication among cores, reducing power, implementing the chip design, and last, but not least—programming.

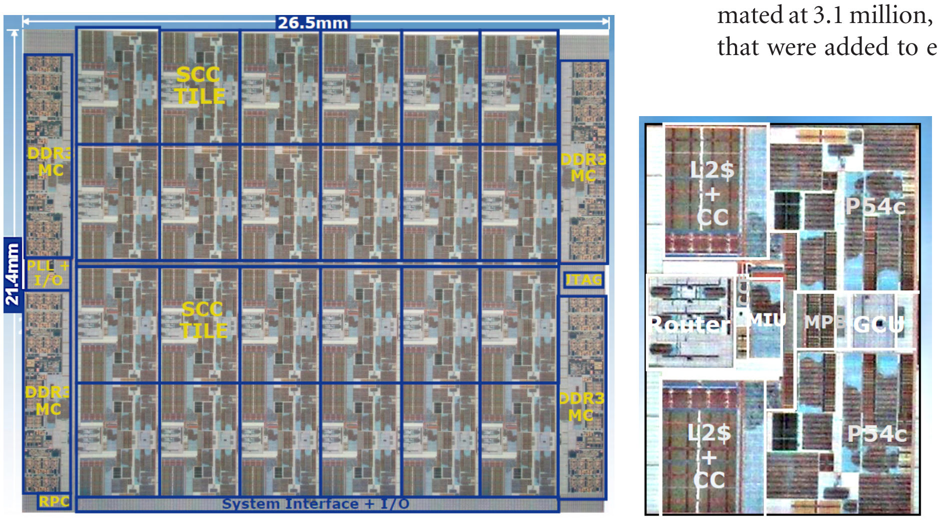

Figure 3. The photomicrograph of the Single-chip Cloud Computer shows its two-cores per tile area surrounded by four memory controllers, system interface, PLL, and JTAG access. Right: tile detail shows two P54C cores plus individual L2 caches. The router is the tile's means of communication with other tiles and external memory. The MPB (Message-Passing Buffer) is a new introduction supporting cache coherency among processors via software.

As can be seen in Figure 2, the basic configuration of tiles employed in the Single-chip Cloud Computer is similar to the one implemented by the company in its older 80-core TRP experimental chip. Tiles in the new SCC incorporate two cores instead of the previous single core implemented in the 80-core TRP, but continue to include the router and its crossbar links to other routers, external memory controllers, and system interface. The SCC incorporates 24 tiles laid out in two mirrored groups of 12. Each tile includes two P54C cores bringing the total number of cores up to 48. Each of the two core-processors is separately supported by its own one-cycle-access L1 cache of 32KB, a change introduced into the P54C that we must assume is implemented as an equally split 16KB instruction and 16KB data cache—a much better L1 cache support than the 80-core TRP provided its cores: 2KB for data and 3KB for instructions but, we should remember that at the time, the L1 allocation was intended to support different workloads.

Each of the P54C cores is also supported by 256KB of unified 8-core-cycle-access L2 cache, effectively duplicating the conditions under which most of the original Pentium5s chips were implemented, but replacing their direct access to external L3 and/or off-chip memory, by the mesh of routers providing access to the L3 and/or off-chip memory resources, in addition to cooperating processor cores.

The photomicrographs in Figure 3 can be used to extract additional detail about the configuration of the tile. The 48 million transistor tile compared with 6.2 million required to implement two original P54Cs sans L2 caches, each estimated at 3.1 million, shows the amount of memory and logic that were added to each tile—the most significant block of which is contributed by the individual L2 cache provided to each core. At half a megabyte per tile, the 24 L2 caches account for 12MB of high-speed, on-chip SRAM. With P54C cores and L2 caches accounted for, we look next at the logic and memory that characterize the new design. Placed between the two L2 caches, and to be separately described, the router is estimated to occupy approximately 2.7mm^2—about the same order of magnitude as one of the cores that require a silicon real-estate of 3.9mm^2.

Other logic blocks seen in the photomicrograph of the 18.7mm^2 tile are the MIU—the mesh interface unit, the Uncore logic that facilitates interaction among units and the GCU, one of the key units supporting control of power consumption by matching voltages and frequencies between a tile and its router. Generally specified to run at 1.0GHz, the two cores and Uncore units can be easily matched to the router and its links that can be clocked at up to 2.0GHz to gain bandwidth—but the matching of frequencies and voltages becomes very complex when, to reduce power consumption, tile frequency and voltage are tuned for most efficient operation to the point where they may be placed in deep sleep or be completely turned off.

With voltage matching intuitively easier to design, frequency matching can be more difficult to implement and may generate more communication errors that need to be corrected. Integer frequency ratios are simpler to implement, but, if needed, the company could be using mesochronous circuits to match arbitrary frequencies—the same type of circuits that were used in the 80-core TRP. According to the chip's architects, mesochronous circuits were determined to be overdesign and were omitted for energy savings. The implementation uses a synchronous design with 18B-wide, clock-crossing FIFOs to match between domains running at multiples of a common clock—a simpler solution but—as all are—possibly subject to metastable events. To minimize the possibility of a metastable event, the clock-crossing FIFOs insure that a read and a write never occur simultaneously at the same FIFO entry.

Last on the die, and probably one of the most explicit supports of coherency—we note the 16KB message-passing buffer that indicates the designers' intention that cache coherency employing SMP (symmetric multiprocessing) configurations will be dropped in favor of the more economical, more scalable approach that uses coherency via software. The 16KB message-passing buffer can be split evenly between the two cores, this being the default defined for Intel's Linux cluster configuration.

Continuing the brief comparison with the 80-core TRP, we note that Intel's experimental SCC chip is implemented in 45nm Hi-K gate technology with interconnect based on 1 poly and 9 metal (Cu) while the three years older 80-core TRP was built employing Intel's 65nm semiconductor technology with interconnect based on 1 poly and 8 metal (Cu) . The SCC required the use of 1.3B transistors instead of the 100 million used to make the 80-core TRP sans its intended stacked SDRAM die.

The difference in die area is, at first glance, less indicative of the degree of integration than the number of transistors. Compare the SCC's Itanium-sized 567.1mm^2 die area that's just about twice the area of the 80-core TRP's die size of 275mm^2, with the 13x ratio in transistor numbers between the two chips.

There are several reasons that, together, can explain the difference in die sizes: Moore's Law will only account for a 2x factor between the 65nm CMOS technology node associated by ITRS with the year 2007 vs. the 45nm technology node predicted for 2010. The remaining 3.4x ratio must be due mainly to the predominance of the regular structure of large, on-chip memory and the optimized P54C layout generated for the Pentium5 production chips that were sold into PCs and other platforms.

The Router

The SCC's 2D mesh is controlled by routers each employing five sets of 16B-wide data links plus 2B sidebands. The sidebands are generated by the routers and are used for functions such as error detection and routing. One of the five sets of data links is dedicated to the tile that's incorporating the router. The other four communicate to routers situated at the NSEW compass directions.

The mesh frequency of operation is the same as the router's—designed to run at up to 2GHz. The chip bisection bandwidth is 2.0 Tb/s (Terabits per second) estimated at the narrow bisection cutting through four routers.

The router is configured to enable communications via virtual channels to eliminate deadlock. Of the eight virtual channels supported, two are allocated to support two message classes. The request and response message classes were allocated two of the virtual channels, one each for request and response to avoid deadlocks. The remaining six virtual channels were allocated as a resource pool for traffic of packets of code and data between core processors on one hand, and on-chip and off-chip memory on the other.

Packets can switch speculatively their virtual channel allocation as they proceed from hop to hop, like cars in a city changing portions of their route depending on traffic conditions. Packets can switch virtual channel but do not change the path from router to router that is controlled by fixed X, Y routing. Compared with vector routing, the X, Y routing implemented in the SCC defines movement along one axis, and then along the next until the destination core is reached.

An ad-hoc selection of the next hop, or adaptive routing, that differs significantly from the preset routes defined by the programs running on some multiple-core processors has probably been left for future implementation. An adaptive SCC router's definition of the next hop may yield a less predictable latency of program execution, but, in exchange, it could offer scalability of cores and transportability of code. According to Intel's architects, the hop is pre-computed in a router to define the outgoing port to be used in the next router. The pre-computation helps speed up the arbitration for the next router's crossbar. Following arbitration, the crossbar switch allocation takes only one cycle.

As can be seen from Figure 4-a, which details the router functions performed in its four pipe-stages, during pipe-stage one, packets can enter into one of the router's five 16B-wide 24-deep FIFO queues from which the input arbiter selects the packets, whose route during pipe-stage two, will be pre-computed and switch-arbitrated. Switch arbitration determines which packets will be sent on to the crossbar switch and which of its ports will be allocated for

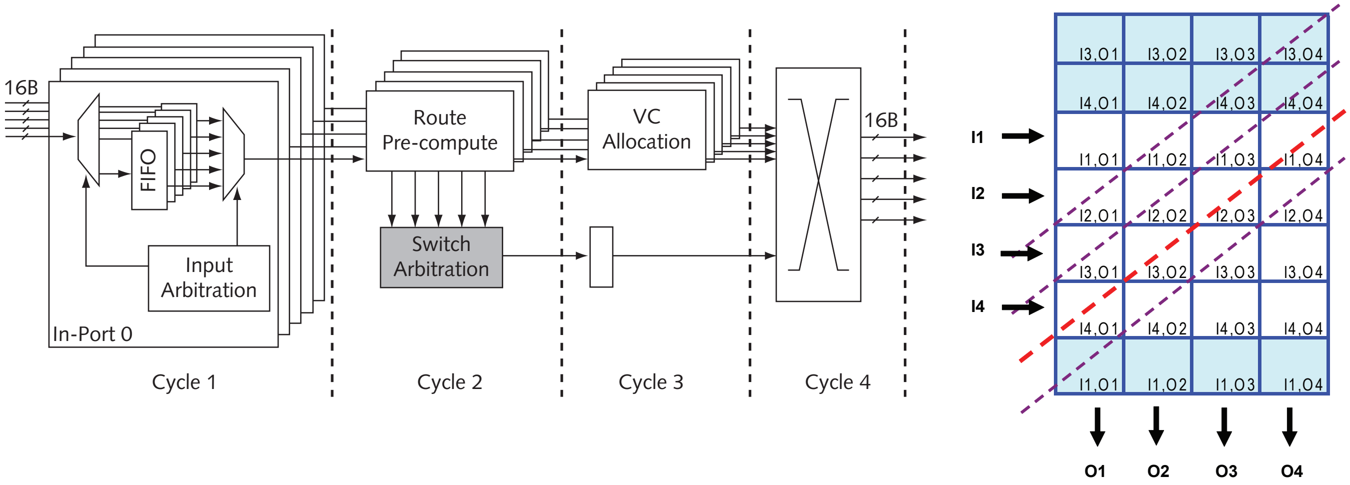

Figure 4. The router block diagram (a) shows the distribution of functions across four clock cycles: input to FIFOs and input arbitration, Precompute route and WWFA (Wrapped Wave Front Arbitration) for crossbar access, virtual channel allocation, and crossbar function. The WWF arbitration principle (b) can be illustrated as a preference for assigning high priority to wrapped matrix diagonals (see text).

each packet. The switch arbitration, together with the virtual channel allocation in stage three, determines the control of the crossbar switch during the fourth stage that also includes in the same cycle, the time required to traverse the link to the next router.

The switch arbitration algorithm selected by the SCC designers is the Wrapped Wave Front Arbitration (WWFA) depicted by *MPR* for simplicity as a 4 x 4 matrix in Figure 4-b.

The WWFA is based on the simple rule that the router is fully utilized when for each clock cycle, each of the input FIFOs is active in sending / receiving and when, during the same clock cycle each of the crossbar's links was assigned data to receive or send onwards. Since no two different FIFOs can be allowed to require the same crossbar link at the same time, a wave front describing the allocation of input FIFOs to crossbar ports can be described by one of the matrix's wrapped diagonals. The matrix diagonal receiving the highest priority can be changed each cycle. In Figure 4-b, the diagonal going through Input 4–Output 1 (I4,O1) has been given priority over the others.

Message-Passing: More Pluses than Minuses

Message-passing is a software means to provide cache coherency among cooperating cores—without the support offered in SMP configurations by bus snooping. Message-passing is the basis for the cluster programming model that Intel's architects are implementing on-chip; software-managed coherence is used in the distributed shared memory sometimes used in such programming. The approach is an alternative to the shared-memory programming model, where information is explicitly communicated between processes rather than through coordinated writes to a shared memory. Note that the same approach can also be used to implement software-managed cache coherence for regions of shared memory.

The principal benefits offered by message-passing are:

1. Economy of silicon real-estate and power consumption since snooping logic, and sometimes special buses, are needed for bus snooping; the snooping approach makes more sense for relatively small numbers of processors incorporated in a SMP configuration.
2. Scalability helping to increase the number of cores used.
3. Executable code can use cores that have no snooping support and—at cost in latency—even processors that are not located on the same chip.

Whether one uses message-passing, snooping, or shared memory coherence, one must use a test-and-set operation or, for example, an interrupt, to insure the integrity of the data to be delivered from the processor sourcing it to the destination processor.

Compared with the "automatic" hardware sending or reading shared data, a minus of the message-passing protocol requires the programmer to perform explicit cache operations in addition to the test-and-set sequence required for data integrity.

The following description of resource allocation presents the default configuration chosen in Intel's reference implementation of a Linux cluster. Note that the configuration is strictly a boot-time configuration choice and memory up to an addressable 4GB per core (the P54C limit) can be partitioned and allocated in different ways.

In the default configuration, the SCC is not implementing a single memory that's shared among all processor cores. Cores have private memory allocations. From the viewpoint of the individual P54C core, the synchronization of its cache with the private memory space allocated to the core continues to be supported via hardware cache write through. The SCC fully supports non-cacheable memory accesses. All non-cacheable memory accesses are handled by SCC as they would be by any other processor.

However, the cache coherency, respective to the memory space shared with other cores, has been placed under software control. Figure 5 describes the memory hierarchy and the memory space defined by a core-individual lookup table (LUT), assigned to each of the 48 cores of the SCC, but accessible for read-write by all cores.

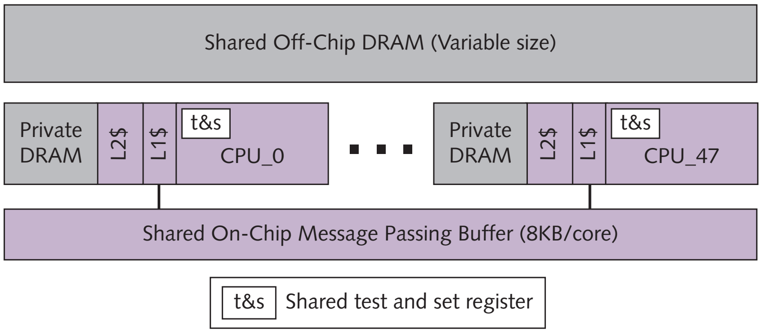
Shared Off-Chip DRAM (Variable size)
Private DRAM
L2$
L1$
t&s
CPU_0
Private DRAM
L2$
L1$
t&s
CPU_47
Shared On-Chip Message Passing Buffer (8KB/core)
t&s
Shared test and set register

Figure 5. The memory hierarchy, as seen by cores, includes, in addition to the usual L1 and L2 caches, private DRAM, access to shared off-chip DRAM, and a shared on-chip message-passing buffer supporting software cache coherency among cores. Shared areas are protected by test-and-set registers.

The total memory space assigned by the default to a core is 2.0GB. It includes boot sequences, Voltage Regulator Control (VRC) state support, the configurable shared on-chip message-passing buffer allocating by default 8KB per core, a shared off-chip DRAM that can be used to hold shared data that requires more than the 8KB provided on-chip, and an individual 1.0GB DRAM.

A few caveats concerning the control of LUTs: the LUT of a processor core can be dynamically programmed by any of the cores. It can also be transferred to another core to increase its available resources and there are no hardware obstacles preventing two cores from using the same memory space as their individual memory. To obtain the highest flexibility, the configuration of the memory hierarchy at boot time and—if needed—the re-configuration of its resources at runtime, are left to the programmer. Like in weakly-typed programming languages, the flexibility so important to research requires careful definitions.

The default-set 8KB message-passing buffer plays an important role in Intel's support of synchronization via software. It can best be viewed as an on-chip, high-speed SRAM with which processor cores can synchronize specific areas in their caches. A possible example of how the message-passing process may be working begins with a core writing data to the area of its cache that is mapped to the message-passing buffer, then causing it to be written—under test-and-set protection to the message-passing buffer. The receiving core must periodically invalidate its own cache area that's mapped to the same place in the message buffer, until it finds, as permitted by the test-and-set protocol, the data that was sent to it.

Intel's SCC architects have compared software-managed coherence vs. hardware coherence on a 32-way SMP Xeon executing the Black-Scholes and Art benchmarks and have reached the conclusion that the software-managed coherency can, for some workloads, be as efficient as hardware-supported cache coherency (see Figure 6). The lengths of the threads used to run the benchmarks will, however, have an impact on the comparison. The execution of one floating point iteration of the Black-Scholes differential equation may take just a few tens or hundreds of cycles, depending on the processor executing it. Short threads will make more visible the differences between hardware and software cache coherency than longer threads that concatenate several iterations. Compare, for instance, the performance differences between 8 threads and 32 threads that, for the same task, can use shorter threads.

We also note that the way communications among processor cores are implemented at present—irrespective of their being based on hardware or software, checking for data sent by a cooperating processor core and not finding it ready, will introduce more delays in single-thread cores than in cores equipped with support for multiple threads. In using the SCC to research software-managed coherency on

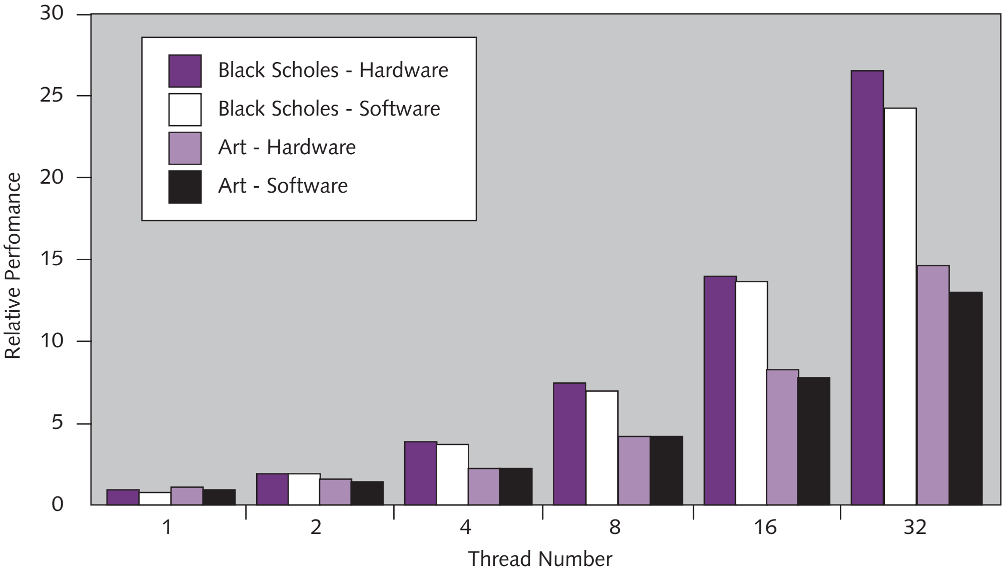

Figure 6. Black Scholes and Art benchmarks show just a small difference between hardware cache coherency among cores and software coherency implemented via message passing. The differences become more pronounced with the shorter thread lengths shown when executing 32 threads.

cores that, unlike the P54C, can support multiple threads, some of the P54C results will have to be interpreted by simulation or instrumentation.

The SCC's Dynamic Power Management

The SCC's implementation has obviously included multiple ways in which power consumption can be reduced, just a few of which are the use of the appropriate silicon technology node, libraries, and implementation of circuits. Most of the details that affect this type of static power control have not been made public. Dynamic power management, however, has been defined in more detail because a large part of the control has been left to the software running the system.

Figure 7-a provides an overview of the clock gating that can help reduce power consumption. The basic blocks of logic shown to be under clock gating control are each of

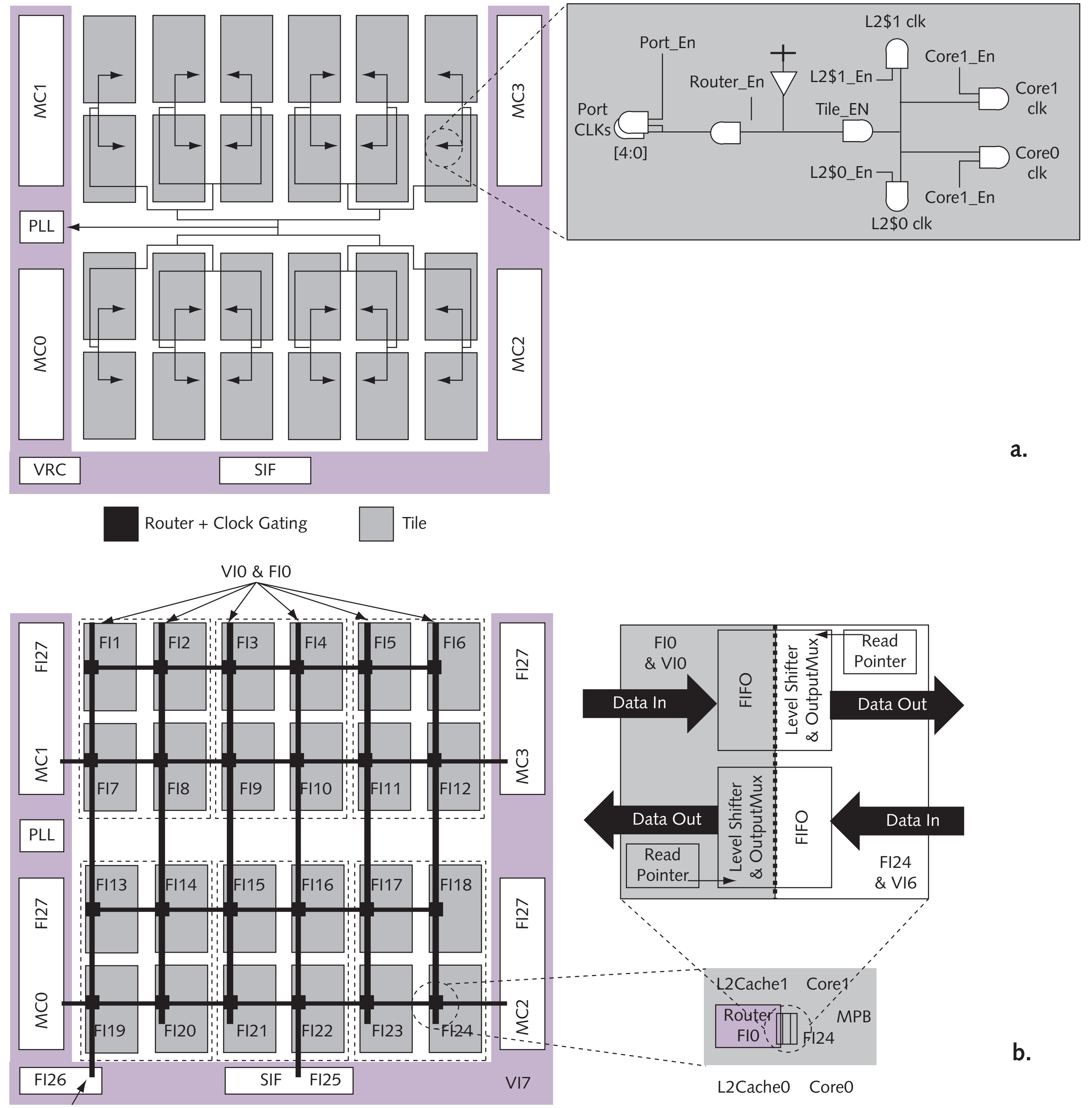

Figure 7. "Macro" clock gating (a) shows the main blocks whose clock enables can be turned on and off. Tiles are shown as rectangles. Routers and links are shown as dark lines. Note that clock gating can be applied to a tile's port but not to its router. The SCC can further manage its power by employing voltage/frequency scaling (b). The Single-chip Cloud Computer implements 28 frequency islands and 8 voltage islands (see text).

the P54C cores, the respective L2 caches, the tile-associated router and the individual crossbar port clocks. At MPR we believe that cycle-dependent "automatic" clock gating has been introduced at the detailed circuit levels, but it was not necessary to make its extent public. The type of clock gating that was disclosed about the SCC is the macro control that must be known and is normally left to system software because it can turn on and off individual cores, tiles, and the router port that connects to inactive tiles.

The router itself has not been designed to be clock-gated as a block, but any power reduction, such as obtained by gating the port connected to the tile—can help decrease the 500mW required by the router powered by 1.1V to run at 2.0GHz. At this power dissipation, the router and ports will account for 12W at 50ºC. Note also that the architecture of the tile employing two separate P54C cores with their own L2 caches was designed to save routers and ports in order to save power and reduce complexity. The SCC's mesh implementation provides far more bandwidth than a single P54C core can exploit. With more modern cores this balance would be revisited. Aside from sharing and competing for one router, the two P54C cores will behave as if they were on separate individual tiles.

Figure 7-b shows the distribution of frequency and voltage across tiles. The SCC designers have implemented 28 frequency islands, marked in the figure as "FIx" and 8 voltage islands, identified in the block diagram as "VIx." Each frequency island can be clocked at one of 16 frequencies obtained by dividing 4GHz by 16. The main clock can be set at 4GHz or lower at boot time. Frequency islands FI1 to FI24 have been assigned to the 24 tiles. FI0 controls mesh frequency. FI27 is a fixed frequency island since it's controlled by the DDR3 specification. FI26 has been assigned to the voltage regulator, and a lower 50–500MHz frequency FI25 can be delivered to the chip's system interface that could be implemented by an external FPGA. The router frequency cannot be controlled dynamically, but it can be set at boot time.

The eight voltage islands have also been determined based on the nature of the islands requiring them. Six four-tile islands containing a total of 8 cores receive voltages VI1 to VI6 that can be tuned between 0.0V and 1.3V in steps of 6.25mV. The rather minute size of the voltage increment is important since power consumption increases with the square of the voltage. The best match between the frequency required by the workload and the voltage needed to enable it will require a minimum of power consumption. Similar 6.25mV increments are available to the voltage controlling the mesh and system interface, but the voltage delivered to the memory controllers is, again, determined by the DDR3 specification.

The SCC's design proved to be on-target, clocking core and local SRAM at 1.0GHz with router and ports at 2.0GHz. The relationship between voltage and frequency is almost linear for voltages between 0.95V and 1.35V (see Figure 8-a), but, as usual for these portions of the curve, it becomes more abrupt for the lower voltage range. A comparison of frequencies between router and core shows that the frequency of the core pipe stages is falling faster than the router.

In the unlikely case that the SCC team redesigned the P54C to use fewer pipe stages, the team may have packed more FO4 equivalents per stage than the router needed at double the frequency.

The most likely assumption, however, is that the old design was used and scaled almost as is and its voltage-frequency ratio was the best achievable. As it turns out, the design, aside from minor changes (a new instruction), was just ported to the SCC.

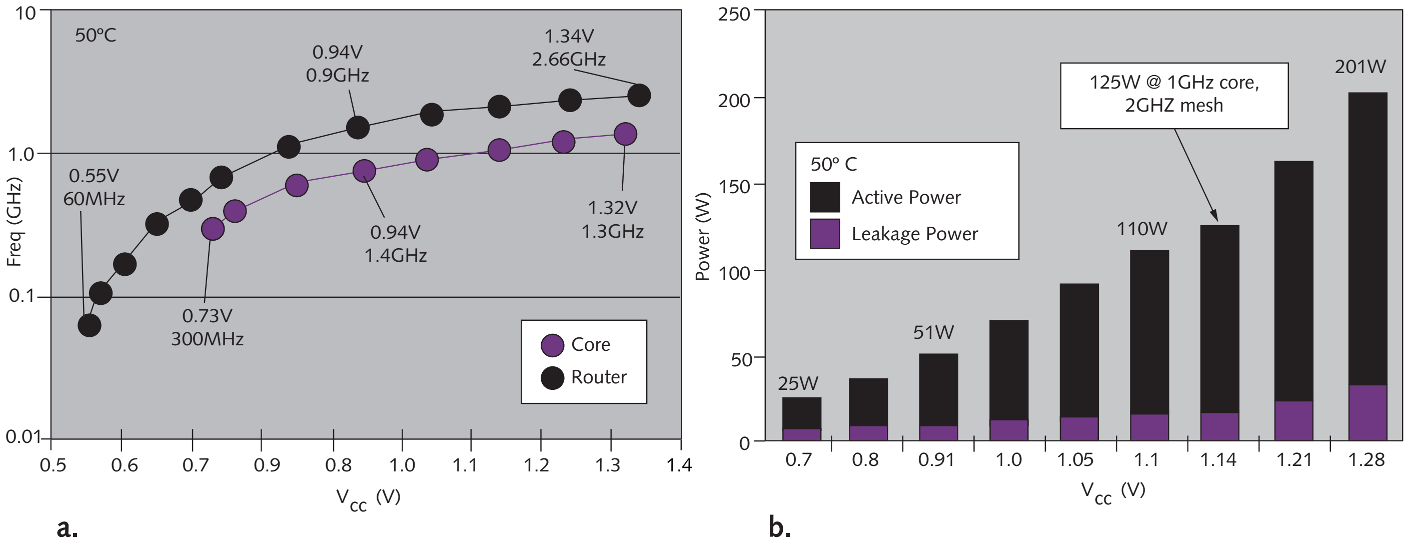

Figure 8. (a) The frequency-voltage chart shows an almost linear relationship between 0.9V and 1.3V for both core and router. (b) Power consumption specified at the low end (300MHz) is 25W (see also text); at the highest frequency specified (1.0GHz) it is 125W. The mesh will require at 2.0GHz approximately 12W.

For More Information

Please visit www.intel.com

From Figure 8-b, we find the lowest power number quoted by Intel for the experimental SCC to be 25W at approximately 0.7V. At that point in the chart, the SCC core frequency is 300MHz with router frequency at about 0.8GHz, estimated from the logarithmic scale of the figure. The data provided to MPR by Intel shows 25W at 0.7V at 125MHz. The P54C design allows the core to go down to DC without losing state. One draws the conclusion that the router voltage could be reduced or, at any point in the curve, intentionally increased, along with router and port frequency as a way to increase communications bandwidth. At 1.14V, the chip dissipates 125W with cores running at 1.0GHz; the router plus ports could be boot-time set to be clocked a bit faster at cost of additional power.

Since no two tiles come into direct electrical contact with each other, the router and ports must bear the responsibility of matching voltages, as seen in Figure 7-b, and frequencies. Frequency matching is accomplished by clock-crossing FIFO circuits.

Freedom to Experiment

Intel's shift from the scientific workload market—if such intentions could have been read into the creation of the 80-core TRP—to cloud computing—will bring a product based on the SCC chip into a market dense with competitors, one of the most formidable of which is Intel itself.

This article appears less than a month after Intel's recent announcement of the Xeon processor 7500 series, built with next-generation Intel Microarchitecture (Nehalem), a server processor series that Intel announced on March 30. The announcement comes less than two years from the previous introduction, in 3Q08, of the six-core one-thread, 1.9B-transistor 45nm Intel processor Xeon x7460 that comes equipped with a 16MB L2 cache and 1066MHz FSB, and is clocked at 2.66GHz. Its power specification is 130W. Its die size is 503mm^2.

The new 45nm Hi-K Intel Xeon processor 7500 series is implemented with 8 cores supporting a total of 16 threads and, according to the company, it shows a three-fold performance improvement on most workloads when compared with previous chips. The 7500 series is an SMP architecture, it is provided with a 24MB L3 cache, and it can communicate at 25.6GB/s among processors and I/O. It can access up to 2TB of memory. Its member chips are powered between 95W and 130W and may be clocked between 1.86GHz and 2.26GHz.

Intel's Single-chip Cloud Computer is a bridge to the future. It offers many options for learning and experimentation to programmers interested in finding the most efficient ways to write code for a modern machine—many more than offered by the 80-core TRP or some of the other existing multiple-core chips. ◇