4.4 TILE64™ Processor: A 64-Core SoC with Mesh Interconnect

Shane Bell, Bruce Edwards, John Amann, Rich Conlin, Kevin Joyce, Vince Leung, John MacKay, Mike Reif, Liewei Bao, John Brown, Matthew Mattina, Chyi-Chang Miao, Carl Ramey, David Wentzlaff, Walker Anderson, Ethan Berger, Nat Fairbanks, Durlov Khan, Froilan Montenegro, Jay Stickney, John Zook

Tilera, Westborough, MA

Centralized monolithic processor designs, such as single core and shared bus structures, are not scaling, leading to multicore designs becoming the norm [1-3]. Research highlights the benefits of mesh networks connecting these cores. [4,5]. The TILE64™ processor [6] is a multicore SoC targeting the high-performance demands of a wide range of embedded applications across networking and digital multimedia applications. Figure 4.4.1 shows a block diagram with 64 tile processors arranged in an 8×8 array. These tiles connect through a scalable 2D mesh network with high-speed I/Os on the periphery. Each general-purpose processor is identical and capable of running SMP Linux. Chip peak memory bandwidth is over 25GB/s using four 72-bit 800MHz DDR2 interfaces. Two PCI-e ×4 interfaces, two XAUI interfaces and two RGMII interfaces supply over 40Gb/s of I/O bandwidth. Flexible general purpose I/O ports and low-speed interfaces provide seamless integration with a variety of systems.

Each tile processor (Figs. 4.4.2 and 4.4.3) is a 3-wide VLIW machine with a 64-bit instruction word. The integer datapaths are 32b wide, supporting 32-bit, 16-bit and 8-bit operations. The TILE64 processor attains 144, 192 and 384 aggregate GOPS respectively, at 750MHz. The processor has a 32-bit virtual address space that translates to a 36-bit physical address through the 8-entry instruction TLB or the 16-entry data TLB. The translation buffers offer support for both private and shared memory. The TLBs also support pinning blocks of memory in the cache. There are separate 8KB L1 instruction and data caches. The L1 caches are backed by a unified 2-way 64KB L2 cache. The L2 cache can be shared among tiles, effectively providing up to 4MB of shared L3 Cache. An autonomous 2D DMA engine in each tile supports block copy functions. Block copies can be cache-to-memory, memory-to-cache, and cache-to-cache.

Tiles are connected through 2D mesh networks. The network routing logic is included in the SoC building blocks. Each of the five independent networks supports a distinct function: the static network (STN), the tile dynamic network (TDN), the user dynamic network (UDN), the memory dynamic network (MDN) and the I/O dynamic network (IDN). No hardware virtual channels are required on any network. Each network data width is independent of the other networks. For this implementation, all networks are 32b wide. Each network has five full-duplex ports—one each for the four compass points of routing (North, East, South and West) and a fifth connection to the processor. Each tile in the network supports 120GB/s interconnect bandwidth. The total network supports 240GB/s of bisection bandwidth.

The STN is a software-driven, software-routed scalar network for low-latency scalar communication between the tiles. The MDN and TDN networks are dynamically routed networks that implement the memory subsystem. The UDN and IDN are software-driven, message-oriented dynamically routed networks. All networks have single-cycle hop latency from tile to tile. The three software-visible networks are register mapped to support low-overhead access to them. Software on two adjacent tiles can communicate via register-to-register transfers with a two-cycle routing latency.

The four dynamic networks are dimensional-ordered wormhole routed. The switch (Fig. 4.4.4) is a full crossbar for non-blocking routing, with credit-based flow control. Buffering is implemented on all switch inputs and on the outputs from the tile processor to the switch. The MDN transports data requests from the tile to any of the four memory controllers on cache misses. Cache-line fill responses back to the tile are also transported along the MDN, while the TDN supports memory communication between tiles. The TDN can be used directly by the processor or the DMA engine to access another tile's L2 cache. This mechanism allows tiles' L2 caches to be aggregated as a distributed L3 cache.

The UDN and IDN networks are directly accessible by the processor ALU by mapping the networks within the register space. The UDN is used for user-level program communication. The IDN is used for communication with I/O interfaces. The OS also uses the IDN for inter-tile OS communication. IDN/UDN traffic is message-based and variable-length. Message-based communication introduces the risk of head-of-line blocking. Algorithms may require messages to be processed in an order different from the order received. To support this, messages are tagged to allow the receiver tile to sort messages into multiple queues. Messages on the IDN and UDN networks are tagged to specify in which one of seven destination queues they should be placed. Two additional catch-all queues handle any non-matching tags. A shared single-port SRAM stores the message words from the two software-visible networks (Fig. 4.4.5) to allow message reordering. Small per-flow queues provide a direct connection to the ALU. To reduce latency, the RAM is bypassed if the per-flow queues are not full. RAM allocation is controlled via a free-list, and head and tail pointers. The balance of RAM utilization between IDN and UDN is software controllable by defining the percent allocation allowed per network. Oversubscription by both networks allows the RAM to allocate on a first-come first-served basis. The message words are stored in linked lists with the head and tail pointers in the control logic and the RAM storing the next pointers along with the data. The RAM output de-multiplexes into several queues as defined by the original message tag. Each queue is mapped into the tile processor's register address namespace to allow the appropriate message to be read. The bandwidth of the RAM is designed to match the tile read bandwidth of a single de-multiplex queue. The UDN/IDN egress ports from the ALU are not multiplexed—an entire message is sent over the network before a new message can be started.

Figure 4.4.7 shows the die micrograph with major SoC blocks highlighted. The SoC blocks are designed using a standard place-and-route flow, augmented by external IP and internally designed register-file and custom cells. The architecture of the register file (Fig. 4.4.6) requires 3 write ports and 7 read ports. The read ports are designed as flow-through static reads with partial decodes of the address bits controlling the muxing of the entries. Over 615 million transistors are fabricated in a 90nm triple-V_t CMOS process. The chip is routed in 9 layers of Cu interconnect with 1 additional Cu redistribution layer to the C4 bumps. The package is a 1517-pin, organic substrate BGA using 10 layers and has 786 signal pins. Low-voltage supply and clock gating enables power efficient execution. A deep-packet inspection application running on TT silicon utilizing all the tiles is measured at 10.8W core power at 1V, 750MHz 85°C. First silicon is fully functional and boots SMP Linux.

Acknowledgements:
We thank the entire Tilera team for their excellent work and support.

References:
[1] J. Friedrich, B. McCredie, N. James, et al., "Design of the Power6™ Microprocessor", *ISSCC Dig. Tech. Papers*, pp. 96-97, Feb. 2007.
[2] J. Dorsey, S. Searles, M. Ciraula, et al., "An Integrated Quad-Core™ Opteron Processor", *ISSCC Dig. Tech. Papers*, pp. 102-103, Feb. 2007.
[3] U. Nawathe, M. Hassan, L. Warriner, et al., "An 8-Core 64-Thread 65b Power-Efficient SPARC SoC", *ISSCC Dig. Tech. Papers*, pp. 108-109 Feb. 2007.
[4] M. Taylor, J. Kim, J. Miller, et al., "A 16-Issue Multiple-Program-Counter Microprocessor with Point-to-Point Scalar Operand Network", *ISSCC Dig. Tech. Papers*, pp. 170-171, Feb. 2003.
[5] S. Vangal, et al, "An 80-tile 1.28TFLOPS Network-on-Chip in 65nm CMOS", *ISSCC Dig. Tech. Papers*, pp. 98, Feb. 2007.
[6] A. Agarwal, L. Bao, J. Brown, et al, "Tile Processor: Embedded Multicore for Networking and Digital Multimedia", *Hot Chips*, Aug. 2007.

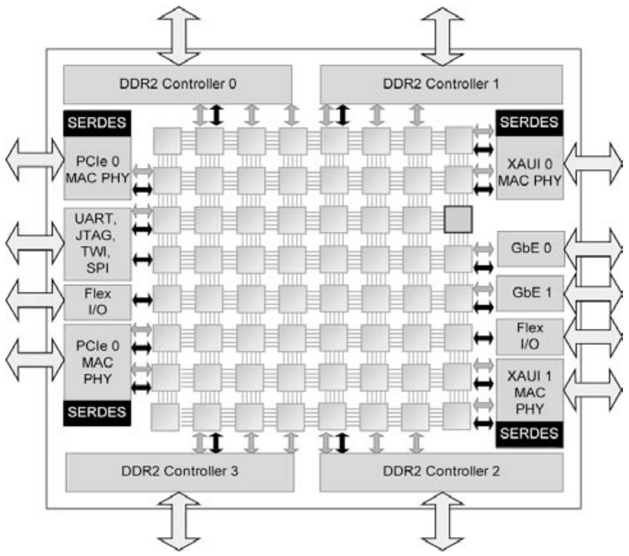


Figure 4.4.1: TILE64™ block diagram.

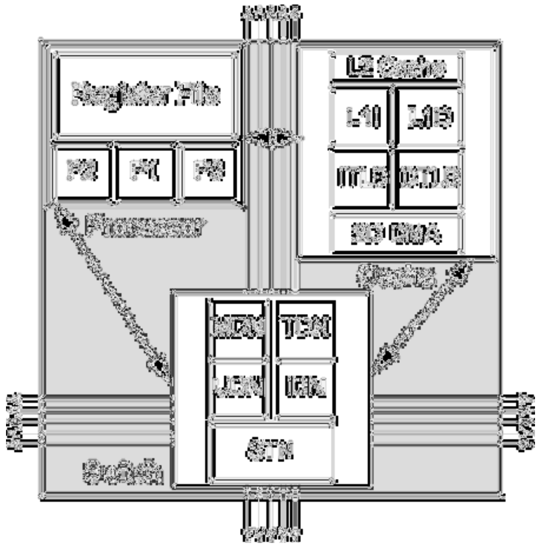
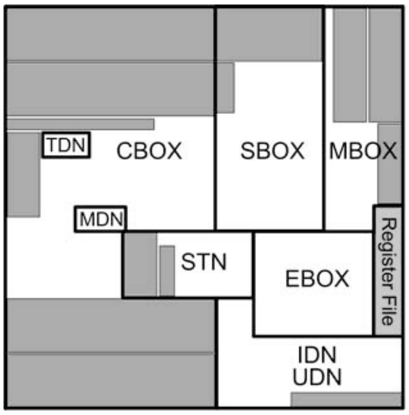


Figure 4.4.2: Single tile block diagram.



Cbox	L2 Cache Subsystem
Mbox	L1 Dcache LD/ST
Sbox	L1 Icache Fetch & Issue
Ebox	Integer Exectuion

Figure 4.4.3: Single tile physical layout.

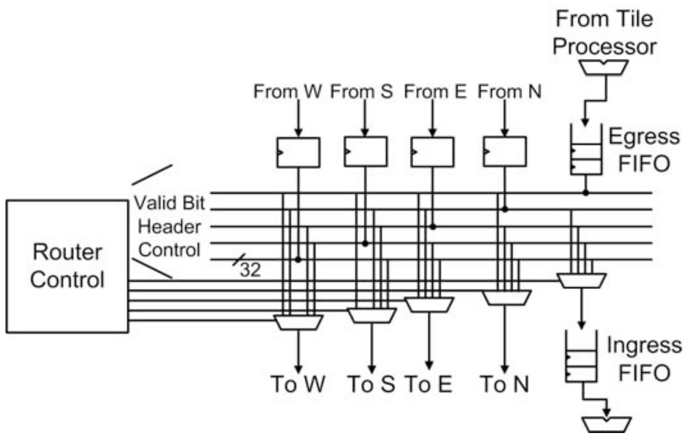


Figure 4.4.4: Dynamic crossbar switch.

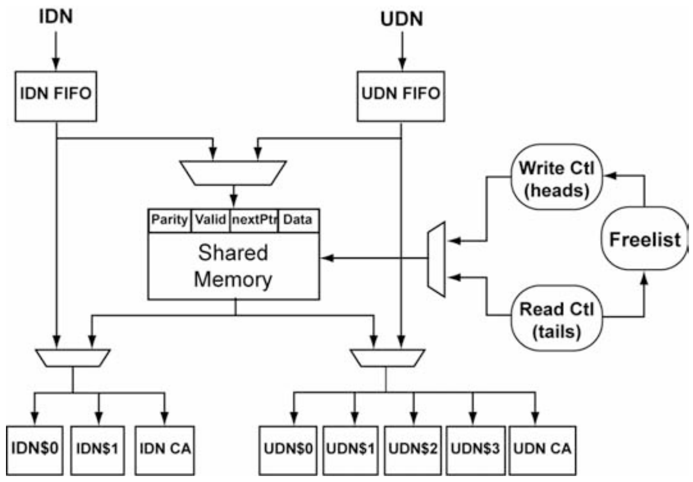


Figure 4.4.5: IDN/UDN de-multiplexor queues.

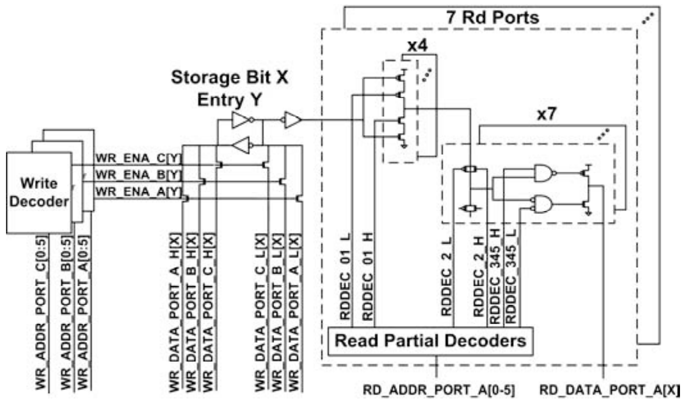


Figure 4.4.6: Register file.

Continued on Page 598

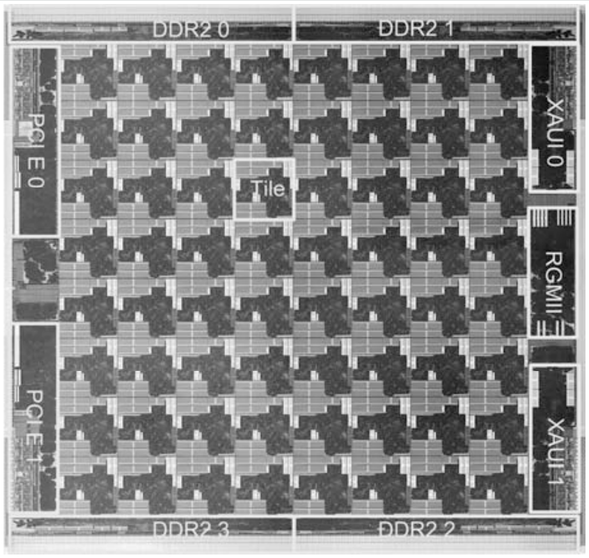


Figure 4.4.7: Die micrograph.