9.7 A 16-Issue Multiple-Program-Counter Microprocessor with Point-to-Point Scalar Operand Network

Michael Bedford Taylor, Jason Kim, Jason Miller, David Wentzlaff, Fae Ghodrat, Ben Greenwald, Henry Hoffman, Paul Johnson, Walter Lee, Arvind Saraf, Nathan Shnidman, Volker Strumpen, Saman Amarasinghe, Anant Agarwal

Massachusetts Institute of Technology, Cambridge, MA

The drive for performance in the face of increasing wire delay blurs the line between microprocessors and multiprocessors. Microprocessor designs such as the Alpha 21464 have multi-cycle "network" latencies between ALUs [1], and come close to having multiple, parallel fetch units, much as a multiprocessor. A recent paper [2] identifies the existence of a *scalar operand network* as a minimal criterion for a design to be a microprocessor. Specifically, a scalar operand network is the operand transport mechanism that joins the dynamic scalar operands and operations of a program to meet in space to enact the computation specified by a program graph. Scalar operand networks have grown more sophisticated starting from accumulator-ALU pairs, adding register files, bypassing networks, multiple ALUs, renaming, out-of-order execution, and most recently transitioning to asymmetric designs where routing an operand from the output of one ALU to the input of another takes multiple cycles.

The Raw microprocessor at MIT was implemented to explore architectural solutions to scalability problems in scalar operand networks [2,3]. Today's microprocessor designers are finding it increasingly difficult to convert burgeoning silicon resources into usable, general-purpose functional units. In particular, the use of global, centralized structures to implement operand naming, routing, scheduling, and other parts of the scalar operand network makes it difficult to scale the issue-width without impacting the frequency.

The Raw design divides the silicon area into an array of 16 identical, programmable tiles. A tile contains an 8-stage in-order single-issue MIPS-style compute processor, a 4-stage pipelined FPU, a 32kB data cache, two types of communication routers --- static and dynamic, and 96kB of instruction cache. These tiles are interconnected to neighboring tiles using four full duplex 32b networks, two static and two dynamic. The static router controls the static networks, which are used as point-to-point scalar transport for operands between the tiles. The dynamic routers and networks are used for all other traffic such as memory, interrupts, I/O, and message passing codes.

Each tile is sized so that a signal can travel through a small amount of logic and across the tile in one clock cycle. All signals are registered at tile boundaries, thereby allowing the clock frequency to remain constant even as issue width increases (i.e., as more tiles are instantiated). Larger Raw systems can be designed simply by stamping out more tiles. Figure 9.7.1 shows the array of Raw tiles, an individual Raw tile, and its registered-on-input network wires.

The static router is the main component of Raw's scalar operand network. The static router routes the outputs of instructions on one tile to the inputs of dependent instructions on other tiles. If the output is required on the same tile, then the 0-cycle latency internal compute processor bypass paths can be used. Live but not active values can be stored in the switch register file, compute-processor register file, or in the FIFOs of the network.

The 5-stage static router controls two routing crossbars and thus two physical networks. Each crossbar routes values between seven entities: the static router pipeline, north, east, south, west, the com-pute processor, and the other crossbar. The static router fetches 64b instruction words from an 8k-entry cache. Each word simultaneously encodes a small command (branch and decrement, local register file accesses), and 13 routes, one for each crossbar output. For each operand sent between tiles on the static network, there is a corresponding instruction in the instruction cache of each router through which the word will travel. These instructions are programmed by the compiler. Thus, the static routers collectively reconfigure the entire communication pattern of the network on a cycle-by-cycle basis, and enable Raw to handle both scalar and streaming data types.

The static router is flow-controlled, and does not proceed to the next instruction until all of the routes in the current instruction have completed. This ensures that destination tiles receive incoming words in a known order, even when tiles suffer unpredictable delays from cache misses, interrupts, or branch mispredictions. The static router provides single-cycle-per-hop latencies and can route two values in each direction per cycle. Because Raw's network is point-to-point, and because operands are routed only to those tiles that need them, the Raw design decimates the bandwidth required for operand transport relative to a comparable broadcast-based superscalar.

As shown in Fig. 9.7.2, tile-to-tile communication latency is reduced by integrating the networks directly into the bypass paths of the compute processor. Registers 24..27 are mapped to the four physical networks on the chip. For example, a read from register 24 pulls an element from an input FIFO, while a write to register 24 places data into an output FIFO. The output FIFOs automatically pull the oldest value out of the pipeline as soon as it is ready, rather than just at the writeback stage. This decreases the ALU-to-network latency by as much as 4 cycles. This logic is exactly like the standard bypass logic of a processor pipeline except that it gives priority to older instructions rather than newer instructions.

Unlike the 21464, where the assignment of operations to distributed ALUs is made at run-time, the assignment of operations to tiles for Raw is done by a sophisticated compiler at compile time. The compiler uses CAD-like algorithms to partition operations, place the operations on tiles, and specify the static router's route instructions to transfer operands between tiles [4]. Figure 9.7.3 overviews this process and Fig. 9.7.4 shows some application performance numbers.

The Raw chip 16-tile prototype is built in IBM's SA-27E, a 0.15μm, 1.8 V, 6-level Cu ASIC process. Although the Raw array is 256mm^2, we used a 331mm^2 die to allow a 1657 pin CCGA package. These HSTL pins provide 14 full-duplex 32-bit chip-speed channels that can be connected to either DRAM or stream I/O devices. The chip taped out in August 2002 at a worst-case frequency of 225 MHz. Figures 9.7.5 and 9.7.6 show details of the Raw chip and of a single Raw tile, respectively.

References:

[1] Personal communication, Joel Emer.

[2] M. Taylor et al. "Scalar Operand Networks," *Proceedings of HPCA*, February 2003.

[3] M. Taylor et al. "The Raw Microprocessor: A Computational Fabric for Software Circuits and General-Purpose Programs," *IEEE Micro*, March 2002.

[4] W. Lee et al, "Space-time scheduling of ILP on a Raw machine," *Proceedings of ASPLOS*, 1998.

 0-000-000-0/00/$10.00

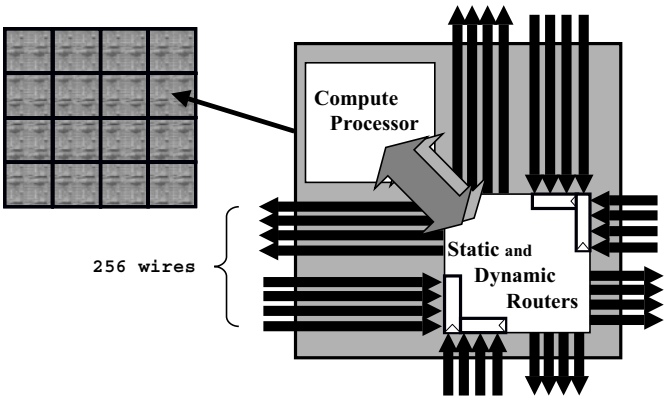

Figure 9.7.1: Raw tiles and networks.

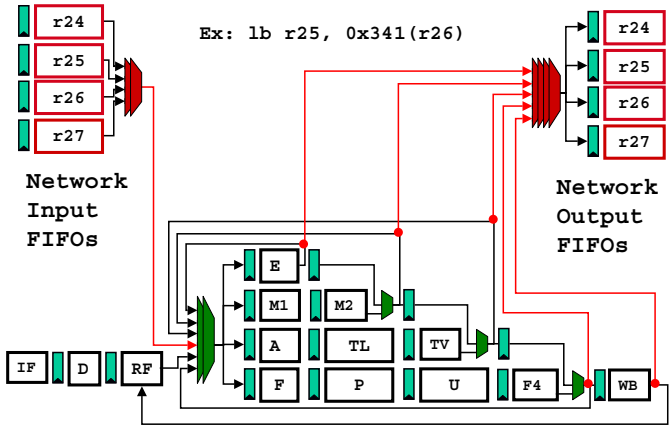

Figure 9.7.2: Network integrated into compute processor bypass paths.

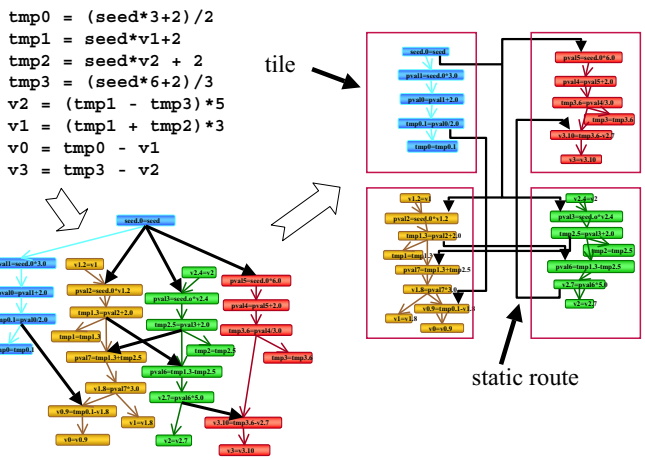

Figure 9.7.3: The raw compiler.

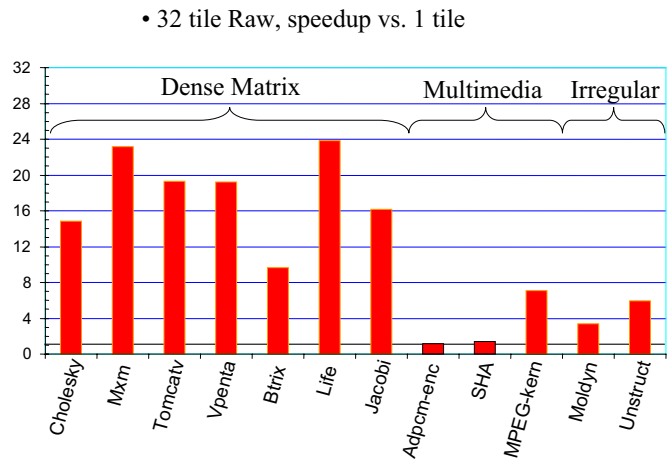

Figure 9.7.4: Raw performance on various benchmarks.

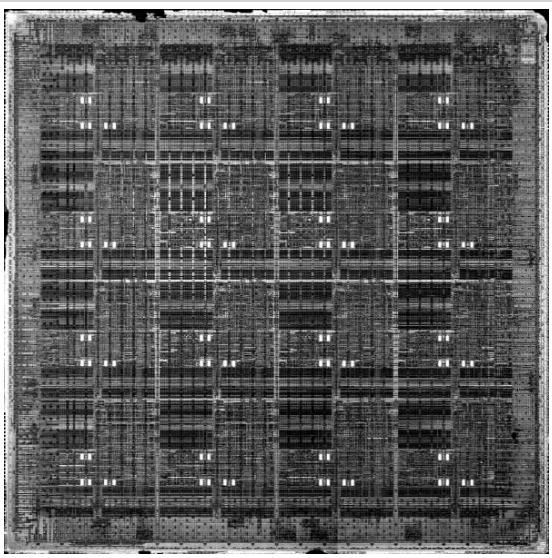

Figure 9.7.5: Chip micograph.

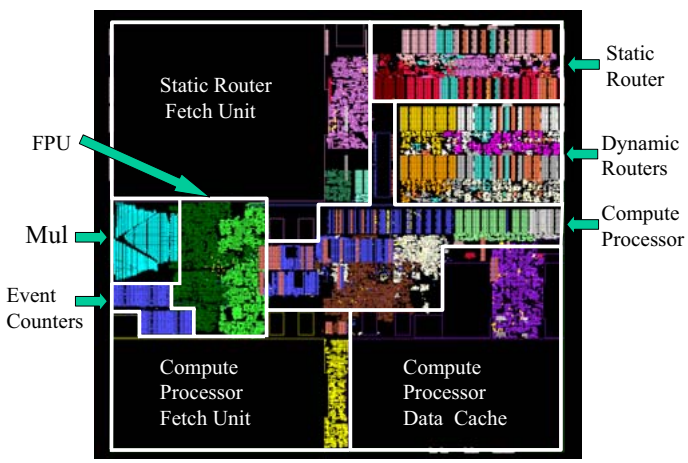

Figure 9.7.6: Raw tile layout.