SP 22.3: A Dual-Execution Pipelined Floating-Point CMOS Processor

John A. Kowaleski Jr., Gilbert M. Wolrich, Timothy C. Fischer, Robert J. Dupcak, Patricia L. Kroesen, Tung Pham, Andy Olesin

Digital Equipment Corporation, Hudson, MA

A floating point unit initially implemented on a 300MHz microprocessor [1, 2] in a 0.5μm/4-metal layer CMOS process and subsequently scaled for use in a 433MHz version of the microprocessor [3] in 0.35μm/4-metal layer CMOS process is described. This floating-point unit executes two floating-point instructions per cycle achieving 600Mflops (peak) performance at 300MHz and 866Mflops (peak) at 433MHz. It supports IEEE and VAX data types and rounding modes, including IEEE rounding to plus infinity and minus infinity. The floating-point unit contains 263,000 transistors and uses 6825μmx2025μm of chip area in the 0.35μm process. A micrograph of the chip is shown as Figure 1 in Paper 13.7. A single wire two phase system is used to provide a 3.3ns cycle consisting of two 1.65ns phases in the 300MHz implementation, and a 2.3ns cycle consisting of two 1.15ns phases in the 433MHz implementation.

The floating-point unit, Figure 1, consists of a register file, and two functional sub-units: add pipeline and multiply pipeline. The add pipeline incorporates an iterative divide unit, that, combined, execute all floating-point instructions except for multiply. The register file has nine ports: two read ports and one write port per functional pipeline operand access, one read port for stores, and two write ports for register file loads. Bypass paths forward data from each of the four write buses of the register file to each of the five register file read buses. This configuration supports the loading/unloading of data for two floating-point operations per cycle. The latency for both add and multiply pipelines is four clock cycles. Carry propagate adders of 56b and 64b in width are used in the fraction data path to perform effective subtract, to normalize and round fractions, to generate 3x the multiplicand for the radix-8 Booth multiply array and to perform remainder reduction in the divider. These adders along with the fraction multiplier array represent the core components in the floating-point unit.

The fraction adder design uses both carry look-ahead and carry select logic, and executes in a single clock phase. Domino logic stages are employed from propagate-kill (PK) development through prefix global carry resolution. The evaluation sequence is moderated by buffered clocking to improve noise immunity and to prevent race through of new cycles precharge into result latches.

The adder input stage, a precharged domino cross-coupled circuit, develops the bitwise PK signals (Figure 2). The PK structure is a cross-coupled precharge circuit, gated with a clocking signal, either raw or derived. This guarantees a glitch free rising kill or propagate, necessary to gate the next stage of the domino sequence. The downstream logic assumes a generate at each bit position if neither kill nor propagate is enabled.

The adder is partitioned into groups typically eight bits in width. A precharged n-stack circuit determines group-kill and a group-propagate signals from bitwise kills and propagates (Figure 4). A parallel structure is built across the more significant half of the group to speed up group kill evaluation. The group-kill and group-propagate signals feed into the global carry array, Figure 3.

This global array decodes group kills and group propagates into intergroup carry signals. The intergroup carry signals are broadcast from each source group to every more significant group that might be affected by the carry. This stage is constructed as a distributed precharge NOR assuming dominoed inputs. The group carry selects are determined from the less significant intergroup carries by a NOR structure and a static load. During normal operation, delayed precharge on the intergroup signals reduces the interval in which the static load draws current. The static load device is turned off when the adder is not in use. Special care is taken to check for any race through conditions that might result from entering and exiting a power down condition in the static devices. This static load is used in the second serial distributed NOR, providing a restoring fanout to the precharged array and speeding up the full carry evaluation. Its output is used both directly and in its inverted form to control the selection of the appropriate bitwise sum.

The bitwise sum cells are cascode logic XOR-latch structures with a dominoed propagate as one input and a pair of bit specific carries along with the group carry select [4]. Each sum cell instance is tailored for its specific datapath function.

Datapath adder performance enabled its use for divider. A normalizing nonrestoring algorithm is used and a complete partial remainder is produced on each cycle. Decode on the most significant 4b of the remainder and an optimizing shift over strings of zeroes and ones yields an average of 2.4 quotient bits per cycle [5].

Rounding in the add pipe is facilitated by the structure of the fraction adder, described above, and a half adder. The data that is to be simultaneously added and rounded, is preconditioned through a half adder producing new sum and carry shifted one bit position toward the MSB, that are the actual inputs to the add pipe fraction round adder. A carry-in is applied to the vacated LSB position in the shifted operand if required. For rounding, decode logic is added to the four LSBs of the adder that pre-computes if there will be a carry out of this section as a result of rounding. These decodes pre-compute carries into the fraction for two cases: 1) if the rounded result would have a 0 in the MSB position, a normally injected round would be selected, or 2) if there is a 1 in the MSB position, an injected round shifted 1b closer to the MSB is selected to compensate for the subsequent normalization. Precomputed carries as a result of rounding are combined with group carry selects generated in the global carry array of the adder, precomputed signals that indicate the distance a round carry will propagate, and the value of the MSB prior to rounding to choose bitwise sums for the result. This result of round adder is normalized as a function of the MSB and returned to the register file. Figure 5 is the multiplier CSA and MUX. Figure 6 is a micrograph of the FPU.

Acknowledgments:

The authors acknowledge contributions of R. Allmon, W. Bowhill, T. Broch, J. J. Ceparski, E. C. de Briae, J. J. Ellis, J. P. Ellis, B. Gieseke, J. Huggins, J. Laderoute, M. D. Matson, T. McDermott, P. Rubinfeld, B. Shah, C. Stark, and M. Tareila

References:

[1] Edmondson, J., et al., "Superscalar Instruction Execution in the 21164 Alpha Microprocessor," IEEE Micro, April, 1995.

[2] Bowhill, W., et al., "A 300MHz 64b Quad-Issue CMOS Microprocessor," ISSCC Digest of Technical Papers, Feb., 1995.

[3] Gronowski, P., et al., "A 433MHz 64b Quad-Issue CMOS RISC Microprocessor," ISSCC Digest of Technical Papers, pp. 222-223, Feb., 1996.

[4] Dobberpuhl, D., et al., "A 200MHz 64b Dual-Issue CMOS Microprocessor," IEEE J. of Solid-State Circuits, No. 11, Vol. 27, Nov., 1992.

[5] MacSorley, O. L., "High-Speed Arithmetic in Binary Computers," Proceedings of the IRE, Jan., 1961.

0-7803-3136-2/96/$5.00©IEEE

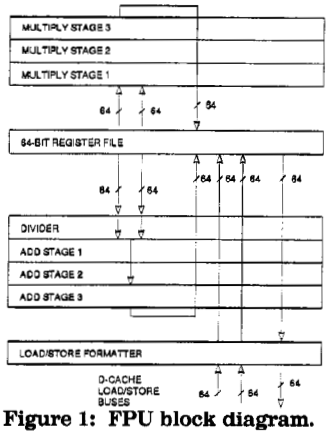

Figure 1: FPU block diagram.

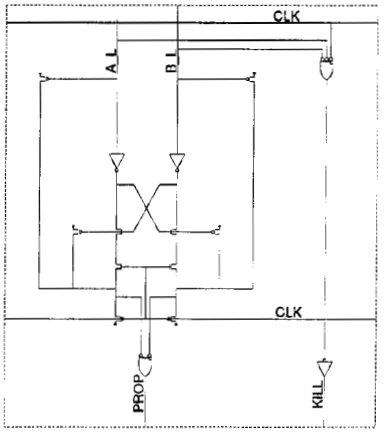

Figure 2: Domino PK cell.

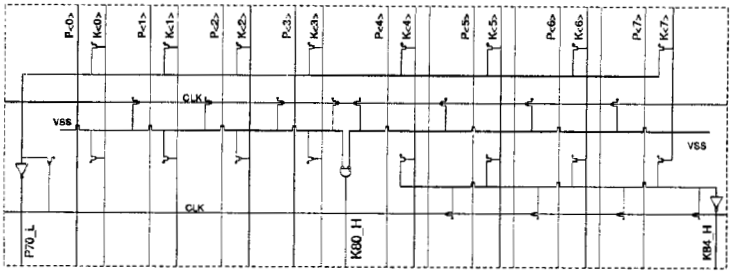

Figure 4: Group propagate-kill.

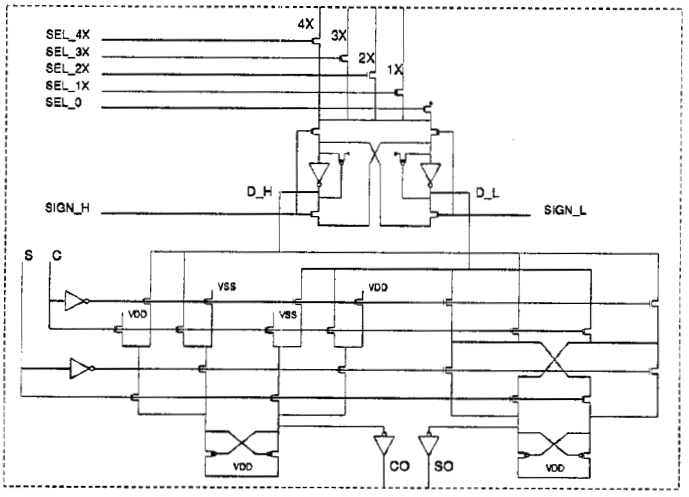

Figure 5: Multiplier CSA and MUX.

Figure 6: See page 473.

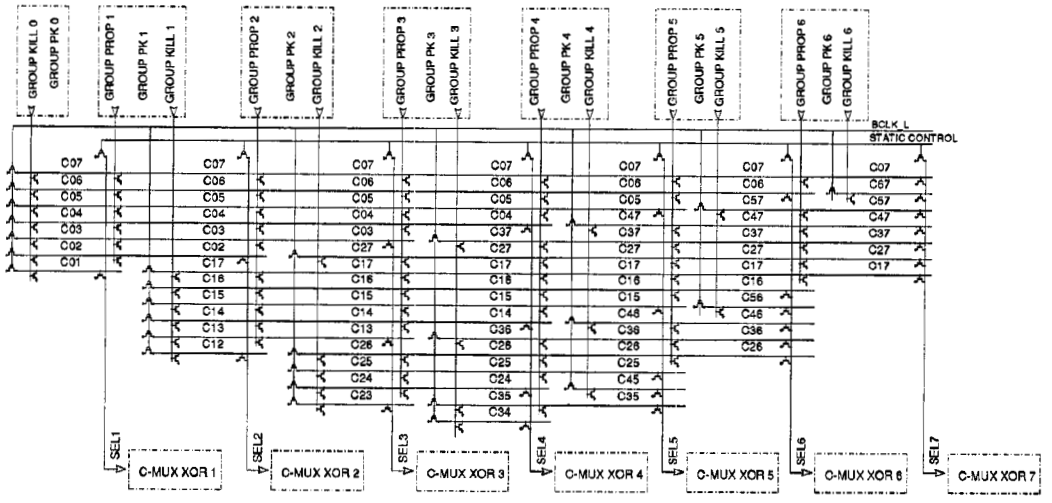

Figure 3: Diagram of global carry array.

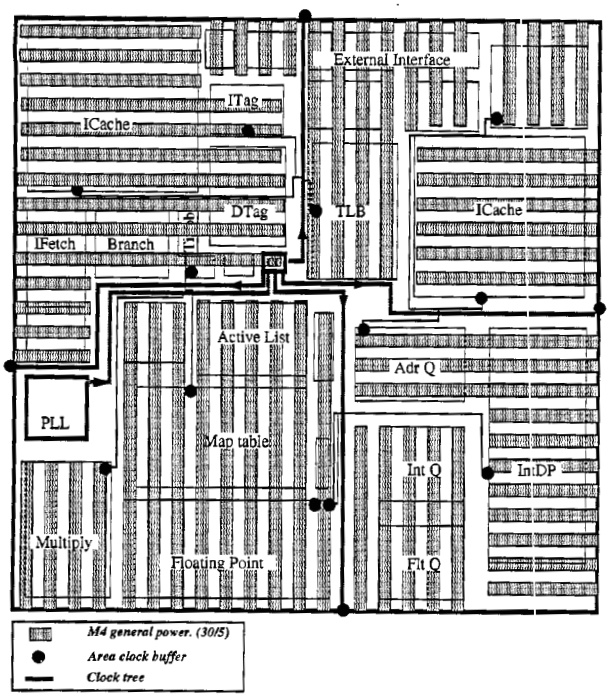

Figure 1: Clock tree fits with M4 power distribution.

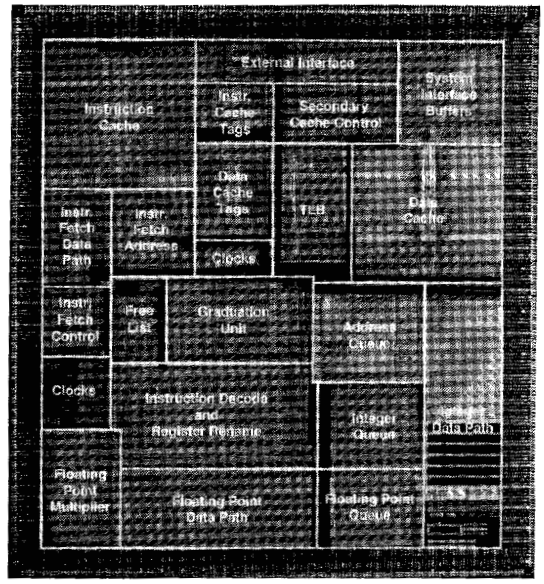

Figure 8: Chip micrograph.

SP 22.3: A Dual-Execution Pipelined Floating-Point CMOS Processor

(Continued from page 359)

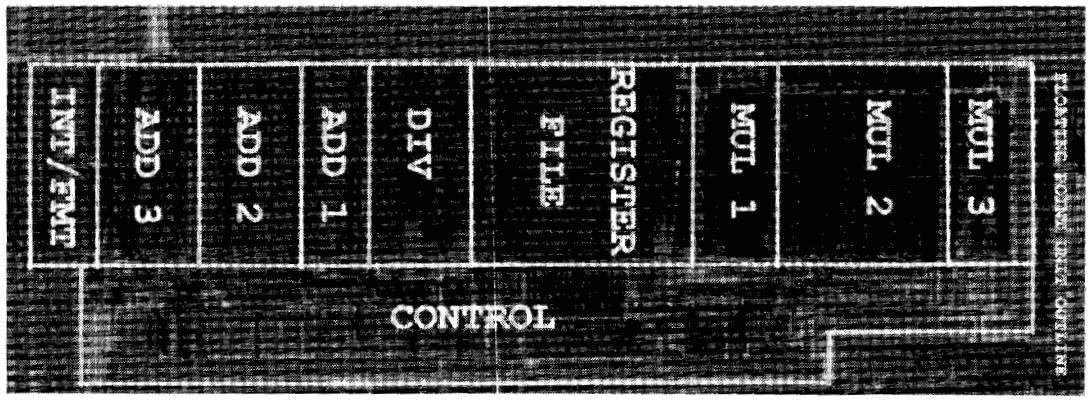

Figure 6: Floating-point unit micrograph.