StrongARM: A High-Performance ARM Processor

Rich Witek and James Montanaro
Digital Equipment Corporation

Abstract

A 32-bit 162MHz/215MHz custom VLSI ARM(tm) microprocessor is described. The chip contains two 16Kbyte, 32-way set associative caches for instructions and data. The 2.1M transistor chip is fabricated in a 2.0V, 0.35µm, 3-layer metal CMOS process. It dissipates 0.5W at 162MHz/1.5v and 1.1W at 215MHz/2.0v

Introduction

StrongARM 110 (tm), a custom VLSI implementation of the ARM (tm) architecture, delivers 184 Drystone/MIPS at 162MHz while dissipating 0.5W using an internal supply of 1.5V. In applications which require higher performance, the chip may be operated at 215MHz with a 2.0V internal supply dissipating 1.1W. The external interface always runs at 3.3V. The die contains 2.1 million transistors and measures 7.8mm x 6.4mm. It is fabricated in a 2.0V, 0.35µm drawn (0.25µm effective), 3-layer metal CMOS process and packaged in a 144-pin thin quad flat pack. Clock generation is accomplished using an on-chip PLL with 3.68MHz input clock to minimize high frequency clock signals on the board [1]. The chip is pseudo-static and clocks may be stopped in either phase to minimize power consumption

Processor Overview

The major elements of the processor are shown in the block diagram (figure 1).

The register file has three read ports and two write ports. The three read ports were chosen to simplify the control by providing all the arguments to most instructions in a single cycle. The two write ports allowed most instructions to be fully pipelined and write their results in a single cycle.

The EBOX contains a single-cycle ALU with a full 32-bit bidirectional shifter on one of the input operands. In one cycle, the EBOX can perform a 0 to 32-bit shift, a 32-bit ALU operation, and compute the condition codes of zero, overflow, carry, and negative on the result in time to affect a branch in the Issue stage of the same cycle.

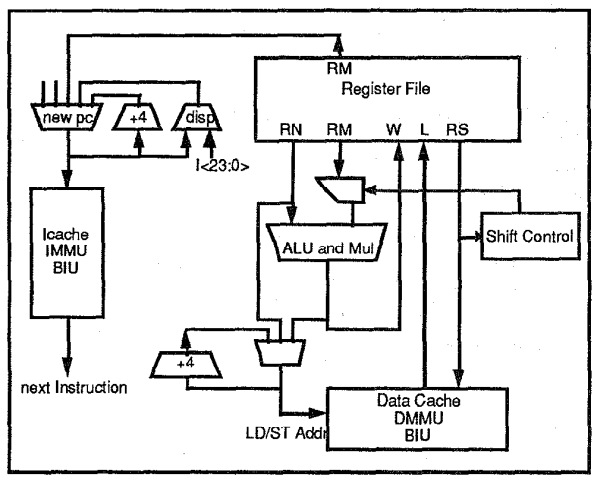

Figure 1: StrongARM Block Diagram

The EBOX also contains a 32x32-bit multiplier. The multiplier consist of a 12x32-bit carry save multiplier array which is used for one to three cycles depending on one of the input operands and a 32-bit final adder to reduce the carry save result. For Multiply Accumulate instructions, the accumulate value is inserted into the array so that an additional cycle for the add operation is not required.

The processor contains separate 32 entry, fully associative translation buffers (TB) for instruction fetches and data accesses. Each entry in the TB can map a 1Mbyte section, or a 64Kbyte large page, or a 4Kbyte small page. TB fills are performed by hardware without using a software exception routine.

The processor features separate 16Kbyte, 32-way set associative virtual caches for instructions (Icache) and data (Dcache). The split Instruction and Data caches simplify the control and allow full pipelining of load and store instructions. The Dcache is writeback and stores the physical address along with the data at fill time. The stored physical address is used when the block is castout

1063-6390/96 $5.00 © 1996 IEEE
Proceedings of COMPCON '96

of the cache so a translation is not need at replacement time. The Dcache has one dirty bit for each 16 byte subblock. This reduces the amount of data written back to memory when only part of the block has written. Each cache is implemented as 16 fully associative blocks. This allows a small section of the cache to be powered up and only the needed 32-bit word to be read without resorting to a direct mapped cache.

The processor includes a write buffer with 8 entries of 16 bytes. The write buffer is used to buffer stores that miss in the Dcache and castouts from the data cache. Load misses are allowed to pass stores in the write buffer. Each block of the write buffer has a physical address and a comparator to check for load to a piece of data currently in the write buffer. In the case of a load hit in the write buffer, the load is stalled until the write is completed.

Processor Pipeline

The processor is a single issue design with a classical 5 stage pipeline. The stages of the pipeline are Fetch (F), Issue (I), Execute (E), Buffer and cache access (B), and result Write (W) as show in figure 2.

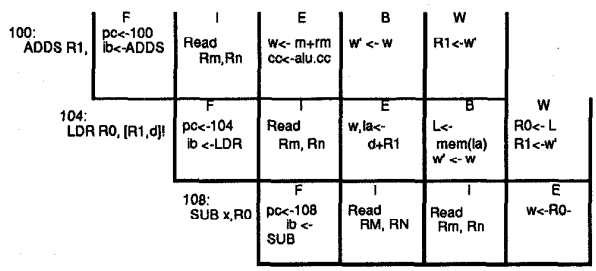

Figure 2: StrongARM Pipeline

The instruction unit (IBOX) operates in the first 2 stages (Fetch and Issue) of the pipeline. In the Fetch cycle, the next instruction is fetched from the instruction cache and latched into the instruction buffer. In the Issue stage, all register dependencies and execution unit (EBOX) availabilities are checked and the instruction is issued if there are no conflicts. If there are conflicts, the instruction is held in the IBOX until all the dependencies have cleared. Result forwarding is provided for all results between their generation and register file writing.

Most instructions only need one Issue cycle. The Load Multiple, Store Multiple, Swap, and Multiply Long instructions need additional cycles. The IBOX holds these instructions while advancing the rest of the pipeline to decompose these complex instructions to their fundamental single issue cycle operations. The Load Multiple and Store Multiple instructions are decomposed by the IBOX into a series of single Load/Store instructions as seen by the rest of the machine. Likewise the Swap instruction is decomposed into a Load and a Store. The Multiply Long and Multiply Long Accumulate (MLA) instructions use a second issue slot to acquire a second register file write slot for the second 32 bits of the result. The MLA instruction performs an B=A*X+B operation where A and X are 32 bits, and B is 64 bits. The high order 32bits of B are read and the second register file write slot is reserved in the second issue cycle of a MLA .

The IBOX performs PC-relative branches and subroutine calls, and MOV to PC (return instructions) in the Issue stage. A dedicated PC adder is used along with the PC incrementer to perform these operations. The IBOX can resolve conditional branches in the Issue stage even when the condition codes are being updated in the current Execute cycle. By providing this optimized path, the IBOX is able to turn branches with only a one cycle penalty and no cycles lost for branch not taken. By handling the branches very early in the pipe the need for branch prediction hardware was eliminated (Figure 3).

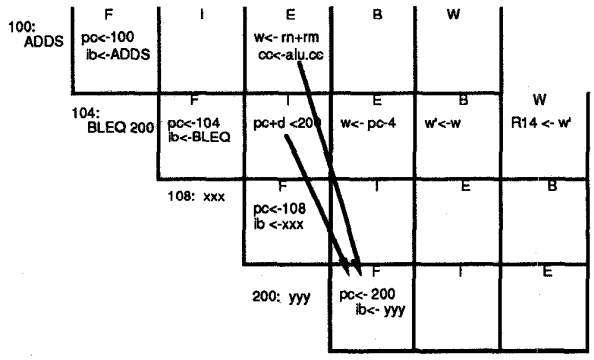

Figure 3: Branch Pipeline

For Load and Store operations the effective address is calculated in the E stage of the pipe and the cache and translation buffers (TB) are accessed in the B stage. The cache is accessed in a single cycle for both reads and writes. On writes the cache is first read and the data is latched in the sense amps. If the write is aborted due to memory management unit (MMU) access violations, the original data is written back in the cycle that follows the exception while the first instruction of the exception handler is being fetched. The Dcache provides a 2-cycle latency for load return data to the register file.

Low Power Design

Our past designs have emphasized performance without significant concern for power dissipation.[2,3] Since it was intended for the portable market, this design needed the maximum performance possible for 0.5W or less. This goal was addressed through a variety of avenues:

Reducing the power supply voltage was paramount. The use of a 2.0v, 0.35mμ process with 0.35v device thresholds joined with traditional high-performance custom design techniques to enable a high speed design, even at 1.5v.

Proceedings of the COMPCON Spring '96 - 41st IEEE International Computer Conference
1063-6390/96 $10.00 © 1996 IEEE

Conditional clocking was pursued aggressively from the beginning of the design. While this created some of the most challenging critical paths in the design, it allowed us to reduce switching and, in some cases, eliminate latches. Edge-triggered latches were used through the majority of the design to minimize the gate load on the clock. Power is further reduced by running the internal clocks off the slower bus clock during cache fills.

The constraints on total power led to limiting the clock frequency and moving instead to a longer tick model of the design. One single cycle path includes a 32-bit shift, 32-bit add and setting condition codes to direct the PC MUX in the next cycle. Another is the 16K associative cache access, from address in to data out in a single cycle. In past designs driven solely by performance, direct-mapped instruction and data caches have been used since their shorter access time provides higher overall performance, despite the higher hit rate of an associative cache. The long cycle time of this design allows time for the associative cache access and the increased hit rate minimizes pin power and increase performance.

The choice of internal supply voltage and clock frequency was made based on power analysis of an earlier design.[2] This analysis indicated that we could meet the 0.5W goal with largely the same circuit techniques used on past chips, adapted slightly to make the chip fully static. This conclusion was dependent on the low-voltage characteristics of the process, particularly the low Vts. However, the leakage characteristics of these devices posed problems for the powerdown modes and on certain pseudo-dynamic nodes. The chip supports two powerdown modes, Idle (20mW) and Sleep (50uA). In Idle, the PLL is running but the internal clock grid is stopped. The cache array devices were lengthened to reduce their leakage so that the power requirement for Idle could be met. In Sleep, the leakage requirement is much tighter so it is addressed by turning off the internal power supply. The I/O circuitry is powered in Sleep so that we can drive valid data on the pins.

Clocking

In an earlier design[2], the clock was distributed as a single monolithic node. While this technique has advantages with respect to speed and simplicity, it is not appropriate on a low-power chip using conditional clocking. On this design, the internal clock grid is a standard H-tree distribution system with minor variations. Since the pins can run asynchronous to the core, there is a separate clock region associated with the pin circuitry. Inverter buffers were used in both distribution systems to allow use of narrow wires and reduce the grid capacitance.

The chip has five clock regions, the pins, bus interface control and write buffer, core (integer unit and memory managment units), Icache, and Dcache. The first two regions run off the bus clock while the final three are connected to the internal clock grid. During cache fills, the internal clock grid is clocked by the bus clock rather than the PLL to save power and to minimize the synchronization time during fills. The write buffer receives clocks from the internal clock grid and the bus clocks since during stores it must pass data across this boundary.

Latches

Differential edge-triggered latches were used to reduce total latch delay and reduce clock load. The majority of the latching occurs on the rising edge of the clock but falling edge-triggered latches were sometimes used, primarily when crossing clock region boundaries.

External Interface

The pin interface supports different modes of operation to allow compatibility with systems using existing ARM chips while enabling system cost savings and performance improvements for new systems. The on chip PLL multiplies the 3.68MHz input clock by 24 to 78 to generate the high speed core clock. The speed is selected by four pins. For compatibility with existing ARM systems the bus clock can be supplied by the system. For new designs the processor can drive the bus clock by a divider on the PLL clock. The dividers supported range from 2..9 to provide a large range of bus speeds. Configuration pins allow selection of PLL speed, source of bus clock, speed of bus clock and other bus timing options. When the processor is driving the bus clock a Wait For Interrupt instruction places the processor in Idle mode where the core and bus clock are stopped until an external interrupt is asserted.

Summary

The StrongARM 110 provides a low cost, low power, high performance processor that will enable more complex applications to be provided in handheld devices. The StrongARM 110 will also provide a new level of performance for low cost embedded processors.

References

[1] von Kaenel, V. et al., "A 320MHz, 1.5mW CMOS PLL for microprocessor clock generation", ISSCC 96, February 1996.

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

[3] Bowhill B., et al., "A 300MHz 64b Quad-Issue CMOS RISC Microprocessor", ISSCC 1995, February 1995.

Proceedings of the COMPCON Spring '96 - 41st IEEE International Computer Conference
1063-6390/96 $10.00 © 1996 IEEE

ACKNOWLEDGMENTS

We would like to acknowledge the contributions of the following people: Krishna Anne, Andrew J. Black, Elizabeth M. Cooper, Daniel W. Dobberpuhl, Paul M. Donahue, Jim Eno, Alejandro Farell, Gregory W. Hoeppner, David Kruckemyer, Thomas H. Lee, Peter Lin, Liam Madden, Daniel Murray, Mark Pearce, Sribalan Santhanam, Kathryn J. Snyder, Ray Stephany, Stephen C. Thierauf, F. Aires, M. Bazor, G. Cheney, K. Chui, M. Culbert, T. Daum, K. Fielding, J. Gee, J. Grodstein, L. Hall, J. Hancock, H. Horovitz, C. Houghton, L. Howarth, D. Jaggar, G. Joe, R. Kaye, I. Kim, S. Lum, D. Noorlag, L. O'Donnell, K. Patton, J. Reinschmidt, S. Roberts, A.Silveria, D. Souyadalay, E. Supnet, L. Tran, D. Zoehrer and the PLL design team at CSEM.

The support which we received on many aspects of the design from the people at Advanced RISC Machines, Ltd. was very important and appreciated.

Proceedings of the COMPCON Spring '96 - 41st IEEE International Computer Conference
1063-6390/96 $10.00 © 1996 IEEE