FPGA Coprocessing in Multi-Core Architectures for DSP

J Ryan Kenny, Altera Corporation, rkenny@altera.com
Bryce Mackin, Altera Corporation, bmackin@altera.com

Overview

The market for high-performance multi-core processing is expanding rapidly, though there are currently few market-specific (tailored application) coprocessing solutions available.

With the proliferation of multi-core processing computers, complex signal-processing applications can be accelerated with distributed functions, although not always in an optimized fashion. The processing cores are still limited by cache size and coherency, memory bandwidth, and in some cases, power and cooling.

One approach to improving processing power and efficiency in multi-core processors is the use of FPGAs as application-specific coprocessors. These devices can be designed to fit into AMD Opteron® sockets so as to replace one of the processors with pipelined FPGA logic. The advantages of this approach have been explored by Altera Corporation, and applied in some cases by XtremeData Computing. This paper explores some of the advantages and applications to sensor signal processing.

Coprocessing Architecture

Two example architecture sets will be shown for this paper. As shown in Figures 1 and 2, the first is a set of notional coprocessing architectures for the Intel Xeon® Quad Core and AMD Opteron layout. The second (shown in Figure 3) is a representative system developed by XtremeData, Inc. utilizing an FPGA in an XDI module. In all three cases, FPGAs are placed in CPU sockets, and require application hardware to best utilize the FPGA logic.

In the Xeon architecture, processors and coprocessor are connected using front side bus (FSB) architecture. A north bridge (information available from Intel) is used to connect each FSB to one another.

The AMD Opteron architecture uses direct HyperTransport interconnects between each processor\coprocessor socket.

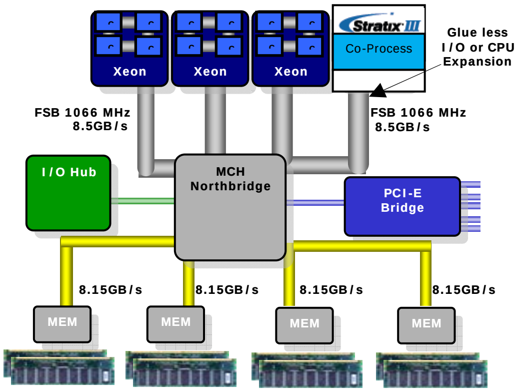


Figure 1: Notional Intel Xeon Quad Core Coprocessing Architecture

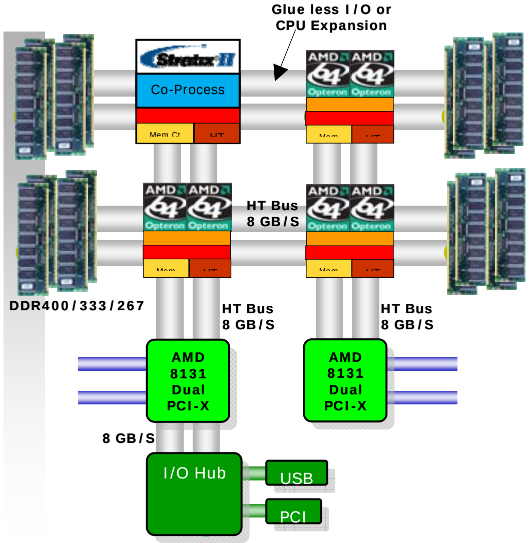


Figure 2: Notional AMD Opteron Quad Core Coprocessing Architecture

The final architecture in Figure 3 was created by XtremeData Computing. It utilizes a dual-processor motherboard, with an AMD socket-compatible module interface for the FPGA and other components (see Figure 4).

This finished motherboard plugs into standard commercial enterprise, rack, or blade servers. It utilizes a low-latency, high-bandwidth HyperTransport interface to the other processor. The FPGA uses all resources intended for a CPU: power supply, heat sinks, HyperTransport links, and is programmed with on-chip memory controllers. No hardware change was made to the motherboard.

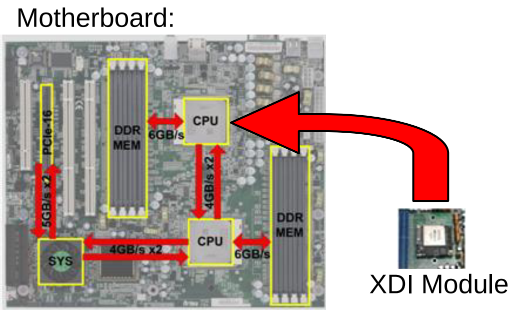


Figure 3: XtremeData Implementation of FPGA as Coprocessor

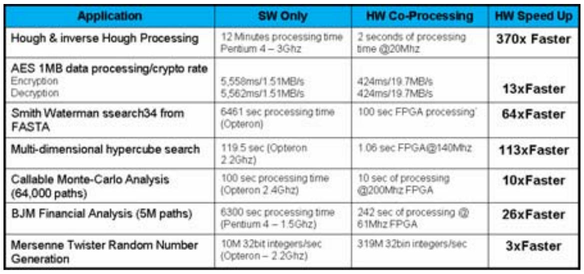
Figure 4: XtremeData Coprocessor Module

Potential Algorithmic Returns

Hardware acceleration using the FPGA as a coprocessor is intended to offer 10X to 100X speed improvement for tailored algorithms, and anywhere from 3X to 50X speed improvement for the user application. These target numbers are based on commercial applications such financial analysis, data warehousing, and biosciences.

Several selected algorithms taken from these commercial applications have been implemented and tested on FPGA coprocessors for evaluation. These are shown in the table below.

Table 1: Example Algorithm Acceleration



Application	SW Only	HW Co-Processing	HW Speed Up
Hough & inverse Hough Processing	12 Minutes processing time Pentium 4 – 3Ghz	2 seconds of processing time @20Mhz	370x Faster
AES 1MB data processing/crypto rate Encryption Decryption	5,558ms/1.51MB/s 5,562ms/1.51MB/s	424ms/19.7MB/s 424ms/19.7MB/s	13xFaster
Smith Waterman ssearch34 from FASTA	6461 sec processing time (Opteron)	100 sec FPGA processing	64xFaster
Multi-dimensional hypercube search	119.5 sec (Opteron 2.2Ghz)	1.06 sec FPGA@140Mhz	113xFaster
Callable Monte-Carlo Analysis (64,000 paths)	100 sec processing time (Opteron 2.4Ghz)	10 sec of processing @200Mhz FPGA	10xFaster
BJM Financial Analysis (5M paths)	6300 sec processing time (Pentium 4 – 1.5Ghz)	242 sec of processing @ 61Mhz FPGA	26xFaster
Mersenne Twister Random Number Generation	10M 32bit integers/sec (Opteron – 2.2Ghz)	319M 32bin integers/sec	3xFaster

In addition to the performance advantages of hardware acceleration, replacement of either an Intel or AMD processor significantly reduces power consumption and heating in the system. This is a result of the streamlining of software and hardware functions to where they are best suited. Initial benchmarks on the expected power savings depends on the algorithm, but these benchmarks have been performed on one medical and one financial example.

Implementation

In addition to the FPGA design and drop-in module make\buy, the application engineer has the non-trivial task of modifying the application software to exercise the FPGA coprocessor for tailored hardware operations. There are several approaches to this process, which can be logically broken out into two steps.

First, the systems engineer needs to decide which functions in the system will be offloaded into the FPGA for accelerated processing. This process can be laborious for complex financial algorithms, but is somewhat straight-forward for digital signal processing applications. An example design flow with coprocessing will be offered.

Second, the designer must identify the function calls and interfaces for coprocessing, as well as the task distribution among the remaining processors.

As multi-core processing becomes more prevalent, more software tools will become available to help the designer parallelize and compile their systems for N hardware nodes. Function calls can be translated from C to efficient HDL using proprietary tools (these will be described in the paper). This application tailoring does not require FPGA knowledge, allowing the system designer to build algorithm architectures confidently.

Summary

This paper will expand on and explore applications for digital signal processing, to include new benchmarks and developer survey information on average dedicated algorithmic content to DSP devices in sensor design.