Hardware Accelerated Decoding of FIX/FAST and Book Building of Market Data

Project Proposal – CSEE E4840 Spring 2013

Danqing Hua, Chang Liu, Junkang Ren, and Raghavan Santhanam
Columbia University
{dh2604, cl3078, jr3289, rs3294}@columbia.edu

I. Overview

In this project, we are using Solarflare AoE board to accelerate decoding of FIX/FAST, a protocol used for efficiently compressing market data, and build a book to record such data. Thanks to this high-speed board, we can implement such functionality on hardware that is much faster than software. By decoding the raw data sending from MAC, we use the predefined templates with default values and delta-type operations that rely on previous values from other entries to get a much more compressed format. After that, we would build a book to store such information and update the book by three types of command such as update, insert, and delete. The book reports summarized order quantities and order counts at a given price level. The depth represents the number of price levels that are supported in the feed. As Figure 1 shows, the view can be represented as a number of rows in a table for each of the bid and ask sides. On each side, there are a number of rows showing the quantity available at a number of price levels. An aggregate depth book is sequenced by price, descending for bid and ascending for ask. Finally, the book is transmitted as a new UDP packet to the PC in which the AoE is installed.

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	100	9427.50	9428.00	40	2
19	500	9427.00	9428.50	600	35
34	750	9426.50	9429.00	850	55
25	400	9426.00	9429.50	350	21
14	300	9425.50	9430.00	150	12

Figure 1 5-Deep - Best Bid/Ask

II. Background

The Financial Information eXchange (FIX) Protocol is a series of messaging specifications for the electronic communication of trade-related messages. Because of the huge increase in volume of data transferred in today's markets, the FIX Adapted for STreaming (FAST) Protocol has been developed as part of the FIX Market Data Optimization Working Group. FIX messages, like any self-describing message syntax, have a relatively high overhead of message descriptor. The FAST Protocol is a way of eliminating this overhead by exchanging the message description separately from the message. For example, in traditional FIX messages each field takes the form "Tag=Value<SOH>", FAST eliminates redundancy with a template that describes the message structure. This technique is known as implicit tagging as the FIX tags become implicit in the data.

III. Hardware

A packet consists of IP header, UDP header and UDP payload. Each of these components consists of various length of 8-bit FLIT. This design handles ONE FLIT at a clock cycle. Thus, based on the length of the packet, it takes various numbers of cycles to process an order. However, the overall throughput is 1 flit per clock cycle, as long as the pipe is filled.

There are five cascaded blocks in this hardware design, which is a pipeline structure. Each pipe stage consists of 1 or 2 sub-stage. Please Refer the Figure 2 for details.

1 Ethernet Driver Block

This block is a hardware diver for the Ethernet port. As this block may be provided by the manufacturer, we do not need to build it. However, we require this block to meet certain specification.
1) Buffer a packet one it is received.
2) Put the next flit on 8 bit output every clock cycle once selected
3) Send a "END" signal once the packet is finished

2 Packet Processing Block

There are three units in this block, but only one of them, "Packetizer", is one the pipeline. Thus, this block consumes one clock cycle.

Channel select unit basically provides select signal for 2 input Ethernet ports. When a channel 1 is selected, an ACK signal is asserted for Driver 1 and 0 for channel 2.

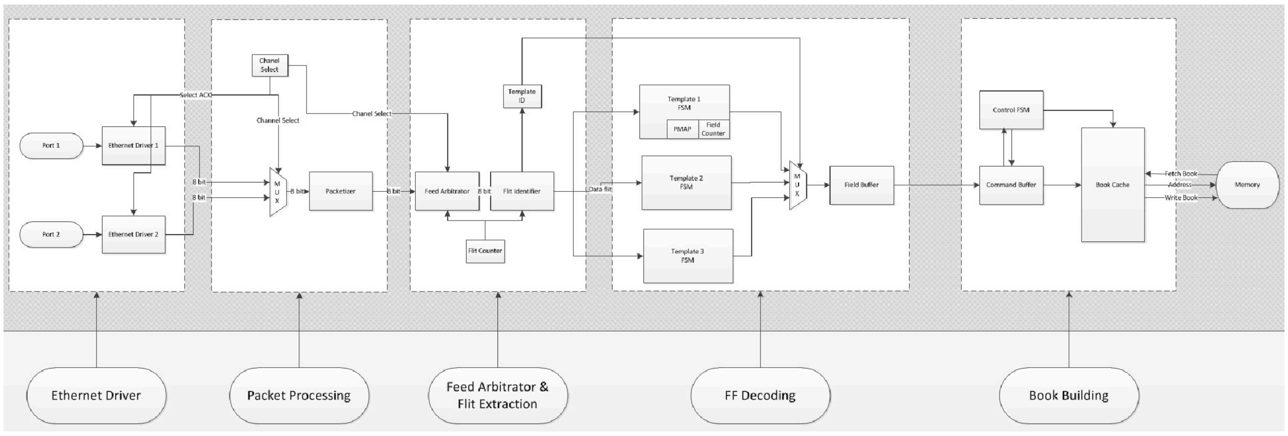


Figure 2 high-level architecture

Meanwhile, the select signal is also given to mux and Feed Arbitrator to make correct decision.

Packetizer is the main function unit in this Block. It rips always the IP header and UDP header, only passes along the IP serial number and UDP payload. It also calculates the checksum and tosses the flit always in case the checksum is bad.

3 Feed Arbitration and Flit Extraction Block

Feed Arbitrator Unit and Flit identifier unit is on the main pipeline. Thus, this block consumes two clock cycles.

Feed Arbitrator unit is to compare the IP serial number with previous serial number of the same channel. If the serial number is added exactly by one, which indicates correct, the flit is passed along. Else, some error signal is asserted.

Flit identifier unit is used to identify the ROLE of each flit. For example, 1st to 42nd flit is IP header and UDP header. 20th flit is IP serial number. 43rd flit should be PMAP. 44th flit should be Template ID. After 44th, all flits are real field data. This block can be used to tell Template FSM what type of flit they are receiving. However, it is possible to integrate this block inside the FSM. Thus, the necessarily of existence of this block is TBD.

Flit counter unit counts the flit received and input the count to Flit Identifier.

4 FF Decoding Unit

Feed Arbitrator Unit and Flit identifier unit is on the main pipeline. Thus, this block consumes two clock cycles.

Feed Arbitrator unit is to compare the IP serial number with previous serial number of the same channel. If the serial number is added exactly by one, which indicates correct, the flit is passed along. Else, some error signal is asserted.

Flit identifier unit is used to identify the ROLE of each flit. For example, 1st to 42nd flit is IP header and UDP header. 20th flit is IP serial number. 43rd flit should be PMAP. 44th flit should be Template ID. After 44th, all flits are real field data. This block can be used to tell Template FSM what type of flit they are receiving. However, it is possible to integrate this block inside the FSM. Thus, the necessity of existence of this block is TBD.

Flit counter unit counts the flit received and input the count to Flit Identifier.

5 BOOK Building Unit

Blocks before BOOK Building Block are like "Instruction Fetch & Decode unit" in a CPU. Book Building Block is the "Execution Unit" in a CPU.

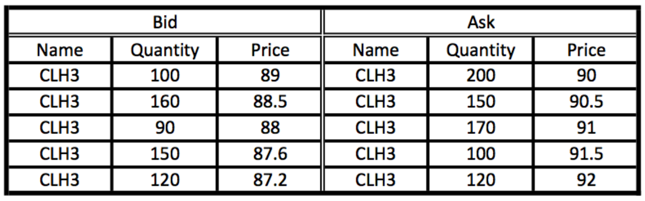
There are three major units in this block.

The command buffer is where decoded instructions from previous Template FSM are stored. Basically, this buffer can be a FIFO waiting for Control FSM to execute its content one by one.

Control FSM is control unit that regulates the operation inside the block. It first looks at the command buffer, then based on the property of the command, tells Book Cache to either Update, Delete, Insert, Fetch or Write Back. Meanwhile, it must provide accurate address for memory operation.

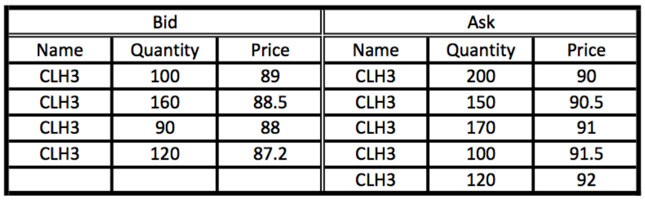
Book Cache is the memory where one can store whole book for a particular symbol. Meanwhile, it can operate the commands from Control FSM. For example, if you want to make change to MICROSOFT, the "Microsoft book" will be fetched from memory. Then the control FSM will tell Book cache to update certain field. When finished, "Microsoft book" will be written back to memory.

Here is an example as shown in following figures.



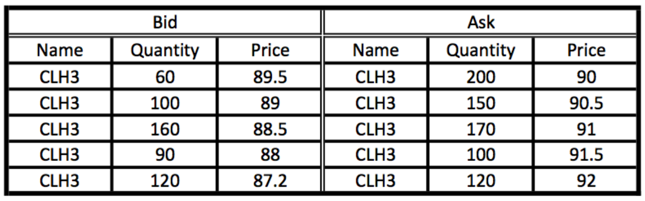
Bid			Ask		
Name	Quantity	Price	Name	Quantity	Price
CLH3	100	89	CLH3	200	90
CLH3	160	88.5	CLH3	150	90.5
CLH3	90	88	CLH3	170	91
CLH3	150	87.6	CLH3	100	91.5
CLH3	120	87.2	CLH3	120	92

Delete Level 4 of Bid of CLH3



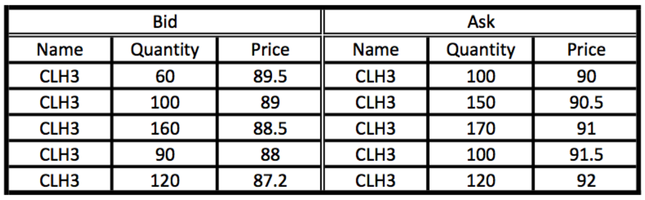
Bid			Ask		
Name	Quantity	Price	Name	Quantity	Price
CLH3	100	89	CLH3	200	90
CLH3	160	88.5	CLH3	150	90.5
CLH3	90	88	CLH3	170	91
CLH3	120	87.2	CLH3	100	91.5
			CLH3	120	92

Insert Bid of CH3 with a Prize of $89.5, 60 slots



Bid			Ask		
Name	Quantity	Price	Name	Quantity	Price
CLH3	60	89.5	CLH3	200	90
CLH3	100	89	CLH3	150	90.5
CLH3	160	88.5	CLH3	170	91
CLH3	90	88	CLH3	100	91.5
CLH3	120	87.2	CLH3	120	92

Decrease Level 1 of Ask of CLH3 with a Price of $90, 100 slots



Bid			Ask		
Name	Quantity	Price	Name	Quantity	Price
CLH3	60	89.5	CLH3	100	90
CLH3	100	89	CLH3	150	90.5
CLH3	160	88.5	CLH3	170	91
CLH3	90	88	CLH3	100	91.5
CLH3	120	87.2	CLH3	120	92

IV. Software

The software being written for validation would build books based on different messages(which are in fact, queries): modify, insert and delete. These messages are fed into the validation software in a similar fashion as that to the VHDL modules so that there will be a consistency between the hardware and software-based book-building operations and hence the validation turning out to be correct.

The software thus built would be targeted for an x86 machine with an AoE card in it. And this machine(PC) would be receiving UDP packets wherein each packet represent a single book structure which can contain a maximum of N entries where N is pre-determined one. More specifically, 'book snapshots'(all 10 levels for both bid/ask) will be sent to this x86 software over PCI-express bus. And the software will be able to read data off the FPGA using the APIs provided. In addition, a 'ticker plant' would also be built which would take the same 'book snapshot' and send it over the network as UDP packets. This 'ticker plant' approach would leverage the usage of the same UDP packets in both broadcasting out onto the network and also sending to the Solarflare ASIC which is then passed to the regular software running pn the PCIE listening for packets.

In addition, the validation software written in C will use one of the OpenSource software-based decoders capable of parsing the XML content(representing various templates identified by an ID which is the TemplateID for the respective template) and processing the different types of book-update messages. By using the parsed content, a VHDL module is generated which will be associated with the specific instruments and the corresponding information stored in terms of layout called Books. The software written would simultaneously track say 5-500 symbols to decide upon the templates in this regard.

The software written will be tested upon the decoded content given by the C++ decoder available at CME group website(http://www.cmegroup.com/globex/resources/fix-fast-decoder-agreement-thanks.html).

V. Responsibility

In this project, Danqing Hua and Junkang Ren are responsible for the hardware part and software part would be done by Chang Liu and Raghavan Santhanam.