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

Final Presentation

Danqing Hua, Junkang Ren, Chang Liu, Raghavan Santhanam
Outline

- Overview of Fix-fast protocol
- Overview of book builder.
- Work flow of the entire project.
- Software design
- Hardware design
- Demo on AOE board.
1. What is Fix-Fast?
   - FIX for financial information exchange.
   - It’s a series of messaging specifications for electronic communication of trade related messages.

2. What’s the protocol like?

   - Ethernet Packet Header
   - IP Packet Header
   - UDP Packet Header
   - UDP Payload
   - Msg Seq Number (4 bytes)
   - Subchannel ID
   - Presences Map (Fix-Fast Data)
   - Rest of Fix-Fast Data
3. How to decode fix-fast message?

- XML templates
- Presence Map (PMap)
- Big Endian
- Most significant bit serve as indicator of stop byte.

Heximal:
01 44 49 82

Decimal:
0000 0001 0100 0100 0100 1001 1000 0010

After Decoding:
000 0001 100 0100 100 1001 000 0010
Book Builder

- What is Book in trading?
  - Records of bid and ask information in trading activity

- Important variables related to book builder
  - MDEntryType: Decides whether we are working on book with bidding information or on book with asking information. (0 for bid, 1 for ask)
  - MDUpdateAction: 3 actions in total. “0” means add a new level (item) in the book; “1” means modify a certain level in the book; “2” means delete an existing level in the book.
  - MDPriceLevel: Decides which item of the book we are working on.
  - MDEntryPx: Price of a stock
  - MDEntrySize: The amount of a certain stock
  - NumberOfOrders
Work Flow of The Project

- The flow on the top is software validation
- The flow on the bottom is hardware implementation
Software Design

1. Validation:
   - Decoder: Implementation of the template decoder in C.
     - Parse the output file of csv format from Decoder
     - Generate the book as two separate list (one for bid, one for ask)
     - Output the snapshot of book through Ethernet packet. Packed all the packets into pcap format.

2. Software Support:
   - Using Perl to generate VHDL testbench for book builder.
   - Parsing the XML templates using Python to auto generate decoder for all templates.
   - Python and Shell script to compare output from decoder and golden output in simulation.
   - Tcl script to help compile run VHDL simulation in one command.
Packetizer Module
- passes along the UDP payload

Packetizer_fifo Module
- convert data to a 8-bits flit
- store the data which hasn’t been processed
Hardware Design

- Template Decoder Module

Diagram showing the flow of data through the Template Decoder Module, including states such as `Start`, `Pmap`, `TemplateID`, `SeqNum`, `Sending Time`, `PosDupFlag`, `End_of_packet = 1`, `count < 5`, `Entry Time`, `Entry Type`, `Price Level`, `Update Action`, `NoMD Entries`, `Trade Date`, `Security ID`, `RptSeq`, `Entry Px_base`, `Entry Px_m`, `Entry Size`, `Number OfOrders`, `Trading SessionID`, and conditions such as `flit_in(7) = '1'`, `flit_out(10 downto 8) <= '110'`, `flit_out(10 downto 8) <= '010'`, `flit_out(10 downto 8) <= '001'`, and `flit_out(10 downto 8) <= '000'`. The diagram illustrates the transition between these states and conditions, indicating how data is processed and routed through the module.
Hardware Design

- Command Buffer Module

- combines specified information to a piece of command
Hardware Design

- Book Builder Module

Diagram:
- Memory controller
- RAM
- Book
- Output
Hardware Design

- Book Builder Module

<table>
<thead>
<tr>
<th>Source State</th>
<th>Destination State</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>execute</td>
<td>writeback</td>
<td>(command_status[0]).(command_status[1]).(!reset)</td>
</tr>
<tr>
<td>execute</td>
<td>idle</td>
<td>(reset)</td>
</tr>
<tr>
<td>execute</td>
<td>execute</td>
<td>(!command_status[0]).(!reset) + (command_status[0]).(!command_status[1]).(!reset)</td>
</tr>
<tr>
<td>fetch</td>
<td>idle</td>
<td>(reset)</td>
</tr>
<tr>
<td>fetch</td>
<td>execute</td>
<td>(!reset)</td>
</tr>
<tr>
<td>idle</td>
<td>idle</td>
<td>(!command_status[1]) + (command_status[1]).(reset)</td>
</tr>
<tr>
<td>idle</td>
<td>fetch</td>
<td>(command_status[1]).(!reset)</td>
</tr>
<tr>
<td>writeback</td>
<td>idle</td>
<td>(command_status[0]).(!reset)</td>
</tr>
</tbody>
</table>
Hardware Design

- Output Module

- Output the snapshot of the book with whole book information

- Equip with IP header and UDP header
Verification of Functionality
Verification

- Runtime Verification: Signal Tap