Risk Management System

based on FIX protocal

CSEE 4840 Project Design

Kaixi Ji (kj2330)

Modi Yan (my2408)

1. Introduction

Financial Information eXchange (FIX) protocol is intended for information security transactions in markets. FIX session is layered on TCP and is used as the standard electronic protocol between the buy side (institution) and sell side (brokers/dealers) of the financial markets, as shown in Figure 1.1. It has become the global de facto messaging standard for pre-trade communication and trade execution.

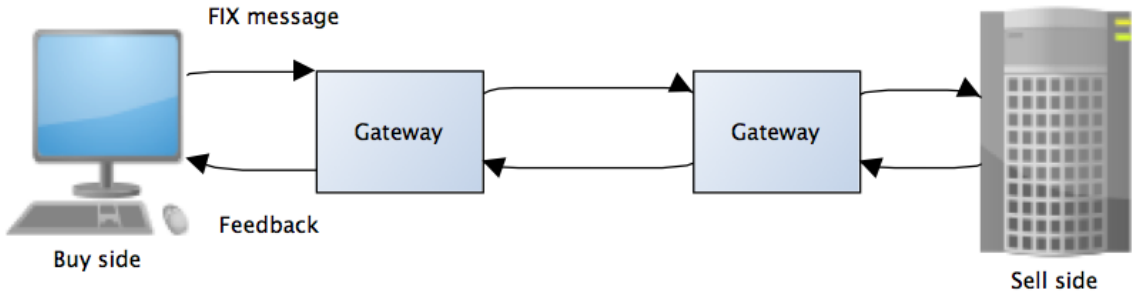

Figure 1.1

But as "disorder" might happen in buy side's system occasionally, there are chances that the institution sends out the unintended FIX message through its gateway to the broker and loses fortune in just a millisecond. In our design, we add a risk management system before the buy side's gateway to check the FIX message and drop the unintended message before sending it through. We may also check the feedback message from the sell side as well. The new information exchange system is shown in Figure 1.2.

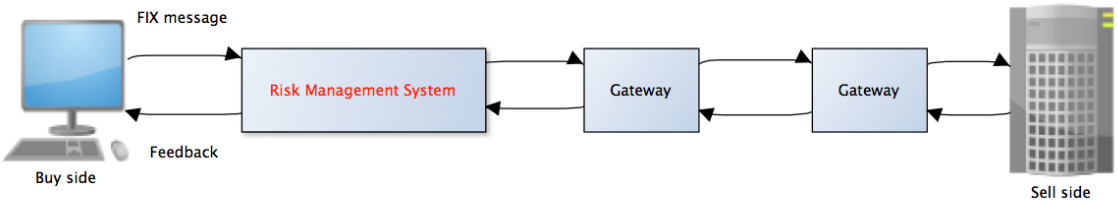

Figure 1.2

As a part of the risk management system, our project intends to unpack and decode the FIX message from the buy side, read and check the information it contains and then make the decision of whether to send it through the gateway or not. We will explain the implementation of the project briefly in the following sections.

2. Design

We use the SFP+ transceiver to receive message from the institution and send the message to our board through its XFI interface. Our design uses the SolarFlare AOE board. Hardware design block is shown in Figure 2.1.

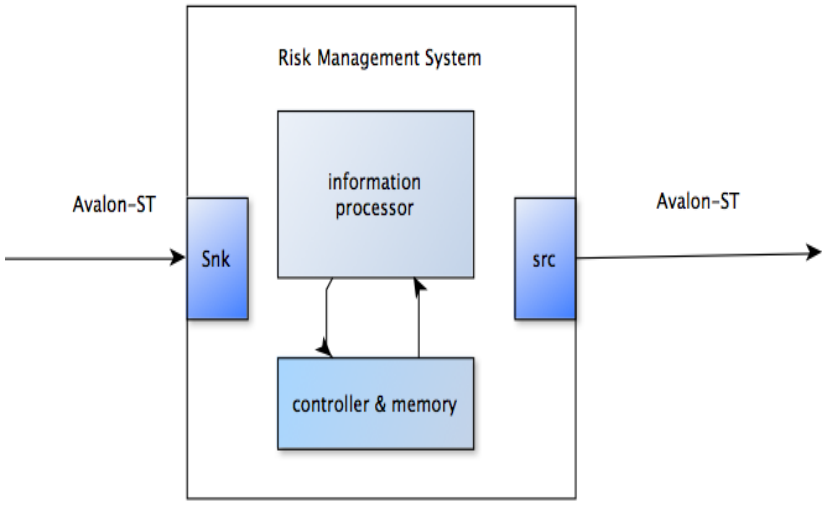

Figure 2.1

We use Avalon-ST to control data flow in the system. Packet comes into the information processor where we extract Tag and Value information and store the relative information that we are interested in. Controller then decides whether the current information is useful and does further processing to make the decision of dropping the packet or not. Packet that passes the risk management system goes to gateway of the institution.

In the information processor that is shown in Figure 2.2, packetizer receives packet at the speed of 8 bytes/clk and then strips the TCP and IP header off it. The packetizer then passes FIX message to the shift register also at the speed of 8 bytes/clk. If the fix parser sets the input ready signal to 1, shift register passes message to the next stage at 1 byte/clk. This means that incoming messages in a packet are going to accumulate in the shift register and wait to be processed. As each packet is expected to be no more than 300 bytes in length, thus takes up about 2K memory space, and considering the relatively slow rate of FIX packets coming in, a shift register of 10K should be enough.

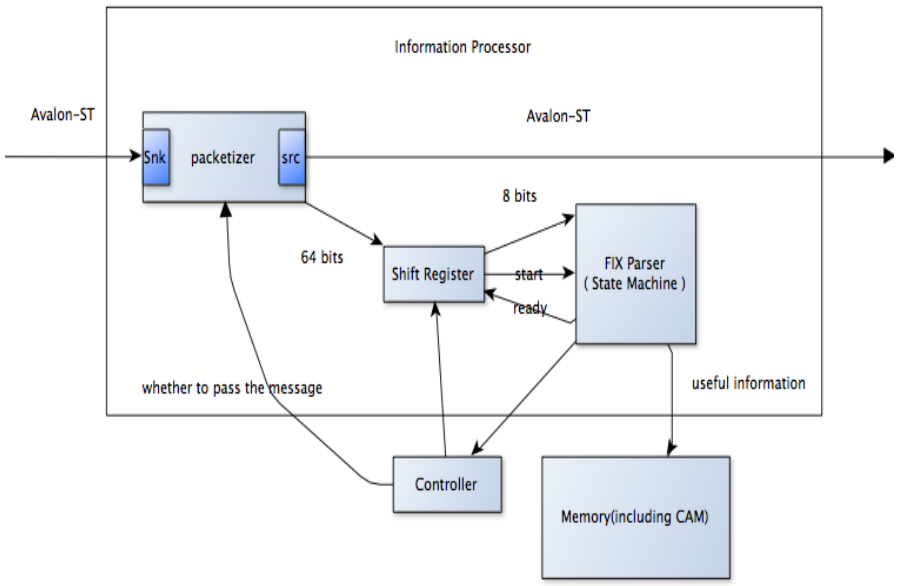

Figure 2.2

The state machine (Figure 2.3) is responsible for extracting information from each field of the FIX message. In FIX message, fields are separated by "|" (01), and Tag and Value in a field are separated by "=" (3D). So we wait for the input from shift register to be "head of packet" or "01" to decide whether we are receiving message from a new field. Once we know that we are receiving message from a new field, we put the received Tag message into a register. Tag message ends when we receive a "3D", then we know that the Value message begins. As some of the Value messages are pretty long, such as SendingTime (eg. 20140325-16:22:19.047), we may need a FIFO to store the Value message that we get. At the end of the Value message, we are expected to get another "01" signal and start to receive message from the next field, but we must wait for the ready signal from the interrupt stage. Interrupt stage receives message from Tag register and Value FIFO and then connect to memory structure and the controller for further operation. After interrupt stage has received Tag and Value message successfully, it should give the start stage a ready signal so that we can empty the Tag register and Value FIFO and then read message from the next field.

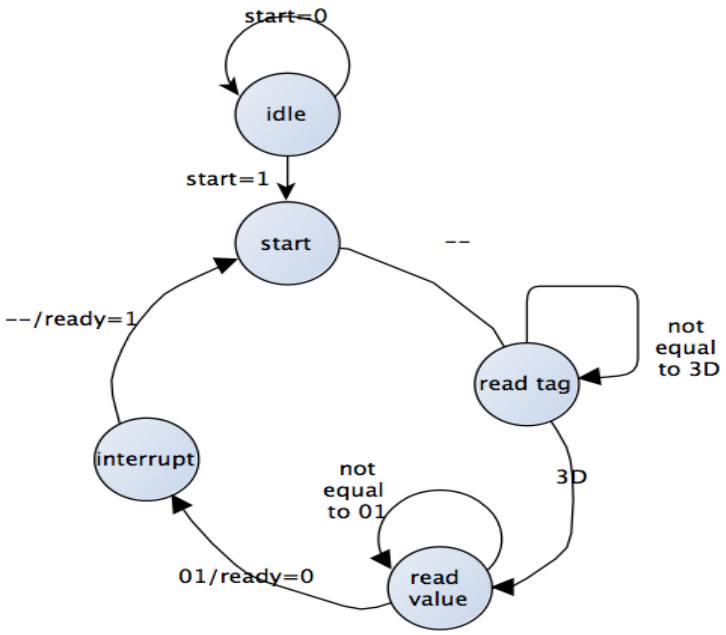

Figure 2.3

3. Using Scenario

In our project, we first concern about the buy side's ID, sell side's ID, transaction time, and the total value of transactions. That is to say, we need to care about packets that contain the information of logon and every single trade. In FIX message, that corresponds to the tag 35 (MsgType) , 49 (SenderCompID), 56 (TargetCompID), 52 (SendingTime), 38 (OrderQty) and so on.

For example for Logon packet (35=A), if we read Tag = 35 and Value = A, interrupt stage identifies the packet as a Logon packet, and our controller starts to wait for message containing the two ID (tag 49 and 56) and the time stamp (tag 52).

Another example is for NewOrderSingle packet (35=D), if we read Tag = 35 and Value = D, interrupt stage identifies the packet as a NewOrderSingle packet, and our controller starts to wait for message containing the two ID (tag 49 and 56), the time stamp (tag 52), and the price (tag 38). By tracking the NewOrderSingle packets during a specific period of time and putting a limit on the total value of the packets, the controller decides to drop the packet once the total value exceeds the limit.

Here is an example of why this Risk Manage System would be useful: Buyer A plans to spend one million dollars buying stocks from company B by buying 10 thousand shares each time. Because the price of stocks fluctuates quickly and there may be difficulty for the buyer's trading system to keep track of the price consistently, the money spent by A may exceed far from the expected amount. By using the Risk Management System, we keep track of every packet the buy side is going to send and drop the packet once the total amount of money spent on the given company's stocks exceeds the limited value.

4. Milestones

Milestone 1:

Finish SystemVerilog code for the information processor and part of the controller, and verify code function in Modelsim.

Milestone 2:

Implement the whole Risk Management System on board for a simple using scenario.

Milestone 3:

Come up with multiple scenarios in the stock market and use those scenarios to refine the system.

Reference:

[1] *Financial Information eXchange*,
http://en.wikipedia.org/wiki/Financial_information_exchange;
[2] *Avalon Interface Specifications*,
http://www.altera.com/literature/manual/mnl_avalon_spec.pdf.