Market Data Services

Depth of Market Feed v2.0 – Programmer's Guide

For use with FIX software version 4.4

Issue date: October 3, 2008

Produced by:
International Securities Exchange, Inc.
60 Broad Street, New York NY 10004
www.ise.com

Table of Contents

About This Document

This document is a programmer's reference guide intended to aid in the development and integration of the ISE Depth of Market Feed. It is intended to cover the general business behavior of the market data feed and the technology standards and techniques employed to provide this service. **The most recent version is available at www.ise.com/depth.**

Intended Audience

This document is for programmers, analysts and IT managers who are developing ISE Depth of Market Feed client applications.

Related Documents

The ISE Depth of Market Feed makes use of the FAST protocol (FIX Adapted for STreaming data). Users without prior knowledge of FAST should review these documents.

Document	Description	Location
FAST Technical Overview	Explains in detail how FAST successfully presents a solution to the problem of spiraling market data volumes.	http://fixprotocol.org/documents/2801/FIX%20Adapted%20for%20STreaming%20-%20FAST%20Protocol.pdf
FAST Protocol Specification v1.1	Defines the structure and semantics of FAST	http://www.fixprotocol.org/documents/3066/FAST%20Specification%201%20x%201.pdf
Transfer Encoding Specification v1.01	Describes the serialization process used to reduce the size of a data stream	http://www.fixprotocol.org/documents/3062/FAST%2520Transfer%2520Encoding%2520Specification%25201.0.2.pdf
Field Encoding Specification v1.0	Describes field-level operations used to reduce redundant information	http://www.fixprotocol.org/documents/3063/FAST%2520Field%2520Encoding%2520Specification%25201.0.pdf
Basic FAST Users Guide	Describes the proper use of the FAST Protocol in a one-way exchange of data	http://fixprotocol.org/documents/2301/A%20Basic%20Guide%20to%20FAST%20v1.0.pdf
FIX Protocol Version 4.4 Recommended Book Management Practices	FIX Technical Specifications Volume 3 (Pre Trade)	www.fixprotocol.org/specifications

The following are related documents to the ISE Depth of Market Feed.

Document	Description	Location
ISE Depth Feed Quick Start Guide	Overview details and contacts for those involved with the implementation of the feed	www.ise.com/depth
ISE Depth FAST Demo Application & Installation Guide	Open Source Demonstration Application and Installation Instructions	www.ise.com/depth
ISE Technical Bulletin – Depth Feed v2.0 Release Notes	Release notes for new version of Depth Feed for May 2008	www.ise.com/depth

1 Introduction

1.1 About ISE Depth of Market

The ISE Depth of Market feed provides price and aggregated volume information for the ISE options order book. The data feed includes information such as bid/ask quote and total quantity and customer quantity for the top five price levels. The number of price levels of aggregated quotes and orders may increase at some point in the future.

The feed does not contain trade data or information from other options exchanges. The top price-level or BBO contains the identical quotes that are provided in the ISE BBO feed to OPRA, with the exception of customer quantity. The Depth of Market feed may also be used to extract top of the book information for ISE instruments.

This broadcast market data service is based on industry and technology standards including Financial Information eXchange (FIX) protocol for business level messaging, FIX Adapted for STreaming (FAST) protocol for efficient network data distribution and representation, and UDP and IPv4 standards for transmission of broadcast data dissemination. The ISE Depth Feed is currently distributed by BT Radianz via the Radianz Shared Market Infrastructure, by NYSE Euronext Advanced Trading Solutions Secure Financial Transaction Infrastructure (SFTI®) and by SAVVIS Financial Services. Members may use their existing connections to access this data but their routers may need to be upgraded to support multicast data.

The ISE provides an open source utility called the FAST Demo Application that will read and decode the FAST messages and write them to a file. This is a free tool that can be used as a guide for developing a subscriber's own decoder. The decoder does not need to be a separate stand alone application. It could be integrated directly into the subscriber's application so it decodes the messages directly off the network, which we expect would be more efficient than first translating the data to FIX (ASCII) prior to processing. Due to firewall issues, the FAST Demo Application can be requested by sending an email to depth@ise.com or calling (212)897-8160.

1.2 Overview of Depth Feed

The ISE Depth feed has three message types. A Security Status message provides quote condition information, an Incremental Refresh message provides intraday updates to the order book, and Full Refresh message provides a snapshot of the order book.

The fields provided for each instrument are: OPRA symbol, CFI code (Call/Put), expiration date, underlying security symbol, underlying number, series number, trading status, and five levels of depth. At each level you are provided with total quantity, customer quantity, price, side, and price-level. **Customer Quantity** is a new field that will show aggregate customer bid/ask size at each price level. Quote Condition may appear in the template but is no longer used in this version.

The data feed will commence each day with a Full Refresh message for each series, setting the state to Pre-Open. Prior to the market open, the Full Refresh message will not have any price information but will provide details about each series i.e. expiration date. Once the market opens, Incremental Refresh messages will be sent to update the order book. The Full Refresh messages will continue to cycle through the order book every two minutes throughout the day and reflect the state of the order book as at the last Incremental Refresh message.

The market data is distributed over 16 multicast addresses and the ISE will proactively balance the load across the 16 channels. Each symbol is assigned to one of the 16 channels and can change channels from day to day, but will not change channels within a trading day. The data feed is being multicast over various networks in an A-feed / B-feed format, similar to OPRA.

If you are late to join the data feed or a packet is lost, you must process a complete cycle of the Full Refresh messages to ensure that the order book data is accurate. The full refresh or complete rotation of the order book for all series will take two minutes and only consumes approximately five percent of the bandwidth. Once you process a complete Full Refresh you can process Incremental Refresh messages for that series. It is not possible to request retransmissions.

The messages are defined using the FIX.4.4 standard for market data, and follow the best practices outlined by the FIX Market Data Working Group. The data is transmitted in the FAST v1.1 encoding method to minimize bandwidth and reduce latency.

For more information about the FIX and FAST Protocol, please refer to the following Web sites: http://fixprotocol.org/specification and http://fixprotocol.org/fast.

1.3 Summary of Changes to v2.0

The full release notes are available in the Documentation section at www.ise.com/depth. The following is a summary of the changes and the expected benefits.

#	Change	Description	Benefit
1	**Remove Instrument Details from IncrementalRefresh and Security Status**	The following fields have been removed from the Incremental Refresh and Security Status messages: • Symbol • CFI Code • MaturityMonthYear • Strike Price • SecurityDesc These fields have been replaced with the following fields: • UnderlyingNumber • Series Number	The fields being removed are redundant across the message types. This will reduce the size and processing time of the Incremental and Status messages. We also expect this to reduce bandwidth requirements since Incremental Refresh messages are approximately 95% of the traffic.
2	**Alter Startup Sequence**	During the start-up the Security Status message will no longer be distributed. All relevant series information can be obtained by referring to the Full Refresh message.	Removing the Security Status message reduces the amount of data.
3	**Change to Sending Time in of all three messages**	The Sending Time value within the common header of each message will be changed to SendingTimeJavaEpoch. This time is in milli-seconds from 1/1/1970 which is standard Java time format.	This standard routine replaces a string value which is more efficient and requires less processing.
4	**Headers are now consistent across all three messages**	All three messages have the same header: • BeginString • MsgType • SenderCompID • MsgSeqNum • SendingTimeJavaEpoch	Objective is to have consistent header across all three message types.
5	**Refresh Indicator and QtyCustomer FAST operator change**	RefreshIndicator and QuantityCustomer now have one FAST operator, which is "Default". The "Copy" operator has been removed	Removes ambiguity
6	**QuantityCustomer change**	The Presence 'Optional' has been removed from QuantityCustomer. The value sent in this message will then be the actual value, since there is no need to support a "null" value.	QuantityCustomer is new enhancement and is no longer an optional field.

1.4 Hours of Operation

Normal trading hours for the ISE Options market are from 9:30am to 4:00pm ET and 4:15pm ET for Index Options.

The ISE Depth of Market Feed will be brought online each day at approximately 6:00am ET and will continue to broadcast information throughout the trading day. The service will stop broadcasting at approximately 5:00pm ET.

Time	Depth Feed Activity
6:00am	The Depth of Market feed begins. Market Data Full Refresh messages with no price level information are sent. The Security Trading Status = 21 (pre-open) and will be sent for all series every two minutes. ISE will begin to queue orders. No price level information is sent on the Depth Feed until the market opens.
9:30am	The Market opens and regular trading begins. A Security Status message is sent as each series is opened with status=17 (ready to trade) Incremental Refresh messages are sent to update the book as the market changes and account for approximately 95% of the traffic. Full Refresh messages are sent continuously with a complete refresh of the market every two minutes. At any time, a Security Status message may be received with either a Halt (SecurityTradingStatus=2) or Fast Trading (SecurityTradingStatus=23). See complete list of SecurityTradingStatus message types on page 10.
4:00pm/ 4:15pm	Regular trading ends. A Security Status message is sent as each series is closed with status=21 (Pre-open) since there is no 'market closed' message. Full Refresh messages continue to be disseminated with end of day information.
5:00pm	The Depth of Market feed stops disseminating Full Refresh messages.

Table 1 - Normal Processing Schedule

1.5 Support and Connectivity

ISE support for the Depth Feed is available from 8 am to 6 pm (Eastern Time) on market days and the contacts are as follows:

ISE Contacts		
Business Issues	212 897-8160	depth@ise.com
Technical Support	212 897-0284	computeroperations@ise.com
API Support	212 897-0244, #1	tms@ise.com
Member Connectivity	212 897-0244, #3	connect@ise.com

The ISE Depth of Market Feed is currently distributed by **BT Radianz** via the Radianz Shared Market Infrastructure, by **NYSE Euronext Advanced Trading Solutions** Secure Financial Transaction Infrastructure (SFTI®) and by **SAVVIS Financial Services**. Members may use their existing connections to access this data but their routers may need to be upgraded to support multicast data. **We recommend a 50 MB redundant connection with an aggregate of 100MB of bandwidth**, which represents 100% head room based on the peak data rates as of January 2008 (see Section 5). The contacts for support and connectivity are as follows:

BT Radianz Inc.		
Edmond Esquilin	212 415 4664	edmond.esquilin@bt.com
Support	877 882-1497	

NYSE Euronext Advanced Trading Solutions - SFTI (Secure Financial Transaction Infrastructure)		
Sales	212 244-5551	sales@transacttools.com
Support	800 873-7422	SFTI@SIAC.com

SAVVIS Financial Services		
Sales	800 463-8294	teamise@savvis.net
Support	888 638-6771	

For additional information please send email to depth@ise.com.

2 ISE Depth of Book Message Types

This section describes the ISE Depth Feed, the different message types that are sent by ISE and how they are used to maintain the order book.

The ISE Depth of Market Feed provides subscribers with a view of the order book where all quotes and orders are aggregated at each price level, for the top five price levels. The data feed provides the quantity available at each price level, the quantity of customer orders at each price level (if present), and the trading status of the security. There is no trade information in this data feed.

An example of the ISE order book provided in the Depth of Book feed is as follows:

IBMJD						
Status – Ready to trade (Open)						
	Bid			**Ask**		
Level	**Price**	**Quantity**	**Customer**	**Price**	**Quantity**	**Customer**
1	0.98	20	10	1.00	50	10
2	0.97	30	0	1.01	30	10
3	0.96	10	5	1.03	10	5
4	0.94	80	40	1.05	10	10
5	0.93	10	5	1.08	10	1

The ISE Depth of Market Feed utilizes three FIX message types as listed in the table below. These three following message formats are used to denote five different types of events.

MsgType (tag=35)	Msg ID	Message Name	Notes
X	100	Market Data Incremental Refresh	The MDUpdateAction tag indicates the usage of the message type: ▪ MDUpdateAction = 0 - New Price Level ▪ MDUpdateAction = 1 - Change Price Level ▪ MDUpdateAction = 2 - Delete Price Level
W	500	Market Data Full Refresh	Snapshots of all price levels
f	400	Security Status	Trading Status of a Security

Table 2 - Market Data Messages

The message formats are based on the FIX.4.4 specification, with some extensions as noted in Section 4.

The ISE begins disseminating **Full Refresh** messages at 6:00am ET and the security trading status will be flagged as 'pre-open' and the Refresh Indicator will be set to '1 – to process.' At this point the Full Refresh will not have any price information and only contain static data for each series e.g. Symbol, Underlying Number, Series Number, Security Description, CFI Code, Maturity and Strike Price. There will be a single Full Refresh message for every series, once every two minutes. Once the complete rotation for all series has been sent, another rotation will begin and the Refresh Indictor will be set to "0 – not required to process.'

When the market opens at 9:30 am ET, a **Security Status** message will be disseminated for each series and the security trading status will be flagged 'opening.' As soon as each series begins to trade an **Incremental Refresh** messages will be disseminated for every quote change e.g. price or size and the security trading status will be flagged as 'ready to trade'. Each message will have one instruction (MDUpdateAction) that can add, change or delete a price level. **Each message contains one instruction that changes one side of one price level.**

The following is an example of five Incremental Refresh messages:

Add:	Price Level 1,	BUY	10 @ 0.98
Add:	Price Level 1,	SELL	50 @ 1.01; 20 Quantity Customer
Add:	Price Level 2,	BUY	30 @ 0.96
Chg:	Price Level 1,	BUY	90 @ 0.98
Del:	Price Level 2,	BUY	30 @ 0.96

Full Refresh messages are being continuously sent for every instrument and a complete rotation takes two minutes. It would be possible at market open to receive a **Security Status** message followed by a **Full Refresh** message before you receive an I**ncremental Refresh** if that series did not change the order book at the open. Full Refresh messages continue to be disseminated every two minutes, for each series, until approximately 5pm ET. The Full Refresh Snapshot message always reflects the state of the last Incremental Refresh message.

A Full Refresh message that followed the previous example would contain the following data:
Price Level 1, Buy, 90 @ 98
Price Level 1, Sell, 50 @ 1.01; 20 Quantity Customer

Full Refresh messages can be ignored unless the field 'Refresh Indicator' has the value '1'. This is used in the event of a system failure that requires the ISE to send snapshots for all instruments to refresh the order book or if a new series is added intraday.

The following sections go through each of the message types in more detail.

2.1 Full Refresh (Snapshot)

The term Snapshot and Full Refresh are interchangeable throughout this document. A Full Refresh message will provide a complete picture of the order book. This Snapshot message for every instrument is sent out every two minutes and it also serves as a heartbeat mechanism so that client applications can constantly check that they are receiving data.

An example of a Full Refresh sent prior to the open would appear as the following FIX message:

Tag Number		Tag Name	Value	Description
Standard FIX Header MsgType = "W"				
	55	Symbol	IBMJD	OPRA Symbol (Root, Month, Strike)
	107	SecurityDesc	IBM	Underlying Symbol
	5295	UnderlyingNumber	131	Number assigned by ISE between 1-32,767
	5296	Series Number	212	Number assigned by ISE between 1-65,635
	461	CFICode	OC	"OC" for Option Call "OP" for Option Put
	200	MaturityMonthYear	20081017	Expiration date
	202	StrikePrice	120	Strike price
	326	SecurityTradingStatus	21	21 = Pre-Open
	1200	RefreshIndicator	0	'0' or blank means you do not have to process; '1' means you do have to process
	268	NoMDEntries	0	Number of repeating entries

The information being conveyed would be represented as:

IBMJD						
Status – PreOpen						
Bid				**Ask**		
Level	**Price**	**Quantity**	**Customer**	**Price**	**Quantity**	**Customer**

Once the market opens, Incremental Refresh messages are sent to add, change, and delete price levels in the order book. The Snapshot messages continue to be sent every two minutes and will reflect the current state of the book as at the last Incremental Refresh message.

Full Refresh Example - An example of a Snapshot following the opening would be as follows:

Tag Number		Tag Name	Value	Description
Standard FIX Header MsgType = "W"				
	55	Symbol	IBMJD	OPRA Symbol (Root, Month, Strike)
	107	SecurityDesc	IBM	Underlying Symbol
	5295	UnderlyingNumber	131	Number assigned by ISE between 1-32,767
	5296	Series Number	212	Number assigned by ISE between 1-65,635
	461	CFICode	OC	Put or Call "OC" for Option Call "OP" for Option Put
	200	MaturityMonthYear	20081017	Expiration date
	202	StrikePrice	120	Strike price
	326	SecurityTradingStatus	17	21 = Pre-Open (or not trading) 22 = Opening 17 = Ready to trade 23 = Fast Market 2 = Trading Halt
	1200	RefreshIndicator	1	'0' or blank means you do not have

	Tag Number	Tag Name	Value	Description
Standard FIX Header MsgType = "W"				
				to process; '1' means you do have to process
	268	NoMDEntries	3	Number of repeating entries
>	269	MDEntryType	0	Bid (0=Bid, 1=Ask)
	270	MDEntryPx	0.98	Price
	271	MDEntrySize	20	Quantity
	1023	MDPriceLevel	1	Price Level 1
	9050	QuantityCustomer	10	Shows size of customer orders on the Bid or Ask
	276	QuoteCondition		This tag is no longer used
>	269	MDEntryType	0	Bid (0=Bid, 1=Ask)
	270	MDEntryPx	0.97	Price
	271	MDEntrySize	30	Quantity
	1023	MDPriceLevel	2	Price Level 2
>	269	MDEntryType	1	Ask (0=Bid, 1=Ask)
	1023	MDPriceLevel	1	Price Level 1
	270	MDEntryPx	1.00	Price
	271	MDEntrySize	50	Quantity

The information being conveyed would be represented as:

IBMJD						
Status – Ready to trade (Open)						
Bid				**Ask**		
Level	**Price**	**Quantity**	**Customer**	**Price**	**Quantity**	**Customer**
1	0.98	20	10	1.00	50	0
2	0.97	30	0			

2.2 Incremental Refresh - New Price Level

When a new price level is created in the order book, an Incremental Refresh, message Type X is sent with MDUpdateAction set to 0. This indicates that the new price level is to be inserted at the specified price level. All rows in the order book are to be pushed down and if there were already five price levels then the new sixth price level should be deleted.

The tag MDPriceLevel is used to identify which level is being added. If set to "1", it will update the top level. The subscriber's application should check that there are no prices higher than this price level and if they do exist then they should be deleted. In normal operations this should never happen.

Incremental Refresh Example 1 – A new order sets the top price level:
State of the book before a new order is entered:

IBMJD						
Status – Ready to trade (Open)						
Bid				**Ask**		
Level	**Price**	**Quantity**	**Customer**	**Price**	**Quantity**	**Customer**
1	0.97	30	15	1.00	50	0
2	0.94	80	0			

A new Customer buy order for 20 contracts is added creating a new top price level bid in IBM October 120 Calls. The order is priced at $0.98.

	Tag Number	Tag Name	Value	Description
Standard FIX Header MsgType = "X"				
	268	NoMDEntries	1	One Market Data entry
>	**279**	**MDUpdateAction**	**0**	**New**
	269	MDEntryType	0	Bid (0=Bid, 1=Ask)
	5295	UnderlyingNumber	131	Number assigned by ISE between 1-32,767
	5296	Series Number	212	Number assigned by ISE between 1-65,635
	270	MDEntryPx	0.98	Price
	271	MDEntrySize	20	Quantity
	1023	**MDPriceLevel**	**1**	**This is the key to inserting the order at the proper level**
	9050	QuantityCustomer	20	Shows size of customers orders on the Bid or Ask
	276	QuoteCondition		This tag is no longer used but is still in the template

State of the book after the order is entered:

IBMJD						
Status – Ready to trade (Open)						
Bid				**Ask**		
Level	**Price**	**Quantity**	**Customer**	**Price**	**Quantity**	**Customer**
1	0.98	20	20	1.00	50	0
2	0.97	30	15			
3	0.94	80	0			

Incremental Refresh Example 2 – A new order sets the new third price level:

This example shows an order arriving at price level three that pushes existing price levels down and the old price level 5 is to be discarded.

State of the book before the order is processed:

IBMJD						
Status – Ready to trade (Open)						
Bid				**Ask**		
Level	**Price**	**Quantity**	**Customer**	**Price**	**Quantity**	**Customer**
1	0.98	20	20	1.00	50	0
2	0.97	30	15			
3	0.94	80	0			
4	0.92	60	0			
5	0.90	50	50			

A new buy order for 10 contracts is added to the Order book in IBM October 120 Calls. The order is priced at $0.96 which is better then the current 3rd level. This results in a new price level 3 and the sixth price level is to be deleted by subscriber.

	Tag Number	Tag Name	Value	Description
Standard FIX Header MsgType = "X"				
	268	NoMDEntries	1	One MD entry
>	**279**	**MDUpdateAction**	**0**	**New**
	269	MDEntryType	0	Bid (0=Bid, 1=Ask)
	5295	UnderlyingNumber	131	Number assigned by ISE between 1-32,767
	5296	Series Number	212	Number assigned by ISE between 1-65,635
	270	**MDEntryPx**	**0.96**	**Price**
	271	MDEntrySize	10	Quantity
	1023	**MDPriceLevel**	**3**	**This is the key to inserting the price at the proper level**

State of the book after the order is entered:

IBMJD						
Status – Ready to trade (Open)						
Bid				**Ask**		
Level	**Price**	**Quantity**	**Customer**	**Price**	**Quantity**	**Customer**
1	0.98	20	20	1.00	50	0
2	0.97	30	15			
3	0.96	10	0			
4	0.94	80	0			
5	0.92	60	0			

2.3 Change Price Level

An Incremental Refresh Message with MDUpdateAction equal to "1" will indicate a change at a given price level and all fields on the specified side at the price level should be updated. This includes Price, Quantity and Customer Quantity.

Incremental Refresh Example 3 – One of the orders at the top price level is processed:

State of the book before the order is processed:

IBMJD						
Status – Ready to trade (Open)						
Bid				**Ask**		
Level	**Price**	**Quantity**	**Customer**	**Price**	**Quantity**	**Customer**
1	0.98	20	20	1.00	50	0
2	0.97	30	15			
3	0.96	10	0			
4	0.94	80	0			
5	0.92	60	0			

The quantity of an existing buy order in IBM October 120 Calls is reduced from 20 contracts to 10. The price remains at $0.98. The Price level remains Level 1 and the size is reduced.

	Tag Number	Tag Name	Value	Description
Standard FIX Header MsgType = "X"				
	268	NoMDEntries	1	One MD entry
>	**279**	**MDUpdateAction**	**1**	**Change**
	269	MDEntryType	0	Bid (0=Bid, 1=Ask)
	5295	UnderlyingNumber	131	Number assigned by ISE between 1-32,767
	5296	Series Number	212	Number assigned by ISE between 1-65,635
	270	MDEntryPx	0.98	Price
	271	**MDEntrySize**	**10**	**Quantity**
	1023	**MDPriceLevel**	**1**	
	9050	QuantityCustomer	10	Shows size of customer orders on the Bid or Ask

State of the book after the order is processed:

IBMJD						
Status – Ready to trade (Open)						
Bid				**Ask**		
Level	**Price**	**Quantity**	**Customer**	**Price**	**Quantity**	**Customer**
1	0.98	10	10	1.00	50	0
2	0.97	30	15			
3	0.96	10	0			
4	0.94	80	0			
5	0.92	60	0			

2.4 Delete Price Level

An Incremental Refresh with MDUpdateAction equal to "2" is used to indicate a price level delete.

Incremental Refresh Example 4 – All bid orders at the top price level are processed:

State of the book before the orders are processed:

IBMJD						
Status – Ready to trade (Open)						
Bid				**Ask**		
Level	**Price**	**Quantity**	**Customer**	**Price**	**Quantity**	**Customer**
1	0.98	10	10	1.00	50	0
2	0.97	30	15			
3	0.96	10	0			
4	0.94	80	0			
5	0.92	60	0			

Delete price level message comes in, specifying which price level to delete.

	Tag Number	Tag Name	Value	Description
Standard FIX Header MsgType = "X"				
	268	NoMDEntries	1	One MD entry
>	**279**	**MDUpdateAction**	**2 = Delete**	**Delete**
	269	MDEntryType	0	Bid (0=Bid, 1=Ask)
	5295	UnderlyingNumber	131	Number assigned by ISE
	5296	Series Number	212	Number assigned by ISE
	270	MDEntryPx	0.98	Price
	271	MDEntrySize	10	Quantity
	1023	**MDPriceLevel**	**1**	**Delete Level 1**

State of the book after the delete message is received (all price levels on the bid side are shifted one level up). Note that if there were further price levels, ISE will send out a new Add message to insert the new price level 5 into the order book.

IBMJD						
Status – Ready to trade (Open)						
Bid				**Ask**		
Level	**Price**	**Quantity**	**Customer**	**Price**	**Quantity**	**Customer**
1	0.97	30	15	1.00	50	0
2	0.96	10	0			
3	0.94	80	0			
4	0.92	60	0			

3 FAST Message Decoding

The ISE Depth of Market feed utilizes the FAST Protocol to achieve efficient compaction of the FIX Market Data messages.

FAST uses a template to describe each message but the FIX tags are not sent. The fields are transmitted in the sequence defined by the template and a presence map (PMAP) at the start of each message. The template indicates which fields are present in each message:

- Numeric fields are sent in binary format
- Price fields are sent in decimal format as two integers
- All fields, both string and binary are sent with 7-bit bytes, the eighth bit or stop bit, is used to indicate the end of a field

There are a number of operators that leverage previous field values so that repetitive and constant data is NOT transmitted to the subscriber.

On receipt of a UDP packet by the subscriber's application, the byte stream must be decoded. The subscriber can use a FIX FAST decoder to convert the data stream into standard FIX messages which can then be processed by subscriber's applications. However we do not anticipate anyone would want to first decode the data to FIX. Rather the subscriber will most likely read and decode the byte stream directly into an application which will avoid the translation of binary data into ASCII for FIX and then back into binary for the subscriber's application. An example of a decoder is provided with the Demo Application which will assist the subscriber to write their own decoder (See Section 6 for additional information).

A UDP packet will contain one or more FAST messages. The first message in each packet will be a FAST Reset (Message ID = 120) which will reset or clear the FAST cache. **Values are not cached across UDP packets.**

The ISE implementation of FAST utilizes the following FAST data types which are defined in the next section:

- decimal
- length
- string
- uInt32/uInt64

The ISE implementation of FAST utilizes the following FAST operations which are defined in Section 3.2:

- constant
- copy
- default
- increment
- tail
- optional

3.1 Decoding FAST Data

This is a summary of the steps required to decode the ISE Depth FAST data stream:

1. The Depth Feed is sent as UDP multicast blocks and the subscriber must process one block at a time.

2. Each UDP block consists of a number of messages and the maximum block size is 1,000 bytes.

3. Each message has the following format: Presence Map (PMAP), Template ID and Payload

4. If the PMAP does not have the bit set for the Template ID then use the same Template ID as the previous message. The first message in the UDP packet always has the Template ID.

5. The subscriber must get the map and the Template ID to decode the message.

6. Many fields use the previous values that are saved in the cache. If the PMAP is not set then, depending on the FAST operator for this field, use previous value from the cache.

7. FAST does not carry fields over from previous UDP packets. The first message in every block is a reset message, which instructs the decoder to clear the cache and start again.

8. When processing the sample log data, the beginning of each UDP packet can be found by searching for C0 F8, which is the reset message. The first message in every block is a reset message and this is the only time the reset message is sent.

Fields have four data types and appear as follows:

Strings: Strings are transmitted as ASCII data where the last byte in the string has bit-8 set (stop-bit). The decoder must process each byte until the stop-bit set is reached and then remove that stop bit before appending it to the end of the field. The resulting data is then saved in the FAST cache and then the field is returned.

Example: “ABC” is transmitted as 41 42 C3

uInt32/uInt64: Integers are transmitted as binary numbers and numbers are sent as a variable number of bytes, using only the bottom 7 bits of each byte. The last byte has bit 8 set. The decoder must use a variable of the specified size, then starting with zero, process each byte until the last byte is done. Shift the previous value 7 bits left and add the next byte, until the last byte is processed. The result is stored in the FAST Cache and returned. If the result is needed in ASCII format then the result is converted to ASCII and returned.

Numbers that will not exceed 32 bits, such as Quantity Customer, Underlying Number and Series Number are defined as uInt32 in template. Numbers that exceed 32 bits, such as SendingTimeJavaEpoch, are defined as uInt64 in template.

Example: 25,000 in binary is 110 0001 1010 1000
Put into 7-bit bytes and the stop bit is added to the last byte. This is sent as hex 01 43 A8 (stop bits indicated in **bold).**

Native Hex/Binary	FAST Hex/Binary
N/A	0x80 **1**0000000
0x00 0	0x81 **1**0000001
0x01 00000001	0x82 **1**0000010

Decimal: Prices are sent in decimal format. We only use Decimal for prices but it can be used for any floating point number. In FAST a decimal number is sent as two integers. The first integer is the exponent and the second is the mantissa and both fields are signed, as in positive or negative. Each integer has its own stop-bit but together they are treated as one field in the presence map.

The exponent has a sign bit. It is negative when you need fractions of a dollar i.e. $2.50 is sent as 25 x 10^{-1} and in Hex FAST it would appear as FF 99.

The mantissa also has a sign bit and it is negative when the value is negative. We do not have any negative decimal values at present but an example is -$3.50 which is sent as -35 x 10^{-1} and in Hex FAST it would appear as FF E3. We may in the future include prices for option strategies, such as spreads and straddles, which could have a negative price.

Example: $1.95 is sent as two integers: -2 and 195. This value is transmitted at FE 01 C3.
Exponent: After removing the stop bit, FE is 7E, which is -2 in twos complement.
Mantissa: 01 C3 is represented as 00 C3. Remove the stop bit and shift the top byte down, which equals 195. Note that both mantissa and exponent are sign fields. Bits 1-6 are used for the value, Bit 7 is a sign and Bit 8 is the stop bit. If the number is between 64-127 then a zero byte will be sent i.e. 65 = 81 00 C1.

Length is an integer that specifies the number of times that a repeating group occurs.

3.2 Decoding Field Operators

A FAST Decoder is required to process the ISE Depth feed. A standard FIX FAST decoder will convert the data feed to FIX messages or a decoder can be programmed directly into an application to return the messages with each field in text and binary format.

The decoder should use the following steps:

1. The templates specify the data type and sequence of fields in each message and the FAST operators for each field.
2. The presence map (Pmap) indicates which fields are present in the message.

3. **The decoder must save the last value of each field in a cache.** If the field is not present in a message then the decoder retrieves the previous value from the cache and uses it as specified by the operator to generate the field value.
4. Upon receipt of a FAST Reset message, the decoder must clear the cache and mark all fields as undefined.

Note that the cache stores the last value of each field regardless of which message type was previously used i.e.. fields are saved by name across message types. The Series Number is stored only once, regardless of message type.

The following is a list of the FAST operators.

CONSTANT: A field that has the constant operator is not present in the message and does not use a bit in the Pmap. The decoder must generate the value specified in the template. This operator is used for the following fields: SenderCompID, BeginString and MessageType.

COPY: A field that has the COPY operator is not present if it is the same value as the previous time it was sent. When present, the decoder uses the value and saves the value in the cache. When it is not present then the decoder must use the value from the cache. This operator is used when fields often have the same value such as Symbol, Description and CFI Code.

DEFAULT: A field that has the Default indicator is only present if its value is different then the value specified in the template. If the field is not sent then the decoder is to use the default value from the template. This operator is used for number of entries and Refresh Indicator which are often zero.

INCREMENT: The Increment operator is used when integers normally increment, such as sequence number. If the Pmap indicates that the field is present in the message then that value is to be used and saved in the cache. If the Pmap indicates that the field is not present in the message then the decoder should increment the value in the cache and use that value.

TAIL: This operator is used for fixed length fields where the end of the field often changes but the start remains the same. This operator is used for the field MaturityMonthYear. If the field is present in the message and the cache does not have a previous value saved, the field in the message is to be saved in the cache. **If the field is present and there is already a value in the cache, the field in the message is to be used to replace the equivalent number of bytes from the end of the value in the cache and the full value from the cache is used. If the field is not present in the message then the value from the cache is to be used**.

OPTIONAL: If fields in the template are defined as optional then it has nullable property. All nullable types are constructed in such a way that NULL is represented as 7-bit entity value where all bits are set to zero. This operator was used by Quote Condition but Quote Condition has been replaced with Quantity Customer so it is no longer used.

Repeating Groups: The first item in a repeating group is a Pmap that applies to that group. The Incremental Refresh and Full Refresh messages have repeating groups. The first Pmap in the message covers the fields up to and including the number of items. Each repeating group then has its own pmap to indicate which fields are present in that group.

3.3 FAST Reset Message

This message is used to reset or clear the data cache and it is the first message in each UDP packet.

Message ID = 120 FAST Reset
No fields

Table 3 - FAST Reset Template

Example: The following sample is a hex dump (in gray) with comments above each row (in yellow) of hex data.

```
     Prev packet                                   |reset | MAP |
1)  C8 81 84 09 80 56 42 54 41 CA 81 85 C0 F8 7F E0
```

Notes on FAST Reset example:

- Row 1: The reset message is the first message in each UDP packet and it is always followed by the presence map of the next message.

4 Message Formats

This section describes each message type in detail. Each message is described in its FIX format and then the FAST schema is provided. At the end of this section there is an example of the message in FAST format.

4.1 FIX Message Headers

Every message will have a standard FIX.4.4 header with the following tags:

Tag Number	Tag Name	Maximum Number of Bytes	Notes
8	BeginString	7	FIX.4.4 - Always first field in message

35	MsgType	2	Always second field in message (X, W, or f)
			X = Market Data Incremental Refresh
			W = Market Data Full Refresh
			f = Security Status

49	SenderCompID	0	ISE denotes the sender is the exchange
34	MsgSeqNum	4	A unique sequence number given to each message sent. **Each of the 16 channels will have its own separate set of sequence numbers that will update sequentially with each message.**
5297	SendingTime-JavaEpoch	8	The Sending Time value within the common header of each message is now SendingTimeJavaEpoch. This time is in milli-seconds from 1/1/1970 which is standard Java time format.

Table 4 - FIX Messages in Headers

4.2 Market Data Incremental Refresh Message

The Market Data Incremental Refresh message will contain the fields listed in the following table below. The repeating group is noted by the "**>**" in the far left hand column.

	Tag Number	Tag Name	Maximum Number of Bytes	Description
Standard FIX Header Msg Type = "X"				
	268	NoMDEntries	1	Always set to 1
>	279	MDUpdate Action	1	Type of Market Data update action: 0 = New 1 = Change 2 = Delete
	269	MDEntryType	1	Type of Market Data entry: 0 = Bid 1 = Ask
	5295	UnderlyingNumber	2	Number assigned by ISE between 1-32,767
	5296	Series Number	2	Number assigned by ISE between 1-65,535
	270	MDEntryPx	5	Price of the Market Data Entry. Decimal position is implied: last two digits are decimals.
	271	MDEntrySize	3	Order volume of the Market Data Entry. Not included for delete messages.
	1023	MDPriceLevel	1	This is the key to inserting the order at the proper level. This is an extension to the FIX.4.4 specification.
	9050	QuantityCustomer	3	Shows size of customer orders on the Bid or Ask
	276	QuoteCondition	0	This tag is no longer used but is still in the template

Table 5 - Market Data Incremental Refresh

4.2.1 Update Action

The Incremental Refresh message is used to maintain the Depth for each series. **Each message has only one item that can update one side of one price level but there can be many messages in a network packet.** The field MDUpdateAction specifies the type of the change.

MDUpdateAction (tag=279) value of 0 indicates a new price level to be inserted into the order book. Pre-existing price level data is expected to be shifted downwards. For example, if there is a new price level 2, the prior price level 2 will now be price level 3, the prior price level 3 will now be price level 4, and so on. The old price level 5 is deleted by the subscriber. **Subscriber applications should not maintain price level data beyond the maximum number of levels specified (currently 5).**

MDUpdateAction (tag=279) value of 1 indicates a change in the price level and the information in the new message should overwrite the information in the corresponding price level in the order book. A field value not included in the message indicates the field in the cache should be cleared or defaulted. **For example, if the current value of QuantityCustomer is 10 and the Incremental Refresh message does not include a new value for QuantityCustomer then the value will be removed from the order book.**

MDUpdateAction (tag=279) value of 2 indicates that the corresponding price level in the cache should be removed or deleted and any remaining lower price levels should be shifted upwards.

For example, if price level 1 is deleted, price level 2 should become price level 1 and price level 3 should become price level 2 and so on. ISE will disseminate a new price level 5 following the delete price level event if there are additional price levels in the book.

See Section 2 for the examples on how price level information should be maintained.

4.2.2 Incremental Refresh FAST Template

This table is the template for the Market Data Incremental Refresh message. It describes the data in the FAST message stream and specifies how the data is to be decoded.

Message ID = 100 Market Data Incremental Refresh					
Field Name		FIX Tag	Data Type	Field Encoding	Comments
BeginString		8	String	Constant	value = FIX.4.4
MsgType		35	String	Constant	value = X
SenderCompID		49	String	Constant	value = ISE
MsgSeqNum		34	uInt32	Increment	
SendingTimeJavaEpoch		5297	uInt64	Copy	This time is in msec from 1/1/1970 which is standard Java time format.
NoMDEntries		268	Length	Default	default value is 0
>	MDUpdateAction	279	String	Copy	
	MDEntryType	269	String	Copy	
	UnderlyingNumber	5295	uInt32	Copy	Number assigned by ISE
	SeriesNumber	5296	uInt32	Copy	Number assigned by ISE
	MDEntryPx	270	Decimal	Copy	
	MDEntrySize	271	uInt32	Copy	
	MDPriceLevel	1023	uInt32	Copy	
	QuantityCustomer	9050	uInt32	Default	
	QuoteCondition	276	String	Copy (Optional)	Field no longer used

Table 6 – Incremental Refresh FAST Template

4.2.3 Incremental Refresh Fields by Presence Map

The following table is used to find the fields in the FAST data stream based on the contents on the presence map. The example field values are shown with stop bit set.

Pmap	**Field**	**Values**
1	MessageID	E4 (100), Copy if not present
2	MsgSeqNum	
3	SendingTime	
4	NoMDEntries	81

Repeating Items

Pmap	**Field**	**Values**
1	MDUpdateAction	80=New, 81=Change, 82=delete
2	MDEntryType	80=Buy, 81=Sell
3	UnderlyingNumber	Number assigned by ISE
4	SeriesNumber	Number assigned by ISE
5	MDEntryPx	Current price, Exponent then mantissa
6	MDEntrySize	Current Size
7	MDPrice Level	1, 2, 3, 4, or 5
8	QuantityCustomer	Actual total of Customer bid/ask quantity
9	QuoteCondition	No longer used but still in the template

4.3 Market Data Full Refresh Message

The Market Data Full Refresh message is used to refresh the book and will contain the fields listed in the table below. The repeating group is noted by the "**>**" in the far left hand column.

	Tag Number	Tag Name	Maximum Number of Bytes	Description
Standard FIX Header Msg Type = "W"				
	55	*Standard FIX Header Msg Type = "X"*		
	55	Symbol	5	Full OPRA Symbol (Root, Month, Strike)
	5296	Series Number	2	Number assigned by ISE
	461	CFICode	2	Put or Call: "OC" for Option Call "OP" for Option Put
	200	MaturityMonthYear	8	Expiration date in format: YYYYMMDD
	202	StrikePrice	5	Strike price in decimals
	107	SecurityDesc	7	Underlying Security Symbol (range 1-65,535)
	5295	UnderlyingNumber	2	Number assigned by ISE (range 1-32,767)
	326	SecurityTradingStatus	1	Identifies the trading status applicable to the transaction. This is an extension to FIX.4.4 and the following values are used: 21 = Pre-Open 22 = Opening 17 = Ready to trade 23 = Fast Market 2 = Trading Halt At market close, the trading status goes back to Pre-Open.
	1200	RefreshIndicator	1	'0' or blank means you do not have to process; '1' means you do have to process
	268	NoMDEntries	1	Number of repeating entries
>	269	MDEntryType	1	Type of Market Data entry: 0 = Bid 1 = Ask
	270	MDEntryPx	5	Price of the Market Data Entry. Decimal position is implied: last two digits are decimals.
	271	MDEntrySize	3	Order volume of the Market Data Entry. Not included for delete messages.
	1023	MDPriceLevel	1	This is the key to inserting the order at the proper level.
	9050	QuantityCustomer	3	Shows size of customer order on the Bid or Ask
	276	QuoteCondition	0	Field no longer used

Table 7 - Market Data Full Refresh

4.3.1 Initial Market Data Full Refresh

Prior to opening, the ISE will disseminate Snapshot messages to indicate the full list of instruments that will be traded during the day and the SecurityTradingStatus will be '21' or pre-open. Once trading begins, Incremental Refresh messages will be used by the client to maintain the price level cache and the SecurityTradingStatus will be '17' or ready to trade. After trading has ceased, the Snapshot messages will continue to broadcast the last state of the book at the time of the suspension or official close and the SecurityTradingStatus will be '21' or pre-open since there is no indicator for 'market closed'.

The initial Market Data Snapshot messages generated prior to market open will have no price level information. These messages are intended to provide a list of instruments that will be updated across the 16 IP addresses (Section 5.3) for the trading day. Series from prior trading sessions that are not included in the initial start of day Full Refresh messages should be removed or aged from the subscribing applications.

If a subscriber is only interested in depth data for specific securities they should use the pre-opening Snapshot messages to identify which communications channel those securities will be trading on for the day. The subscriber will only have to listen to the channels that have the securities they are interested in processing if they do not want to process the entire feed.

The Snapshot message will always reflect the order book as at the previous Incremental Refresh message except when there has been a failure in the service or when there is a new symbol published during the day and the ISE is required to force a complete refresh of the book to all subscribers. In this case only, the tag 'RefreshIndicator' will be set to '1' which indicates that the subscriber should discard the contents in the order book and completely replace it with the contents of this Snapshot message.

4.3.2 Continuous Full Refresh (Recovery)

There is no need for subscribers to request a retransmission or a refresh since the ISE will re-disseminate all price levels of depth at regular intervals via the Full Refresh messages, allowing a user to recover the current status of the order book. This function is called Continuous Full Refresh (CFR). CFR will send a Snapshot message for every instrument, every two minutes (configurable by the ISE.) Under normal operation Snapshot messages will account for less than 5% of bandwidth.

Note - After the close, the CFR will continue to be disseminated with the information and market state reflected at the close of the market.

4.3.3 Market Data Full Refresh Template

This section will show the FAST Full Refresh template.

Message ID = 500 Market Data Full Refresh					
Field Name		FIX Tag	Data Type	Field Encoding	Comments
BeginString		8	String	Constant	value = FIX.4.4
MsgType		35	String	Constant	value = W
SenderCompID		49	String	Constant	value = ISE
MsgSeqNum		34	uInt32	Increment	
SendingTimeJavaEpoch		5297	uInt64	Copy	This time is in msec from 1/1/1970 which is standard Java time format.
Symbol		55	String	Copy	Full OPRA Symbol (root, month, strike)
SeriesNumber		5296	uInt32	Copy	Number assigned by ISE
CFICode		461	String	Copy	
MaturityMonthYear		200	String	Tail	
StrikePrice		202	Decimal	Copy	
SecurityDesc		107	String	Copy	Underlying symbol
UnderlyingNumber		5295	uInt32	Copy	Number assigned by ISE
SecurityTradingStatus		326	uInt32	Copy	
RefreshIndicator		1200	String	Default	'0' or blank means you do not have to process; '1' means you do have to process
NoMDEntries		268	Length	Default	Default value is 0
>	MDEntryType	269	String	Copy	
	MDEntryPx	270	Decimal	Copy	
	MDEntrySize	271	uInt32	Copy	
	MDPriceLevel	1023	uInt32	Copy	
	QuantityCustomer	9050	uInt32	Default	
	QuoteCondition	276	String	Copy	Field no longer used

Table 8 – Full Refresh FAST Template

4.3.4 Full Refresh Fields by Presence Map

The following table describes the fields as they appear in the presence map and the FAST message. Each repeating item has its own Pmap so there is one table for the main part of the message and another for each repeating item.

Pmap Bit	**Field**	**Values**
1	Message ID	500 = 03 F4
2	Message Seq Numb	
3	Sending Time	This time is in msec from 1/1/1970 which is standard Java time format.
4	Symbol	
5	SeriesNumber	Number assigned by ISE
6	CFI Code	OC (4F C3) or OP (4F D0)
7	MaturityMonthYear	Expiration date , eg 20070614 as a string
8	StrikePrice	Mantissa and exponent eg 82 84 = 400
9	SecurityDescription	Underlying Stock name
10	UnderlyingNumber	Number assigned by ISE
11	SecurityTradingStatus	21=pre-open (95) 17=open (91)
12	RefreshIndicator	'0' or blank means you do not have to process; '1' means you do have to process
13	NoMDEntries	Not present equals no price levels, 10 = all levels; Currently set to a maximum of 5 price levels

Presence map and fields for each repeating item.

Pmap Bit	Field	Values
1	MDEntryType	0=Bid, 1=Ask
2	MDEntryPx	Mantissa and exponent eg 82 84 = 400
3	MDEntrySize	Integer
4	MDPriceLevel	Integer 1 through 5
5	QuantityCustomer	Actual total of customer bid/ask quantity
6	QuoteCondition	Field no longer used

4.4 Security Status Messages

A Security Status messages will be sent when an instrument changes trading state and one message will be sent for each security.

4.4.1 Security Status FIX Message format

In addition to the fields in the standard header, the fields contained in the FIX Security Status message are listed in the table below.

Tag Number	Tag Name	Maximum Number of Bytes	Description
5295	UnderlyingNumber	2	Number assigned by ISE
5296	Series Number	2	Number assigned by ISE
326	SecurityTradingStatus	1	This tag identifies the trading status applicable to the transaction. This is an extension to the FIX.4.4 specification. The following values are used: 21 = Pre-Open or not trading 22 = Opening 17 = Ready to trade 23 = Fast Market 2 = Trading Halt

Table 9 - Security Status Message

4.4.2 Security Status FAST Template

Message ID = 400 Security Status				
Field Name	FIX Tag	Data Type	Field Encoding	Comments
BeginString	8	String	Constant	value = FIX.4.4
MsgType	35	String	Constant	value = f
SenderCompID	49	String	Constant	value = ISE
MsgSeqNum	34	uInt32	Increment	
SendingTimeJavaEpoch	5297	uInt64	Copy	This time is in msec from 1/1/1970 which is standard Java time format.
UnderlyingNumber	5295	uInt32	Copy	Number assigned by ISE between 1-32,767
Series Number	5296	uInt32	Copy	Number assigned by ISE between 1-65,535
SecurityTradingStatus	326	uInt32	Copy	

Table 10 – Security Status FAST Template

4.4.3 Security Status Fields by Presence Map

The following table describes the fields as they appear in the presence map and the FAST message.

Pmap Bit	Field	Values
1	MessageID	400 = 03 90
2	MsgSeqNum	
3	SendingTimeJavaEpoch	This time is in msec from 1/1/1970 which is standard Java time format.
4	UnderlyingNumber	Assigned by ISE
5	SeriesNumber	Assigned by ISE
6	SecurityTradingStatus	21=preopen (95) 17=open (91)

4.5 Examples of Each Message Type using Depth Feed Data

The examples on the next few pages show hex dumps of the different message types. There are comments (in yellow) above each row of hex data (in green). Below each row of hex data is the value of each field (in blue). A notes section follows each example.

You will find two files of sample data at www.ise.com/depth.

1) DepthSample_v2_pt1 (383k) – This data includes the initial full rotation of the first Snapshot messages sent at the beginning of the day.

2) DepthSample_v2_pt2 (317k) – This data consists of the raw log file used for Examples 1-4 on the following pages.

4.5.1 Example 1: Full Refresh Message Prior to Market Open

The actual data from the feed is as follows:

Full Refresh Message - Prior to market open:

```
  |Reset | Map |  ID  |Seq | Sendingtime       | symbol
1) C0 F8 7F FC 03 F4 81  23 05 7D 53 5D 93 41 50 43 51 D1
    120/      / 500/  1 /   11: 02: 15    /A  P  C  Q  Q
```

```
  |S# | CFI | maturity                     | strike |
2) C9 4F D0 32 30 30 38 30 35 31 B7 80 00 D5
  73/ O P / 2  0  0  8  0  5  1  7/  8 5
```

```
    SecDsc  | und# |st| RI| map | symbol        |S#| CFI
3) 41 50 C3 03 E2 95 B1 0F F4 4F 49 55 41 C9 BC 4F C3
    A P C / 4 8 2/21/ 1/    / O  I  U  A  I /60/ O C
```

```
    maturity    |strike| SecDsc    | und# |RI|map
4) 39 30 31 31 B7 80 AD 49 4E 54 D5 01 A2 B1 0F F4
   9  0  1  1  7/ 45 / I  N  T  U / 162 /1/
```

```
  Symbol     |S#| CFI  | maturity     | strike |
5)4E 54 4F D7 DD 4F D0 38 30 33 32 B2 FF 01 AF
  N  T  O  W/93/  OP  / 8  0  3  2  2/ 17.5/
```

```
  SecDsc| und#|RI|map|
6)4E D4 03 D6 B1 0F F4 xx xx xx xx xx xx xx xx xx
  N  T / 470 /1/
```

Notes:
Row 1: Map 7F FC means fields 1-12 are present
Row 1: Message ID 500 means this is a Snapshot / Full Refresh Message
Row 1: Sequence number = 81 = 1 which means first message of the day
Row 1: Time is GMT
Row 2: Series number starts at 1 within each underlying
Row 2: Strike price 80 00 D5 = 85 x 10^0
Row 3: SecDsc is underlying symbol which is APC and the underlying number for APC is 482
Row 3: Refresh Indicator of 1 means you must process this message
Row 3: Map 0F F4 means fields 4-10 and 12 are present
Row 5: Strike FF 01 AF = 175 x 10^{-1} = 17.5

4.5.2 Example 2: Full Refresh Message of Series ALLCH

This Snapshot message once decoded is as follows, where the highlighted fields are read directly from the feed (gray) or copied from cache (yellow). The decoded network packet contains:

```
MsgID=500 |MsgSeqNum=14219 |SendingTime=1204196634252 |
Sym ALLCH  |SeriesNumber=28 |CFICode=OC |MaturityMonthDay=20080322|
Strike=40| SecDesc=ALL |UnderlyingNumber= 234 |Status=21 |RefreshIndicator=1
```

The actual data from the feed is as follows:

Full Refresh Message - Prior to market open:

```
                                 | map | symbol
    xx xx xx xx xx xx xx xx xx xx 0D F4 41 4C 4C 43 C8
                                        / A  L  L  C H

      |S#|maturity|strike| SecDsc  |und# |St |map  |
      9C 33 32 B2 81 84 41 4C CC 01 EA B1 0F F4 44
      28/ 3 2 2 /  4 0 /A  L  L / 234/ 2 /
```

Notes:
Row 1: Map 0D F4 means fields 4, 5, 7-10 and 12 are present
Row 1: Symbol is OPRA code = ALLCH
Row 2: Series number starts at 1 within each underlying
Row 2: Maturity date copies '2008 0' from cache and combines with '322' which makes maturity date 20080322
Row 2: Strike price is 40 and CFI =OC = Call option (copied from cache)
Row 2: SecDsc is underlying symbol which is ALL and the underlying number for ALL is 234
Row 2: Number of entries is not present in map and defaults to zero

4.5.3 Example 3: Security Status followed by Incremental Refresh Message

The message once decoded is as follows, where the highlighted fields are read directly from the feed (gray) or copied from cache (yellow). The decoded network packet contains:

```
FAST Reset (C0 F8)

MsgID=400|MsgSeq=1251004 | SendingTimeJavaEpoch=1204205190340|
UnderlyingNumber=234| SeriesNumber=28 |SecurityStatus=17

MsgID=100 |MsgSeq=1251005 | SendingTimeJavaEpoch=1204205190340|
MDEntries=1|MDUpdateAction=0 |MDEntryType=0 |UnderlyingNumber=234 |
SeriesNumber=28|MDEntryPrice=1.5 | MDEntrySize=100 |MDPriceLevel=1 |
QuantityCustomer=0 |QuoteCondition=””

MsgID=100 |MsgSeqNum=1251006 |SendingTimejavaEpoch=1204205190340|
MDEntries=1|MDUpdateAction=0 |MDEntryType=1 |UnderlyingNumber=234 |SeriesNumber=28|
MDEntryPrice=2.5|MDEntrySize=100 |MDPriceLevel=1 |QuantityCustomer=0
|QuoteCondition=””
```

The actual data from the feed is as follows:

```
                                      |Reset|map| ID
   B0 B0 FF 8F E4 81 88 81 A4 B1 FF 99 C0 F8 FE 03 90
1)                                     120/    / 400
```

Security Status Message:

```
     |Seq      | Sendingtime      |und# | s# |st|map|
      4C 2D BC 23 06 01 63 79 C4 01 EA 9C 91 C8
2)   / 1251004/ 1204205190340  / 234 / 28/17/  /
```

Incremental Refresh Message:

```
   ID | #e|map|ac|ty| prc  |siz|PL|map|#e|map|ty| prc |Reset|
   E4  81  E7 B0 B0 FF 8F E4 81 88 81 A4 B1 FF 99 C0 F8
3) 100/ 1/  / 0/ 0/ 1.5/ 100/ 1/  / 1/  /1 / 2.5 / 120
```

Notes:
Row 1: Map FE means fields 1-6 are present
Row 2: Map C8 means fields 1 and 4 are present
Row 3: Message ID 100 = Incremental Refresh message with one entry
Row 3: Map E7 means fields 1, 2, 5, 6 and 7 present
Row 3: Ty = Entry Type = 0 = Bid
Row 3: Map 88 means field 4 is present
Row 3: Map A4 means fields 2 and 5 are present
Row 3: Ty = Entry Type = 1 = Offer

4.5.4 Example 4: Full Refresh Message after an Incremental Refresh Message

This Snapshot comes after an Incremental Refresh message in the middle of the packet so that many fields are defaulted. The message once decoded is as follows, where the highlighted fields are read directly from the feed (gray) or copied from cache (yellow). The decoded network packet contains:

```
MsgID=500 |MsgSeqNum=1265324 | SendingTime=1204205275574 | Symbol=ALLCH
SeriesNumber= 28 |CFICode=OC|Maturity= 20080322 |StrikePrice=40|
SecDesc=ALL |Und 234 |SecurityStatus=17 |RefreshInd=0| MDEntries= 2|

MDEntryType=1 |MDEntryPrice=2.5 |MDEntrySize=100|
MDPriceLevel=1|QuantityCust=0 |QuoteCondition=””|

MDEntryType=0  |MDEntryPrice=1.5|MDEntrySize=100 |MDPriceLevel=1| QuantityCust=0
|QuoteCondition=””
```

The actual data from the feed is as follows:

Full Refresh Message

```
                       |map  |symbol          |S#|maturity|
  82 46 43 D3 01 A1 0D FA 41 4C 4C 43 C8 9C 33 32 B2
1)                  /     /A  L  L  C  H/28/  3  2  2

   |strike| SecDsc  |und# |st|#e|map|ty|prc   |map|ty|
   81 84 41 4C CC 01 EA 91 82 E0 B1 FF 99 E0 B0
2)  40 / A  L  L / 234/17/ 2/   /1/ 2.50 /  / 0

     Prc   | map   |
   FF 8F 0F F8 44 49 41 58 C8 05 D3 4F D0 31 32 32
3) 1.50/
```

Notes:
Row 1: Map OD FA means fields 4, 5, 7-11 and 13 are present
Row 1: Maturity date only has three characters off of the feed and the rest are in the cache
Row 2: Map EO means fields 1 and 2 are present

5 Communications

The ISE Depth of Market Feed is currently distributed by BT Radianz via the Radianz Shared Market Infrastructure and by NYSE Euronext Advanced Trading Solutions Secure Financial Transaction Infrastructure (SFTI®). These providers use advanced telecommunications protocols, and was designed to support a number of industry-standard protocols including IP and UDP as defined by the Internet Engineering Task Force (IETF). We expect additional extranet providers to distribute this data.

ISE Depth of Market data is disseminated using multicast via two redundant lines (A and B) that are intended to provide a level of fault tolerance. **We recommend a 50 MB redundant connection with an aggregate of 100MB of bandwidth**, which represents 100% head room based on the peak data rates as of January 2008. A separate set of addresses and ports are used for functional testing. Please refer to Section 5.2 for failover, Section 5.3 for production IP addresses and 5.4 for the test IP addresses.

5.1 Transmission Standards

The ISE Depth of Market feed utilizes IP version 4 (IPv4) over UDP and Ethernet standards. The figure below illustrates the header information as defined in IETF RFC 791 (IPv4) and RFC 768 (UDP) transmission protocol standards.

Bits	0-7	8-15	16-23	24-31
0	Source Address			
32	Destination Address			
64	Zeros	Protocol	UDP Length	
96	Source Port		Destination Port	
128	Length		Checksum	
160	Data			

Figure 1 - UDP Header

One or more FAST encoded FIX.4.4 messages may be included in a single UDP packet.

5.2 Failover

The ISE Depth of Market Feed has been architected for fault tolerance by providing two redundant streams of data disseminated from two diversely located data centers. There is one feed from each site: 'A' feed from the A site/server and 'B' feed from the B site/server. Each feed has 16 IP addresses and the two feeds carry identical information.

Sequencing is the responsibility of a single server (primary and backup configuration). In the event of a server failure, a delay of several seconds may occur while the backup server resumes operation. In this instance, Snapshot messages of all instruments will be sent before the updates resume. These Snapshots could include state changes of the book that have not been included in update messages and must be processed by client systems to assure data integrity. The full refresh of the order book will take two minutes.

When there has been a failure in the service at the ISE the 'RefreshIndicator' in the message will be set to '1' which indicates that the subscriber should discard the contents in the order book completely and replace it with the contents of this Snapshot message. The 'RefreshIndicator' will also allow the subscriber to only process Snapshots that are set to '1' once the market is open.

The following figure illustrates the general architecture of the communications infrastructure and key software components, including the FAST encoders and decoders.

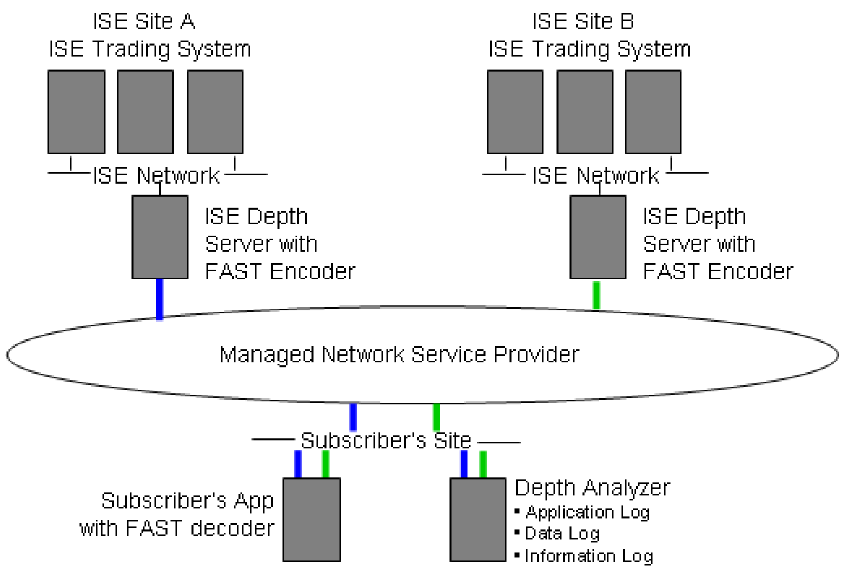

Figure 2: Service Network Redundancy

5.3 Production IP Groups

The table below identifies the primary and secondary multicast address and port information for the production communication services.

	Secondary (Group A)		**Primary (Group B)**	
Line	**Address**	**Port**	**Address**	**Port**
1	233.104.73.1	53001	233.104.73.65	53065
2	233.104.73.2	53002	233.104.73.66	53066
3	233.104.73.3	53003	233.104.73.67	53067
4	233.104.73.4	53004	233.104.73.68	53068
5	233.104.73.5	53005	233.104.73.69	53069
6	233.104.73.6	53006	233.104.73.70	53070
7	233.104.73.7	53007	233.104.73.71	53071
8	233.104.73.8	53008	233.104.73.72	53072
9	233.104.73.9	53009	233.104.73.73	53073
10	233.104.73.10	53010	233.104.73.74	53074
11	233.104.73.11	53011	233.104.73.75	53075
12	233.104.73.12	53012	233.104.73.76	53076
13	233.104.73.13	53013	233.104.73.77	53077
14	233.104.73.14	53014	233.104.73.78	53078
15	233.104.73.15	53015	233.104.73.79	53079
16	233.104.73.16	53016	233.104.73.80	53080

Table 11: Production Multicast Lines

It is the objective of the ISE to balance the load of the market data generated from the Depth Feed across these assigned IP addresses. Therefore quote messages for each symbol will be transmitted over a single line for the entire trading day. However a symbol is not permanently assigned to any single IP address.

Subscribers that chose to listen to a limited number of IPs because they are only interested in a limited number of symbols must realize that if the symbol changes IPs they may no longer have access to the data. In this event, subscriber must request access to the new IP where the symbol resides.

5.4 Testing IP Groups

A third sub-set of addresses and ports will be used to provide test data both during and after market hours but not on weekends. The following symbols are available on each IP address.

Test Group and Symbols			
	Address	**Port**	**Symbols included**
1	233.104.73.129	53129	YHOO, IBM, QQQQ, INTC, SPY, MSFT, ALL,
2	233.104.73.130	53130	IWM
3	233.104.73.131	53131	CHK, HOV, SMH
4	233.104.73.132	53132	C, TIF, SGP

Table 12: Test Multicast Lines

6 Demonstration Application

The ISE FAST Demonstration Application is a simple program that provides an example of how the FAST messages sent by the ISE can be received and decoded at a subscriber's organization. The subscriber can use this open source application to build their own decoder.

The demonstration application will read a single channel of multicast data, interpret that data according to the ISE Depth Feed templates and convert that data into FIX formatted messages. The FIX messages are then written to a log file. The data log files created will eventually consume a lot of hard drive space so we recommend that you perform periodic maintenance on the log files.

The ISE FAST Demonstration Application was generated from an XML format template to decode the FAST messages. It has been developed in Java and is platform independent.

The computer on which the sample application is installed and operated should have the following minimum specifications:

- Processor - 2.0 GHz Intel (or AMD equivalent)
    - RAM - 1 GB
    - Disk - 80 GB
- Operating system - Microsoft® Windows® XP, Service Pack 2.
    - Java™ - Java™ Runtime Environment 1.5 or later.
- Network and Authorization – SFTI network

Please refer to the ISE FAST Demonstration Application Installation Guide for more details.

Due to firewall issues you must request to receive this free application. Please email request to depth@ise.com or call (212)897-8160.