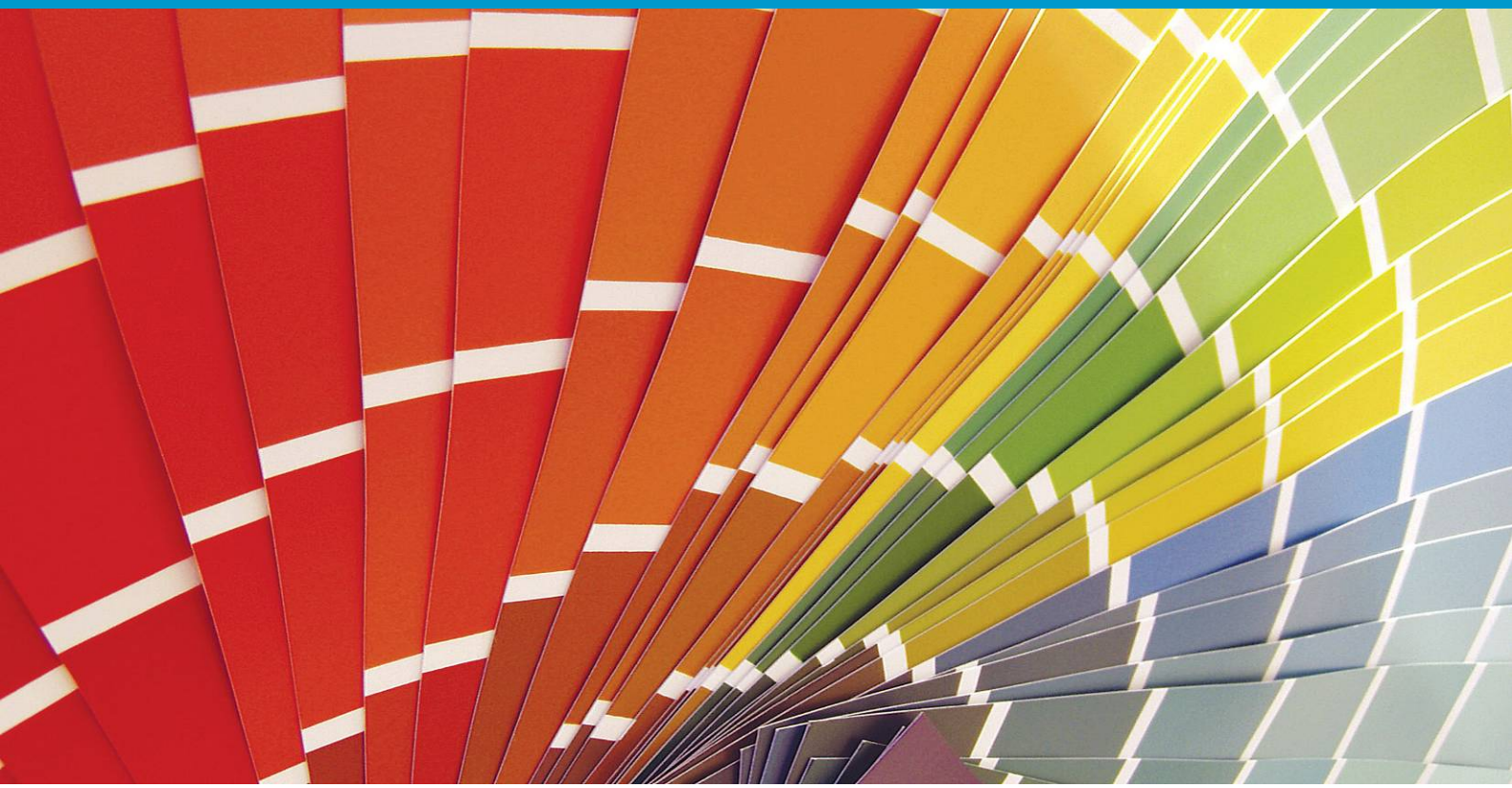
MarketPoint™

ASX FIX™ Implementation for ASX Trade™ Market Data

Version 1.7

The definitive reference data service, direct from the source



ASX Market Information
Information Solutions from the Source

ASX Limited
ABN 98 008 624 691

20 Bridge Street, Sydney NSW 2000
PO Box H224
Australia Square NSW 1215

Telephone: +61 (02) 9227 0000
Facsimile: +61 (02) 9227 0859
Email: info@asx.com.au

Updated: February 2011

Contents

Introduction

FIX Protocol Ltd

The Financial Information eXchange (FIX™) protocol is a purpose-built language for communicating financial information electronically between two parties in a secure environment. FIX is not a software application and does not have a graphical user interface. The FIX protocol is the globally recognised messaging standard enabling the real-time electronic communication of pre-trade, trade and post-trade messages between financial institutions, primarily investment managers, broker/dealers, ECNs and stock exchanges across all instrument types. FIX defines the format of messages sent and received, and is a medium through which two unrelated applications can communicate.

The FIX Protocol is built and maintained by FIX Protocol Ltd; an independent committee of industry participants from across the globe. For the latest FIX specifications or more information on FIX Protocol Ltd and the FIX Protocol, please visit http://www.fixprotocol.org

You can download the FIX standard Version 4.4 specifications from the following link: http://www.fixprotocol.org/specifications/ See Version 4.4 (with Errata 20030618). Refer to Volume 3 for the section on market data.

Ullink Pty Ltd (http://www.ullink.com/) provides the technology used by ASX to generate FIX market data. Additionally, Ullink are an independent provider of connectivity and trading solutions which simplify the processing of FIX market data by brokers or any other party that receives FIX market data from an exchange.

ASX FIX Implementation Manual

This manual describes the ASX FIX Market Data Server which forms part of the ASX MarketPoint data feed. The ASX FIX Market Data Server is the ASX implementation of the industry standard FIX protocol. This implementation has been designed for ASX customers who connect to the ASX's ASX Trade™
Trading System using a Computer to Computer Interface (CTCI).

This manual details the message specifications of all of the FIX messages that are being supported by ASX in its FIX Market Data Server interface to ASX Trade™.

ASX Adherence to FIX Standard

This document assumes the default definition of the FIX standard. Any deviations from this standard will be clearly stated in the Manual. Only the messages and fields which are supported by ASX (for market data) are included in this Manual.

Overview

ASX Implementation of the MarketPoint FIX service

The ASX MarketPoint data feed in the FIX format is a product delivery aimed at supporting the needs of our information vendors, brokers and fund managers alike. Users connect to the MarketPoint FIX gateway using a FIX based interface over TCP/IP.

Please refer to the information on the following few pages for specific details of ASX's implementation of the FIX service.

ASX FIX Version Support

ASX MarketPoint currently supports FIX versions 4.2 and 4.4.

FIX Message Syntax

ASX currently supports the standard "Tag=Value" syntax. The general format of a FIX message applies. Messages consist of a series of "Tag=Value" fields separated by a field delimiter. The delimiter is ASCII 1 (SOH) symbol. All messages begin with a standard header and are terminated with a standard trailer.

Connecting to MarketPoint FIX

The customer's application connects to a predefined TCP port on an ASX gateway from an IP address that has been configured in ASX's system as a source (customer) IP address. The MarketPoint FIX service is available from 2:35am – 7:30pm (Mon-Fri) AEST

There are several methods for users to connect to ASX using the FIX protocol.

1. The first method of connection to ASX is via a gateway located at the customer's premises. The FIX server software will be installed on the customer's gateway by ASX. Connectivity will be achieved utilising the ASX trading network, ASX Net, based on point to point frame relay lines.

2. Alternatively customers will be able to connect to a series of gateways located at ASX data centres, connectivity to these gateways can be achieved via an approved VPN (Virtual Private network). Currently the Optus eFinity, Telstra Coin and Radianz VPN's are approved to carry ASX MarketPoint data.

FIX User Authentication and Logon

There are several stages of authentication when connecting to MarketPoint.

1. The first stage of authentication is accomplished by ASX's system checking the user's IP address and SenderCompID (FIX tag 49). The customer's source IP address will have previously been permissioned by ASX. It is important that customers notify ASX of any changes of their IP address before such changes occur. Customers may do this by contacting the ASX Market Access department via email to Market.Access@asx.com.au during the initial stage of the FIX Market Data Server implementation.

2. The second stage of authentication is achieved within the FIX Logon process. The user sends their unique ASX Trade™ username and password to ASX as part of the FIX Logon message. Users will need to make use of the 553 Username and 554 Password fields when sending the Logon message to ASX, regardless of which FIX version they are using – 4.2 or 4.4 because ASX will be making these two fields standard in its implementation of the ASX FIX Market Data Server.

   Only when the user's ASX Trade™ username and password are verified by ASX Trade™ and a Logon confirmation message is sent back to the user from ASX, will the user have achieved a successful logon to ASX Trade™ via the FIX gateway. User's are required to maintain passwords in accordance with the following ASX Trade™ password expiration & validation criteria:
   - Password Expiration days = 90
   - Password history keep days = 365
   - Number of allowed login attempts = 3
   - Number of allowed inactive days = 60
   - Minimum number of characters = 8
   - Allowed character set = All printable characters within the ASCII character set, except for percent (%) and backslash (\).
   - Minimum alpha characters =1
   - Minimum numerals = 1
   - Minimum special characters = 1

refer BE: User request message for details on changing a password

FIX Sessions and Message Sequence Numbers

A FIX session is established through a gateway with a dedicated link in to the ASX network (ASX Net) using TCP/IP over a leased line or a VPN connection. The ASX gateway acts as the server and the customer's system acts as a client.

A FIX session consists of a series of messages with a continuous set of sequence numbers. Resetting the inbound and outbound sequence numbers back to 1 establishes a new FIX session. A client can establish a new FIX session by setting the ResetSeqNumFlag [141] to "Y" in the login message. ASX does not initiate sequence number resets.

Inbound and outbound sequence numbers should be tracked independently by the user. This should be carried out so as to identify gaps in message delivery. After authentication, the client and host must synchronize their message sequence numbers before any new messages can be sent.

If the sequence number of the message received is higher then the expected sequence number, the client should respond with a Resend Request specifying the range of missing sequence numbers in the BeginSeqNo [7] and EndSeqNo [16] tags. ASX supports the value of 0 (infinity) in tag 16 which would indicate the highest expected sequence number in tag 16.

If the sequence number of the message received is less then the expected sequence number and the PossDupFlag [43] is not set to "Y", it will not be possible to establish a FIX session until the sent sequence number is equal to or greater then the one expected. A reject message will be sent with the expected sequence number sent in the Text [58] tag.

By default, ASX does not reset the sequence numbers at the end of each day. However it is possible for users to program their FIX scripts to do this for them. Typically sequence numbers are reset to 1 (inbound and outbound) each morning at approximately 02:30 am.

FIX Message Recovery

Message gaps may be discovered during login or during a FIX session. Such gaps require message recovery. The user can send a Resend Request message requesting a range of missing messages. The server will then respond with a Sequence Reset message that has the GapFillFlag [123] set to "Y" and the PossDupFlag [43] set to "Y". Tag 36 will be set to the sequence number of the message to be redelivered next. Following this, the missing Application level messages will be resent. This process of Sequence Reset / message delivery will continue until all of the relevant messages within the specified range have been delivered.

Most Administrative messages will not be resent, namely: Logon, Logout, Resend Request, Heartbeat, Test Request, Sequence Reset.

Market Data Coverage

One of the advantages of using FIX to retrieve Equity market data, is that the FIX engine uses Equity detailed orderbook changes and top level price depth provided by ASX's ASX Trade™ Trading System to build Equity depth for the FIX end user. You have an added configurability option with FIX in that you can request from 0 up to N levels of Equity depth.

Alternatively, using the OMX Genium INET℠ API (the native language of the ASX's ASX Trade™ Trading System) to retrieve Equity market data, only Equity top level price depth and detailed orderbook changes are available. Note that this restriction does not apply

when using FIX because of the fact that you have the flexibility of being able to request from 0 to N levels of Equity depth.

For Derivatives, the top 5 levels of Derivatives price depth is provided dynamically using the OMX Genium INET[SM] API whereas using FIX, you have the flexibility of being able to request anything from the top level to the top 5 levels of Derivatives depth.

ASX FIX Market Data

Types of Market Data

Market data can be categorised as follows:

- Top of Book
- Depth
    - Aggregated
    - Unaggregated (i.e. Detail)

In other words, market data can be categorized as "top of book" and "depth". In addition there are two types of depth: aggregated and unaggregated.

Top of book data just gives the best prices and quantities for a given instrument. It also provides other single valued information such as the last trade price and quantity, opening price, closing price, high and low price for the current trading day, etc.

Depth data also gives you the second best price and quantity, third best, etc. Depth can be limited - for example only the top 5 levels, or unlimited.

Aggregated depth just displays the total quantity available in the market at each price level. Unaggregated depth (sometimes called "detail") shows all the orders in the market - so there may be several entries at a given price level.

Types of Market Data Update

There are two types of market data update:

- Snapshot (or Full Refresh)
- Incremental Refresh

In a snapshot, everything is sent. For example, all top of book data, or the full depth. Snapshots are easier for a client to process.

In incremental updates, only the data that has changed is sent. Incremental updates send less data so can be faster and use less bandwidth. On the other hand, they are more difficult for a client to process since each change has to be applied to a store of existing data maintained by the client.

Message Dissemination

FIX market data is sent in response to subscription requests. Typically a FIX client will connect and logon to the source of FIX market data in the usual manner. Then the FIX client will send one or more messages indicating which instruments it wants to receive market data for. The FIX source will respond by sending back FIX messages containing market data for those instruments.

When the FIX client logs off, all subscription information is lost. If the client connects and logs on again, it will have to resend subscription requests before it will receive market data.

There are only four types of message associated with FIX market data:

- *Market Data Request* (MsgType = V) - Used to subscribe (and unsubscribe) to market data.
- *Market Data Reject* (MsgType = Y) - Sent in response to a Market Data Request that cannot be satisfied for some reason.
- *Market Data - Snapshot* (MsgType = W) - Market data snapshot message.
- *Market Data - Incremental Refresh* (MsgType = X) - Market data subscription message containing initial snapshot and subsequent incremental updates.

Subscribing for Updates

Each subscription request has a unique request ID. All market data messages received in response to that request will have that ID as part of the message.

Subscription requests can specify the following information:

- The instrument(s) for which market data is requested. (See FIX tag Symbol (55)).
- The market data fields being requested on each update message. (See FIX tag MDEntryType (269)).
- The type of market data being requested: top of book or depth - limited or complete, aggregated or not aggregated. (See FIX tags MarketDepth (264), AggregatedBook (266)).
- The type of updates requested: snapshot or incremental, one off or continuously updating. (See FIX tags SubscriptionRequestType (263), MDUpdateType (265)).

It is possible that a subscription may not be supported by a source of market data. For example, the instrument may be unknown. Or the source may only supply a limited number of price depth levels dynamically- not full order depth/ detail. Or the requested market data fields are not supported. In such cases a Market Data Reject message is sent in response to the subscription request.

Depth Data

Maintaining depth data from snapshot updates is straightforward since each update gives you the complete new depth. You just replace any previous depth data with the new depth.

Maintaining depth data from incremental market data updates is more complex. You need to modify your existing depth data. For example, you may receive an update which asks you to update the quantity of the 7th buy order to some new value. You have to locate that particular depth entry in your existing data and update it.

This raises the question of how to identify a particular depth entry. The FIX standard does not mandate how to do this. It provides several methods and it is up to the market data source to decide which method(s) it will provide.

The options are:

- Identify each depth entry with a unique ID (FIX tag MDEntryID (278)).
- Identify a depth entry by its instrument (FIX tag Symbol (55)), side (buy or sell) (FIX tag MDEntryType (269)) and price (FIX tag MDEntryPx (270)). This only works for aggregated depth where there is only one entry for each price.
- Identify a depth entry by its instrument, side and position in the depth (FIX tag MDEntryPositionNo (290)).

Aggregated Data versus Non-aggregated Data

With FIX, you have the option to turn data aggregation on or off. You do this with the use of tag 266 AggregatedBook.

By default, tag 266 is set to equal 'N'. In other words aggregation is turned off by default.

The only time you should turn aggregation on is when you subscribe to depth and top of market. You can request from 0 up to N levels of depth for Equities and from 0 up to 5 levels of depth for Derivatives.

Following is a table that summarises how and when tag 266 AggregatedBook should be used.

Using FIX tags to request, top of market, depth and detail			
	Top of market	Depth	Detail
Equities	264=1 and 266=Y	264=0-N (excluding 1) and 266=Y	264=0-N (excluding 1) and 266=N
Derivatives	264=1 and 266=Y	264=2-5 and 266=Y	n/a

General Characteristics

- Supports top of book
- Supports unlimited, un-aggregated depth
- Supports incremental updates
- Uses unique ID's to identify depth entries

Special Characteristics

- Prices disseminated by the MarketPoint FIX server are transmitted in tenths of a cent, e.g. $1.00 would be sent as 1000. Note that this format is not consistent with ASX's Order Management FIX server which disseminates prices as dollar values.

Depth Entry Identification

Each depth entry in the market is identified by a unique ID. This ID appears in the FIX tag MDEntryID (278) of all depth market data update messages.

In standard FIX this ID can be any String. However, in MarketPoint FIX for market data, this field is always a non zero, positive number - suitable for using as the index into an array. When a depth entry is deleted from the market, its ID becomes available again and can be reused. This means that the values the ID can take is bounded to a reasonably small number. It should never be more than the maximum number of the orders in the market at any one time.

The fact that the ID is a relatively small positive number makes it practical to maintain a simple array of references to all depth entries in a client's depth data. This allows for extremely efficient updating of depth data from incremental market data updates. Just extract the MDEntryID from the update and use it to locate the corresponding depth entry by indexing into an array of depth entries. It is not necessary to know the side, price or even the instrument associated with the update.

Unsupported Messages and Fields

If the FIX Client sends an unsupported message, then they will get back a rejection message.

When the FIX Client sends a supported message:

- If they send an unsupported field as part of this supported message, then this field will be completely ignored by the FIX Server. The Client will not get a rejection message for this field.

- If the Client sends a supported field as part of this supported message, and this field contains an invalid value, the FIX Client will get a rejection message for this field indicating that the Server doesn't understand the value sent by the Client.

Supported Top of Book Fields

MDEntryType	FIX Tag	Notes
Bid (0)	MDEntryPx (270) MDEntrySize (271)	Best bid price Best bid quantity
Offer (1)	MDEntryPx (270) MDEntrySize (271)	Best offer price Best offer quantity
Trade (2)	MDEntryPx (270) MDEntrySize (271)	Last trade price Last trade quantity
Index value (3)	MDEntryPx (270)	Last index value
Opening price (4)	MDEntryPx (270)	Opening price
Closing price (5)	MDEntryPx (270)	Closing price
Trading session high price (7)	MDEntryPx (270)	High price
Trading session high price (8)	MDEntryPx (270)	Low price
Trade volume (B)	TotalVolumeTraded (387)	Total trade volume
Cumulative Value (H)	MDEntryPx (270)	Total trade value
Indicative Auction Price (I)	MDEntryPx (270) MDEntrySize (271)	Indicative Auction Price Equilibrium Volume
Surplus Volume (J)	MDEntrySize (271) TradeCondition (277)	Surplus volume Flag indicating side of surplus

Supported Depth Fields

MDEntryType	FIX Tag	Notes
Bid (0)	MDEntryPx (270) MDEntrySize (271) OrderID (37)	Bid depth entry price Bid depth entry quantity Order ID associated with bid depth entry
Offer (1)	MDEntryPx (270) MDEntrySize (271) OrderID (37)	Offer depth entry price Offer depth entry quantity Order ID associated with offer depth entry

FIX Messages Supported by ASX

Within the FIX specification there are many optional fields. This document defines the fields and conventions that ASX uses in its implementation of its MarketPoint FIX feed.

Additionally, ASX uses some messages and fields specific to FIX versions 4.4 in its implementation of FIX 4.2. ASX may also make use of FIX custom tags in its future implementations of FIX. Please refer to later sections of this Manual for specific details of messages and fields that ASX is supporting in its implementation of the FIX MarketPoint feed.

Following is a list of FIX messages supported by ASX in its current implementation of the FIX service.

FIX Message Name	Category	Comments
Standard Message Header	Standard Header/Trailer	This is prefixed to all incoming and outgoing FIX messages. Refer to FIX specifications for full details.
Standard message Trailer	Standard Header/Trailer	This is appended to all incoming and outgoing FIX messages. Refer to FIX specifications for full details.
Heartbeat	Administrative Messages	
Logon	Administrative Messages	The fields 553 *Username* and *554 Password* will be implemented across all the FIX versions that ASX is supporting – 4.2 and 4.4.
Test Request	Administrative Messages	
Resend Request	Administrative Messages	
Sequence Reset	Administrative Messages	
Reject	Administrative Messages	
Logout	Administrative Messages	
Market Data Request	Application Messages – Pre-Trade Messages (Market Data)	
Market Data Snapshot Full Refresh	Application Messages – Pre-Trade Messages (Market Data)	
Market Data Incremental Refresh	Application Messages – Pre-Trade Messages (Market Data)	
Market Data Request Reject	Application Messages – Pre-Trade Messages (Market Data)	
Security Status Request	Application Messages – Pre-Trade Messages (Security and Trading	

FIX Message Name	Category	Comments
	Session Definition/Status)	
Security Status	Application Messages – Pre-Trade Messages (Security and Trading Session Definition/Status)	
Trading Session Status Request	Application Messages – Pre-Trade Messages (Security and Trading Session Definition/Status)	
Trading Session Status	Application Messages – Pre-Trade Messages (Security and Trading Session Definition/Status)	

The following sections contain details of the FIX fields supported by ASX in its implementation of the FIX service.

Message Formats

All messages begin with a standard header and end with a trailer that comprises a checksum.

Note: The FIX Specification fields (except the trailer) can be sent in any order as per FIX Specification 4.2 FIX message format and delivery.

Message header

The comment for the *MsgType* (35) field lists all messages that are supported. Unsupported messages and field values are rejected using the *Business Message Reject* (j) message with explanatory text.

Standard message header

Tag	Field Name	Rqd	Format	Definition
8	BeginString	Y	String	Protocol versions. Valid Values : FIX.4.2 FIX.4.4
9	BodyLength	Y	Integer 2-byte unsigned	Length of body not including header/trailer.
35	MsgType	Y	String	Message type. Valid values: 0 = Heartbeat 1 = Test Request 2 = Resend Request 3 = Reject 4 = Sequence Reset 5 = Logout A = Logon V = Market Data Request W = Market Data-Snapshot X = Market Data-Incremental Refresh Y = Market Data Request Reject e = Security Status Request f = Security Status j = Business Message Reject

Tag	Field Name	Rqd	Format	Definition
49	SenderCompID	Y	String	Sender company ID. Messages sent to ASX need to contain this field. Every FIX user will be assigned a unique SenderCompID by ASX. Customers should contact ASX to obtain their SenderCompID.
56	TargetCompID	Y	String	Target company ID. In messages sent to the ASX, this value should always be set to “ASX”.
34	MsgSeqNum	Y	Integer 4-byte unsigned	Message sequence number, cannot be zero, starts at 1 at the start of each day. Refer to FIX Specification V4.2 message format and delivery, sequence number.
43	PossDupFlag	N	Boolean	Indicates possible retransmission of message with this sequence number. Valid values: Y = Possible duplicate N = Original transmission If not specified, defaults to ‘N’.
97	PossResend	N	Boolean	Indicates that message may contain information that has been sent under another sequence number. Valid values: Y = Possible resend N = Original Transmission If not specified, defaults to ‘N’.
52	SendingTime	Y	UTC Time Stamp	Time of message transmission. Valid format YYYYMMDD-HH:MM:SS
122	OrigSendingTime	N	UTC Time Stamp	Original time of message transmission. Mandatory for message resends. If data is not available set to same value as SendingTime Valid format YYYYMMDD-HH:MM:SS

Tag	Field Name	Rqd	Format	Definition
369	LastMsgSeqNumProcessed	N	Integer 4-byte unsigned	The last MsgSeqNum value received and processed. Can be specified on every message sent. Useful for detecting a backlog with the counterparty.

Message trailer

A standard trailer terminates each message, administrative or application.

Refer to *Volume 2 of the FIX 4.4 Specification* for the details on the calculation of the checksum.

Standard Message Trailer

Tag	Field Name	Rqd	Format	Definition
10	CheckSum	Y	String	Three bytes with preceding zeroes, simple check sum, modulo 256. Calculated for entire message (i.e. includes all header fields and body).

Administrative and Other Messages

This section details the FIX administrative messages supported by the ASX FIXGW.

0: Heartbeat

As defined in the FIX specification.
FIX Version Support: 4.2 and 4.4

1: Test Request

As defined in the FIX specification.
FIX Version Support: 4.2 and 4.4

2: Resend Request

As defined in the FIX specification.
FIX Version Support: 4.2 and 4.4

3: Reject

As defined in the FIX specification.
FIX Version Support: 4.2 and 4.4

4: Sequence Reset

As defined in the FIX specification.
FIX Version Support: 4.2 and 4.4

5: Logout

As defined in the FIX specification.
FIX Version Support: 4.2 and 4.4

j: Business Message Reject

As defined in the FIX specification.
FIX Version Support: 4.2 and 4.4

A: Logon

To establish a session, send the *A: Logon* message. This will establish a FIX session with the ASX FIXGW. The username and password must be sent in the *A: Logon* message.

The ASX FIXGW resets FIX session sequence numbers to 1 at 00:30 am on daily basis. It is recommended that the FIX Client logs out shortly before the specified reset time. If this is not done, the FIX session will be logged out at the reset time. After that the client should start a new session by sending a Logon message with the sequence number set to 1.

If a client loses connection to the gateway and is forced to connect to a different gateway, the client will need to set its sequence numbers back to 1.

Although a client can log on and off several times over the same FIX session, only one unique username should be used per FIX session. If a request is sent by the client for a different username, this request will be rejected. If the ASX FIXGW loses connection to ASX Trade™, the FIX session will be logged out. The client should then send a new *A: Logon* message.

There is currently no mechanism to change a password in MarketPoint FIX, and the number of days to password expiry is not sent via MarketPoint FIX. Monitoring password expiry and password change must be performed through ASX Trade™.

Tag	Field Name	Rqd	Format	Definition
	Standard Header	Y		MsgType = A
98	EncryptMethod	Y	Int	Always 0 (unencrypted)
108	HeartBtInt	Y	Int	Heartbeat interval in seconds, or 0 for no heartbeat required. Note same value used by both sides
141	ResetSeqNumFlag	N	Boolean	Indicates both sides of a FIX session should reset sequence numbers.
553	Username	Y	String	Userid or username.
554	Password	Y	String	Password or passphrase.
	Standard Trailer	Y		

FIX Version Support: 4.2 and 4.4

A: Logon Message Example 1

A FIX 4.4 client logs on setting heartbeat interval to 0 (heartbeats not required).

Message Representation (Logon request):

```
8=FIX.4.4|9=99|35=A|49=Client2|56=ASX|34=1|52=20080110-
06:23:18|98=0|108=0|141=Y|553=110SHARPLES|554=apitestpwd_4|10=052|
```

Message Representation (Logon response - success):

```
8=FIX.4.4|9=66|35=A|49=ASX|56=Client2|34=1|52=20080110-
06:23:23|108=0|141=Y|98=0|10=167|
```

BE: User Request

In order to receive market data, a client application needs to have signed on specifying a valid username and password. The username and password may be sent in the ***Error! Reference source not found.*** message or by sending a *BE: User Request* message with Tag 924 = 1 (LogOnUser).

To logout from a FIX session a client application can perform either of the following

1. Send a *5: Logout* message
2. Send a *BE: User Request* message with Tag 924 = 2 (LogOffUser) followed by a *5: Logout* message

To change the password a client application must send a *BE: User Request* message with Tag 924 = 2 (ChangePasswordForUser). Although the *BE: User Request* message is a FIX 4.4 message, this message is required to be sent by the client for all FIX versions.
If the user's password has expired, then a *BF: User Response* message with Tag 926 (UserStatus) = 6 (Other) and Tag 927 (UserStatusText) = "Password has expired" will be sent.

Tag	Field Name	Req'd	Format	Description
	Standard Header	Y		MsgType = BE
923	UserRequestID	Y	String	Unique identifier for a User Request.
924	UserRequestType	Y	Int	Indicates the action required by a User Request Message Valid values: 1 = LogOnUser 2 = LogOffUser 3 = ChangePasswordForUser
553	Username	Y	String	Userid or username.
554	Password	N	String	Password or passphrase.
925	NewPassword	N	String	New Password or passphrase
	Standard Trailer	Y		

FIX Version Support: 4.2 and 4.4 (Although this message is a FIX 4.4 message it is required to be sent by the client for all versions)

FIX 4.2 Notes:

Tag 7951 is not supported.

User Request Message Example 1

A FIX 4.4 client logs on with existing password.

Message Representation:

```
8=FIX.4.4|9=90|35=BE|49=CLIENT|56=ASX|34=6|52=20050725-
06:33:26|923=UR65|924=1|553=Username|554=Password|10=093
```

User Request Message Example 2

A FIX 4.4 client logs off.

Message Representation:

```
8=FIX.4.4|9=76|35=BE|49=CLIENT|56=ASX|34=64|52=20050726-
05:43:23|923=66|924=2|553=Username|10=191
```

User Request Message Example 3

A FIX 4.4 client changes password.

Message Representation:

```
8=FIX.4.4|9=105|35=BE|49=CLIENT|56=ASX|34=38|52=20050725-
07:48:02|923=66|924=3|553=Username|554=Password|925=NewPassword|10=119
```

BF: User Response

A *BF: User Response* message will be sent in response to a ***Error! Reference source not found.*** message.
The tag 7954 (DaysTillPasswordExpiry) will only be sent if the client set tag 7953 (ReceiveCustomTags) = Y was sent in the *A: Logon* message.

Tag	Field Name	Req'd	Format	Description
	Standard Header	Y		MsgType = "BF"
923	UserRequestID	Y	String	
553	Username	Y	String	

926	UserStatus	N	Int	Indicates the status of a user Valid values: 1 = Logged In 2 = Not Logged In 3 = User Not Recognised 4 = Password Incorrect 5 = Password Changed 6 = Other
927	UserStatusText	N	String	Reason a request was not carried out
7954	DaysTillPasswordExpiry	N	Int	The number of days till the user's password expires. This tag will only be set if the tag 7953 (ReceivedCustomTags) was set to 'Y' in the Logon request message.
	Standard Trailer	Y		

FIX Version Support: 4.2 and 4.4 (Although this message is a FIX 4.4 message it is required to be sent by the server for all versions)

User Response Message Example 1

The ASX FIXGW sends a successful logon response to a FIX 4.4 client with no custom tags sent.

Message Representation:

```
8=FIX.4.4|9=82|35=BF|49=ASX|56=CLIENT|34=100|52=20050728-
02:25:57|553=Username|923=UR101|926=1|10=136
```

User Response Message Example 2

The ASX FIXGW rejects a BE: User Request due to an incorrect password.

Message Representation:

```
8=FIX.4.4|9=131|35=BF|49=ASX|56=CLIENT|34=340|52=20050728-
06:17:14|553=Username|923=UR115|926=2|927=OM Login failed -2024
Invalid login attempt.|10=089
```

User Response Message Example 3

The ASX FIXGW sends a successful user response to a FIX 4.4 client with custom tag 7954 sent.

Message Representation:

```
8=FIX.4.4|9=113|35=BF|49=ASX|56=CLIENT|34=400|52=20050906-
01:37:40|57=203bajer_e|553= Username|923=UR101|926=1|7954=83|10=117
```

Market Data Messages

V: Market Data Request (from client)

The V: *Market Data Request* message is used to request market data on an instrument.

If *SubscriptionRequestType (263) = 0* (a request for a snapshot) a W: *Market Data Snapshot* message containing snapshot information will be returned. If the request cannot be honoured a Y: Market Data - Request Reject message will be returned.

If *SubscriptionRequestType (263) = 1* (a request for snapshot plus updates) an initial X: Market Data Incremental Refresh message will be sent containing snapshot information followed by subsequent X: Market Data Incremental Refresh messages containing market data updates as those updates occur. If the subscription cannot be honoured a Y: Market Data – Request/Reject message will be returned.

If *SubscriptionRequestType (263) = 2* (unsubscribe) an X: Market Data – Incremental Refresh message without update information will be sent. No further updates will then be received for the subscription that was unsubscribed. If the unsubscription cannot be honoured, a Y: Market Data - Request Reject message will be returned.

For any other value of *SubscriptionRequestType* a Y: Market Data - Request Reject message will be returned.

Notes:

- Multiple instruments can be sent in a single Market Data Request message (security, index, ETF or EIN).

- A request for a Snapshot (SubscriptionRequestType = 0) causes the current state of the market for the instrument specified in the request to be sent to the client.

- A request for Snapshot and Updates (SubscriptionRequestType = 1) causes the current state of the market for the instrument specified in the request to be sent and any updates as they occur, until the client requests that the Snapshot and Updates be disabled. The complete data for the instrument specified in the request is sent in an initial Market Data – Incremental Refresh message, and then any updates are sent in subsequent Market Data – Incremental Refresh messages.

- The different subscription types are: Top of Market, Depth and Detail.

- If a new subscription is sent for an instrument while there is an existing subscription for that instrument, the new subscription request will overwrite the previous subscription request of the same type. The different subscription types are: Top of Market, Depth and Detail. For example, a new subscription for Top of Book for a security would overwrite an existing prior subscription for Top of Book for the same security. If these two subscriptions differ in terms of 269 MDEntryType (bid, offer, last

etc), then the MDEntryType requested in the first subscription will no longer be disseminated after you send your second Top of Book subscription. What will be disseminated is the MDEntryType you requested in your second subscription.

- All subscriptions are lost on loss of FIX connection. The client must re-subscribe.

- An unsubscribe request must contain an instrument code matching the original subscription. Additionally an unsubscribe cancels subscriptions to all MDEntryTypes for that instrument code.

- If a user sends a request for best bid and offer for an instrument for which there are currently no bids or offers in the market, the tags related to bid and offer will not be returned to the user in the reply message.

The table below illustrates which MDEntryType values are valid depending on whether the security requested is a Security, Index, ETF or NAV (EINs) within the market data messages.

Instrument type	MDEntryType (269)
Security	0,1,2,4,5,7,8,B,H,I,J
Index	3 = Index value
ETF	0,1,2,4,5,7,8,B,H,I,J
EIN	3=Index value

V: Market Data Request

Tag	Field Name	Rqd	Format	Definition
	Standard Header	Y		MsgType = V
98	EncryptMethod	Y		Valid values: 0 = None Always set to zero. ASX does not support encryption.
262	MDReqID	Y	String	Unique identifier for Market Data Request message. Must be unique, or the ID of a subscribing Market Data Request to disable if SubscriptionRequestType = Disable previous Snapshot + Updates Request (2). That is, must be unique within the FIX connection. ASX validation: minimum string length is 1, maximum is 255 characters. Must refer to the MDReqID of the request. This message is always a response to a Market Data Request message (since MDUpdateType=0, full refresh is not supported). MDReqID will be echoed from request message.
263	SubscriptionRequestType	Y	Character	*SubscriptionRequestType* indicates to the other party what type of response is expected. 0 = Snapshot – only requests current information 1 = Snapshot and Updates (Subscribe) requests current information plus updates as the status change 2 = Disable previous Snapshot and Updates (Unsubscribe) request cancels any future update messages from the counter party

Tag	Field Name	Rqd	Format	Definition
264	MarketDepth	Y	Integer	Depth of market. Valid values: 0 = Full Book *(market detail / Level 2)* 1 = Top of Book *(Level 1)* 5 = 5 levels of Market Depth 5>N>1 = Report best N price tiers of data A value other than 1 (Top of Book) is only supported for the following MDEntryType (tag 267) values 0 = Bid 1 = Offer
265	MDUpdateType	N	Integer	Market data update type. Mandatory if *SubscriptionRequestType* = Snapshot and Updates (1). Not permitted otherwise. Valid values: 1 = Incremental Refresh
266	AggregratedBook	N	Character	Valid values: Y = aggregated prices N = non-aggregated prices This tag is set to N by default. This tag should not be sent if MarketDepth (tag 264) = 1 (Top of Book). If the request is for a Derivatives stock and MarketDepth (tag 264) is not set to 1 (top of book), this field must be set to Y. Otherwise you will get an error Subscription failed : Status = -39.
267	NoMDEntryTypes	Y	Integer	Number of *MDEntryType* fields requested. For unsubscriptions (tag 263 = 2) this must be 0, otherwise it must be > 0.

Tag	Field Name	Rqd	Format	Definition
→ 269	MDEntryType	Y	Character	Will not exist if unsubscription (i.e. tag 263 = 2 and tag 267 = 0), otherwise this group will repeat as many times as specified by the value in NoMDEntryTypes. Mandatory if NoMDEntryTypes is non-zero. Valid values: 0 = Bid (returns price and quantity) 1 = Offer (returns price and quantity) 2 = Trade (returns price and quantity) 3 = Index value (returns index value as a price) 4 = Opening Price (returns price) 5 = Closing Price (returns price) 7 = Trading Session High Price (returns price) 8 = Trading Session Low Price (returns price) B = Trade Volume (returns total volume traded) H = Cumulative Value (returns price) I = Indicative Auction Price J = Surplus Volume (returns quantity and side). *price* is returned in the field MDEntryPx (tag 270). *quantity* is returned in the field MDEntrySize (tag 271). *total volume traded* is returned in the field TotalVolumeTraded (tag 387). *side* is returned as an imbalance flag in the field TradeCondition (tag 277)
7956	UndisclosedQtyIndicator	Y	Character	Valid Values: Y = Undisclosed Quantity is present N = Undisclosed Quantity is not present
1070	MDQuoteType	Y	Integer	0= Indicative 1 = price_type_equilibrium i.e. Normal IAP/ Equilib Price 2 = matching_price_type_fixed i.e. Volume Match Fixed Price
146	NoRelatedSym	Y	Integer	Number of instruments (ASX symbols) requested. This tag must precede tag 55 in this message. Minimum value is 1. Maximum value is 24.

Tag	Field Name	Rqd	Format	Definition
→ 55	Symbol	Y	String	Ticker symbol. Common, "human understood" representation of the security. Instrument code, this field will repeat as many times as specified by the value in NoRelatedSym. If SubscriptionRequestType = 2, this must match the value of Symbol specified in the subscribing Market Data Request message
	Standard Trailer	Y		

FIX Version Support: 4.2 and 4.4

V: Market Data Request Message Example 1

A FIX 4.4 client subscribes to Market Data index values for 24 indices.

Request Message – from Client (Market Data Request):

```
8=FIX.4.4|9=272|35=V|49=Client2|56=ASX|34=2|52=20080110-
05:40:41|262=XAO3|263=1|264=1|265=1|266=Y|267=1|269=3|146=24|55=XAO|55
=XBW|55=XDJ|55=XEJ|55=XFJ|55=XFL|55=XHJ|55=XIJ|55=XJO|55=XKO|55=XMD|55
=XMJ|55=XNJ|55=XPJ|55=XSJ|55=XSO|55=XTJ|55=XTL|55=XTO|55=XUJ|55=XXJ|55
=YSFY|55=YSLF|55=YSTW|10=210|
```

Response Messages – from Server (Market Data - Incremental Refresh):

```
8=FIX.4.4|9=93|35=X|49=ASX|56=Client2|34=2|52=20080110-
05:40:41|262=XAO3|268=1|279=0|269=3|55=XAO|270=67796|10=72|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=3|52=20080110-
05:40:41|262=XAO3|268=1|279=0|269=3|55=XBW|270=476711|10=122|
8=FIX.4.4|9=93|35=X|49=ASX|56=Client2|34=4|52=20080110-
05:40:41|262=XAO3|268=1|279=0|269=3|55=XDJ|270=28361|10=57|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=5|52=20080110-
05:40:41|262=XAO3|268=1|279=0|269=3|55=XEJ|270=161075|10=108|
8=FIX.4.4|9=93|35=X|49=ASX|56=Client2|34=6|52=20080110-
05:40:41|262=XAO3|268=1|279=0|269=3|55=XFJ|270=75131|10=58|
8=FIX.4.4|9=93|35=X|49=ASX|56=Client2|34=7|52=20080110-
05:40:41|262=XAO3|268=1|279=0|269=3|55=XFL|270=66191|10=67|
8=FIX.4.4|9=93|35=X|49=ASX|56=Client2|34=8|52=20080110-
05:40:41|262=XAO3|268=1|279=0|269=3|55=XHJ|270=91456|10=70|
8=FIX.4.4|9=92|35=X|49=ASX|56=Client2|34=9|52=20080110-
05:40:41|262=XAO3|268=1|279=0|269=3|55=XIJ|270=5704|10=14|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=10|52=20080110-
05:40:41|262=XAO3|268=1|279=0|269=3|55=XJO|270=67719|10=124|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=11|52=20080110-
05:40:41|262=XAO3|268=1|279=0|269=3|55=XKO|270=67817|10=125|
```

```
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=12|52=20080110-
05:40:42|262=XAO3|268=1|279=0|269=3|55=XMD|270=67939|10=123|
8=FIX.4.4|9=95|35=X|49=ASX|56=Client2|34=13|52=20080110-
05:40:42|262=XAO3|268=1|279=0|269=3|55=XMJ|270=158677|10=179|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=14|52=20080110-
05:40:42|262=XAO3|268=1|279=0|269=3|55=XNJ|270=73313|10=115|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=15|52=20080110-
05:40:42|262=XAO3|268=1|279=0|269=3|55=XPJ|270=24897|10=131|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=16|52=20080110-
05:40:42|262=XAO3|268=1|279=0|269=3|55=XSJ|270=85703|10=128|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=17|52=20080110-
05:40:42|262=XAO3|268=1|279=0|269=3|55=XSO|270=39904|10=136|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=18|52=20080110-
05:40:42|262=XAO3|268=1|279=0|269=3|55=XTJ|270=16399|10=136|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=19|52=20080110-
05:40:42|262=XAO3|268=1|279=0|269=3|55=XTL|270=37209|10=132|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=20|52=20080110-
05:40:42|262=XAO3|268=1|279=0|269=3|55=XTO|270=54877|10=137|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=21|52=20080110-
05:40:42|262=XAO3|268=1|279=0|269=3|55=XUJ|270=73945|10=131|
8=FIX.4.4|9=94|35=X|49=ASX|56=Client2|34=22|52=20080110-
05:40:42|262=XAO3|268=1|279=0|269=3|55=XXJ|270=78147|10=134|
8=FIX.4.4|9=95|35=X|49=ASX|56=Client2|34=23|52=20080110-
05:40:43|262=XAO3|268=1|279=0|269=3|55=YSFY|270=65044|10=210|
8=FIX.4.4|9=95|35=X|49=ASX|56=Client2|34=25|52=20080110-
05:40:43|262=XAO3|268=1|279=0|269=3|55=YSLF|270=22923|10=198|
8=FIX.4.4|9=95|35=X|49=ASX|56=Client2|34=26|52=20080110-
05:40:43|262=XAO3|268=1|279=0|269=3|55=YSTW|270=65751|10=230|
```

V: Market Data Request Message Example 2

A FIX 4.4 client requests a snapshot of Market Data detail of bids and offers for security ZYL. There are no buy orders and 1 sell order in the market for the security.

Request Message – from Client (Market Data Request):

```
8=FIX.4.4|9=104|35=V|49=Client2|56=ASX|34=20|52=20080102-
03:45:02|262=CLong2|263=0|264=0|267=2|269=0|269=1|146=1|55=ZYL|10=209|
```

Response Message – from Server (Market Data - Snapshot):

```
8=FIX.4.4|9=138|35=W|49=ASX|56=Client2|34=13|52=20080102-
03:45:04|262=CLong2|55=ZYL|268=1|269=1|278=1|270=999990|271=999999999|
37=S-ZYL-6B605440:47AAF12B|10=207|
```

W: Market Data – Snapshot (from server)

The W*: Market Data – Snapshot* message is sent as a response to a V: *Market Data Request* message with *SubscriptionRequestType* = *0* specified (a request for a snapshot).

Notes:

- In all cases each W*: Market Data – Snapshot* message corresponds to a single V: *Market Data Request*. The message echoes tag 262 *MDReqID* sent in the V: Market Data Request message.
- The W*: Market Data – Snapshot* message format contains the entirety of the data requested for that instrument, not just the changed Market Data Entry.
- A W*: Market Data* entry will be sent with NoMDEntries (268) = 0 if there is no value in the market. For example, if a request is for best bid only and there is no best bid, the response will contain NoMDEntries (268) = 0.
- If an Index is not being updated, e.g. data is not received from index calculation, then an index value of 0 is disseminated to indicate that the index value is not being updated. The value 0 would then be sent in MDEntryPx (270) for the index. An index does not have an MDEntrySize (271) value.

W: Market Data – Snapshot

Tag	Field Name	Rqd	Format	Definition
	Standard Header	Y		MsgType = W
262	MDReqID	Y	String	Mandatory. This message is always a response to a Market Data Request message. MDReqID will be echoed from the request message.
55	Symbol	Y	String	Ticker symbol. Common, "human understood" representation of the security. Mandatory because MDEntryRefID is not used.
268	NoMDEntries	Y	Integer	Number of entries following.

Tag	Field Name	Rqd	Format	Definition
				If the Market Data Entries requested by the client do not exist this field will be sent back as 0.
→ 269	MDEntryType	Y	Char	Mandatory if *NoMDEntries* is non-zero. Valid values: 0 = Bid (returns price and quantity) 1 = Offer (returns price and quantity) 2 = Trade (returns price and quantity) 3 = Index value (returns index value as a price) 4 = Opening Price (returns price) 5 = Closing Price (returns price) 7 = Trading Session High Price (returns price) 8 = Trading Session Low Price (returns price) B = Trade Volume (returns total volume traded) H = Cumulative Value (returns price) I = Indicative Auction Price J = Surplus Volume (returns quantity and side). *price* is returned in the field MDEntryPx (tag 270). *quantity* is returned in the field MDEntrySize (tag 271). *total volume traded* is returned in the field TotalVolumeTraded (tag 387). *side* is returned as an imbalance flag in the field TradeCondition (tag 277)
7956	UndisclosedQtyIndicator	Y	Character	Valid Values: Y = Undisclosed Quantity is present N = Undisclosed Quantity is not present
1070	MDQuoteType	Y	Integer	0= Indicative 1 = price_type_equilibrium i.e. Normal IAP/ Equilib Price 2 = matching_price_type_fixed i.e. Volume Match Fixed Price

Tag	Field Name	Rqd	Format	Definition
→ 278	MDEntryID	N	String	Unique Market Data Entry identifier. Each depth entry in the market is identified by a unique ID. Only sent when overlay buffering is configured in the serverconfig.xml file.
→ 270	MDEntryPx	N	Price	Price per share Or Index Value for *MDEntryType* = Index Value (3). Prices are sent in tenths of a cent, e.g. $1.23 is sent as 1230. For indices and EINs the range is from 0.0001 to 99999.9999. This field is sent as 0 when index value is requested but no value is available for the index requested. This tag can be zero for the associated MDEntryType vales 0 (bid) & 1 (offer) This tag is not returned if there is no value for the associated MDEntryType. When detailing an EIN the EIN value is sent.
→ 271	MDEntrySize	N	Qty	Total disclosed quantity at the price.
→ 37	OrderID	N	String	Unique identifier for Order as assigned by sell-side (broker, exchange, ECN). Uniqueness must be guaranteed within a single trading day. Only sent when - the request was for market detail; and - MDEntryType = 0 (Bid) or 1 (Offer)
→ 277	TradeCondition	N	Char	Space-delimited list of conditions describing a trade & as an imbalance flag to indicate side of surplus volume. Valid values: P = Imbalance More Buyers (Cannot be used in combination with Q) Q = Imbalance More Sellers (Cannot be used in combination with P)

Tag	Field Name	Rqd	Format	Definition
				This field will be sent when Surplus Volume is non-zero.
→ 387	TotalVolumeTraded	N	Qty	Total volume traded for this security. Part of Trade Market Data Entry. May be zero.
	Standard Trailer	Y		

FIX Version Support: 4.2 and 4.4

X: Market Data – Incremental Refresh (from server)

The X: *Market Data – Incremental Refresh* message is sent as a response to a V: *Market Data Request* message that has *SubscriptionRequestType = 1* specified (a request for a snapshot plus updates) or *SubscriptionRequestType* = 2 (a request to unsubscribe). It is also sent unsolicited if a user has subscribed to market data and there is a broadcast received for that market data.

Notes:

- The initial X: *Market Data – Incremental Refresh* message contains snapshot information for the specified instrument. Each Market Data Entry in this snapshot has MDUpdateAction (tag 279) = 0 (New).

- Subsequent X: *Market Data – Incremental Refresh* messages contain update information related to one or more Market Data Entries.

- A new Market Data Entry defaults to the same instrument as the previous Market Data Entry in the same X: *Market Data – Incremental Refresh* message. This means the instrument is specified only for the first Market Data Entry within the Market Data Entry group.

- For Unsubscribe Acknowledgments the following fields are sent:

    - 262 *MDReqID* (same as original request),
    - 268 *NoMDEntries* (0).

- A Market Data Entry will not be sent if there is no value in the market; e.g. if request for top of market bids only, and there are no bids, the response will contain NoMDEntries (tag 268) = 0.

X: Market Data – Incremental Refresh

Tag	Field Name	Rqd	Format	Definition
	Standard Header	Y		MsgType = X
262	MDReqID	Y	String	Mandatory. This will echo MDReqID from the V: *Market Data Request* message.
268	NoMDEntries	Y	Integer	Number of entries following. It is sent back as 0 if no market data entries exist.
→	MDUpdateAction	Y	Char	Mandatory if *NoMDEntries* is non-zero.

Tag	Field Name	Rqd	Format	Definition
279				Must be first field in this repeating group. Valid values: 0 = New 1 = Change 2 = Delete
→ 269	MDEntryType	Y	Char	Mandatory if *NoMDEntries* is non-zero. Valid values: 0 = Bid (returns price and quantity) 1 = Offer (returns price and quantity) 2 = Trade (returns price and quantity) 3 = Index value (returns index value as a price) 4 = Opening Price (returns price) 5 = Closing Price (returns price) 7 = Trading Session High Price (returns price) 8 = Trading Session Low Price (returns price) B = Trade Volume (returns total volume traded) H = Cumulative Value (returns price) I = Indicative Auction Price J = Surplus Volume (returns quantity and side). *price* is returned in the field MDEntryPx (tag 270). *quantity* is returned in the field MDEntrySize (tag 271). *total volume traded* is returned in the field TotalVolumeTraded (tag 387). *side* is returned as an imbalance flag in the field TradeCondition (tag 277)
7956	UndisclosedQtyIndicator	Y	Character	Valid Values: Y = Undisclosed Quantity is present N = Undisclosed Quantity is not present
1070	MDQuoteType	Y	Integer	0= Indicative 1 = price_type_equilibrium i.e. Normal IAP/ Equilib Price 2 = matching_price_type_fixed i.e. Volume Match Fixed Price
→ 55	Symbol	Y	String	Mandatory because *MDEntryRefID* is not used. Ticker symbol. Common, “human

Tag	Field Name	Rqd	Format	Definition
				understood" representation of the security.
→ 346	NumberOfOrders	N	Integer	Number of orders in the market.
→ 278	MDEntryID	N	String	Unique Market Data Entry identifier. Each depth entry in the market is identified by a unique ID. Only sent when the request was for market depth or market detail. Not sent for top of market data.
→ 270	MDEntryPx	N	Price	Price per share Or Index Value for *MDEntryType* = Index Value (3). Prices are sent in tenths of a cent, e.g. $1.23 is sent as 1230. For indices and EINs the range is from 0.0001 to 99999.9999. This field is sent as 0 when index value is requested but no value is available for the index requested. This tag can be zero for the associated MDEntryType vales 0 (bid) & 1 (offer) This tag is not returned if there is no value for the associated MDEntryType. When detailing an EIN the EIN value is sent.
→ 271	MDEntrySize	N	Qty	Not sent where *MDUpdateAction* (279) = 2. Otherwise conditionally required if *MDEntryType* has one of the following values - Bid (0) - Offer (1) - Trade (2) - Indicative Auction Price (I) - Surplus Volume (J) Not sent otherwise. May be zero in messages from ASX.
→	MDEntryPositionNo	N	Integer	Display position of a bid or offer, numbered

Tag	Field Name	Rqd	Format	Definition
290				from most competitive to least competitive, per market side, beginning with 1 Not sent when the request was for top of market data (MarketDepth = 1).
→ 387	TotalVolumeTraded	N	Qty	Total volume traded for this security. Part of Trade Market Data Entry. Not sent where *MDUpdateAction* (279) = 2. May be zero.
→ 37	OrderID	N	String	Unique identifier for Order as assigned by sell-side (broker, exchange, ECN). Uniqueness must be guaranteed within a single trading day. Only sent when - the request was for detail (unaggregated depth); and - MDEntryType = 0 (Bid) or 1 (Offer)
	Standard Trailer	Y		

FIX Version Support: 4.2 and 4.4

Y: Market Data – Request Reject (from server)

The Y: *Market Data Request Reject* message is sent when the *Market Data Request* cannot be honoured.

Y: Market Data Request Reject

Tag	Field Name	Rqd	Format	Definition
	Standard Header	Y		MsgType = Y
58	Text	N	String	Error text, free form text. Refer Appendix F
262	MDReqID	Y	String	Mandatory. This will echo MDReqID from the V: *Market Data Request* message.
1070	MDQuoteType	Y	Integer	0= Indicative 1 = price_type_equilibrium i.e. Normal IAP/ Equilib Price 2 = matching_price_type_fixed i.e. Volume Match Fixed Price

Tag	Field Name	Rqd	Format	Definition
281	MDReqRejReason	N	Character	Reject reason code. Valid values: 0 = Unknown symbol 1 = Duplicate MDReqID 3 = Insufficient permissions 4 = Unsupported SubscriptionRequestType 5 = Unsupported MarketDepth 6 = Unsupported MDUpdateType 8 = Unsupported MDEntryType
	Standard Trailer	Y		

FIX Version Support: 4.2 and 4.4

e: Security Status Request (from client)

The e: *Security Status Request* message is used to obtain the status of a security as a snapshot or a snapshot plus updates. Only one subscription per security is permitted.

If *SubscriptionRequestType (263)* = *0* (a request for a snapshot) an f: Security Status message containing snapshot information will be sent. If the request cannot be honoured a Request/Reject message will be sent.

If *SubscriptionRequestType (263)* = *1* (a request for snapshot plus updates) an initial f: *Security Status* message will be sent containing snapshot information followed by Security Status refresh messages containing security status updates as those updates occur. If the subscription cannot be honoured a *Security Status – Request/Reject* message will be sent.

If *SubscriptionRequestType (263)* = 2 (unsubscribe) an f: *Security Status* – message without update information will be sent. No further updates will then be received for the subscription that was unsubscribed. If the unsubscription cannot be honoured a *Security Status – Request/Reject* message will be sent.

For any other value of *SubscriptionRequestType* an f: *Security Status – Request/Reject* message will be sent.

e: Security Status Request

Tag	Field Name	Rqd	Format	Definition
	Standard Header	Y		MsgType = e (lower case)
324	SecurityStatusReqID	Y	String	Must be unique within the FIX connection, or if *SubscriptionRequestType* = Disable Previous Snapshot and Updates Request (2) it must be the ID of a previous Security Status Request to disable. ASX validation: Minimum string length is 1, maximum is 255 characters.
55	Symbol	Y	String	If *SubscriptionRequestType* = 2, must match code requested in previous *SecurityStatusReqID*. Ticker symbol. Common, "human understood" representation of the security.
263	SubscriptionRequestType	Y	Character	*SubscriptionRequestType* indicates to the other party what type of response is expected.

Tag	Field Name	Rqd	Format	Definition
				0 = Snapshot – only requests current information 1 = Snapshot and Updates (Subscribe) requests current information plus updates as the status change 2 = Disable previous Snapshot and Updates Request (Unsubscribe) cancels any future update messages from the counter party
1070	MDQuoteType	Y	Integer	0= Indicative 1 = price_type_equilibrium i.e. Normal IAP/ Equilib Price 2 = matching_price_type_fixed i.e. Volume Match Fixed Price
	Standard Trailer	Y		

FIX Version Support: 4.2 and 4.4

e: Security Status Request Message Example 1

A FIX 4.4 client subscribes to Security Status data for the equity AIA.

Request Message – from Client (Security Status Request):

```
8=FIX.4.4|9=72|35=e|49=Client2|56=Server|34=3|52=20061009-
01:28:28|324=A0|55=AIA|263=1|10=018|
```

Response Messages – from Server (Security Status):

```
8=FIX.4.4|9=99|35=f|49=Server|56=Client2|34=3|52=20061009-
01:28:30|324=A0|55=AIA|326=17|625=OPEN|332=1|333=1|31=1|
```

f: Security Status (from server)

The f: *Security Status* message is used to send the status of a security. A snapshot or a snapshot plus updates can be requested.

This message will always be sent in response to a Security Status Request message.

Security status information includes:
- Corporate action information (e.g. Ex Dividend).
- Exchange action information (e.g. Trading Halt).
- Security type information (e.g. Warrants).
- Security trading session status information (e.g. Pre-Open).

In this message the following fields will take Stock Exchange specific values:
- *CorporateAction* FIX field (292) – for corporate action information
- *SecurityTradingStatus* FIX field (326) – for exchange action information

For unsubscribe acknowledgments, where SubscriptionRequestType (263) = 2 the following is sent:
- SecurityStatusReqID (same as original request)
- Symbol (same as original request)

For Security Status rejects the following is sent:
- SecurityStatusReqID (same as original request)
- Symbol (same as original request)
- Text (containing a description of the reason why the request was rejected)

Notes:

In the scenario when a stock is in pre-open and the market is in close, the value that is disseminated in the SecurityTradingStatus field is the one having the higher priority on the ASX trading system back-end, ASX Trade™. (ASX Trade™ is the name of ASX's Trading System).

The value is calculated based on the logic used by ASX Trade™ (as detailed in ASX Trade™ Open Interface Manual).

If the priority of the Instrument Session State is higher than or equal to the priority of the Trading Session State, then the Active State equals the Instrument Session State.

If the priority of the Instrument Session State is lower than the priority of the Trading Session State, then the Active State equals the Trading Session State.

The following psuedocode implements the algorithm:

TSS: (strictly hierarchical)
Active_TSS := (TSS of the Market of the Series);
If (TSS of Instrument Type of Series) != nil

Active_TSS := (TSS of Instrument Type of Series);
If (TSS of Instrument Class of Series) != nil

Active_TSS := (TSS of Instrument Class of Series);
ISS: (priority based)

Active_ISS := (ISS of Series);
If (ISS of Underlying of Series) > Active_ISS

Active_ISS := (ISS of Underlying of Series);
If (ISS of LinkedUnderlying of Series) > Active_ISS

Active_ISS := (ISS of LinkedUnderlying of Series);
TSS_and_ISS: (priority based)
If Active_ISS >= Active_TSS
  ActiveState := Active_ISS;
Else
  ActiveState := Active_TSS;

Therefore if CLOSE is of a higher priority than PRE-OPEN, then CLOSE (STATUS_EndOfSession = 18) is disseminated in the SecurityTradingStatus field in this case.

f: Security Status

Tag	Field Name	Rqd	Format	Definition
	Standard Header	Y		MsgType = f (lower case)
324	SecurityStatusReqID	Y	String	The same value as the request message.
55	Symbol	Y	String	Ticker symbol. Common, "human understood" representation of the security. If SubscriptionRequestType = 2, must match code requested in previous SecurityStatusReqID.
326	SecurityTradingStatus	N	Integer	The trading status applicable to the security. For translation and mapping see Appendix C.

Tag	Field Name	Rqd	Format	Definition
				Valid values: 2 17 18 21 100 (custom value) 101 (custom value) 102 (custom value) 103 (custom value) 104 (custom value) 105 (custom value) 106 (custom value) 107 (custom value)
292	CorporateAction	N	Character	Corporate action information. Sends snapshot of current values. For translation see Appendix C ASX validation: A security can have a maximum total of 5 corporate actions. Valid values: NR RE SH XB XC XD XE XF XI XR Please refer to Appendix for mapping and interpretation.
332	HighPx	N	Price	Represents an indication of the high end of the price range for a security. Value is sent in tenths of a cent, e.g. $1.23 is sent as 1230.
333	LowPx	N	Price	Represents an indication of the low end of the price range for a security. Value is sent in tenths of a cent.
31	LastPx	N	Price	Last trade price sent in tenths of a cent.

Tag	Field Name	Rqd	Format	Definition
58	Text	N	String	Error text, free form text. Refer Appendix F Required for a reject.
	Standard Trailer	Y		

FIX Version Support: 4.2 and 4.4

g: Trading Session Status Request (from client)

This message is not currently supported but may be supported in the future.

The g: *Trading Session Status Request* message is used to request information on the status of a market. A snapshot or a snapshot plus updates can be requested. Only one subscription is permitted.

Expect an h: *Trading Session Status* message.

If *SubscriptionRequestType (263)* = 0 (a request for a snapshot) a Trading Session Status message containing snapshot information will be sent. If the request cannot be honoured a Request/Reject message will be sent.

If *SubscriptionRequestType (263)* = *1* (a request for snapshot plus updates) an initial Trading Session Status message will be sent containing snapshot information followed by Trading Session Status Incremental refresh messages containing updates as those updates occur. If the subscription cannot be honoured a Trading Session Status– Request/Reject message will be sent.

If *SubscriptionRequestType (263)* = 2 (unsubscribe) a Trading Session Status– Incremental Refresh message without update information will be sent. No further updates will then be received for the subscription that was unsubscribed. If the unsubscription cannot be honoured a Trading Session Status– Request/Reject message will be sent.

For any other value of *SubscriptionRequestType* a Trading Session Status containing tag 58 Text with an error message will be sent.

While the FIX server application is running and accepting logons a client will always be able to subscribe to, and maintain, the trading session status; regardless of whether the market is available.

g: Trading Session Status Request

Tag	Field Name	Rqd	Format	Definition
	Standard Header	Y		MsgType = g (lower case)
335	TradSesReqID	Y	String	Must be unique within the FIX connection, or the ID of previous Trading Session Status Request to disable if *SubscriptionRequestType* = Disable previous Snapshot and Updates Request (2). ASX validation: Minimum string length is 1, maximum is 255 characters.
263	SubscriptionRequestType	Y	Character	*SubscriptionRequestType* indicates to the other party what type of response is expected. 0 = Snapshot – only requests current information 1 = Snapshot and Updates (Subscribe) requests current information plus updates as the status change 2 = Disable previous Snapshot and Updates Request (Unsubscribe) cancels any future update messages from the counter party
	Standard Trailer	Y		

FIX Version Support: Currently not supported

h: Trading Session Status (from server)

This message is not currently supported but may be supported in the future.

The h: *Trading Session Status* message provides information on the status of a market. This message is sent as a response to a g: *Trading Session Status Request* message requesting a snapshot. It can also be sent unsolicited to provide market status updates or when the g: *Trading Session Status Request* message cannot be honoured.

In this message the *TradSesStatus* field (340) field takes exchange specific values: 1-11 inclusive and 99.

For Trading Session Status rejects, this message contains:

- TradSesReqID (same as original request)
- TradingSessionID (NONE)
- ExExchangeSpecificValues (Y)
- TradSesStatus (99 – Unknown or Invalid)
    - Text (reject reason).

For Unsubscribe Acknowledgments this message contains:

- TradSesReqID (same as original request)
- TradingSessionID (current trading cycle)
- ExExchangeSpecificValues (Y)
- TradSesStatus (current status).

h: Trading Session Status

Tag	Field Name	Rqd	Format	Definition
	Standard Header	Y		MsgType = h (lower case)
58	Text	N	String	Mandatory for rejects, not sent otherwise. Refer Appendix F
325	UnsolicitedIndicator	N	Boolean	'Y' if message is sent unsolicited as a result of a previous subscription request. Not sent otherwise.

Tag	Field Name	Rqd	Format	Definition
335	TradSesReqID	N	String	Not sent in unsolicited messages, otherwise always present. When solicited the TradSesReqID is echoed back from g: Trading Session Status Request message. This is a mandatory field. See Appendix C for mapping.
336	TradingSessionID	Y	String	Identifier for market trading session. Valid values: AGRIC AUS INDEX EQTY1 EQTY2 EQTY3 EQTY4 EQTY5 INDX WAR IRM PRAC PRV ABB VM Please refer to Appendix C for mapping.
339	TradSesMode	N	Integer	Trading Session Mode. Valid values: 1 = Testing 2 = Simulated 3 = Production

Tag	Field Name	Rqd	Format	Definition
340	TradSesStatus	Y	Integer	Market status information. See Appendix C for mapping. Value '99' – Unknown or Invalid' is sent in Trading Session Status rejects. Valid values: 2 3 4 100 (custom value) 1 4 2 101 (custom value) 102 (custom value) 103 (custom value) 104 (custom value) 4 3 4 105 (custom value) 3 3
	Standard Trailer	Y		

FIX Version Support: Currently not supported

Appendix A – ASX Contact Details

Product Content & System Support

Subscribers with data content and production problem queries can contact the Trading and Market Information Support Team for customer support from 08:00 hours to 18:00 hours on ASX designated trading days on the following numbers:

1800 663 053

OR

(+61 2) 9227-0347
(+61 2) 9227-0956

OR via email to: Market.Information@asx.com.au

Subscribers requiring after-hours support for production problems can receive assistance on +61 2 9227 0821. Data content queries are not supported after hours.

Written queries may be addressed to:

Trading and Market Information Support
Australian Securities Exchange
P.O. Box H224
Australia Square
SYDNEY
NSW 2000

Or sent by facsimile to: (+61 2) 9227-0859

Sales & Business Development Queries

Houda Harb, Product Analyst
houda.harb@asx.com.au or Phone: +61 2 9227 0422

Appendix B – Market Data for Trades and Orders

Trade data:

Every time a trade occurs on ASX's ASX Trade™ Trading System, updates to Last Price, Traded Volume, Open, High and Low will be sent.

They will be sent to FIX users in the form of a **FIX Market Data Incremental Refresh message**, assuming the FIX user has subscribed for an incremental refresh.

Order data:

Every time there is a change in the bid, ask, number of buyers, number of sellers, an updated **FIX Market Data Incremental Refresh message** will be sent to the FIX user with updated bid and ask.

Appendix C – ASX Custom Valid Values for FIX fields

FIX tag 326 SecurityTradingStatus standard and non-standard valid values are:

ASX Trade™ Session State	FIX Value
OPEN	17
CLOSE	18
PRE_OPEN	21
LATE_TRADING	100 (custom value)
TRADING_HALT	2
PRE_NIGHT-TRADING	101 (custom value)
OPEN_NIGHT-TRADING	102 (custom value)
SUSPEND	103 (custom value)
ABB_AUCTION	104 (custom value)
ADJUST	105 (custom value)
CSPA	106 (custom value)
PRE_NR	21
ENQUIRE	18
PRE_CSPA	21
ADJUST_ON	107 (custom value)
OPEN_QUOTE-DISPLAY	114 (custom value)
OPEN_VMB	112 (custom value)
WAIT_VMB	113 (custom value)

Table above ordered by FIX value:

ASX Trade™ Session State	FIX Value
TRADING_HALT	2
OPEN	17
CLOSE	18
ENQUIRE	18
PRE_OPEN	21
PRE_NR	21
PRE_CSPA	21
LATE_TRADING	100 (custom value)
PRE_NIGHT-TRADING	101 (custom value)
OPEN_NIGHT-TRADING	102 (custom value)
SUSPEND	103 (custom value)
ABB_AUCTION	104 (custom value)
ADJUST	105 (custom value)
CSPA	106 (custom value)
ADJUST_ON	107 (custom value)
OPEN_QUOTE-DISPLAY	114 (custom value)
OPEN_VMB	112 (custom value)
WAIT_VMB	113 (custom value)

FIX tag 292 CorporateAction standard and non-standard valid values are:

Corporate Action ID (this is the value sent in FIX tag 292)	Name
NR	Notice Received
RE	Reconstructed
SH	Short Sell permitted for this security
XB	Ex Bonus Issue
XC	Ex Return of Capital
XD	Ex Dividend
XE	Ex Entitlement
XF	Ex Takeover Offer
XI	Ex Interest
XR	Ex Rights Issues

FIX tag 336 TradingSessionID standard and non-standard valid values are:

Market ID (this is the value sent in FIX tag 336)	Name
VMB	VolumeMatch Book
QDB	Quote Display Board
EQTY1	Equity Market Group 1 (A-B)
EQTY2	Equity Market Group 2 (C-F)
EQTY3	Equity Market Group 3 (G-M)
EQTY4	Equity Market Group 4 (N-R)
EQTY5	Equity Market Group 5 (S-Z)
IRM	Interest Rate Market
WAR	Warrants Market
INDX	Stock Index Market
INDEX	Index Derivatives Market
AUS	Stock Derivatives Market
AGRIC	Agricultural Derivatives Market
ABB	ASX Bookbuild
PRAC	Practice Market

FIX tag 340 TradSesStatus standard and non-standard valid values are:

Session State Name	FIX Value
OPEN	2
CLOSE	3
PRE_OPEN	4
LATE_TRADING	100 (custom value)
TRADING_HALT	1
PRE_NIGHT-TRADING	4
OPEN_NIGHT-TRADING	2
SUSPEND	101 (custom value)
ABB_AUCTION	102 (custom value)
ADJUST	103 (custom value)
CSPA	104 (custom value)
PRE_NR	4
ENQUIRE	3
PRE_CSPA	4
ADJUST_ON	105 (custom value)
SYSTEM_MAINTENANCE	3
PURGE_ORDER	3
OPEN_QUOTE-DISPLAY	114 (custom value)
OPEN_VMB	112 (custom value)
WAIT_VMB	113 (custom value)

Table above ordered by FIX value:

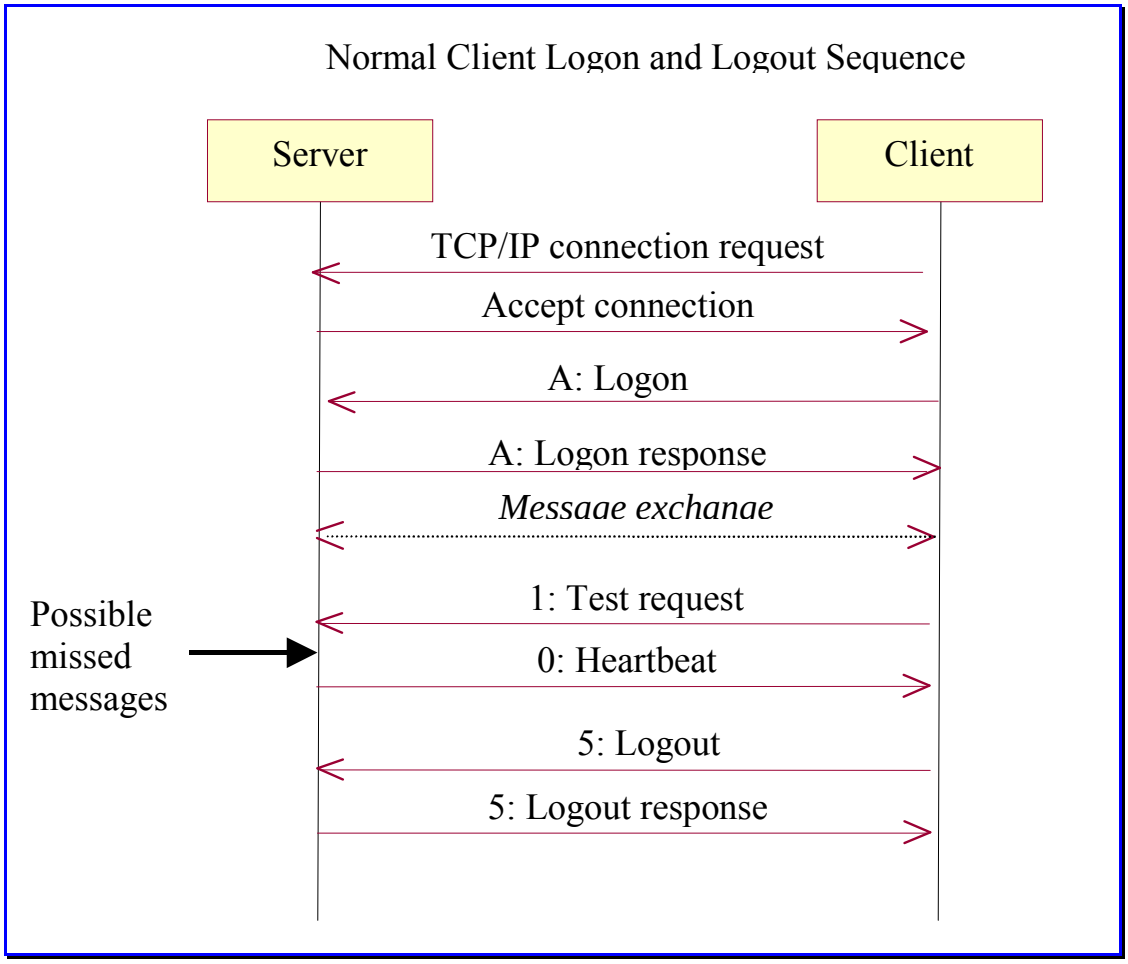
Session State Name	FIX Value
TRADING_HALT	1
OPEN	2
OPEN_NIGHT-TRADING	2
CLOSE	3
ENQUIRE	3
SYSTEM_MAINTENANCE	3
PURGE_ORDER	3
PRE_OPEN	4
PRE_NIGHT-TRADING	4
PRE_NR	4
PRE_CSPA	4
LATE_TRADING	100 (custom value)
SUSPEND	101 (custom value)
ABB_AUCTION	102 (custom value)
ADJUST	103 (custom value)
CSPA	104 (custom value)
ADJUST_ON	105 (custom value)
OPEN_QUOTE-DISPLAY	114 (custom value)
OPEN_VMB	112 (custom value)
WAIT_VMB	113 (custom value)

Appendix D – FIX Order of Operations

Client logon and logout

The diagram below shows the message flow when a client establishes a FIX connection with the server.

Neither party should send any queued or new application messages until the process of authentication and synchronisation has been completed. The client will wait for the Logon confirmation message and for the completion of any resend processing required for gap recovery before sending any queued or new application messages.



The message flow for a normal log on and log off is:

1. **Establishing telecommunications link.** The client and server establish a socket connection over TCP/IP at the request of the client application.

2. **Logon.** The client sends a Logon message containing the user name and password.

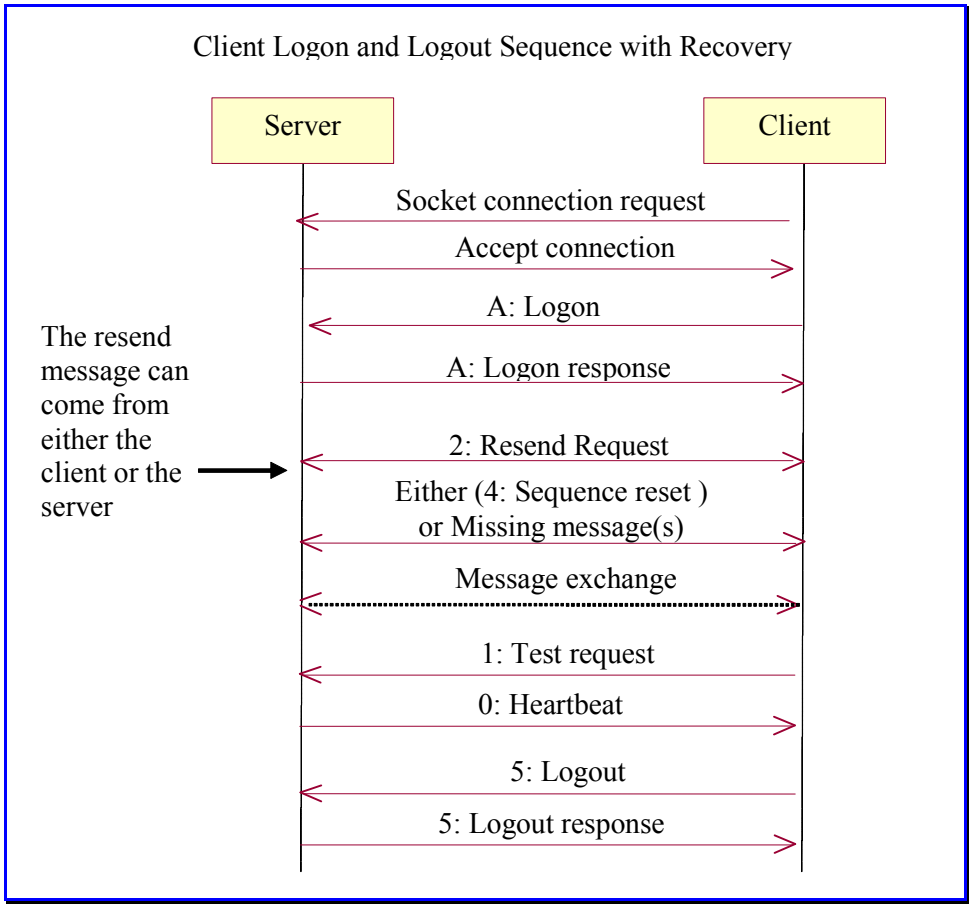
3. **Logon Confirmation.** If the authentication process is successful the server sends a Logon confirmation message.

4. **Message Exchange.** When all synchronisation processing has completed the client can send any queued or new application messages.

5. **Client Test Request.** Prior to Logout the client sends a Test Request message. The client uses the Heartbeat response as confirmation that the server has completed any necessary resend processing.

6. **Client Logout.**

7. **Logout Confirmation.** The server sends a Logout confirmation message. The server then disconnects the socket connection to the client.



The message flow for a log on and log off where message recovery is necessary is as follows:

1. **Establishing telecommunications link.** The client and server establish a socket connection over TCP/IP at the request of the client application.

2. **Logon.** The client sends a Logon message containing the user name and password.

3. **Logon Confirmation.** If the authentication process is successful the server sends a Logon confirmation message.

4. **Server Resend Processing.** If the server or the client detects a message gap from inspection of the *MsgSeqNum* field in the Logon request, it will send a 2: Resend request and receive missing messages.

5. **Server Heartbeat.** On completion the server sends a Heartbeat message in response to the client's Test Request, indicating it has completed resend processing.

6. **Message Exchange.** When all synchronisation processing has completed the client can send any queued or new application messages.

7. **Client Test Request.** Prior to Logout the client sends a Test Request message. The client uses the Heartbeat response as confirmation that the server has completed any necessary resend processing.

8. **Client Logout.** The client sends a Logout message.

9. **Logout Confirmation.** The server sends a Logout confirmation message. The server then disconnects the socket connection to the client.

Appendix E – Field Definitions

Following is a list of field definitions for all the fields that will be supported by ASX in its implementation of FIX for Market Data. These fields are ordered by field tag number and they provide details of field size, format, description and valid values.

Please note that for all field definitions:

- Whenever a field shows a blank, it contains the ASCII space character (hex 20)
- All string fields are left-justified and null terminated, unless otherwise stated.
- All messages have a fixed format.
- All times disseminated are based on a 24 hour clock and are UTC times.

Tag	Field Name	Format	Definition	Comments
	Standard Header			This is prefixed to all incoming and outgoing FIX messages. Refer to FIX specifications for full details.
	Standard Trailer			This is prefixed to all incoming and outgoing FIX messages. Refer to FIX specifications for full details.
8	BeginString	String	Protocol versions. Valid Values : FIX.4.2 FIX.4.4	Users must populate this tag with the FIX version they are using for validation purposes. Otherwise errors will ensue. The comp ID must be mapped to the FIX version in the configuration file
9	BodyLength	Integer 2-byte unsigned	Length of body not including header/trailer.	
10	CheckSum	String	Byte sum modulo 256, for all fields up to but not including the checksum. Send as 3 ASCII digits with leading 0s.	

Tag	Field Name	Format	Definition	Comments
34	MsgSeqNum	Integer 4-byte unsigned	Message sequence number, cannot be zero, starts at 1 each SOD. Refer to FIX Specification V4.2 message format and delivery, sequence number.	
35	MsgType	String	Message type. Valid values: 0 = Heartbeat 1 = Test Request 2 = Resend Request 3 = Reject 4 = Sequence Reset 5 = Logout A = Logon BE = User Response BF= User Request V = Market Data Request W = Market Data-Snapshot X = Market Data-Incremental Refresh Y = Market Data Request Reject e = Security Status Request f = Security Status g = Trading Session Status Request h = Trading Session Status j = Business Message Reject	
37	OrderID	String	Unique identifier for Order as assigned by sell-side (broker, exchange, ECN). Uniqueness must be guaranteed within a single trading day. Firms which accept multi-day orders should consider embedding a date within the OrderID field to assure uniqueness across days.	
43	PossDupFlag	Boolean	Indicates possible retransmission of message with this sequence number. Valid values: Y = Possible duplicate N = Original transmission If not specified, defaults to 'N'.	
49	SenderCompID	String	Sender company ID. ASX assigns this ID to the customer. Each customer has a unique SenderCompID.	
52	SendingTime	UTC Time Stamp	Time of message transmission. Valid format YYYYMMDD-HH:MM:SS	
55	Symbol	String	Ticker symbol. Common, "human understood" representation of the security.	
56	TargetCompID	String	Target company ID. Assigned value used to identify receiving firm. This is the value the FIX end user sends to identify the ASX. This value is always set to "ASX".	
58	Text	String	Free format text string (Note: this field does not have a specified maximum length)	Error text, free form text. Refer Appendix F

Tag	Field Name	Format	Definition	Comments
97	PossResend	Boolean	Indicates that message may contain information that has been sent under another sequence number. Valid values: Y = Possible resend N = Original Transmission If not specified, defaults to 'N'.	
122	OrigSendingTime	UTC Time Stamp	Original time of message transmission. Mandatory for message resends. Valid format YYYYMMDD-HH:MM:SS Always expressed in UTC (Universal Time Coordinated) also known as "GMT", when transmitting orders as the result of a resend request.	
123	GapFillFlag	Integer	Indicates if Gap Fill mode or Reset mode is to be used. Valid values: Y = Gap Fill mode N = Reset mode	
146	NoRelatedSym	Integer	Number of instruments requested.	
262	MDReqID	String	Unique identifier for Market Data Request message. Must be unique within the FIX connection, or the ID of previous Market Data Request to disable if SubscriptionRequestType = Disable previous Snapshot and Updates Request (2). ASX validation: Minimum string length is 1, maximum is 255 characters.	
263	SubscriptionRequestType	Character	SubscriptionRequestType indicates to the other party what type of response is expected. 0 = Snapshot – only requests current information 1 = Snapshot and Updates (Subscribe) requests current information plus updates as the status change 2 = Disable previous Snapshot and Updates Request (Unsubscribe) cancels any future update messages from the counter party	

Tag	Field Name	Format	Definition	Comments
264	MarketDepth	Integer	Depth of market. Valid values: 0 = Full Book (market detail / Level 2) 1 = Top of Book (Level 1) 5 = 5 levels of Market Depth N>1 = Report best N price tiers of data One of the advantages of using FIX to retrieve Equity market data, is that the FIX engine uses Equity detail provided by ASX Trade™ to build Equity depth for the FIX end user. You have an added configurability option with FIX in that you can request from 0 up to N levels of Equity depth. On the other hand using OMX Genium INET[SM] API (the native language of the ASX's ASX Trade™ Trading System) to retrieve Equity market data, the top 5 levels of Equity depth are available but only to Sydney and Melbourne users in Australia. Note that this restriction does not apply when using FIX because of the fact that you have the flexibility of being able to request from 0 to N levels of Equity depth. For Derivatives, the top 5 levels of Derivatives depth is provided using the OMX Genium INET[SM] API whereas using FIX, you have the flexibility of being able to request anything from the top 2 levels of Derivatives depth to the top 5 levels of Derivatives depth.	
265	MDUpdateType	Integer	Specifies the type of Market data update. Mandatory if SubscriptionRequestType = Snapshot and Updates (1). Not permitted otherwise. Valid values: 1 = Incremental Refresh	
266	AggregratedBook	Boolean	Specifies whether or not book entries should be aggregated. Valid values: Y = one book entry per side per price N = Multiple entries per side per price allowed Note that the default value for this tag is N (Detail). So if you want to subscribe for DEPTH (Derivative stock) you have to send tag 266=Y. If you do not set this field to 266=Y for a Derivatives stock, you will get an error Subscription failed: Status = -39.	In FIX version 4.4, this is an optional field in the V: Market Data Request message sent by the FIX client.
267	NoMDEntryTypes	Integer	Number of MDEntryType fields requested. For unsubscriptions (tag 263 = 2) NoMDEntryTypes must be zero, otherwise must be > zero.	

Tag	Field Name	Format	Definition	Comments
268	NoMDEntries	Integer	Number of entries following within Market Data message. If the Market Data Entries requested by the client do not exist in the market then this field is sent back as zero.	
269	MDEntryType	Character	Valid values: 0 = Bid (returns price and quantity) 1 = Offer (returns price and quantity) 2 = Trade (returns price and quantity) 3 = Index value (returns index value as a price) 4 = Opening Price (returns price) 5 = Closing Price (returns price) 7 = Trading Session High Price (returns price) 8 = Trading Session Low Price (returns price) B = Trade Volume (returns total volume traded) H = Cumulative Value (returns price) I = Indicative Auction Price J = Surplus Volume (returns quantity and side).	
270	MDEntryPx	Price	Price of the Market Data entry. Price per share or index/EIN value for MDEntryType = Index Value (3). Prices are sent in tenths of a cent. For indices the range is from 0.0001 to 99999.9999. This field is sent as zero when an index value is requested and when it is not yet available. When detailing an EIN, the EIN value is sent. May be zero in messages from ASX.	
271	MDEntrySize	Qty	Quantity or volume represented by the Market Data entry. Total disclosed quantity at the price. May be zero in messages from ASX.	
277	TradeCondition	String	Space-delimited list of conditions describing a trade & as an imbalance flag to indicate side of surplus volume.. Valid values: P = Imbalance More Buyers (Cannot be used in combination with Q) Q = Imbalance More Sellers (Cannot be used in combination with P) This field will be sent when Surplus Volume is sent.	
278	MDEntryID	String	Unique Market Data Entry identifier Each depth or detail entry in the market is identified by a unique ID.	

Tag	Field Name	Format	Definition	Comments
279	MDUpdateAction	char	Valid values: 0 = New 1 = Change 2 = Delete	
281	MDReqRejReason	char	Reject reason code. Valid values: 0 = Unknown symbol 1 = Duplicate MDReqID 2 = Insufficient Bandwidth 3 = Insufficient Permissions 4 = Unsupported SubscriptionRequestType 5 = Unsupported MarketDepth 6 = Unsupported MDUpdateType 7 = Unsupported AggregatedBook 8 = Unsupported MDEntryType 9 = Unsupported TradingSessionID A = Unsupported Scope B = Unsupported OpenCloseSettleFlag C = Unsupported MDImplicitDelete	
290	MDEntryPositionNo	integer	Display position of a bid or offer, numbered from most competitive to least competitive, per market side, beginning with 1	
292	CorporateAction	char	Corporate action information. Sent as a snapshot of current values. Valid values: XR XI XD SH NR XC XB RE XE XF Please refer to Appendix C for mapping and interpretation.	
324	SecurityStatusReqID	String	Unique ID of a Security Status Request message. Must be unique within the FIX connection, or if SubscriptionRequestType = Disable Previous Snapshot and Updates Request (2) it must be the ID of a previous Security Status Request to disable. ASX validation: Minimum string length is 1, maximum is 255 characters.	
325	UnsolicitedIndicator	Boolean	'Y' if message is sent unsolicited as a result of a previous subscription request. Not sent otherwise.	

Tag	Field Name	Format	Definition	Comments
326	SecurityTradingStatus	integer	Identifies the trading status applicable to the transaction. Please see Appendix C for mapping. Exchange action information. Valid values: 2 17 18 21 100 (custom value) 101 (custom value) 102 (custom value) 103 (custom value) 104 (custom value) 105 (custom value) 106 (custom value) 107 (custom value)	
332	HighPx	Price	Represents an indication of the high end of the price range for a security.	Sent in Security Status reply.
333	LowPx	Price	Represents an indication of the low end of the price range for a security.	Sent in Security Status reply.
335	TradSesReqID	String	Unique ID of a Trading Session Status message. Must be unique within the FIX connection, or the ID of previous Trading Session Status Request to disable if SubscriptionRequestType = Disable previous Snapshot and Updates Request (2). ASX validation: Minimum string length is 1, maximum is 255 characters.	

Tag	Field Name	Format	Definition	Comments
336	TradingSessionID	String	Identifier for market trading session. Valid values: AGRIC AUS INDEX EQTY1 EQTY2 EQTY3 EQTY4 EQTY5 INDX WAR IRM PRAC PRV ABB VM Please refer to Appendix C for mapping.	
339	TradSesMode	integer	Trading Session Mode. Valid values: 1 = Testing 2 = Simulated 3 = Production	
340	TradSesStatus	integer	Market status information. Refer to Appendix C for mapping and translation. Value '99' – Unknown or Invalid' is sent in Trading Session Status rejects. Valid values: 1 2 3 4 100 (custom value) 101 (custom value) 102 (custom value) 103 (custom value) 104 (custom value) 105 (custom value)	
346	NumberOfOrders	integer	Number of orders in the market.	

Tag	Field Name	Format	Definition	Comments
369	LastMsgSeqNumProcessed	integer 4-byte unsigned	The last MsgSeqNum (34) value received by the FIX engine and processed by downstream application, such as trading engine or order routing system. Can be specified on every message sent. Useful for detecting a backlog with a counterparty.	
553	Username	String	This is the ASX customer's ASX Trade™ username.	In ASX's implementation of FIX for Market Data, this field is part of the Logon message for all FIX versions that ASX is supporting – 4.2 and 4.4.
554	Password	String	This is the ASX customer's ASX Trade™ password.	In ASX's implementation of FIX for Market Data, this field is part of the Logon message for all FIX versions that ASX is supporting – 4.2 and 4.4.
1070	MDQuoteType	Integer	0= Indicative 1 = price_type_equilibrium i.e. Normal IAP/ Equilib Price 2 = matching_price_type_fixed i.e. Volume Match Fixed Price	Translation logic for differentiating Equilibrium/ IAP and VM Price types - Mapped to ASX Trade™ matching_price_type structure/ values
7956	UndisclosedQtyIndicator	Character	Valid Values: Y = Undisclosed Quantity is present N = Undisclosed Quantity is not present	

Appendix F – Reject messages

FIX Tag 58 Error Text

- Required tag "%s" is missing.
- Conditionally required tag "%s" is missing.
- Repetition %d of Repeating group "%s" is missing.
- Message type "%s" is an unknown message type
- Bad format for tag "%s": expecting %s.
- CompID problem : %s
- Incorrect data format(%s) for tag(%s)
- Incorrect value : %s
- Invalid MsgType
- Invalid Tag Number
- Tag not defined for message type : %s
- OrigSendingTime accuracy problem - Difference of %d seconds
- Required Tag(%s) Missing
- Required Tag(%s) Missing - %s
- SendingTime accuracy problem - Difference of %d seconds
- Tag(%s) Without Value
- Session is not ready
- Invalid tag value: %s=[%s]
- Invalid tag value: %s=[%s] %s
- Missing tag: [%s]
- Missing tag: [%s] %s

Examples:

Bad format for tag "21": expecting ^[1-3]$.

^ means beginning of a context

$ means end of a context

1-3 express as number 1,2,3 respectively

Therefore, this tag 21 only allows one character with range from 1 to 3.

Bad format for tag "22": expecting ^[1-9A-J]$.

tag 22 only allows one character accepting from 1 to 9 or A to J

Bad format for tag "167": expecting ^(CS|FUT|OPT|NONE)$.

tag 167 only allows CS,FUT,OPT,NONE. No head / tail space allow.