



## ASX Trade ITCH and Glimpse Specification

Q2 2015 Release – SR8

## Table of Contents

<b>1. INTRODUCTION .....</b>	<b>4</b>
1.1. ASX ITCH .....	4
1.2. GLIMPSE .....	4
<b>2. ASX ITCH.....</b>	<b>6</b>
2.1. ARCHITECTURE.....	6
2.1.1. Protocol.....	6
2.2. DATA TYPES .....	6
2.3. MESSAGE FORMATS.....	6
2.3.1. Time Messages.....	6
2.3.1.1. Seconds Message .....	7
2.4. REFERENCE DATA MESSAGES .....	7
2.4.1. Order Book Directory .....	7
2.4.2. Combination Order Book Directory .....	8
2.4.3. Tick Size Table Entry .....	10
2.5. EVENT AND STATE CHANGE MESSAGES .....	11
2.5.1. System Event Message .....	11
2.5.2. Order Book State Message.....	11
2.6. MARKET BY ORDER MESSAGES .....	12
2.6.1. Add Order Messages.....	12
2.6.1.1. Add Order – No Participant ID .....	12
2.6.1.2. Add Order With Participant ID.....	13
2.6.2. Modify Order Messages.....	14
2.6.2.1. Order Executed Message .....	14
2.6.2.2. Order Executed with Price Message .....	15
2.6.2.3. Order Replace Message .....	16
2.6.2.4. Order Delete Message.....	17
2.7. TRADE MESSAGES .....	17
2.8. AUCTION MESSAGES – EQUILIBRIUM PRICE UPDATE .....	18
2.9. BUILDING AN ORDER BOOK VIEW .....	19
2.10. BUILDING A TRADE TICKER .....	20
<b>3. GLIMPSE .....</b>	<b>21</b>
3.1. ARCHITECTURE.....	21
3.1.1. Protocol.....	21
3.2. DATA TYPES .....	21
3.3. GLIMPSE MESSAGES.....	21
3.3.1. End of Snapshot Message.....	21
<b>4. CONFIGURATION AND INFRASTRUCTURE .....</b>	<b>22</b>
4.1. CONNECTIVITY REQUIREMENTS .....	22
4.1.1. Overview of Feeds .....	22
4.1.2. Feed Address Examples.....	22
4.1.3. Configuration Details.....	23
<b>5. APPENDIX 1 – ASX ITCH SCENARIOS.....</b>	<b>26</b>
5.1. DISCLOSED ORDERS, CONTINUOUS TRADING .....	27
5.1.1. New Order Entry, Amend and Delete.....	27
5.1.2. New Order Fully Trades with Existing Order(s).....	27
5.1.3. New Order Partially Trades with Existing Order.....	27
5.1.4. Existing Order Amended and Immediately Fully Trades with Opposing Order.....	28
5.1.5. Existing Order Amended and Immediately Partially Trades with Existing Order .....	29
5.2. UNDISCLOSED ORDER, CONTINUOUS TRADING.....	30
5.2.1. Undisclosed Order Entered Into Book.....	30
5.2.2. New Order Fully Trades with Existing Undisclosed Order. Undisclosed Order Remains above the Undisclosed Threshold .....	30
5.2.3. New Order Fully Trades with Existing Undisclosed Order. Undisclosed Order Falls Below the Undisclosed Threshold .....	30

5.2.4.	<i>New Order Fully Trades with Existing Undisclosed Order. Undisclosed Order Has Fully Traded</i>	31
5.3.	ICEBERG ORDER EXECUTION, CONTINUOUS TRADING .....	31
5.3.1.	<i>New Order Fully Trades with Existing Iceberg Order. New Order Quantity Less than Iceberg Order's Shown Quantity</i> .....	31
5.3.2.	<i>New Order Fully Trades with Existing Iceberg Order. New Order Quantity Equal to Iceberg Order's Shown Quantity</i> .....	32
5.3.3.	<i>New Order Fully Trades with Existing Iceberg Order. New Order Quantity Greater than the Iceberg Order's Shown Quantity but Less than Total Quantity</i> .....	32
5.3.4.	<i>New Order Fully Trades with Existing Iceberg Order, Fully Trading out the Iceberg Order</i> .....	33
5.4.	DISCLOSED ORDER EXECUTION, AUCTIONS .....	33
5.4.1.	<i>Orders Trade in Auction</i> .....	33
5.5.	TAILOR MADE COMBINATIONS .....	34
5.5.1.	<i>New TMC Order Trades with Existing TMC Order</i> .....	34
5.5.2.	<i>New TMC Order Trades with Existing Orders in the Book</i> .....	34
5.5.3.	<i>TMC Order Trades with an Opposing TMC Order during an Auction</i> .....	36
5.6.	INSTRUMENT CREATION .....	37
5.6.1.	<i>TMC Creation</i> .....	37
5.6.2.	<i>Standard Instrument Creation</i> .....	37
5.7.	MISCELLANEOUS TASKS .....	37
5.7.1.	<i>Customer to Customer Order Transfer</i> .....	37
<b>6.</b>	<b>APPENDIX 2 – ASX MOLDUDP64 PROTOCOL SPECIFICATION .....</b>	<b>39</b>
6.1.	OVERVIEW .....	39
6.2.	ASSUMPTIONS .....	39
6.3.	TERMS .....	39
6.3.1.	<i>Message</i> .....	39
6.3.2.	<i>Session</i> .....	39
6.4.	DOWNSTREAM PACKET .....	40
6.4.1.	<i>Header</i> .....	40
6.4.2.	<i>Sequence Number</i> .....	40
6.4.3.	<i>Message Count</i> .....	40
6.4.4.	<i>Message Block</i> .....	40
6.4.5.	<i>Message Length</i> .....	40
6.4.6.	<i>Message Data</i> .....	41
6.4.7.	<i>Heartbeats</i> .....	41
6.4.8.	<i>End of Session</i> .....	41
6.4.9.	<i>Request Packet</i> .....	41
6.4.10.	<i>Sequence Number</i> .....	41
6.4.11.	<i>Requested Message Count</i> .....	41
6.5.	RECEIVER EXAMPLE .....	41
<b>7.</b>	<b>APPENDIX 3 - SOUPBINTCP PROTOCOL .....</b>	<b>43</b>
7.1.	OVERVIEW .....	43
7.1.1.	<i>SoupBinTCP Logical Packets</i> .....	43
7.1.2.	<i>Protocol Flow</i> .....	43
7.1.3.	<i>Heartbeats</i> .....	44
7.1.4.	<i>End of Session Marker</i> .....	44
7.1.5.	<i>Data Types</i> .....	44
7.2.	SOUPBINTCP PACKET TYPES .....	44
7.2.1.	<i>Debug Packet</i> .....	44
7.2.2.	<i>Logical Packets Sent by a SoupBinTCP Server</i> .....	45
7.2.2.1.	<i>Login Accepted Packet</i> .....	45
7.2.2.2.	<i>Login Rejected Packet</i> .....	45
7.2.2.3.	<i>Sequenced Data Packet</i> .....	45
7.2.2.4.	<i>Server Heartbeat Packet</i> .....	46
7.2.2.5.	<i>End of Session Packet</i> .....	46
7.2.3.	<i>Logical Packets Sent by the SoupBinTCP Client</i> .....	46
7.2.3.1.	<i>Login Request Packet</i> .....	46
7.2.3.2.	<i>Un-sequenced Data Packets</i> .....	47
7.2.3.3.	<i>Client Heartbeat Packets</i> .....	48
<b>8.</b>	<b>DOCUMENT VERSION CONTROL .....</b>	<b>49</b>

## 1. Introduction

ASX ITCH is a direct data feed delivered via User Datagram Protocols (UDPs) and Transmission Control Protocols (TCPs). ASX ITCH enables the tracking of the status of an order from the time it is first entered until the time it is either executed or cancelled. Administrative messages can also be received. ASX ITCH is intended for information exchange only. Active trading is supported by the ASX OUCH protocol.

ASX ITCH is the premium ultra-low latency protocol for accessing ASX Market Information, delivered via a multicast connection directly from the ASX Trade platform. ASX ITCH has been developed to maximise performance and to meet the requirements of latency sensitive traders.

The ASX ITCH protocol provides:

- Ultra-low latency market information access (post full acceleration up to three times faster than current Liquidity Cross Connects (LCCs))
- Improved latency stability (reduced jitter)
- Control of the socket
- Multicast stream of order book changes
- Full order detail, meaning every quote and every order in every ASX lit order book (ASX TradeMatch and PureMatch)
- Trade data for all ASX order books (ASX TradeMatch, Centre Point and PureMatch)
- Access to all asset classes available in ASX Trade
- Access to security status messages
- Access to basic security data including ISIN code, financial product and tick size
- Internationally recognised and standardised protocol
- Time-stamping from ASX Trade to the nano-second.

### 1.1. ASX ITCH

ASX ITCH is a direct data feed product that features the following data elements:

- Order level data Market by Order (MBO) with broker ID - The system provides its full order depth using the standard ASX ITCH format. ASX ITCH uses a series of order messages to track the life of a customer order. The ASX ITCH message displays customer ID for non-anonymous instruments.
- Trade messages - ASX ITCH supports trade messages to reflect matches in lit and dark order books, with the exception of trade reports.
- Reference data:
  - Order Book Directory messages provide basic security data such as the ISIN code and financial product
  - Tick Size Table Entry messages to convey tick sizes for order books.
- Event controls, such as the states of the different order books
  - Order Book State message to inform receivers of state changes.

### 1.2. Glimpse

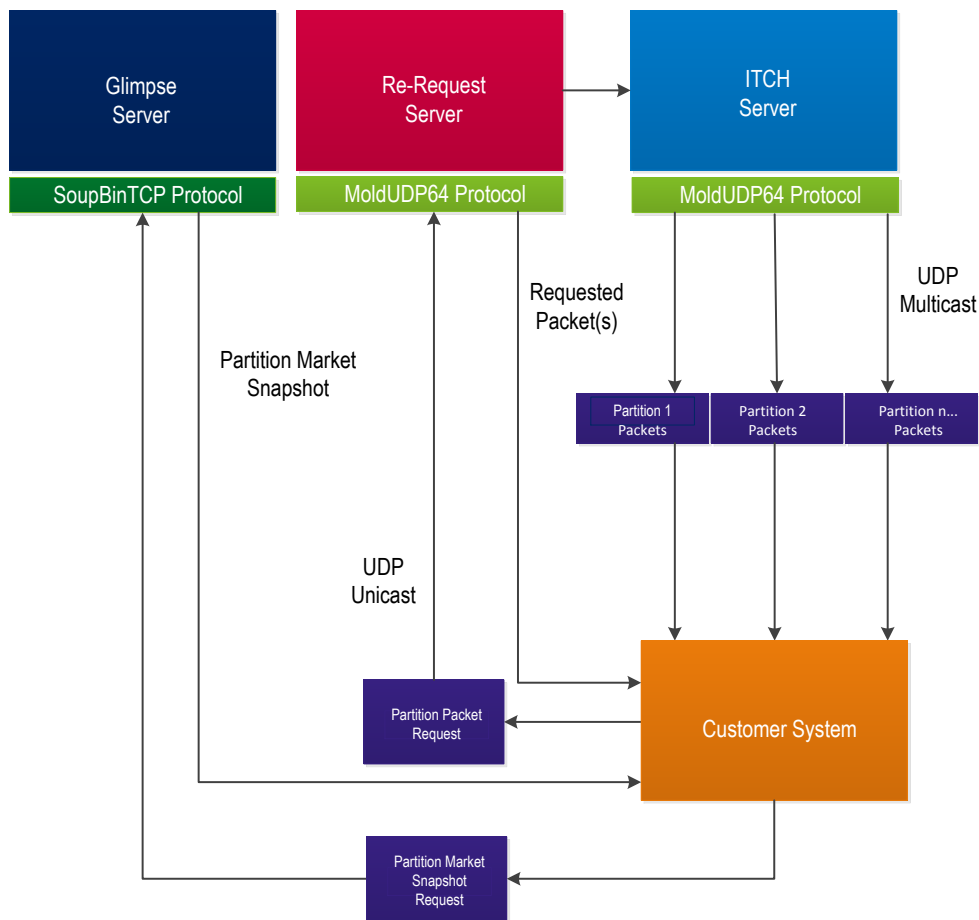
Complementing the ASX ITCH real-time data feed product, Glimpse is a point-to-point data feed connection that provides direct data feed customers with a snapshot of the current state of the order books traded in ASX Trade. Glimpse uses the same message formats as ASX ITCH.

Glimpse can be used to quickly sync up with the ASX ITCH feed. At the end of the Glimpse snapshot a sequence number is provided that can be used to connect and sync up with the real-time ASX ITCH feed.

Glimpse provides the following:

- Basic Reference Data for each order book including intra-day updates up until the time of login
- Current trading state of each order book
- All lit orders for each order book
- An End of Snapshot message providing the ASX ITCH sequence number to use when connecting to the real-time ASX ITCH feed.

The following diagram shows the relationship between the ASX ITCH, Glimpse and Re-Request Server. For more information on the MoldUDP64 protocol, see *Appendix 2 – ASX MoldUDP64 Protocol Specification*. For more information on the SoupBinTCP see *Appendix 3 - SoupBinTCP Protocol*.



For ASX ITCH/Glimpse questions contact ASX Customer Technical Support (CTS) team either via email on [cts@asx.com.au](mailto:cts@asx.com.au) or phone 1800 663 053 (or on +61 2 9227 0372 from outside Australia).

## 2. ASX ITCH

### 2.1. Architecture

The ASX ITCH feed is made up of a series of sequenced messages. Each message is variable in length, based on the message type. The messages will be binary encoded using MoldUDP64. The messages that make up the ASX ITCH protocol are delivered using a higher level protocol that takes care of sequencing and delivery guarantees.

#### 2.1.1. Protocol

The ASX ITCH data feed is offered in the following:

Protocol Option	Description
MoldUDP64	MoldUDP64 is a light-weight networking protocol built on top of UDP that provides a mechanism for listeners to detect and re-request missed packets. Each message is explicitly sequence numbered. If a packet loss is detected by the client, it can re-request that packet from the MoldUDP64 rewind server, and it will be resent as a UDP unicast to that client.

### 2.2. Data Types

All numeric fields are composed of binary encoded numbers.

All alpha fields are left justified and padded on the right with spaces. The alpha fields are composed of non-control ISO 8859-1 (Latin-1) encoded bytes.

Type	Size	Notes
Numeric	1, 2, 4, 8 or 16 bytes	Unsigned big-endian binary encoded numbers. <b>Note:</b> The transport layer, MoldUDP64, uses big-endian for its numeric values.
Alpha	Variable	Left justified and padded on the right with spaces.
Price	4 bytes	Prices are signed big-endian fields. Number of decimals is specified in the Order Book Directory message. <b>Note:</b> The number of decimal places for prices in ASX ITCH will universally be disseminated with 2 decimal places, regardless of the configured number of decimal places for the instrument in ASX Trade.

### 2.3. Message Formats

The ASX ITCH feed is composed of a series of messages that describe orders added to, removed from and executed on ASX Trade. It also contains messages for basic reference data of the order books as well as state changes and halts.

#### 2.3.1. Time Messages

For bandwidth efficiency, the ASX ITCH timestamp is separated into two parts:

Timestamp Portion	Message Type	Notes
Seconds	Standalone message	Unix time (number of seconds since 1970-01-01 00:00:00 UTC). <b>Note:</b> A Timestamp - Second message will be disseminated for every second for which there is at least one message.

Timestamp Portion	Message Type	Notes
Nanoseconds	Field within individual messages	Reflects the number of nanoseconds since the most recent Timestamp - Seconds message that the data message was generated.

### 2.3.1.1. Seconds Message

This message is sent every second for which at least one ASX ITCH message is being generated. The message contains the number of seconds since the start of 1970-01-01 00:00:00 UTC, also called Unix Time.

Name	Length	Value	Notes
Message Type	1	"T"	Seconds Message.
Second	4	Numeric	Unix time (number of seconds since 1970-01-01 00:00:00 UTC).

## 2.4. Reference Data Messages

### 2.4.1. Order Book Directory

At the start of each trading day, Order Book Directory messages are disseminated for all active instruments.



**Note:**

Intra-day transmissions of this message will occur when new order books (instruments) are added to the system. Updates to existing order books will also be represented by intra-day Order Book Directory messages.

Name	Length	Value	Notes
Message Type	1	"R"	Order Book Directory Message
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.
Order Book ID	4	Numeric	Denotes the primary identifier of an order book (instrument). <b>Note:</b> Expired Order Book IDs may be reused for new instruments.
Symbol	32	Alpha	The unique instrument series name (ins_id_s in ASX Trade).
Long Name	32	Alpha	Additional instrument series information. May be blank and may not necessarily be unique across all series (long_ins_id_s in ASX Trade).
ISIN	12	Alpha	ISIN code identifying the instrument.
Financial Product	1	Numeric	Values: 1 = Option 3 = Future 5 = Cash 11 = Standard combination. <b>Note:</b> Warrants have a value of 1 (Option).
Trading Currency	3	Alpha	Trading currency

Name	Length	Value	Notes
Number of decimals in Price	2	Numeric	This value defines the number of decimals used in price for this order book. <b>Note:</b> The number of decimal places for prices in ASX ITCH will universally be disseminated with 2 decimal places, regardless of the configured number of decimal places for the instrument in ASX Trade.
Number of decimals in Nominal Value	2	Numeric	This value defines the number of decimals used in the nominal value for this order book.
Odd Lot Size	4	Numeric	Indicates the number of securities that represent an odd lot for the order book. <b>Note:</b> A value of 0 indicates that this lot type is undefined for the order book.
Round Lot Size	4	Numeric	Indicates the number of shares that represent a round lot for the order book.
Block Lot Size	4	Numeric	Indicates the number of securities that represent a block lot for the order book. <b>Note:</b> A value of 0 indicates that this lot type is undefined for the order book.
Nominal Value	8	Numeric	Nominal value

#### Order Book Directory Example Message

Shows the Order Book ID (3104bc88) and other attributes for XJO11SEP4550EC.J88 for current day:

R,840622,3104bc88,XJO11SEP4550EC.J88,SEP-11 CALL OPT 4550  
[XJOJ88],AU000XJOJ888,1,AUD,2,0,0,1,0,0.

#### 2.4.2. Combination Order Book Directory

The Combination Order Book Directory is a specialised directory message used for combinations. It represents both standard combinations defined by ASX, and tailor-made combinations created by customers.



**Note:**

Intra-day transmissions of this message will occur when new combination order books are added in ASX Trade. This is typically the case for tailor-made combinations. Updates to existing combination order books may also be represented by intra-day Combination Order Book Directory messages.

Name	Length	Value	Notes
Message Type	1	"M"	Combination Order Book Directory Message.
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.



Name	Length	Value	Notes
Order Book ID	4	Numeric	Denotes the primary identifier of an order book (instrument). <b>Note:</b> Expired Order Book IDs may be reused for new instruments.
Symbol	32	Alpha	The unique instrument series name (ins_id_s in ASX Trade).
Long Name	32	Alpha	Additional instrument series information. May be blank and may not necessarily be unique across all series (long_ins_id_s in ASX Trade).
ISIN	12	Alpha	ISIN code identifying security.
Financial Product	1	Numeric	Values: 1 = Option 3 = Future 5 = Cash 11 = Standard combination. <b>Note:</b> Warrants have a value of 1 (Option)
Trading Currency	3	Alpha	Trading currency
Number of decimals in Price	2	Numeric	This value defines the number of decimals used in price for this order book. <b>Note:</b> The number of decimal places for prices in ASX ITCH will universally be disseminated with 2 decimal places, regardless of the configured number of decimal places for the instrument in ASX Trade.
Number of decimals in Nominal Value	2	Numeric	This value defines the number of decimals used in the nominal value for this order book.
Odd Lot Size	4	Numeric	Indicates the number of securities that represent an odd lot for the order book. <b>Note:</b> A value of 0 indicates that this lot type is undefined for the order book.
Round Lot Size	4	Numeric	Indicates the number of shares that represent a round lot for the order book.
Block Lot Size	4	Numeric	Indicates the number of securities that represent a block lot for the order book. <b>Note:</b> A value of 0 indicates that this lot type is undefined for the order book.
Nominal Value	8	Numeric	Nominal value
Leg 1, Symbol	32	Alpha	The unique series name of the leg.
Leg 1, Side	1	Alpha	Values: B = Buy leg C = Sell leg.

Name	Length	Value	Notes
Leg 1, Ratio	4	Numeric	Relative numbers of contracts in comparison to other legs.
Leg 2, Symbol	32	Alpha	The unique series name of the leg.
Leg 2, Side	1	Alpha	Values: B = Buy leg C = Sell leg.
Leg 2, Ratio	4	Numeric	Relative numbers of contracts in comparison to other legs.
Leg 3, Symbol	32	Alpha	The unique series name of the leg.
Leg 3, Side	1	Alpha	Values: B = Buy leg C = Sell leg ? = not defined (i.e. no leg 3 present).
Leg 3, Ratio	4	Numeric	Relative numbers of contracts in comparison to other legs.
Leg 4, Symbol	32	Alpha	The unique series name of the leg.
Leg 4, Side	1	Alpha	Values: B = Buy leg C = Sell leg ? = not defined (i.e. no leg 4 present).
Leg 4, Ratio	4	Numeric	Relative numbers of contracts in comparison to other legs.

#### Combination Order Book Directory Example Message

Shows the individual legs that comprise the **standard** combination WAW11SEP12JAN\_TF2:

M,604644,300058c3,WAW11SEP12JAN\_TF2,,11,AUD,2,0,0,1,0,0,WAW11SEPF.1U,C,1,WAW12JANF.2F,B,1,,?,0,,?,0

Shows the individual legs that comprise a **tailor-made** combination for BHP:

M,493439,14d6fff,TMC\_BHP\_D\_001,,,11,AUD,2,0,0,1,0,0,BHP,B,1,BHP12DEC3456C,C,1,,?,0,,?,0

#### 2.4.3. Tick Size Table Entry

This message contains information on a tick size for a price range. Together, all Tick Size messages with the same Order Book ID form a complete Tick Size Table. Each order book has a set of Tick Size Table Entries to define its tick size table.

Name	Length	Value	Notes
Message Type	1	"L"	Tick Size Message.
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.
Order Book ID	4	Numeric	The order book this entry belongs to (instrument).
Tick Size	8	Numeric	Tick Size for the given price range.

Name	Length	Value	Notes
Price From	4	Price	Start of price range for this entry. <b>Note:</b> The number of decimal places for prices in ITCH will universally be disseminated with 2 decimal places, regardless of the configured number of decimal places for the instrument in ASX Trade.
Price To	4	Price	End of price range for this entry. <b>Note:</b> The number of decimal places for prices in ITCH will universally be disseminated with 2 decimal places, regardless of the configured number of decimal places for the instrument in ASX Trade.

#### Tick Size Table Entry Example Message

Shows tick size table for BHP:

L, 698551,BHP,1,10,990  
 L, 698551,BHP,5,1000,19990  
 L, 698551,BHP,10,20000,2147483600

## 2.5. Event and State Change Messages

### 2.5.1. System Event Message

The system event message type is used to signal a market or data feed handler event. The format is specified below.

System Event Message			
Name	Length	Value	Notes
Message Types	1	"S"	System Event Message
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.
Event Code	1	Alpha	See System Event Codes below.

The system supports the following event codes on a daily basis.

System Event Codes – Daily	
Code	Explanation
"O"	<b>Start of Messages</b> Outside of time stamp messages, the start of day message is the first message sent every trading day.
"C"	<b>End of Messages</b> This is always the last message sent every trading day.

### 2.5.2. Order Book State Message

The Order Book State message relays information on session state changes.

Name	Length	Value	Notes
Message Types	1	"O"	Order Book State Message
Timestamp –	4	Numeric	Nanoseconds portion of the

Name	Length	Value	Notes
Nanoseconds			timestamp.
Order Book ID	4	Numeric	Order Book Identifier (instrument).
State Name	20	Alpha	Name of the session state.

#### Order Book State Change Example Message

Shows XJO11SEP4550EC.J88 moving to CLOSE:

O,040413, XJO11SEP4550EC.J88,CLOSE

## 2.6. Market by Order Messages



#### Note:

Order IDs are only unique per order book and side. When modifying or deleting orders, be careful to only update the order of the applicable side and order book, since the same Order ID may be present in multiple order books and/or sides.

### 2.6.1. Add Order Messages

An Add Order Message indicates that a new order has been accepted by ASX Trade and was added to the lit order book. The message includes an Order ID that is unique per order book and side.

ASX ITCH supports two variations of the Add Order message.

#### 2.6.1.1. Add Order – No Participant ID

This message will be generated for anonymous instruments in ASX Trade.

Name	Length	Value	Notes
Message Type	1	"A"	Add Order Message.
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.
Order ID	8	Numeric	The identifier assigned to the new order. Note that the number is <i>only</i> unique per order book and side.
Order Book ID	4	Numeric	Order Book Identifier (instrument).
Side	1	Alpha	The type of order being added. Values: "B" = buy order "S" = sell order.
Order Book Position	4	Numeric	Rank within the order book. See Building an Order Book View for details.
Quantity	8	Numeric	The visible quantity of the order. <b>Note:</b> Orders with an undisclosed quantity will have this field set to 0.
Price	4	Price	The price of the new order. Refer to Data Types for field processing notes.

Name	Length	Value	Notes
Exchange Order Type	2	Numeric	Additional order attributes. Values: 4 = Market Bid 8 = Price Stabilisation 32 = Undisclosed. <b>Note:</b> This field is a bit map. Multiple values may be set simultaneously.
Lot Type	1	Numeric	Lot Type. Values: 0 = Undefined 1 = Odd Lot 2 = Round Lot 3 = Block Lot 4 = All or None Lot.

#### Add Order without Customer ID Example Message

Shows a buy order entered in IRE for 876 @ 717 cents:

A, 845211,549f2944:005c8c1c,IRE,B,3,876,71700,0,2.

#### 2.6.1.2. Add Order With Participant ID

This message will be generated for non-anonymous instruments in ASX Trade.

Name	Length	Value	Notes
Message Type	1	"F"	Add Order Message.
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.
Order ID	8	Numeric	The identifier assigned to the new order. Note that the number is <i>only</i> unique per order book and side.
Order Book ID	4	Numeric	Order Book Identifier (instrument).
Side	1	Alpha	The type of order being added. Values: "B" = buy order "S" = sell order.
Order Book Position	4	Numeric	Rank within order book. See <i>Building an Order Book View</i> for details.
Quantity	8	Numeric	The visible quantity of the order. <b>Note:</b> Orders with an undisclosed quantity will have this field set to 0.
Price	4	Price	The price of the new order. Refer to Data Types for field processing notes.
Exchange Order Type	2	Numeric	Additional order attributes. Values: 4 = Market Bid 8 = Price Stabilisation 32 = Undisclosed. <b>Note:</b> This field is a bit map. Multiple values may be set simultaneously.

Name	Length	Value	Notes
Lot Type	1	Numeric	Lot Type. Values: 0 = Undefined 1 = Odd Lot 2 = Round Lot 3 = Block Lot 4 = All or None Lot.
Participant ID	7	Alpha	Participant identifier associated with the entered order.

#### Add Order with Participant ID Example Message

Shows a sell order entered by customer AU310 in XJO12MAR1500.EC for 35 @ 3 cents:

F, 313081,549f2943:005a8ae0,XJO12MAR1500EC.D27,S,1,35,300,0,2,AU310

## 2.6.2. Modify Order Messages

Modify Order messages always include the Order ID, Order Book ID and Side of the order to which the update applies. To determine the visible quantity of an order, ASX ITCH subscribers must deduct the executed quantity stated in the Modify Order message from the original quantity stated in the Add Order message with the same Order ID. ASX ITCH may send multiple Modify Order messages for the same order and the effects are cumulative. When the visible quantity for an order reaches zero, the order should be removed from the order book.

### 2.6.2.1. Order Executed Message

This message is sent whenever an order in the book is executed in whole or in part.

If the incoming order causing the match cannot be fully filled, the remainder will be placed in the order book after the match has occurred.

It is possible to receive several Order Executed Messages for the same order if that order is executed in several parts. Multiple Order Executed Messages on the same order are cumulative.

Name	Length	Value	Notes
Message Type	1	"E"	Order Executed message
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.
Order ID	8	Numeric	The order ID associated with the executed order.
Order Book ID	4	Numeric	Order Book Identifier (instrument).
Side	1	Alpha	The type of order being executed. Values: "B" = buy order "S" = sell order.
Executed Quantity	8	Numeric	The traded quantity.
Match ID	12	Numeric	Assigned by the system to each match executed.
Participant ID, owner	7	Alpha	Participant identifier of the owner of the order. Blank for anonymous instruments.
Participant ID, counterparty	7	Alpha	Participant identifier of the counterparty to the execution. Blank for anonymous instruments.

**Order Executed Example Message**

Shows order ID 5314aec1:00007957 trading 100 lots:

E,420177,5314aec1:00007957,  
XJO12MAR1500.EC,S,100,00000000:00d4afc1:00000009,AU550,AU551

**2.6.2.2. Order Executed with Price Message**

This message is sent when an order in the book is executed in whole or in part with a price different than the initial display price.

If the incoming order causing the match cannot be fully filled, the remainder will be placed in the order book after the match has occurred.

It is possible to receive several Order Executed messages for the same order if that order is executed in several parts. Multiple Order Executed messages on the same order are cumulative.

The executions may be marked as non-printable. If a customer is looking to use the ASX ITCH data in trade tickers or volume calculations, ASX recommends that customers ignore messages marked as non-printable to prevent double counting.

Name	Length	Value	Notes
Message Type	1	"C"	Order Executed message
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.
Order ID	8	Numeric	The order ID associated with the executed order.
Order Book ID	4	Numeric	Order Book Identifier (instrument).
Side	1	Alpha	The type of order being executed. Values: "B" = buy order "S" = sell order.
Executed Quantity	8	Numeric	The traded quantity.
Match ID	12	Numeric	Assigned by the system to each match executed.
Participant ID, owner	7	Alpha	Participant identifier of the owner of the order. Blank for anonymous instruments.
Participant ID, counterparty	7	Alpha	Participant identifier of the counterparty to the execution. Blank for anonymous instruments.
Trade Price	4	Price	The traded price.
Occurred at Cross	1	Alpha	Indicates if an order was executed in an auction. Values: "Y" = order executed in an auction "N" = order executed in continuous matching.

Name	Length	Value	Notes
Printable	1	Alpha	Indicates if the execution should be included in trade tickers and volume calculations. Values: "N" = do not include in trade tickers and volume calculations "Y" = include in trade tickers and volume calculations.

**Order Executed at Price Example Message**

Shows order 549e7402:000fe88c trading 100 lots at price \$31.25:

C,029192,549e7402:000fe88c,ASX,S,100,00d8c5c2:00000003:00000002,,,312500,Y,Y

**2.6.2.3. Order Replace Message**

This message is sent whenever an order in the book has been replaced. The remaining quantity from the original order is no longer accessible and must be removed.

The Side, Order Book ID, and Participant Identifier (if non-anonymous) remain the same as the original order. Customer identifiers are not part of the Order Replace message.

Name	Length	Value	Notes
Message Type	1	"U"	Order Replace message
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.
Order ID	8	Numeric	The original order identifier of the order being replaced. Note that the Order ID is only unique per order book and side. <b>Note:</b> The Order ID does not change when the order is replaced.
Order Book ID	4	Numeric	Order Book identifier (instrument).
Side	1	Alpha	The type of order being replaced. Values: "B" = buy order "S" = sell order.
New Order Book Position	4	Numeric	New rank within the order book. See Building an Order Book View for more information.
Quantity	8	Numeric	The new visible quantity of the order. <b>Note:</b> Orders with an undisclosed quantity will have this field set to 0.
Price	4	Price	The new price of the order.
Exchange Order Type	2	Numeric	Additional order attributes. Values: 4 = Market Bid 8 = Price Stabilisation 32 = Undisclosed. <b>Note:</b> This field is a bit map. Multiple values may be set simultaneously.

**Order Replace Example Message**

Shows sell order for BHP replaced for 1,000 @ 3570 cents at new position 7:



U,835289,54ad5081:00011f68,BHP,S,7,1000,357000,0

#### 2.6.2.4. Order Delete Message

This message is sent whenever an order in the book is deleted. There will be no remaining quantity, so the order should be removed from the book.

Please note that normally no Order Delete message is sent when an order is completely filled. The receiver needs to keep track of the remaining quantity on all orders by recalculating the remaining quantity on each Order Executed message received. Orders must be removed from the book when remaining quantity reaches zero.



**Note:**

Order Delete messages are sent when orders with undisclosed quantity are fully filled.



**Note:**

Order Delete messages are sent when orders are inactivated. When central inactive orders are reactivated by a customer, this order will be added as a new order (Add Order message) with the same Order ID.

Name	Length	Value	Notes
Message Type	1	"D"	Order Delete message.
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.
Order ID	8	Numeric	The ID of the order being deleted. <b>Note:</b> The Order ID is only unique per order book and side.
Order Book ID	4	Numeric	Order Book identifier (instrument).
Side	1	Alpha	The type of order being deleted. Values: "B" = buy order "S" = sell order.

#### Order Delete Example Message

Shows buy order in RIO being deleted:

D,029597,54ad5081:00011f6b,RIO,B

## 2.7. Trade Messages

The Trade Message is designed to provide details for executions in dark order books and reporting the individual legs of traded combinations.

Since no Add Order Message is generated when a dark order is initially entered, the Order Executed message cannot be used for those matches. The Trade Message is used to report a match for a non-displayable order in the book.

It is possible to receive multiple Trade Messages for the same order if that order is executed in several parts. Trade Messages for the same order are cumulative.

Trade Messages should be included in trade tickers as well as volume and other market statistics when they have the Printable flag set to Y. Since Trade Messages do not affect the displayed book, they may be ignored by customers just looking to build and track the order book view.

Name	Length	Value	Notes
Message Type	1	"P"	Trade message identifier
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.
Match ID	12	Numeric	The unique reference number assigned to the order on the book being executed.
Side	1	Alpha	The type of non-display order being matched. Values: "B" =buy order "S" =sell order.
Quantity	8	Numeric	The quantity being matched in this execution.
Order Book ID	4	Numeric	Order Book identifier (instrument).
Trade Price	4	Price	The price of the trade.
Participant ID, owner	7	Alpha	Participant identifier of the owner of the order. Blank for anonymous instruments.
Participant ID, counterparty	7	Alpha	Participant identifier of the counterparty to the execution. Blank for anonymous instruments.
Printable	1	Alpha	Indicates if the execution should be included in trade tickers and volume calculations. Values: "N" = Do not include in trade tickers and volume calculations "Y" = Include in trade tickers and volume calculations.
Occurred at Cross	1	Alpha	Values: "Y" = Yes, trade occurred in an auction. "N" = No, trade occurred in continuous matching.

#### Trade Example Message

Shows execution for QAN in Centre Point for 2 @ 173.75 cents

P,381984,00d8c5c3:00000001:00000001,,2,QAN,17375,,,Y,N

## 2.8. Auction Messages – Equilibrium Price Update

This message is used when auctions occur. The message provides the changes in equilibrium (auction) price. Note that subtracting the Ask Quantity from the Bid Quantity will yield the Surplus Volume.

Name	Length	Value	Notes
Message Type	1	"Z"	Equilibrium Price Update message.
Timestamp – Nanoseconds	4	Numeric	Nanoseconds portion of the timestamp.
Order Book ID	4	Numeric	Order Book identifier (instrument).

Name	Length	Value	Notes
Bid Quantity	8	Numeric	Total Bid Quantity available for execution.
Ask Quantity	8	Numeric	Total Ask Quantity available for execution.
Equilibrium Price	4	Price	The price at which matching will occur.
Best Bid Price	4	Price	Best Bid Price.
Best Ask Price	4	Price	Best Ask Price.
Best Bid Quantity	8	Numeric	Quantity at Best Bid Price.
Best Ask Quantity	8	Numeric	Quantity at Best Ask Price.

#### Equilibrium Price Update Example Messages

Shows BHP in PRE\_OPEN state, 30 executable bids and 12 executable asks, (surplus volume is 18), equilibrium price is \$44.00, best bid quantity is 30 @ \$44.00, best ask quantity is 6 @ \$43.90:

Z,909219,BHP,30,12,440000,440000,439000,30,6

Shows BHP moving to OPEN state:

O,800972,BHP,OPEN

Shows that the auction has occurred and an equilibrium price no longer exists:

Z,800972,BHP,0,0,-2147483648,-2147483648,-2147483648,0,0

## 2.9. Building an Order Book View

The information needed to build an order book view from the ASX ITCH message flow is contained in Add Order Messages and Modify Order Messages. The messages are:

- Add Order – No Participant ID
- Add Order – With Participant ID
- Order Executed
- Order Executed with Price
- Order Replace
- Order Delete.

The two variants of the Add Order messages have the same meaning; an order is added to the book. Orders are ranked by:

- Price
- Order Book Position - "1" denotes the highest ranked order. For an Order Replace, the order must be removed from its previous position and inserted at the new Order Book Position. An order inserted at an existing position shifts the order in that position down (and all orders below as well). A deleted or fully filled order causes existing orders below it to shift their position up one step to fill the "void".

The Order Executed and Order Executed with Price messages signal a partial or full fill. The order quantity must be reduced by the quantity of the message.

The Order Replace message signals that the order has been modified. The current rank may or may not be lost in the process. The Order Book Position will show the new rank within the book.

The Order Delete message tells the recipient to remove the referenced order.

## 2.10. Building a Trade Ticker

The Trade Ticker is based on the following messages:

- Order Executed
- Order Executed with Price
- Trade.



**Note:**

Trades and Order Executed with Price messages marked as non-printable should be excluded to avoid double booking of trades.



**Note:**

Trade Reports are not included in ASX ITCH.

### 3. Glimpse

#### 3.1. Architecture

The Glimpse data feed is made up of a series of sequenced messages. Each message is variable in length based on the message type. The messages will be binary encoded using SoupBinTCP. SoupBinTCP takes care of sequencing and delivery guarantees.



**Note:**

Customers must log in with SoupBinTCP sequence number 1 to correctly receive data.

##### 3.1.1. Protocol

Name	Length
SoupBinTCP	<p>SoupBinTCP is a lightweight point-to-point protocol, built on top of TCP/IP sockets that allow delivery of a set of sequenced messages from a server to a customer in real-time. SoupBinTCP guarantees that the customer receives each message generated by the server in sequence, even across underlying TCP/IP socket connection failures.</p> <p>The sequence numbers are implicit, meaning that the customer maintains a counter that is increased every time a message is received. To reconnect after a connection loss, the customer submits the last seen sequence number in its Logon message, and the server resends every message starting from that sequence number.</p> <p><b>Note:</b> Refer to the Nasdaq Trader website (<a href="https://www.nasdaqtrader.com">https://www.nasdaqtrader.com</a>) for the latest SoupBinTCP manual.</p>

#### 3.2. Data Types

Glimpse messages have the same data types as ASX ITCH messages.

#### 3.3. Glimpse Messages

Glimpse uses a subset of ASX ITCH messages.

Glimpse utilises all messages used by ASX ITCH to describe the current picture of the book, with the exception of those messages that change the picture of the order book (e.g. Delete, Execute and Trade).

##### 3.3.1. End of Snapshot Message

The end of snapshot message is sent at the end of a Glimpse snapshot and returns the current ASX ITCH sequence number to be used when connecting to the ASX ITCH feed.

To maintain a real-time order display, real-time processing of ASX ITCH messages should begin with the sequence number stated in this snapshot message.

Name	Length	Value	Notes
Message Type	1	"G"	End of Snapshot Message.
Sequence Number	20	Alpha	<p>ASX ITCH sequence number when the snapshot was taken. To be used when applying messages received via the ASX ITCH MoldUDP feed.</p> <p><b>Note:</b> While Glimpse is a binary feed, the SoupBinTCP uses ASCII characters to represent the sequence number.</p>

## 4. Configuration and Infrastructure

### 4.1. Connectivity Requirements

#### 4.1.1. Overview of Feeds

The following provides an overview of the ASX ITCH multicast feed components:

- Each ASX switched environment provides dual redundant ASX ITCH data services
- Each data service consists of:
  - A multicast base address and base port for real time data dissemination using MoldUDP64 protocol
  - An IP address and base port for snapshot retrieval using SoupTCP protocol
  - An IP address and base port for MoldUDP64 retransmit requests.
- Each base address or port requires the addition of the matching engine partition number to determine the actual address or port on which to receive the ASX ITCH data. ASX Trade currently has five matching engine partitions. For example, multicast base address 233.54.12.231 port 21000 yields:
  - Address 233.54.12.232:21001 for partition 1 data
  - Address 233.54.12.233:21002 for partition 2 data
  - Address 233.54.12.232:21003 for partition 3 data
  - Address 233.54.12.233:21004 for partition 4 data
  - Address 233.54.12.233:21005 for partition 5 data.
- Upon completion of the snapshot retrieval (Glimpse) an “end of snapshot” message will be received containing the ASX ITCH sequence number with which to begin processing real time data.
- It is mandatory to begin each day's session with a Glimpse snapshot retrieval as these messages contain directory information in addition to the current order book snapshot.
- The MoldUDP64 messages sent by the retransmit server as the result of a retransmit request will be received by the sending application on the port from which the request was sent.

#### 4.1.2. Feed Address Examples

The following table provides examples of the various base addresses and the actual addresses which would be used for the matching engine partitions. Note these are from the Functional Test Environment and are different from the actual Production addresses that will be provided by CTS. The table below shows the first five partitions.

ITCH Feed A		ITCH Feed B	
MoldUDP64 Base Address and Port			
233.54.12.223 21000 (Multicast UDP)		233.54.12.239 21100 (Multicast UDP)	
Partition 1	233.54.12.224:21001	Partition 1	233.54.12.240:21101
Partition 2	233.54.12.225:21002	Partition 2	233.54.12.241:21102
Partition 3	233.54.12.226:21003	Partition 3	233.54.12.242:21103
Partition 4	233.54.12.226:21004	Partition 4	233.54.12.242:21104
Partition 5	233.54.12.226:21005	Partition 5	233.54.12.242:21105
Glimpse IP Address and Port (TCP)			
203.0.119.228 21800		203.0.119.229 21800	

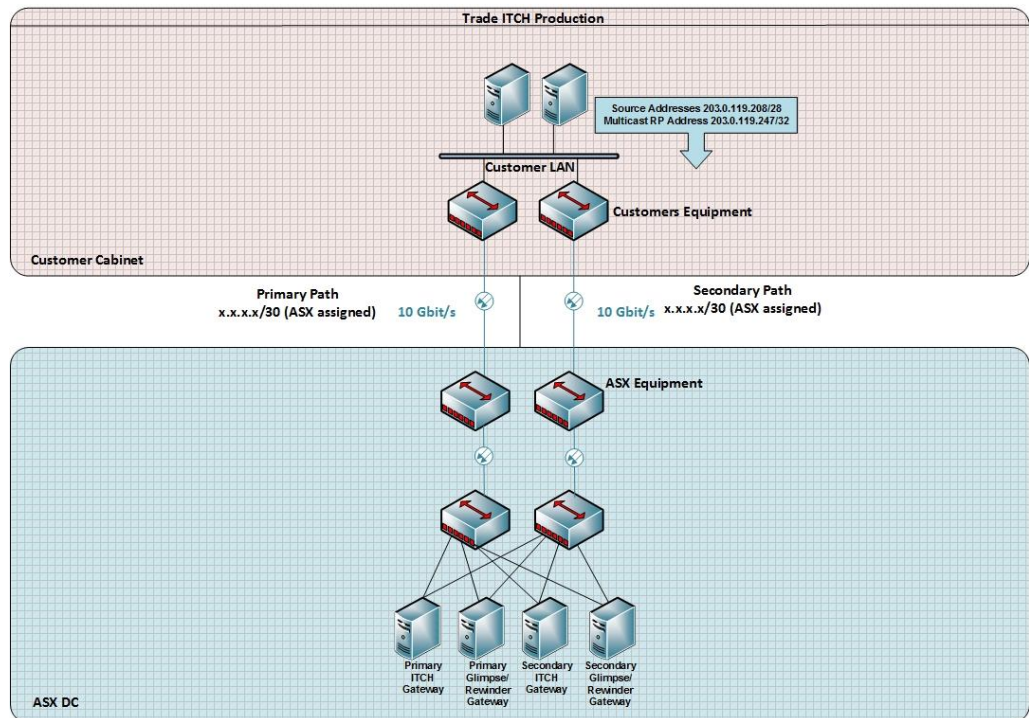
ITCH Feed A		ITCH Feed B	
Partition 1	203.0.119.228:21801	Partition 1	203.0.119.229:21801
Partition 2	203.0.119.228:21802	Partition 2	203.0.119.229:21802
Partition 3	203.0.119.228:21803	Partition 3	203.0.119.229:21803
Partition 4	203.0.119.228:21804	Partition 4	203.0.119.229:21804
Partition 5	203.0.119.228:21805	Partition 5	203.0.119.229:21805
Retransmit IP Address and Port (UDP) – Response Received on Sending Port			
203.0.119.228 24000		203.0.119.229 24000	
Partition 1	203.0.119.228:24001	Partition 1	203.0.119.229:24001
Partition 2	203.0.119.228:24002	Partition 2	203.0.119.229:24002
Partition 3	203.0.119.228:24003	Partition 3	203.0.119.229:24003
Partition 4	203.0.119.228:24004	Partition 4	203.0.119.229:24004
Partition 5	203.0.119.228:24005	Partition 5	203.0.119.229:24005

#### 4.1.3. Configuration Details

ASX ITCH utilises multicast through switched infrastructure. Customers of ASX ITCH will be required to allow for the following infrastructure requirements:

- Physical Layer:
  - The service will be delivered on OM3 multimode duplex LC connector.
  - The Media standard is 10GBASE-SR 10 Gigabit Ethernet connection.
  - It is the customer's responsibility to provide SFP's/XFP's to terminate the fibres.
- Customer Addressing:
  - All addressing is IP Version 4
  - The customer can use their own globally unique addressing, or
  - The customer can be allocated a private /24 address range and two /30 address ranges by ASX to be used on the customer's infrastructure from the following ranges:
    - 172.30.0.0
    - 10.30.0.0
    - 192.168.0.0
- ASX Addressing:
  - Multicast configuration (addressing, static RP, multicast groups) to be advised by ASX
  - Unicast services addressing to be advised by ASX
- Routing:
  - ASX standard routing is static
  - ASX offers BGP for customers upon request
  - ASX numbers will vary for each service and connection
  - MD5 BGP key will be required and can be set by either ASX or the customer
  - Sending and receiving prefixes will also need to be agreed upon
- Customers infrastructure:
  - Customer's network infrastructure will require the multicast feature set.

- Customer's infrastructure should be sized so it can buffer 10Gbit/sec bursts.
- Customers can elect to have separate connections for ASX ITCH and T24ITCH but must terminate it on separate infrastructure.
- Customer test environment:
  - The customer test environment is provided on separate infrastructure thus a separate cross connect is required.
  - The physical layer specifications are the same a production.
  - The customer test environment only has one channel.



PIM timers, PIM DR and IGMP timers have been tuned to obtain the fastest failover times in the event of a network failure within an ASX environment.

#### Production Multicast and Unicast Addressing

Aggregate Address	203.0.119.208/28	233.71.185.8/29					
Production Channel A	Multicast Source	Multicast Group	Multicast Port	Glimpse Address	Glimpse Port TCP	Rewinder IP Address	Rewinder UDP Port
Partition 1	203.0.119.212	233.71.185.8	21001	203.0.119.213	21801	203.0.119.213	24001
Partition 2	203.0.119.212	233.71.185.9	21002	203.0.119.213	21802	203.0.119.213	24002
Partition 3	203.0.119.212	233.71.185.10	21003	203.0.119.213	21803	203.0.119.213	24003
Partition 4	203.0.119.212	233.71.185.11	21004	203.0.119.213	21804	203.0.119.213	24004
Partition 5	203.0.119.212	233.71.185.12	21005	203.0.119.213	21805	203.0.119.213	24005

Aggregate Address	203.0.119.208/28	233.71.185.16/29					
Production Channel A	Multicast Source	Multicast Group	Multicast Port	Glimpse Address	Glimpse Port TCP	Rewinder IP Address	Rewinder UDP Port
Partition 1	203.0.119.214	233.71.185.16	21001	203.0.119.215	21801	203.0.119.215	24001
Partition 2	203.0.119.214	233.71.185.17	21002	203.0.119.215	21802	203.0.119.215	24002



Aggregate Address	203.0.119.208 /28	233.71.185.16/29					
Production Channel A	Multicast Source	Multicast Group	Multicast Port	Glimpse Address	Glimpse Port TCP	Rewinder IP Address	Rewinder UDP Port
Partition 3	203.0.119.214	233.71.185.18	21003	203.0.119.215	21803	203.0.119.215	24003
Partition 4	203.0.119.214	233.71.185.19	21004	203.0.119.215	21804	203.0.119.215	24004
Partition 5	203.0.119.214	233.71.185.20	21005	203.0.119.215	21805	203.0.119.215	24005

Multicast PIM Rendezvous Point production.

Multicast RP Address	203.0.119.247/32
----------------------	------------------

## 5. Appendix 1 – ASX ITCH Scenarios

To illustrate ASX ITCH functionality for both common and ad-hoc scenarios, the following examples are provided.

### Disclosed Orders, Continuous Trading

- New order entry, amend and delete
- New order fully trades with existing order(s)
- New order partially trades with existing order(s)
- Existing order is amended and immediately fully trades with opposing order
- Existing order is amended and immediately partially trades with opposing order.

### Undisclosed Orders, Continuous Trading

- Undisclosed order entered into book
- New order fully trades with existing undisclosed order. Undisclosed order remains above the undisclosed threshold
- New order fully trades with existing undisclosed order. Undisclosed order falls below the undisclosed threshold
- New order fully trades with existing undisclosed order. Undisclosed order has fully traded.

### Iceberg Order Execution, Continuous Trading

- New order fully trades with existing iceberg order. New order quantity less than iceberg order's shown quantity.
- New order fully trades with existing iceberg order. New order quantity equal to iceberg order's shown quantity.
- New order fully trades with existing iceberg order. New order quantity greater than iceberg order's shown quantity but less than total quantity.
- New order partially trades with existing iceberg order, fully trading out the iceberg order.

### Disclosed Order Execution, Auctions

- Orders trade in auction.

### Tailor Made Combinations (TMCs)

- New TMC order trades with existing TMC order
- New TMC order trades with existing orders in the book
- TMC order trade with opposing TMC order during an auction.

### Instrument Creation

- TMC creation
- Standard instrument creation.

### Miscellaneous Tasks

- Customer to customer order transfer.



#### Note:

**For Repeating Steps** - A single asterisk (\*) against a step indicates that the step may be repeated. Double asterisks (\*\*) groups the blocks of steps that may repeat together.

## 5.1. Disclosed Orders, Continuous Trading

### 5.1.1. New Order Entry, Amend and Delete

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Order A entered into book	A	Order ID, quantity and price	Store Order ID, quantity and price for Order A.
2	Order A amended	U	Order ID, quantity and price	Modify quantity and price for Order A.
3	Order A deleted	D	Order ID	Remove Order A.

Step	Scenario – With Participant ID	ITCH Message	Attributes	Activity
1	Order A entered into book	F	Order ID, quantity and price	Store Order ID, quantity and price for Order A.
2	Order A amended	U	Order ID, quantity and price	Modify quantity and price for Order A.
3	Order A deleted	D	Order ID	Remove Order A.

### 5.1.2. New Order Fully Trades with Existing Order(s)

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Order A entered into book	A	Order ID, quantity and price of Order A	Store Order ID, quantity and price for Order A.
2*	New Order B is entered and fully trades with Order A (and possibly other existing orders)	E	Order ID and executed quantity for Order A	Subtract executed quantity from stored Order A. If stored Order A quantity is zero, delete the stored order.

Step	Scenario – With Participant ID	ITCH Message	Attributes	Activity
1	Order A entered Into book	F	Order ID, quantity and price of Order A	Store Order ID, quantity and price for Order A.
2*	New Order B is entered and fully trades with Order A (and possibly other existing orders)	E	Order ID and executed quantity for Order A	Subtract executed quantity from stored Order A. If stored Order A quantity is zero, delete the stored order.

### 5.1.3. New Order Partially Trades with Existing Order

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Order A entered into	A	Order ID, quantity	Store Order ID, quantity

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
	book		and price of Order A	and price of Order A.
2	New Order B is entered and partially trades with Order. Order A is fully executed.	E	Order ID and executed quantity for Order A	Subtract executed quantity from stored Order A. Order A has fully traded, therefore the quantity results in zero. Delete stored Order A.
3		A	Residual Order ID, quantity and price of Order B	Store Order ID, quantity and price for Order B.

Step	Scenario – With Participant ID	ITCH Message	Attributes	Activity
1	Order A entered into book	F	Order ID, quantity and price of Order A	Store Order ID, quantity and price of Order A.
2	New Order B is entered and partially trades with Order. Order A is fully executed.	E	Order ID and executed quantity for Order A	Subtract executed quantity from stored Order A. Order A has fully traded, therefore the quantity results in zero. Delete stored Order A
3		F	Residual Order ID, quantity and price of Order B	Store Order ID, quantity and price for Order B.

#### 5.1.4. Existing Order Amended and Immediately Fully Trades with Opposing Order

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Order A entered into book	A	Order ID, quantity and price of Order A	Store Order ID, quantity and price of Order A.
2	Order B entered into book	A	Order ID, quantity and price of Order B	Store Order ID, quantity and price of Order B.
3	Order B amended to trade with Order A (and possibly other existing orders)	D	Order ID of Order B	Remove Order B.
4*		E	Order ID and executed quantity for Order A	Subtract executed quantity from stored Order A. Order A has fully traded, therefore the quantity results in zero. Delete stored Order A.

Step	Scenario – With Participant ID	ITCH Message	Attributes	Activity
1	Order A entered into book	F	Order ID, quantity and price of Order A	Store Order ID, quantity and price of Order A.
2	Order B entered into book	F	Order ID, quantity and price of Order B	Store Order ID, quantity and price of Order B.
3	Order B amended to trade with Order A (and possibly other existing orders)	D	Order ID of Order B	Remove Order B.
4*		E	Order ID and executed quantity for Order A	Subtract executed quantity from stored Order A. Order A has fully traded, therefore the quantity results in zero. Delete stored Order A.

#### 5.1.5. Existing Order Amended and Immediately Partially Trades with Existing Order

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Order A entered into book	A	Order ID, quantity and price of Order A	Store Order ID, quantity and price of Order A.
2	Order B entered into book	A	Order ID, quantity and price of Order B	Store Order ID, quantity and price of Order B.
3	Order B amended to trade with Order A. Order A fully trades.	D	Order ID of Order B	Remove Order ID B.
4		E	Order ID and executed quantity for Order A	Subtract executed quantity from stored Order A. Order A has fully traded, therefore the quantity results in zero. Delete stored Order A
5		A	Residual quantity and price of Order B.	Store Order ID, quantity and price of Order B. Order ID will be the same as original the original Order B.

Step	Scenario – With Participant ID	ITCH Message	Attributes	Activity
1	Order A entered into book	F	Order ID, quantity and price of Order A	Store Order ID, quantity and price of Order A.
2	Order B entered into	F	Order ID, quantity	Store Order ID, quantity

Step	Scenario – With Participant ID	ITCH Message	Attributes	Activity
	book		and price of Order B	and price of Order B.
3	Order B amended to trade with Order A. Order A fully trades.	D	Order ID of Order B	Remove Order B.
4		E	Order ID and executed quantity for Order A	Subtract executed quantity from stored Order A. Order A has fully traded, therefore the quantity results in zero. Delete stored Order A.
5		F	Residual quantity and price of Order B.	Store Order ID, quantity and price of Order B. Order ID will be the same as original the original Order B.

## 5.2. Undisclosed Order, Continuous Trading

### 5.2.1. Undisclosed Order Entered Into Book

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Undisclosed Order A entered into book	A	Order ID and price of Order A. Quantity is 0 and Exchange Order Type is 32.	Store Order ID, unknown quantity and price of Order A.

### 5.2.2. New Order Fully Trades with Existing Undisclosed Order. Undisclosed Order Remains above the Undisclosed Threshold

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Undisclosed Order A entered into book	A	Order ID and price of Order A. Quantity is 0 and Exchange Order Type is 32.	Store Order ID, unknown quantity and price of Order A.
2	New Order B fully trades with Order A, leaving Order A above the undisclosed threshold.	P	Executed quantity and price. Printable flag set to Y.	Update trade volume only.

### 5.2.3. New Order Fully Trades with Existing Undisclosed Order. Undisclosed Order Falls Below the Undisclosed Threshold

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Undisclosed Order A entered into book	A	Order ID and price of Order A.	Store Order ID, unknown quantity and

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
			Quantity is 0 and Exchange Order Type is 32.	price of Order A.
2	New Order B fully trades with Order A. Order A falls below the undisclosed threshold.	P	Executed quantity and price. Printable flag set to Y.	Update trade volume only.
3		D	Order ID of Order A	Remove Order A.
4		A	Order ID, residual disclosed quantity and price of Order A. Exchange Order Type is 0.	Store Order ID, quantity and price of Order A. Order ID will be the same as original the original Order A.

#### 5.2.4. New Order Fully Trades with Existing Undisclosed Order. Undisclosed Order Has Fully Traded

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Undisclosed Order A entered into book	A	Order ID and price of Order A. Quantity is 0 and Exchange Order Type is 32.	Store Order ID, unknown quantity and price of Order A.
2	New Order B fully trades with Order A. Order A fully trades.	P	Executed quantity and price. Printable flag set to Y.	Update trade volume only.
3		D	Order ID of Order A.	Remove Order A.

### 5.3. Iceberg Order Execution, Continuous Trading

#### 5.3.1. New Order Fully Trades with Existing Iceberg Order. New Order Quantity Less than Iceberg Order's Shown Quantity

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Iceberg Order A entered into book.	A	Order ID, shown quantity and price of Order A	Store Order ID, shown quantity and price of Order A.
2	New Order B fully trades with Order A. The executed quantity is less than Order A's shown quantity.	E	Executed quantity and price of Order A.	Subtract executed quantity from stored quantity for Order A.

### 5.3.2. New Order Fully Trades with Existing Iceberg Order. New Order Quantity Equal to Iceberg Order's Shown Quantity

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Iceberg Order A entered into book.	A	Order ID, shown quantity and price of Order A	Store Order ID, shown quantity and price of Order A.
2	New Order B fully trades out Order A's shown quantity.	D	Order ID of Order A	Remove Order A.
3		P	Executed quantity and price. Printable flag set to Y.	Update trade volume only.
4	<i>Refreshed iceberg has changed position and/or quantity.</i>	A	Order ID, shown quantity and price of Order A.	Order ID, refreshed shown quantity, price and order book position of Order A. Order ID will be the same as original the original Order A.

### 5.3.3. New Order Fully Trades with Existing Iceberg Order. New Order Quantity Greater than the Iceberg Order's Shown Quantity but Less than Total Quantity

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Iceberg Order A entered into book.	A	Order ID, shown quantity and price of Order A	Store Order ID, shown quantity and price of Order A.
2	New Order B fully trades with Order A. The executed quantity is greater than Order A's shown quantity.	D	Order ID of Order A	Remove Order A.
3		P	Executed quantity and price. Printable flag set to Y.	Update trade volume only.
4*	Order B trades with other normal orders	E	Executed quantity of other orders.	Subtract executed quantity from stored other orders.
5	<i>Refreshed iceberg has changed position and/or quantity</i>	A	Order ID, shown quantity and price of Order A.	Order ID, refreshed shown quantity, price and order book position of Order A. Order ID will be the same as original the original Order A.



#### 5.3.4. New Order Fully Trades with Existing Iceberg Order, Fully Trading out the Iceberg Order

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Iceberg Order A entered into book.	A	Order ID, shown quantity and price of Order A	Store Order ID, shown quantity and price of Order A.
2	New Order B fully trades with Order A. Order A is fully filled.	E	Executed shown quantity of Order A	Subtract executed quantity from stored Order A. Order A has fully traded, therefore the quantity results in zero. Delete stored Order A.
3		P	Executed quantity of remainder of Order B. Printable flag set to Y.	No change to order book. Volume is total traded volume less volume executed in Step 2.

#### 5.4. Disclosed Order Execution, Auctions

##### 5.4.1. Orders Trade in Auction

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1*	Instrument series in PRE_OPEN.	O	Instrument series PRE_OPEN	Market is in PRE_OPEN. Session state change for each applicable instrument series.
2	New Sell Order A entered into book.	A	Order ID, quantity and price of Order A	Store Order ID, quantity and price of Order A.
3		Z	Equilibrium price message for instrument series if applicable.	Equilibrium message indicates auction price and surplus volume if applicable.
4	New Buy Order B entered into book.	A	Order ID, quantity and price of Order B	Store Order ID, quantity and price of Order B.
5		Z	Equilibrium price message for instrument series if applicable.	Equilibrium message indicates auction price and surplus volume if applicable.
6*	Instrument series in OPEN.	O	Instrument series OPEN	Market opens. Session state change for each applicable instrument series.
7**		C	Executed quantity and price of Order A. Printable is set to Y (on Sell side).	Subtract executed quantity from stored Order A. If stored Order A quantity is zero, delete the stored order.

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
8**		C	Executed quantity and price of Order B. Printable is set to N (on Buy side).	Subtract executed quantity from stored Order B. If stored Order B quantity is zero, delete the stored order.

## 5.5. Tailor Made Combinations

### 5.5.1. New TMC Order Trades with Existing TMC Order

Step	Scenario – With Participant ID	ITCH Message	Attributes	Activity
1	Order A for TMC entered into book	F	Order ID, quantity and price of Order A.	Store Order ID, quantity and price of Order A.
2	New Order B fully trades with Order A.	C	Executed quantity and price of Order A. Printable flag set to N.	Subtract executed quantity from stored Order A. If stored Order A quantity is zero, delete the stored order.
3*	<i>Execution quantities for TMC legs are reported</i>	P	Executed quantity and price of each TMC leg. Printable flag set to Y.	No change to order book. Update trade volume statistics.

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Order A for TMC entered into book.	A	Order ID, quantity and price of Order A.	Store Order ID, quantity and price of Order A.
2	New Order B fully trades with Order A.	C	Executed quantity and price of Order A. Printable flag set to N.	Subtract executed quantity from stored Order A. If stored Order A quantity is zero, delete the stored order.
3*	<i>Execution quantities for TMC legs are reported</i>	P	Executed quantity and price of each TMC leg. Printable flag set to Y.	No change to order book. Update trade volume statistics.

### 5.5.2. New TMC Order Trades with Existing Orders in the Book

Step	Scenario – With Participant ID	ITCH Message	Attributes	Activity
1	Order A for TMC entered into book.	F	Order ID, quantity and price of Order A.	Store Order ID, quantity and price of Order A.

Step	Scenario – With Participant ID	ITCH Message	Attributes	Activity
2	New Order B entered against one TMC leg.	F	Order ID, quantity and price of Order B.	Store Order ID, quantity and price of Order B.
3	<i>Bait generated by system for remaining leg</i>	F	Order ID, quantity and price of bait.	Store Order ID, quantity and price of bait.
4	New Order C entered to trade with bait.	E	Executed quantity of bait.	Subtract executed quantity from stored quantity for the bait. If stored quantity is zero, delete the bait.
5	<i>Order B trades against TMC leg</i>	E	Executed quantity of Order B.	Subtract executed quantity from stored quantity for Order B. If stored Order B quantity is zero, delete the stored order.
6	<i>Quantity of Order A is reduced</i>	C	Executed quantity and price of Order A. Printable flag is set to N.	Subtract executed quantity from stored quantity of Order A. If stored Order A quantity is zero, delete the stored order.

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
1	Order A for TMC entered into book.	A	Order ID, quantity and price of Order A.	Store Order ID, quantity and price of Order A.
2	New Order B entered against one TMC leg.	A	Order ID, quantity and price of Order B.	Store Order ID, quantity and price of Order B.
3	<i>Bait generated by system for remaining leg</i>	A	Order ID, quantity and price of bait.	Store Order ID, quantity and price of bait.
4	New Order C entered to trade with bait.	E	Executed quantity of bait.	Subtract executed quantity from stored quantity for the bait. If stored quantity is zero, delete the bait.
5	<i>Order B trades against TMC leg</i>	E	Executed quantity of Order B.	Subtract executed quantity from stored quantity for Order B. If stored Order B quantity is zero, delete the stored order.
6	<i>Quantity of Order A is reduced</i>	C	Executed quantity and price of Order A. Printable flag is set to N.	Subtract executed quantity from stored quantity of Order A. If stored Order A quantity is zero, delete

Step	Scenario – No Participant ID	ITCH Message	Attributes	Activity
				the stored order.

### 5.5.3. TMC Order Trades with an Opposing TMC Order during an Auction

Step	Scenario – With Participant ID	ITCH Message	Attributes	Activity
1	Instrument series in PRE_OPEN	O	Instrument series PRE_OPEN	Market is in PRE_OPEN. Session state change for each applicable instrument series.
2	Sell Order A for TMC entered into book.	F	Order ID, quantity and price of Order A.	Store Order ID, quantity and price of Order A.
3		Z	Equilibrium price message for instrument series if applicable.	Equilibrium message indicates auction price and surplus volume if applicable.
4	Buy Order B for TMC entered into book.	F	Order ID, quantity and price of Order B.	Store Order ID, quantity and price of Order B.
5		Z	Equilibrium price message for instrument series if applicable.	Equilibrium message indicates auction price and surplus volume if applicable.
6	Instrument series in OPEN	O	Instrument series OPEN	Market opens. Session state change for each applicable Instrument Series.
7		C	Executed quantity and price for Order A. Printable flag set to N.	Subtract executed quantity from stored Order A. If stored Order A quantity is zero, delete the stored order.
8		C	Executed quantity and price for Order B. Printable flag set to N.	Subtract executed quantity from stored Order B. If stored Order B quantity is zero, delete the stored order.
9*	<i>Executed quantities for TMC legs are reported</i>	P	Executed quantity and price of each TMC leg. Printable flag set to Y. Occurred at Cross set to N. (TMC trades do not indicate that they resulted from an auction)	No change to order book. Update trade volume statistics.

## 5.6. Instrument Creation

### 5.6.1. TMC Creation

Step	Scenario	ITCH Message	Attributes	Activity
1	TMC instrument created	M	Combination name TMC_XXX_X_NNN	Combination name, Order Book ID and leg details provided.
2*		L	TMC_XXX_X_NNN tick table entry	The tick table definitions for the instrument.
3		O	Instrument series session state	Indicates active session state for instrument series.

### 5.6.2. Standard Instrument Creation

Step	Scenario	Tag	Attributes	Activity
1	Instrument Created	R	Instrument name XXX	Instrument name and Order Book ID provided.
2*		L	XXX tick table entry	The tick table definition for the instrument.
3		O	Instrument series session state	Indicates current session state for instrument series.

## 5.7. Miscellaneous Tasks

### 5.7.1. Customer to Customer Order Transfer

Step	Scenario – No Participant ID	ITCHTag	Attributes	Activity
1	Order A entered into book	A	Order ID of Order A. Order book position 1.	Store Order ID and order book position.
2	Customer to Customer Order Transfer from MMM to NNN	D	Order A	Delete the order from order book.
3		A	Order ID of Order A. Order book position 1.	Store Order ID and order book position.

Step	Scenario – With Participant ID	ITCHTag	Attributes	Activity
1	Order A entered into book	F	Order ID of Order A. Order book position 1.	Store Order ID and order book position.
2	Customer to Customer Order Transfer from MMM to NNN	D	Order A	Delete the order from order book.
3		F	Order ID of Order	Store Order ID and

Step	Scenario – With Participant ID	ITCHTag	Attributes	Activity
			A. Order book position 1.	order book position.

## 6. Appendix 2 – ASX MoldUDP64 Protocol Specification

### 6.1. Overview

MoldUDP64 is a networking protocol that allows efficient and scalable transmission of data messages in a “one transmitter to many listeners” scenario. MoldUDP64 is a lightweight protocol layer built on top of UDP that provides a mechanism for listeners to detect and re-request missed packets.

In MoldUDP64, each outbound packet is transmitted only once regardless of the number of listeners. Multiple messages may also be aggregated into a single network packet to reduce network traffic. Optional caching request servers can be placed nearby remote receivers to reduce latency and bandwidth over WAN links.

This section describes the messages sent between a MoldUDP64 server and its clients. MoldUDP64 transmitters send downstream packets via UDP multicast to transport the normal data stream sent to the listeners. These packets are also sent via UDP unicast in response to a Request message submitted by a listener. MoldUDP64 clients can send these Request messages to request the retransmission of any desired packets from the data stream.

The MoldUDP64 server will transmit on a well-known multicast group for each type of downstream MoldUDP64 data stream on a network. The listeners must subscribe to this multicast group to receive the downstream data. One or more request servers may also be deployed to service any unicast client requests for retransmission of specific messages. The listeners must be configured with these IP addresses and port combinations to which they can submit the requests.

### 6.2. Assumptions

All number fields in the MoldUDP64 messages specified in this document (i.e. sequence number, message counts and message lengths etc.) are binary numbers formatted in Big Endian mode (i.e. most significant byte first).



**Note:**

This need not apply to the data contained the Message Data fields of the Message Blocks.

### 6.3. Terms

#### 6.3.1. Message

A message is an atomic piece of information carried by the MoldUDP64 protocol.

MoldUDP64 can theoretically handle individual messages from zero bytes up to 64KB in length although individual messages should be kept small enough so that the UDP underlying network protocol can efficiently carry the resulting MoldUDP64 packets.

The contents of a MoldUDP64 message are defined by the higher level application.

#### 6.3.2. Session

A session is a sequence of one or more messages.

While a single session can last indefinitely, typically the application will define a session to logically group messages together based on time delimitation.

Once a session is terminated, no more messages can be sent on that session. Depending on the design of the MoldUDP64 system and the application, receivers may still be able to re-request messages from a terminated session.

A session is considered active if it has started but not yet been terminated.

## 6.4. Downstream Packet

A MoldUDP64 transmitter sends “downstream” packets that are received by MoldUDP64 listeners. A MoldUDP64 packet may contain a payload of 0 or more data stream messages.

Each MoldUDP64 packet consists of a Downstream Packet Header and of a series of Message Blocks. The Message Blocks carry the actual data of the stream.

### 6.4.1. Header

#### Downstream Packet Header

Field Name	Offset	Length	Value	Notes
Session	0	10	ANUM	Indicates the session to which this packet belongs.
Sequence Number	10	8	NUM	The sequence number of the first message in the packet.
Message Count	18	2	NUM	The count of messages contained in this packet.

### 6.4.2. Sequence Number

The Sequence Number field of the Packet Header indicates the sequence number of the first message in the packet. If there is more than one message contained in a packet, any messages following the first message are implicitly numbered sequentially.

### 6.4.3. Message Count

The number of Message Blocks contained in a MoldUDP64 packet is specified by the Message Count field of the Packet Header. The maximum payload size of a Downstream Packet is determined by the sender.



**Note:**

A Message Count of zero denotes a heartbeat and a Message Count of 0xFFFF (hex, or 65535 in decimal) denotes end of session

### 6.4.4. Message Block

The first field of a Message Block is the two byte Message Length. The remainder of the Message Block is the variable length Message Data field. The first Message Block field will always start immediately following the Header which is 20 bytes from the beginning of the packet. Subsequent Message Blocks will begin after the last byte of the previous Message Block.

#### Downstream Packet Message Block

Field Name	Offset	Length	Value	Notes
Message Length	*	2	NUM	Indicates the length in bytes of the message contained in the message block.
Message Data	*	*	ANUM	The message data.

\* = Variable Values

### 6.4.5. Message Length

The Message Length is an unsigned binary count representing the number of message data bytes following this Message Length field. The message length field value does not include the two bytes occupied by the message length field. The total size of the message block is the value of the message length field plus two.



#### 6.4.6. Message Data

The Message Data is actual data of the message being transmitted by MoldUDP64. It is variable length and can be zero length. The meaning of the data is application specific.

#### 6.4.7. Heartbeats

Heartbeats are sent periodically by the server so receivers can sense packet loss even during times of low traffic. Typically, these packets are transmitted every 50 milliseconds and contain the next expected Sequence Number. A heartbeat packet is a MoldUDP64 packet with a Message Count of zero.

#### 6.4.8. End of Session

When the current session is complete, Downstream Packets are sent with a Message Count of 0xFFFF (hex, or 65535 in decimal) for a short time in place of heartbeats. These Downstream Packets contain the next expected Sequence Number, just like heartbeats. While the End of Session messages persist, re-requests may be made on the current session. This is the last chance to ensure that all messages have been received.

#### 6.4.9. Request Packet

The Request Packet is sent to request the retransmission of a particular message or group of messages. The request packet is sent to a request server. A receiver may need to send this request when it detects a sequence number gap in received messages. The response to a valid Request Packet is a standard Downstream Packet unicast back to the source of the retransmission request. This allows downstream MoldUDP64 customers to read the retransmitted Downstream Packet in their multicast processing socket if the request was made from that socket. In other words, the client need only have one socket open to listen to the multicast and to process retransmissions, even though the retransmissions are not multicast.

##### Request Packet

Field Name	Offset	Length	Value	Notes
Session	0	10	ANUM	Indicates the session to which this packet belongs.
Sequence Number	10	8	NUM	First requested sequence number.
Requested Message Count	18	2	NUM	The number of messages requested for retransmission.

#### 6.4.10. Sequence Number

The Sequence Number field of the Packet Header indicates the sequence number of the first message requested.

#### 6.4.11. Requested Message Count

The Message Count indicates how many messages should be retransmitted. If the total size of the requested messages exceeds the maximum payload size of the of one UDP packet, only the number of messages that completely fit will be returned. Additional retransmission requests must be made for the subsequent messages if they are still desired.

### 6.5. Receiver Example

A typical MoldUDP64 receiver client would be configured with the following parameters:

- The UDP port to listen on and the Multicast group to join
- A list of one or more Request Servers that are available to answer retransmission requests for this stream. Each server is specified as a host IP address and a UDP port to which to send requests.

- A session and sequence number of the next expected message if the client is being restarted.

A typical MoldUDP64 receiver client may have the following process:

1. Subscribe to multicast.
2. Buffer messages received.
3. Connect to Glimpse.
4. Process Glimpse messages.
5. Receive G message with sequence number.
6. Discard ASX ITCH message below sequence number.
7. Process buffered ASX ITCH messages remaining.
8. Continue to process new incoming ASX ITCH messages.
9. Send retransmit request for any missing sequences.

## 7. Appendix 3 - SoupBinTCP Protocol

### 7.1. Overview

SoupBinTCP is a lightweight point-to-point protocol, built on top of TCP/IP sockets that allow delivery of a set of sequenced messages from a server to a client in real-time. SoupBinTCP guarantees that the client receives each message generated by the server in sequence, even across underlying TCP/IP socket connection failures.

SoupBinTCP clients can send messages to the server. These messages are not sequenced and may be lost in the case of a TCP/IP socket failure.

SoupBinTCP is ideal for systems where a server needs to deliver a logical stream of sequenced messages to a client in real-time but does not require the same level of guarantees for client generated messages either because the data stream is unidirectional or because the server application generates higher-level sequenced acknowledgments for any important client-generated messages.

SoupBinTCP is designed to be used in conjunction with higher level protocols that specify the contents of the messages that SoupBinTCP messages deliver. The SoupBinTCP protocol layer is opaque to the higher-level messages.



**Note:**

Unlike the ASCII version, messages may include any possibly byte.

SoupBinTCP also includes a simple scheme that allows the server to authenticate the client on login.

#### 7.1.1. SoupBinTCP Logical Packets

The SoupBinTCP client and server communicate by exchanging a series of logical packets.

Each SoupBinTCP logical packet has a:

- Two byte big-endian length that indicates the length of rest of the packet (meaning the length of the payload plus the length of the packet type – which is 1)
- Single byte header which indicates the packet type
- Variable length payload.

#### SoupBinTCP Logical Packet Structure

Two Byte Packet Length	Packet Type	Variable Length Payload
------------------------	-------------	-------------------------



**Note:**

The SoupBinTCP logical packets do not necessarily map directly to physical packets on the underlying network socket; they may be broken apart or aggregated by the TCP/IP stack.

The SoupBinTCP protocol does not define a maximum payload length.

Unlike the ASCII version, the payload may contain the line feed character or any character.

#### 7.1.2. Protocol Flow

A SoupBinTCP connection begins with the client opening a TCP/IP socket to the server and sending a Login Request Packet. If the login request is valid, the server responds with a Login Accepted Packet and begins sending Sequenced Data Packets. The connection continues until the TCP/IP socket is broken.

Each Sequenced Data Packet carries a single higher-level protocol message. Sequenced Data Packets do not contain an explicit sequence number; instead both client and server compute the sequence number locally by counting messages as they go.

The sequence number of the first sequenced message in each session is always 1.

Typically, when initially logging into a server the client will set the Requested Sequence Number field to 1 and leave the Requested Session field blank in the Login Request Packet. The client will then inspect the Login Accepted Packet to determine the currently active session. Starting at 1, the client begins incrementing its local sequence number each time a Sequenced Data Packet is received. If the TCP/IP connection is ever broken, the client can then re-log into the server indicating the current session and its next expected sequence number. By doing this, the client is guaranteed to always receive every sequenced message in order, despite TCP/IP connection failures.

SoupBinTCP also permits the client to send messages to the server using Un-sequenced Data Packets at any time after the Login Accepted Packet is received. These messages may be lost during TCP/IP socket connection failures.

### 7.1.3. Heartbeats

SoupBinTCP uses logical heartbeat packets to quickly detect link failures. The server must send a Server Heartbeat packet anytime more than 1 second has passed since the server last sent any data. This ensures that the client will receive data on a regular basis. If the client does not receive anything (neither data nor heartbeats) for an extended period of time, it can assume that the link is down and attempt to reconnect using a new TCP/IP socket.

Similarly, once logged in, the client must send a Client Heartbeat packet anytime more than 1 second has passed since the client last sent anything. If the server doesn't receive anything from the client for an extended period of time (typically 15 seconds), it can close the existing socket and listen for a new connection.

### 7.1.4. End of Session Marker

The server indicates that the current session has terminated by sending an End of Session Message. This indicates that there will be no more messages contained in this session.

The client will have to reconnect and login with the new Session ID or a blank (space filled) Session ID to begin receiving messages for the next available session.

### 7.1.5. Data Types

Character data fields are standard ASCII bytes. Integer fields are binary big-endian values.

## 7.2. SoupBinTCP Packet Types

### 7.2.1. Debug Packet

A debug packet can be sent by either side of a SoupBinTCP connection at any time. Debug packets are intended to provide human readable text that may aid in debugging problems. Debug Packets should be ignored by both client and server application software.

#### Debug Packet

Field Name	Offset	Length	Value	Notes
Packet Length	0	2	Integer	Number of bytes after this field until the next packet.
Packet Type	2	1	"+"	Debug Packet.
Text	3	*	Alpha Numeric	Free form human readable text.

\* = Variable Values

## 7.2.2. Logical Packets Sent by a SoupBinTCP Server

### 7.2.2.1. Login Accepted Packet

The SoupBinTCP server sends a Login Accepted Packet in response to receiving a valid Login Request from the client. This packet will always be the first non-debug packet sent by the server after a successful login request.

#### Login Accepted Packet

Field Name	Offset	Length	Value	Notes
Packet Length	0	2	Integer	Number of bytes after this field until the next packet.
Packet Type	2	1	"A"	Login Accepted Packet.
Session	3	10	Alpha Numeric	The session ID of the session that is now logged into. Left padded with spaces.
Sequence Number	13	20	Numeric	The sequence number in ASCII of the next Sequenced Message to be sent. Left padded with spaces.

### 7.2.2.2. Login Rejected Packet

The SoupBinTCP server sends this packet in response to an invalid Login Request Packet from the client. The server closes the socket connection after sending the Login Reject Packet. The Login Rejected Packet will be the only non-debug packet sent by the server in the case of an unsuccessful login attempt.

#### Login Reject Packet

Field Name	Offset	Length	Value	Notes
Packet Length	0	2	Integer	Number of bytes after this field until the next packet.
Packet Type	2	1	"J"	Login Rejected Packet.
Reject Reason Code	3	10	Alpha Numeric	See Login Reject Codes below.

#### Login Reject Codes

Code	Explanation
"A"	Not Authorised. There was an invalid username and password combination in the Login Request Message.
"S"	Session not available. The Requested Session in the Login Request Packet was either invalid or not available.

### 7.2.2.3. Sequenced Data Packet

The Sequenced Data Packets act as an envelope to carry the actual sequenced data messages that are transferred from the server to the client. Each Sequenced Data Packet carries one message from the higher-level protocol. The sequence number of each message is implied; the initial sequence number of the first Sequenced Data Packet for a given TCP/IP connection is specified in the Login Accepted Packet and the sequence number increments by 1 for each Sequenced Data Packet transmitted.

Since SoupBinTCP logical packets are carried via TCP/IP sockets, the only way logical packets can be lost is in the event of a TCP/IP socket connection failure. In this case, the client can reconnect to the server and request the next expected sequence number and pick up where it left off.

**Sequence Data Packet**

Field Name	Offset	Length	Value	Notes
Packet Length	0	2	Integer	Number of bytes after this field until the next packet.
Packet Type	2	1	"S"	Sequenced Data Packet.
Message	3	*	Anv	Defined by a higher-level protocol, but must not contain any embedded linefeeds.

\* = Variable values

**7.2.2.4. Server Heartbeat Packet**

The server should send a Server Heartbeat Packet anytime more than 1 second passes where no data has been sent to the client. The client can then assume that the link is lost if it does not receive anything for an extended period of time.

**Server Heartbeat Packet**

Field Name	Offset	Length	Value	Notes
Packet Length	0	2	Integer	Number of bytes after this field until the next packet.
Packet Type	2	1	"H"	Server Heartbeat Packet.

**7.2.2.5. End of Session Packet**

The server will send an End of Session Packet to denote that the current session is finished. The connection will be closed shortly after this packet, and the customer will no longer be able to reconnect to the current session.

**End of Session Pack**

Field Name	Offset	Length	Value	Notes
Packet Length	0	2	Integer	Number of bytes after this field until the next packet.
Packet Type	2	1	"Z"	End of Session Packet

**7.2.3. Logical Packets Sent by the SoupBinTCP Client****7.2.3.1. Login Request Packet**

The SoupBinTCP client must send a Login Request Packet immediately upon establishing a new TCP/IP socket connection to the server.

Client and server must have mutually agreed upon the username and password fields. They provide simple authentication to prevent a client from inadvertently connecting to the wrong server.

Both username and password are case-insensitive and should be padded on the right with spaces.

The server can terminate an incoming TCP/IP socket if it does not receive a Login Request Packet within a reasonable period of time (typically 30 seconds).

**Login Request Packet**

Field Name	Offset	Length	Value	Notes
Packet Length	0	2	Integer	Number of bytes after this field until the next packet.
Packet Type	2	1	"L"	Login Request Packet

Field Name	Offset	Length	Value	Notes
Username	3	6	Alpha-numeric	Username
Password	9	10	Alpha-numeric	Password
Requested Session	19	10	Alpha-numeric	Specifies the session the client would like to log into, or all blanks to log into the currently active session.
Requested Sequence Number	29	20	Numeric	Specifies the next sequence number in ASCII the client wants to receive upon connection, or 0 to start receiving the most recently generated message.

### Glimpse Login

The following is a tcpdump capture displaying the sequence of bytes sent for a successful Glimpse Login.

Length of message=47 (%x2f). 2 bytes, big endian

Login Request Type="L". 1 byte

Username. 6 bytes space padded

```

13:32:46.795458 IP 172.31.8.2.47981 > 203.0.119.228.21804: P 0:49(49) ack 1 win 32
0x0000: 0002 c92e 7580 001c 730f 2e80 0800 4500 .....E.
0x0010: 0059 1994 4000 3f06 2b05 ac1f 0802 cb00 .Y..@.?+.....
0x0020: 77e4 bb6d 552c b9c7 3852 a62d 7621 5018 w..mU,..6R.-v!P.
0x0030: 0020 d3b5 0000 002f 4c46 4153 5830 3161 ...../LFASX01a
0x0040: 7378 7472 6164 6520 2020 2020 2020 2020 extrade.....
0x0050: 2020 2031 2020 2020 2020 2020 2020 2020 ..1.....
0x0060: 2020 2020 2020 20
13:32:46.799316 IP 172.31.8.2.47981 > 203.0.119.228.21804: . ack 34 win 32
0x0000: 0002 c92e 7580 001c 730f 2e80 0800 4500 .....E.

```

Password. 10 bytes, space padded

Sequence Number="1". 10 bytes space padded

Session (blank). 10 bytes, space padded



#### Note:

The spaces required after the sequence number to fill the packet, and ensure numbers are left justified.

ASX ITCH uses big endians.

Big endian stores the most significant to the least significant bytes from left (lowest address) to right, i.e. it stores 47 (%x2f) in a short (16 bit) number as 00-2f.

This differs to a Little-endian which stores the least significant in the left, up to the most significant byte in the right, i.e. 2f-00.

### 7.2.3.2. Un-sequenced Data Packets

The Un-sequenced Data Packets act as an envelope to carry the actual data messages that are transferred from the client to the server. These messages are not sequenced and may be lost in the event of a socket failure. The higher-level protocol must be able to handle these lost messages in the case of a TCP/IP socket connection failure.

#### Un-sequenced Data Packet

Field Name	Offset	Length	Value	Notes
Packet Length	0	2	Integer	Number of bytes after this field until the next packet.

Field Name	Offset	Length	Value	Notes
Packet Type	2	1	"U"	Un-sequenced Data Packet
Message	3	*	Alpha-numeric	Defined by a higher-level protocol, but must not contain any embedded linefeeds.

\* = Variable values

### 7.2.3.3. *Client Heartbeat Packets*

The client should send a Client Heartbeat Packet anytime more than 1 second passes where no data has been sent to the server. The server can then assume that the link is lost if it does not receive anything for an extended period of time.

#### **Client Heartbeat Packet**

Field Name	Offset	Length	Value	Notes
Packet Length	0	2	Integer	Number of bytes after this field until the next packet.
Packet Type	2	1	"R"	Logout Request Packet



## 8. Document Version Control

Date	Version	Change	Requested By	Author
30/03/15	V2.0	The Login Rejected Packet, the Packet Type Field Name was changed had the value I changed to J.	Silvia Ohr	Joanne Mottram

### Information Classification - Public

### Disclaimer

This document provides general information only and may be subject to change at any time without notice. ASX Limited (ABN 98 008 624 691) and its related bodies corporate ("ASX") makes no representation or warranty with respect to the accuracy, reliability or completeness of this information. To the extent permitted by law, ASX and its employees, officers and contractors shall not be liable for any loss or damage arising in any way, including by way of negligence, from or in connection with any information provided or omitted, or from anyone acting or refraining to act in reliance on this information. The information in this document is not a substitute for any relevant operating rules, and in the event of any inconsistency between this document and the operating rules, the operating rules prevail to the extent of the inconsistency.

### ASX Trademarks

The trademarks listed below are trademarks of ASX. Where a mark is indicated as registered it is registered in Australia and may also be registered in other countries. Nothing contained in this document should be construed as being any licence or right to use of any trademark contained within the document.

ASX®, ASX Trade®, ASX TradeMatch®, PureMatch®