XDP COMMON CLIENT SPECIFICATION

Preface

DOCUMENT HISTORY

The following table provides a description of all changes to this document.

Version	Date	Change Description
1.5	11/6/2013	Section 1.3.14 Symbol Index Mapping FTP – Update to Listed Markets, Section 2.13 Message Body – Update to Security Types Section 2.18 MSG Body – Updates to various fields
1.6	4/9/2014	Updated Security Status and Symbol Index messages to include notes reflecting current implementation Updated Security Status fields to have standalone SSRState, MarketState, and SessionState fields
2.0	01/08/2015	Re-branded for ICE Structure and explanatory text revised for clarity and simplicity NYSE Security Status msg's Transaction ID field changed to Reserved Source Time Reference msg's SymbolIndex field changed to System ID IBF 2.0
2.0a	02/16/2015	Noted shorter length of Symbol Index Mapping (msg type 3) and Security Status (msg type 34) for Arca Integrated Feed only Minor tweaks to language
2.0b	02/18/2015	Corrections in section 3.6.2 regarding Order ID's Corrected Packet Header DeliveryFlags for Seq Num Reset messages (4.3 and 9.2)

REFERENCE MATERIAL

The following lists the associated documents, which either should be read in conjunction with this document or which provide other relevant information for the user:

- SFTI US Technical Specification

- SFTI US Customer Guide

- NYSE Symbology

CONTACT INFORMATION

Service Desk (NYSE, NYSE MKT, NYSE Arca)

- Australia Toll Free: +61.1800506227
- China Toll Free: +86.4008822979
- US Toll Free +1.866.873.7422
- UK Toll Free 0808.101.3866
- US Paid +1.212.656.3640
- Email: service.desk@nyx.com

FURTHER INFORMATION

- For additional information about market data products, visit http://www.nyxdata.com/Data-Products

- For updated capacity figures, visit our capacity pages at: http://www.nyxdata.com/capacity

- For details of IP addresses, visit our IP address pages at: http://www.nyxdata.com/ipaddresses

- For a full glossary, visit: http://www.nyxdata.com/glossary/

Contents

1. Introduction

1.1 RECEIVING REAL TIME MARKET DATA

Real-time XDP data is published in the form of messages with fixed length fields. All fields are binary except a very small number that are in ASCII format. For efficient use of the network, the messages are bundled into application packets, and the packets are published via the multicast protocol.

For capacity reasons, packets are routed over a number of predefined data sets called channels. Each channel is duplicated and published to two distinct multicast groups for redundancy. The two redundant multicast groups per channel (called lines) are referred to as line A and line B. The union of the data in all channels that make up a product is called a feed.

The IP addresses and port numbers of the production and test channels for each XDP feed can be found at www.nyxdata.com/ipaddresses. A client application receives a product by subscribing to some or all of the channels that make up the feed.

1.2 RECOVERING FROM ERRORS
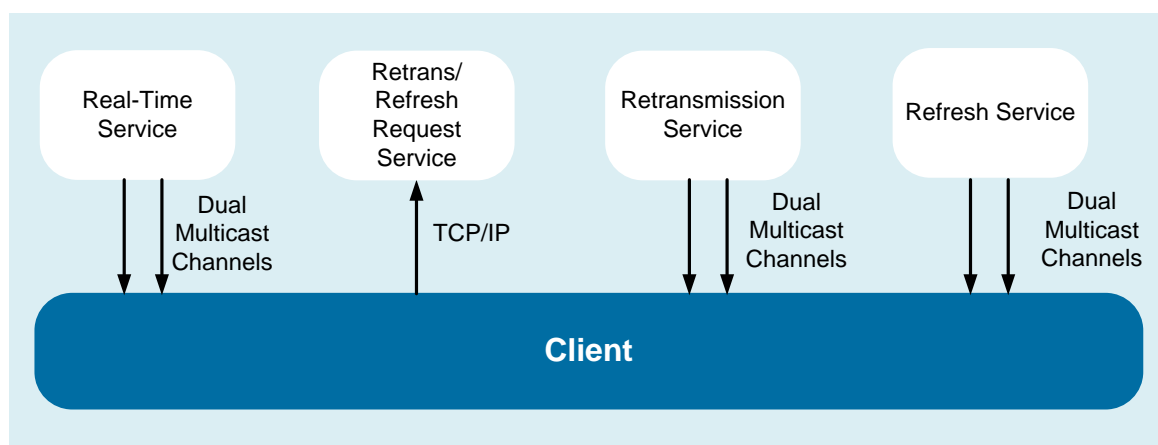
- In case of dropped multicast packets, the client can connect to a Request Server via TCP/IP to request retransmissions of missed messages.

- In case of client late start or intraday failure, the client can connect to the Request Server and request snapshot refreshes of the state of the market.

- At system startup, each channel publishes referential data about all symbols published on the channel. If a client process misses this initial spin of symbol data, he can connect to the Request Server and request a refresh of some or all of the missed data.

In response to these requests, retransmission and refresh data is published by the exchange over dedicated multicast channels which correspond one-to-one with the real-time channels.

See Error Handling and the Request Server for complete information.

2.   Packets and Heartbeats

2.1 PACKET HEADER

All packets sent on any XDP feed have an XDP Packet Header followed by one or more messages (with the exception of Heartbeat packets which do not contain any messages).

The maximum length of a packet is 1400 bytes, so no message can be longer than 1400 – 16 bytes (max packet size - the length of the Packet Header).

Packet Header Structure

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
PktSize	0	2	Binary	The size of the packet in bytes, including this 16-byte packet header
DeliveryFlag	2	1	Binary	A flag that indicates whether this is an original, retransmitted, or 'replayed' message. Valid values include: ▪ 1 – Heartbeat ▪ 10 – XDP Failover (see XDP Publisher Failover) ▪ 11 – Original Message ▪ 12 – Sequence Number Reset Message ▪ 13 – Only one packet in retransmission sequence ▪ 15 – Part of a retransmission sequence ▪ 17 – Only one packet in Refresh sequence ▪ 18 – Start of Refresh sequence ▪ 19 – Part of a Refresh sequence ▪ 20 – End of Refresh sequence ▪ 21 – Message Unavailable
NumberMsgs	3	1	Binary	The number of messages in this packet
SeqNum	4	4	Binary	The message sequence number of the first message in this packet
SendTime	8	4	Binary	The time when this packet was published to the multicast channel, in seconds since Jan 1, 1970 00:00:00 UTC.
SendTimeNS	12	4	Binary	The nanosecond offset from the Send Time

2.2 HEARTBEATS

Application heartbeats are sent once a minute by the real-time publishing servers and by the Request Server so that the client can confirm connection health.

A heartbeat consists of a packet containing a Packet Header and no messages. The Packet Header's Delivery Flag is set to 1 and Number Msgs is 0. Since a heartbeat packet contains no messages, a heartbeat does not increment the next expected sequence number. See Sequence Numbers.

Heartbeats sent by the Request Server must be acknowledged by the client. See Request Server.

3.   Message Field Content

Messages are contiguous data structures consisting of fixed-length fields. No names or 'tags' appear in the message.

- Message fields align on 1 byte boundaries, so there are no filler fields for alignment purposes

- Binary fields are published in Little-Endian ordering

- All ASCII string fields are left aligned and null padded

- Segmentation of messages across packets is not supported, so a message will never straddle a packet boundary.

- The length of a message as actually published may differ from the length of the message structure defined in the client specifications. See Msg Size Field below for details.

3.1 MESSAGE HEADER
The format of each message varies according to type, but each type starts with a standard two-byte message header:

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	The size of this message in bytes
MsgType	2	2	Binary	The type of this message

3.1.1   Msg Size Field

In order to handle future releases of XDP feeds smoothly, clients should never hard code msg sizes in feed handlers. Instead, the feed handler should use the Msg Size field to determine where the next message in a packet begins.

This allows

- Support of XDP format variations among markets
- Client flexibility when revised message structures go live in production

In example 1 below, a message type is defined in the specification to have different lengths in different markets. The trailing field is not published in the Arca market. An Arca-coded client can process NYSE data correctly (but of course cannot use the trailing Volume field without field-specific coding).

Example 1: Message type with format variations across markets

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION	
Msg Size	0	2	Binary	Size of the message. **NYSE – 24 bytes** **NYSE MKT – 24 bytes** **NYSE Arca - 20 bytes**	Look at the Msg Size field to know where the next message starts.
Msg Type	2	2	Binary	The type of this message: 998 – Example 1 msg type	
SourceTimeNS	4	4	Binary		
SymbolIndex	8	4	Binary		
OrderID	12	4	Binary		
Price	16	4	Binary		
Volume	28	4	Binary	Not published in Arca market	Market-specific content

The variable message size can also insulate client code from future field additions that you may not need.

In example 2, an existing message type is 16 bytes long.

Example 2: Release N

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary	Size of the message: 16 bytes
Msg Type	2	2	Binary	The type of this message: 999 – Price message example
SourceTimeNS	4	4	Binary	
SymbolIndex	8	4	Binary	
Price	12	4	Binary	

Look at the Msg Size field to know where the next message starts.

In a future release, a four-byte volume field will be added, increasing the Msg Size to 20 bytes.

If the client wishes to delay upgrading his feed handler for the new content, no coding is needed at the time of the release. Proper coding of the MsgSize field up front allows the client to handle the unforeseen 20-byte format.   On his own schedule, the client can upgrade his feed handler to process the new field.

Example 2: Release N+1: a new field is added

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary	Size of the message: 20 bytes
Msg Type	2	2	Binary	The type of this message: 999 – Price message example
SourceTimeNS	4	4	Binary	
SymbolIndex	8	4	Binary	
Price	12	4	Binary	
Volume	16	4	Binary	New field

Look at the Msg Size field to know where the next message starts.

Unmodified clients can handle longer message structure (but can't benefit from new content)

3.2 DATE AND TIME CONVENTIONS

Dates and times are in UTC (Universal Time, Coordinated), and are expressed in nanoseconds since the Unix Epoch (Jan 1, 1970 00:00:00).   A complete timestamp consists of two 4-byte fields: seconds since the Unix Epoch, and nanoseconds within the current second, as in a Unix timespec structure.

The XDP Packet Header contains SendTime and SendTimeNS fields to show the time that the packet was published to the wire by the XDP Publisher.

Most XDP messages additionally contain a timestamp called Source Time to shows the time of the Matching Engine event that caused the publication of this message.

Many of the higher-volume XDP feeds such as Integrated and BBO explicitly publish only the nanoseconds portion of the Source Time in each message. The seconds portion is explicitly published in a Source Time Reference Message (Msg Type 2) once a second.

For NYSE and MKT Integrated feeds, Source Time Reference messages are published per Matching Engine partition (server). For all other products, they are published per symbol. Publishing per Matching Engine partition results in a much lower volume of Time Reference messages, and this approach will become standard across all feeds in future releases.

3.3 SEQUENCE NUMBERS

Each message in a given channel is assigned a unique sequence number. Sequence numbers increase monotonically per channel, and can be used to detect publication gaps.

To optimize publication efficiency, the sequence number is not explicitly published in each message. Instead, the Packet Header contains the sequence number of the first message in the packet, along with the number of messages in the packet. Using these fields, the client can easily associate the correct sequence number with each message.

The sequence number combined with the channel ID form a message ID which is unique across the feed.

3.4 SYMBOL SEQUENCE NUMBERS

In addition to the sequence number, many message types explicitly include a field called Symbol Sequence Number, which identifies the message's position in the sequence of all messages published by the feed for a given symbol.

Clients who are tracking only a small number of symbols may opt to ignore sequence numbers and track only Symbol Sequence Numbers for each symbol of interest. If such a client ever experiences a Symbol Sequence Number gap, he can request a refresh for that symbol.

3.5 PRICES

All price fields are published as binary integers. To interpret a price correctly, the client must use the published price value as a numerator along with the Price Scale Code in the symbol's Symbol Index Mapping Message (Msg Type 3) as follows:

$$Price = \frac{Numerator}{10^{PriceScaleCode}}$$

For example, a price of $27.56 is represented as a published price field of 2756 and a PriceScaleCode of 2.

3.6 ORDER ID'S AND TRADE ID'S

The Order ID and the Trade ID in order-based feeds such as NYSE Integrated Feed are binary integers that uniquely identify an order or an execution. Order IDs are valid for the trading day only. Unexecuted, uncancelled GTC orders are assigned a new Order ID on the next trading day.

3.6.1   Matching Trade ID's to order entry messages

At present, Trade IDs do not correspond to any ID in the order entry messaging formats.

After future matching engine releases, Trade IDs will correspond to the exchange-generated ID returned to the client in the Execution Report. This will make it easy for the client to find his own trades in the market data feeds.

3.6.2   Matching Order ID's to order entry messages

In the NYSE and MKT Integrated Feed, Order ID's are 8 bytes long and they correspond to the 4 byte MEOrderID in the order entry messaging APIs.

At present in the Arca market, Order ID's are 4 bytes long. They can be combined with other fields present in the feeds to match them up with the exchange-generated ID that is returned to the client in the order acknowledgement message on order entry.

After future matching engine releases, Order IDs for all markets

- Will be 8 bytes long
- Will correspond to the 8-byte exchange Order ID returned to the client in the order entry APIs
- Will be unique per market across all symbols
-

Matching Arca XDP Order ID with the Order Acknowledgement Message

The Arca matching engine generates a unique 64-bit order ID that consists of four concatenated data values - the Order ID, Market ID, System ID, and GTCIndicator. In all XDP feeds, the Market ID, and System ID are provided in the Symbol Index Mapping, since these values are static for the trading day. The Order ID and GTCIndicator are contained in the order-related data messages.

The table below shows the data structure required to combine the Order ID, Market ID, System ID, and GTC Indicator to obtain the full UTP matching engine Order ID. The table assumes the client byte ordering is Little Endian. If the client byte ordering is Big Endian, the byte order is reversed.

Matching Arca XDP Order ID with Order Acknowledgment Message

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
GTCIndicator	0	1	Binary	This field specifies if Trade Order ID is a GTC order: ▪ 0 – Day Order ▪ 1- GTC Order
System ID	1	1	Binary	ID of the Originating System in the Symbol Index Mapping
Market ID	2	2	Binary	ID of the Originating market in the Symbol Index Mapping
OrderID	4	4	Binary	Order ID

3.7 SYMBOL INDEXES

In all XDP feeds, symbol-specific referential data is published in a Symbol Index Mapping Message (Msg Type 3) at system startup. Symbol Index Mapping messages appear in each channel only for the symbols that appear in that channel.

Any client who misses this initial spin can request a refresh of Symbol Indexes by sending a Symbol Index Mapping Request Message (Msg Type 13) to the Request Server. The requested Symbol Index Mapping messages will be re-published over the Refresh channels.

The Symbol Index Mapping message includes the ASCII symbol in NYSE format along with a unique ID called a Symbol Index. Other symbol-specific messages such as Trade and BBO messages contain only the Symbol Index and no other referential data.

Symbol Indexes are unique per market only, so for example the Symbol Index for IBM may be different in the Arca equities market vs. the NYSE market. However in any one market, Symbol Indexes are the same every day for each symbol, and Symbol Indexes for obsolete instruments are not re-used.

After future releases of the matching engine, Symbol Indexes will be the same across all markets.

4.   Messages Sent by the Publisher

4.1 SYMBOL INDEX MAPPING MESSAGE (MSG TYPE 3)

This message is published over the real-time data channels at system startup or in the context of a refresh sequence after a Matching Engine or XDP Publisher failover. It provides referential data for a single specified symbol.

See Symbol Indexes for more information.

Note that for the NYSE Arca Integrated Feed, this message is only 38 bytes long, and ends with the Round Lot field. In a future release, Arca Integrated will migrate to version 2.0, and at that time, this message will be the full 44 bytes long.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary	Size of the message: ▪ All feeds but Arca Integrated - 44 ytes ▪ Arca Integrated – 38 Bytes
Msg Type	2	2	Binary	The type of this message: 3 – Symbol Index Mapping Message
SymbolIndex	4	4	Binary	The unique ID of this symbol for all products within this market. This ID cannot be used to cross reference a security between markets.
Symbol	8	11	ASCII	Null-terminated ASCII symbol in NYSE Symbology. For more information, see the NYSE Symbology Spec.
Reserved	19	1	Binary	This field is reserved for future use
Market ID	20	2	Binary	ID of the Originating Market: ▪ 1 - NYSE Cash ▪ 3 – NYSE Arca Cash ▪ 4 – NYSE Arca Options ▪ 5 – NYSE Bonds ▪ 6 – Global OTC ▪ 8 – NYSE Amex Options ▪ 9 - NYSE MKT Cash
System ID	22	1	Binary	ID of the Originating System.
Exchange Code	23	1	ASCII	Exchange where the this symbol is listed: ▪ 'N' – NYSE ▪ 'P' – NYSE Arca ▪ 'Q' – NASDAQ ▪ 'A' – NYSE MKT ▪ 'U' – OTCBB symbols for Global OTC ▪ 'V' – Other OTC symbols for Global OTC ▪ 'Z' – BATS
PriceScaleCode	24	1	Binary	Specifies placement of the decimal point in price fields for this security. See Prices .
Security Type	25	1	ASCII	Type of Security used Arca: ▪ 'A' – ADR ▪ 'C' - COMMON STOCK ▪ 'D' – DEBENTURES ▪ 'E' – ETF ▪ 'F' – FOREIGN ▪ 'H' – AMERICAN DEPOSITARY SHARES ▪ 'I' – UNITS

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
				▪ 'L' – INDEX LINKED NOTES ▪ 'M' - MISC/LIQUID TRUST ▪ 'O' – ORDINARY SHARES ▪ 'P' - PREFERRED STOCK ▪ 'R' – RIGHTS ▪ 'S' - SHARES OF BENEFICIARY INTEREST ▪ 'T' – TEST ▪ 'U' – UNITS ▪ 'W' – WARRANT Type of Security used by NYSE and NYSE MKT: ▪ 'A' – COMMON STOCK ▪ 'B' – PREFERRED STOCK ▪ 'C' – WARRANT ▪ 'D' – RIGHT ▪ 'E' – CORPORATE BOND ▪ 'F' – TREASURY BOND ▪ 'G' – STRUCTURED PRODUCT ▪ 'H' – ADR COMMON ▪ 'I' – ADR PREFERRED ▪ 'J'-ADR WARRANTS ▪ 'K' – ADR RIGHTS ▪ 'L' – ADR CORPORATE BOND ▪ 'M' – NY REGISTERED SHARE ▪ 'N' – GLOBAL REGISTERED SHARE ▪ 'O' – INDEX ▪ 'P' – FUND ▪ 'Q' – BASKET ▪ 'R' – UNIT ▪ 'S' – LIQUIDATING TRUST ▪ 'U' - UNKOWN
Lot Size	26	2	Binary	Round lot size in shares.
PrevClosePrice	28	4	Binary	The previous day's closing price for this security. Use the price scale code.
PrevCloseVolume	32	4	Binary	The previous day's closing volume for the security. Use the price scale code.
Price Resolution	36	1	Binary	▪ 0 - All Penny ▪ 1 - Penny/Nickel ▪ 5 - Nickel/Dime
Round Lot	37	1	ASCII	Round Lots Accepted: ▪ 'Y' – Yes ▪ 'N' – No
MPV	38	2	Binary	Note: This field is available on NYSE and NYSE MKT only, and is left as a future enhancement on Arca. Clients will be notified upon availability.
Unit of Trade	40	2	Binary	This field specifies the security Unit of Trade in shares. Valid values are 1, 10, 50 and 100 Note: This field is available on NYSE and NYSE MKT only, and is left as a future enhancement on Arca. Clients will be notified upon availability.
Reserved	42	2	Binary	Reserved for future use. Disregard any content.

4.2 SECURITY STATUS MESSAGE (MSG TYPE 34)

This message will be sent to inform clients of significant events regarding a specific security, such as Trading Halts, Short Sale Restriction state changes, etc.

Note that for the NYSE Arca Integrated Feed, this message is only 22 bytes long, and ends with the Halt Condition field. In a future release, Arca Integrated will migrate to version 2.0, and at that time, this message will be 45 bytes long.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: ▪ All feeds but Arca Integrated Feed - 46 Bytes ▪ Arca Integrated Feed - 22 Bytes
MsgType	2	2	Binary	The type of this message: 34 – Security Status Message
SourceTime	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
SourceTimeNS	8	4	Binary	The nanosecond offset from the SourceTime
SymbolIndex	12	4	Binary	The unique ID of the symbol in the Symbol Index msg
SymbolSeqNum	16	4	Binary	The unique ID of this message in the sequence of messages published for this specific symbol.
Security Status	20	1	ASCII	The following are Halt Status Codes: ▪ '3' - Opening Delay (NYSE/MKT only) ▪ '4' - Trading Halt ▪ '5' - Resume ▪ '6' - No open/no resume (NYSE/MKT only) The following are Short Sale Restriction Codes: ▪ 'A' – Short Sale Restriction Activated (Day 1) ▪ 'C' – Short Sale Restriction Continued (Day 2) ▪ 'D' - Short Sale Restriction Deactivated ▪ 'E' – Short Sale Restriction in Effect – Arca only. Appears in the event of a prior day correction/cancel affecting the Short Sale restriction. NYSE Market State values : ▪ 'O' – Opened ▪ 'P' – Pre-opening ▪ 'X' -- Closed The following values are the Price Indication values: ▪ 'T' – T - Time ▪ 'I' – Price Indication ▪ 'G' – Pre-Opening Price Indication ▪ 'R' – Rule 15 Indication.
Halt Condition	21	1	ASCII	▪ '0x20' – Not applicable ▪ '~' - Security not delayed/halted ▪ 'D' - News dissemination ▪ 'I' - Order imbalance ▪ 'P' - News pending ▪ 'M' – LULD pause ▪ 'S' - Related security (not used) ▪ 'X' - Equipment changeover ▪ 'Z' - No open/No resume

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
				Market Wide Circuit Breakers: ▪ '1' - Market Wide Circuit Breaker Halt Level 1 ▪ '2' - Market Wide Circuit Breaker Halt Level 2 ▪ '3' - Market Wide Circuit Breaker Halt Level 3
Reserved	22	4	Binary	Future use. Any field content should be ignored.
Price 1	26	4	Binary	Default value is 0. ▪ If securityStatus = 'A', then this is the SSR Triggering Trade Price ▪ If securityStatus = 'G', then this is Pre-Opening Low Price Indication. ▪ If securityStatus = 'I', then this is Low Price Indication ▪ If securityStatus = 'R', then this is Rule 15 Low Indication Price. Note: This field is not yet implemented and is defaulted to binary 0. Clients will be notified upon availability.
Price 2	30	4	Binary	Default value is 0 ▪ If securityStatus = 'I', then this is High Price Indication ▪ If securityStatus = 'G', then this is Pre-Opening Price Indication ▪ If securityStatus = 'R', then this is Rule 15 High Price Indication Note: This field is not yet implemented and is defaulted to binary 0. Clients will be notified upon availability.
SSR Triggering Exchange ID	34	1	Binary	Default value is 0. This field is only populated when securityStatus = 'A'. This field is the Binary representation of the character values mentioned below. Valid Values are: ▪ 'N' – NYSE ▪ 'P' – NYSE Arca ▪ 'Q' – NASDAQ ▪ 'A' – NYSE MKT ▪ 'U' – OTCBB symbol for Global OTC ▪ 'V' – Other OTC symbols for Global OTC ▪ 'B' – NASDAQ OMX BX ▪ 'C' – NSX ▪ 'D' – FINRA ▪ 'I' – ISE ▪ 'J' – EDGA ▪ 'K' – EDGX ▪ 'M' – CHX ▪ 'N' – NYSE ▪ 'S' – CTS ▪ 'T' – NASDAQ OMX ▪ 'W' – CBSX ▪ 'X' – NASDAQ OMX PSX ▪ 'Y' – BATS Y ▪ 'Z' – BATS

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
				Note: This field is not yet implemented and is defaulted to binary 0. Clients will be notified upon availability..
SSR Triggering Volume	35	4	Binary	Default value is 0. This field is only populated when securityStatus = 'A' Note: This field is not yet implemented and is defaulted to binary 0. Clients will be notified upon availability.
Time	39	4	Binary	Default value is 0. ▪ If securityStatus = 'A' , then this is SSR Trigger Time ▪ If securityStatus = 'T', then it is T-Time Note: This field is not yet implemented and is defaulted to binary 0. Clients will be notified upon availability..
SSRState	43	1	ASCII	Short Sale Restriction values: ▪ '~' – No Short Sale in Effect ▪ 'E' – Short Sale Restriction in Effect Note: For NYSE Arca, only the value 'E' is applicable Note: This field is not yet implemented and is defaulted to binary 0. Clients will be notified upon availability.
MarketState	44	1	ASCII	This field is applicable to NYSE and NYSE MKT only Market State values: ▪ 'O' – Opened ▪ 'P' – Pre-Opening ▪ 'X' – Closed
SessionState	45	1	ASCII	This field is applicable to NYSE Arca and is for future use. Session State values: ▪ 'X' – Early Session State ▪ 'Y' – Core Session State ▪ 'Z' – Late Session State ▪ Note: This field is not yet implemented and is defaulted to a space character (0x20). Clients will be notified upon availability.

4.3 SEQUENCE NUMBER RESET MESSAGE (MSG TYPE 1)

This message is sent to reset the Message Sequence Number at start of day, or in response to failures.

This message always appears in its own dedicated packet with a Sequence Number of 1 (the new, reset number). The packet Delivery Flag is normally 12, as on system startup, but during failover events it is set to 10.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTIO
MsgSize	0	2	Binary	Size of the message: 14 Bytes
MsgType	2	2	Binary	The type of this message: 1 – Sequence Number Reset message
SourceTime	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
SourceTimeNS	8	4	Binary	The nanosecond offset from the SourceTime
ProductID	12	1	Binary	The unique ID for this NYSE feed listed in the feed's client specification.
ChannelID	13	1	Binary	The ID of the multicast channel over which the packet was sent.

4.4 SOURCE TIME REFERENCE MESSAGE (MSG TYPE 2)

For high-volume feeds such as XDP Integrated, this message is sent at the start of every second during periods of active data publication. Unlike some control messages, Source Time Reference messages can come in packets containing market data messages.

The client can concatenate the SourceTime field with the SourceTimeNS field in subsequent market data messages to get full 8-byte Matching Engine event timestamps. The contents of the ID field can be linked via the Symbol Index Mapping Message (Msg Type 3) to the applicable data messages.

See Date and Time Conventions for more information.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 16 bytes
MsgType	2	2	Binary	The type of message: 2 – Source Time Reference Message
ID	4	4	Binary	**For NYSE and MKT Integrated feeds:** ID of the originating Matching Engine partition to which this message applies. This usage will become standard across all products in future releases. **For all other feeds:** Symbol Index of the symbol to which this message applies.
SymbolSeqNum	8	4	Binary	**For NYSE and MKT Integrated feeds:** Reserved for future use. Ignore any content. This usage will become standard across all products in future releases. **For all other feeds:** The unique ID of this message in the sequence of messages published for this specific symbol.
SourceTime	12	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.

4.5 SYMBOL CLEAR MESSAGE (MSG TYPE 32)

In case of a failure and recovery of a Matching Engine or an XDP Publisher, the publisher may send a full state refresh for every symbol affected. This kind of unrequested refresh is preceded by a Symbol Clear message. The client should react to receipt of a Symbol Clear message by clearing all state information for the specified symbol in anticipation of receiving a full state refresh.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 20 Bytes
MsgType	2	2	Binary	The type of this message: 32 – Symbol Clear
SourceTime	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
SourceTimeNS	8	4	Binary	The nanosecond offset from the SourceTime
SymbolIndex	12	4	Binary	The unique ID of the symbol in the Symbol Index msg
NextSourceSeqNum	16	4	Binary	The sequence number in the next message for this symbol

4.6 TRADING SESSION CHANGE MESSAGE (MSG TYPE 33)

This message announces the start of a new trading session for a specified symbol.   It is only used in the Arca market.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 21 Bytes
MsgType	2	2	Binary	The type of this message: 33 – Trading Session Change
SourceTime	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
SourceTimeNS	8	4	Binary	The nanosecond offset from the SourceTime
SymbolIndex	12	4	Binary	The unique ID of the symbol in the Symbol Index msg
SymbolSeqNum	16	4	Binary	The unique ID of this message in the sequence of messages published for this specific symbol.
Trading Session	20	1	Binary	Valid values: ▪ 0x01 - Morning hours ▪ 0x02 - National hours (core) ▪ 0x04 - Late hours Only 0x02 is used by the NYSE and NYSE MKT markets.

5.   Messages Sent by Refresh and Retrans Servers Only

5.1 REFRESH HEADER (MSG TYPE 35)

The first message in each packet of refresh messages published over the Refresh multicast channels is of this type.

Valid values for the DeliveryFlag in the PacketHeader are:

- 17 – Only one packet in Refresh sequence
- 18 – Start of Refresh sequence
- 19 – Part of a Refresh sequence
- 20 – End of Refresh sequence
-

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 16 Bytes
MsgType	2	2	Binary	The type of this message: 35 – Refresh Header Message
CurrentRefreshPkt	4	2	Binary	The current refresh packet in the update
TotalRefreshPkts	6	2	Binary	The total number of refresh packets you should expect in the update
LastSeqNum	8	4	Binary	The last sequence number sent on the feed. The refresh is the state of the book as of this sequence number.
LastSymbolSeqNum	12	4	Binary	The last symbol sequence number sent on the feed. The refresh is the symbol state of the book as of this sequence number.

5.1.1   Shortened Refresh Header

The first message in the first packet for this symbol is a full 16-byte Refresh header message. Every packet for this symbol after that contains an 8byte refresh header. The LastSeqNum and the LastSymbolSeqNum fields are removed to avoid sending duplicate information in every packet.

After the first packet the refresh header looks as follows:

5.1.2   Refresh Example

Assuming the refresh of symbol XYZ requires three packets.

The first Packet structure will look as follows:

PACKET HEADER	REFRESH HEADER	MESSAGE 1	MESSAGE 2	...	MESSAGE N

For a depth of book feed such as XDP Integrated or XDP Depth of Book, the refresh header will be followed by the following message types:

1. Symbol Index Mapping Message (Msg Type 3)
2. Imbalance Message (Msg Type 105), if there is a current imbalance
3. Security Status Message (Msg Type 34)
4. Trade Session Change (Msg Type 33), indicates the current trading session (applies to Arca market only)
5. Add Order Refresh (Msg Type 106), repeated as needed to specify the book state for this symbol

5.2 MESSAGE UNAVAILABLE MESSAGE (MSG TYPE 31)

This message will be sent over the Retransmission multicast channels to inform clients of unavailability of a range of messages (or part of a range) for which they may have requested a retransmission.

The Message Unavailable message will be sent as the only message in a packet, and the Packet Header Delivery Flag will be set to 21.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 14 Bytes
MsgType	2	2	Binary	The type of this message: 31 – Message Unavailable
BeginSeqNum	4	4	Binary	The beginning sequence number of the unavailable range of messages.
EndSeqNum	8	4	Binary	The ending sequence number of the unavailable range of messages.
ProductID	12	1	Binary	The unique ID of the feed for which the retransmission was requested (listed in the feed's client specification).
ChannelID	13	1	Binary	The ID of the multicast channel for which the retransmission was requested.

6. Messages Sent by the Client to the Request Server

6.1 RETRANSMISSION REQUEST MESSAGE (MSG TYPE 10)

Clients who have experienced a sequence number gap and need a retransmission of the missed messages should send a Retransmission Request message via TCP to the Request Controller. A Request Response message will be sent over the TCP connection back to the client, and if the request was valid, the requested message(s) will be re-published over the relevant Retransmission multicast channel.

The retransmitted message(s) will have the same message format and content as the original messages that were missed.

Retransmission Requests should be sent in a packet whose Packet Header Delivery Flag is set to 11, and which contains a valid sequence number.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 24 Bytes
MsgType	2	2	Binary	The type of this message: 10 – Retransmission Request message
BeginSeqNum	4	4	Binary	The beginning sequence number of the range of messages to be retransmitted.
EndSeqNum	8	4	Binary	The end sequence number of the range of messages to be retransmitted.
SourceID	12	10	ASCII	The ID of the client requesting this retransmission . This field is up to 9 characters, null terminated.
ProductID	22	1	Binary	The unique ID of the feed for which a retransmission is requested (listed in the feed's client specification).
ChannelID	23	1	Binary	The ID of the multicast channel on which the gap occurred.

6.2 REFRESH REQUEST MESSAGE (MSG TYPE 15)

Clients who have experienced a failure and need a refresh of the state of one or all symbols in a specific channel should send a Retransmission Request message via TCP to the Request Controller. A Request Response message will be sent over the TCP connection back to the client, and if the request was valid, the requested message(s) will be published over the relevant Refresh multicast channel.

Retransmission Requests should be sent in a packet whose Packet Header Delivery Flag is set to 11, and which contains a valid sequence number.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 20 Bytes
MsgType	2	2	Binary	The type of this message: 15 – Refresh Request Message
SymbolIndex	4	4	Binary	The ID (from the Symbol Index msg) of the symbol for which a refresh is requested. To request a refresh for all symbols in the channel, set this field to 0.
SourceID	8	10	ASCII	The ID of the client requesting this refresh. This field is up to 9 characters, null terminated.
ProductID	18	1	Binary	The unique ID of the feed for which the refresh is requested (listed in the feed's client specification).
ChannelID	19	1	Binary	The ID of the multicast channel for which the refresh is requested.

6.3 SYMBOL INDEX MAPPING REQUEST MESSAGE (MSG TYPE 13)

This message is sent by clients via TCP/IP requesting the Symbol Index Mapping messages for one or all symbols in a specified channel.

The Symbol Index Mapping Request messages should be sent in a packet whose Packet Header Delivery Flag is set to 11, and which contains a valid sequence number.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 21 Bytes
MsgType	2	2	Binary	The type of this message: 13 – Symbol Index Mapping Request Message
SymbolIndex	4	4	Binary	The ID (from the Symbol Index msg) of the symbol for which a refresh is requested. To request a refresh for all symbols in the specified channel, set this field to 0.
SourceID	8	10	ASCII	The ID of the client requesting this symbol refresh. This field is up to 9 characters, null terminated.
ProductID	18	1	Binary	The unique ID of the feed for which the refresh is requested (listed in the feed's client specification).
ChannelID	19	1	Binary	The ID of the multicast channel for which the refresh is requested.
RetransmitMethod	20	1	Binary	The delivery method for the requested symbol index mapping information. Valid values: ▪ 0 – deliver via UDP

6.4 HEARTBEAT RESPONSE MESSAGE (MSG TYPE 12)

Clients who remain connected to the Retransmission Server intraday must respond to a Heartbeat with a Heartbeat Response message within 5 seconds. If no timely client response is received, the connection will be closed.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 14 Bytes
MsgType	2	2	Binary	The type of this message: 12 – Heartbeat Response message
SourceID	4	10	ASCII	The ID of the connected client . This field is up to 9 characters, null terminated.

7.   Messages Sent by Request Server

7.1 REQUEST RESPONSE MESSAGE (MSG TYPE 11)

This message will be sent immediately via TCP/IP in response to the client's request for retransmission, refresh or Symbol Mapping messages.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 29 Bytes
MsgType	2	2	Binary	The type of this message: 11 – Request Response Message
RequestSeqNum	4	4	Binary	The sequence number of the request message sent by the client. This can be used by the client to couple this response with the original request message.
BeginSeqNum	8	4	Binary	For Retrans Request responses, the beginning sequence number of the requested retransmission range. For responses to Refresh or Symbol Mapping Requests, 0.
EndSeqNum	12	4	Binary	For Retrans Request responses, the ending sequence number of the requested retransmission range. For responses to Refresh or Symbol Mapping Requests, 0.
SourceID	16	10	ASCII	The ID of the client making the request. This field is up to 9 characters, null terminated.
ProductID	26	1	Binary	The unique ID of the feed for which the request was made (listed in the feed's client specification).
ChannelID	27	1	Binary	The ID of the multicast channel for which the request was made.
Status	28	1	ASCII	The reason why the request was rejected. Valid values: ▪ '0' – Message was accepted ▪ '1' – Rejected due to an Invalid Source ID ▪ '2' – Rejected due to invalid sequence range ▪ '3' – Rejected due to maximum sequence range (see threshold limits) ▪ '4' – Rejected due to maximum request in a day ▪ '5' – Rejected due to maximum number of refresh requests in a day ▪ '6' – Rejected. Request message SeqNum TTL (Time to live) is too old. Use refresh to recover current state if necessary. ▪ '7' – Rejected due to an Invalid Channel ID ▪ '8' – Rejected due to an Invalid Product ID ▪ '9' – Rejected due to: 1) Invalid MsgType, or 2) Mismatch between MsgType and MsgSize

8.   Error Handling and the Request Server

8.1 HANDLING SEQUENCE NUMBER GAPS

Since multicast is an unreliable protocol, messages can be dropped. For this reason, clients are advised to process both lines in a channel. If a gap occurs on one line, the gap can be filled immediately from the other.

If a gap occurs on both lines simultaneously, the client can send a Retransmission Request Message (Msg Type 10) via TCP to the Request Server. The Retransmission Request contains a unique client ID called a Source ID, along with the Product and Channel IDs and the sequence number range of the missing messages.

On receipt of a Retransmission Request message, the Request Server will send back a Request Response Message (Msg Type 11). If any of the fields of the the Retransmission Request contained malformed or meaningless information, the request is rejected. If the request is accepted, the Retransmission Server will re-send the requested messages via multicast over the Retransmission channels.

If the request is rejected for exceeding a predefined system limit, the client will be prevented from making any further requests. See Retransmission/Refresh Request Limitations. If further requests are required, please contact NYSE.

8.1.1   Retransmission Format

Retransmitted messages have the same message format and content as the originally published messages (including the Sequence Numbers, but they may be packetized differently for best efficiency.

Packets of retransmitted messages have special Delivery Flag values in the Packet Header:

- 13 – Only one packet in retransmission sequence
- 15 – Part of a retransmission sequence

8.2 RECOVERING FROM CLIENT LATE STARTS OR INTRADAY FAILURES

If a client process experiences a late start or an intraday failure, the client will usually want to receive snapshots of the current market state for each symbol before resuming processing of real-time data. To do this, the client requests a refresh from the Refresh Server.

Specifically, the client should

1. Subscribe to the Refresh multicast channels
2. If not continuously connected to the Request Server already, make a TCP/IP connection
3. Send a Refresh Request Message (Msg Type 15) to the Request Server

The Refresh Request contains

1. A unique client ID called a Source ID
2. Product and Channel IDs
3. A Symbol Index, specifying a particular symbol to be refreshed or else if 0, specifying all symbols.

On receipt of a Refresh Request message, the Request Server will send back a Request Response Message (Msg Type 11). If any of the fields of the the Refresh Request contained malformed or meaningless information, the request is rejected. If the request is accepted, the Refresh Server will send the snapshot message(s) over the specific Refresh channel.

If the request is rejected for exceeding a predefined system limit, the client will be prevented from making any further requests. See Retransmission/Refresh Request Limitations. If further requests are required, please contact NYSE.

8.2.1   Refresh Format

Each refresh packet begins with a Packet Header, followed by a Refresh Header (Msg Type 35).

The Packet Header for a refresh packet has special Delivery Flag values:

- 17 – Only one packet in Refresh sequence
- 18 – Start of Refresh sequence
- 19 – Part of a Refresh sequence
- 20 – End of Refresh sequence

These Delivery Flag values are used as follows:

- **Refreshes of a Single Symbol**   The Delivery Flags of all refreshes of a single symbol that span multiple packets contain 'Part' indications only. You use the PktNum and NumPkts fields to know when the complete refresh has been received.

- **Refreshes of All Symbols**   Start, Part and End Delivery Flags are used for refreshes of all symbols. All packets of the first symbol are marked Start and all packets of the last symbol are marked End. This applies also to refreshes of all Symbol Index Mapping messages.

The Refresh Header identifies the position of the current packet is in this sequence of Refresh packets, along with the total number packets in this sequence. By use of the Delivery Flag and the packet sequence information in the Refresh Header, the client can know when the last packet of the refresh sequence has been received.

No dedicated retransmission service is available for the Refresh Server; if message loss is detected, clients should submit another refresh request.

8.3 REFRESHING SYMBOL INFORMATION

At system startup, each channel publishes a Symbol Index Mapping Message (Msg Type 3) for each symbol published on this channel.

If a client process misses the initial spin of symbol information for whatever reason, he may wish to receive a refresh of some or all Symbol Index Mapping messages before resuming processing of real-time data. To do this, the client should follow the procedure described in Recovering from Client Late Starts or Intraday Failures, but send a Symbol Index Mapping Request Message (Msg Type 13) to the Request Server instead of a Refresh Request Message.

8.3.1   Symbol Index Mapping Refresh Format

Requested Symbol Index Mapping messages are published by the Refresh Server with the same Packet Header Delivery Flags used for Refresh publications. Refresh Headers are not used in Symbol Index Mapping refreshes.

8.4 REQUEST SERVER

It is possible to connect to the Request Server only as needed, disconnecting after each request, but it is recommended that you remain connected to the Request Server for the entire trading day.

Once a client establishes a TCP/IP connection, the Request Server will send a heartbeat to the client approximately every 60 seconds. Clients must respond to with a Heartbeat Response message within 5 seconds, otherwise the Request Server will assume the client or the network has failed and close the connection.

8.4.1   Request Queuing

Clients may send several requests at the same time with the same Source ID. There is no need to wait for one request to be fulfilled before requesting another one.

Responses to all requests are published in the order in which they are received, although overlapping requests may be de-duplicated for efficiency.

8.4.2   Request Quotas

Due to resource constraints, a set of retransmission/refresh request limitations is enforced by the Request Server. If the client exceeds these intraday limits, he will be prevented from making any further requests. If further requests are required, the client can contact NYSE to have his quotas manually reset.

CAPABILITY	DESCRIPTION	THRESHOLD	ACTION
Prevention of invalid subscribers	Incoming requests from subscribers that are not in the enabled subscriber's Source ID list will not be honored. XDP subscribers will need a Source ID, which is a string that uniquely identifies the subscriber of the retransmission requests. Please contact the NYSE ARCA Service Desk to get a unique Source ID.	N/A	Request will not be processed.
Limitation of Retransmission Message Requests	Only retransmission requests for 1000 messages or less will be honored.	1000	Request will not be processed.
Limitation of Generic Request Age in Seq Nums	If the generic request on a message which is not within this threshold, the request will not be honored.	75000	Request will not be honored.
Limitation of Generic Requests	Generic requests for messages not within the threshold number of requests per day will not be honored during that particular day.	500	Subsequent retransmissions requests from that subscriber will be blocked.
Limitation of requests for refresh messages	Up to 5000 refresh requests will be honored.	x	Request will not be honored.
Limitation of Index Mapping Requests	Up to x Symbol Index Mapping requests will be honored.	x	Request will not be honored.

9.   Operational Information

9.1 SYSTEM BEHAVIOR ON START AND RESTART

At system startup or at start of system recovery following a failure, XDP feeds send the following messages over each channel:

- 10 Heartbeats with Delivery Flag set to 1, and sequence number set to 1 (next expected sequence number)
- Sequence Number Reset Message (Msg Type 1), the sequence number is 1 and the packet DeliveryFlag is 12
- A full spin of Symbol Index Mapping Messages (Msg Type 3), for securities published on this channel

9.2 XDP PUBLISHER FAILOVER

When failing over, the following refresh information is published.

Note: During the failover refresh, all Packet Header DeliveryFlag fields are set to 10.

1. A Sequence Number Reset Message (Msg Type 1) is sent in its own packet

2. For each symbol, the following are published,

   - Symbol Index Mapping Messages (Msg Type 3)
   - Symbol Clear Message (Msg Type 32)
   - The last Security Status Message (Msg Type 34)
   - The last Trading Session Change Message (Msg Type 33) (applies to the Arca market only)
   - All required refresh messages
   - The last Source Time Reference Message (Msg Type 2)

Once all symbols have been refreshed, Packet Header DeliveryFlag fields return to the normal 11.

9.3 DISASTER RECOVERY SITE

All XDP feeds are published out of the NYSE Mahwah data center. In case of catastrophic failure in Mahwah, all affected systems including XDP feeds will be coldstarted at the Cermak Disaster Recovery site in Chicago. The Cermak configuration of channels and multicast groups is identical to Mahwah for all feeds, but of course the source IPs are different.

9.4 XDP PRODUCTION HOURS

The following table describes the main daily event generating activity and indicative time on the XDP Integrated feed.

EVENT	NYSE/ NYSE MKT TIME (EST)	NYSE ARCA TIME (EST)	COMMENT
Sequence Number reset	~1:00am	~1:00am	
Symbol Mapping	~1:00am	~1:00am	
Pre-Open			
Open	9:30am	4:00am	Open time and close time are for the first financial products to open or close
Close	4:00pm	8:00pm	

9.5 XDP TEST HOURS

UAT testing hours are approximately normal hours approximately 1:00 AM - 8:15 PM.

10. Symbol Index Mapping File

For customers that would prefer to download the symbol index mapping from an FTP server, a file containing a subset of the index mapping information is made available via FTP at 12:30am EST every trading day at the following location.

- **NYSE**

Pipe-delimited: ftp://ftp.nyxdata.com/NYSESymbolMapping/NYSESymbolMapping_NMS.txt
XML format: ftp://ftp.nyxdata.com/NYSESymbolMapping/NYSESymbolMapping_NMS.XML

- **NYSE ARCA**

Pipe-delimited: ftp://ftp.nyxdata.com/ARCASymbolMapping/ARCASymbolMapping.txt
XML format: ftp://ftp.nyxdata.com/ARCASymbolMapping/ARCASymbolMapping.XML

Symbol Index Mapping File Format

FIELD NAME	FORMAT	DESCRIPTION
Symbol	ASCII	This field contains the full symbol in NYSE Symbology. A sequence of characters representing the symbol, padded with NULLs. For more information, see NYSE Symbology.
CQS Symbol	ASCII	This field contains the full symbol in CTS and UTP line format.
SymbolIndex	Numeric	The unique ID for this symbol. See Symbol Indexes. This field is unique for products within each respective market and cannot be used to cross reference a security between markets.
NYSE Market	Character	This field represents the exchange within NYSE trading the symbol: • 'N' – NYSE • 'P' – NYSE Arca • 'A' – NYSE MKT
Listed Market	Character	This field represents the exchange where the symbol is listed: • 'A' – NYSE MKT • 'N' – NYSE • 'P' – NYSE Arca • 'Q' – NASDAQ • 'T' – NASDAQ stock traded on NYSE Arca • 'U' – OTCBB symbol for Global OTC • 'V' – Other OTC symbols for Global OTC • 'Z' - BATS
TickerDesignation	Character	This represents on which ticker the symbol will be printed: • 'A' – CTA Tape A • 'B' – CTA Tape B • 'Q' – CTA Tape C
UOT	Numeric	This field represents the stock symbol's unit of trade.
PriceScaleCode	Numeric	This field contains the price scale for price conversion of the symbol. Prices in this feed are represented by two fields, separating the denominator and the numerator. All prices in the feed share a common denominator (unless otherwise specified), which is represented in the PriceScaleCode. The PriceScaleCode field value represents the common denominator for the following formula: $$Price = \frac{Numerator}{10^{PriceScaleCode}}$$

FIELD NAME	FORMAT	DESCRIPTION
		For example, a price of 27.56 is represented by a Numerator of 2756 and a PriceScaleCode equals to 2.
SystemID	Numeric	This represents the matching engine ID where the symbol is trading.
Bloomberg BSID	Alphanumeric	Unique Bloomberg identifier (BSID) including the Bloomberg source for all Bloomberg securities as an integer value. This field is used for B Pipe and subscription services.
Bloomberg Global ID	Alphanumeric	Unique global identifier assigned by Bloomberg for all securities across every asset class.