TEXAS STOCK EXCHANGE

SEED order entry

Contents

0.1 Revision History

Date	Version	Notes
July 10, 2025	0.1	Initial version. Known gaps: Risk Controls, Liquidity code and Symbology.
September 09, 2025	0.6.draft	Alpha release. Refined message layout, identifiers and enumerations in line with early internal feedback.
September 30, 2025	0.7.draft	Increased Presence Bits storage

1 Overview

The **SEED** protocol is the high-performance order entry interface designed to provide market participants with low-latency access to Texas Stock Exchange (TXSE) equity markets. SEED combines the best practices from proven exchange technologies with innovative enhancements specifically tailored to meet the evolving demands of modern electronic trading.

Session and Framing

SEED does not include a framing or session protocol. This function is performed by a higher-level protocol such as RAKE TCP.

1.1 Optional Fields / Presence bits

SEED uses presence bits to include only the fields actually needed in each message. Each bit indicates whether an optional field is present, minimizing bandwidth usage while maintaining message flexibility.

Message Structure

When optional fields are present, messages follow this layout:

```
[Message Type][Presence Bits][Fixed Fields][Optional Field 1][Optional Field 2]...[Optional Field N]
```

The presence bits appear immediately after the message type identifier and before the fixed fields. Optional fields are appended in bit order after all fixed fields.

Example: LimitOrder with MPID and MaxFloor

```
Message Type: 'L' (0x4c)
Presence Bits: 0x0005 (bits 0 and 2 set 0b0101)
Fixed Fields: [clOrdId][orderQty][bitFields][symbolId][price]
Optional Fields: [MPID (4 bytes)][MaxFloor (4 bytes)]
```

In this example, presence bits `0x0005` indicate that `MPID` (bit 0) and `maxFloor` (bit 2) are included, while all other optional fields are omitted. Optional fields are always in the same order as their corresponding bit positions.

1.2 Modify and Replace Behaviors

Requested Quantity

Modifies will not lose priority and so will only accept quantities which remain the same or decrease relative to the resting order. Replaces will always lose priority and so can accept quantities which remain the same, decrease or increase relative to the resting order.

Executed Quantity

When processing modify or replace requests, the Exchange accounts for executed quantity in the requested quantity to determine the order's leaves quantity.

If the requested quantity is less than or equal to the already executed quantity, the order's remaining leaves quantity is set to zero and the order closes automatically. No rejection or cancellation messages are generated in this scenario.

Example:

```
Original order: Buy 1,000 shares
Executed: 600 shares (400 remaining)
Modify request: Change order to 500 shares
Modify accepted: Order quantity = 600, Leaves quantity = 0
```

1.3 Pegged Orders

Pegged orders automatically price relative to the National Best Bid and Offer (NBBO) using the `referencePriceTarget` field, specified in basis points relative to the NBBO spread.

Pricing Calculation

$$P_{\text{buy}} = \text{NBB} + \left(\frac{\text{referencePriceTarget}}{10,000} \times (\text{NBO} - \text{NBB}) \right) \tag{1}$$

$$P_{\text{sell}} = \text{NBO} - \left(\frac{\text{referencePriceTarget}}{10,000} \times (\text{NBO} - \text{NBB}) \right) \tag{2}$$

Examples

BUY Order:

```
referencePriceTarget: 2,500 (25%)
NBB: $10.00, NBO: $11.00 (spread: $1.00)
Order Price: $10.00 + (2,500/10,000 × $1.00) = $10.25
```

SELL Order:

```
referencePriceTarget: 7,000 (70%)
NBB: $10.00, NBO: $11.00 (spread: $1.00)
Order Price: $11.00 - (7,000/10,000 × $1.00) = $10.30
```

Common Peg Types

Standard peg types can be implemented using specific `referencePriceTarget` values:

- **PRIMARY Peg**: `referencePriceTarget = 0` (0%) - Orders price at NBB (buy) or NBO (sell)
- **MID Peg**: `referencePriceTarget = 5,000` (50%) - Orders price at NBBO midpoint
- **MARKET Peg**: `referencePriceTarget = 10,000` (100%) - Orders price at NBO (buy) or NBB (sell)

Execution Restrictions

Pegged orders will not execute in locked, crossed, one-sided, or no-sided markets where normal NBBO pricing relationships do not exist.

Priority and Price Rounding

When different `referencePriceTarget` values calculate to the same final price due to rounding, arrival time determines priority.

Narrow Spread Example: Both orders price at $10.02, so if the 70% order arrived first, it has priority.

```
NBB: $10.00, NBO: $10.03, Spread: $0.03
70% BUY: $10.00 + (0.70 × $0.03) = $10.021 → $10.02
90% BUY: $10.00 + (0.90 × $0.03) = $10.027 → $10.02
```

Wide Spread Example: Different final prices result in standard price-time priority as 90% BUY has better price.

```
NBB: $10.00, NBO: $10.10, Spread: $0.10
70% BUY: $10.00 + (0.70 × $0.10) = $10.07
90% BUY: $10.00 + (0.90 × $0.10) = $10.09
```

Midpoint Order with a 50% `referencePriceTarget` will trade at midpoint of the NBBO, which may result in a sub-penny execution price for stocks over $1.00. Midpoint executions might also apply to other peg targets depending on the spread at the time of execution. For example, for a stock with an NBBO greater than $1.00 and a spread of $0.01, all `referencePriceTargets` greater than or equal to 50% and less than 100% are all effectively midpoint pegs, until the spread changes.

Sub $1.00 stocks The same peg math applied to stocks under $1.00, but valid peg prices are floored to an MPV of $0.0001

Peg Reference Price Target Rounding While the Member is free to enter any value for the `referencePriceTarget`, the Exchange may floor your peg target to a minimum variation for a given stock. For example, the Exchange may determine that for a specific stock, only increments of 10% are economically meaningful. If the Member enters a `referencePriceTarget` of 23%, the Exchange will floor it to 20%. The Member will see the result on the order acknowledgment. The Exchange reserves the right to change the minimum peg target increment at any time for any stock. Values of 0%, 50% and 100% will never be altered.

1.4 Display Price Slide Instructions

Display Price Slide Instructions automatically adjust order display prices to comply with Regulation NMS Rule 610(d), preventing locked or crossed markets. When an order would lock (match) or cross (trade through) the Reg NMS markets, the Exchange will reprice the displayed price while preserving the original limit price for execution.

Price Slide Behavior

When sliding occurs:

- **Internal ranking**: Order is ranked at the locking price for priority purposes
- **Market display**: Order is displayed one tick away
- **Original limit price**: Retained for execution purposes

Message Flow

Price sliding generates the following message sequence:

- `LimitOrderAccepted`: Acknowledges order with price field set to original limit price
- `OrderRestated`: Reports the ranked price with `reason = REPRICED` when sliding occurs

Examples

Locking Scenario:

```
Market: AAPL $150.00 x $150.01
Incoming order: Buy 1000 AAPL @ $150.01 (would lock the offer)
```

- `NO_PRICE_SLIDE`: Order canceled
- `SINGLE_PRICE_SLIDE_ON_LOCK_AND_CROSS`: Ranked at $150.01, Displayed at $150.00
- `MULTIPLE_PRICE_SLIDES_ON_LOCK_AND_CROSS`: Ranked at $150.01, Displayed at $150.00

Crossing Scenario:

```
Market: AAPL $150.00 x $150.01
Incoming order: Buy 1000 AAPL @ $150.02 (would cross the offer)
```

- `NO_PRICE_SLIDE`: Order canceled
- `SINGLE_PRICE_SLIDE_ON_LOCK_AND_CROSS`: Ranked at $150.01, Displayed at $150.00
- `MULTIPLE_PRICE_SLIDES_ON_LOCK_AND_CROSS`: Ranked at $150.01, Displayed at $150.00
- `SINGLE_PRICE_SLIDE_LOCK_ONLY`: Order canceled (crossing not allowed)
- `MULTIPLE_PRICE_SLIDES_LOCK_ONLY`: Order canceled (crossing not allowed)

Priority Impact

Each reprice creates a new timestamp and the order loses time priority at its new price level. Orders retain relative priority among other repriced orders based on original entry time.

1.5 Market Orders

Market orders provide a convenient way to execute immediate trades without specifying a price. Internally, they are handled as limit orders with a 100% peg to the reference price, ensuring aggressive pricing (maximum price for buy orders, minimum price for sell orders). Market orders are configured with no resting capability and will immediately cancel any unfilled portion after execution attempts, guaranteeing they never rest on the book.

1.6 Hidden Orders

Hidden orders do not have Price Slide Instructions; however, the Exchange will collar the limit price. If at order entry the order would cross the NBO (for BUY) or NBB (for SELL), the Exchange will adjust the order's limit price to the locking price. This adjustment happens only once at time of entry and the order is **NOT** repriced subsequently.

Example

```
Market: AAPL $150.00 x $150.01
Incoming order: Buy Hidden 1000 AAPL @ $150.10 (would cross the offer)
Order accepted: Buy Hidden 1000 AAPL @ $150.01 (price adjusted to lock)
```

NOTE: The `LimitOrderAccepted::price` field will reflect the adjusted limit price.

1.7 Reserve Orders

Reserve orders utilize a "Reserve Quantity" - the non-displayed portion of an order while a portion is displayed on the TXSE Book. Both the displayed portion and the Reserve Quantity are available for execution.

Basic Behavior

Reserve orders are created by setting `isHidden = true` and specifying a `maxFloorQty` that is less than the total `orderQty`. The Exchange displays a portion while keeping the remaining quantity (Reserve Quantity) hidden.

Replenishment occurs when the displayed portion is reduced to less than a Round Lot (typically 100 shares). Each replenishment creates a new OrderId while preserving the original ClOrdId.

Replenishment Types

- **Fixed Replenishment**: Default
  - Quantity: Replenishes to maxFloorQty
  - Time: Immediate replenishment
- **Random Quantity Replenishment**: Replenishment quantities are randomly selected within a range in Round Lot increments only
  - Range: (`maxFloorQty` - `maxReplenishQtyRange`) to (`maxFloorQty` + `maxReplenishQtyRange`)
  - Example: MaxFloor = 500, Range = 200 → Possible quantities: [300, 400, 500, 600, 700] shares
- **Random Time Replenishment**: Replenishment occurs after a random delay
  - Range: 0 to `maxReplenishTimeRange`
  - Example: MaxReplenishTimeRange=500us → Random delay between 0-500us after execution

Note: Random Quantity and Random Time can be configured together on the same order.

Active Replenishments

At any point in time, a reserve order can have:

- 0 replenishments: During time delay periods (Random Time configuration) or transient system states due to non-atomic replenishment processing
- 1 replenishment: Normal use case with active displayed quantity
- 2 replenishments: When the first replenishment falls below Round Lot size, triggering a second replenishment while the first may still have displayed leaves quantity

Order Restated Messages

Members receive OrderRestated messages for each replenishment:

```
messageType: 'F' (OrderRestated)
orderId: [New replenishment OrderId]
reason: RESERVE_REPLENISHED
quantity: [New displayed quantity] (if present)
```

Key Technical Details

- Each replenishment creates a new OrderId but preserves the original ClOrdId
- Exchange obfuscates OrderIds on data feeds for replenishment orders
- Reserve Quantity remains fully executable during replenishment delays
- Random replenishment quantities must be in Round Lot increments

- If remainder is less than replenishment amount, entire remainder is displayed
- Order management operations use the original ClOrdId

1.8 Post-Only Orders

Post-Only orders are designed to add liquidity to the order book without removing existing liquidity. These orders will only post to the book as resting orders and will never execute immediately against existing orders.

Behavior

When a Post-Only order is submitted:

- If marketable without Display Price Slide Instructions: Order is canceled (would immediately execute against existing orders)
- If marketable with Display Price Slide Instructions: Order may be repriced to avoid immediate execution, depending on the slide instruction
- If not marketable: Order posts to the book at the specified limit price

Key Characteristics

- Liquidity provision only: Guarantees the order will either rest on the book or be canceled
- No immediate execution: Eliminates the risk of unexpected immediate fills
- Works with Display Price Slide Instructions: Can be combined with price slide instructions to automatically adjust marketable orders instead of canceling them

1.9 Self Match Prevention

TXSE's Self Match Prevention functionality prevents orders from the same trading entity from executing against each other, supporting regulatory compliance and risk management requirements consistent with TXSE Rule 11.007(d). The system operates across configurable scopes including by Member (`BY_MEMBER`), by Market Participant Identifier (`BY_MPID`), by Member Group (`BY_MEMBER_GROUP`), or a combination of MPID and Member Group (`BY_MPID_AND_MEMBER_GROUP`).

Self-match prevention settings may be configured at the port level as default parameters for all orders submitted through that connection. These port-level defaults can be overridden on a per-order basis through the optional `selfMatchScope` and `selfMatchInstruction` fields, providing members with both operational efficiency and granular control over their order interaction policies.

When a potential self-match is detected between orders, the Exchange applies the specified prevention instruction: cancel the incoming order (`CANCEL_NEWEST`), cancel the resting order (`CANCEL_OLDEST`), cancel both orders entirely (`CANCEL_BOTH`), cancel only the smaller order if sizes differ or both if equal (`CANCEL_SMALLEST`), or cancel the overlapping portion while preserving the remainder of the larger order (`DECREMENT_AND_CANCEL`). Note: Only the aggressive order's self-match prevention instructions are considered during a match. The resting order's instructions are ignored.

Orders canceled due to self-trade prevention will generate an `OrderCanceled` message with reason `SELF_MATCH_PREVENTION`. This hierarchical configuration approach enables members to establish consistent firm-wide policies while maintaining the flexibility to customize behavior for specific trading strategies or order flows, ensuring compliance with Exchange Rules while supporting diverse trading requirements.

1.10 Mass Cancel Behavior

Cancel a large group of orders based on the Member Owned `scope`.

Member Owned Scope

- `BY_MEMBER_OWNED_SENDER_COMPS`: Scope mass cancels to all the ports owned by the Member
- `BY_MEMBER_OWNED_MPIDS`: Scope mass cancels to all the MPIDs owned by the Member

Optional Filter Fields

The optional filter fields are logically `AND`ed together. Each additional field narrows the number of orders affected.

The `ClOrdId` must be paired with `SenderComp`, and will cancel the single specific order from the SEED port associated with the `SenderComp`.

Canceling of orders is done on a best-effort basis. It is not atomic. The OrderCanceled messages will flow back to the SEED ports that entered the orders, and any drop copy ports that are following the orders. They will not flow back to the SEED port that sent the `MassCancel` message, except where orders entered by the same port are affected.

Examples

Mass Cancel on all the Member Owned Ports

Mass cancel all orders entered on ports owned by the Member, but **NOT** orders entered on behalf of the Member via service bureau owned ports

```
Message Type: 'w' (0x77)
scope: BY_MEMBER_OWNED_SENDER_COMPS
```

Mass Cancel on all the Member Owned MPIDs

Mass cancel all orders associated with the Member, across all member owned ports **AND** service bureau owned ports (on-behalf-of orders)

```
Message Type: 'w' (0x77)
scope: BY_MEMBER_OWNED_MPIDS
```

Mass Cancel a specific MPID by a Member

Mass cancel all orders associated with the MPID, across all member owned ports **AND** service-bureau owned ports (on-behalf-of orders)

```
Message Type: 'w' (0x77)
scope: BY_MEMBER_OWNED_MPIDS
mpid: ABCD
```

Mass Cancel a specific MPID by an on-behalf-of service bureau

Mass cancel all orders associated with the MPID, across **ONLY** the service bureau owned ports (on-behalf-of orders)

```
Message Type: 'w' (0x77)
scope: BY_MEMBER_OWNED_SENDER_COMPS
mpid: ABCD
```

1.11 Crossed Market Conditions

When a Protected Bid crosses a Protected Offer, the Exchange restricts executions to prevent trades at unreasonable prices, per Regulation NMS Rule 610(e).

Execution Price Restrictions During Crossed Markets

Per Rule 11.009(a)(2), during crossed markets the Exchange will not execute:

- Buy orders at prices more than 0.5% (or $0.05, whichever is greater) above the lowest Protected Offer
- Sell orders at prices more than 0.5% (or $0.05, whichever is greater) below the highest Protected Bid

Cancel on Crossed Market Instruction

Users may include a `cancelAtEntryIfCrossed` instruction on orders. When set, the Exchange will immediately cancel the incoming order if a Protected Bid is crossing a Protected Offer at time of entry, rather than attempting execution within the price restrictions.

Order Handling During Crossed Markets

- Limit Orders will follow Display Price Slide instructions; refer to documentation for details
- Market Orders will not execute when a locked or crossed market exists
- Pegged Orders will not execute when a locked or crossed market exists

Examples

Crossed Market - Buy Order Execution Cap:

```
NBBO: $10.05 x $10.03 (crossed)
Max Buy Execution Price: $10.03 + max($0.05, $10.03 × 0.005) = $10.08
Incoming: Buy 1000 @ $10.10
Result: Can execute up to $10.08; orders/liquidity above $10.08 unavailable
```

Crossed Market - With Cancel Instruction:

```
NBBO: $10.05 x $10.03 (crossed)
Incoming: Buy 1000 @ $10.10 with cancelAtEntryIfCrossed
Result: Order immediately canceled (no execution attempt)
```

High-Priced Stock Example:

```
NBBO: $1,005 x $1,000 (crossed)
Max Buy Price: $1,000 + max($0.05, $5.00) = $1,005
Incoming: Buy 100 @ $1,010 with cancelAtEntryIfCrossed =false
Result: Can execute up to $1,005; cannot access liquidity above $1,005
```

ISO Exception

Orders marked as Intermarket Sweep Orders (ISO) may execute beyond these price restrictions, as the sender assumes responsibility for simultaneously routing orders to protected quotations.

2 Types

- This document uses the terms `Byte, Short, Int, Long` to refer to numeric values that are 8,16,32,64 bit long respectively. All are two's complement signed little-endian encoded.

Type	Byte Length
`Byte`	1
`Short`	2
`Int`	4
`Long`	8

- Strings are all `ASCII`, fixed-length. Strings shorter than the fixed-length should be left justified, by right padded with spaces (`ASCII` 0x20) to the full string length
- Enums are encoded as a single `Byte` unless they appear in a bitfield, then they use the number of bits specified.
- TimeStamps are nanoseconds since the Unix epoch in UTC time zone, encapsulated by a `Long`
- Prices are a `Long` with an implied scale of 100,000,000. Each unit is $0.00000001
- Durations are represented as nanoseconds using a `Long` value

2.1 Side

Determines direction of a trading action, including short-sell indicators

Enum Constant	value	Notes
BUY	0	Order is a buy order
LONG_SELL	1	Sell order of existing inventory
SHORT_SELL	2	Sell order of borrowed shares
SHORT_EXEMPT	3	Sell order of borrowed shares exempt from certain regulations of Regulation SHO

2.2 TimeInForce

Determines time when an order goes live and when it will be canceled. The Exchange has three trading sessions: Early Session, Regular Session and After Hours Session. The Regular Session is the "normal" trading day, typically between 9:30AM EST and 16:00PM EST. The Early and After Hours Sessions are bounded by the Regular Session and the Exchange's start and end times.

Enum Constant	value	Notes
SYS	1	Accepted during any session. Live immediately upon entry. Canceled at the end of the After Hours Session
IOC	2	Similar to `SYS`, but canceled immediately after trading with resting liquidity on the Exchange
GTT	3	Similar to `SYS`, but canceled after the requested time has passed or the end of the After Hours Session, whichever comes first. See expireTime field in LimitOrder
DAY	4	Accepted during Early and Regular Session. Order becomes live immediately upon entry and is canceled at the end of the Regular Session. Will be rejected during the After Hours Session
RHO	5	Regular Hours Only. If entered before the Regular Session, the order will be queued until the Regular Session begins. If entered during the Regular Session, the order is live immediately. If entered during the After Hours Session the order will be rejected. All RHO orders are canceled at the end of the Regular Session

2.3 OrderCapacity

Broker capacity in sending order

Enum Constant	value	Notes
AGENCY	1	Acting on behalf of clients
PRINCIPAL	2	Acting on behalf of Broker-Dealer
RISKLESS_PRINCIPAL	3	Order to offset client order

2.4 SelfMatchScope

Determines the scope for self match prevention. For additional information, refer to Self Match Prevention documentation

Enum Constant	value	Notes
BY_MEMBER	0	
BY_MPID	1	
BY_MEMBER_GROUP	2	
BY_MPID_AND_MEMBER_GROUP	3	

2.5 SelfMatchInstruction

Determines the behavior when orders from the same entity would match. For additional information, refer to Self Match Prevention documentation

Enum Constant	value	Notes
NO_SELF_MATCH_PREVENTION	0	
CANCEL_NEWEST	1	Cancels the incoming order. Keeps the resting order on the book
CANCEL_OLDEST	2	Cancels the resting order. Keeps the incoming order on the book
CANCEL_BOTH	3	Cancels both orders entirely regardless of size
CANCEL_SMALLEST	4	If orders are equal size: cancels both. If different sizes: cancels the smaller order, keeps the larger one
DECREMENT_AND_CANCEL	5	If orders are equal size: cancels both. If different sizes: cancels the overlapping portion, keeps the remainder of the larger order

2.6 PriceSlideInstruction

For additional information, refer to Display Price Slide Instructions documentation

Enum Constant	value	Notes
NO_PRICE_SLIDE	0	No Display-Price Slide instruction. Orders that would create locking or crossing quotations will be canceled
SINGLE_PRICE_SLIDE_ON_LOCK_AND_CROSS	1	Single price slide applied to both locking and crossing quotations. Adjusts order price only upon entry and one additional time following an NBBO change
MULTIPLE_PRICE_SLIDES_ON_LOCK_AND_CROSS	2	Multiple price slides applied to both locking and crossing quotations. Continuously re-ranks and re-displays orders as NBBO changes throughout the order's life
SINGLE_PRICE_SLIDE_LOCK_ONLY	3	Single price slide applied only to locking quotations. Crossing quotations will be canceled. Note: Given this only serves locked quotations, a multiple price slide option could only slide once, thus redundant

2.7 MassCancelScope

Determines the Member-Owned scope for mass cancels. For additional information, refer to Mass Cancel Behavior documentation

Enum Constant	value	Notes
BY_MEMBER_OWNED_SENDER_COMPS	0	
BY_MEMBER_OWNED_MPIDS	1	

2.8 RejectReason

Indicates the reason that a customer action was rejected by the exchange

Enum Constant	value	Notes
INVALID_CLIENT_ORDER_ID	1	Client Order Id is invalid
DUPLICATE_CLIENT_ORDER_ID	2	Client Order Id has already been used for the current session on this port
UNKNOWN_ORIGINAL_CLIENT_ORDER_ID	3	Attempt to modify or replace a client order that was not found for the current session on this port
NO_LONGER_ON_BOOK	4	Attempt to modify, replace, or cancel an order that has already canceled or fully executed
INVALID_SYMBOL	5	Symbol ID is not valid for the current session
INVALID_PRICE	6	Invalid limit price
INVALID_ORDER_QUANTITY	7	Invalid order quantity
INVALID_REFERENCE_PRICE_TARGET	8	Invalid reference price target related to the order attributes
INVALID_IS_HIDDEN_FLAG	9	Invalid 'Is Hidden' flag related to the order attributes
INVALID_ORDER_TYPE	10	Invalid order type related to the order attributes
INVALID_SIDE	11	Invalid side
INVALID_MAX_FLOOR_QUANTITY	12	Invalid max floor quantity related to the order attributes
INVALID_MAX_REPLENISH_QUANTITY_RANGE	13	Invalid maximum replenish quantity range
INVALID_MAX_REPLENISH_TIME_RANGE	14	Invalid maximum replenish time range
INVALID_MINIMUM_QUANTITY	15	Invalid minimum quantity
INVALID_LOCATE_REQUIRED_FLAG	16	Invalid locate-required flag related to the order attributes
INVALID_TIME_IN_FORCE	17	Invalid time in force
UNSUPPORTED_MODIFICATION	18	Invalid modification related to the order attributes
INVALID_MPID	19	Invalid market participant identifier
INVALID_SENDER_COMP	20	Invalid sender comp. Must correspond to a configured order entry gateway

2.9 CancelReason

Reason that an order was canceled

Enum Constant	value	Notes
REQUESTED_BY_USER	1	Cancel due to request by client
RELATED_TO_TIME_IN_FORCE	2	Cancel due to time in force expiration
RELATED_TO_MIN_QTY	3	Cancel due to inability to meet minimum quantity
NMS_VIOLATION_NO_SLIDE	4	Canceled as order would violate Regulation NMS Rule 610(d) by creating a locked or crossed market, and price sliding was either not selected or not applicable
MARKETABLE_RESERVE	5	Reserve orders will be canceled when deemed marketable at another exchange either at new order entry or replenishment
SELF_MATCH_PREVENTION	6	Cancel due to self match prevention instructions
REPLENISHMENT_CANCELED_DUE_TO_RESERVE	7	Replenishment canceled related to the Reserve order being canceled or replaced
RELATED_TO_ORDER_TYPE	8	Cancel due to order type e.g. Market Orders do not rest
CANCELED_DUE_TO_CROSSED_MARKETS	9	Cancel due to client instruction on crossed markets
CANCELED_DUE_TO_MASS_CANCEL_REQUEST	10	Cancel due to a mass cancel request

2.10 RestatementReason

Enum Constant	value	Notes
RESERVE_REPLENISHED	1	A new displayed slice of a reserve order has been added to the displayed book
REPRICED	2	A reprice eligible order has been repriced to a new price

2.11 SymbolTradingState

Symbol-specific trading state indicating current availability

Enum Constant	value	Notes
TRADING	1	Symbol is open and available for trading
HALTED	2	Trading suspended pending news or regulatory action

2.12 ShortSaleRestrictionState

Short sale restriction state under SEC Rule 201

Enum Constant	value	Notes
NONE	0	No short sale restriction in effect (CTA: Space/D, UTP: 0)
ACTIVATED	1	Restriction activated intra-day (CTA: A, UTP: 1)
CONTINUED	2	Restriction carried over from previous trading day or in effect (CTA: C/E, UTP: 2)

2.13 OperationalHaltReason

Operational and technical reasons for trading halts initiated by exchange operations

Enum Constant	value	Notes
ADMINISTRATIVE	0	Administrative or technical halt initiated by exchange

2.14 RegulatoryHaltReason

Regulatory reasons for trading halts and pauses

Enum Constant	value	Notes
REASON_NOT_AVAILABLE	0	Reason not available or not applicable (CTA: Space, UTP: Space)
NEWS_PENDING	1	Trading halted pending release of material news (CTA: P, UTP: T1)
NEWS_DISSEMINATION	2	Trading halted for dissemination of material news (CTA: D, UTP: T2)
INFO_REQUESTED	3	Additional information requested by exchange (UTP: T12)
ORDER_IMBALANCE	4	Order imbalance (CTA: I)
EXTRAORDINARY_MARKET_ACTIVITY	5	Extraordinary market activity (UTP: T6)
LULD_PAUSE	6	Limit Up-Limit Down pause (CTA: M, UTP: LUDP/T5)
NON_COMPLIANCE	7	Non-compliance with listing standards (UTP: H4)
FILINGS_NOT_CURRENT	8	Required filings not current (UTP: H9)
SEC_SUSPENSION	9	SEC trading suspension (UTP: H10)
REGULATORY_CONCERN	10	Regulatory concern from other markets (UTP: H11)
SUB_PENNY_TRADING	11	Sub-penny trading violation (CTA: Y)
ETF	12	ETF-specific halt (UTP: T8)
IPO_NOT_TRADING	13	IPO not yet trading (UTP: IPO1)
CORPORATE_ACTION	14	Corporate action in progress (UTP: M1)
CIRCUIT_BREAKER_L1	15	Market-wide circuit breaker Level 1 (CTA: 1, UTP: MWC1)
CIRCUIT_BREAKER_L2	16	Market-wide circuit breaker Level 2 (CTA: 2, UTP: MWC2)
CIRCUIT_BREAKER_L3	17	Market-wide circuit breaker Level 3 (CTA: 3, UTP: MWC3)

2.15 MarketHoursState

Current phase of the market trading day

Enum Constant	value	Notes
CLOSED_BEFORE_HOURS	0	Markets are closed before trading hours
EARLY_SESSION	1	Early trading session is active (pre-market)
REGULAR_SESSION	2	Regular trading session is active
AFTER_HOURS_SESSION	3	After-hours trading session is active (post-market)
CLOSED_AFTER_HOURS	4	Markets are closed after trading hours

2.16 SessionTradingState

Market-wide trading state indicating the default state for all symbols (individual symbols may have additional restrictions)

Enum Constant	value	Notes
CLOSED	0	Market session is closed - outside of trading hours or not yet opened
TRADING	1	Market session is open and trading is active (individual symbols may still be halted/paused)
HALTED	2	Market-wide trading halt - all symbols halted (circuit breaker or system-wide halt)

3 Messages from Member to Exchange

Messages sent from the Member to the Exchange

3.1 LimitOrder

Field Name	Offset	Size	Type	Notes
messageType	0	1	`Byte`	'L'/0x4c/76
presenceBits	1	4	`Int`	See Optional Fields
clOrdId	5	8	`Long`	Client-specified order identifier
orderQty	13	4	`Int`	
limitOrderBitFields	17	4		
symbolId	21	2	`Short`	
price	23	8	`Long`	

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	true if a locate is required for short-sale, false otherwise
timeInForce	4	4	Enum TimeInForce.
orderCapacity	8	3	Enum OrderCapacity.
isIso	11	1	true for Reg NMS Intermarket sweep orders, false otherwise
isHidden	12	1	true if any of the order quantity is hidden, false for order quantity to be displayed
isPostOnly	13	1	true to only add the order to the book if it would be added as a resting order with no quantity executed
cancelAtEntryIfCrossed	14	1	true to cancel the order at entry if market conditions are crossed
reserved	15	17	Do not use – reserved for the future

3.1.1 LimitOrder Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasSelfMatchScope	0x00000001
hasSelfMatchInstruction	0x00000002
hasPriceSlideInstruction	0x00000004
hasMinQty	0x00000008
hasMaxFloorQty	0x00000010
hasMaxReplenishQtyRange	0x00000020
hasMaxReplenishTimeRange	0x00000040
hasReferencePriceTarget	0x00000080
hasExpireTime	0x00000100
hasUserData	0x00000200
hasMpid	0x00000400
hasMemberGroup	0x00000800
hasLocateBroker	0x00001000
reserved	0x7FFFE000

3.1.2 LimitOrder Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
selfMatchScope	1	Enum	Enum SelfMatchScope. Determines the scope for self match prevention. For additional information, refer to Self Match Prevention documentation
selfMatchInstruction	1	Enum	Enum SelfMatchInstruction. Determines the behavior when orders from the same entity would match. For additional information, refer to Self Match Prevention documentation
priceSlideInstruction	1	Enum	Enum PriceSlideInstruction. For additional information, refer to Display Price Slide Instructions documentation
minQty	4	Int	Number of shares that must be executed to execute any shares
maxFloorQty	4	Int	Quantity to be displayed at one time with remaining quantity not displayed on the book
maxReplenishQtyRange	4	Int	Defines the maximum range to be used when calculating a random refresh quantity
maxReplenishTimeRange	8	Duration	Defines the maximum range to be used when calculating a random refresh time, in nanoseconds
referencePriceTarget	2	Short	The target relative to the reference base price in basis points of the NBBO spread
expireTime	8	TimeStamp	Timestamp at which the order's open quantity will be canceled, valid only with GTT time in force
userData	8	Long	Pass-through field for use by clients, not used by the exchange
mpid	4	Str(4)	Market Participant Identifier. MPID should contain upper-case alpha only
memberGroup	2	Str(2)	Member supplied grouping. Used for both self-match prevention and mass cancel, in conjunction with MPID or SenderComp
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only

3.2 MarketOrder

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'A'/0x41/65
presenceBits	1	2	Short	See Optional Fields
clOrdId	3	8	Long	Client-specified order identifier
orderQty	11	4	Int	
marketOrderBitFields	15	2		
symbolId	17	2	Short	

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	true if a locate is required for short-sale, false otherwise
timeInForce	4	4	Enum TimeInForce.
orderCapacity	8	3	Enum OrderCapacity.
reserved	11	5	Do not use – reserved for the future

3.2.1 MarketOrder Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasSelfMatchScope	0x0001
hasSelfMatchInstruction	0x0002
hasUserData	0x0004
hasMpid	0x0008
hasMemberGroup	0x0010
hasLocateBroker	0x0020
reserved	0xFFC0

3.2.2 MarketOrder Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
selfMatchScope	1	Enum	Enum SelfMatchScope. Determines the scope for self match prevention. For additional information, refer to Self Match Prevention documentation
selfMatchInstruction	1	Enum	Enum SelfMatchInstruction. Determines the behavior when orders from the same entity would match. For additional information, refer to Self Match Prevention documentation
userData	8	Long	Pass-through field for use by clients, not used by the exchange
mpid	4	Str(4)	Market Participant Identifier. MPID should contain upper-case alpha only
memberGroup	2	Str(2)	Member supplied grouping. Used for both self-match prevention and mass cancel, in conjunction with MPID or SenderComp
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only

3.3 CancelOrder

Cancel all outstanding shares of an order

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'C'/0x43/67
origClOrdId	1	8	Long	The ClOrdId of the order to be canceled

3.4 ModifyOrder

Request to modify an existing order without losing book priority

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'M'/0x4d/77
presenceBits	1	1	Byte	See Optional Fields
clOrdId	2	8	Long	The ClOrdId for this modify message. Must be day-unique
origClOrdId	10	8	Long	The ClOrdId of the order to be modified

3.4.1 ModifyOrder Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasOrderQty	0x01
hasModifyBitFields	0x02
hasLocateBroker	0x04
reserved	0xF8

3.4.2 ModifyOrder Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
orderQty	4	Int	Quantity can remain the same or decrease. Quantity cannot increase. For additional information, refer to Modify Behavior documentation
modifyBitFields	1		
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	Processed for Sell Short and Sell Short Exempt in which client affirms ability to borrow (isLocateRequired = false) or client does not affirm ability to borrow (isLocateRequired = true)
reserved	4	4	reserved for future use

3.5 ReplaceOrder

Replace an existing order with a new order, always losing priority

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'R'/0x52/82
presenceBits	1	2	Short	See Optional Fields
clOrdId	3	8	Long	The ClOrdId for this modify message. Must be day-unique
origClOrdId	11	8	Long	The ClOrdId of the order to be replaced
replaceBitFields	19	1		

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	Processed for Sell Short and Sell Short Exempt in which client affirms ability to borrow (isLocateRequired = false) or client does not affirm ability to borrow (isLocateRequired = true)
isIso	4	1	true for Reg NMS Inter-market sweep orders, false otherwise
cancelAtEntryIfCrossed	5	1	true to cancel the order at entry if market conditions are crossed
reserved	6	2	reserved for future use

3.5.1 ReplaceOrder Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasPrice	0x0001
hasOrderQty	0x0002
hasMaxFloorQty	0x0004
hasLocateBroker	0x0008
reserved	0xFFF0

3.5.2 ReplaceOrder Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
price	8	Long	
orderQty	4	Int	Quantity can remain the same, decrease or increase. For additional information, refer to Replace Behavior documentation
maxFloorQty	4	Int	Quantity to be displayed at one time with remaining quantity not displayed on the book
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only

3.6 MassCancel

Mass cancel orders

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'V'/0x56/86
presenceBits	1	1	Byte	See Optional Fields
massCancelRequestId	2	8	Long	The MassCancelId assigned by the Member
scope	10	1	Enum	Enum MassCancelScope. For additional information, refer to Mass Cancel Behavior documentation

3.6.1 MassCancel Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasMpid	0x01
hasSenderComp	0x02
hasMemberGroup	0x04
hasClOrdId	0x08
reserved	0xF0

3.6.2 MassCancel Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
mpid	4	Str(4)	Market Participant Identifier. MPID should contain upper-case alpha only
senderComp	8	Str(8)	
memberGroup	2	Str(2)	Member supplied grouping. Used for both self-match prevention and mass cancel, in conjunction with MPID or SenderComp
clOrdId	8	Long	Client-specified order identifier

4 Messages from Exchange to Member

Messages sent from the Exchange to the Member

4.1 TradingSessionStatus

Market-wide status representing the overall state of the trading session

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'i'/0x69/105
presenceBits	1	1	Byte	See Optional Fields
transactTime	2	8	TimeStamp	
marketHoursState	10	1	Enum	Enum MarketHoursState. Current phase of the market trading day
tradingState	11	1	Enum	Enum SessionTradingState. Market-wide trading state indicating the default state for all symbols (individual symbols may have additional restrictions)

4.1.1 TradingSessionStatus Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasOperationalHaltReason	0x01
hasRegulatoryHaltReason	0x02
reserved	0xFC

4.1.2 TradingSessionStatus Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
operationalHaltReason	1	Enum	Enum OperationalHaltReason. Operational and technical reasons for trading halts initiated by exchange operations
regulatoryHaltReason	1	Enum	Enum RegulatoryHaltReason. Regulatory reasons for trading halts and pauses

4.2 DefineSymbol

Symbol definition containing instrument details and trading parameters for a security

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	's'/0x73/115
transactTime	1	8	TimeStamp	
symbolId	9	2	Short	
symbol	11	8	Str(8)	Security identifier root represented in CMS format
suffix	19	8	Str(8)	Security identifier suffix represented in CMS format
matchingEngineId	27	1	Byte	The matching engine to which this symbol is assigned for the trading session
defineSymbolBitFields	28	1		
lotSize	29	4	Int	

Bit Field Name	Offset	Size	Notes
isTest	0	1	Test symbol flag
reserved	1	7	Reserved for future use

4.3 SymbolStatus

Symbol-specific trading status representing the current state and restrictions for an individual security

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'y'/0x79/121
presenceBits	1	1	Byte	See Optional Fields
transactTime	2	8	TimeStamp	
symbolId	10	2	Short	
tradingState	12	1	Enum	Enum SymbolTradingState. Symbol-specific trading state indicating current availability
shortSaleRestrictionState	13	1	Enum	Enum ShortSaleRestrictionState. Short sale restriction state under SEC Rule 201

4.3.1 SymbolStatus Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasOperationalHaltReason	0x01
hasRegulatoryHaltReason	0x02
reserved	0xFC

4.3.2 SymbolStatus Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
operationalHaltReason	1	Enum	Enum OperationalHaltReason. Operational and technical reasons for trading halts initiated by exchange operations
regulatoryHaltReason	1	Enum	Enum RegulatoryHaltReason. Regulatory reasons for trading halts and pauses

4.4 LimitOrderAccepted

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'I'/0x49/73
presenceBits	1	4	Int	See Optional Fields
transactTime	5	8	TimeStamp	
orderId	13	8	Long	
clOrdId	21	8	Long	Client-specified order identifier
orderQty	29	4	Int	
limitOrderBitFields	33	4		
symbolId	37	2	Short	
price	39	8	Long	

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	true if a locate is required for short-sale, false otherwise
timeInForce	4	4	Enum TimeInForce.
orderCapacity	8	3	Enum OrderCapacity.
isIso	11	1	true for Reg NMS Intermarket sweep orders, false otherwise
isHidden	12	1	true if any of the order quantity is hidden, false for order quantity to be displayed
isPostOnly	13	1	true to only add the order to the book if it would be added as a resting order with no quantity executed
cancelAtEntryIfCrossed	14	1	true to cancel the order at entry if market conditions are crossed
reserved	15	17	Do not use – reserved for the future

4.4.1 LimitOrderAccepted Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasSelfMatchScope	0x00000001
hasSelfMatchInstruction	0x00000002
hasPriceSlideInstruction	0x00000004
hasMinQty	0x00000008
hasMaxFloorQty	0x00000010
hasMaxReplenishQtyRange	0x00000020
hasMaxReplenishTimeRange	0x00000040
hasReferencePriceTarget	0x00000080
hasExpireTime	0x00000100
hasUserData	0x00000200
hasMpid	0x00000400
hasMemberGroup	0x00000800
hasLocateBroker	0x00001000
hasRankPrice	0x00002000
hasDisplayPrice	0x00004000
reserved	0x7FFF8000

4.4.2 LimitOrderAccepted Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
selfMatchScope	1	Enum	Enum SelfMatchScope. Determines the scope for self match prevention. For additional information, refer to Self Match Prevention documentation
selfMatchInstruction	1	Enum	Enum SelfMatchInstruction. Determines the behavior when orders from the same entity would match. For additional information, refer to Self Match Prevention documentation
priceSlideInstruction	1	Enum	Enum PriceSlideInstruction. For additional information, refer to Display Price Slide Instructions documentation
minQty	4	Int	Number of shares that must be executed to execute any shares
maxFloorQty	4	Int	Quantity to be displayed at one time with remaining quantity not displayed on the book
maxReplenishQtyRange	4	Int	Defines the maximum range to be used when calculating a random refresh quantity
maxReplenishTimeRange	8	Duration	Defines the maximum range to be used when calculating a random refresh time, in nanoseconds
referencePriceTarget	2	Short	The target relative to the reference base price in basis points of the NBBO spread
expireTime	8	TimeStamp	Timestamp at which the order's open quantity will be canceled, valid only with GTT time in force
userData	8	Long	Pass-through field for use by clients, not used by the exchange
mpid	4	Str(4)	Market Participant Identifier. MPID should contain upper-case alpha only
memberGroup	2	Str(2)	Member supplied grouping. Used for both self-match prevention and mass cancel, in conjunction with MPID or SenderComp
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only
rankPrice	8	Long	Optional rank price, when different from limit price
displayPrice	8	Long	Optional display price, when different from limit price

4.5 LimitOrderRejected

The submitted limit order has been rejected by the Exchange

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'U'/0x55/85
presenceBits	1	4	Int	See Optional Fields
transactTime	5	8	TimeStamp	
clOrdId	13	8	Long	Client-specified order identifier
orderQty	21	4	Int	
limitOrderBitFields	25	4		
symbolId	29	2	Short	
price	31	8	Long	
reason	39	1	Enum	Enum RejectReason. Indicates the reason that a customer action was rejected by the exchange

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	true if a locate is required for short-sale, false otherwise
timeInForce	4	4	Enum TimeInForce.
orderCapacity	8	3	Enum OrderCapacity.
isIso	11	1	true for Reg NMS Intermarket sweep orders, false otherwise
isHidden	12	1	true if any of the order quantity is hidden, false for order quantity to be displayed
isPostOnly	13	1	true to only add the order to the book if it would be added as a resting order with no quantity executed
cancelAtEntryIfCrossed	14	1	true to cancel the order at entry if market conditions are crossed
reserved	15	17	Do not use – reserved for the future

4.5.1 LimitOrderRejected Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasSelfMatchScope	0x00000001
hasSelfMatchInstruction	0x00000002
hasPriceSlideInstruction	0x00000004
hasMinQty	0x00000008
hasMaxFloorQty	0x00000010
hasMaxReplenishQtyRange	0x00000020
hasMaxReplenishTimeRange	0x00000040
hasReferencePriceTarget	0x00000080
hasExpireTime	0x00000100
hasUserData	0x00000200
hasMpid	0x00000400
hasMemberGroup	0x00000800
hasLocateBroker	0x00001000
reserved	0x7FFFE000

4.5.2 LimitOrderRejected Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
selfMatchScope	1	Enum	Enum SelfMatchScope. Determines the scope for self match prevention. For additional information, refer to Self Match Prevention documentation
selfMatchInstruction	1	Enum	Enum SelfMatchInstruction. Determines the behavior when orders from the same entity would match. For additional information, refer to Self Match Prevention documentation
priceSlideInstruction	1	Enum	Enum PriceSlideInstruction. For additional information, refer to Display Price Slide Instructions documentation
minQty	4	Int	Number of shares that must be executed to execute any shares
maxFloorQty	4	Int	Quantity to be displayed at one time with remaining quantity not displayed on the book
maxReplenishQtyRange	4	Int	Defines the maximum range to be used when calculating a random refresh quantity
maxReplenishTimeRange	8	Duration	Defines the maximum range to be used when calculating a random refresh time, in nanoseconds
referencePriceTarget	2	Short	The target relative to the reference base price in basis points of the NBBO spread
expireTime	8	TimeStamp	Timestamp at which the order's open quantity will be canceled, valid only with GTT time in force
userData	8	Long	Pass-through field for use by clients, not used by the exchange
mpid	4	Str(4)	Market Participant Identifier. MPID should contain upper-case alpha only
memberGroup	2	Str(2)	Member supplied grouping. Used for both self-match prevention and mass cancel, in conjunction with MPID or SenderComp
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only

4.6 MarketOrderAccepted

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'D'/0x44/68
presenceBits	1	2	Short	See Optional Fields
transactTime	3	8	TimeStamp	
orderId	11	8	Long	
clOrdId	19	8	Long	Client-specified order identifier
orderQty	27	4	Int	
marketOrderBitFields	31	2		
symbolId	33	2	Short	

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	true if a locate is required for short-sale, false otherwise
timeInForce	4	4	Enum TimeInForce.
orderCapacity	8	3	Enum OrderCapacity.
reserved	11	5	Do not use – reserved for the future

4.6.1 MarketOrderAccepted Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasSelfMatchScope	0x0001
hasSelfMatchInstruction	0x0002
hasUserData	0x0004
hasMpid	0x0008
hasMemberGroup	0x0010
hasLocateBroker	0x0020
reserved	0xFFC0

4.6.2 MarketOrderAccepted Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
selfMatchScope	1	Enum	Enum SelfMatchScope. Determines the scope for self match prevention. For additional information, refer to Self Match Prevention documentation
selfMatchInstruction	1	Enum	Enum SelfMatchInstruction. Determines the behavior when orders from the same entity would match. For additional information, refer to Self Match Prevention documentation
userData	8	Long	Pass-through field for use by clients, not used by the exchange
mpid	4	Str(4)	Market Participant Identifier. MPID should contain upper-case alpha only
memberGroup	2	Str(2)	Member supplied grouping. Used for both self-match prevention and mass cancel, in conjunction with MPID or SenderComp
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only

4.7 MarketOrderRejected

The submitted market order has been rejected by the Exchange

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'T'/0x54/84
presenceBits	1	2	Short	See Optional Fields
transactTime	3	8	TimeStamp	
clOrdId	11	8	Long	Client-specified order identifier
orderQty	19	4	Int	
marketOrderBitFields	23	2		
symbolId	25	2	Short	
reason	27	1	Enum	Enum RejectReason. Indicates the reason that a customer action was rejected by the exchange

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	true if a locate is required for short-sale, false otherwise
timeInForce	4	4	Enum TimeInForce.
orderCapacity	8	3	Enum OrderCapacity.
reserved	11	5	Do not use – reserved for the future

4.7.1 MarketOrderRejected Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasSelfMatchScope	0x0001
hasSelfMatchInstruction	0x0002
hasUserData	0x0004
hasMpid	0x0008
hasMemberGroup	0x0010
hasLocateBroker	0x0020
reserved	0xFFC0

4.7.2 MarketOrderRejected Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
selfMatchScope	1	Enum	Enum SelfMatchScope. Determines the scope for self match prevention. For additional information, refer to Self Match Prevention documentation
selfMatchInstruction	1	Enum	Enum SelfMatchInstruction. Determines the behavior when orders from the same entity would match. For additional information, refer to Self Match Prevention documentation
userData	8	Long	Pass-through field for use by clients, not used by the exchange
mpid	4	Str(4)	Market Participant Identifier. MPID should contain upper-case alpha only
memberGroup	2	Str(2)	Member supplied grouping. Used for both self-match prevention and mass cancel, in conjunction with MPID or SenderComp
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only

4.8 OrderCanceled

This order has been canceled. No shares are left on the Exchange. You will receive this message in response to a CancelOrder message, as well as unsolicited from the Exchange for various reasons. See reason field for details.

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'X'/0x58/88
transactTime	1	8	TimeStamp	
orderId	9	8	Long	
origClOrdId	17	8	Long	The ClOrdId of the order to be canceled
reason	25	1	Enum	Enum CancelReason. Reason that an order was canceled

4.9 CancelRejected

The requested CancelOrder has been rejected. See reason code for details.

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'W'/0x57/87
transactTime	1	8	TimeStamp	
origClOrdId	9	8	Long	The ClOrdId of the order to be canceled
reason	17	1	Enum	Enum RejectReason. Indicates the reason that a customer action was rejected by the exchange

4.10 OrderModified

An order has been modified

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'Y'/0x59/89
presenceBits	1	1	Byte	See Optional Fields
transactTime	2	8	TimeStamp	
orderId	10	8	Long	
clOrdId	18	8	Long	The ClOrdId for this modify message. Must be day-unique
origClOrdId	26	8	Long	The ClOrdId of the order to be modified
leavesQty	34	4	Int	Number of shares that were still available to execute after the modify

4.10.1 OrderModified Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasOrderQty	0x01
hasModifyBitFields	0x02
hasLocateBroker	0x04
reserved	0xF8

4.10.2 OrderModified Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
orderQty	4	Int	Quantity can remain the same or decrease. Quantity cannot increase. For additional information, refer to Modify Behavior documentation
modifyBitFields	1		
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	Processed for Sell Short and Sell Short Exempt in which client affirms ability to borrow (isLocateRequired = false) or client does not affirm ability to borrow (isLocateRequired = true)
reserved	4	4	reserved for future use

4.11 ModifyRejected

The requested ModifyOrder has been rejected. See reason code for details.

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'N'/0x4e/78
presenceBits	1	1	Byte	See Optional Fields
transactTime	2	8	TimeStamp	
clOrdId	10	8	Long	The ClOrdId for this modify message. Must be day-unique
origClOrdId	18	8	Long	The ClOrdId of the order to be modified
reason	26	1	Enum	Enum RejectReason. Indicates the reason that a customer action was rejected by the exchange

4.11.1 ModifyRejected Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasOrderQty	0x01
hasModifyBitFields	0x02
hasLocateBroker	0x04
reserved	0xF8

4.11.2 ModifyRejected Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
orderQty	4	Int	Quantity can remain the same or decrease. Quantity cannot increase. For additional information, refer to Modify Behavior documentation
modifyBitFields	1		
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	Processed for Sell Short and Sell Short Exempt in which client affirms ability to borrow (isLocateRequired = false) or client does not affirm ability to borrow (isLocateRequired = true)
reserved	4	4	reserved for future use

4.12 OrderReplaced

Replace an existing order with a new order

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'J'/0x4a/74
presenceBits	1	2	Short	See Optional Fields
transactTime	3	8	TimeStamp	
orderId	11	8	Long	The new Exchange OrderId of the replace order
clOrdId	19	8	Long	The ClOrdId for this modify message. Must be day-unique
origClOrdId	27	8	Long	The ClOrdId of the order to be replaced
replaceBitFields	35	1		

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	Processed for Sell Short and Sell Short Exempt in which client affirms ability to borrow (isLocateRequired = false) or client does not affirm ability to borrow (isLocateRequired = true)
isIso	4	1	true for Reg NMS Intermarket sweep orders, false otherwise
cancelAtEntryIfCrossed	5	1	true to cancel the order at entry if market conditions are crossed
reserved	6	2	reserved for future use

leavesQty	36	4	Int	Number of shares that were still available to execute after the replace

4.12.1 OrderReplaced Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasPrice	0x0001
hasOrderQty	0x0002
hasMaxFloorQty	0x0004
hasLocateBroker	0x0008
hasRankPrice	0x0010
hasDisplayPrice	0x0020
reserved	0xFFC0

4.12.2 OrderReplaced Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
price	8	Long	
orderQty	4	Int	Quantity can remain the same, decrease or increase. For additional information, refer to Replace Behavior documentation
maxFloorQty	4	Int	Quantity to be displayed at one time with remaining quantity not displayed on the book
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only
rankPrice	8	Long	Optional rank price, when different from limit price
displayPrice	8	Long	Optional display price, when different from limit price

4.13 ReplaceRejected

The requested ReplaceOrder has been rejected. See reason code for details.

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'K'/0x4b/75
presenceBits	1	2	Short	See Optional Fields
transactTime	3	8	TimeStamp	
clOrdId	11	8	Long	The ClOrdId for this modify message. Must be day-unique
origClOrdId	19	8	Long	The ClOrdId of the order to be replaced
replaceBitFields	27	1		

Bit Field Name	Offset	Size	Notes
side	0	3	Enum Side.
isLocateRequired	3	1	Processed for Sell Short and Sell Short Exempt in which client affirms ability to borrow (isLocateRequired = false) or client does not affirm ability to borrow (isLocateRequired = true)
isIso	4	1	true for Reg NMS Intermarket sweep orders, false otherwise
cancelAtEntryIfCrossed	5	1	true to cancel the order at entry if market conditions are crossed
reserved	6	2	reserved for future use

reason	28	1	Enum	Enum RejectReason. Indicates the reason that a customer action was rejected by the exchange

4.13.1 ReplaceRejected Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasPrice	0x0001
hasOrderQty	0x0002
hasMaxFloorQty	0x0004
hasLocateBroker	0x0008
reserved	0xFFF0

4.13.2 ReplaceRejected Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
price	8	Long	
orderQty	4	Int	Quantity can remain the same, decrease or increase. For additional information, refer to Replace Behavior documentation
maxFloorQty	4	Int	Quantity to be displayed at one time with remaining quantity not displayed on the book
locateBroker	4	Str(4)	Used for short sale orders to identify the broker that has loaned the stock to settle the short sale. MPID should contain upper-case alpha only

4.14 OrderExecuted

An order has executed

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'E'/0x45/69
transactTime	1	8	TimeStamp	
orderId	9	8	Long	
clOrdId	17	8	Long	Client-specified order identifier
price	25	8	Long	The price these shares were executed at
execId	33	8	Long	
execQty	41	4	Int	Number of shares that were executed
leavesQty	45	4	Int	Number of shares that were still available to execute after the execution. If this value is 0, then the order has been fully executed.

4.15 OrderRestated

Informational message from the Exchange. Will be sent by the Exchange when it does work on your order. Either your order has been repriced after entry, or a replenish has happened on a reserve order. In the case of a replenishment, the OrderId is the for the new displayed portion of your order. For a replace, it is the new OrderId for your entire order. You can still refer to this order with the Member assigned ClOrdId provided when entering this order. This message is informational. No action is required by the Member.

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'F'/0x46/70
presenceBits	1	1	Byte	See Optional Fields
transactTime	2	8	TimeStamp	
orderId	10	8	Long	A new OrderId has been created, and new priority for this action
clOrdId	18	8	Long	Client-specified order identifier
reason	26	1	Enum	Enum RestatementReason.

4.15.1 OrderRestated Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasRankPrice	0x01
hasDisplayPrice	0x02
hasOrderQty	0x04
reserved	0xF8

4.15.2 OrderRestated Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
rankPrice	8	Long	The rank price. Only present on re-priced orders
displayPrice	8	Long	The display price, when different from rank price. Only present on re-priced orders
orderQty	4	Int	The quantity of the displayed replenished reserve order. Only present on reserve orders

4.16 MassCancelAccepted

The Mass Cancel has been accepted and is being processed

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'O'/0x4f/79
presenceBits	1	1	Byte	See Optional Fields
transactTime	2	8	TimeStamp	
massCancelRequestId	10	8	Long	The MassCancelId assigned by the Member
scope	18	1	Enum	Enum MassCancelScope. For additional information, refer to Mass Cancel Behavior documentation
massCancelId	19	8	Long	An identifier assigned by the Exchange for this Mass Cancel request. For additional information, refer to Mass Cancel documentation.

4.16.1 MassCancelAccepted Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasMpid	0x01
hasSenderComp	0x02
hasMemberGroup	0x04
hasClOrdId	0x08
reserved	0xF0

4.16.2 MassCancelAccepted Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
mpid	4	Str(4)	Market Participant Identifier. MPID should contain upper-case alpha only
senderComp	8	Str(8)	
memberGroup	2	Str(2)	Member supplied grouping. Used for both self-match prevention and mass cancel, in conjunction with MPID or SenderComp
clOrdId	8	Long	Client-specified order identifier

4.17 MassCancelRejected

The Mass Cancel has been rejected

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'P'/0x50/80
presenceBits	1	1	Byte	See Optional Fields
transactTime	2	8	TimeStamp	
massCancelRequestId	10	8	Long	The MassCancelId assigned by the Member
scope	18	1	Enum	Enum MassCancelScope. For additional information, refer to Mass Cancel Behavior documentation
reason	19	1	Enum	Enum RejectReason. Indicates the reason that a customer action was rejected by the exchange

4.17.1 MassCancelRejected Presence Bits

These presence bits are as follows. Any added optional fields must be in their presence bit order See Optional Fields

Bit Name	Set Mask
hasMpid	0x01
hasSenderComp	0x02
hasMemberGroup	0x04
hasClOrdId	0x08
reserved	0xF0

4.17.2 MassCancelRejected Optional Fields

The optional fields begin directly after the end of the field above. If a bit is set in the presence bits, the field must be present, and in the order listed in the presence bits. See Optional Fields

Field Name	Size	Type	Notes
mpid	4	Str(4)	Market Participant Identifier. MPID should contain upper-case alpha only
senderComp	8	Str(8)	
memberGroup	2	Str(2)	Member supplied grouping. Used for both self-match prevention and mass cancel, in conjunction with MPID or SenderComp
clOrdId	8	Long	Client-specified order identifier

4.18 MassCancelResult

The result of the Mass Cancel request

Field Name	Offset	Size	Type	Notes
messageType	0	1	Byte	'Q'/0x51/81
transactTime	1	8	TimeStamp	
massCancelRequestId	9	8	Long	The MassCancelId assigned by the Member
massCancelId	17	8	Long	The MassCancelId assigned by the Exchange, found in the MassCancelAccepted
canceledCount	25	4	Int	The number of orders canceled from the request