

# FAST for OPRA

SIAC Technical Information for  
OPRA Data Recipients

Document Version: 02.10

FAST Version 2

Date: April 5, 2010

## FAST for OPRA

### Revision History

VERSION 1.0 – December 8, 2006	
PAGE(S)	DESCRIPTION
All	Initial Version

VERSION 1.00.01 – February 27, 2007	
PAGE(S)	DESCRIPTION
2	- Updated URL link for FAST 1.1 specification
3	- Updated the Template (Field Operator) Definition Table to reflect OPRA Field Name, Field Operator and Data Type
3	- Added SOH, US, and ETX fields for start and end of packet

VERSION 1.00.02 – March 26, 2007	
PAGE(S)	DESCRIPTION
ALL	<ul style="list-style-type: none"> <li>- Added Revision History</li> <li>- Added title to the FAST Template (Field Operator) descriptions table</li> <li>- Added additional FAST Template (Field Operator) descriptions</li> <li>- Updated definitions to reflect changes in how OPRA Fields are encoded</li> <li>- Added 1 byte Message Length (Message Size) field in front of every encoded message</li> <li>- Change BID_INDEX_VALUE field to be encoded as a STRING Data Type</li> <li>- Change OFFER_INDEX_VALUE field to be encoded as a STRING Data Type</li> <li>- Updated sample code to reflect new Message Length field</li> <li>- Added note regarding elimination of Unit Separator in encoded packets</li> <li>- Added note regarding location of message Category field in encoded messages</li> <li>- A sample 'C' language OPRA FAST decoder supporting all OPRA message formats has been provided in attached zip files</li> </ul>

FAST for OPRA

VERSION 2.00 – January 23, 2008	
PAGE(S)	DESCRIPTION
ALL	<ul style="list-style-type: none"> <li>- Reflects Symbology enhancements in all message formats</li> <li>- Provides templates for all messages</li> <li>- Supports the current version (v1) of FAST for OPRA in new decoder</li> <li>- Supports FAST for Symbology (v2) Phase 1 and Phase 2 in new decoder</li> <li>- Reflects Packet header enhancements</li> <li>- Support for FCO's removed (v2)</li> <li>- Reserved Fields removed in FAST for Symbology (v2) message formats</li> <li>- Support for Line Integrity Messages</li> </ul>
VERSION 2.10 – April 5, 2010	
8 thru 15 8 thru 15 & 17	<ul style="list-style-type: none"> <li>- Strike Price Code is space filled (not removed)</li> <li>- Message Sequence Number updated to reflect "Unsigned Long" and decode_u64</li> </ul>

## FAST

### Overview:

FAST (<http://www.fixprotocol.org/fast>) is an acronym for **F**IX **A**dapted for **S**Treaming. The FAST protocol is defined by the FIX protocol Market Data Optimization Working Group, whose purpose is to develop recommended enhancements to support high frequency market data applications. A technical overview of the protocol can be downloaded from <http://www.fixprotocol.org/documents/2801/FIX%20Adapted%20for%20Streaming%20-%20FAST%20Protocol.pdf>

### Why FAST:

FAST is designed to develop solutions for efficient dissemination of market data. The protocol optimizes communications in the electronic exchange of financial data by reducing the bandwidth between sender and receiver via algorithmic data compaction within each packet.

## FAST for OPRA

The FAST protocol has been integrated with OPRA to reduce the bandwidth of OPRA messages.

### Implementation:

FAST API (<http://www.fixprotocol.org/documents/2317/fastapi-1.0.zip>), a 'C' language implementation of the FAST Protocol, is used to encode OPRA message formats into the FAST format. The FAST Protocol API must be used by OPRA Data Recipients to decode the encoded OPRA messages. The API reference manual and sample implementation can be downloaded from <http://www.fixprotocol.org/fastdownload>

### OPRA FAST Template (Field Operator):

OPRA encodes data utilizing FAST Templates (Field Operators) as described in the table below. A Field Operator defines the structure of encoded data and specifies how data in each field of the OPRA message format is encoded. Data recipients should use these Field Operators during their decoding process. Please refer to chapter 6 of [FAST Specification Version 1.1](#) for a detailed explanation of the Field Operators.

**Encode/Decode field Operator explanation:**

**COPY CODE:** Fields that frequently have the same value in successive instances.

The copy operator specifies that the value of a field is optionally present in the stream. If the value is present in the stream it becomes the new previous value.

When the value is not present in the stream there are three cases depending on the state of the previous value:

- assigned – the value of the field is the previous value.
  
- undefined – the value of the field is the initial value that also becomes the new previous value. Unless the field has optional presence, it is a dynamic error [ERR D5] if the instruction context has no initial value. If the field has optional presence and no initial value, the field is considered absent and the state of the previous value is changed to empty.
  
- empty – the value of the field is empty. If the field is optional the value is considered absent. It is a dynamic error [ERR D6] if the field is mandatory.

The copy operator is applicable to all field types.

**Description:**

The value of {F} will be equal to the previous instance of {F} or {V} if it is the first instance of {F}. A protocol error should be signaled if there is no previous value and {V} has not been specified.

The value of {F} can be set to NULL if the field type supports a NULL value.

**Examples:**

Field Operator Entry	Previous Value	Field Content	Field Value
167=	[None]	[Empty]	[Error]
167=	FUT	[Empty]	FUT
167=FUT	[None]	[Empty]	FUT
167=FUT	IDX	[Empty]	IDX
167=FUT	IDX	FUT	FUT

**INCREMENT:** Fields that frequently have successive values which are incrementally larger than the previous value (sequence numbers).

The increment operator specifies that the value of a field is optionally present in the stream. If the value is present in the stream it becomes the new previous value.

When the value is not present in the stream there are three cases depending on the state of the previous value:

- assigned – the value of the field is the previous value incremented by one. The incremented value also becomes the new previous value.

## FAST for OPRA

- Undefined – the value of the field is the initial value that also becomes the new previous value. Unless the field has optional presence, it is a dynamic error [ERR D5] if the instruction context has no initial value. If the field has optional presence and no initial value, the field is considered absent and the state of the previous value is changed to empty.
- Empty – the value of the field is empty. If the field is optional, the value is considered absent. It is a dynamic error [ERR D6] if the field is mandatory.

The increment operator is applicable to integer field types.

An integer is incremented by adding one to it. If the value is the maximum value of the type it becomes the minimum value after the increment.

### Description:

{N} is a numeric default value.

The value of {F}, if not specified in a message field, will be the value of the previous value of {F} incremented by one, or {N} if it is the first instance of {F}.

If a value is specified in the message it will be used as the current value of {F} and it will be used as the previous value in a subsequent instance of {F} in the same message.

The value of {F} can be set to NULL. The increment of NULL is NULL (which is essentially copy coding behavior for a NULL previous value).

### Examples:

Field Operator Entry	Previous Value	Field Content	Field Value
34+	[None]	[Empty]	[Error]
34+	325	[Empty]	326
34+1	[None]	[Empty]	1
34+1	325	[Empty]	326
34+1	325	401	401

**DELTA:** Fields that frequently have values that are almost equal to the previous value in the same message.

The delta operator specifies that a delta value is present in the stream. If the field has optional presence, the delta value can be NULL. In that case the value of the field is considered absent. Otherwise the field is obtained by combining the delta value with a base value.

**Delta = element delta { opContext }**

The base value depends on the state of the previous value in the following way:

- assigned – the base value is the previous value.
- Undefined – the base value is the initial value if present in the instruction context. Otherwise a type dependant default base value is used.
- Empty – it is a dynamic error [ERR D6] if the previous value is empty.

The following sections define the delta value representations, the default base values and how values are combined depending on type.

## FAST for OPRA

### Description:

The value of {F} will be the value of the previous instance of {F} plus the (delta) value specified for the current instance of {F} (the value given in the message is the delta from the previous instance of {F}). If the value is not specified, zero (0) is used as a default delta value.

FAST Field Encoding Specification 8 (10) 2006-01-13 \* \*

For character string fields, the delta is defined as being the tail characters of the field. As a consequence, the delta value coding can only be used on character string fields with a fixed length.

The value of {F} can be set to NULL if the field type supports a NULL value. The default delta of NULL is NULL (which is essentially copy coding behavior for a NULL previous value).

### Examples:

Field Operator Entry	Previous Value	Field Content	Field Value
270-	[None]	[Empty]	[Error]
270-	[None]	1010	1010
270-	1010	[Empty]	1010
270-	1010	0	1010
270-	1010	-20	990
48-	[None]	[Empty]	[NULL]
48-	[None]	CME000150112	CME000150112
48-	CME000150112	[Empty]	CME000150112
48-	CME000150112	413	CME000150413
48-	CME000150413	0	CME000150410

## General Template for all Messages

ID defines the serialization of the encoding and decoding of FAST for OPRA messages.

FIELD NAME	ID	ENCODE Operator	DATA TYPE
MESSAGE_CATEGORY	0	COPY CODE	Unsigned Integer
MESSAGE_TYPE	1	COPY CODE	Unsigned Integer
PARTICIPANT_ID	2	COPY CODE	Unsigned Integer
RETRANSMISSION_REQUESTER	3	COPY CODE	Unsigned Integer
MESSAGE_SEQUENCE_NUMBER	4	INCREMENT	Unsigned Long
TIME	5	COPY CODE	Unsigned Integer
SECURITY_SYMBOL	6	COPY CODE	STRING
EXPIRATION_MONTH	7	COPY CODE	Unsigned Integer
EXPIRATION_DATE	8	COPY CODE	Unsigned Integer
YEAR	9	COPY CODE	Unsigned Integer
STRIKE_PRICE_DENOMINATOR_CODE	10	COPY CODE	Unsigned Integer
EXPLICIT_STRIKE_PRICE	11	COPY CODE	Unsigned Integer
STRIKE_PRICE_CODE (Space Filled)	12	COPY CODE	Unsigned Integer
VOLUME	13	COPY CODE	Unsigned Integer
OPEN_INT_VOLUME	14	COPY CODE	Unsigned Integer
PREMIUM_PRICE_DENOMINATOR_CODE	15	COPY CODE	Unsigned Integer
PREMIUM_PRICE	16	COPY CODE	Unsigned Integer
OPEN_PRICE	17	COPY CODE	Unsigned Integer
HIGH_PRICE	18	COPY CODE	Unsigned Integer
LOW_PRICE	19	COPY CODE	Unsigned Integer
LAST_PRICE	20	COPY CODE	Unsigned Integer
NET_CHANGE_INDICATOR	21	COPY CODE	Unsigned Integer
NET_CHANGE	22	COPY CODE	Unsigned Integer
UNDERLYING_PRICE_DENOM	23	COPY CODE	Unsigned Integer
UNDERLYING_STOCK_PRICE	24	COPY CODE	Unsigned Integer
BID_PRICE	25	COPY CODE	Unsigned Integer
BID_SIZE	26	COPY CODE	Unsigned Integer
OFFER_PRICE	27	COPY CODE	Unsigned Integer
OFFER_SIZE	28	COPY CODE	Unsigned Integer
SESSION_INDICATOR	29	COPY CODE	Unsigned Integer
BBO_INDICATOR	30	COPY CODE	Unsigned Integer
BEST_BID_PARTICIPANT_ID	31	COPY CODE	Unsigned Integer
BEST_BID_PRICE_DENOMINATOR_CODE	32	COPY CODE	Unsigned Integer
BEST_BID_PRICE	33	COPY CODE	Unsigned Integer
BEST_BID_SIZE	34	COPY CODE	Unsigned Integer
BEST_OFFER_PARTICIPANT_ID	35	COPY CODE	Unsigned Integer
BEST_OFFER_PRICE_DENOMINATOR_CODE	36	COPY CODE	Unsigned Integer
BEST_OFFER_PRICE	37	COPY CODE	Unsigned Integer
BEST_OFFER_SIZE	38	COPY CODE	Unsigned Integer
NUMBER_OF_INDICES_IN_GROUP	39	COPY CODE	Unsigned Integer
NUMBER_OF_FOREIGN_CURRENCY_SPOT_VALUES_IN_GROUP	40	COPY CODE	Unsigned Integer
INDEX_SYMBOL	41	COPY CODE	STRING
INDEX_VALUE	42	COPY CODE	STRING
BID_INDEX_VALUE	43	COPY CODE	STRING
OFFER_INDEX_VALUE	44	COPY CODE	STRING
FCO_SYMBOL	45	COPY CODE	STRING
DECIMAL_PLACEMENT_INDICATOR	46	COPY CODE	Unsigned Integer
FOREIGN_CURRENCY_SPOT_VALUE	47	COPY CODE	Unsigned Integer
TEXT	48	COPY CODE	STRING
DEF_MSG	49	COPY CODE	STRING

## Templates for individual messages

ID defines the serialization of the encoding and decoding of FAST for OPRA messages.

### Category 'a'

FIELD NAME	ID	ENCODE Operator	DATA TYPE
MESSAGE_CATEGORY	0	COPY CODE	Unsigned Integer
MESSAGE_TYPE	1	COPY CODE	Unsigned Integer
PARTICIPANT_ID	2	COPY CODE	Unsigned Integer
RETRANSMISSION_REQUESTER	3	COPY CODE	Unsigned Integer
MESSAGE_SEQUENCE_NUMBER	4	INCREMENT	Unsigned Long
TIME	5	COPY CODE	Unsigned Integer
SECURITY_SYMBOL	6	COPY CODE	STRING
EXPIRATION_MONTH	7	COPY CODE	Unsigned Integer
EXPIRATION_DATE	8	COPY CODE	Unsigned Integer
YEAR	9	COPY CODE	Unsigned Integer
STRIKE_PRICE_DENOMINATOR_CODE	10	COPY CODE	Unsigned Integer
EXPLICIT_STRIKE_PRICE	11	COPY CODE	Unsigned Integer
STRIKE_PRICE_CODE (Space Filled)	12	COPY CODE	Unsigned Integer
VOLUME	13	COPY CODE	Unsigned Integer
PREMIUM_PRICE_DENOMINATOR_CODE	15	COPY CODE	Unsigned Integer
PREMIUM_PRICE	16	COPY CODE	Unsigned Integer
SESSION_INDICATOR	29	COPY CODE	Unsigned Integer

## Category 'k'

ID defines the serialization of the encoding and decoding of FAST for OPRA messages.

FIELD NAME	ID	ENCODE Operator	DATA TYPE
MESSAGE_CATEGORY	0	COPY CODE	Unsigned Integer
MESSAGE_TYPE	1	COPY CODE	Unsigned Integer
PARTICIPANT_ID	2	COPY CODE	Unsigned Integer
RETRANSMISSION_REQUESTER	3	COPY CODE	Unsigned Integer
MESSAGE_SEQUENCE_NUMBER	4	INCREMENT	Unsigned Long
TIME	5	COPY CODE	Unsigned Integer
SECURITY_SYMBOL	6	COPY CODE	STRING
EXPIRATION_MONTH	7	COPY CODE	Unsigned Integer
EXPIRATION_DATE	8	COPY CODE	Unsigned Integer
YEAR	9	COPY CODE	Unsigned Integer
STRIKE_PRICE_DENOMINATOR_CODE	10	COPY CODE	Unsigned Integer
EXPLICIT_STRIKE_PRICE	11	COPY CODE	Unsigned Integer
STRIKE_PRICE_CODE (Space Filled)	12	COPY CODE	Unsigned Integer
PREMIUM_PRICE_DENOMINATOR_CODE	15	COPY CODE	Unsigned Integer
BID_PRICE	25	COPY CODE	Unsigned Integer
BID_SIZE	26	COPY CODE	Unsigned Integer
OFFER_PRICE	27	COPY CODE	Unsigned Integer
OFFER_SIZE	28	COPY CODE	Unsigned Integer
SESSION_INDICATOR	29	COPY CODE	Unsigned Integer
BBO_INDICATOR	30	COPY CODE	Unsigned Integer
BEST_BID_PARTICIPANT_ID	31	COPY CODE	Unsigned Integer
BEST_BID_PRICE_DENOMINATOR_CODE	32	COPY CODE	Unsigned Integer
BEST_BID_PRICE	33	COPY CODE	Unsigned Integer
BEST_BID_SIZE	34	COPY CODE	Unsigned Integer
BEST_OFFER_PARTICIPANT_ID	35	COPY CODE	Unsigned Integer
BEST_OFFER_PRICE_DENOMINATOR_CODE	36	COPY CODE	Unsigned Integer
BEST_OFFER_PRICE	37	COPY CODE	Unsigned Integer
BEST_OFFER_SIZE	38	COPY CODE	Unsigned Integer

## Category 'd'

ID defines the serialization of the encoding and decoding of FAST for OPRA messages.

FIELD NAME	ID	ENCODE Operator	DATA TYPE
MESSAGE_CATEGORY	0	COPY CODE	Unsigned Integer
MESSAGE_TYPE	1	COPY CODE	Unsigned Integer
PARTICIPANT_ID	2	COPY CODE	Unsigned Integer
RETRANSMISSION_REQUESTER	3	COPY CODE	Unsigned Integer
MESSAGE_SEQUENCE_NUMBER	4	INCREMENT	Unsigned Long
TIME	5	COPY CODE	Unsigned Integer
SECURITY_SYMBOL	6	COPY CODE	STRING
EXPIRATION_MONTH	7	COPY CODE	Unsigned Integer
EXPIRATION_DATE	8	COPY CODE	Unsigned Integer
YEAR	9	COPY CODE	Unsigned Integer
STRIKE_PRICE_DENOMINATOR_CODE	10	COPY CODE	Unsigned Integer
EXPLICIT_STRIKE_PRICE	11	COPY CODE	Unsigned Integer
STRIKE_PRICE_CODE (Space Filled)	12	COPY CODE	Unsigned Integer
OPEN_INT_VOLUME	14	COPY CODE	Unsigned Integer

## Category ‘Y’

ID defines the serialization of the encoding and decoding of FAST for OPRA messages.

FIELD NAME	ID	ENCODE Operator	DATA TYPE
MESSAGE_CATEGORY	0	COPY CODE	Unsigned Integer
MESSAGE_TYPE	1	COPY CODE	Unsigned Integer
PARTICIPANT_ID	2	COPY CODE	Unsigned Integer
RETRANSMISSION_REQUESTER	3	COPY CODE	Unsigned Integer
MESSAGE_SEQUENCE_NUMBER	4	INCREMENT	Unsigned Long
TIME	5	COPY CODE	Unsigned Integer
NUMBER_OF_INDICES_IN_GROUP	39	COPY CODE	Unsigned Integer
NUMBER_OF_FOREIGN_CURRENCY_SPOT_VALUES_IN_GROUP	40	COPY CODE	Unsigned Integer
INDEX_SYMBOL	41	COPY CODE	STRING
INDEX_VALUE	42	COPY CODE	STRING
BID_INDEX_VALUE	43	COPY CODE	STRING
OFFER_INDEX_VALUE	44	COPY CODE	STRING
FCO_SYMBOL	45	COPY CODE	STRING
DECIMAL_PLACEMENT_INDICATOR	46	COPY CODE	Unsigned Integer
FOREIGN_CURRENCY_SPOT_VALUE	47	COPY CODE	Unsigned Integer

## Category 'f'

ID defines the serialization of the encoding and decoding of FAST for OPRA messages.

FIELD NAME	ID	ENCODE Operator	DATA TYPE
MESSAGE_CATEGORY	0	COPY CODE	Unsigned Integer
MESSAGE_TYPE	1	COPY CODE	Unsigned Integer
PARTICIPANT_ID	2	COPY CODE	Unsigned Integer
RETRANSMISSION_REQUESTER	3	COPY CODE	Unsigned Integer
MESSAGE_SEQUENCE_NUMBER	4	INCREMENT	Unsigned Long
TIME	5	COPY CODE	Unsigned Integer
SECURITY_SYMBOL	6	COPY CODE	STRING
EXPIRATION_MONTH	7	COPY CODE	Unsigned Integer
EXPIRATION_DATE	8	COPY CODE	Unsigned Integer
YEAR	9	COPY CODE	Unsigned Integer
STRIKE_PRICE_DENOMINATOR_CODE	10	COPY CODE	Unsigned Integer
EXPLICIT_STRIKE_PRICE	11	COPY CODE	Unsigned Integer
STRIKE_PRICE_CODE (Space Filled)	12	COPY CODE	Unsigned Integer
VOLUME	13	COPY CODE	Unsigned Integer
OPEN_INT_VOLUME	14	COPY CODE	Unsigned Integer
PREMIUM_PRICE_DENOMINATOR_CODE	15	COPY CODE	Unsigned Integer
OPEN_PRICE	17	COPY CODE	Unsigned Integer
HIGH_PRICE	18	COPY CODE	Unsigned Integer
LOW_PRICE	19	COPY CODE	Unsigned Integer
LAST_PRICE	20	COPY CODE	Unsigned Integer
NET_CHANGE_INDICATOR	21	COPY CODE	Unsigned Integer
NET_CHANGE	22	COPY CODE	Unsigned Integer
UNDERLYING_PRICE_DENOM	23	COPY CODE	Unsigned Integer
UNDERLYING_STOCK_PRICE	24	COPY CODE	Unsigned Integer
BID_PRICE	25	COPY CODE	Unsigned Integer
OFFER_PRICE	27	COPY CODE	Unsigned Integer

## Category 'C'

ID defines the serialization of the encoding and decoding of FAST for OPRA messages.

FIELD NAME	ID	ENCODE Operator	DATA TYPE
MESSAGE_CATEGORY	0	COPY CODE	Unsigned Integer
MESSAGE_TYPE	1	COPY CODE	Unsigned Integer
PARTICIPANT_ID	2	COPY CODE	Unsigned Integer
RETRANSMISSION_REQUESTER	3	COPY CODE	Unsigned Integer
MESSAGE_SEQUENCE_NUMBER	4	INCREMENT	Unsigned Long
TIME	5	COPY CODE	Unsigned Integer
TEXT	48	COPY CODE	STRING

## Category 'H'

ID defines the serialization of the encoding and decoding of FAST for OPRA messages.

FIELD NAME	ID	ENCODE Operator	DATA TYPE
MESSAGE_CATEGORY	0	COPY CODE	Unsigned Integer
MESSAGE_TYPE	1	COPY CODE	Unsigned Integer
PARTICIPANT_ID	2	COPY CODE	Unsigned Integer
RETRANSMISSION_REQUESTER	3	COPY CODE	Unsigned Integer
MESSAGE_SEQUENCE_NUMBER	4	INCREMENT	Unsigned Long
TIME	5	COPY CODE	Unsigned Integer
TEXT	48	COPY CODE	STRING

## Notes:

1. New OPRA packet format:

SOH	Version Number	Packet Sequence Number	Packet Messages	Message Length	Encoded Messages	----- ----- -----	ETX
1 Byte	1 Byte	10 Bytes	3 Bytes	1 Byte	Variable	----- -----	1 Byte

2. The Start Of Header (SOH) and End Of Text (ETX) in ASCII.
3. An OPRA FAST Version Number field (Version 2) will be added to each packet, representing the version of OPRA FAST contained within the current packet. This field will be one (1) byte, binary.
4. A Packet Sequence Number field will be added to each packet representing the sequence number of the first message within the current packet. This field will be 10 bytes, ASCII, right justified.
5. A Packet Messages field will be added to each packet representing the number of messages contained within the current packet. This field will be three (3) bytes, ASCII, right justified.
6. Message Length indicates message length (MsgSize), is provided in front of every encoded OPRA FAST message.
7. If the encoded message length is larger than 254 then the new field will contain the binary value of 255 (0xFF). In this situation, the end of message will be determined by the ETX delimiter following the message.
8. The Unit separator will not be part of ORPA encoded message. The decoder will add the Unit Separator to comply with the OPRA specification.
9. The message Category field will be located at the beginning of the encoded message as opposed to it's position in the OPRA ASCII format. Please refer OPRA template provided above.
10. u32 to ascii is a utility function that converts integers to ASCII.
11. A sample 'C' language OPRA FAST decoder supporting all FAST for OPRA message formats (current FAST, and Fast for Symbology Phase 1 and Phase 2) has been provided. This decoder can be used to reconstitute the original OPRA formatted packets. In order to reconstitute the original OPRA formatted packet, the appropriate fields should be decoded, converted and or added as described above. Data recipients are welcome to use any FAST decoder program that uses the FAST API described in [FAST Specification Version 1.1](#). A general template has been provided in this document for this purpose.
12. In the event an invalid message category is received by OPRA, the entire message body would be encoded as a string.
13. All Reserved fields have been removed in FAST for Symbology (v2) Phase 1 and Phase 2.
14. Support for Line Integrity messages.

## Sample code for decoding category 'k' OPRA messages:

```
main
{
    OPRA_K_MSG kMsg;
    // add SOH if present – present in first message.
    If (decode_U32(START_OF_HEADER) != -1)
        //add SOH

    Check_OPRA_FAST_Version()

    codec = create_fast_codec()

    while (msgs)
    {
        fast_decode_new_msg(OPRA_BASE_TID) // OPRA template ID

        msgCategory = decode_U32(MESSAGE_CATEGORY)
        switch(msgCategory)
        {
            case 'k':
                decode_equity_index_quote_with_size_ver#(kMsg)
                break
            .....
            .....
            .....
            default:
                decode_opra_default_ver#( kMsg)
                break
        }

        // append decoded msg

        // append unit separator if present –not present in last message.
        If (decode_U32(UNIT_SEPERATOR) != -1)
            //append US

        // append ETX if present – present in the last message.
        If (decode_U32(END_OF_TEXT) != -1)
            //append ETX

        fast_decode_end_msg(OPRA_BASE_TID)
    }
    // append ETX
}

//following decode function is for Ver 2
decode_equity_index_quote_with_size_ver2(OPRA_K_MSG kMsg)
{
    kMsg.category = 'k'
    kMsg.type = decode_u32(MESSAGE_TYPE);
    kMsg.participantId = decode_u32(PARTICIPANT_ID);
    kMsg.retran = decode_u32(RETRANSMISSION_REQUESTER)
    u32_to_ascii(kMsg.seqNumber, sizeof(kMsg.seqNumber),
                decode_u64(MESSAGE_SEQUENCE_NUMBER))
    u32_to_ascii(kMsg.time, sizeof(kMsg.time), decode_u32(TIME))

    memset(kMsg.symbol, ' ', sizeof(kMsg.symbol)) // left justified
    decode_str(SECURITY_SYMBOL, kMsg.symbol, sizeof(kMsg.symbol))
}
```

```

kMsg.expirationMonth = decode_u32(EXPIRATION_MONTH)
u32_to_ascii(kMsg.expirationDate, sizeof(kMsg.expirationDate), decode_u32(EXPIRATION_DATE))
u32_to_ascii(kMsg.year, sizeof(kMsg.year), decode_u32(YEAR))
kMsg.strikePriceDenomCode = decode_u32(STRIKE_PRICE_DENOM_CODE)
u32_to_ascii(kMsg.explicitStrike, sizeof(kMsg.explicitStrike),
             decode_u32(EXPLICIT_STRIKE_PRICE))
kMsg.strikePriceCode = decode_u32(STRIKE_PRICE_CODE)
kMsg.premiumPriceDenomCode = decode_u32(PREMIUM_PRICE_DENOM_CODE)
u32_to_ascii(kMsg.bidQuote, sizeof(kMsg.bidQuote), decode_u32(BID_PRICE))
u32_to_ascii(kMsg.bidSize, sizeof(kMsg.bidSize), decode_u32(BID_SIZE))
u32_to_ascii(kMsg.askQuote, sizeof(kMsg.askQuote), decode_u32(ASK_PRICE))
u32_to_ascii(kMsg.askSize, sizeof(kMsg.askSize), decode_u32(ASK_SIZE))
kMsg.sessionIndicator = decode_u32(SESSION_INDICATOR)
kMsg.bboIndicator = decode_u32(BBO_INDICATOR)
switch(kMsg.bboIndicator)
{
    case 'A': // No Best Bid Change, No Best Offer Change
    case 'B': // No Best Bid Change, Quote Contains Best Offer
    case 'D': // No Best Bid Change, No Best Offer
    case 'E': // Quote Contains Best Bid, No Best Offer Change
    case 'F': // Quote Contains Best Bid, Quote Contains Best Offer
    case 'H': // Quote Contains Best Bid, No Best Offer
    case 'I': // No Best Bid, No Best Offer Change
    case 'J': // No Best Bid, Quote Contains Best Offer
    case 'L': // No Best Bid, No Best Offer
        break;

    case 'C': // No Best Bid Change, Best Offer
    case 'G': // Quote Contains Best Bid, Best Offer
    case 'K': // No Best Bid, Best Offer
        kMsg.bbo.bestOffer.partId = decode_u32(BEST_OFFER_PART_ID)
        kMsg.bbo.bestOffer.denominator =
            decode_u32(BEST_OFFER_DENOM_CODE)
        u32_to_ascii(kMsg.bbo.bestOffer.price, sizeof(kMsg.bbo.bestOffer.price),
                    decode_u32(BEST_OFFER_PRICE))
        u32_to_ascii(kMsg.bbo.bestOffer.size, sizeof(kMsg.bbo.bestOffer.size),
                    decode_u32(BEST_OFFER_SIZE))

        break;

    case 'M': // Best Bid , No Best Offer Change
    case 'P': // Best Bid , No Best Offer
    case 'N': // Best Bid , Quote Contains Best Offer
        kMsg.bbo.bestBid.partId = decode_u32(BEST_BID_PART_ID)
        kMsg.bbo.bestBid.denominator = decode_u32(BEST_BID_DENOM_CODE)
        u32_to_ascii(kMsg.bbo.bestBid.price, sizeof(kMsg.bbo.bestBid.price),
                    decode_u32(BEST_BID_PRICE))
        u32_to_ascii(kMsg.bbo.bestBid.size, sizeof(kMsg.bbo.bestBid.size),
                    decode_u32(BEST_BID_SIZE))

        break;

    case 'O': // Best Bid , Best Offer
        kMsg.bbo.bestBidOffer.bestBid.partId = decode_u32(BEST_BID_PART_ID)
        kMsg.bbo.bestBidOffer.bestBid.denominator =
            decode_u32(BEST_BID_DENOM_CODE)
        u32_to_ascii(kMsg.bbo.bestBidOffer.bestBid.price,
                    sizeof(kMsg.bbo.bestBidOffer.bestBid.price),
                    decode_u32(BEST_BID_PRICE))
        u32_to_ascii(kMsg.bbo.bestBidOffer.bestBid.size,
                    sizeof(kMsg.bbo.bestBidOffer.bestBid.size),
                    decode_u32(BEST_BID_SIZE))

        kMsg.bbo.bestBidOffer.bestOffer.partId =
            decode_u32(BEST_OFFER_PART_ID)
        kMsg.bbo.bestBidOffer.bestOffer.denominator =

```

```
        decode_u32(BEST_OFFER_DENOM_CODE)
    u32_to_ascii(kMsg.bbo.bestBidOffer.bestOffer.price,
        sizeof(kMsg.bbo.bestBidOffer.bestOffer.price),
        decode_u32(BEST_OFFER_PRICE))
    u32_to_ascii(kMsg.bbo.bestBidOffer.bestOffer.size,
        sizeof(kMsg.bbo.bestBidOffer.bestOffer.size),
        decode_u32(BEST_OFFER_SIZE))
```

```
break;
```

```
}
```

```
}
```