

ArcaBook® Specification for Complex Options

For NYSE Arca Options Exchange

January 23, 2009

Version 3.04

Copyright © 2008 NYSE Arca, Inc. All Rights Reserved. No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language in any form by any means without the written permission of NYSE Arca, Inc.

NYSE®, NYSE Arca®Archipelago®, Archipelago Exchange®, ArcaEx®, ArcaBook® and ARCA® are registered trademarks of NYSE Euronex, Inc. Other third party product names used herein are used to identify such products and for descriptive purposes only. Such names may be marks and/or registered marks of their respective owners.

Revision History

Date	Revision	Change Made by:	Synopsis of Change
08-02-2007	Alpha 0.1	Fred Jones	Created Spec
08-13-2007	Alpha 0.2	Fred Jones	Content changes based on input from peers and management.
02-26-2007	Alpha 0.4	Fred Jones	Added Arca FIX Fast Template section. Replace most occurrences of Archipelago with NYSE Arca.
4-2-2008	1.0	Fred Jones	Added RFQ (Request For Quote) field to quote message. Changed encoding for Leg Definitions to BitMap.
4-9-2008			Indicate RFQ for future use.
6-6-2008	1.1	Dzintars Dzilna	Added bandwidth information to Communications chapter.
7-23-2008	1.2	Fred Jones	Indicate price can be positive or negative in Complex Top Quote Message.
8-27-2008	1.3	Fred Jones	Add RFQ
10-14-2008	3.0	David Le	Update with Index Mapping and 64 bit Complex ID
11-14-2008	3.01	Chris McCown	Included Old Ips
12-12-2008	3.02	David Le	Update fast Temple
01-14-2009	3.03	David Le	Added source IP
01-22-2009	3.04	Anil Nagasamudra	Added additional details to index mapping request. Made underlying quantity reserved for future use.
01-31-2009	3.05	Anil Nagasamudra	Changed the Ips, ports to reflect SFTI

Table of Contents

ArcaBook® Specification for Complex Options	i
1 Introduction	5
ArcaBook Interface for Complex Options	5
Data Feeds	5
NYSE Arca API Certification	6
2 Communication	7
Access	7
Multicast Complex Feeds	7
TCP/IP Recovery	8
Bandwidth	8
3 Messages	10
Data Types	10
Sequence Numbers	10
Prices	10
Complex ID	10
Timestamps	11
Packets	11
Compaction	11
Compacted Messages Expanded	11
Recovery	11
4 UDP Multicast Messages	13
Underlying Index Mapping	Error! Bookmark not defined.
Series Index Mapping	Error! Bookmark not defined.
Complex Option Leg Definition Message	13
Top Quote Message	16
Top of Book Messaging	17
System Event Message	18
5 TCP/IP Recovery Messages	19
ArcaBook for Complex Options Recovery Server Messages	19
Messages Sent by a Subscriber	19
Login Request Message	19
Heartbeat Response Message	19
Test Request Message	19
Dropped Packet Request Message	20
Index Mapping Request Message	20
Leg Definitions Request Message	20
Messages Sent by a NYSE Arca	21
Login Accepted Message	21
Login Rejected Message	21
HeartBeat Request Message	22
Test Response Message	22
Packet Replay Message	22

Leg Definition Replay Message	23
6 FIX FAST Protocol	24
Overview	24
A FAST Message.....	24
The ArcaBook for Complex Options FAST Implementation.....	25
Sample Source Code	25
Template Information	26

1 Introduction

ArcaBook is a real-time data feed that disseminates equity, bond and option order/consolidated book information to subscribers for NYSE Arca exchanges. ArcaBook allows subscribers to produce and display the NYSE Arca® open order or consolidated book and ticker. Order routing algorithms can also use ArcaBook data.

NYSE Arca provides four ArcaBook interfaces to meet different customer requirements:

Interface	Equities	Bonds	Options	Description
ArcaBook for Equities and Bonds (formerly ArcaBook Binary)	✓	✓		A binary data feed or FIX Adapted for Streaming (FAST) compacted data feed for Listed, OTC, or Bulletin Board equity orders and/or bond orders.
ArcaBook Multicast for Equities	✓			A binary data feed or FIX Adapted for Streaming (FAST) compacted data feed for Listed and OTC.
ArcaBook for Options			✓	A FAST compacted data feed for aggregate option quotes of the consolidated book.
ArcaBook for Complex Options			✓	A binary data feed for complex options top of book quotes.

This specification is for developers that wish to write applications that interface with ArcaBook for Complex Options (Top of Book). It provides leg definitions and top quotes for Complex Options orders.

ArcaBook Interface for Complex Options

This API is message-based, using fixed length messages over the UDP IP protocol with binary numeric and fixed length ASCII fields. Binary values are in network Endian (Bid Endian) format.

Data Feeds

Customers must connect to one multicast group and make one tcp/ip connection in order to receive the Complex Leg Definitions and to process Top of Book Quotes for complex orders.

- **Complex Leg Definitions:** Complex Leg definitions indicated the available legs submitted in a complex order. Each definition will have a complex.
- **Top of Book:** the single best price-level in the Consolidated Book. This has a price-level index of 1. Quotes are referenced by the complex id.
- **Recovery/Retransmission:** Packet Retransmission from Top of Book and Retransmission of Complex Leg Definitions will be sent down the TCP/IP sessions.

NYSE Arca API Certification

Subscribers must certify their ArcaBook subscription clients with NYSE Arca. NYSE Arca provides an IP address, port number, username, and password to use for testing. To schedule a test, please call the FIX/Connectivity hotline at 888-689-7739 (Option 1) or email connectivity@nyx.com.

2 Communication

Access

ArcaBook for Complex Options subscribers connect to multicast addresses for the primary complex quote feed and can also connect to a TCP/IP server for packet retransmissions.

Multicast Complex Feeds

Multicasts for ArcaBook for Complex Options use UDP (User Datagram Protocol). Currently only one multicast subscription address is used.

The Complex Option subscription address has two data feeds each with an IP address and port. For example:

Current NYSE Arca

Subscription	Type	Underlyings	Primary Multicast IP	Primary Port	Secondary Multicast IP	Secondary Port
252	Complex	A - ZZ	224.1.2.26	11252	224.1.2.46	12252

Future NYSE Arca

Subscription	Type	Underlyings	Primary Multicast IP	Primary Port	Secondary Multicast IP	Secondary Port
124	Complex	A - ZZ	224.0.41.59	11059	224.0.41.187	12187

AMEX

Subscription	Type	Underlyings	Primary Multicast IP	Primary Port	Secondary Multicast IP	Secondary Port
124	Complex	A - ZZ	224.0.58.59	11059	224.0.58.187	12187

Note: The number of subscriptions and their configuration has not been finalized. This information is subject to change.

ArcaBook for Complex Options messages are compacted before transmission and several are transmitted in a single packet. Each packet has a header containing the packet size and sequence number. Packet headers are not compacted. Clients expand compacted messages before processing them.

In the event a packet is lost on the primary feed for a subscription, clients can retrieve the lost packet from the secondary feed. Because UDP is unreliable and may drop packets from both feeds, NYSE ARCA provides a TCP/IP Recovery Server from which clients can request dropped packets. See the [Recovery](#) section for more information.

TCP Source IP addresses

The table below outlines the TCP Source IP addresses, applicable to each Port.

TCP Source IP NYSE ARCA	TCP Source IP AMEX
63.211.72.xxx	208.92.194.xxx
8.9.19.xxx	8.9.33.xxx

Current OX CERT Breakout

Subscription	Type	Underlyings	Primary Multicast IP	Primary Port	Secondary Multicast IP	Secondary Port
252	Complex	A-ZZ	224.1.2.27	10080	224.1.2.47	20080

TCP Source IP CERT addresses

The table below outlines the TCP Source IP addresses, applicable to each Port.

TCP Source IP CERT NYSE ARCA
63.211.72.xxx

TCP/IP Recovery

Subscribers may connect to the TCP/IP Recovery Server to request dropped packets from the ArcaBook for Complex Options multicast feed. The Recovery Server accepts connections on predefined addresses and ports and requires a login before responding to requests. It accepts primary and backup connections to assist recovery on the subscriber's end.

NYSE Arca supplies subscribers with the following parameters:

- An IP address
- A port
- A username
- A password

Subscribers supply NYSE Arca with the IP address for their connection.

Bandwidth

We suggest that the subscribers be able to handle the following message rates and sizes for Complex Options traffic:

Rate/size	Estimate
Complex Options bandwidth (Mbps)	5
Peak message per second rate*	2,000
Packet size	Typically up to 100 bytes**
Maximum number of packets in a day	TBD
Maximum total number of individual quotes messages in a day	TBD

* Function of number of Complex Option instruments; estimate of 2,000 messages per second as peak rate assumes 2,000 Complex Option instruments retained overnight and initial mapping for which to be sent out at start of day.

** Packet size is variable; usually packet will be one message.

3 Messages

Data Types

All numeric fields are in unsigned binary. Binary data is in network Endian (Big Endian) format. All alphanumeric fields are left justified and null padded.

Sequence Numbers

Sequence Numbers for packets and for messages are four byte integers. These numbers start the data feed session at one and increment by one for each new packet or message. See the [Recovery](#) section for more information on sequence numbers.

Prices

Prices are four byte integers in binary scaled to four decimal positions. To determine the decimal price, divide the whole integer price by 10,000.

- **Example 1:** Whole integer price is 135000. The decimal price is $135000 \div 10,000 = 13.50$.
- **Example 2:** Whole integer price is 13500. The decimal price is $13500 \div 10,000 = 1.35$.

Complex ID

The Complex ID will be a value of a 64 bit long. To convert it in a 32 bit processing environment the following example can be used. *Note compaction will use the below structure.*

- **Example 1:** Big Endian Environment

Complex ID	Offset	Len	Type	Fast Template	Notes and Values
ID	0	4	Binary	AB_ORDER_ID	OrderID (1 – 4,294,967,294)
MarketID	4	2	Binary	AB_MARKET_ID	Market ID (0-65,536)
SystemID	6	1	Binary	AB_SYSTEM_ID	Matching engine number (0-256)
Bit	7	1	char	AB_BIT	NULL

- **Example 2:** Little Endian Environment

Complex ID	Offset	Len	Type	Fast Template	Notes and Values
Bit	0	1	char	AB_BIT	NULL
SystemID	1	1	Binary	AB_SYSTEM_ID	Matching engine number (0-256)
MarketID	2	2	Binary	AB_MARKET_ID	Market ID (0-65,536)
ID	4	4	Binary	AB_ORDER_ID	OrderID (1 – 4,294,967,294)

Timestamps

The timestamp field is a four byte integer that provides time in milliseconds starting from Midnight (00:00:00:000) of the trading day. ArcaBook for Complex Options computes timestamps as:

Seconds x 1000 + milliseconds

For example, the timestamp for 10:00:00:376 is converted to:

$(36000 \times 1000) + 376 = 36000376.$

Packets

All Packets are encapsulated in variable length Transmission Blocks, as shown below.

Packet Length	Type	Subscription	Packet Sequence Number	Compacted Messages
The full length of the packet as 2 byte Numeric Binary	A 1 byte Alpha/Numeric code: M = Message B = Heartbeat N = Not found (for TCP/IP packet replay only)	The subscription number for the packet as 1 byte Numeric Binary (252)	A 4 Byte Numeric Binary. For heartbeat packets (Type=B), this is the last Packet Sequence Number sent.	Messages are not present in heartbeat or not found packets (Type B or N).

Compacted messages within a packet may be for different option series.

Heartbeat packets contain no compacted messages and do not increment the Packet Sequence Number. Heartbeat packets are only sent during periods of inactivity to indicate the connection is still open.

Compaction

NYSE Arca compacts all ArcaBook for Complex Options messages using the FIX Adapted for Streaming (FAST) Protocol. The FAST protocol uses several encoding methods to decrease the size of messages sent over the network. The compaction algorithm reduces the size of messages by an average of 75%. See the FIX FAST Protocol section for a detailed explanation and encoding methods.

Compacted Messages Expanded

Compacted messages are a header and body with no delimiters between messages, as shown below:

Message Header and Body	Message Header and Body	Message Header and Body	...
-------------------------	-------------------------	-------------------------	-----

Recovery

In the event a packet is lost on the primary ArcaBook for Complex Options feed for a subscription, clients can check the secondary feed for the lost packet. If both feeds have dropped the packet, clients can request retransmission from the TCP/IP Recovery Server.

Clients use the subscription number and the packet sequence number to request missing packets in the Dropped Packet Request Message. Packet sequence numbers start from one each day for a specific multicast subscription number.

Each quote message within a packet also has a message sequence number. Message sequence numbers start from one each day and are incremented by one per unique complex option id. Message sequence numbers can be used to facilitate the continued processing of the stream while the dropped packets are being recovered.

4 UDP Multicast Messages

All ArcaBook for Complex Options messages are sent over multicast. Once uncompact, they all begin with the following header.

ArcaBook for Complex Options Message Header	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Message Length	0	2	Binary	0 – 65535 (value includes 8 byte header)
Message Type	2	1	Alpha/Numeric	A single character to identify the message: c = complex quote d = complex leg definition
Subscription	3	1	Binary	124 identifying the Subscription.
Time Stamp	4	4	Binary	Milliseconds since Midnight

Underlying Index Mapping

Underlying Index Messages will be the first messages sent down the multicast feeds. Each subscription will send the underlying mappings used on its subscription.

Underlying Index Mapping Message	Offset	Len	Type	Fast Template	Notes and Values
Header (8 bytes)					
Message Length	0	2	Binary		32
Message Type	2	1	Alpha/Numeric	AQ_MSG_TYPE	'n'
Subscription	3	1	Binary	AQ_SUBSCRIPTION	0-255 identifying the
Time Stamp	4	4	Binary	AQ_TIME	Milliseconds since M
Message Body(24 Bytes)					
Underlying Index	8	4	Binary	AQ_UNDERLYING_INDEX	Underlying Stock Ma
Market ID	12	2	Binary	AQ_MARKET_ID	Identifies Market Ori
System ID	14	1	Binary	AQ_SYSTEM_ID	Identifies Trading Er
Bit	15	1	Alpha	AQ_BIT	NULL
Reserved6	16	4	Binary	AQ_RESERVE_6	NULL
Price Scale	20	1	Binary	AQ_PRICE_SCALE	Decimal places on p
Price Resolution	21	1	Alpha	AQ_PRICE_RESOLUTION	'0' = Penny '1' = Penny/Nickel '5' = Nickel/Dime
Exchange Code	22	1	Alpha	AQ_EXCHANGE_CODE	'N' = NYSE

Underlying Index Mapping Message	Offset	Len	Type	Fast Template	Notes and Values
					'Q' = Nasdaq 'P' = Arca 'A' = AMEX
Security Type	23	1	Alpha	AQ_SECURITY_TYPE	A = ADR C = Common E = ETF F = Foreign, I = Units M = Misc, P = Preferred R = Rights S = Shares of Ben I T = Test U = Units W = Warrant
Symbol	24	6	Alpha	AQ_SYMBOL	Stock Ticker Symbol NULL Padded
Reserved3	30	1	Alpha	AQ_RESERVE_3	NULL
Reserved4	31	1	Alpha	AQ_RESERVE_4	NULL

Series Index Mapping

Series Index Messages will be immediately follow the underlying index messages. Each subscription will send the series mappings used on it's subscription.

Series Index Mapping Message	Offset	Len	Type	Fast Template	Notes
Header (8 bytes)					
Message Length	0	2	Binary		60
Message Type	2	1	Alpha/Numeric	AQ_MSG_TYPE	'm'
Subscription	3	1	Binary	AQ_SUBSCRIPTION	0-255
Time Stamp	4	4	Binary	AQ_TIME	Millis
Message Body(52 Bytes)					
Series Index	8	4	Binary	AQ_SERIES_INDEX	Series
Market ID	12	2	Binary	AQ_MARKET_ID	Ident
System ID	14	1	Binary	AQ_SYSTEM_ID	Ident
Bit	15	1	Alpha	AQ_BIT	NULL
Underlying Index	16	4	Binary	AQ_UNDERLYING_INDEX	Under
Reserved1	20	2	Binary	AQ_RESERVE_1	NULL

Series Index Mapping Message	Offset	Len	Type	Fast Template	Notes
Reserved2	22	1	Binary	AQ_RESERVE_2	NUL
Reserved3	23	1	Alpha	AQ_RESERVE_3	NUL
Reserved6	24	4	Binary	AQ_RESERVE_6	NUL
Underlying Quantity	28	4	Binary	AQ_UNDERLYING_QUANTITY	0 – F
Underlying Symbol	32	6	Alpha	AQ_SYMBOL	Unde spac
Expiry Year	38	2	Alpha	AQ_EXPIRE_YEAR	YY
Expiry Month	40	2	Alpha	AQ_EXPIRE_MONTH	MM
Expiry Day	42	2	Alpha	AQ_EXPIRE_DAY	DD
Put or Call	44	1	Alpha	AQ_PUT_CALL	'P' = 'C' =
Strike Price Whole	45	5	Alpha	AQ_STRIKE_PRICE	Righ
Strike Price Decimal	50	3	Alpha	AQ_STRIKE_DECIMAL	Left
Price Scale	53	1	Binary	AQ_PRICE_SCALE	Deci
Option Symbol	54	5	Alpha	AQ_OCC_SYMBOL	The Justi
Reserved4	59	1	Alpha	AQ_RESERVE_4	NUL

Complex Option Leg Definition Message

The Leg Definition message indicates the parts of an option spread. Leg Definition message will reference a unique complex series id.

A single Leg can be defined as an option series leg or a stock leg. This is indicated in the LegSecurityType field.

For Complex Options Leg Definitions, multiple messages can be included in a single packet. The number of Leg Definitions is provided in the message.

The entire complex leg definition will be encoded as AQ_BITMAP.

Leg Definition Message	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Message Length	0	2	Binary	32+ - multiple legs definition can be defined
Message Type	2	1	Alpha/Numeric	'd'
Subscription	3	1	Binary	124 identifying the Subscription
Time Stamp	4	4	Binary	Milliseconds since Midnight
Message Body(32+ Bytes)				
Complex ID	8	8	Binary	NYSE Arca assigned complex id.
NumLegs	16	1	Binary	Indicates number Legs of 1 - 255

Leg Definition Message	Offset	Len	Type	Notes and Values
Pad	17	7	Alpha	Null
Repeating Leg Definition(16+ Bytes)				
Index	24	4	Binary	Stock or Series Index
Market ID	28	2	Binary	NULL
System ID	30	1	Binary	NULL
Bit	31	1	char	NULL
Leg Ratio Quantity	32	4	Binary	Leg Ratio
Leg Side	36	1	Alpha	B = Bid S = Offer
Leg Security Type	37	1	Alpha	O=Option Series Leg E=Equity Stock Leg
Pad	38	2	Alpha	Null

Top Quote Message

Top-of-Book identifies one price-level for one side of the ArcaBook for Complex Options. This is the same method used in ArcaBook for Options Aggregated Quote Message for both top and depth. Each Quote Message has two Price-Level fields identifying the price-level index and the operation that the message indicates for that level:

- Delete Price-Level = the index of the price-level that should be deleted.
- Insert Price-Level. = the index of the price-level that should be inserted.

The Top of Book quote for complex will always modify price-level 1 (for the top of the book).

The Message Sequence Number is on a per complex series id/subscription basis. Each complex series has a set of sequence numbers so that if a packet is lost, the book can still be updated without interruption for any complex series that is not missing messages because of lost packets. Quote Messages and System Event Messages will use the same Message Sequence Numbers because they are on the same subscription. See section 5 for information on recovering missing packets.

Top Quote Message	Offset	Len	Type	Fast Template	Notes and Values
Header (8 bytes)					
Message Length	0	2	Binary		32
Message Type	2	1	Alpha/Numeric	AQ_MSG_TYPE	'c'
Subscription	3	1	Binary	AQ_SUBSCRIPTION	124 identify
Time Stamp	4	4	Binary	AQ_TIME	Milliseconds
Message Body(32 Bytes)					
Complex ID	8	8	Binary	See Complex ID Definition	NYSE Arca 4,294,967,2

	Delete Price Level	Insert Price Level	Insert Price	Quantity
Top Bid Quote	1	1	0	0

	Price Level	Price	Quantity
Original Top of Book	1	6	200
Modified Top of Book	1	0	0

System Event Message

System Event messages are used to clear option quotes and send halt notifications.

System Event Message	Offset	Len	Type	Fast Template	Notes and
Header (8 bytes)					
Message Length	0	2	Binary		24
Message Type	2	1	Alpha/Numeric	AQ_MSG_TYPE	'e'
Subscription	3	1	Binary	AQ_SUBSCRIPTION	124 identify
Time Stamp	4	4	Binary	AQ_TIME	Milliseconds
Message Body(16 Bytes)					
Complex ID	8	8	Binary	See Complex ID Definition	NYSE Arca
Message Sequence Number	16	4	Binary	AQ_SEQUENCE	NYSE Arca 4,294,967,2
Reserve5	20	2	Alpha	AQ_RESERVE_5	NULL
Event Code	22	1	Alpha	AQ_EVENT_CODE	C = Clear C I = Clear Su
Reset Code	23	1	Alpha	AQ_RESET_CODE	C = Continu numbers. R = Reset F expected M for this subs a Packet Se be the first r subscription during a Sys used in conj

5 TCP/IP Recovery Messages

ArcaBook for Complex Options Recovery Server Messages

Messages Sent by a Subscriber

Login Request Message

Clients send this message to log into the ArcaBook for Options Recovery Server. The server responds with a Login Accepted or Login Rejected message.

Logon Request Message	Offset	Len	Type	Notes and Values
Header (8bytes)				
Message Length	0	2	Binary	28 bytes
Message Type	2	1	Alpha/Numeric	'L'
Padding	3	1		
Time Stamp	4	4	Binary	Milliseconds since Midnight (not required)
Message Body(20 Bytes)				
Username	8	8	Alpha	Username
Password	16	12	Alpha	Password

Heartbeat Response Message

Clients send this message as a response to the Heartbeat Request message from the ArcaBook for Options Recovery Server.. If the server does not receive a Heartbeat Response within 60 seconds of sending the Heartbeat Request message, the server closes the connection. This message only contains a header.

HeartBeat Response Message	Offset	Len	Type	Notes and Values
Message Length	0	2	Binary	8 bytes
Message Type	2	1	Alpha/Numeric	'H'
Padding	3	1		
Time Stamp	4	4	Binary	Milliseconds since Midnight (not required)

Test Request Message

Clients send this message to request a response from the ArcaBook for Options Recovery Server during periods of inactivity. The client can specify a text message for the server to echo backing its response.

Test Request Message	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Message Length	0	2	Binary	28 bytes
Message Type	2	1	Alpha/Numeric	'T'

Test Request Message	Offset	Len	Type	Notes and Values
Padding	3	1		
Time Stamp	4	4	Binary	Milliseconds since Midnight (not required)
Message Body(20 Bytes)				
Test Message	8	20	Alpha	Text to be echoed

Dropped Packet Request Message

Clients request missing packets with this message. This can be a single packet or a contiguous set of packets. Packets are identified by the Subscription number (multicast address) and the packet number.

Packet Request Message	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Message Length	0	2	Binary	16 bytes
Message Type	2	1	Alpha/Numeric	'P'
Subscription	3	1	Binary	124, the multicast subscription number for this missed packet
Time Stamp	4	4	Binary	Milliseconds since Midnight (not required)
Message Body(8 Bytes)				
Starting Packet Number	8	4	Binary	1 – 4,294,967,294
Ending Packet Number	12	4	Binary	1 – 4,294,967,294

Index Mapping Request Message

Clients send this message to request a response from the ArcaBook for Options Recovery Server to get a full mapping of underlying stock and series details. The mapping messages are sent over the TCP socket uncompacted for all subscriptions.

Test Request Message	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Message Length	0	2	Binary	8 bytes
Message Type	2	1	Alpha/Numeric	'M'
Subscription	3	1	Binary	124, the multicast subscription number
Time Stamp	4	4	Binary	Milliseconds since Midnight (not required)

Leg Definitions Request Message

Clients request leg definitions with this message. Client can request all definitions for the specific subscription by specifying a 0 (zero) in the Complex ID field or clients can request a single definition by specifying a valid Complex ID (1 – N).

Packet Request Message	Offset	Len	Type	Notes and Values
------------------------	--------	-----	------	------------------

Packet Request Message	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Message Length	0	2	Binary	20 bytes
Message Type	2	1	Alpha/Numeric	'D'
Subscription	3	1	Binary	124, the multicast subscription number
Time Stamp	4	4	Binary	Milliseconds since Midnight (not required)
Message Body(8 Bytes)				
Complex ID	8	8	Binary	Complex Series Id.
Pad	16	4	Alpha	Null

Messages Sent by a NYSE Arca

Login Accepted Message

The ArcaBook for Options Recovery Server sends this message to indicate that a client's login request has been accepted.

Login Accepted Message	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Message Length	0	2	Binary	12 bytes
Message Type	2	1	Alpha/Numeric	'l'
Padding	3	1		
Time Stamp	4	4	Binary	Milliseconds since Midnight
Message Body(4 Bytes)				
Padding	8	4	Alpha	

Login Rejected Message

The ArcaBook for Options Recovery Server sends this message when a client request to log in is invalid. This message is also sent when the Recovery Server has exceeded the maximum connection limit for this port or when a connection has timed out (client connects and does not log in within 30 seconds). The Reject Code field indicates the reason for the rejection. ArcaBook for Options Recovery Server closes the socket connection after sending this message.

Login Rejected Message	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Message Length	0	2	Binary	12 bytes
Message Type	2	1	Alpha/Numeric	'r'
Padding	3	1		
Time Stamp	4	4	Binary	Milliseconds since Midnight
Message Body(4 Bytes)				
Reject Code	8	1	Alpha	A=Not Authorized

Login Rejected Message	Offset	Len	Type	Notes and Values
				M=Maximum Server Connections Reached T=Timeout
Padding	9	3	Alpha	

HeartBeat Request Message

The Recovery server sends this message every 60 seconds. This prevents some firewalls from timing out the TCP/IP connection. Clients must respond with a Heartbeat Response message. This message only has a message header.

HeartBeat Message	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Message Length	0	2	Binary	8 bytes
Message Type	2	1	Alpha/Numeric	'h'
Padding	3	1		
Time Stamp	4	4	Binary	Milliseconds since Midnight

Test Response Message

The Recovery Server sends this message in response to a Test Request message from a client.

Test Response Message	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Message Length	0	2	Binary	28 bytes
Message Type	2	1	Alpha/Numeric	't'
Padding	3	1		
Time Stamp	4	4	Binary	Milliseconds since Midnight
Message Body(20 Bytes)				
Test Message	8	20	Alpha	The client text to echo from the Test Request message

Packet Replay Message

The ArcaBook for Options Recovery Server sends this message in response to client requests for missing packets.

Packet Replay Message	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Packet Length	0	2	Binary	Size of Compacted Messages
Message Type	2	1	Alpha	See the Packets section for more information. M= Message N=Not found if the packet requested is unknown

Packet Replay Message	Offset	Len	Type	Notes and Values
Subscription	3	1	Binary	124 identifying the Subscription
Packet Sequence Number	4	4	Binary	Sequence Number of Packet
Message Body				
Messages	8	Variable	Same as original Packet	See the Each quote message within a packet also has a message sequence number. Message sequence numbers start from one each day and are incremented by one per unique complex option id. Message sequence numbers can be used to facilitate the continued processing of the stream while the dropped packets are being recovered. UDP Multicast Messages section for more information.

Leg Definition Replay Message

The ArcaBook for Options Recovery Server sends this message in response to client requests for leg definitions. This is same leg definition message that was sent in the UDP feed with the exception that the message is sent back via TCP/IP and is not compressed. Please See section 4 Complex Option Leg Definition Message.

Leg Definition Message	Offset	Len	Type	Notes and Values
Header (8 bytes)				
Message Length	0	2	Binary	32+ - multiple legs definition can be defined
Message Type	2	1	Alpha/Numeric	'd'
Subscription	3	1	Binary	124 identifying the Subscription
Time Stamp	4	4	Binary	Milliseconds since Midnight
Message Body(24+ Bytes)				
Complex ID	8	8	Binary	NYSE Arca assigned complex series id.
NumLegs	16	1	Binary	Indicates number Legs of 1 - 255
Pad	17	3	Alpha	Null
Repeating Leg Definition(12+ Bytes)				
Leg Ratio Quantity	20	4	Binary	Leg Ratio
Index	24	4	Binary	Stock or Series Index
Leg Side	28	1	Alpha	B = Bid S = Offer
Leg Security Type	29	1	Alpha	O=Option Series Leg E=Equity Stock Leg
Pad	30	2	Alpha	Null

6 FIX FAST Protocol

Overview

Subscribers receive the ArcaBook for Complex Options real-time data feed in the FAST Protocol. This protocol is a standard method for compacting real-time market data resulting in reduced bandwidth and reduced latency. The complete FAST specification is available at:

<http://fixprotocol.org/documents/1766/FAST%20SERDES%20Specification%200.5%202005-07-28.zip>

and <http://fixprotocol.org/documents/1536/BMF%20Specification%200.14.zip>

The FAST Protocol uses two main approaches to reduce bandwidth:

- **Omit Redundant Fields:** this uses two FAST features:
 - FAST Templates that specify the FAST field encoding to control field omission and reconstitution. Field encoding schemes define whether fields can be omitted and how they should be interpreted if omitted.

For example, Copy encoding specifies that if a field is not present, you should use a copy of the field from the previous message. Increment encoding specifies that you should use the previous value and increment it by some constant (usually 1). A field defined with an encoding scheme of None means that it will always be present.
 - Presence Map that indicates which fields are actually present in a message.

The combination of field encoding templates and presence maps allows the contents of a message to be communicated fully while reducing the number of bytes on the wire.

- **Variable Length Fields:** that compact the bits used to represent a field's value. This uses continuation bit encoding to separate the fields. Only the first seven bits of a byte transmit data. The high bit is the continuation bit that indicates whether data for the field continues or stops. When the high bit is set, this is called a stop bit and indicates the end of the variable length field.

A FAST Message

A FAST message consists of a minimum of a one byte Presence Map (pmap) followed by zero or more bytes of field data, as shown below:

```
FastMessage ::= ::= < pmap { pmap} > < { field } >
```

The pmap may be more than one byte and also uses continuation bit encoding (it ends in a stop bit). The pmap sets individual bits to either 1 or 0 to indicate if a specific field is present in the FAST message.

A field within a FAST message can represent one of four data types:

- signed integer
- unsigned integer
- ASCII string
- Bitmap

All fields are variable length, ending in a stop bit.

The ArcaBook for Complex Options FAST Implementation

The ArcaBook for Complex Options FAST implementation reduces bandwidth requirements by up to 80%, or a ratio of 5 to 1. Each message within the FAST ArcaBook for Complex Options data feed has a minimum of two bytes: a Presence Map of at least one byte and a Message Type field of one byte. Note that there may be more than one byte in the pmap, but there will always be at least one. The encoding scheme of None for the message type field guarantees that it will be present in every message.

Sample Source Code

To help subscribers process the ArcaBook for Complex Options FAST feed, NYSE Arca provides source code to decode ArcaBook for Complex Options FAST messages into ArcaBook for Complex Options binary messages. A single, C language routine, `AQFastDecodeComplex()`, decodes ArcaBook for Complex Options FAST messages. The following pseudo code describes the decoding process.

```
Define some variables to hold our input buffer and results
Integer length
Integer result
Byte buffer[2048]
ArcaBookOptionsMessage message;

Process until we are told to stop
Do

    Call the decode routine, we decode the FAST message in
    "buffer" and place the result in "message", "length" will
    contain the number of bytes we processed in "buffer".
    result = AQComplexFastDecode(buffer, length, message)

    Check the result code
    If result == AB_OK Then
        process the ArcaBookOptionsBinary message, and
        advance the buffer to buffer + length
        ProcessMessage(message)
    Else If result == AQ_INCOMPLETE_ERROR Then
        buffer did not contain a full FAST message, so
        read more bytes from the Multicast or TCP Recovery socket
        and place the result into buffer
        length = SocketRead(buffer,1024)
    Else
        We encountered some other error
        ProcessError(result)
    End

While Stop == False
```

This pseudo code is a very basic example. Please see the provided C source code for a full, working example.

Template Information

The FAST template for each message indicates which fields may be omitted from a message and how clients should interpret omitted fields. ArcaBook FAST messages use the message type as the FAST template ID. Once clients have parsed the message type, the rest of the message can be parsed based on the template shown in Table 1.

Table 1: ArcaBook for Options FAST Message Template

Field ID	Field Name	In Messages of Type	FAST Type	Encoding
0	AB_MSG_TYPE	m, n, c, e,	Unsigned8	None
1	AB_SUBSCRIPTION	m, n, c, e,	Unsigned8	Copy
2	AB_TIME	m, n, c, e,	Unsigned32	Copy
3	AB_SEQUENCE	c, e,	Unsigned32	Increment
4	AB_SERIES_INDEX	m,	Unsigned32	Copy
	AB_ORDER_ID	c, e,		
5	AB_UNDERLYING_INDEX	m, n,	Unsigned32	Copy
	AB_COMPLEX_ID	c, e,		
6	AB_PRICE	c,	Unsigned32	Copy
7	AB_UNDERLYING_QUANTITY	m,	Unsigned32	Copy
	AB_QUANTITY	c,		
9	AB_MARKET_ID	m, n, c, e,	Unsigned16	Copy
10	AB_INSERT_LEVEL	c,	Unsigned8	Copy
11	AB_DELETE_LEVEL	c,	Unsigned8	Copy
12	AB_SYSTEM_ID	m, n, c, e,	Unsigned8	Copy
13	AB_PRICE_SCALE	m, n,	Unsigned8	Copy
	AB_RFQ	c,		
14	AB_SYMBOL	m, n,	ascii	Copy
15	AB_EXPIRE_YEAR	m,	ascii	Copy
16	AB_EXPIRE_MONTH	m,	ascii	Copy
17	AB_EXPIRE_DAY	m,	ascii	Copy
18	AB_STRIKE_PRICE	m,	ascii	Copy
19	AB_STRIKE_DECIMAL	m,	ascii	Copy
20	AB_OCC_SYMBOL	m,	ascii	Copy
21	AB_PUT_CALL	m,	char	Copy
	AB_SECURITY_TYPE	n,		
	AB_BUY_SELL	c,		
	AB_EVENT_CODE	e,		
22	AB_BIT	m, n, c, e,	char	Copy
23	AB_PRICE_RESOLUTION	n,	char	Copy
	AB_RESET_CODE	e,		
24	AB_EXCHANGE_CODE	n,	char	Copy
25	AB_RESERVE_1	n,	Unsigned16	Copy

Field ID	Field Name	In Messages of Type	FAST Type	Encoding
26	AB_RESERVE_2	n,	Unsigned8	Copy
27	AB_RESERVE_3	n, m,	char	Copy
28	AB_RESERVE_4	n, m,	char	Copy
29	AB_RESERVE_5	e,	ascii	Copy
30	AB_RESERVE_6	c,	Unsigned32	Copy
31	AB_BITMAP	d,	Bitmap	None

Note: Field ID's with multiple Field Names are guaranteed never to occur more than once in a given message.