CLAIMS

1 . A system for pause-less end-of-interval messaging with failover protection, comprising:

one or more processors programmed to execute a first instance of a matching engine and a second instance of the matching engine, the first instance and the second instance executing on the same or different processors of the system, wherein the second instance is a backup instance for fault tolerance if the first instance becomes unavailable;

wherein each of the first instance and the second instance is programmed to:

receive a plurality of messages over time from a sequencer, each message of the plurality of messages identifying a respective order from a participant to be matched with another order via the matching engine and having been sequenced with respect to other messages of the plurality of messages, wherein each order has an expiration after which the order should no longer be matched;

receive, from the sequencer, an end-of-interval (EOI) rollover message associated with an EOI expiration that indicates orders that have expired based on a respective expiration of the order and the EOI expiration are to be expired and no longer matched to other orders;

responsive to the EOI rollover message:

identify a to-be-expired order based on its expiration,

generate a loopback message that identifies the to-be-expired order without expiring the to-be-expired order while continuing to receive new messages for new orders from the sequencer, and

transmit the loopback message to the sequencer, which will sequence the loopback message along with new messages for new orders that are received after the EOI rollover message;

receive one or more new messages from the sequencer;

determine that at least one new message of the one or more new messages is the loopback message or a counterpart loopback message sent to the sequencer by the other instance of the matching engine; and

expire the order corresponding to the loopback message or the counterpart loopback message, wherein expiration of the order occurs on both the first instance and the second instance without blocking new messages from the sequencer.

2. The system of claim 1 , wherein the first instance and the second instance each is to:

store the plurality of messages, as they are received, in a message log; and

responsive to receipt of the EOI rollover message, generate a current EOI snapshot based on the plurality of messages in the message log to maintain a snapshot of messages received in a current EOI time period, the current EOI snapshot being used to recover messages received during the time period before the EOI rollover message is received and after a prior EOI rollover message was received.

3. The system of claim 1 , wherein the first instance and the second instance each accesses a respective loopback message send queue, and each of the first instance and the second instance is to:

add the loopback message to the respective loopback message send queue along with other loopback messages that each identify a respective to-be-expired order; wherein to transmit the loopback message to the sequencer, the first instance and the second instance each transmit the loopback message from its respective loopback message send queue.

4. The system of claim 3, wherein to transmit the loopback message from its respective loopback message send queue, each of the first instance and the second instance is to:

execute a throttling thread that transmits the loopback message from its respective loopback message send queue based on one or more throttling parameters that define a number or rate of loopback messages to be transmitted to the sequencer to control the number or rate of loopback messages that are transmitted to the sequencer from each of the first instance and the second instance.

5. The system of claim 1 , wherein each of the first instance and the second instance is to:

after the EOI rollover message is received, receive a new message from the sequencer, the new message comprising a new order to be matched; and

process the new order to be matched without pause while continuing to process the EOI rollover message, optionally wherein each of the first instance and the second instance is to:

determine that the new order specifies a counter-part order that is-to-be expired; and

expire the counter-part order, further optionally wherein each of the first instance and the second instance is to:

determine that all to-be-expired orders have been expired; and

transmit a rollover completed loopback message indicating that all of the to-be- expired orders have been expired.

6. The system of claim 1 , wherein the at least one new message of the one or more new messages is the loopback message.

7. The system of claim 1 , wherein the at least one new message of the one or more new messages is the counterpart loopback message.

8. The system of claim 1 , wherein each of the first instance and the second instance is to:

add the plurality of messages and the one or more new messages into a respective inbound message queue.

9. The system of claim 1 , wherein the one or more processors are further programmed to execute a sequencer, the sequencer to:

receive a plurality of messages from (i) one or more market participants and/or (ii) the first instance and the second instance;

sequence the plurality of messages; and

transmit the sequenced plurality of messages to each of the first instance and the second instance.

10. The system of claim 1 , wherein the one or more processors are further programmed to execute a sequencer, the sequencer to:

transmit an EOI rollover message to each of the first and second instances to indicate that an EOI rollover is to occur.

11. A method, comprising:

receiving, by a matching engine instance of a computer system, a plurality of messages over time from a sequencer, each message of the plurality of messages identifying a respective order from a participant to be matched with another order

via the matching engine and having been sequenced with respect to other messages of the plurality of messages, wherein each order has an expiration after which the order should no longer be matched;

receiving, by the matching engine instance from the sequencer, an end-of-interval (EOI) rollover message associated with an EOI expiration that indicates orders that have expired based on a respective expiration of the order and the EOI expiration are to be expired and no longer matched to other orders;

responsive to the EOI rollover message:

identifying, by the matching engine instance, a to-be-expired order based on its expiration,

generating, by the matching engine instance, a loopback message that identifies the to-be-expired order without expiring the to-be-expired order while continuing to receive new messages for new orders from the sequencer, and

transmitting, by the matching engine instance, the loopback message to the sequencer, which will sequence the loopback message along with new messages for new orders that are received after the EOI rollover message;

receiving, by the matching engine instance, one or more new messages from the sequencer;

determining, by the matching engine instance, that at least one new message of the one or more new messages is the loopback message or a counterpart loopback message sent to the sequencer by the other instance of the matching engine; and

expiring, by the matching engine instance, the order corresponding to the loopback message or the counterpart loopback message, wherein expiration of the order occurs without blocking new messages from the sequencer.

12. The method of claim 11, further comprising:

storing the plurality of messages, as they are received, in a message log; and

responsive to receipt of the EOI rollover message, generating a current EOI snapshot based on the plurality of messages in the message log to maintain a snapshot of messages received in a current EOI time period, the current EOI snapshot being used to recover messages received during the time period before the EOI rollover message is received and after a prior EOI rollover message was received.

13. The method of claim 11, further comprising:

accessing a respective loopback message send queue;

adding the loopback message to the respective loopback message send queue along with other loopback messages that each identify a respective to-be-expired order; wherein to transmit the loopback message to the sequencer, the first instance and the second instance each transmit the loopback message from its respective loopback message send queue, optionally wherein transmitting the loopback message comprises:

executing a throttling threat that transmits the loopback message from its respective loopback message send queue based on one or more throttling parameters that define a number or rate of loopback messages to be transmitted to the sequencer to control the number or rate of loopback messages that are transmitted to the sequencer from each of the first instance and the second instance.

14. The method of claim 11, further comprising:

after the EOI rollover message is received, receiving a new message from the sequencer, the new message comprising a new order to be matched; and

processing the new order to be matched without pause while continuing to process the EOI rollover message, optionally further comprising:
determining that the new order specifies a counter-part order that is-to-be expired; and expiring the counter-part order, further optionally further comprising:

determining that all to-be-expired orders have been expired; and

transmitting a rollover completed loopback message indicating that all of the to- be-expired orders have been expired.

15. The method of claim 11, further comprising:

receiving, by a sequencer of the computer system, a plurality of messages from
(i) one or more market participants and/or (ii) the first instance and the second instance;

sequencing, by the sequencer, the plurality of messages; and

transmitting, by the sequencer, the sequenced plurality of messages to each of the first instance and the second instance.