PMSF
IT Consulting

RFMI Protocol Specification

Version 0.91 Draft

Pierre R. Mai <pmai@pmsf.de>

2017-10-27

Contents

1 General Information

The Remote FMI (RFMI) Protocol is intended to be a low-overhead, low-latency, high-bandwidth protocol for accessing the functionalities of FMUs hosted by an RFMI server from RFMI clients across TCP/IP-based networks or potentially other transport mechanisms, like (R)DMA-based communication mechanisms.

All communication is initiated by an RFMI Client, with the RFMI server, which is hosting the FMUs, passively responding to incoming RFMI client requests.

1.1 Data Formats

1.1.1 Byte Size and Byte-Ordering

All information is exchanged based on 8-bit bytes (octets). The byte-ordering for all larger data-types is determined upon session start-up by negotiation, with a default of little-endian (Intel) byte-order.

1.1.2 Data Types in Messages

All integer fields in messages are unsigned integers, unless specifically noted otherwise, in the byte-ordering that has been determined upton session start-up by negotiation, again unless specifically noted otherwise.

All designated string fields in messages are transmitted in UTF-8 encoding with a trailing zero termination and a leading 32bit unsigned integer length field, which will indicate the number of bytes the string field contains, including the zero termination but excluding the length field, and excluding any additional zero-byte padding at the end of the string. All string fields are zero-byte padded to end on a 4 byte (32-bit) boundary.

1.1.3 Defined Data Types for FMU values

For the transmission of FMU values inside frames, the following set of data types are currently defined:

Type Id	Type Name	Size	Alignment	Description
0x0011	Boolean	1	1	0 is False, 1 is True, FMI 1.0
0x0012	Boolean2	4	4	0 is False, 1 is True, FMI 2.0
0x0021	Integer	4	4	32bit Signed Integer
0x0031	Real	8	8	IEEE 754 Double Precision Float
0x0041	String	n	4	Zero-terminated UTF-8 String
0x0051	Binary	n	4	Length-terminated Binary

Note that string values in frames are prepended by a leading 32bit length count, which gives the length of the zero-terminated UTF-8 string, including the zero termination but excluding the length field and excluding any zero-byte padding at the end of the string. All string values are zero-byte padded to end on a 4 byte (32-bit) boundary.

There are two different boolean types, Boolean (Type Id 0x0011) and Boolean2 (Type Id 0x0012) to efficiently support the differing type sizes in FMI 1.0 (Booleans are byte-sized) and FMI 2.0 (Booleans are the same size as Integer).

The RFMI protocol also supports transmission of length-terminated binary value fields, as needed e.g. to support OSMP (OSI Sensor Model Packaging) compliant FMUs. The representation of these binary fields is similar to the representation of string fields, with the difference that binary fields are not zero-terminated. The binary data is prepended by a leading 32bit length count, which gives the length of the binary data, excluding the length field and any zero-byte padding at the end of the string. All binary values are zero-byte padded to end on a 4 byte (32-bit) boundary.

1.1.4 Data Layout in Frames

When transmitting values inside a frame, each sub-frame is aligned to the alignment of its data type. If necessary, zero padding is added before the sub-frame (after the preceding sub-frame) to get proper alignment. It is therefore recommended to define frames with sub-frames in the order from largest to smallest alignment (with the exception of strings), i.e. real before integer before boolean before string; in this way no padding is necessary, except the padding inherent in strings. This order also ensures that the statically sized part of a frame is contiguous at the beginning, with the dynamically sized string part (if any) at the end.

Note that all frames start at an 8 byte aligned boundary. If a message contains data before the frame data, the start of the frame data will still always be aligned to an 8 byte aligned boundary, if necessary by including padding before the start of the frame data.

1.2 Generic Message Structure

All messages being communicated contain the following initial fields:

Offset	Size	Description
0x0000	4	Command/Response Code: 32bit unsigned integer field in actual byte-order This field indicates the command or response being sent from the RFMI client to the RFMI server (command) or vice-versa (response). Note that command and response codes are choosen in such a way that they indicate a readable mnemonic if output in little-endian byte order and the reverse in big-endian byte order. Note also that command codes always use all upper-case ASCII letters whereas response codes always use all lower-case ASCII letters.
0x0004	4	Command/Response Flags: This 32bit unsigned byte field can contain additional flag information for the message. The actual meaning of the flags will depend on the message command/response code.
0x0008	8	Message Length: 64bit unsigned byte length of the complete message in bytes, including the command/response code and the message length byte. This message length field can be used to skip unknown messages and/or messages containing unknown additional content, and also to check for certain forms of message corruption.

Following the initial fields is the actual message payload, which depends on the message being sent.

1.3 Generic Response Codes

The RFMI server can respond to all command messages with one of the following generic response messages, usually indicating a fault situation:

1.3.1 Fatal Error Response Message

This message is sent by the RFMI server when it encountered a fatal error condition which necessitates the shutdown of the connection. The RFMI server will shut down its side of the connection the moment it has sent the fatal error response message, so the RFMI client should not expect to be able to communicate with the RFMI server on this connection any longer.

Offset	Size	Description
0x0000	4	Select FMU Response: 0x6C746166 ('fatl')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 22 + Error Length
0x0010	4	Error Code: 32bit unsigned integer giving a more specific cause for the error. Default Value is 0x00000000, which indicates a generic error cause.
0x0014	4	Error Length: 32bit unsigned length of error string
0x0018	n	Error String: Error Length 8bit bytes, UTF-8 encoded, zero-terminated message describing the fault condition leading to the fatal error.

1.3.2 Error Response Message

This message is sent by the RFMI server when it encountered a non-fatal error condition, i.e. the connection can stay up and the RFMI server is prepared to handle further messages. Whether those messages will succeed will depend on the internal state of the RFMI server, i.e. the RFMI client should be prepared to handle additional error conditions, potentially chosing to abort the connection.

Offset	Size	Description
0x0000	4	Select FMU Response: 0x726F7265 ('eror')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 22 + Error Length
0x0010	4	Error Code: 32bit unsigned integer giving a more specific cause for the error. Default Value is 0x00000000, which indicates a generic error cause.
0x0014	4	Error Length: 32bit unsigned length of error string
0x0018	n	Error String: Error Length 8bit bytes, UTF-8 encoded, zero-terminated message describing the fault condition leading to the non-fatal error.

1.3.3 Unsupported Response Message

This message is sent by the RFMI server when it encountered a command message it does not support, either completely or partially (e.g. certain flags or optional features are not supported), i.e. the connection will stay up and the RFMI server is prepared to handle further messages, including retrying the current command message, e.g. without options.

Offset	Size	Description
0x0000	4	Select FMU Response: 0x70736E75 ('unsp')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 22 + Error Length
0x0010	4	Error Code: 32bit unsigned integer giving a more specific cause for the error. Default Value is 0x00000000, which indicates a generic error cause.
0x0014	4	Error Length: 32bit unsigned length of error string
0x0018	n	Error String: Error Length 8bit bytes, UTF-8 encoded, zero-terminated message describing the unsupported command/options leading to this message being sent.

1.3.4 NACK Response Message

This message is sent by the RFMI server when it encountered a command message it does understand but declines to take into account, e.g. when it declines a new frame definition. The connection will stay up and the RFMI server is prepared to handle further messages, including retrying the current command message with a different contents.

Offset	Size	Description
0x0000	4	Select FMU Response: 0x6B63616E ('nack')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 22 + Error Length
0x0010	4	Error Code: 32bit unsigned integer giving a more specific cause for the error. Default Value is 0x00000000, which indicates a generic error cause.
0x0014	4	Error Length: 32bit unsigned length of error string
0x0018	n	Error String: Error Length 8bit bytes, UTF-8 encoded, zero-terminated message describing the declined command/options leading to this message being sent.

2 Session Handling

All communication is initiated by the RFMI client opening a TCP/IP connection to the RFMI server on a pre-determined port, the default port being port 11711, and sending an RFMI Hello Command; the RFMI Server will respond with an RFMI Hello Response.

If at any time the connection is lost or otherwise interrupted, both sides will close the connection; it is suggested that the connection is closed in an abortive fashion, i.e. causing a TCP RST to be sent, instead of the normal FIN shutdown sequence.

The server can try to keep the session state alive for reconnection until an implementation-defined time out. If the time-out occurs, or the server otherwise determines not to keep the session alive, it will have to clean up all internal state, including FMU state in a safe way (e.g. potentially resetting the FMU, or otherwise causing its cleanup routines to run).

The RFMI client can try to reconnect to an existing session by reopening a connection to the RFMI server, and providing the session id in the RFMI Hello Command message. If possible, the RFMI server will acknowledge the reconnection by answering with an RFMI Hello Response message with the same session id. The session state and FMU state will then be unchanged. If the RFMI server cannot reconnect the session, it will respond with a different, new session id.

Once started up, a session is terminated either through a Session Shutdown Command message and corresponding response, or through connection loss and time-out.

2.1 Session Startup Phase

Upon opening a connection to the RFMI server, the connection is in the Session Startup Phase, requiring the completion of an RFMI Hello Command / RFMI Hello Response Message sequence in order to start or reconnect to a session, and transition to the FMU Selection Phase.

2.1.1 RFMI Hello Command Message

Upon opening the connection, the RFMI client will send a RFMI Hello Command Message:

Offset	Size	Description
0x0000	4	RFMI Protocol and Byte-Ordering Marker: 32bit Unsigned Integer 0x494D4652 in requested byte order, yielding 'RFMI' for little-endian, and 'IMFR' for big-endian.
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 24
0x0010	2	16bit Protocol Major Version Field: 0x0001 The major protocol version requested by the RFMI client. Major version changes in the protocol indicate changes that are not backward-compatible. The client should use the maximum version number it is prepared to handle.
0x0012	2	16bit Protocol Minor Version Field: 0x0000 The minor protocol version requested by the RFMI client. Minor version changes in the protocol indicate changes that are backward-compatible. The client should use the maximum version number it is prepared to handle.
0x0014	4	Session Restart Id: If the client is trying to reconnect to a pre-existing session, it can supply the session id as a 32bit unsigned integer in this field. If the client is not trying to reconnect, this field will contain the value 0x00000000.

2.1.2 RFMI Hello Response Message

If the RFMI server is able to process the connection request, the RFMI server will answer with a RFMI Hello Response Message:

Offset	Size	Description
0x0000	4	RFMI Protocol and Byte-Ordering Marker: 32bit Unsigned Integer 0x696D6672 in the byte order that is choosen by the server, yielding ’rfmi’ for little-endian and ’imfr’ for big-endian. NB: The choice of the server can take the requested byte-order of the client as supplied in the RFMI Hello Command Message, but is not required to do so, and may select a byte-order of its own choosing and/or capability.
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 24
0x0010	2	16bit Protocol Major Version Field: 0x0001 The major version number supported by the server, which can be equal to or less than the version number requested by the client. If the client is not prepared to handle a lower version number it must shutdown the session after receiving the server response with the non-supported lower version number.
0x0012	2	16bit Protocol Minor Version Field: 0x0000 The minor version number supported by the server, which can be equal, lower or higher than the version number requested by the client.
0x0014	4	Session Id: New session Id for this session. If the client requested a session restart in its RFMI Hello Command Message, and the FMI server was able to restart that session, the session id will be the one requested. In all other cases a new session id will be generated and returned. Note that the FMU server is free to reuse session ids after a session has been properly shut down or after a non-shut down session has timed-out. The server will ensure that if the client requested the resumption of a session and the server is not able to comply the returned new session id will be unequal to the requested session id.

2.2 Session Shutdown

The RFMI client can end a session at any time after the session startup by sending a session shutdown command message and closing its side of the connection. The RFMI server will respond with a session shutdown response message and also close its side of the connection.

2.2.1 Session Shutdown Command Message

This message can be sent at any time after session startup by the RFMI client in order to shutdown the session and end communication.

Offset	Size	Description
0x0000	4	Session Shutdown Command: 0x46464F53 ('SOFF')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

2.2.2 Session Shutdown Response Message

Offset	Size	Description
0x0000	4	Session Shutdown Response: 0x66666F73 ('soff')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

3 FMU Selection and Frame Setup Phases

3.1 FMU Selection Phase

After session startup has completed, if the session is not a reconnected session that already transitioned to the Frame Definition Phase prior to reconnection, the FMU selection sequence is initiated through the client sending either the List FMUs Command Message, which is then followed by a Select FMU Command Message, or directly sending a Select FMU Command Message.

The RFMI server will supply the set of variables of the FMU in its FMU Selected Response Message. Once the RFMI server has sent this reply, the session will transition to the Frame Setup Phase.

3.1.1 List FMUs Command Message

The RFMI client can optionally send a List FMUs Command Message, in order to list the FMUs available in the RFMI server:

Offset	Size	Description
0x0000	4	List FMUs Command: 0x554D464C ('LFMU')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

3.1.2 List FMUs Response Message

The RFMI server will reply with an List FMUs Response Message:

Offset	Size	Description
0x0000	4	List FMUs Response: 0x756D666C ('lfmu')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 20 + FMU Descriptions Length
0x0010	4	32bit Unsigned Int: Number of FMU Descriptions in Response
0x0014	...	n FMU Descriptions, each with the following fields:
+0x00	2	FMI Major Version: 16bit Unsigned Integer
+0x02	2	FMI Minor Version: 16bit Unsigned Integer
+0x04	2	FMU Kind: 16bit Unsigned, 0x0000 = Co-Simulation
+0x06	2	FMU Capabilities: 16bit Unsigned Integer
+0x08	4	Name Length: 32bit unsigned length of name string
+0x0C	x	FMU Name: Name Length 8bit bytes, UTF-8 encoded, zero-terminated, padded with zero bytes to next 32bit boundary, so that next FMU Description starts 32bit aligned.

The FMU Kind field indicates the type of FMU, with 0x0000 being defined for Co-Simulation FMUs. Currently no other kind is defined, however support for Model-Exchange and potentially other kinds of FMUs (hybrid co-simulation, etc.) is to be added in the future.

The FMU Capabilities field will indicate the advanced capabilities that the FMU supports. Currently this field is defined to be 0, however expect field values for support of event-handling, etc. to be defined in the future.

3.1.3 Select FMU Command Message

Either directly or after having received the List FMUs Response Message, the client will select the FMU to be used in the session through the Select FMU Command Message:

Offset	Size	Description
0x0000	4	Select FMU Command: 0x4C455346 ('FSEL')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 20 + FMU Name Length
0x0010	4	Name Length: 32bit unsigned length of name string
0x0014	n	FMU Name: Name Length 8bit bytes, UTF-8 encoded, zero-terminated, padded with zero bytes to next 32bit boundary.

3.1.4 Select FMU Response Message

The RFMI server responds with a Select FMU Response Message, which will contain additional information on the FMU, including a list of all variables of the FMU.

Offset	Size	Description
0x0000	4	Select FMU Response: 0x6C657366 ('fsel')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 28 + FMU Name Length + Variables
0x0010	4	Name Length: 32bit unsigned length of name string
0x0014	n	FMU Name: Name Length 8bit bytes, UTF-8 encoded, zero-terminated, with zero-byte padding to 64bit alignment ...
0x0014 + n	8	64bit Unsigned Int: Number of Variable Definitions in Response
0x001C + n	...	m Variable Definitions, each with the following fields:
+0x00	2	Variable Kind: 16bit unsigned integer field giving the kind
+0x02	2	Variable Type: 16bit unsigned integer field giving the FMU basic data type of the variable.
+0x04	4	Variable Value Reference: 32bit unsigned integer id of the variable unique within its basic data type.
+0x08	4	Name Length: 32bit unsigned length of variable name string
+0x0C	n	Variable Name: Name Length 8bit bytes, UTF-8 encoded, zero-terminated, with zero-byte padding to 32bit alignment

The variable kind field indicates the kind of variable, e.g. input, output, parameter, etc. It is currently defined to consist of two 8bit sub-fields, indicating the variable variability field in the least significant byte, and the causality in the most significant byte, with the following definitions:

1. Variable Causality

Value	FMI Version	Description
0x00	1.0 + 2.0	Input, variable to be set by connections from the outside of the FMU, i.e. proper inputs to the FMU.
0x01	1.0 + 2.0	Output, variable can be used as an output of the FMU in connections to other systems.
0x02	1.0	Internal, variable is not to be used in connections, but can be changed during initialization, e.g. this is often used for parameters in FMI 1.0.
0x03	1.0	None, variable has no effect on the FMU model itself but is used e.g. for debug flags, etc.
0x04	2.0	Local is the FMI 2.0 equivalent of internal used for non-parameters, see Parameter/Calculated Parameter
0x05	2.0	Parameter is used in FMI 2.0 for parameters that are to be set by the outside.
0x06	2.0	Calculated Parameter is used in FMI 2.0 for those parameters that are calculated internally in the FMU based on other parameters and internal calculations.
0x07	2.0	Independent variable, i.e the time variable if it is present in the exposed variables.

2. Variable Variability

Value	FMI Version	Description
0x00	1.0 + 2.0	Constant, value never changes
0x01	1.0	Parameter, value only changes during initialization
0x02	1.0 + 2.0	Discrete, value changes only at event time instants, i.e. discrete variables of types boolean, integer, enumeration, string.
0x03	1.0 + 2.0	Continuous, value changes at any time, i.e. real variables.
0x04	2.0	Fixed, FMI 2.0 equivalent of Parameter, but renamed and slightly changed in semantics, since 2.0 also has tunable parameters that can change during the simulation.
0x05	2.0	Tunable, intended for tunable parameters that can change during the simulation, causing events in the process.

3. Useful combinations for FMI 1.0

Value	Causality	Variability	Description
0x??00	Any	Constant	A constant value, can be treated identically regardless of causality, i.e. 0x0100, 0x0200, 0x0300 and 0x0400
0x0001	Input	Parameter	An input parameter, treat identically to 0x0201 which is also an input parameter.
0x0201	Internal	Parameter	
0x0002	Input	Discrete	Discrete Input
0x0003	Input	Continuous	Continuous Input
0x0101	Output	Parameter	Treat as output, but value potentially never changes during simulation.
0x0102	Output	Discrete	Discrete Output
0x0103	Output	Continuous	Continuous Output
0x0202	Internal	Discrete	Discrete Internal Variable
0x0203	Internal	Continuous	Continuous Internal Variable
0x0301	None	Parameter	Treat as debug information, that can be read and potentially set as indicated by the variability. Best to ignore if not certain.
0x0302	None	Discrete	
0x0303	None	Continuous	

The RFMI Server will treat variables with kind 0x0002 and 0x0003 as inputs and variables with kind 0x0100, 0x0101, 0x0102 and 0x0103 as output variables for default frame building.

4. Useful combinations for FMI 2.0

   See FMI 2.0 standard.

3.2 Frame Setup Phase

In this phase the communication frames to be used later on are setup or inquired. Optionally the RFMI client can query the RFMI server for the contents of the modelDescription.xml file through the Get FMU Model Description Command.

The RFMI client can inquire the set of default frame definitions from the RFMI server with a List Defined Frames Command, and/or it can optionally chose different/additional frame definitions, which it will define with Define Frame Command Messages; if the RFMI server can support these definitions it will respond with Define Frame Response Messages, otherwise it will decline the definitions with a NACK Response Message.

After the frames are defined, the session is completely set-up and the connection can enter the Simulation Initialization Phase, through a Begin Initialization Command message sent by the RFMI client.

3.2.1 Get FMU Model Description Command message

This message requests the current FMU's XML Model Description from the RFMI server. This information can be used to determine default values, minimum/maximum value ranges and other information on the FMU.

Offset	Size	Description
0x0000	4	Get FMU Model Description Command: 0x4C4D5846 ('FXML')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

3.2.2 Get FMU Model Description Response message

This message is the response the RFMI server sends to an RFMI client Get FMU Model Description Command message. It just contains the raw XML model Description file of the selected FMU.

Offset	Size	Description
0x0000	4	Get FMU Model Description Response: 0x6C6D7866 ('fxml')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16 + XML Model Decription String
0x0010	...	XML Model Description of the selected FMU as raw XML, with zero-termination.

3.2.3 List Defined Frames Command message

This message requests a list of currently defined frame definitions from the RFMI server. These include the frames defined by default, as well as any frames that were defined by the RFMI client through Define Frame Messages.

Offset	Size	Description
0x0000	4	List Defined Frames Command: 0x4D52464C ('LFRM')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

3.2.4 List Defined Frames Response message

This message is the response the RFMI server sends to an RFMI client List Defined Frames Command message.

Offset	Size	Description
0x0000	4	List Defined Frames Response: 0x6D72666C ('lfrm')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 20 + Frame Definitions
0x0010	4	32bit Unsigned Int: Number of Frame Definitions in Response
0x0014	...	n Frame Definitions, see below

1. Frame Definition

   Each frame definition consists of a frame identifier and a set of sub-frame definitions:

Offset	Size	Description
0x0000	4	Frame Identifier: 32bit unsigned integer Note that Frame Identifier 0x00000000 is reserved for use as an empty frame, and 0x10000000 is reserved for use as a dynamic frame identifier, see below.
0x0004	4	Frame Sub-Frames: 32bit unsigned integer count of Sub-Frames in the Frame.
0x0008	...	n Sub-Frame Definitions

2. Sub-Frame Definition

Each sub-frame definition consists of a sub-frame field type and a set of sub-frame entries:

Offset	Size	Description
0x0000	2	Sub-Frame Fields Type
0x0002	2	Reserved
0x0004	4	Sub-Frame Entry Count: 32bit unsigned integer count of Sub-Frame Entries in the Sub-Frame.
0x0008	...	n Sub-Frame Entries, each with the following fields:
+0x00	4	Value Reference: 32bit unsigned integer identifying the underlying variable.

3. Pre-Defined Frames

   By default, the following frames are defined for each FMU:

Frame Id	Description
0x00000000	Empty Frame: This frame is defined as the empty frame, which can be used in commands where an empty frame is needed. It is protected against redefinition.
0x00000001	Standard Input Frame: All Continuous and Discrete Input Variables
0x00000002	Standard Output Frame: All Continuous and Discrete Output Variables

Frame Id 0x10000000 is reserved for dynamic frames, where the frame definition is sent inline in the same message, preceding the frame data (hence it is not allowed to permanently define frames with Frame Id 0x10000000 through DFRM). In that case, the frame data will follow directly after the frame definition, but aligned to an 8 byte boundary, so that frame data alignment works out correctly. I.e. if the frame definition does not end at an 8 byte boundary, zero padding is added until it does.

Frame Ids 0x80000000 and above are reserved for new frame definitions by the RFMI client, i.e. no pre-defined frames exist in this region.

3.2.5 Define Frame Command Message

This message can be sent by the RFMI client to the RFMI server in order to define a new frame or redefine an existing frame.

Offset	Size	Description
0x0000	4	Define Frame Command: 0x4D524644 ('DFRM')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16 + Frame Definition
0x0010	...	Frame Definition, see above

3.2.6 Define Frame Response Message

If defining the frame succeeds, then the define frame response message is sent:

Offset	Size	Description
0x0000	4	Define Frame Response: 0x6D726664 ('dfrm')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

If defining the frame is not possible for some reason, the RFMI server will respond with a nack response message.

4 Initialization and Simulation Phases

4.1 Initialization Phase

The Initialization Phase is started by the RFMI client sending the Begin Initialization Command message, which the RFMI server will acknowledge with a Begin Initialization Response message. Once Initialization Phase has been entered, the RFMI client can get and set parameter variables and input variable start values through the Get Variables Command and Set Variables Command messages. Once the client is finished setting parameters, it can transition to the Simulation Phase through sending the Start Simulation Command message.

4.1.1 Begin Initialization Command Message

The sending of this command by the RFMI client to the RFMI server initiates transition from the Frame Setup Phase to the Initialization Phase.

Offset	Size	Description
0x0000	4	Begin Initialization Command: 0x54494E49 ('INIT')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.1.2 Begin Initialization Response Message

The RFMI server answers with a begin initialization response message:

Offset	Size	Description
0x0000	4	Begin Initialization Response: 0x74696E69 ('init')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.1.3 Get Variables Command Message

This message allows the RFMI client to get current values of variables using either permanently defined frames or dynamic frames (see above).

Offset	Size	Description
0x0000	4	Get Variables Command: 0x56544547 ('GETV')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 24 + optional sub-frame definitions
0x0010	4	Output Frame Identifier: 32bit unsigned integer indicating the frame identifier for output values that the server should supply in its response to this message. Can be 0x00000000 if no output values should being provided. If this is 0x10000000 then this word and all following words constitute a a dynamic frame definition, i.e. the frame definition starts at offset 0x0010.
0x0014	4	For Dynamic Frames: Frame Sub-Frames: 32bit unsigned integer count of Sub-Frames in the Frame. Otherwise: Reserved
0x0018	...	For Dynamic Frames Only: n sub-frame definitions.

4.1.4 Get Variables Response Message

The RFMI server responds with a Get Variables Response Message:

Offset	Size	Description
0x0000	4	Get Variables Response: 0x76746567 ('getv')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 24 + Values
0x0010	4	Output Frame Identifier: 32bit unsigned integer indicating the frame identifier for output values that the server is supplying in its response. Can be 0x00000000 if no output values were requested.
0x0014	4	Reserved
0x0018	...	Output Frame Values Output Frame Values are layed out as described in the subsection Data Layout in Frames of the section General Information at the beginning of this document.

4.1.5 Set Variables Command Message

This message allows the RFMI client to set current values of variables using either permanently defined frames or dynamic frames (see above).

Offset	Size	Description
0x0000	4	Set Variables Command: 0x56544553 ('SETV')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 24 + optional sub-frame definitions + values
0x0010	4	Input Frame Identifier: 32bit unsigned integer indicating the frame identifier for input values being provided with this message. Can be 0x00000000 if no input values are being provided. If this is 0x10000000 then this word and following words constitute a dynamic frame definition, i.e. the frame definition starts at offset 0x0010.
0x0014	4	For Dynamic Frames: Frame Sub-Frames: 32bit unsigned integer count of Sub-Frames in the Frame. Otherwise: Reserved
0x0018	...	For Dynamic Frames Only: n sub-frame definitions.
...	...	For Dynamic Frames: After sub-frame definitions, starting on next 8 byte boundary. Otherwise: Starting at Offset 0x0018. Input Frame Values Input Frame Values are layed out as described in the subsection Data Layout in Frames of the section General Information at the beginning of this document.

4.1.6 Set Variables Response Message

The RFMI server responds with a Set Variables Response Message:

Offset	Size	Description
0x0000	4	Set Variables Response: 0x76746573 ('setv')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.2 Simulation Phase

For both FMI 1.0 and 2.0 the Simulation Phase can be started by the RFMI client by sending a Start Simulation Command Message to the RFMI server, which is answered by a Start Simulation Response Message. After this, the RFMI client can use Simulation Step Commands, optionally combined with Set Variables and Get Variables Commands in order to drive the simulation forward.

For FMI 2.0, the Simulation Phase can also be entered through a sequence of Setup Experiment, Enter Initialization Mode and Exit Initialization Mode Command Messages to support the newly enhanced simulation startup state machine offered by FMI 2.0.

Once the RFMI client is finished with a simulation, it can either reset the simulation back to Initialization Phase through the Simulation Reset Command (if such resetting is supported by the underlying FMU), or tear down the simulation (including the FMU instance) with the Simulation Shutdown Command, which will return the session to the Frame Setup Phase, where the session can either be ended through the Session Shutdown Command, or continued normally.

4.2.1 Begin Simulation Command Message

The sending of this command by the RFMI client to the RFMI server initiates transition from the Initialization Phase to the Simulation Phase.

For FMI 2.0 this is achieved through a sequence of calls to setup the experiment, enter and exit the initialization mode, which is directly equivalent to issuing the Setup Experiment, Enter Initialization Mode and Exit Initialization Mode Command Messages in sequence without any intervening messages. Clients that want/need more fine-grained control, should issue the corresponding messages individually.

Offset	Size	Description
0x0000	4	Begin Simulation Command: 0x534D4953 ('SIMS')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 36
0x0010	8	Simulation Start Time: Double Precision Floating-Point value indicating the simulation start time to give to the FMU.
0x0018	8	Simulation Stop Time: Double Precision Floating-Point value indicating the simulation stop time to give to the FMU, if a stop time is known before-hand.
0x0020	1	Simulation Stop Time Valid: Boolean indicating whether the simulation stop time value is provided and valid.
0x0021	3	Reserved

4.2.2 Begin Simulation Response Message

The RFMI server answers with a Begin Simulation Response Message:

Offset	Size	Description
0x0000	4	Begin Simulation Response: 0x736D6973 ('sims')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.2.3 Setup Experiment Command Message

For FMI 2.0 FMUs only, this Command Message provides an alternative way to setup the experiment data; if it is used, it must be issued prior to an Enter Initialization Mode Command Message. It transitions the FMU to the FMU Setup Experiment Phase, where only Set Value Command Messages and the Enter Initialization Mode Command Message are valid.

Offset	Size	Description
0x0000	4	Setup Experiment Command: 0x50584553 ('SEXP')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 36
0x0010	8	Simulation Start Time: Double Precision Floating-Point value indicating the simulation start time to give to the FMU.
0x0018	8	Simulation Stop Time: Double Precision Floating-Point value indicating the simulation stop time to give to the FMU, if a stop time is known before-hand.
0x0020	1	Simulation Stop Time Valid: Boolean indicating whether the simulation stop time value is provided and valid.
0x0021	3	Reserved

4.2.4 Setup Experiment Response Message

The RFMI server answers with a Setup Experiment Response Message:

Offset	Size	Description
0x0000	4	Setup Experiment Response: 0x70786573 ('sexp')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.2.5 Enter Initialization Mode Command Message

For FMI 2.0 FMUs only, this command is issued by the RFMI Client to trigger the entering of initialization mode for the FMU, which is valid after a Setup Experiment Command Message has already been sent. It transitions the FMU to the Initialization Mode Phase, where only Get Value, Set Value and Exit Intitialization Mode Command Messages are valid.

Offset	Size	Description
0x0000	4	Enter Initialization Mode Command: 0x494E4945 ('EINI')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.2.6 Enter Initialization Mode Response Message

The RFMI server answers with an Enter Initialization Mode Response Message:

Offset	Size	Description
0x0000	4	Enter Initialization Mode Response: 0x696E6965 ('eini')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.2.7 Exit Initialization Mode Command Message

For FMI 2.0 FMUs only, this command is issued by the RFMI Client to trigger the exiting of initialization mode and progression to simulation mode for the FMU, which is valid after a preceding Enter Initialization Mode Command Message has already been sent. It transfers the FMU to the Simulation Phase.

Offset	Size	Description
0x0000	4	Exit Initialization Mode Command: 0x494E4958 ('XINI')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.2.8 Exit Initialization Mode Response Message

The RFMI server answers with an Exit Initialization Mode Response Message:

Offset	Size	Description
0x0000	4	Exit Initialization Mode Response: 0x696E6978 ('xini')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.2.9 Simulation Step Command Message

This command (optionally) provides new input values to the FMU, steps the FMU to the current simulation time + step size, and (optionally) requests the values of output values in one go. If used properly the simulation can be driven solely by stringing Simulation Step Command messages together.

The RFMI server will answer with a Simulation Step Response message, if the simulation step finished sucessfully, or any of the error responses, if the step failed for some reason.

Offset	Size	Description
0x0000	4	Simulation Step Command: 0x50455453 ('STEP')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 48 + Input Frame Values
0x0010	8	Current Simulation Time: Double Precision Floating-Point indicating the current simulation time. This value is transmitted to ensure that the current simulation time between client and server is identical.
0x0018	8	Step Size: Double Precision Floating-Point indicating the step size the simulation is supposed to calculate to, i.e. the FMU is supposed to calculate to current simulation time + step size. Note that step size can be 0 during event iteration, if the underlying FMU supports this.
0x0020	1	New Step: Boolean indicating that the prior step has been accepted, and a new step is being started.
0x0021	7	Reserved
0x0028	4	Input Frame Identifier: 32bit unsigned integer indicating the frame identifier for input values being provided with this message. Can be 0x00000000 if no input values are being provided.
0x002C	4	Output Frame Identifier: 32bit unsigned integer indicating the frame identifier for output values that the server should supply in its response to this message. Can be 0x00000000 if no output values should being provided.
0x0030	...	Input Frame Values Input Frame Values are layed out as described in the subsection Data Layout in Frames of the section General Information at the beginning of this document.

4.2.10 Simulation Step Response

The RFMI server responds to the Simulation Step Command message with a Simulation Step Response message if the simulation step completed successfully.

Offset	Size	Description
0x0000	4	Simulation Step Response: 0x70657473 ('step')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 32 + Output Frame Values
0x0010	8	Current Simulation Time: Double Precision Floating-Point indicating the current simulation time. This value is transmitted to ensure that the current simulation time between client and server is identical. Note that this time is identical to the IEEE754 floating-point sum of the current simulation time and step size values from the Simulation Step Command message.
0x0018	4	Output Frame Identifier: 32bit unsigned integer indicating the frame identifier for output values that the server is supplying in its response. Can be 0x00000000 if no output values were requested.
0x001C	4	Reserved
0x0020	...	Output Frame Values Output Frame Values are layed out as described in the subsection Data Layout in Frames of the section General Information at the beginning of this document.

4.3 Simulation Shutdown

4.3.1 Simulation Shutdown Command Message

The sending of this command by the RFMI client to the RFMI server initiates transition from the Simulation Phase back to the Frame Setup Phase.

Offset	Size	Description
0x0000	4	Simulation Shutdown Command: 0x4E574453 ('SDWN')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.3.2 Simulation Shutdown Response Message

Offset	Size	Description
0x0000	4	Simulation Shutdown Response: 0x6E776473 ('sdwn')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.3.3 Simulation Reset Command Message

The sending of this command by the RFMI client to the RFMI server initiates transition from the Simulation Phase back to the Initialization Phase.

Offset	Size	Description
0x0000	4	Simulation Reset Command: 0x54535253 ('SRST')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

4.3.4 Simulation Reset Response Message

Offset	Size	Description
0x0000	4	Simulation Reset Response: 0x74737273 ('srst')
0x0004	4	Message Flags: 0x00000000
0x0008	8	Message Length: 16

5 Example Time Lines

5.1 Basic Simulation Run

RFMI Client	RFMI Server	Description
RFMI		RFMI Hello Command
	rfmi	RFMI Hello Response
LFMU		List FMUs Command
	lfmu	List FMUs Response
FSEL		Select FMU Command
	fsel	Select FMU Response
LFRM		List Defined Frames Command
	lfrm	List Defined Frames Response
INIT		Begin Initialization Command
	init	Begin Initialization Response
SETV		Set Variables Command
	setv	Set Variables Response
SIMS		Begin Simulation Command
	sims	Begin Simulation Response
STEP		Simulation Step Command
	step	Simulation Step Response
STEP		Simulation Step Command
	step	Simulation Step Response
...		(once for each simulation step until the
	...	simulation is complete)
SDWN		Simulation Shutdown Command
	sdwn	Simulation Shutdown Response
SOFF		Session Shutdown Command
	soff	Session Shutdown Response