SpaceWire RMAP GUI

User Guide

Takayuki Yuasa

Japan Aerospace Exploration Agency, Institute of Space and Astronautical Science

yuasa _at_mark_ astro.isas.jaxa.jp

January 06, 2012

Contents

1 Overview of SpaceWire RMAP GUI

SpaceWire RMAP GUI is an easy-to-use graphical user interface front end for SpaceWire-to-GigabitEther. The software runs on Mac OS X, and is capable of sending and receiving raw SpaceWire packets, and performing RMAP[1] read/write transactions using a SpaceWire interface connected to your Mac. SpaceWire-to-GigabitEther is a default SpaceWire interface supported by SpaceWire RMAP GUI. In addition to these, RMAP packet construction/interpretation is provided. The software is developed as an open-source project, and contributions and/or feedbacks from users or developers are highly appreciated.

Several types of information used in SpaceWire RMAP GUI, such as RMAP target node information, memory object on an RMAP target node, can be defined using XML-like file. By loading definition files, users do not need to input logical address, target SpaceWire address, memory address, length, or other properties every RMAP access.

1.1 Screenshot

As shown in Figure 1, the main window consists of three tabs which displays SpaceWire, RMAP, and RMAP packet utility functionalities.

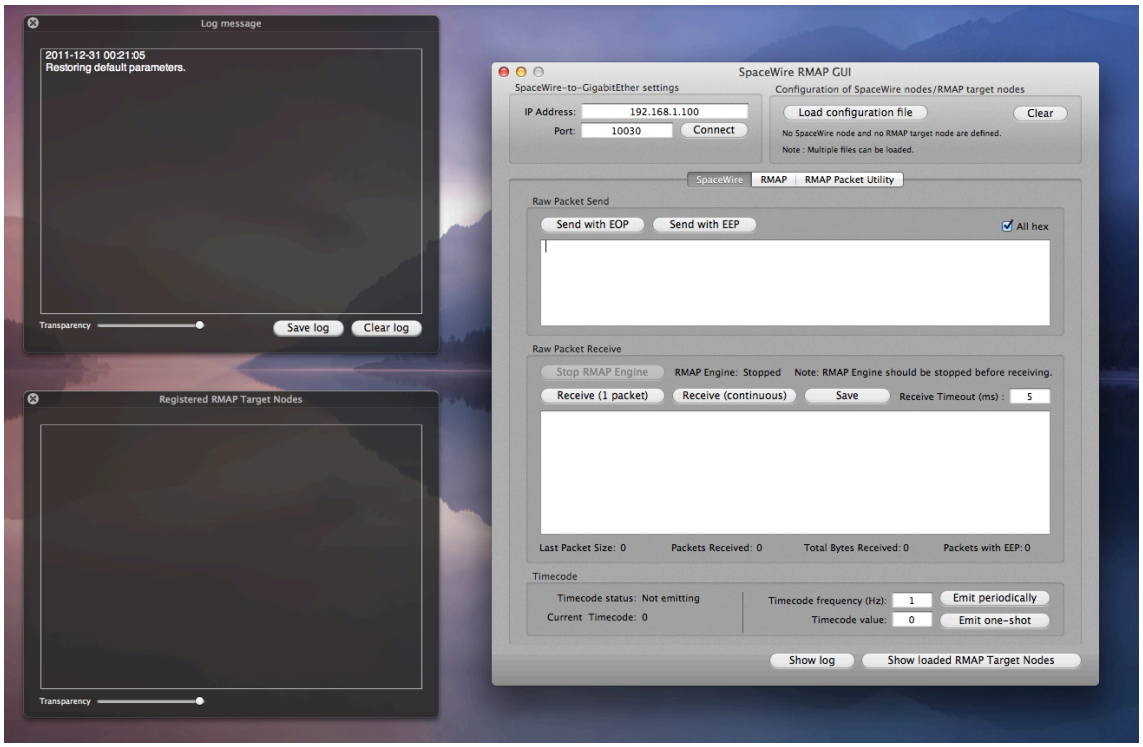

Figure 1: Screenshot of SpaceWire RMAP GUI.

1.2 Dependence

SpaceWire RMAP GUI uses libraries listed below:

- SpaceWire/RMAP Library[2]
- CxxUtilities[3]
- XMLUtility by Soki Sakurai from The University of Tokyo[4]
- xerces-c++ by the Apache project[5]

These libraries are statically linked to the software, and therefore, users are not required to install individual libraries separately.

[1] Remote Memory Access Protocol.
[2] https://github.com/yuasatakayuki/SpaceWireRMAPLibrary
[3] https://github.com/sakuraisoki/XMLUtilities/
[4] https://github.com/sakuraisoki/XMLUtilities/
[5] http://xerces.apache.org/xerces-c/

1.3 Download, install, and updates

SpaceWire RMAP GUI can be downloaded from the Open-source SpaceWire project[6]. Note that the software is distributed for free assuming that this is useful for some users without any warranty. Unzip the archive, and move the extracted "SpaceWire RMAP GUI" folder to the "Applications" folder. If you use the software frequently, register it to the Dock by dragging-and-dropping the icon to the Dock. To uninstall the software, move the "SpaceWire RMAP GUI" folder to the Trash.

Hopefully, SpaceWire RMAP GUI will be updated based on feedbacks from users, and new versions will be available at the same web site. At this moment, there is no auto update function in the software, and therefore, please check the web site for updates occasionally.

2 About this user guide

The user guide is provided *as is* expecting that this is somewhat useful for users to use the software. This is a voluntary mission, and therefore, kind help is always welcome. It is greatly appreciated to make contributions by feed-backing comments, revising documents, and so on.

2.1 Latest information and feedback

Latest information on SpaceWire RMAP GUI and SpaceWire-to-GigabitEther can be found at the open-source SpaceWire project website [6].

2.2 References

Documents listed below can be a nice reference when better understanding SpaceWire RMAP GUI. Some of the documents can be obtained from the open-source SpaceWire project website[6].

- SpaceWire/RMAP Library User Guide
- SpaceWire-to-GigabitEther User Guide
- ECSS-E-ST-50-12C - "SpaceWire - Links, nodes, routers and networks" by ECSS
- ECSS-E-ST-50-52C "SpaceWire - Remote memory access protocol" by ECSS

2.3 Revisions

- 2012-01-01 Takayuki Yuasa. First release.

3 Structure of SpaceWire RMAP GUI

3.1 The main window

SpaceWire RMAP GUI displays single main window when start up. The window contains three tabs (or panels), and each tab collects text fields and buttons which are necessary for SpaceWire, RMAP, and RMAP Packet Utility functions.

The main window is divided into four sections; the SpaceWire-to-GigabitEther setting section (top left; see Section 4.1), the configuration file section (top right; see Section 4.2), the main tab (middle), and the log message section (bottom). When the main window is closed, SpaceWire RMAP GUI quits. If SpaceWire-to-GigabitEther is connected via TCP/IP when quitting, SpaceWire RMAP GUI tries to close the connection. Settings (values contained in each input fields) are saved to the preference file, and loaded again at the start up.

3.2 SpaceWire tab

Figure 3 shows the SpaceWire tab. Receive timeout can be manually set. The tab allows users to send and receive raw SpaceWire packets. Periodic or ordered timecode emission is managed from this tab as well.

[6]The open-source SpaceWire project: https://galaxy.astro.isas.jaxa.jp/ yuasa/SpaceWire

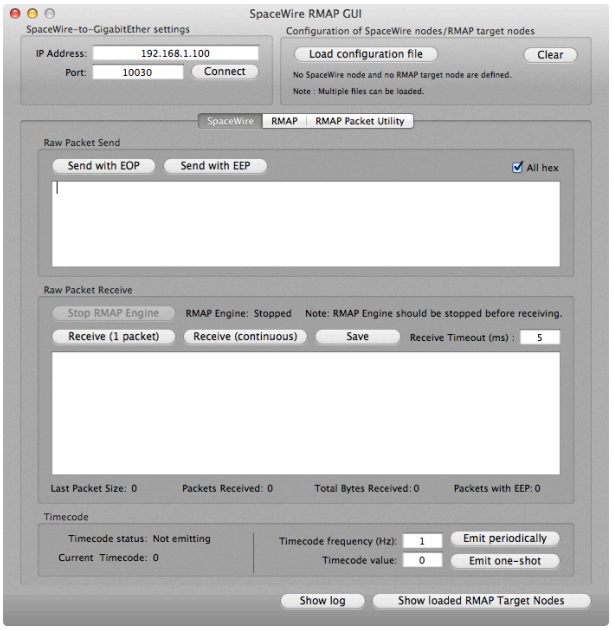

Figure 2: Screenshot of the SpaceWire tab.

3.3 RMAP tab

The RMAP tab provides RMAP read/write initiator functions as shown in Figure **??**. RMAP target node information can be set manually or semi-automatically using XML-like configuration files.

3.4 RMAP Packet Utility tab

The RMAP Packet Utility tab, shown in Figure 4, interprets and constructs RMAP packets. A correct header or data CRC value is displayed when interpreting a packet with an invalid CRC.

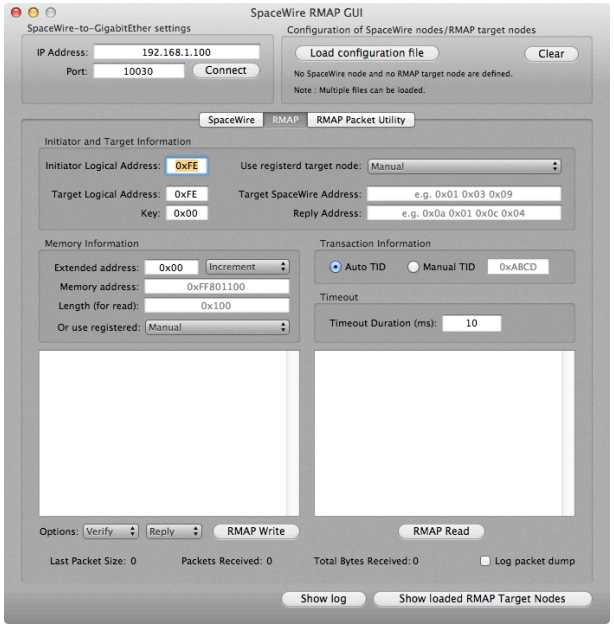

Figure 3: Screenshot of the RMAP tab.

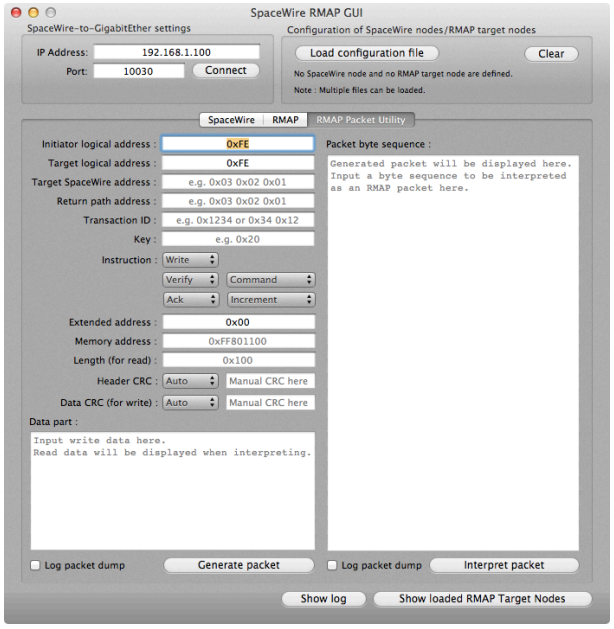

Figure 4: Screenshot of the RMAP Packet Utility tab.

3.5 Log message window

The software displays log messages when a user executes each function. Log messages are tagged with real time, and colored when there happens any error or unexpected event. Users can open the log window, as shown in Figure 5 by clicking the ”Show log” button in the bottom of the main window.

3.6 Registered RMAP Target Nodes window

When loading a configuration file, information of RMAP Target Nodes such as target logical address, key, and so on, are registered, and available with identifiers, or name, of the target. The Registered RMAP Target Nodes window lists registered RMAP Target Nodes and their properties including defined those of memory objects. Figure 6 shows an example of the window when loading a sample XML file for the SpaceWire-to-GigabitEther configuration port.

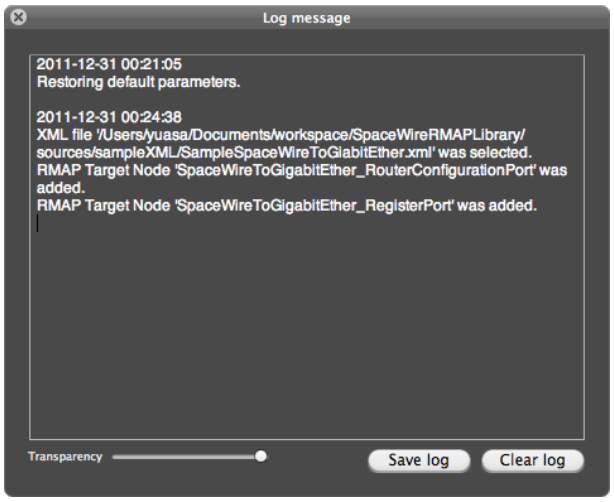

Figure 5: Screenshot of the log window.

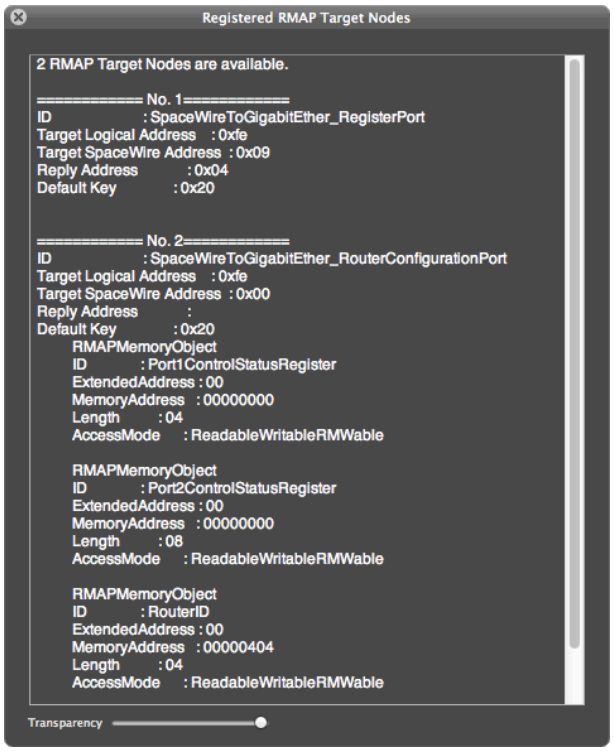

Figure 6: Screenshot of the Registered RMAP Target Nodes window.

3.7 Typical procedure

When analyzing/creating RMAP packets in the RMAP Packet Utility tab, no connection to SpaceWire-to-GigabitEther is necessary. When sending/receiving SpaceWire packets or initiating RMAP read/write, follow the procedure below:

1. Set the IP address and the TCP port number.
2. Click the ”Connect” button.
3. Load XML configuration files (to register RMAP Target Node information) if needed.
4. Send/receive SpaceWire packets (from the SpaceWire tab),
   or initiate RMAP read/write (from the RMAP tab).

3.8 Notation of byte sequence

In SpaceWire RMAP GUI, users are required to input values (e.g. RMAP memory address and transaction ID) and byte sequence (e.g. raw SpaceWire packet and SpaceWire address).

The software interprets a value as hex when a leading ”0x” is attached to numbers (and a-f). Multi-byte numbers can be expressed as for example ”0x1234abcd”, and this example is interpreted as a byte sequence ”0x12 0x34 0xab 0xcd”. When converting a multi-byte number with an odd number of digits e.g. ”0x123” to a byte sequence, one leading 0 is padded, resulting ”0x0123” == ”0x01 0x23”.

Numbers with no ”0x” prefix are basically treated as decimal, but when the ”All hex” option which is available in a part of input fields is enabled, ”0x” is automatically put before the numbers. For example, ”0x45 12 ab 0x88 f0” is converted to a byte sequence ”0x45 0x12 0xab 0x88” when the ”All hex” option is enabled.

4 Settings

4.1 TCP/IP connection to SpaceWire-to-GigabitEther

SpaceWire RMAP GUI should be used with SpaceWire-to-GigabitEther which provides a physical connection to SpaceWire network via TCP/IP over GigabitEthernet. The device can be purchased from Shimafuji Electric as a paid product, or an open-source (free) version on commercial FPGA board (ZestET1 by OrangeTree) is available from the open-source SpaceWire project.

Users can set an IP address (or URL) and utilized port number of SpaceWire-to-GigabitEther in the section on top-left of the main window as shown in Figure 7. When the ”Connect” button is clicked, the software tries to connect to the specified SpaceWire-to-GigabitEther, and then SpaceWire packet sending/receiving and RMAP initiator functions become available if a connection is successfully established. If timeout occurs, an error dialog will be displayed.

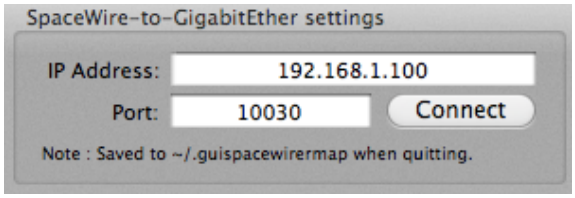

Figure 7: Input files for the IP address and the TCP port number of SpaceWire-to-GigabitEther.

Note: Port number selects SpaceWire port of the internal SpaceWire router (when available on your SpaceWire-to-GigabitEther device). The default value 10030 may correspond to port 6 in Shimafuji's SpaceWire-to-GigabitEther, and port 1 in open-source SpaceWire-to-GigabitEther (which does not implement an internal router, and the port is directly connected to the physical connector on the board). For details, please refer to user guide of your SpaceWire-to-GigabitEther.

4.2 RMAP Target Node definition using XML-like files

In SpaceWire RMAP GUI, an XML-like text file or files is/are used to load RMAP target node information such as a logical address, target SpaceWire address and reply address, and memory object (or register) information on the RMAP target node such as memory address, access length, access modes. This allows users to deal with a large network which consists of large number of RMAP target nodes by avoiding manual reconfiguration of many input fields in the RMAP tab in every RMAP access.

The example shown below tells typical syntax used to define an RMAP Target Node and memory objects on it. See SpaceWire/RMAPLibrary User Guide for details of the format.

Listing 1: An example of an XML-like configuration file.

```
1  <root>
2
3  <RMAPTargetNode id="SpaceWireToGigabitEther_RouterConfigurationPort">
4          <TargetLogicalAddress>0xFE</TargetLogicalAddress>
5          <TargetSpaceWireAddress>0x00</TargetSpaceWireAddress>
6          <ReplyAddress></ReplyAddress>
7          <Key>0x20</Key>
8
9          <RMAPMemoryObject id="Port1ControlStatusRegister">
10                 <ExtendedAddress>0x00</ExtendedAddress>
```

```
                <Address>0x0000</Address>
                <Length>0x04</Length>
                <Key>0x20</Key>
        </RMAPMemoryObject>

        <RMAPMemoryObject id="Port2ControlStatusRegister">
                <ExtendedAddress>0x00</ExtendedAddress>
                <Address>0x0000</Address>
                <Length>0x08</Length>
                <Key>0x20</Key>
        </RMAPMemoryObject>

        <RMAPMemoryObject id="RouterID">
                <ExtendedAddress>0x00</ExtendedAddress>
                <Address>0x404</Address>
                <Length>0x04</Length>
                <Key>0x20</Key>
        </RMAPMemoryObject>
</RMAPTargetNode>

<RMAPTargetNode id="SpaceWireToGigabitEther_RegisterPort">
        <TargetLogicalAddress>0xFE</TargetLogicalAddress>
        <TargetSpaceWireAddress>0x09</TargetSpaceWireAddress>
        <ReplyAddress>0x04</ReplyAddress>
        <Key>0x20</Key>
</RMAPTargetNode>

</root>
```

By loading configuration files from the ”Configuration” section in the main window which is shown in Figure 8, defined RMAP Target Nodes will be listed in the pull-down menu in ”Initiator and Target Information” section of the RMAP tab. When one of defined RMAP Target Node is selected, its information is automatically set to relevant input fields, and correspondingly memory objects will be displayed in the register name pull-down menu in the ”Memory Information” section. When selecting an entry from the register name pull-down menu, memory address and length will be automatically set.

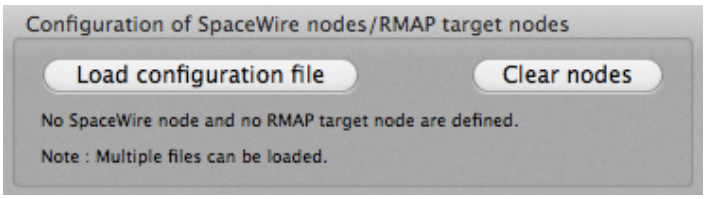

Figure 8: Configuration file section.

The above example results with RMAP Target Nodes named ”SpaceWireToGigabitEther_RouterConfigurationPort” and ”SpaceWireToGigabitEther_RegisterPort”, and memory objects on ”SpaceWireToGigabitEther_RouterConfigurationPort” named ”Port1ControlStatusRegister”, ”Port2ControlStatusRegister”, and ”RouterID” each having 4-byte fields. Figure 9 shows a screenshot of the RMAP Target Node pull-down menu and the register pull-down menu in the RMAP tab after loading the configuration file.

5 SpaceWire functions

5.1 Sending packets

To send packets, input byte sequence in the Raw Packet Send section on the SpaceWire window. Figure 10 shows an example of data input. See §3.8 for details of number input options. Input byte sequence is sent with an end-of-packet marker, EOP or EEP, selected via ”Send with EOP” or ”Send with EEP”.

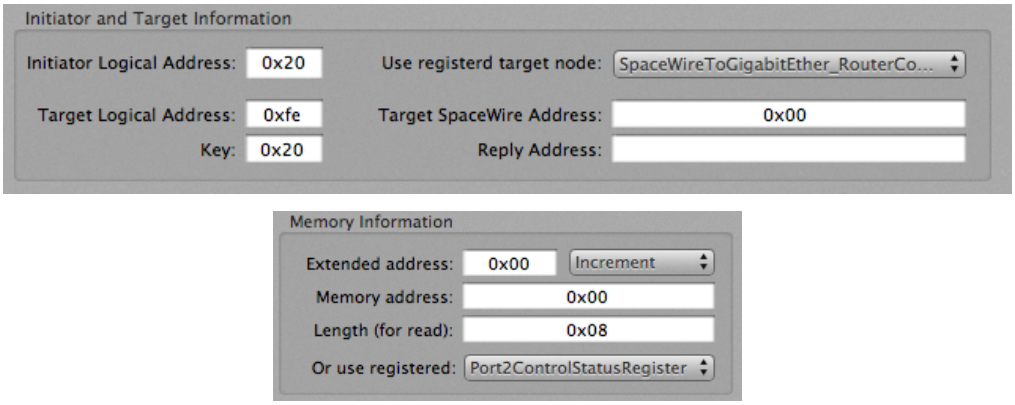

Figure 9: The Initiator and Target Information section (top) and the Memory Information section (bottom) after loading the example XML configuration file, and selecting one of the defined entries in each pull-down menu.

Figure 10: An example of sending a packet.

5.2 Receiving packets

To receive a packet as a raw SpaceWire packet, click "Receive (1 packet)". When a packet is received within a timeout duration which is set via the "Receive timeout" field (in units of milli second), its byte sequence is displayed in the text field.

Multiple packets can be continuously received, by clicking "Receive (continuous)". When many packets are received in a short time, only a part of them is displayed.

Note: To receive packets in the SpaceWire tab, the RMAPEngine used in the RMAP tab should be stopped because RMAPEngine tries to receive all incoming packets in background disabling packet receiving of the SpaceWire tab (this is caused by the two tabs shares one physical SpaceWire connection).

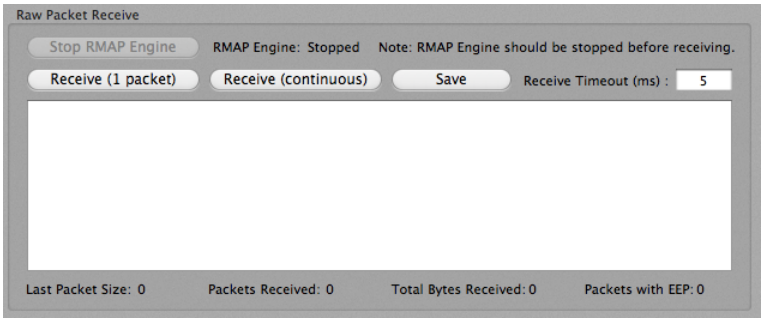

Figure 11: The Raw Packet Receive window.

5.3 Emitting timecode

Timecode character can be emitted by clicking "Emit one-shot" or "Emit periodically". One-shot emission uses timecode value in the Timecode value field. During periodic timecode emission, timecode value is automatically incremented, and emitted at a rate specified in the Timecode frequency field in units of Hz. Too low or too high timecode frequency results an error message to prevent from unexpected response of SpaceWire-to-GigabitEther and connected

SpaceWire nodes. Current timecode value, and periodic emission status is displayed in the left part of the timecode section.

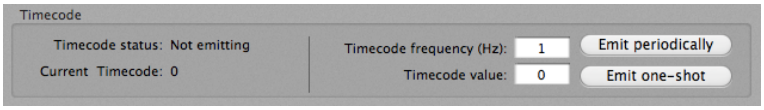

Figure 12: Timecode section in the SpaceWire tab.

6 RMAP functions

To perform RMAP read/write, various types of information must be provided; information of an RMAP target node (i.e. destination of an RMAP command packet), memory information (i.e. address and length), instruction (i.e. read or write with verify/no verify and/or reply/no reply), and information regarding a transaction (transaction ID mode and timeout duration).

Figure 13 shows the Initiator and Target information section, whose content can be manually input, or registered RMAP Target Node information can be loaded from the Use registered target node pull-down menu. Note that the target node address field accepts inputs like ”0x03 0x0f” and ”0x030f” as well. When an input reply address has a length which is not a multiple of 4, leading 0s are automatically padded.

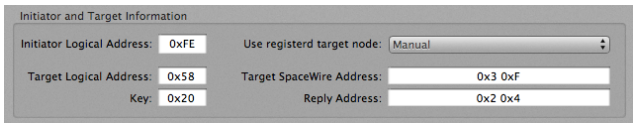

Figure 13: The Initiator and Target information section. Target SpaceWire address and Reply address are interpreted as ”0x03 0x0f” and ”0x00 0x00 0x02 0x04” in this example.

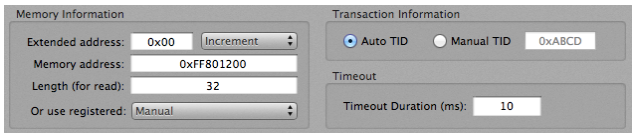

Figure 14: The Memory, the transaction ID, and the timeout sections.

Refer specifications or user manuals of RMAP target devices for memory addresses accessible via RMAP. Length is only necessary for read transactions, and is automatically calculated for write transactions.

Transaction management is done by the internal RMAP Engine based on transaction ID. Although automatic transaction ID assignment, which is default of SpaceWire RMAP GUI, will be acceptable for many use cases, users can also specify a transaction ID manually to force the RMAP Engine to put the transaction ID to an RMAP command packet. When using a manually selected transaction ID, please carefully check if an emitted RMAP command packet has an expected transaction ID (e.g. by checking ”Log packet dump”).

6.1 Performing Write and Read

After setting the initiator and the target information and the memory information, users can perform RMAP write and read very easily. Figure 15 shows the write data section with an example write data. Clicking ”RMAP Write”, an RMAP Write command packet will be sent to a designated target node. In RMAP Write, the data length is automatically calculated from the input data, and the length field is ignored. Whether a user request verification on data (i.e. CRC calculation) and/or a reply for the command packet can be selected from the Options field. When a reply is not received within a specified timeout duration although it (a reply) is requested, an error message will be reported in the log window. Users can input multi-byte data like 0xabcdef1234, and this is treated as an array of 0xab 0xcd 0xef 0x12 0x34 (see §3.8).

In RMAP Read, read length should be specified in the length field (in the Memory information section above). By clicking ”RMAP Read”, an RMAP Read command packet is sent to a target node. When a reply packet is received within a timeout duration, read data will be displayed in the RMAP Read section (byte-to-byte) as shown in Figure 16. Timeout is logged as an error message.

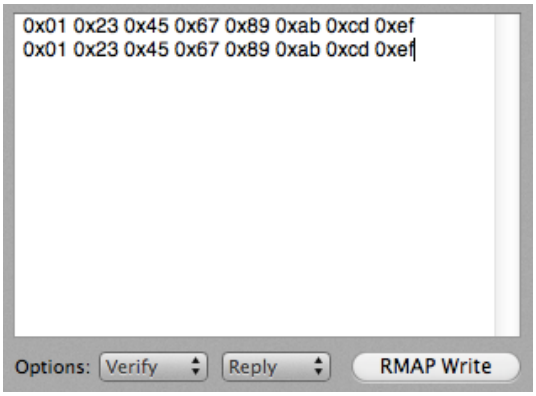

Figure 15: The RMAP Write section with write data.

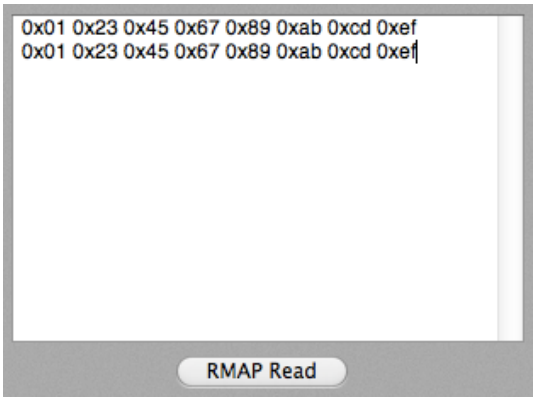

Figure 16: The RMAP Read section with read data.

6.2 Errors and timeout

When an RMAP reply packet with a status other than "Command Successfully Executed (0x00)", an error message is reported in the log window. Timeout is also reported as an error with a message shown in Figure 17.

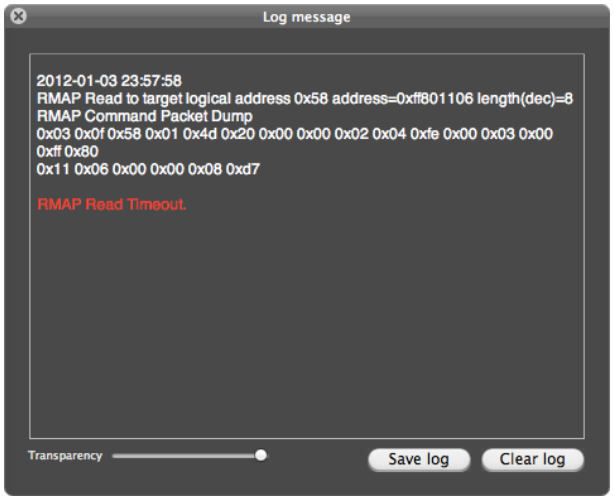

Figure 17: Timeout message reported in the log window.

6.3 Use defined RMAP Target Nodes and memory objects

After loading an XML-like configuration file (§4.2), users can use registered information of RMAP Target Nodes such as target logical address, reply address, memory address, data length, and so on. Select a target node and its memory object from the "Use registered target node" and the "Use registered" pull-down menus, respectively. Registered information will be updated to corresponding fields.

To clear registered RMAP Target Node information, click "Clear nodes" in the Configuration section. Details of registered RMAP Target Nodes (text dump) are available from the sub window (click "Show registered RMAP Target Nodes").

6.4 Logging dump of an RMAP packet

By checking "Log packet dump" beneath the RMAP Read button, all the sent and received RMAP packets are logged so as to allow off-line analyses of the packets. Figure 18 shows an example of logging a transaction, i.e. command and reply packets. Since SpaceWire RMAP GUI has an integrated RMAP packet analysis function in the RMAP Packet Utility tab, users can easily analyze packets by copying and pasting byte sequence to the tab, and then clicking "Interpret as an RMAP packet".

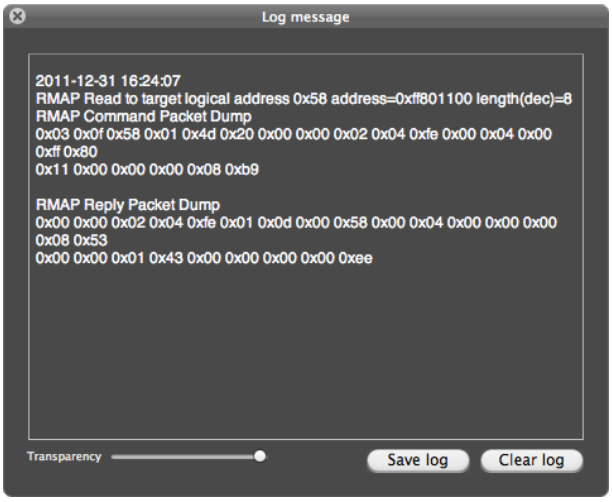

Figure 18: Logged RMAP packet dump.

7 RMAP Packet Utility functions

SpaceWire RMAP GUI provides capabilities of inspecting and constructing RMAP packets. The RMAP Packet Utility tab is an interface to these functionalities.

7.1 Creating RMAP packets

When creating an RMAP packet, input properties of the packet, and then click ”Generate packet”. Resulting packet byte sequence will be shown in the right text field. Figure 19 presents a result of generation of a packet shown in the RMAP standard document (”RMAP non-verified incrementing write-with-reply command - without SpaceWire addresses”; for see A.4 of the document). By checking ”Log dump packet”, details of the generated packet is added as a log message as Figure 20 shows.

7.2 Interpreting RMAP packets

Input byte sequence to the right text field, and then click ”Interpret packet”. Header information and data will be set to text fields in the left column. Like the generation function, detailed packet interpretation log can be displayed by checking ”Log packet dump” next to the ”Interpret packet” button.

7.3 CRC error

The packet interpretation function reports a warning when it detects an invalid CRC in the header or the data as shown in Figures 22, 23, and 24. A correct CRC value is automatically calculated and set to the CRC field in the left column. The value is also displayed in the log window.

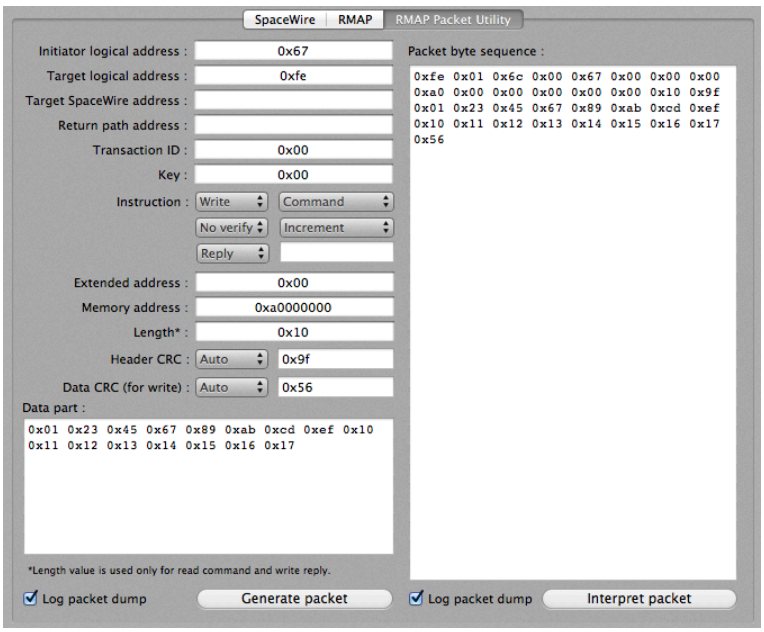

Figure 19: An example of RMAP packet generation/interpretation.

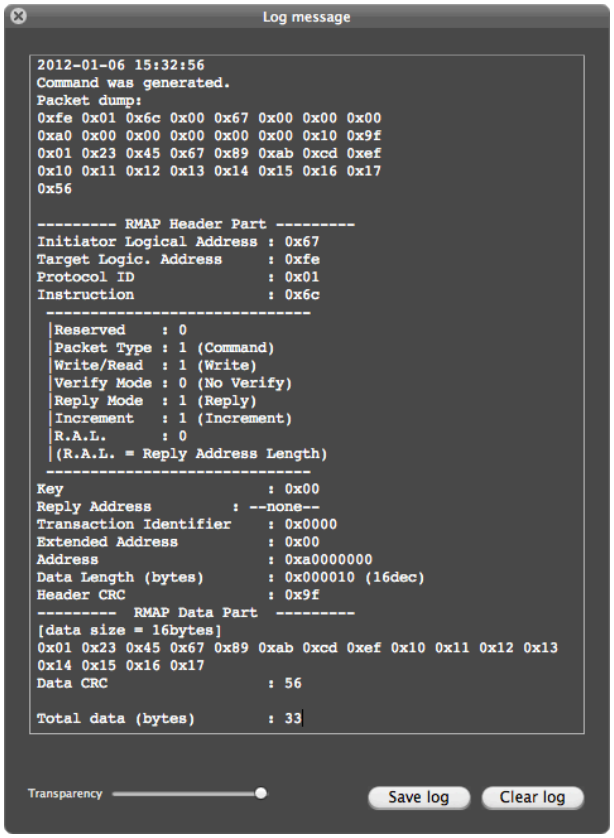

Figure 20: Log message added when generating an RMAP packet. Note that "Log packet button" is checked.

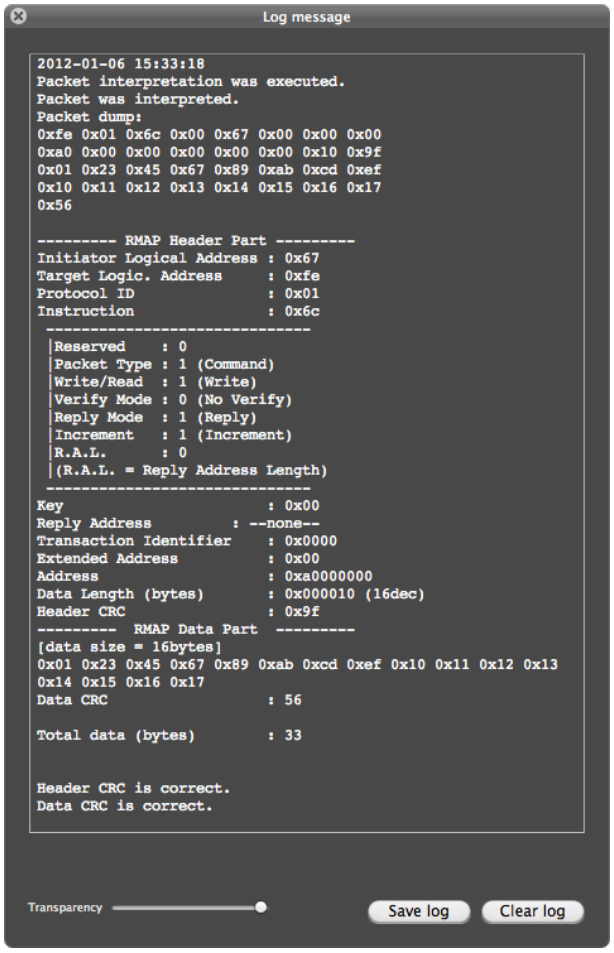

Figure 21: Log message added when interpreting an RMAP packet. Note that "Log packet button" is checked.

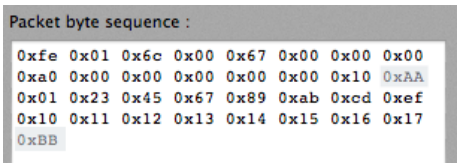

Figure 22: Sample packet with invalid header and data CRCs (shaded bytes).

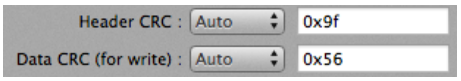

Figure 23: Valid CRC values calculated in interpretation.

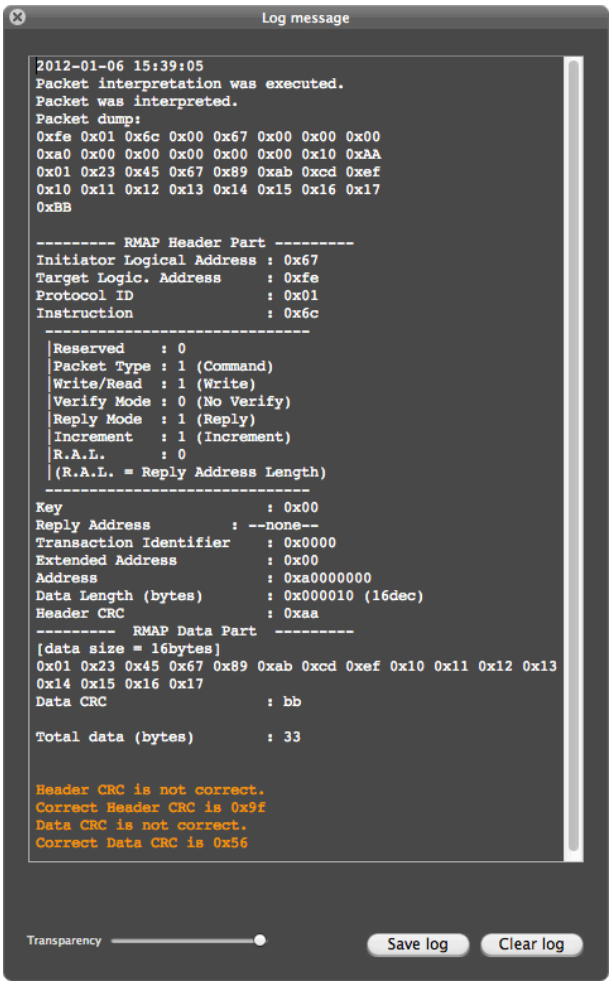

Figure 24: Log message shown when interpreting the same packet shown in Figure 22. Note the orange message telling CRCs are invalid.

A Format of the XML-like configuration file

RMAP Target Node information and memory object information can be stored in an XML-like configuration file. The format is defined in SpaceWire/RMAP Library, specifically, in the RMAPTargetNode class and the RMAPMemoryObject class, and therefore, for details, see SpaceWire/RMAP Library User Guide.

Structures of RMAPTargetNode and RMAPMemoryObject are listed below. When one (or more) of mandatory tag is not found, the configuration file is discarded.

RMAPTargetNode id (name) attribute is mandatory.

TargetLogicalAddress Mandatory.
TargetSpaceWireAddress Mandatory. Array of 0x00-0xFF. (e.g. 0x02 0x0a 0x07 0x01)
ReplyAddress Mandatory. Array of 0x00-0xFF. (e.g. 0x02 0x0a 0x07 0x01)
Key Mandatory. 0x00-0xFF.
InitiatorLogicalAddress Optional. 0x00-0xFF.

RMAPMemoryObject id (name) attribute is mandatory.

ExtendedAddress Optional. Default is 0x00.
Address Mandatory. 0x00000000-0xFFFFFFFF.
Length Mandatory. 0x000000-0xFFFFFF.
Key Optional. 0x00-0xFF. The value defined in the parent RMAPTargetNode is overridden by this value if set.
AccessMode Optional. Any of ReadWrite, ReadOnly, WriteOnly, Readable (=ReadOnly), Writable (=WriteOnly).
IncrementMode Optional. Either of Increment or NoIncrement.

The below shows a template for a configuration file. Note that one file can contain multiple RMAPTargetNodes, and one RMAPTargetNode tag can contains multiple memory object definitions.

Listing 2: Tags which define RMAPTargetNode and RMAPMemoryObject.

```
<root>

<RMAPTargetNode id="NameOfTheRMAPTargetNode">
        <TargetLogicalAddress>0xFE</TargetLogicalAddress>
        <TargetSpaceWireAddress>0x00</TargetSpaceWireAddress>
        <ReplyAddress></ReplyAddress>
        <Key>0x20</Key>
        <InitiatorLogicalAddress>0x35</InitiatorLogicalAddress> <!-- optional -->

        <RMAPMemoryObject id="NameOfTheMemoryObjectOnTheRMAPTargetNode">
                <ExtendedAddress>0x00</ExtendedAddress>
                <Address>0x0000</Address>
                <Length>0x04</Length>
                <Key>0x20</Key> <!-- optional -->
                <IncrementMode>Increment</IncrementMode>
        </RMAPMemoryObject>

        ... other RMAPMemoryObject tags ...

</RMAPTargetNode>

... other RMAPTargetNode tags ...

</root>
```