

A PORTABLE SPACEWIRE/RMAP CLASS LIBRARY FOR SCIENTIFIC DETECTOR READ OUT SYSTEMS

Session: Onboard Equipment and Software

Short Paper

Takayuki Yuasa, Wataru Kokuyama, Kazuo Makishima, Kazuhiro Nakazawa

The University of Tokyo, 7-3-1 Hongo, Bunkyo, Tokyo, Japan 113-0033

Masaharu Nomachi,

*Laboratory of Nuclear Studies, Graduate School of Science, Osaka University, 1-1
Machikaneyama, Toyonaka, Osaka 560-0043*

Motohide Kokubun, Hirokazu Odaka, Takeshi Takashima, Tadayuki Takahashi

*Department of High Energy Astrophysics, Institute of Space and Astronautical Science (ISAS), Japan
Aerospace Exploration Agency (JAXA), 3-1-1 Yoshinodai, Sagamihara, Kanagawa Japan 229-8510*

E-mail: yuasa@amalthea.phys.s.u-tokyo.ac.jp, kokuyama@granite.phys.s.u-tokyo.ac.jp,
maxima@phys.s.u-tokyo.ac.jp, nakazawa@phys.s.u-tokyo.ac.jp,
nomachi@fn.lns.sci.osaka-u.ac.jp, kokubun@astro.isas.jaxa.jp, odaka@astro.isas.jaxa.jp,
ttakeshi@stp.isas.jaxa.jp, takahasi@astro.isas.jaxa.jp

ABSTRACT

We developed a C++ class library which provides modularized scheme to transfer data via RMAP over SpaceWire. The library is designed to be highly portable so that users can execute their products using it on both POSIX (eg. Linux or MacOS X) and TRON environments. TRON is a real-time operating system that is widely adopted in embedded computers, and will also be used on computers, called SpaceCube, to be onboard Japanese scientific satellites. To achieve the portability, we encapsulated hardware- and operating-system-dependent functionalities such as SpaceWire I/F, a multi threading framework, as well as a TCP/IP socket implemented for development phases of scientific payloads.

1 SPACEWIRE/RMAP LIBRARY

1.1 SPACEWIRE-BASED DATA ACQUISITION SYSTEM

We are developing a new data acquisition (DAQ) system based on SpaceWire and RMAP (Remote Memory Access Protocol) to be widely used in future satellite-borne scientific instrument developments [1,2]. Compared to conventional systems, a SpaceWire network-based DAQ system has features of high scalability and compactness. Using SpaceWire I/F from early stages of developments will enable smooth transition from R&D phase to actual fabrication and integration of the flight hardware. To reduce costs of individual developers, a standard framework for SpaceWire-based DAQ has been developed and distributed by Japan SpaceWire Users Group which consists of research institutes (including JAXA and universities), and industrial enterprises.

As shown in Figure 1, present DAQ system consists of the following three components.

- **Detector:** Outputs analog or digital signals. Controlled through specific interfaces.
- **SpaceWire I/F Circuit Board:** Digitizes and stores output signals to make them accessible from SpaceWire network through their SpaceWire interfaces. Equipped with reconfigurable FPGAs to implement user-dependent logics for detector control and signal processing.
- **SpaceCube:** Small-sized computer with SpaceWire interfaces. Operated with a real-time operating system, on which user-dependent read-out programs written in C/C++ runs. "SpaceCube" is a name of the architecture, not a product name.

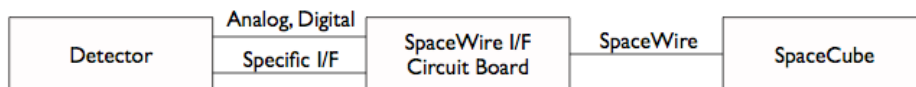


Figure 1. Components used in SpaceWire-based data acquisition

Detectors are designed and developed by individual users. Several types of general-purpose SpaceWire I/F circuit boards have been developed each with different functionality such as analog-to-digital conversion (ADC) and digital input/output. As SpaceCube computer, we usually use SpaceCube1 developed by Shimafuji Electric for ground-based experiments. Currently available space-qualified SpaceCube computers include SpaceCube2 (NEC) and SpaceCard (Mitsubishi Heavy Industrial).

1.2 CONCEPT AND STRUCTURE OF SPACEWIRE/RMAP LIBRARY

Individual instrument developments need their own read-out program to perform different SpaceWire and RMAP accesses. However, basic functions needed therein are essentially the same; examples include send/receive packets and interpret/create RMAP packets. Therefore we developed a common software library scheme to deal with SpaceWire/RMAP-related functions, called SpaceWire/RMAP Library. It is designed to be modularized and portable, to improve re-usability of products and to make validation processes easier. Since ground-based and spacecraft-borne SpaceCube computers could have different operating systems, dependencies on the hardware and operating system are carefully minimized.

The library is written in C++ language, and has no dependency on external library except for STL (Standard Template Library; widely available in many operating systems and compilers). Figure 2 shows the whole structure of the library. SpaceWire I/F class is an abstract wrapper class for SpaceWire I/F hardware and drivers loaded in operating system kernels. It holds virtual methods for device initialization/finalization (open()/close()), packet transfer (send()/receive()), and device configuration (setLinkStatus()). Actual SpaceWire

interfaces can be utilized by filling each virtual method with codes specific to those devices. Currently, two types of implementations of SpaceWireIF class are available,

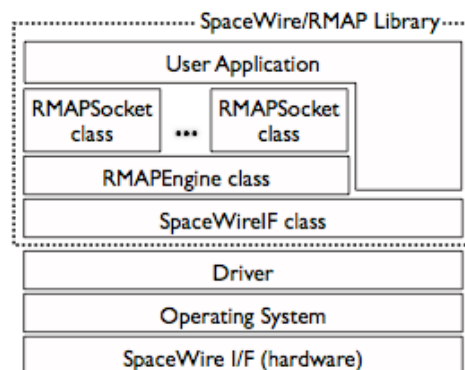


Figure 2. Structure of SpaceWire/RMAP Library.

each for SpaceWire interfaces (or IP cores) developed by Shimafuji Electric and by NEC Software, respectively.

RMAP transaction is realized by two conjunct classes, RMAPSocket and RMAPEngine. RMAPSocket, which corresponds to socket in a TCP/IP protocol stack, provides read() and write() methods for user application to perform RMAP transactions to a designated RMAP destination node. RMAPEngine handles requests from multiple RMAPSockets and realizes concurrent multiple RMAP transactions (to different RMAP nodes) in multi-threading environments. RMAP packets, errors, and destination information including routing pathways, are also expressed as classes.

A DAQ read-out program often uses multi threading and TCP/IP data transfer. Although those functions have less relevance to SpaceWire or RMAP, we also included them into the library as encapsulating classes, called Thread Library and IP Socket Library, referring to Java Thread and Socket in their naming style. Table 1 lists classes included in these libraries. At present, these libraries provide implementation classes for POSIX and T-Kernel¹ environments, and therefore, ensure a high portability of the programs developed using these libraries. In section 2.2, we present an example of the portability by executing a multi-thread read-out program on both Macintosh (POSIX) and SpaceCube1 (T-Kernel) without changing the source code.

Table 1. Contents of Thread and IP Socket Libraries.

- Thread Library
 - Thread class
 - Condition class
 - Mutex class
 - Message class
- IP Socket Library
 - IPSocket class
 - IPClientSocket class
 - IPServerSocket class

1.3 CURRENT STATUS OF THE LIBRARY

The total size of SpaceWire/RMAP Library and sub-libraries is about 300,000-line C++ code, including in-source documentation for automatic document generators such as Doxygen or Javadoc. The source code archive is distributed through Japan SpaceWire Users Group, together with the Shimafuji Electric's driver software for SpaceWire I/F IP core on SpaceCube1². GNU Compiler Collection's C++ compiler (GCC g++) can be used to build the library and read-out programs. Since an HTML-based reference of the application programming interface (in English) and an introductory tutorial (in Japanese) are also available, users can start developments immediately. The library has been used in several detector development activities, including the X-ray CCD and the X-ray micro-calorimeter experiments onboard the ASTRO-H satellite, the next generation Japanese X-ray astrophysical observatory to be launched around 2013.

2 PERFORMANCE AND AN EXAMPLE OF APPLICATION

2.1 TRANSFER SPEED

We measured the data transfer speed using SpaceCube1, implemented with Shimafuji Electric SpaceWire IP core and the SpaceWire/RMAP Library. In hardware (physical) layer, the IP core exploits about 80% of link speed; for example, 80 Mbps can be achieved over a 100 MHz link. Overhead time of the host I/F (PCI I/F) and the

¹ T-Kernel is one kind of implementation of TRON operating systems.

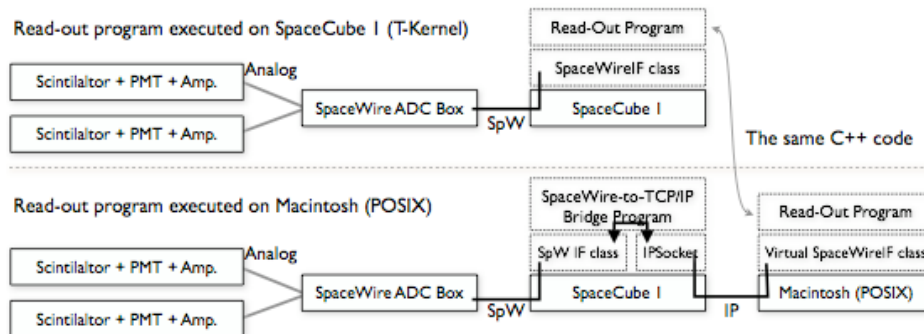
² Which will be released as an open source product in the near future at <http://www.shimafuji.co.jp/>.

driver software reduce the speed to about 32 Mbps (in SpaceWireIF class layer). We also tested practical end-to-end data transfer speed between a RAM on SpaceWire I/F circuit board and a read-out program on SpaceCube1. The SpaceWire I/F circuit board acts as an RMAP target device, and the read-out program utilizes full features of SpaceWire/RMAP Library, such as the concurrent multiple transaction with RMAPSocket and RMAPEngine. When we transferred data with a packet size of 8 kB, the transfer speed of 8 Mbps was obtained.

2.2 EXAMPLE APPLICATION

We applied the SpaceWire/RMAP Library to a simple ground experiment setup. The detector consists of two photo-multiplier tubes (PMTs) with inorganic scintillators and amplifiers attached to them. The SpaceWire I/F circuit board in this case is a octal 50 MHz ADC board, which is controlled and read-out via its SpaceWire I/F. Using SpaceWire/RMAP Library, we developed a read-out program initially for SpaceCube1. To examine the portability, we then built the same source code with a C++ compiler for MacOS X (g++), and executed the generated binary on Macintosh. Since a SpaceWire I/F for Macintosh was not available at that time, the program used the SpaceCube's SpaceWire I/F via TCP/IP. To realize a SpaceWire-to-TCP/IP converter, we also executed a converter server daemon program on the SpaceCube³.

The read-out program executed on Macintosh communicates with the SpaceCube via TCP/IP to send/receive SpaceWire packets to/from SpaceWire ADC Box (Figure 3). Such procedures are automatically performed inside the library, and hence, the read-out program need not to be modified. The total DAQ system was also successfully operated in the same way as on SpaceCube1, and correctly transferred 700,000 events with an event size of 10 bytes/event each in 5 minutes in a test measurement wherein we irradiated the detector with gamma rays from radioisotopes.



3 REFERENCES

Figure 3. A block diagram of test measurements.

1. Takayuki Yuasa et al., "Development of a SpW/RMAP-based Data Acquisition Framework for Scientific Detector Applications", International SpaceWire Conference, Dundee, Scotland, September 2007
2. Hirokazu Odaka et al., "Development of a SpaceWire-based Data Acquisition System for a Semiconductor Compton Camera", International SpaceWire Conference, Dundee, Scotland, September 2007

³ The SpaceWire-to-TCP/IP converter program for SpaceCube1 is also included in the library package.