Remote Memory Access in Embedded Networked Systems

V. Olenev, I. Korobkov, L. Koblyakova, F. Shutenko

SUAI
190000, B.Morskaya 67, Saint_Petersburg, Russia
valentin.olenev@guap.ru, zelwa@yandex.ru, Luda_o@rambler.ru, felixshutenko@yandex.ru

Abstract

Remote memory access is one of the important tasks in modern Embedded Networked Systems. Each developer of system has a vision of solution of the given task. In this paper the approach to solution the remote access to memory in standard SpaceWire by means of RMAP protocol is considered.

The newest UniPro standard of data transfer has a similar problem. The solution of remote access to memory in UniPro can be different. In article it is offered to use RMAP protocol for these purposes. Also RMAP software and hardware implementation is presented here with some technical characteristics.

Index Terms: Embedded Networked Systems, SpaceWire, RMAP, UniPro, MCK-01, SWIC.

I. Introduction

Modern Embedded Networked Systems are frequently consist of a considerable quantity of devices (nodes) allocated on different distances from each other. Nodes can be use as initiators or targets. It is necessary to configure the devices: to set the logic address, the routing table, a device operating mode and other parameters. It is also necessary for devices to communicate during the operating process, for example, to update the routing table at connection or disconnection of new devices, or write some information to memory of remote device, e.g. at transferring of useful data from one user to another.

In most cases it could be done by using write and read commands in appropriate program-accessible components of the remote device (e.g. memory, registers). Thereby remote access to memory is one of the most important tasks in Embedded Networked Systems.

II. Main part

There are many approaches to solve the task. For example, ESA (European Space Agency) in collaboration with international space agencies including NASA, JAXA and Roscosmos has designed RMAP (Remote Memory Access Protocol) protocol to support a wide range of SpaceWire standard applications. Let's consider more in detail what is SpaceWire and RMAP.

A. SpaceWire.

SpaceWire is a spacecraft communication network based in part on the IEEE 1355 standard of communications. The purpose of SpaceWire is [2]:

– to facilitate the construction of high performance on board data handling systems;

– to help reduce system integration costs;

– to promote compatibility between data handling equipment and subsystems;

– to encourage reuse of data handling equipment across several different missions.

Within a SpaceWire network the nodes are connected through low-cost, low-latency, full-duplex, point-to-point high-rate serial links and packet switching wormhole routing routers. Note that within SpaceWire, a separate definition of levels that are encompassed by the Physical Layer and Data Link Layer of the OSI model are defined. These are as follows: Physical Level; Signal Level; Character Level; Exchange Level; Packet Level; Network Level.

One of the principal aims of SpaceWire is the support of equipment compatibility and reuse at both the component and subsystem levels. In principle a data handling system developed for an optical instrument, for example, can be used for a radar instrument by unplugging the optical sensor and plugging in the radar one. Processing units, mass memory units and down link telemetry systems developed for one mission can be readily used on another mission, reducing the cost of development, improving reliability and most importantly increasing the amount of scientific work that can be achieved within a limited budget. [2]

This standard addresses the handling of payload data and control information on board a spacecraft. It is a standard for a high speed data link (between 2 and 400 Mb/s), which is intended to meet the needs of future, high capability, remote sensing instruments and other space missions. SpaceWire provides a unified high speed data handling infrastructure for connecting together sensors, processing elements, mass memory units, downlink telemetry subsystems and EGSE equipment. [2]

B. RMAP

The remote memory access protocol (RMAP) provides means for a SpaceWire node to write to and read from memory inside another SpaceWire node. The aim of the RMAP protocol is to standardize the way in which SpaceWire units are configured and to provide a low-level mechanism for the transfer of data between two SpaceWire nodes. [5]

RMAP may be used to configure SpaceWire routing switches, setting their operating parameters and routing table information. It may also be used to monitor the status of those routing switches. RMAP may be used to configure and read the status of nodes on the SpaceWire network. For example, the operating data rate of a node may be set to 100 Mbits/s and the interface may be set to auto-start mode. Also RMAP may be used to configure a camera or a mass memory device. The camera device may then write image data to allocated areas of memory in the mass memory, or the mass memory may read image data from the camera [5].

All read and write operations defined in the RMAP protocol are posted operations i.e. the source does not wait for an acknowledgement or reply to be received. This means that many reads and writes can be outstanding at any time. It also means that there is no timeout mechanism implemented in RMAP for missing acknowledgements or replies. If an acknowledgement or reply timeout mechanism is required it must be implemented in the source user application. [4]

C. RMAP Key Features

Advantages of RMAP protocol:

–simplicity of implementation – 4 write commands (acknowledged or non-acknowledged, verified or non-verified), 1 read command and 1 read-modify command that allows to fulfill atomic operations;

–wide application and flexibility – can be used for a wide range of tasks, interpretation of address fields of a RMAP-command depends on the developer;

–reliability – available CRC-check of header and body of the command;

–transmission bulk of data – can transfer to 16 Mbytes of the data in one command;

–supports 32 and 40-bit address space.

D. SpaceWire-RMAP

Let's consider implementation RMAP on the top of SpaceWire.

The SpaceWire-RMAP intellectual property (IP) core implements the Remote Memory Access Protocol (RMAP) extensions to SpaceWire. RMAP provides a standard mechanism for reading from, and writing to memory in a remote SpaceWire node. [3]

There are two main function types of the RMAP IP core. The first type is referred to as the Initiator RMAP interface, which sends out RMAP commands and receives any replies. The second type is the Target RMAP Interface, which receives RMAP commands, executes them and sends out any required replies. The RMAP IP core also includes the SpaceWire-b CODEC IP block which handles the SpaceWire protocol point to point link. [3]

The RMAP IP core target and initiator function is illustrated in the following diagram.

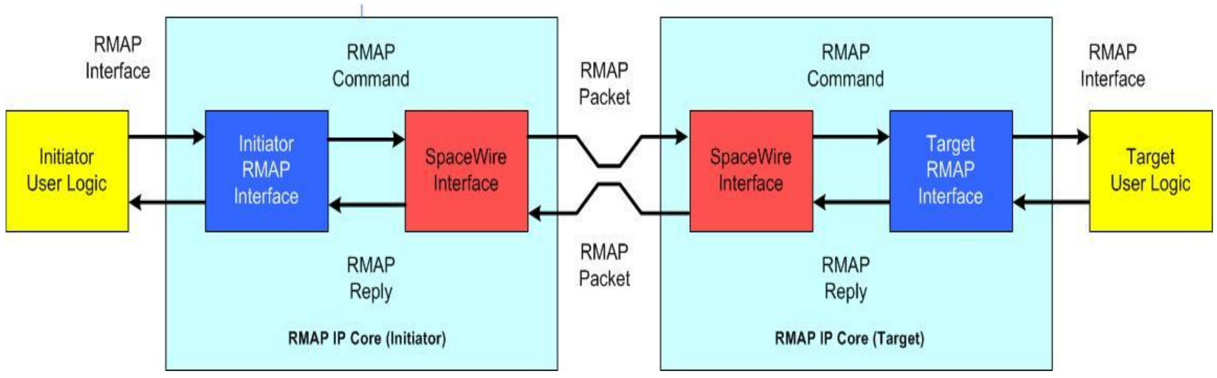

Figure 1. RMAP IP Core Data Flow

RMAP packets are initiated in the initiator user logic, encoded as RMAP packets in the “Initiator RMAP Interface”, sent over the SpaceWire link as an RMAP packet, decoded by the “Target RMAP Interface” and data or information is passed to the target user logic after authorization of the command. The “Target RMAP Interface” formats an RMAP reply packet which is sent over the SpaceWire interface, decoded by the “Initiator RMAP Interface” and the reply data/information is passed to the initiator user logic. The IP core can be configured to be target only, initiator only or both. [3]

E. UniPro

Development of the newest standard UniPro for data transfer in Embedded Networked Systems is now carried on. A similar task – remote access to memory implementation can arise in Embedded Networked Systems also. We will consider more in detail UniPro, its architecture and features.

UniPro represents a multilayer unified protocol (Unified Protocol, abbreviation “UniPro”) on the high-speed serial interface for integration [1] different devices and components with portable systems, such as application processors, modems, video cameras, displays and other peripheral units, and associated with them different traffic types [1], for example, control messages, transmission of bulk data and flow control [1] for data packages.

Developer of UniPro is the MIPI Alliance – an open standards development organization with over 150 member companies, mainly from the mobile communications industry.

Note that UniPro was designed for low-power systems. UniPro protocol is implemented to simplification of development digital devices with complex structure. It is supposed that the future architecture of mobile phones will consist of the interconnected components which co-operate [1] and should be easily compatible similar to PC architecture. As a matter of fact, UniPro is the first serious step in this direction [1].

F. UniPro Key Features

UniPro has a number of key features [7]:

–gigabit/s - serial technology with a number of bandwidth scaling options

–generic - can be used for a wide range of applications and data traffic types

–scalable - from individual links to a network with up to 128 UniPro devices

–low-power - optimized for small battery-powered systems

–reliability - data errors detected and correctable via retransmission

–hardware friendly - can be implemented entirely in hardware where needed

–software friendly - similar concepts to familiar network technologies

–bandwidth utilization - provides features to manage congestion and control arbitration

–shareable - different traffic types and UniPro devices can share pins and wires

–testable - since version 1.1, UniPro mandates features to facilitate automated conformance testing

G. UniPro Architecture

The UniPro protocol stack follows the classical OSI reference architecture. For practical reasons, OSI's Physical Layer is split into two sublayers: Layer 1 (the actual physical layer) and Layer 1.5 (the PHY Adapter layer) which abstracts from differences between alternative Layer 1 technologies. [7]

The UniPro specification itself covers Layers 1.5, 2, 3, and 4. The Application Layer (LA) is out of scope, because different uses of UniPro will require different LA protocols. The Physical Layer (L1) is covered in separate MIPI specifications in order to allow the PHY to be reused by other (less generic) protocols if needed. OSI Layers 5 (Session) and 6 (Presentation) are, where applicable, counted as part of the Application Layer. [7]

UniPro is the standard of the future integration of digital devices, providing high reliability and transfer rate between devices in mobile systems and between mobile systems [1].

UniPro's strict layering enables it to be used for a wide range of applications. The first application protocols designed to run over UniPro is MIPI's "PIE". This application protocol conveys traditional memory-based read/write transactions as found on processor busses. Data streaming applications (e.g. multimedia traffic), command/response-type protocols (e.g. for

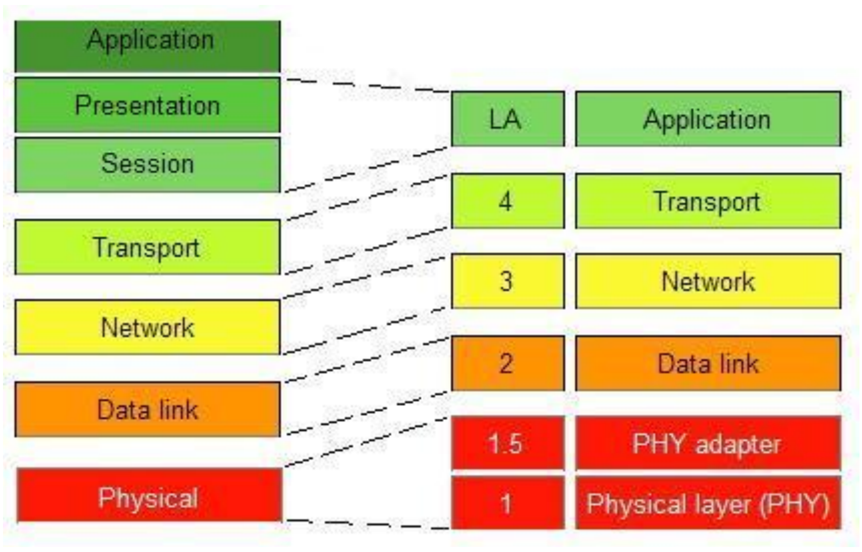

Figure 2. OSI stack Figure 3. UniPro protocol stack

control), and tunneling of popular protocols from other domains (e.g. TCP/IP) are also supported and specifically encouraged because they tend to increase system-level modularity and interoperability due to their higher abstraction level. [7]

H. RMAP over UniPro

So long as PIE protocol is not specified so far, it is possible to use RMAP. RMAP protocol will be in role of a method of remote access to memory, and LA layer will be responsible for logical processing of incoming and outcoming RMAP-commands and RMAP-replies. LA object will be LA Entity. It cans appear both the initiator and the target of RMAP-packages.

The LA Entity target and initiator function is illustrated in the following diagram.

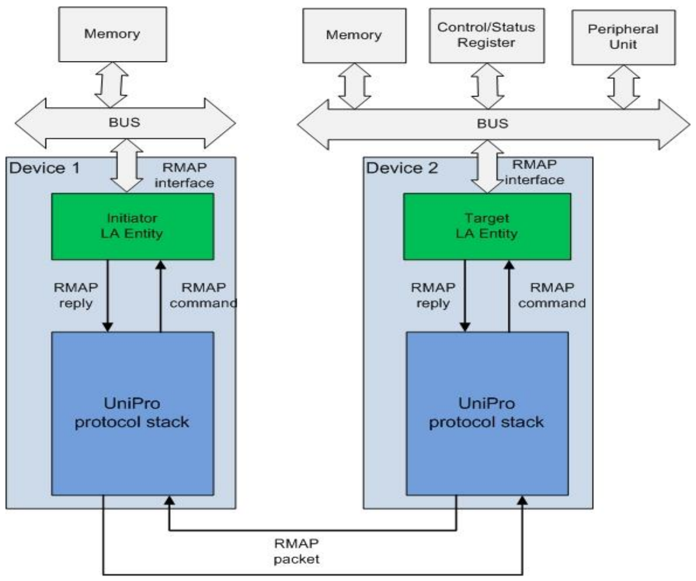

Figure 4. Initiator and Target LA Entity function

RMAP packets are created in the "Initiator LA Entity" and then they encoded as RMAP packets. RMAP-packets go to Device 2 over the UniPro.

These packets are decoded by the "Target LA Entity" into Device 2 and data or information is passed to Memory, Control/Status Register or other Peripheral Unit after authorization of the RMAP-command. The "Target LA Entity" formats an RMAP reply packet which is sent to Device 1 over UniPro link.

RMAP-packet is decoded by the "Initiator LA Entity" into Device1 and the reply data/information is passed to the Memory. LA Entity cans be configured to be target only, initiator only or both function.

Note that communication between LA layer and UniPro protocol stack and communication between LA layer and high-speed serial interface probably implements by the use of SAP (Service Access Point).

I. RMAP software implementation

Let's consider RMAP software implementation in SpaceWire network.

RMAP is used for configuration of any remote routing switch in a SpaceWire network. Each switch MSK-01 has embedded RISC kernel and specialized embedded software (RISC core and Firmware). Firmware supports processing of RMAP-packets.

RMAP software implementation consists of 2 parts:

- Generator of RMAP-packets
- Handler (processor) of RMAP-packets

Example of SpaceWire network is illustrated in the following diagram.

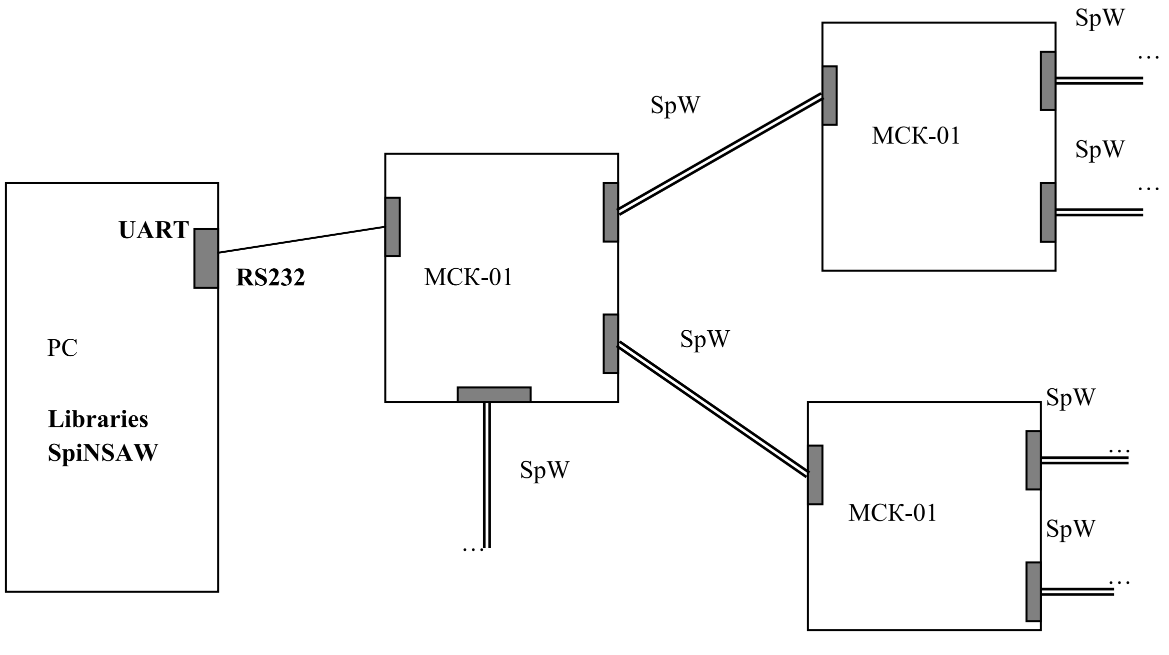

Figure 5. Example of SpaceWire network

Generator of RMAP-packets is MCK-01 routing switch. If RMAP-packet comes to configuration (zero) port of routing switch, this packet writes to memory and interruption is occurred. The Interrupt Handler in MCK-01 assorts the coming package in according with RMAP-format. Then Interrupt Handler executes the specified RMAP command. If it is necessary, Firmware creates RMAP-reply and sends it to network.

Any device (with function of creating packet and sending it to network) may be used as Generator. For example (see Figure 5), MCK-01 unit has COM-port. Note that quantity of MCK-01 should be more than one. PC (Personal Computer) creates RMAP-packet (by the use of special library or application) and sends it through COM-port with special command to switch. On command routing switch gets RMAP-packet from a COM-port and sends it to network. In this example Generator works in PC. Application (with GUI or without) or software-library can be realization of Generator.

Handler (processor) of RMAP-packets is the Interrupt Handler in Firmware. If Handler is a software implementation, processing time essentially above than at hardware implementation.

J. RMAP hardware implementation

One of the RMAP hardware implementation is presented below.

SWIC (SpaceWire) controller is an example of hardware-implemented core of Switch controller (node of embedded network system).

Following diagram illustrates part of SWIC controller that includes RMAP hardware implementation.

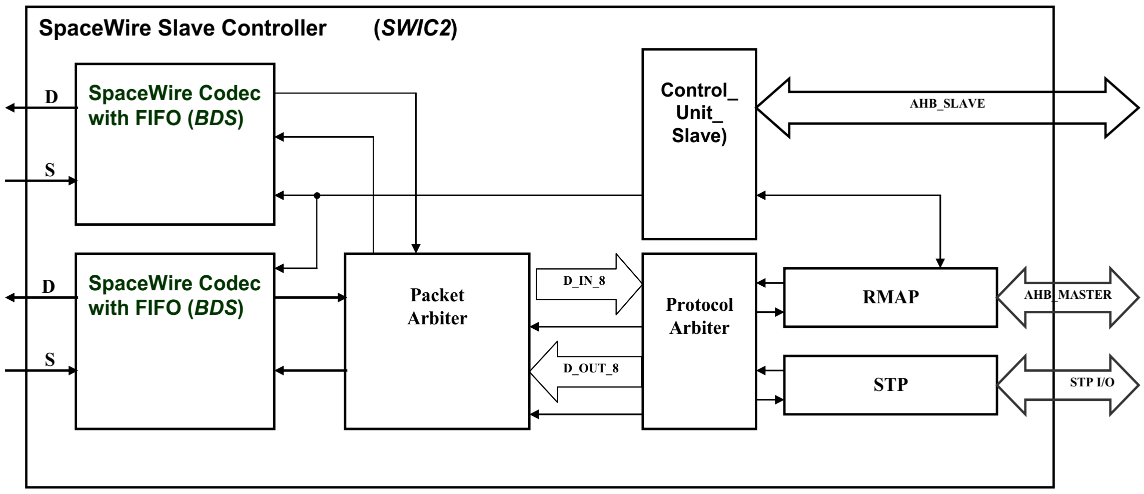

Figure 6. Example of RMAP hardware implementation

SWIC controller includes:

−2 SpaceWire Codecs, to enable insertion the core into an embedded SpaceWire network

−Internal Control Unit (AHB slave, optional, 32-Bit bus). This gives possibility to configure the core from processor core

−RMAP component compatible with ECSS-E-50-11 Draft F (do not support Read-Modify-Write command), to enable remote memory access from embedded application of other nodes

−STP component (Streaming Transport Protocol) works with streaming data like video information etc.

−IRQ interface (Control Unit)

RMAP component of the node supports read and write commands only, and it responsible to send wrire-reply and read-reply packets only. Thus, RMAP component is slave unit.

The hardware module can be implemented as FPGA IP-Core (Spartan or Virtex family). It has the following characteristics:

−1300 slices (2400 LUTs), thus taking 26% of 3s500evq100-4 Spartan-3 device

−65 MHz local clock of 3s500evq100-4 Spartan-3 device

III. CONCLUSION

Thereby, using RMAP protocol it is possible to implement remote access to memory on high-speed channel UniPro. Modeling of the presented solution can be realized effectively by the use of specialized library SystemC.

REFERENCES

[1] I. A. Peshakov, I. L. Korobkov, V. L. Olenev, article “UniPro Stack of transfer protocols of the data in Embedded Systems”, SUAI, St. Petersburg, 2009.

[2] ESA (European Space Agency), standard ECSS-E-50-12A, “Space engineering. SpaceWire – Links, nodes, routers and networks. European cooperation for space standardization”, ESA Publications Division ESTEC, Noordwijk, The Netherlands, 2003.

[3] ESA, article “SpaceWire-RMAP”, http://www.esa.int/TEC/Microelectronics/SEM7N20P0WF_0.html

[4] ESA, specification RMAP, ECSS-E-50-11 Draft F 4th December SpaceWire Remote Memory Access Protocol, Steve Parkes and Chris McClements, University of Dundee, Applied Computing, Dundee, DD1 4HN, Scotland, UK.

[6] V. L. Olenev, article “Analysis of Different Approaches to Modeling Protocol Stacks”, SUAI, St.Petersburg, 2009.

[7] Free Encyclopedia, article "UniPro", http://en.wikipedia.org/wiki/Unipro