SPACEWIRE-RT PROTOTYPING

Session: SpaceWire Networks & Protocols

Long Paper

Albert Ferrer and Steve Parkes

School of Computing, University of Dundee, Dundee, DD1 4HN, Scotland, UK.

E-mail: aferrer@computing.dundee.ac.uk, sparkes@computing.dundee.ac.uk

ABSTRACT

SpaceWire Reliable-Timely (SpW-RT) aims to provide a consistent quality of service (QoS) mechanism for SpaceWire networks. It is intended to provide reliable, high data rate communication services and support for control applications where timely delivery is essential. This paper presents the ongoing prototyping activities related with SpW-RT. It introduces the most important protocol functions and how they may be implemented in a wormhole switching network. The lessons learned from the prototypes built were an important contribution to the design of the current SpW-RT protocol specification.

1 INTRODUCTION

SpaceWire Reliable-Timely (SpW-RT) is being designed to provide quality of service mechanisms over SpaceWire [1,2]. Some important design guidelines and objectives are summarized below:

- Achieve high performance and reliability, while being simple to implement and understand.
- Reduce the protocol complexity required by enforcing and standardizing good network design practices.
- Support for software (slow processing power but big buffers) and hardware implementations (fast processing power but small buffers)
- Support for processor-based intelligent nodes, and dumb nodes interfacing instruments and other devices.
- Compatible with most significant SpW compliant devices.
- Provide enough flexibility to accommodate multiple user cases under a unique protocol definition.

SpW RT targets asynchronous networks and scheduled networks. This includes multiple user cases such as:

- Asynchronous communication with dedicated links. It may require high throughput, reliability and flow control.

- Asynchronous communication using shared links that tolerates variable message latency. It may require reliability and flow control.

- Synchronous system for periodic status messages transferred using a guaranteed service.

- Synchronous system for sporadic control messages transferred using a guaranteed service, with opportunistic use of otherwise unused timeslots.

For easy understanding, in the following chapters the term “message” refers to a complete user data unit. The term “packet” refers to an actual SpW packet. The term “data packet” refers to a packet containing a segment of a user data unit.

2 SpW-RT related topics

This chapter provides background information and presents some concepts involved in the development of the SpW-RT protocol.

2.1 Routing

SpaceWire networks use wormhole switching so the packets are not completed buffered in the routers before they are routed. Instead, packets are immediately routed depending on the content of a single byte header. This provides very low latency when there is no congestion in the network and it is usually very simple to implement. The disadvantage is that a packet will block all network resources (SpW links) used by the packet until the transfer is completed.

Path and logical addressing

SpW routers implement logical and path addressing; optionally using Group Adaptive Routing (GAR). Logical addressing allows the packet routing to be changed by modifying the routing tables of routers, without requiring the SpW nodes being notified. With path addressing, nodes must explicitly provide the path to the destination. Both techniques allow using one or more bytes of the header for the routing. Logical addressing is usually implemented with a single byte that matches the logical address of the destination node. However, it is possible to have multiple routes to the same destination using the router's header deletion feature, in the same way that regional addressing is implemented.

Routing priority

Furthermore, some routers may implement a routing priority scheme to deal with multiple packets waiting to be routed through the same output link. For example, SpW-10X allows two priorities levels within a round robin scheme. However it is not possible with standard SpW routers to pre-empt low priority packets.

2.2 RELIABILITY AND REDUNDANCY

SpW standard provides error detection mechanisms for point to point links but as other link layer standards, it does not provide any end-to-end reliability. This capability must be provided by a suitable transport layer such as SpW-RT.

Reliability can be obtained by acknowledging received packets, optionally retrying in case of an error. Multiple redundant paths can be used simultaneously or sequentially. User data is acknowledged using sequence numbers for the data packets or for the data bytes correctly received. This also ensures that data is not out-of-order.

Sending window

The quantity of data or data packets that can be sent without being acknowledged is usually called the sender window. Note that when flow control is considered, the sender window must take into account the receiver window.

SpW networks typically do not buffer packets in the routers so the delay in the reception of the acknowledgement is introduced by the receiver and the possible network congestion while sending the acknowledgement. The sending window should be adjusted so that the maximum throughput is achieved while minimizing the sender buffer required.

Fault cases

SpW links have theoretically, a very low bit error rate. Packets are more likely to contain errors or be lost because of network congestion, permanent faults or other transient faults. Network congestion may force the routers to timeout the blocked packets and spill them. Network congestion may occur because of an inadequate network analysis or by an error in any part of the network. Use of a retry mechanism over the same path is inadequate in case of permanent faults on this path, but may be appropriate when the fault is not located on the path to the destination.

Redundancy

The GAR mechanism can cope with permanent link errors and blocked links but is not suitable to deal with faulty routers, as they may contain invalid routing tables. Therefore, multiple redundant paths are necessary to provide fault tolerant capabilities. High critical applications running in a scheduled network may have reserved redundant paths. In this case it is recommended to implement the simultaneous retry technique, so packets are sent simultaneously on the primary and redundant paths and the receiver discards duplicated packets. On the other hand, non critical applications may use a redundant path only when the primary one fails.

2.3 TIMELINESS

Packet delivery time or packet latency is variable in asynchronous SpW networks without dedicated links for each data transfer. Packets going to nearby destinations will tend to have lower latency than packets going to more distant destinations. Higher latency implies higher use of bandwidth, as links are being used for longer time. Synchronous networks allow deterministic latency for packets and messages but have a penalty in terms of flexibility and performance. For some applications, the knowledge of the maximum message latency may be sufficient.

Worst case packet latency

Packet latency in asynchronous networks can be very high in some topologies and network traffic. Computation for three simple cases is presented below assuming equal routing priority and round robin mechanism.

a) All links in the destination path are not shared with any other sending entity. Then, the latency will be constant in absence of errors and will depend on the maximum packet size (M), number of hops to the destination (H), link speed (S), and switching delay (Ts):

$$lc = M/S + H \bullet \mathrm{Ts}$$

b) All sending entities send simultaneously a single packet to a unique destination in a network with a simple line topology (Fig 1). The maximum latency corresponds to the furthest possible source destination pair and its value is:

$$l\Big|_{\max} = \sum_{r=1}^{H} (L_r - 2) \bullet \left(M/S + H_r \bullet \mathrm{Ts}\right) + \left(M/S + H_r \bullet \mathrm{Ts}\right)$$

Where *Lr* denotes the number of links in the router *r* and *Hr* is the number of hops to the destination for the nodes attached to router *r*.

c) Same case as b) except that packets are generated continuously. The maximum latency corresponds to the furthest possible source destination pair and its value is:

$$l\Big|_{\max} = \prod_{r=1}^{H} (L_r - 1) \bullet \left(M/S + H_r \bullet \mathrm{Ts}\right)$$

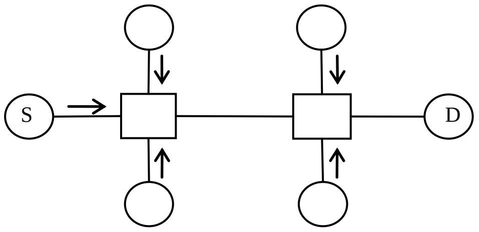


Figure 1: Linear topology with four port routers. The maximum packet latency from S to D is five times the minimum latency if a single packet is sent by each node, and nine times if packets are sent continuously. Note that the worst case is assumed; all nodes send packets to destination D.

Note that it is assumed that packets are consumed at the destination.

The worst case packet latency is not trivial to obtain for arbitrary topologies. Although it may not be possible to express it analytically, it may be computed if the expected network traffic is known. The message latency is usually easily derived from the packet latency.

Time Division Multiplexing (TDM)

Packet latency can be made deterministic and constant by dividing the transmission time in multiple time-slots grouped into epochs. For each timeslot multiple sets of unidirectional links are defined. Each set of links is reserved to be used in this timeslot by a unique source node in such a way that there is no link that is included in the destination path of two different source nodes. Multiple packets can be sent during a timeslot provided that they are delivered before the beginning of the next timeslot. This technique ensures that there is no packet blockage caused by other packets using the same network resources (links).

Scheduled networks using TDM are very efficient when the expected network traffic is periodic and known. Depending on the application it can provide higher throughput and lower message latency than asynchronous networks. However when traffic is sporadic and unknown it tends to provide lower throughput and a message latency that is higher than the average one in asynchronous networks. Nevertheless, the worst case message latency in asynchronous networks may not be acceptable for some applications.

TDM can be easily implemented in SpW networks using time-codes. There are sixty-four possible time-code values that allow up to sixty-four different timeslots to be defined. Time slot configuration may be preset or configured at network setup time. Reconfiguration should be done by a master network manager to ensure the robustness of the system.

Message latency and throughput can be adjusted by assigning more time-slots. They should be equally distributed in the epoch when they are not synchronized with the message production.

2.4 PRIORITIES

Messages usually have different levels of priority. Priority is applied at node and network level over packets containing a message.

Node level

The simplest priority mechanism that can be implemented at node level is to try to send higher priority packets before lower ones. In synchronous networks the highest priority packet within a specific timeslot is resource reserved in this timeslot. Packets with lower priority may be sent only if the higher ones do not need to use the timeslot.

More complex priority mechanisms can be implemented on the top of the previous one. For example, for asynchronous wormhole switching networks, the elastic round robin method [3] provides better fairness, as it takes into account the time used by a channel to send a message.

Network level

Some routers may perform the arbitration taking into account packet priorities. In asynchronous networks this reduces packet latency for high priorities packets but does not provide deterministic timely delivery.

For synchronous networks without a schedule table the following mechanisms may provide determinism using priorities at network level:

a) Master arbitration approach, somehow similar to MIL-STD-1553, based on a master controller periodically polling the terminals.

b) Master arbitration approach in which all nodes are required to send a request to the network master before performing any transaction.

2.5 SEGMENTATION AND ENCAPSULATION

Message segmentation is required to ensure that packets sent over the network have a maximum size. This is a necessary condition to have bounded packet latency and deterministic delivery. Message segments are encapsulated in the transport protocol packet (SpW-RT). The message segments should not contain SpW routing information as this is already handled by the transport protocol.

2.6 END TO END FLOW CONTROL

End to end flow control ensures that there is always space at the destination buffer before sending a data packet. Therefore packet blocking cannot occur when the sender sends data faster than the receiver can process. This is critical to avoid very high network congestion with wormhole switching.

End to end flow control is also important for applications that already provide memory control capabilities, like RMAP, but cannot process multiple requests in parallel. The mechanism guarantees that an application is not deadlocked because of this limitation. When necessary it ensures that a high priority application waits until an ongoing low priority application is processed.

Finally, end to end flow control provides a simple detection mechanism for the sender entity when the destination application is busy or unavailable.

2.7 NETWORK MANAGEMENT

To enforce robustness and reliability, an intelligent node called network manager should be responsible for the whole operation of the network. The network manager may periodically poll routers and nodes to obtain their status, requests and possible notifications. In a scheduled network nodes are usually not allowed to send unscheduled error reports and may need to wait to be polled for the network manager, which should have the most complete and updated network information. Multiple redundant network managers may be present to detect and recover from an error, for example in case the active network manager stops sending Time-Codes.

The network manager is also responsible for configuring the routing tables and required channels parameters. It may also implement FDIR techniques and the Plug and Play protocol.

3 SPW-RT PROTOTYPES

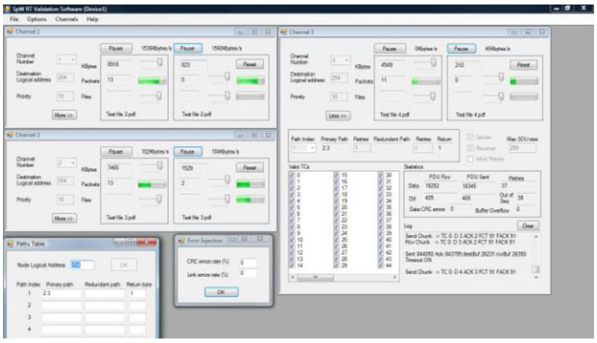
Two software prototypes have been developed for the PC platform using SpW interfaces. The first one is a complete implementation of the first draft of the SpW-RT protocol [4]. It proved that the underlying concepts where valid and it provided and approximation of the performance figures expected. The second prototype was developed to try as many new techniques as possible, making the protocol more efficient but still providing a complete working protocol. The most important results were:

- Bidirectional channels instead of unidirectional channels. It allowed implementing piggybacking for the user data, the acknowledgement and the flow control. It made possible to execute RMAP commands with acknowledgement using only two packets.

- Connection-less protocol. Sequence numbers were reset with a special flag.

- Support for dummy receivers with zero configuration. The return path information was embedded in the packet structure.

- For synchronous systems, the destination buffer space of multiple channels was encapsulated in a single packet to increase the protocol efficiency.



```
Send Chunk -> TC 0 D 3 ACK 2 FCT 91 FACK 91
Rcv Chunk -> TC 0 D 3 ACK 2 FCT 91 FACK 91
Sent 844050 Ack 843795 destBuf 26231 rcvBuf 26350
Timeout ON
Send Chunk -> TC 0 D 4 ACK 3 FCT 91 FACK 91
```

Figure 2: Protocol application layer tool developed to test the SpW-RT prototypes. It can send and receive multiple files concurrently in asynchronous and scheduled networks. It has error injection and the data rates for the generation and consumption of user data can be adjusted dynamically.

4 LESSONS LEARNED

The following considerations were derived from the prototype work and further theoretical analysis:

- Forcing the asynchronous and scheduled networks to have the same unique packet structure for control PDUs increases the complexity of the protocol and reduces its efficiency.

- Bidirectional channels are not efficient for scheduled systems. Moreover, they are more difficult to handle for the final user.

- A connection oriented protocol provides more robustness for high critical applications.

- For better efficiency in scheduled networks, flow control should be send by the receiver just before the sender performs the arbitration. The acknowledgement should be received as soon as possible. Therefore piggybacking the acknowledgement and flow control is not optimum.

- In scheduled networks, sending multiple data packets per timeslot increase the protocol efficiency. 256 bytes per segment and six data packets per timeslot are the optimum values at 200Mbit/s link speed. The efficiency is increased if the data packets are encapsulated in a single SpW packet. Note that typically only one destination per source node is available for each timeslot.

- In order to distinguish between different error cases, it is recommended to send an acknowledge packet even if a SpW error end of packet marker is received. Of course, the sequence number would not acknowledge the data packet with errors.

5 CONCLUSIONS AND FUTURE WORK

SpW-RT prototyping efforts have played an important role in the design of the SpW-RT protocol, providing important information about the advantages and disadvantages of different strategies.

Future efforts will target the prototyping of the lastest version of the SpW-RT protocol for EGSE and space qualified components. Moreover, network design tools and reference architectures with different user cases will be developed to facilitate the deployment of systems based on SpW-RT.

6 REFERENCES

[1] ECSS, “SpaceWire: Links, nodes, routers and networks”, ECSS-E50-12A, January 2003 SpW RMAP

[2] S.M. Parkes, “SpaceWire-RT Requirements”, SpaceNet Report No. SpW-RT WP3-100.1, ESA Contract No. 220774-07-NL/LvH, February 2008

[3] S.M. Parkes and A. Ferrer-Florit, “SpaceWire-RT Initial Protocol Definition”, Draft 1.0, SpaceNet Report No. SpW-RT WP3-200.1, ESA Contract No. 220774-07-NL/LvH, March 2008.

[4] S.S. Kanhere, H. Sethu and A.B. Parekh, “Fair and Efficient Packet Scheduling Using Elastic Round Robin“. IEEE transactions on parallel and distributed systems, vol 13, no 3, march 2002.