THE GEOSTATIONARY OPERATIONAL SATELLITE R SERIES SPACEWIRE BASED DATA SYSTEM ARCHITECTURE

Session: SpaceWire Missions and Applications
Long Paper

Alexander Krimchansky - NASA Goddard Space Flight Center
William H. Anderson, Craig Bearer - MEI Technologies
E-mail: Alexander.Krimchansky@nasa.gov, William.H.Anderson@nasa.gov, Craig.M.Bearer@nasa.gov

ABSTRACT

The GOES-R program selected SpaceWire as the best solution to satisfy the desire for simple and flexible instrument to spacecraft command and telemetry communications. Data generated by GOES-R instruments is critical for meteorological forecasting, public safety, space weather, and other key applications. In addition, GOES-R instrument data is provided to ground stations on a 24/7 basis. GOES-R requires data errors be detected and corrected from origin to final destination. This paper describes GOES-R developed strategy to satisfy this requirement.

1. INTRODUCTION

The Geostationary Operational Environmental Satellite-R Series (GOES-R) program is a key element of the National Oceanic and Atmospheric Administration's (NOAA) operations. As such, the GOES-R and follow on series of satellites will be comprised of improved spacecraft and instrument technologies, which will result in more timely and accurate weather forecasts, and improve support for the detection and observations of meteorological phenomena that directly affect public safety, protection of property, and ultimately, economic health and development. The first launch of the GOES-R series satellite is scheduled for 2015. The GOES-R spacecraft uses European Cooperation for Space Standardization (ECSS) SpaceWire [1] for the transfer of sensor, telemetry, ancillary, command, time code, and time synchronization information between instruments and the spacecraft.

2. RELIABLE DATA DELIVERY PROTOCOL

GOES-R project has developed a Reliable Data Delivery Protocol (GRDDP) that is based on SpaceWire capabilities for link connection and re-connection, error detection, virtual channels and routing. This protocol has been presented to and accepted by the SpaceWire Working Group [2] and assigned a Protocol ID (PID) 238. GRDDP also known as PID 238 does not attempt to duplicate or improve on the considerable capabilities provided by SpaceWire. This protocol builds on top of SpaceWire the ability to recover lost packets, reorder packets, and to ensure to higher level processes that packets are as error free as possible.

GOES-R requirements for PID 238 are to utilize SpaceWire capabilities to provide a packet delivery protocol that is able to detect and recover lost packets. The protocol is also required to be flexible so that it can be adapted as needed to different host data throughput requirements and resources. PID 238 intentionally does not specify an implementation. It

defines a set of capabilities, but does not require that all capabilities be implemented for all applications.

PID 238 is based on the concept of "Virtual Channels" similar to the virtual channels identified in the SpaceWire specification. Any number of virtual channels can coexist on a single SpaceWire link. In PID 238, all channels are completely independent. PID 238 defines virtual channels as a pair of Transport End Points (TEPs). Each TEP is identified by a SpaceWire Logical Address (SLA) and interfaces to a SpaceWire link to send and receive PID 238 packets. Each PID 238 virtual channel transmits data packets in only one direction, thus each channel consists of one Transmit TEP and one Receive TEP. PID 238 protocol is completely specified in terms of the behavior of a Transmit TEP and a Receive TEP. A host will have only one TEP for each virtual channel that it supports. Note that a Transmit TEP sends data packets, but also receives acknowledge packets from the remote Receive TEP. A Receive TEP receives data packets, but also transmits acknowledge packets.

PID 238 specifies a packet format that is consistent with the standard SpaceWire packet. Specifically a packet terminated with an End of Packet (EOP) or Error End of Packet (EEP) character. However, where the SpaceWire standard allows a path address of zero or more bytes, PID 238 requires that exactly one destination SLA be delivered to PID 238 logic at the destination. That destination SLA identifies the virtual channel that is to receive the PID 238 packet. PID 238 does not dictate packet routing through a SpaceWire network be it point-to-point or composed of multiple routers. Packet routing is handled by the SpaceWire layer. PID 238 does not try to improve on SpaceWire packet routing. The one-byte destination SLA is sufficient to get a SpaceWire packet to its destination through a variety of network configurations.

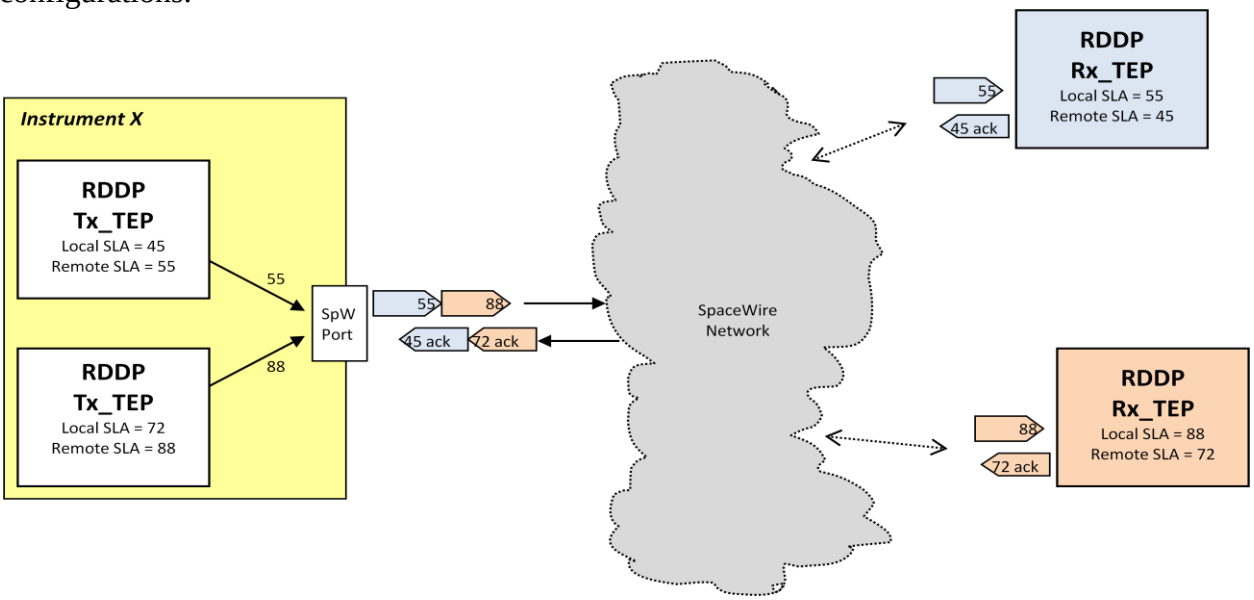

Figure 1. PID 238 Packet Routing

Figure 1 illustrates PID 238 perspective for packet routing. Each of the several virtual channels that may reside on a host has associated with it the destination SLA for that channel's remote TEP. For PID 238, there is no "path" to the remote TEP, only a destination SLA. PID 238 does not have, and does not require, knowledge of how a packet gets to its destination. This keeps PID 238 simple, and allows any implementation using this protocol

to remain independent of possible SpaceWire network changes, and unaffected if a remote TEP for a channel is re-configured to reside on a different host. Figure 2 shows PID 238 format and details can be found in the protocol's document [3].

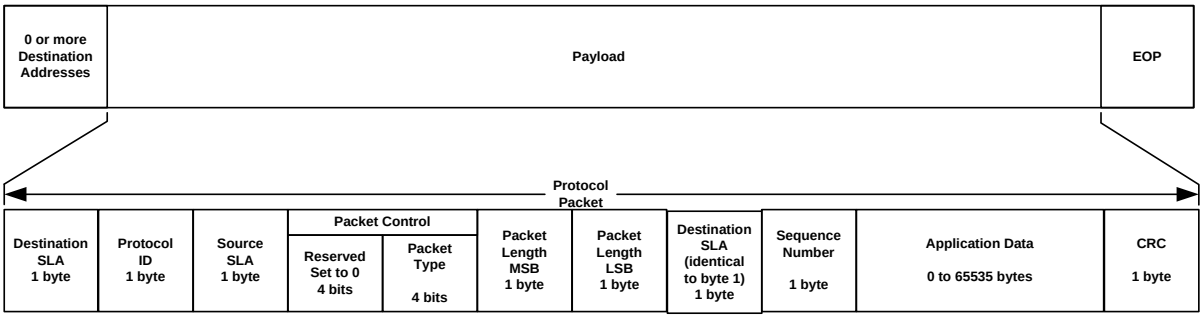

Figure 2. PID 238 Packet Format Inside a SpaceWire Packet

Key features of PID 238 are:

- Defines 4 kinds of packets
- Data Packets have Sequence Numbers from 0 to 255
- ACK Sequence Numbers must match the source Data Packet
- The Urgent Message Sequence Number is always 0
- The Reset Packet Sequence Number is always 0

3. PID 238 OPERATION OVER SPACEWIRE

The TEP for each channel is in one of three possible states Closed, Enabled, or Open. By default, all channels are in the Closed state until the host sets the channel to Enabled. A channel that is in the Closed state will not transmit any packets, and will not acknowledge or process any packets received. When in the Enabled state, a transmit TEP will send only Reset packets to the remote receive TEP. When an Enabled receive TEP gets the Reset Packet, it changes to the Open state and then sends an ACK for the reset. When the transmit TEP receives the ACK it changes to the Open state. The Reset and ACK packets are what lets both ends of the channel know that the other end is "open", or active. The reset also serves to synchronize the "next" Sequence Number expected for both ends of the channel. Note that only the transmit side can open a channel. An enabled receive TEP can only sit back and wait for the transmitter to become active (or enabled) and send the Reset Packet. Until the transmitter is ready to start sending Data Packets there is no reason for the receiver to do anything.

PID 238 provides several capabilities that work together to provide the reliable data delivery that is required for GOES-R.

PID 238:

- Detects lost packets by using positive acknowledge for each packet transmitted.
- Recovers lost packets using an ACK timeout and retransmit.
- Re-orders received packets and removes duplicates using sequence numbers.
- Maximizes data packet throughput using a sliding window range of sequence numbers that allows a transmitter to continue sending packets while waiting for acknowledgements

Each of these capabilities can be independently adapted for each channel so that PID 238 can be adapted to the data throughput and reliability requirements for each data stream.

PID 238 uses a positive acknowledge for each packet transmitted. If an ACK is not received within a timeout interval, that packet is retransmitted. After a maximum number of retries have been exhausted for a packet, the transmitter will declare that virtual channel (and only that channel) closed. The requirement for the receiver to acknowledge each packet allows the transmitter to detect lost packets if an ACK is not received. On the receiver end, a packet is acknowledged if it has a valid PID 238 header and Cyclic Redundancy Check (CRC) character.

In the case where data packets represent a "current" value that changes at a high rate. In this case there is no point in re-transmitting an un-acknowledged old packet when a new packet is available. PID 238 defines an "Urgent Message" packet type that does not need to be acknowledged, and is delivered to the host at a higher priority than the normal data packets. In order to keep things simple, the Urgent Messages do not require a separate channel. Urgent Messages can be intermixed with normal data packets on a single channel.

In order to maximize throughput, PID 238 defines a moving window range of sequence numbers that allows the transmitter to transmit ahead while waiting for ACKs. The size of the window range limits the number of packets that can be transmitted by a Transmit TEP while waiting for an ACK. At the Receive TEP, the window is used to eliminate duplicate packets, and to ensure packets are delivered in correct sequence to the host.

A channel that is required to transmit high rate data would use a large window so that a large number of data packets can be sent ahead of receipt of the ACKs. If an ACK is not received within a timeout interval, the transmit TEP will retransmit only that packet. The Receive TEP implements the same size window to order the packets delivered to the host and will deliver the retransmitted packet in correct sequence. The moving window, and the associated memory buffers required for retransmission of old packets and for re-ordering received packets does require some additional logic and memory resources. The amount of memory required to buffer moving window data packets depends on the maximum packet size and the window size.

In a case where a PID 238 channel does not require a high data rate, the window size can be set to one. A window size of one provides a synchronous transfer where the transmitter waits for each packet to be acknowledged before sending another. Finally the simplest PID 238 implementation is used when data is updating at some known rate and fresh data will quickly replace lost or stale data. This is the Urgent Message service. Urgent Message does not require moving window processing or an ACK packet. It is straight forward fire-and-forget.

PID 238 Mode	Usage
Full Mode Window Size >1	High Data Rate
Full Mode Window Size = 1	Low Data Rate
Urgent Message Window Size = 0	Low Latency

Table 1. PID 238 Usage

In summary, PID 238 specifies the behavior of Transmit and Receive TEPs. Transmit TEPs will transmit only Data and Reset packets. A transmit TEP will never receive a data packet or a reset, and will never transmit an ACK. A Receive TEP will transmit only ACKs, and will receive only data packets or resets. A Receive TEP will never transmit a Data or Reset packet. The rules for when a Transmit TEP sends Data or Reset packets and a Receive TEP sends an ACK are not complicated. Much of PID 238 flexibility for adapting to different instrument requirements has to do with the number of transmit or receive channels, whether Urgent Messages are used, and the window size selected. An instrument with very low rate data requirements might elect to implement one transmit TEP and one receive TEP (only one channel in each direction), and to use a window size of one for both the transmit and receive channels. In this case, PID 238 would provide synchronous data transfers where each packet transmitted must be acknowledged before another can be sent. With a window size of one, the need for transmit or receive buffers is minimal, and there is no need to implement the logic to re-order packets. At a minimum, PID 238 provides multiple virtual channels that can be independently routed via the SpaceWire network, and improved error detection through the CRC.

4. SPACEWIRE AND PID 238 TEST SYSTEM

GOES-R project has maintained a SpaceWire test lab [4], initially to validate PID 238, to evaluate varying the parameters that might affect performance, and for validating different proposed options and configurations that could be used on the GOES-R spacecraft. The GOES-R test system has 2 SpaceWire implementations. The first is a GOES-R test card utilizing the British Aerospace (BAE) SpaceWire Application Specific Integrated Circuit (ASIC). And the other is a Xilinx Field Programmable Gate Array (FPGA) commercial off-the-shelf card. The FPGA card is programmed with a modified version of the Goddard SpaceWire core that removed the SpaceWire worm hole router. Additionally, the FPGA core provides additional diagnostic and error injection capabilities. These test cards reside in off-the-shelf Windows workstations and the initial test system is shown in Figure 3.

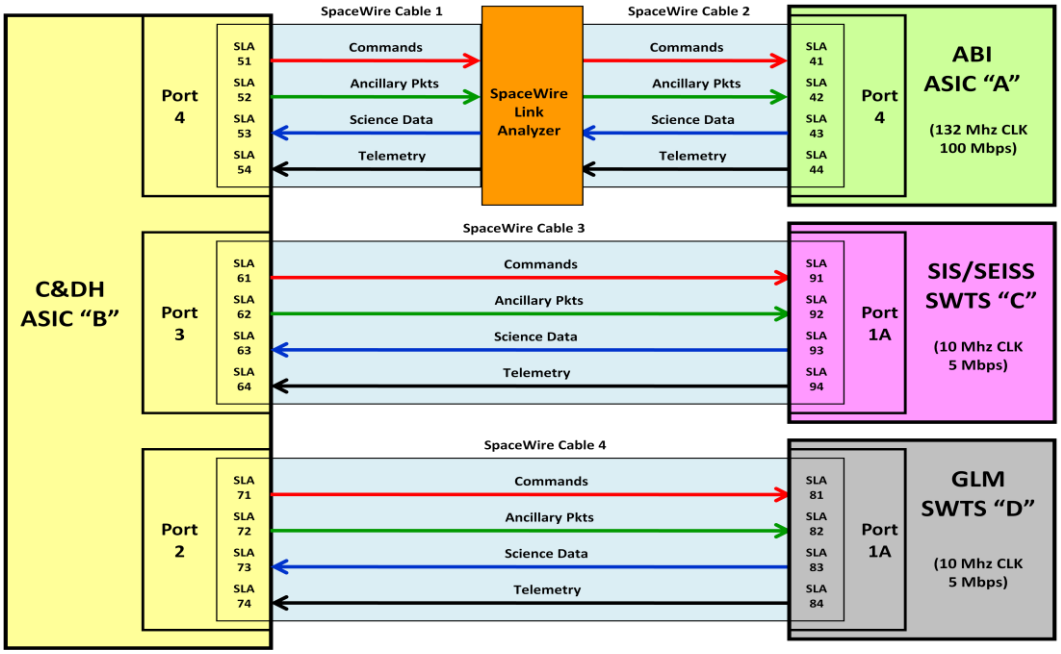

Figure 3. PID 238 Over SpaceWire Original Test System

In all cases, where the BAE SpaceWire ASIC is used to simulate an instrument or the Command and Data Handling (C&DH) system, the same Embedded Microcontroller (EMC) code, including tunable parameters, is used for each test article. PID 238 software implementation is approximately 600 lines of code. Depending on the packet rate, Central Processing Unit (CPU) utilization is in the single digit percentage range on the EMC clocked at 33Mhz. The EMC is capable of operating at clock rates up to 80Mhz.

The probability of an error occurring during transmission of a packet depends on the size of the packet. The ability to recover a lost packet by retransmitting also depends on the size of the packet. Small packets require retransmission of only a small packet making it easier to insert into a data stream. Where large packets have a larger impact and may require more bandwidth margin. GOES-R testing has successfully recovered lost packets, of various sizes, when the normal data stream uses over 90% of the available bandwidth.

5. Next Steps

Of the 5 GOES-R instruments, 2 have implemented PID 238 in FPGAs, and the other three have implemented the protocol in software on the embedded microcontroller in the BAE SpaceWire ASIC. Each of the GOES-R instruments are implementing the SpaceWire and PID 238 interface as a point-to-point architecture. Modeling the proposed spacecraft data system has shown no changes are required in any instrument implementation including the addition of several SpaceWire routers.

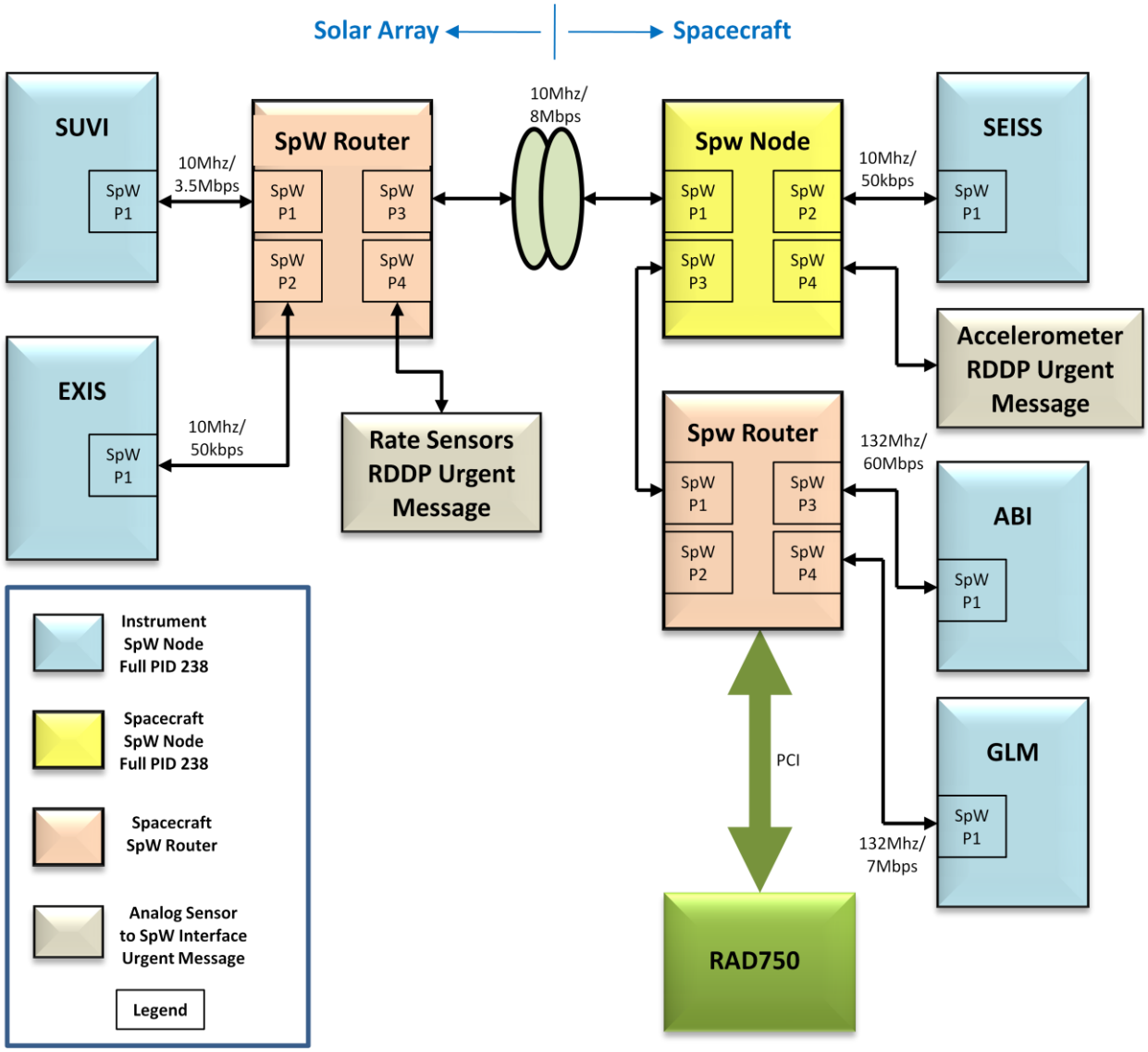

Figure 4. Simplified Spacecraft Design with Multiple SpaceWire Routers

The most simple instrument with very small data throughput requirements and minimal processor resources, the largest instrument with the highest data throughput requirements, and the spacecraft C&DH that interfaces to them all have implemented PID 238 to the same specification. All of the instruments as well as the spacecraft recognize a common method for detecting and recovering data link errors and lost packets. GOES-R has several years experience exercising SpaceWire and PID 238 in a variety of configurations. In all cases SpaceWire and PID 238 have been found to be robust, efficient, and flexible. PID 238 over SpaceWire is documented, tested, and available for use in space data system applications.

6. CONCLUSION

PID 238 was developed to satisfy data system requirements for all GOES-R instruments. The instruments have a wide range of electronics implementations from simple to complex, and a wide range of data rates. SpaceWire transmit clock rates operate at either 10MHz or 132MHz. GOES-R instrument data rates ranging from 50kb to 66MHz are easily managed by the combination of PID 238 over SpaceWire. Many parameters of PID 238 can be tuned to match the reliability requirements and a node's ability to support the required complexity. PID 238 has proven able to adapt to those capabilities and data rates due to its inherent flexibility. PID 238 is documented and extensively tested. It is available and ready to be applied to SpaceWire applications.

7. REFERENCES

1. European Cooperation for Space Standardization, ECSS-E-50-12A, "SpaceWire – Links, Nodes, Routers and Networks", 24 January 2003

2. European Cooperation for Space Standardization, ECSS-E-ST-50-51C, "SpaceWire Protocol Identification", 5 February 2010

3. NASA Goddard Space Flight Center GOES-R Project "GOES-R Reliable Data Delivery Protocol", 417-R-RTP-0050 Version 2.1, 16 January 2008

4. William Anderson MEI Technologies Inc., GOES-R Project, "Reliable Data Delivery Protocol (RDDP)", 2006 MAPLD International Conference, Washington, D.C., September 25, 2006