Design and Prototype of SpaceWire Spacecraft Bus for Solar Probe Plus

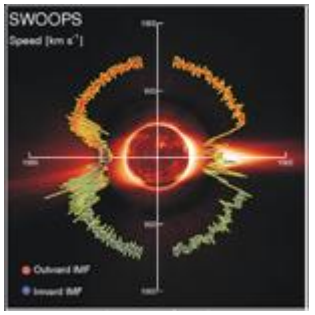
Alan A. Mick, Joseph R. Hennawy, Christopher J. Krupiarz, Horace Malcom, Dan Rodriguez

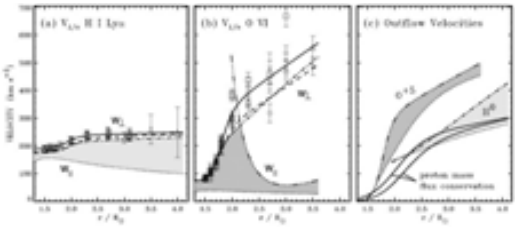
This presentation does not contain ITAR restricted information.

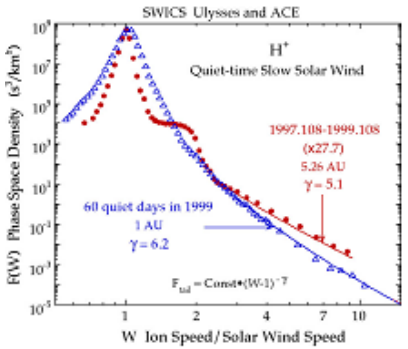
APL
The Johns Hopkins University
APPLIED PHYSICS LABORATORY

Solar Probe Plus Science Objectives

- **Primary Science Goal:**
  - **Determine structure and dynamics of Sun's coronal magnetic field, understand how the solar corona and wind are heated and accelerated, determine what mechanisms accelerate and transport energetic particles**
- **The primary SPP mission science goal defines three overarching science objectives**
  - **Trace the flow of energy that heats and accelerates the solar corona and solar wind**
  - **Determine the structure and dynamics of the plasma and magnetic fields at the sources of the solar wind**
  - **Explore mechanisms that accelerate and transport energetic particles**







Solar Probe Plus – Science Payload

- **SWEAP - count the most abundant particles in the solar wind -- electrons, protons and helium ions -- and measure their properties such as velocity, density, and temperature**
- **FIELDS - direct measurements of electric and magnetic fields and waves, Poynting flux, absolute plasma density and electron temperature, spacecraft floating potential and density fluctuations, and radio emissions.**
- **WISPR - wide-field imager will take images of the solar corona and inner heliosphere. The telescope will also provide images of the solar wind, shocks and other structures as they approach and pass the spacecraft.**
- **ISIS - observations of energetic electrons, protons and heavy ions that are accelerated to high energies (10s of keV to ~100 MeV) in the Sun's atmosphere and inner heliosphere, and correlates them with solar wind and coronal structures.**

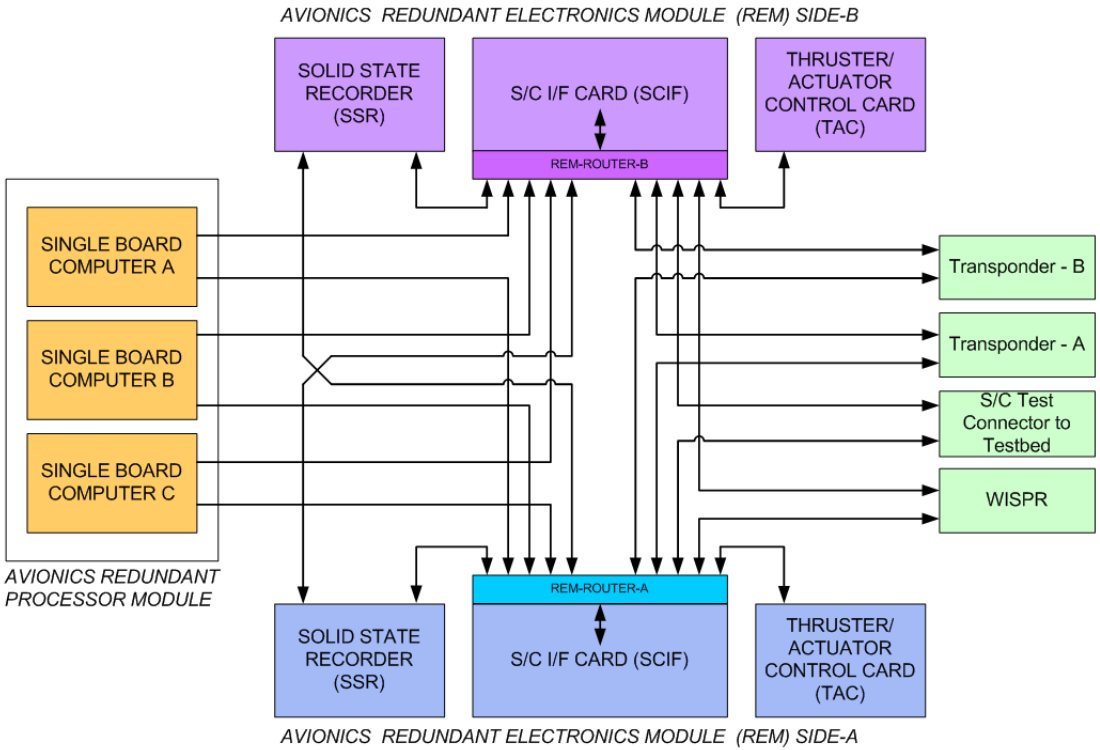
Spacecraft Overview

- Three Axis Stabilized
  - Wheels + Thrusters for Momentum Dumping.
- Ceramic Coated Carbon-Carbon Thermal Protection Shield
- Actively Cooled Solar Power System
  - Water Cooled Solar Array Substrates
  - Mechanical Pump Loop
  - Radiator area under TPS
- Design Drivers:
  - Solar Environment
  - Mass
  - Power

Design Drivers

- Active cooling requires mass and consumes power.
- Solar arrays must be minimally exposed, thus limiting power.
- Thermal protection shield must be pointed towards sun – especially during perihelion science operations.
- Solar pressure torque due to offset between center of pressure and mass tends to offpoint the TPS.
- Maintaining attitude control is very important, even through severe faults.

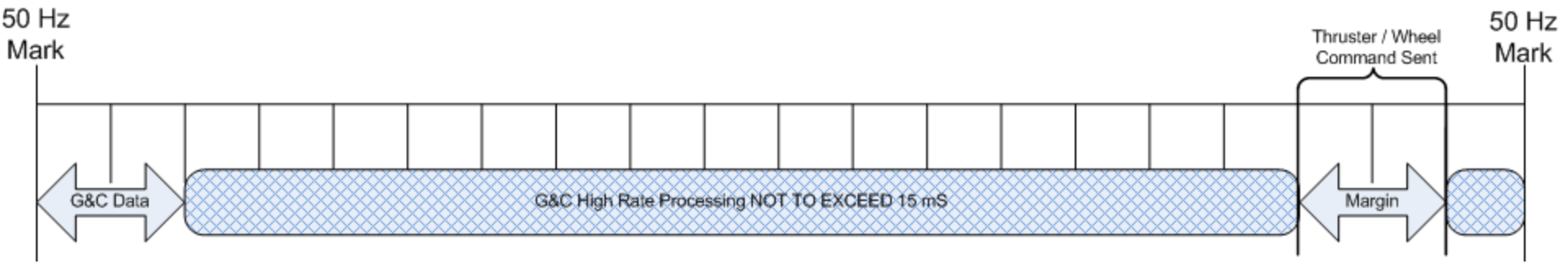
Solar Probe Avionics and SpaceWire Network



- **SpaceWire Selected over 1553:**
  - **Greater Bandwidth**
  - **Lower Emissions**
- **Redundant Processor Module**
  - **Prime, Hot Spare, Warm Spare**
- **Redundant Electronics Modules**
  - **SSRs are Cross Strapped**
- **Two Cross Strapped Transponders**

“Bus Schedule” Requirements

- SpaceWire is not synchronous and deterministic as is the 1553 bus.
- SPP has rigorous response time requirements for G&C.
  - 50 Hz = 20 mS G&C high rate control schedule
  - 15 mS for G&C high rate processing, 1 mS margin for TAC reception and latching, leaves 4 mS for G&C data collection over SpaceWire
- SPP is power constrained – use lowest link rate feasible



Challenge: Implement a predictable, deterministic “bus schedule” in an asynchronous, network infrastructure.

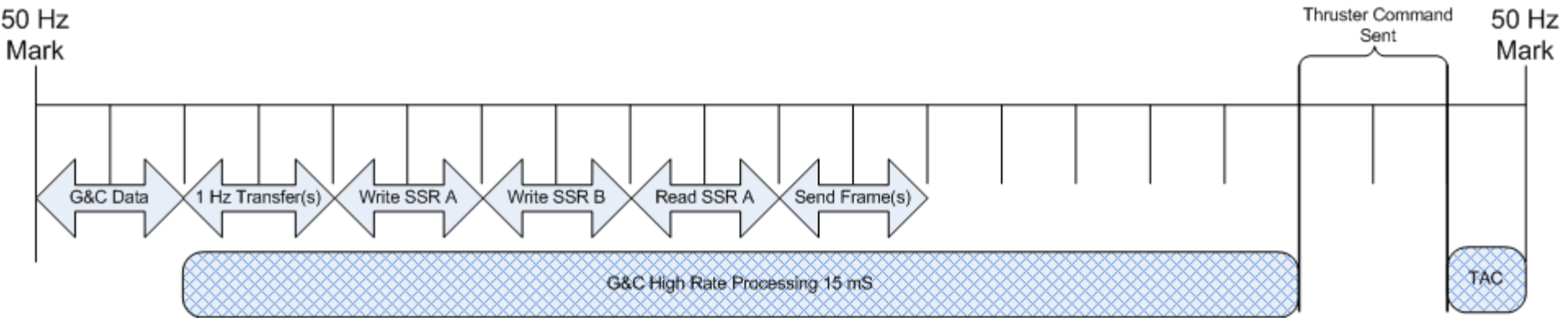
SpaceWire-D and RMAP

- SpaceWire-D is a protocol being developed to implement deterministic response times and assured throughputs over SpaceWire.
- SpaceWire D is built on the SpaceWire time code capability and utilizes the SpaceWire Remote Memory Access Protocol (RMAP).
- The SpaceWire time codes provide time divisions for multiplexing, while the use of RMAP allows deterministic control within a division.
- Difficulties:
  - Development is in the conceptual phase – no known commercial or qualified implementations.
  - Requires a high interrupt rate that would saturate our LEON3 processors.
  - Would be a complex “DIY” project for a specific mission.

- However, we adopted the use of SpaceWire time codes to create 50 Hz partitions and the use of RMAP to control transactions within those partitions.

SpaceWire "Bus" Design Concept

- SpaceWire time codes are generated and distributed by the SCIF / router card
- The SBC Prime uses RMAP to initiate (almost) all data transfer transactions over SpaceWire
- As the initiator, the SBC Prime prioritizes data transfers within each 50 Hz / 20 mS frame to ensure response time and throughput requirements are met:
  - G&C high rate sensor data is collected and G&C processing is initiated
  - One to four 1 Hz command and telemetry transfers are initiated
  - Optionally, data is written to SSR A and B
  - Optionally, data is read from SSR A or B (not both)
  - Optionally, up to two telemetry frames are transferred to transponder A or B (not both).
  - Thruster / Wheel command is transferred to the TAC when ready. This can be done at any time before initiating the next scheduled transfer.



Transaction Conventions

- Almost all transactions are initiated by the SBC Prime using the RMAP protocol. The SBC Prime is the virtual “bus master”.
- The SBCs (Prime, Hot Spare, Cold Spare) will not automatically respond to RMAP requests. The SBC SpaceWire ports will have RMAP disabled.
- Almost all RMAP transactions are structured using a “uniform component buffer” transfer convention. The SSR transactions do not conform to this convention.
- The downlink frame request and telecommand SpaceWire packets are sent asynchronously (relative to the bus schedule) by the transponders using normal SpaceWire packets.

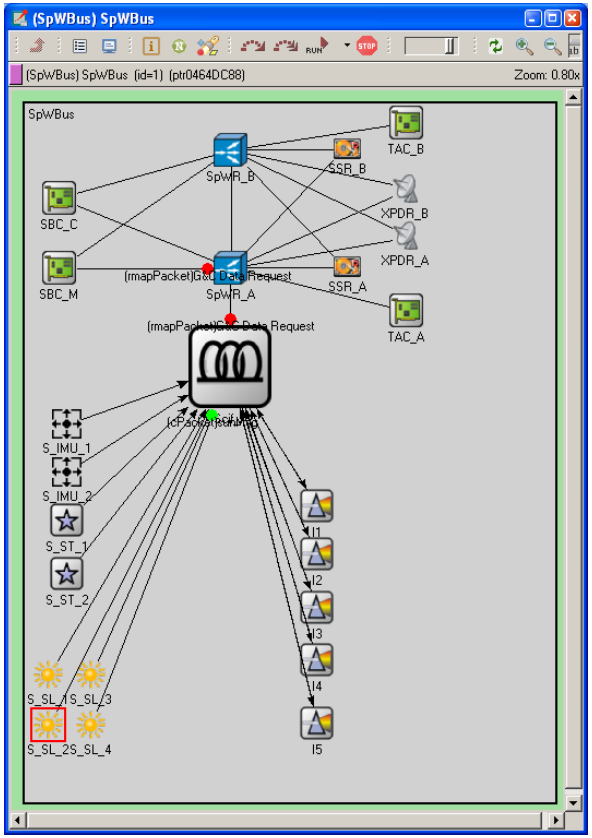
Uniform Component Buffer

- To read telemetry from a 50 Hz (G&C) or a 1 Hz (instrument, subsystem) buffer, two RMAP operations are performed.
- First a register (at a well known address) is read to get the size of the telemetry buffer.
- Then the size is used to read the buffer itself.
- If a particular card (such as the SCIF) has multiple buffers, the size registers can be made contiguous so that one read can be performed to get all the sizes at once.

SSR Interface

- The SSRs follow a “memory mapped” device model.
- One or more control registers are set up using an RMAP write, then
- Data is either read or written by using an RMAP transfer.
- This interface has been successfully prototyped over SpaceWire using our SSR development prototypes.

Discrete Event Model



- A discrete event model of the SpaceWire bus and the transactions has been developed
- Adds SpaceWire and RMAP protocol overhead (headers, ACKs) to the estimates
- Accounts for scheduling inefficiencies
- Allows “bus schedule” to be developed experimentally
- Modeling indicates margins are more than adequate for Phase A

Example Bus Schedule

FRAME	G&C	One Hz Slot 1	One Hz Slot 2	One Hz Slot 3	One Hz Slot 4	Write SSRs	Read SSR X	Write Downlink Frames
Frame 00	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL	X	X	X, X, X
Frame 01	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL		X	X, X
Frame 02	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL	X	X	X, X
Frame 03	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL			X, X
Frame 04	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL	X	X	X, X, X
Frame 05	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL			X, X
Frame 06	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL	X	X	X, X
Frame 07	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL			X, X
Frame 08	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL	X	X	X, X, X
Frame 09	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL			X, X
Frame 10	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL	X	X	X, X
Frame 11	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL		X	X, X
Frame 12	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL	X	X	X, X, X
Frame 13	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL			X, X
Frame 14	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL	X	X	X, X
Frame 15	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL			X, X
Frame 16	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL	X	X	X, X, X
Frame 17	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst1},	NULL	NULL	NULL			X, X
Frame 18	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst2}	NULL	NULL	NULL	X	X	X, X
Frame 19	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst2}	NULL	NULL	NULL			X, X
Frame 20	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst2}	NULL	NULL	NULL	X	X	X, X, X
Frame 21	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst2}	NULL	NULL	NULL		X	X, X
Frame 22	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst2}	NULL	NULL	NULL	X	X	X, X
Frame 23	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst2}	NULL	NULL	NULL			X, X
Frame 24	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst2}	NULL	NULL	NULL	X	X	X, X, X
Frame 25	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst2}	NULL	NULL	NULL			X, X
Frame 26	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst4},	NULL	NULL	NULL	X	X	X, X
Frame 27	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst4},	NULL	NULL	NULL			X, X
Frame 28	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst4},	NULL	NULL	NULL	X	X	X, X, X
Frame 29	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst4},	NULL	NULL	NULL			X, X
Frame 30	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst5},	NULL	NULL	NULL	X	X	X, X
Frame 31	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst5},	NULL	NULL	NULL		X	X, X
Frame 32	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst5},	NULL	NULL	NULL	X	X	X, X, X
Frame 33	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompInst5},	NULL	NULL	NULL			X, X
Frame 34	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, WRITE, CompInst1},	{SCIF_A, WRITE, CompInst2},	{SCIF_A, WRITE, CompInst3},	NULL	X	X	X, X
Frame 35	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, WRITE, CompInst4},	{SCIF_A, WRITE, CompInst5},	{SCIF_A, READ, CompPDU, CompPD	NULL			X, X
Frame 36	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompPSE, CompPSE	{SCIF_A, READ, CompCool},	{SCIF_A, READ, CompRIOs},	NULL	X	X	X, X, X
Frame 37	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{XPD_A, READ, CompXpndrA},	{XPD_B, READ, CompXpndrB},	{SSR_A, READ, CompSSRA},	NULL			X, X
Frame 38	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SSR_B, READ, CompSSRB},	{SBC_C, READ, CompSBC_HS},	{TAC_A, READ, CompProp},	NULL	X	X	X, X
Frame 39	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{TAC_A, READ, CompSA_HGA},	{SCIF_A, WRITE, CompPDU},	{SCIF_A, WRITE, CompPSE},	NULL			X, X
Frame 40	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, WRITE, CompCool},	{XPD_A, WRITE, CompXpndrA},	{XPD_B, WRITE, CompXpndrB},	NULL	X	X	X, X, X
Frame 41	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SSR_A, WRITE, CompSSR},	{SSR_B, WRITE, CompSSR},	{SBC_C, WRITE, CompSBC_HS},	NULL		X	X, X
Frame 42	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{TAC_A, WRITE, CompProp},	{TAC_A, WRITE, CompSA_HGA},	{SCIF_A, WRITE, CompImu1},	NULL	X	X	X, X
Frame 43	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, WRITE, CompStarTracker1	{SCIF_A, WRITE, CompStarTracker2	{SCIF_A, WRITE, CompSunSensor1}	NULL			X, X
Frame 44	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, WRITE, CompSunSensor2}	{SCIF_A, WRITE, CompSunSensor3}	{SCIF_A, WRITE, CompSunSensor4}	NULL	X	X	X, X, X
Frame 45	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	NULL	NULL	NULL	NULL			X, X
Frame 46	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	NULL	NULL	NULL	NULL	X	X	X, X
Frame 47	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	NULL	NULL	NULL	NULL			X, X
Frame 48	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	NULL	NULL	NULL	NULL	X	X	X, X, X
Frame 49	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	NULL	NULL	NULL	NULL			X, X

APL

Detail of Bus Schedule

FRAME	G&C	One Hz Slot 1	One Hz Slot 2	One Hz Slot 3	Write SSRs	Read SSR X	Write Downlink Frames
Frame 34	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, WRITE, CompInst1},	{SCIF_A, WRITE, CompInst2},	{SCIF_A, WRITE, CompInst3},	X	X	X, X
Frame 35	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, WRITE, CompInst4},	{SCIF_A, WRITE, CompInst5},	{SCIF_A, READ, CompPDU, CompPDU},			X, X
Frame 36	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, READ, CompPSE, CompPSE	{SCIF_A, READ, CompCool},	{SCIF_A, READ, CompRIOs},	X	X	X, X, X
Frame 37	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{XPD_A, READ, CompXpndrA},	{XPD_B, READ, CompXpndrB},	{SSR_A, READ, CompSSRA},			X, X
Frame 38	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SSR_B, READ, CompSSRB},	{SBC_C, READ, CompSBC_HS},	{TAC_A, READ, CompProp},	X	X	X, X
Frame 39	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{TAC_A, READ, CompSA_HGA},	{SCIF_A, WRITE, CompPDU},	{SCIF_A, WRITE, CompPSE},			X, X
Frame 40	{SCIF_A, READ, IMU, ST}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, WRITE, CompCool},	{XPD_A, WRITE, CompXpndrA},	{XPD_B, WRITE, CompXpndrB},	X	X	X, X, X
Frame 41	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SSR_A, WRITE, CompSSR},	{SSR_B, WRITE, CompSSR},	{SBC_C, WRITE, CompSBC_HS},		X	X, X
Frame 42	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{TAC_A, WRITE, CompProp},	{TAC_A, WRITE, CompSA_HGA},	{SCIF_A, WRITE, CompImu1},	X	X	X, X
Frame 43	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, WRITE, CompStarTracker1	{SCIF_A, WRITE, CompStarTracker2	{SCIF_A, WRITE, CompSunSensor1},			X, X
Frame 44	{SCIF_A, READ, IMU}, {TAC_A, READ, SLS1, SLS2, SLS3, SLS4}	{SCIF_A, WRITE, CompSunSensor2}	{SCIF_A, WRITE, CompSunSensor3},	{SCIF_A, WRITE, CompSunSensor4}	X	X	X, X, X

Conclusion

- The SpaceWire design concept successfully supports Solar Probe critical requirements
  - Magnetic cleanliness
  - Throughput requirements
  - Responsiveness for attitude control
  - Flexibility for three redundant processors crossed with two redundant electronics strings
- Risk reduction and technology development activities have demonstrated soundness of approach.