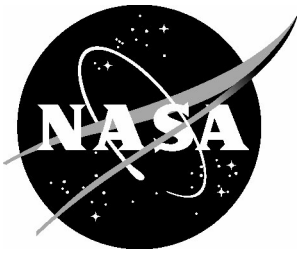
NASA/TM-2008-215322



Design of Test Articles and Monitoring System for the Characterization of HIRF Effects on a Fault-Tolerant Computer Communication System

Wilfredo Torres-Pomales, Mahyar R. Malekpour, Paul S. Miner, and Sandra V. Koppen
Langley Research Center, Hampton, Virginia

July 2008

The NASA STI Program Office . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the lead center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers, but having less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that complement the STI Program Office's diverse offerings include creating custom thesauri, building customized databases, organizing and publishing research results ... even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at ***http://www.sti.nasa.gov***

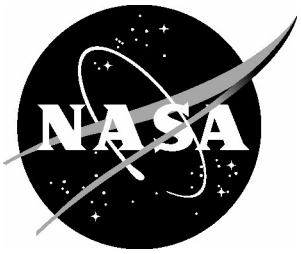
- E-mail your question via the Internet to help@sti.nasa.gov

- Fax your question to the NASA STI Help Desk at (301) 621-0134

- Phone the NASA STI Help Desk at (301) 621-0390

- Write to:
  NASA STI Help Desk
  NASA Center for AeroSpace Information
  7115 Standard Drive
  Hanover, MD 21076-1320

NASA/TM-2008-215322



Design of Test Articles and Monitoring System for the Characterization of HIRF Effects on a Fault-Tolerant Computer Communication System

Wilfredo Torres-Pomales, Mahyar R. Malekpour, Paul S. Miner, and Sandra V. Koppen
Langley Research Center, Hampton, Virginia

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

July 2008

Acknowledgment

The authors are grateful for the contributions of the following individuals to the definition and implementation of this test: Celeste M. Belcastro, Eric G. Cooper, Jay Ely, Dr. Oscar R. Gonzalez, Dr. W. Steven Gray, John J. Mielnik, Jr., Truong X. Nguyen, Maria Theresa Salud, and Laura J. Smith.

Available from:

NASA Center for AeroSpace Information (CASI)
7115 Standard Drive
Hanover, MD 21076-1320
(301) 621-0390

National Technical Information Service (NTIS)
5285 Port Royal Road
Springfield, VA 22161-2171
(703) 605-6000

Abstract

This report describes the design of the test articles and monitoring systems developed to characterize the response of a fault-tolerant computer communication system when stressed beyond the theoretical limits for guaranteed correct performance. A high-intensity radiated electromagnetic field (HIRF) environment was selected as the means of injecting faults, as such environments are known to have the potential to cause arbitrary and coincident common-mode fault manifestations that can overwhelm redundancy management mechanisms. The monitors generate stimuli for the systems-under-test (SUTs) and collect data in real-time on the internal state and the response at the external interfaces. A real-time health assessment capability was developed to support the automation of the test. A detailed description of the nature and structure of the collected data is included. The goal of the report is to provide insight into the design and operation of these systems, and to serve as a reference document for use in post-test analyses.

Table of Contents

1. Overview

The development of the systems presented in this report was supported by the Integrated Vehicle Health Management (IVHM) project under NASA's Aviation Safety Program. Among the objectives of the IVHM project is the development of failure databases and test capabilities suitable for the design of advanced health management systems. In the IVHM project there is also interest in the development of enabling technologies for the design of large-scale robust and reliable distributed processing architectures for vehicle-wide health assessment and management functions. To contribute to these efforts, a test was conceived involving an existing fault-tolerant data communication system exposed to an adverse environment while a real-time monitoring system collects data on the observed effects [1]. The test is intended not only to generate data on the direct effects of the environment, but more importantly, to stress the redundancy management mechanisms of the fault tolerant system beyond the theoretical bounds for guaranteed performance to discover the actual limits of the system implementation and expose weaknesses in the design. The fault-tolerant system selected for this test is a prototype of ROBUS-2 [2, 3], a version of the communication system of the Scalable Processor Independent Design for Extended Reliability (SPIDER) architecture [4]. A high-intensity radiated electromagnetic field (HIRF) environment was selected for the test, as such environments are known to have the potential to cause arbitrary fault manifestations and coincident common-mode faults that can overwhelm current redundancy management mechanisms [5, 6]. The monitoring system is designed to collect data from the interfaces and the internal state of ROBUS-2 that is suitable for detecting faulty behavior and identifying its root cause. It is expected that the results of this test will contribute to the development of resilient processing architectures through more advanced scalable diagnosis strategies and robust distributed protocols.

The SPIDER architecture consists of processing elements (PEs) executing the applications and high-level system functions, and the Reliable Optical Bus (ROBUS) communication system, which provides guaranteed basic services that support PE-level services. The goals of the ROBUS design are to reduce the computational burden on the processing elements, to implement the basic distributed protocols in the way they are most effective (i.e., in hardware), and to provide a simple system abstraction to the PEs for what is an inherently complex distributed processing problem. ROBUS-2, an instance of ROBUS, is a time-division multiple access (TDMA) broadcast communication system with medium access control by means of a time-indexed communication schedule. ROBUS-2 provides guaranteed fault-tolerant services to the attached PEs in the presence of a bounded number of internal faults. These services include message broadcast (Byzantine Agreement), dynamic communication schedule update, time reference (clock synchronization), and distributed diagnosis (group membership). ROBUS-2 also features fault-tolerant startup and restart capabilities, it tolerates internal as well as PE faults, and it incorporates a dynamic self-reconfiguration capability driven by the internal diagnostic system. ROBUS-2 consists of custom-designed hardware-based ROBUS Protocol Processors (RPPs) implementing the ROBUS-2 functionality, and a lower-level physical communication network interconnecting the RPPs. Figure 1 shows the ROBUS topology. The bus has a redundant active-star architecture with the Bus Interface Units (BIUs) serving as the bus access ports, and the Redundancy Management Units (RMUs) providing connectivity as network hubs. The network between BIUs and RMUs forms a complete bipartite graph in which each node is directly connected to every node of the opposite kind. All the communication links are bi-directional.

In order to perform a thorough assessment of this complex communication system, the approach selected for this test is to perform a series of subtests with different configurations of the communication system and targeting (i.e., radiating) components based on the aspect of the system being examined. In particular, it is of interest to assess the effectiveness of the mechanisms for internal-fault masking, node

recovery, and system re-initialization. There is also interest in examining the manifestations of single-node faults, as their severity and frequency are critical pieces of data in the design of redundancy management strategies.

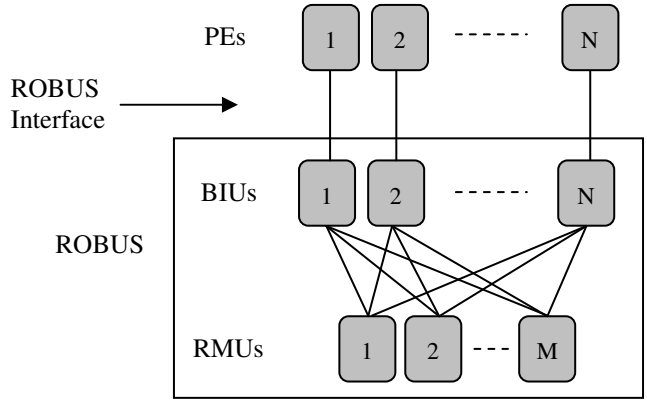


Figure 1: ROBUS Topology

The selected ROBUS-2 configurations are 4x2 (i.e., 4 BIUs and 2 RMUs), 4x3, and 4x4, which are one-, two-, and three-RMU fault tolerant, respectively. The number of BIUs is kept constant because the ROBUS-2 protocols are highly symmetric (i.e., BIUs and RMUs perform largely the same operations) and because, for the most part, the number of BIUs used in practice is determined by the application and application-level fault tolerance considerations. These ROBUS-2 configurations will be tested with one RMU, one BIU, or all BIUs and RMUs targeted in a HIRF radiation chamber. For all these configurations, there will be extensive monitoring of the internal state of all configuration nodes and the interfaces to the PEs. In addition, there will be the capability to monitor the behavior of specific ROBUS-2 nodes at their output interfaces. To enable a more meaningful assessment of the capabilities of ROBUS-2, the test will include a simplex hub configuration that is functionally equivalent to ROBUS-2 at the PE interface but is not internally fault tolerant. Like the ROBUS-2 configurations, the internal state of this single node and its behavior at the PE interfaces will be monitored.

HIRF environments are characterized by a number of parameters, including central frequency, field strength, modulation envelope and duration of the disturbance. In the interest of time, and considering the uncertainty about the effect that the radiation may have on the targeted components, including the possibility of physical damage, only a limited number of field parameter combinations will be tried in this test. A test at a given frequency and envelope modulation is divided into a field-strength susceptibility threshold phase, in which the field strength is incremented until errors in the targeted communication system are detected, and a field effects phase, where the field strength is incremented in small steps up to a maximum predetermined strength above the susceptibility threshold. Each radiation exposure, called a **round**, has a maximum time duration during which the radiation parameters are constant to allow the monitoring system to collect data about the response of the system under test (SUT) to the environment. A detailed description of the test plan is given in [1].

2. Test Control and Monitoring

The implementation of the test plan is supported by a computer system network that controls the generation of the HIRF environment, generates a workload for the SUT, monitors the activity of the SUT

in real time, and collects data for post-test analysis. Figure 2 is a diagram of this control and monitoring system. The Test Controller handles the generation and monitoring of the field, and it also controls the power supply to the SUT targeted components inside the test chamber. The PE Emulator interacts with and monitors the SUT at its interfaces. The Bus Monitor collects state data from the SUT components. The Test Control System (TCS) computers synchronize their activities by communicating via RS-232 links using the Controller Coordination Protocol (CCP) [1]. The CCP is a handshake protocol developed for this test that consists of a check phase, during which the TCS computers prepare for the next radiation round and confirm that the SUT is operating properly, and the test phase, where the radiated field is activated. The CCP allows the TCS to automatically execute parts of the test plan according to a predetermined test sequence. To maintain safe operation, the TCS requires manual acknowledgement by the operators before initiating execution of a round. The TCS was designed to ensure that the test operators are always aware of the state of the test, and it automatically returns the chamber and the radiated SUT components to a safe state after every round. The TCS has built-in exception handling capability to automatically terminate a round if unexpected behavior from the radiated field or the SUT is detected. As a last resort, the test operators can also manually terminate a round.

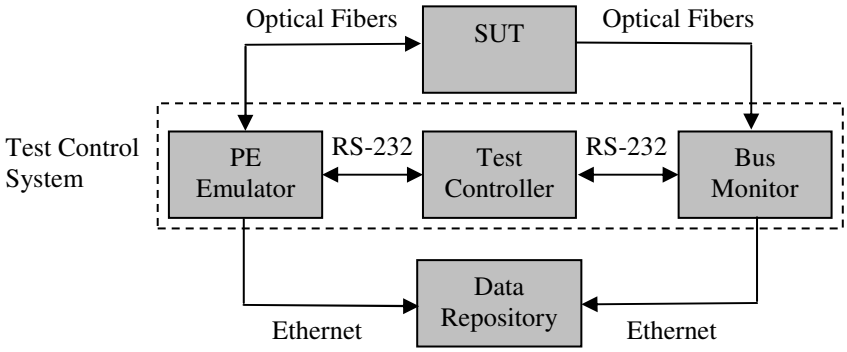


Figure 2: Test Control and Monitoring System

The interaction of the PE Emulator and the Bus Monitor with the SUT takes advantage of the highly deterministic behavior of ROBUS-2 (and the simplex hub). Although ROBUS-2 is a distributed system with complex behavior, its design is simplified by a synchronous-composition model that enables the system nodes to behave according to internal time-driven operation schedules that are synchronized such that the input-output activities of each node seamlessly blend with the input-output activities of the others. This composition model is supported by a highly precise distributed time protocol that maintains node synchrony. To simplify the monitoring of the SUT, the PEs are designed to send the same message sequence to ROBUS-2 every cycle, which under normal fault-free operation will result in ROBUS-2 also sending the same results to the PEs every cycle, with possible slight timing variations that are accounted for in the design of the PEs. Because of this repetitive activity at the PE interfaces, the communication between BIUs and RMUs, as well as the internal state of the nodes, is repeated every cycle. This results in an ideal situation in which the detection of errors can be accomplished by simple direct comparison between the observed results and static expected results.

The PE Emulator has hardware-implemented processes that directly interact with the SUT, and software processes that perform higher level monitoring activities and data collection. The hardware processes are divided into two independent parts: one for interacting with and monitoring the SUT at the PE interfaces, and another for monitoring one specific ROBUS-2 node at its BIU-RMU communication interface. The PE Emulator and the SUT communicate via custom-designed fiber-optic data

communication links. The PEs are designed to synchronize to the SUT and interact with it by sending and receiving messages according to a fixed schedule. There is one independent PE module for each of the four SUT external ports. Each PE performs error checks on all received messages and builds a record each communication cycle summarizing its observations during the cycle. The PEs also forward their detailed observations of each SUT message to a common monitoring module, called the Hub Fault Analyzer (HFA), which classifies the behavior of the SUT by applying a multi-observer-based fault model used in theoretical fault-tolerance analysis. The second part of the PE Emulator hardware process is a Node Fault Analyzer (NFA) module that is a slightly modified version of the HFA applied to the analysis of node-level fault manifestations. The NFA is used only with ROBUS 4x4 configurations with one radiation-targeted node. The inputs to the NFA are derived from the observations by the nodes opposite the targeted one (e.g., the 4 BIU observers for an RMU targeted node). Both HFA and NFA modules generate a summary record of their results at the end of every SUT cycle. The output data records of the PEs, HFA, and NFA modules are forwarded to separate System Health Monitors (SHMs) that independently assess the status of the SUT by comparing the content of each record against reference records corresponding to error-free operations. Each SHM's assessment is combined with its corresponding data record and a sequence number. The new compound record is time tagged using a common hardware-implemented local time reference, and then stored in buffers for the software process to read. The software process executes the CCP protocol, transfers the hardware-generated records to data collection files, assesses the overall status of the SUT based on the SHM results, and informs the operator of the latest assessment results by updating a monitoring display.

The Bus Monitor also consists of hardware and software processes. The Bus Monitor communicates with the SUT via the same kind of fiber-optic data links as the PE Emulator. The hardware processes collect message blocks independently sent by each of the SUT nodes containing snapshots (i.e., samples) of their internal state. Each message block is checked for proper format, tagged with a sequence number, and packaged into a data record. Each monitoring channel has an independent SHM that assesses the status of the corresponding SUT node by comparing the state data records against known error-free reference records. As in the PE Emulator, the SHM results are added to the data record, time tagged and buffered until the software process reads it. The software process of the Bus Monitor performs the same functions as the software process of the PE Emulator.

At the end of every round, the PE Emulator and the Bus Monitor send their data files to the central repository using a dedicated Ethernet network as illustrated in Figure 2. Appendix A describes the content of these files.

3. Physical Test Articles

The ROBUS-2 nodes and the simplex hub are implemented using the commercial off-the-shelf (COTS)-based nodes of the reconfigurable SPIDER prototyping platform (RSPP) developed for NASA by Derivation Systems, Inc (DSI) under a phase III SBIR contract. The RSPP is a Field Programmable Gate Array (FPGA) based development system for the design and testing of SPIDER prototypes. The architecture is a scalable modular system composed of individual RSPP nodes interconnected with point-to-point fiber optic links. Figure 3 shows two views of an RSPP node. An RSPP node is a PC/104-plus computer system with the following functional hardware components.

- **PF3100-2V3000 PC/104+ FPGA Module**: PF3100 with the Xilinx Virtex-II XC2V3000 (3 million gate) FPGA.

- **PFBR104 PC/104 Fiber Optic Transceiver Module**: Interfaces with the PF3100 over the PF3100 IO connector and provides four Agilent HFBR-5905 fiber optic transceivers. Each RSPP node has two PFBR104 modules providing a total of 8 fiber optic IO channels.

- **PC/104+ CPU Module**: The PC/104+ CPU module is a Lippert CRR2 with a 300MHz Pentium class processor, 64MB SDRAM, 256MB Compact Flash, 10/100 Ethernet, VGA, Keyboard, RS232 serial port, parallel port, USB port, and cables. The White Dwarf Linux operating system is installed on the CPU module.

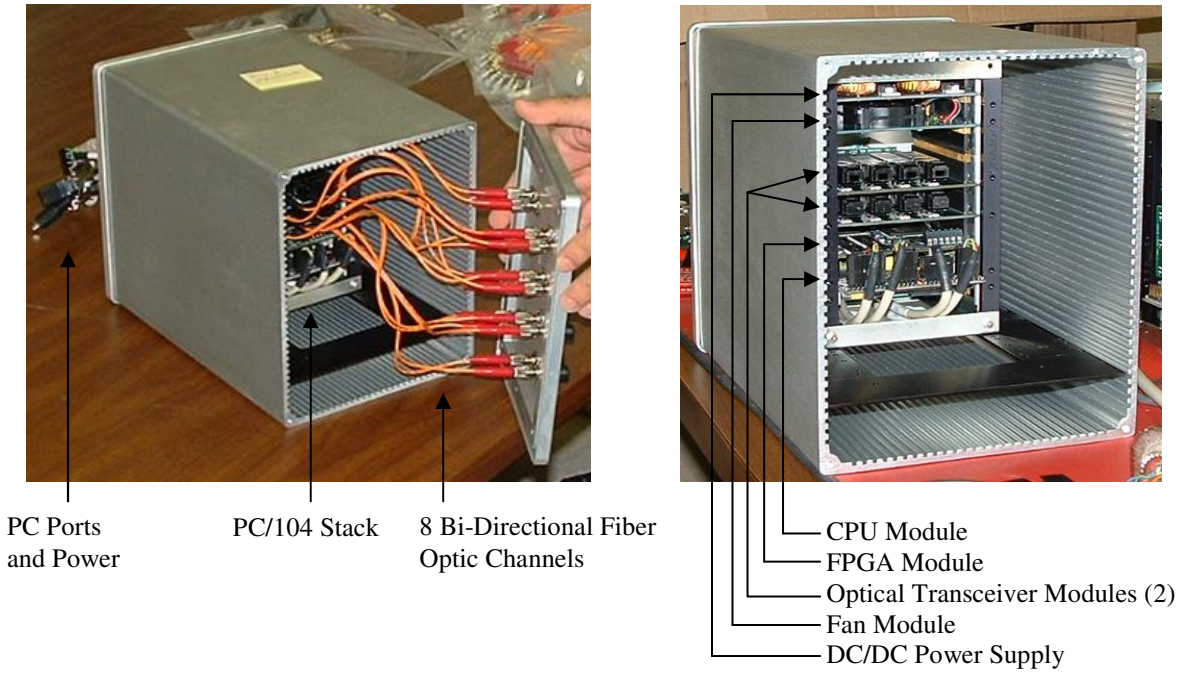


Figure 3: RSPP Node

Figure 4 shows the functional components of an RSPP node. The main function of the node when programmed as a BIU, RMU, or hub is implemented as a hardware process running on the static random-access memory (SRAM)-based field-programmable gate array (FPGA). The program for the FPGA is stored in non-volatile flash memory, together with functional parameters for the hardware process, including the node identification number in the case of ROBUS-2 nodes. The FPGA is connected via low-voltage differential signaling (LVDS) [7] lines to a group of eight fiber-optic transceivers, each composed of independent transmit (Tx) and receive (Rx) sections, used for point-to-point communication with other nodes. The complex programmable logic device (CPLD) controls the loading of the FPGA from flash when triggered by the CPU-controlled reset signal on the ISA bus. When setup as a BIU, RMU or a hub, the CPU is not programmed, and it is only used to assert the ISA reset signal upon power-up and in response to a reset command from the CPLD. The CPLD acts as a pass-through for reset requests generated by the hardware process running on the FPGA. This awkward implementation of a node-wide reset function is a side effect of the limitations of the ISA bus (where only the bus master can assert the bus reset signal), and also of having to modify the program of the CPLD to realize a special feature not originally supported by the reconfigurable module (that is, an FPGA-initiated reset). This node-wide reset feature supports a policy in the design of ROBUS-2 that requires a node to reset itself immediately upon the detection of an internal error [2]. Document [1] has additional information about the RSPP.

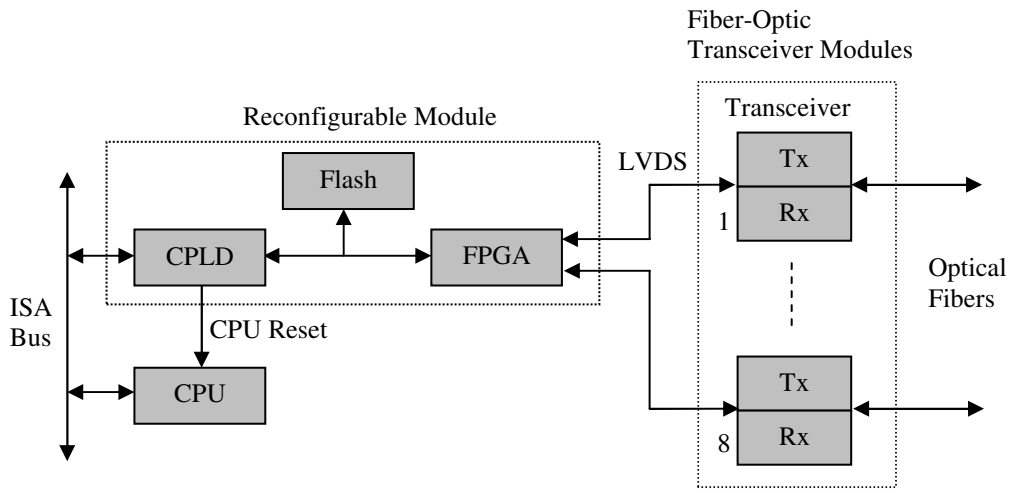


Figure 4: Diagram of the functional components of an RSPP Node

The RSPP nodes are flexible enough to support a wide range of functions with various degrees of complexity. One critical disadvantage of the current RSPP nodes when programmed as ROBUS-2 nodes is their long initialization delay caused by the need to reprogram the FPGA every time there is a reset. This results in large recovery delays, and its effect is especially severe on the re-initialization delay of the full ROBUS-2 system. Future versions of this platform will use a redesigned reconfigurable module that eliminates the CPLD and replaces the SRAM-based FPGA with a flash-based FPGA that does not lose its programming after a power cycle and does not need to be reprogrammed to implement a complete functional reset. An RSPP node with that newer version of the reconfigurable module will not need the CPU to implement a reset.

4. Data Links

All point-to-point communication between ROBUS-2 nodes and between the SUTs and the PE Emulator and the Bus Monitor use a custom-designed serial data link illustrated in Figure 5. The central components of this link are the Manchester-encoding serial transmitter and receiver. The Manchester transmitter reads in a whole word from the source process and packages it into a message, which is then serialized, encoded in Manchester format and sent out to the fiber-optic transmitter. At the receiving end, the fiber-optic receiver recovers the bit stream and sends it to the Manchester receiver, which decodes the bits, rebuilds the message while performing error checks, and then sends the payload to the destination process together with the error syndromes. The data synchronizers on the transmission and reception sides ensure reliable data transfer across clock domain boundaries. The clock frequencies of the source and destination processes (SClk and DClk, respectively) are independent, and the data link does not constrain their values. The design of this serial data link is based on concepts described in [8], as well as the Mil-Std-1553 avionics data bus [9].

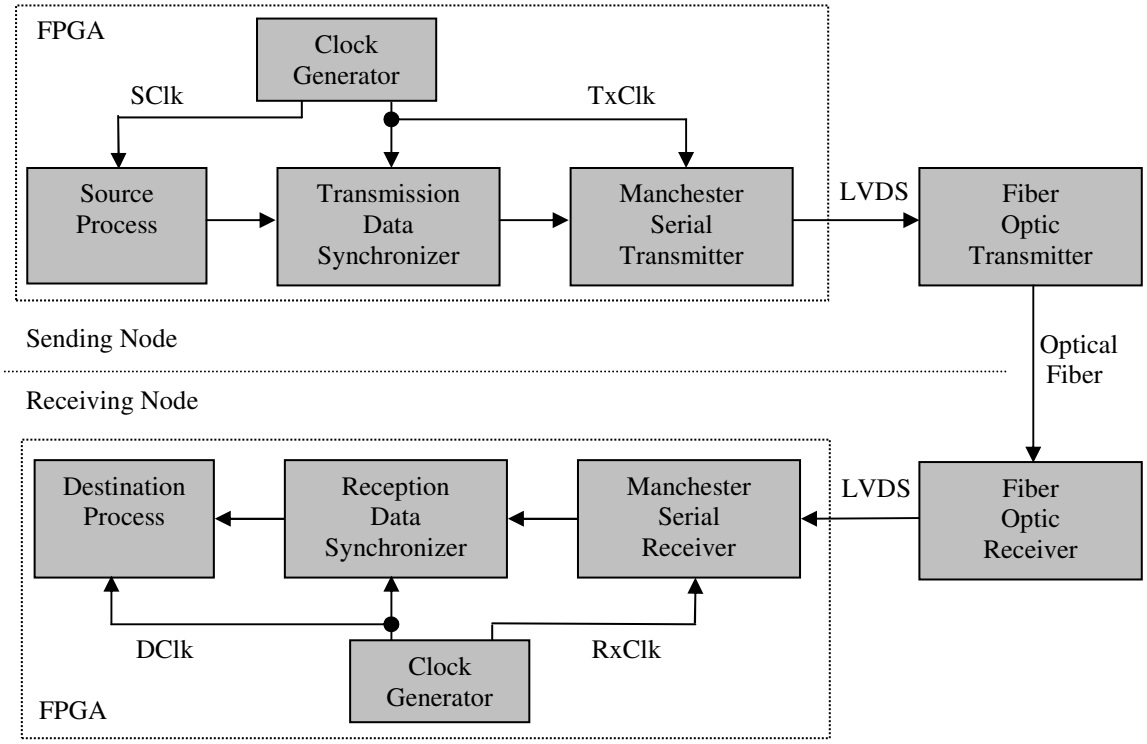


Figure 5: Fiber-optic, Manchester-encoded serial communication data link

Figure 6 shows how data bits in their basic non-return-to-zero (NRZ) format are encoded in Manchester format. Every Manchester bit consists of two half-bits, with the first half matching the level of the NRZ bit and the second half having the opposite level. There is a transition at the mid-point of every Manchester-encoded bit. Manchester bit encoding can be viewed as a type of 2-value phase encoding in which the phase of the signal with respect to a reference clock indicates the corresponding value of the data bit. From this perspective, a 0^0 phase corresponds to a bit value of 1, and a 180^0 phase corresponds to a value of 0.

Figure 7 illustrates the data-link message format. The first part of the message is a unique synchronization (sync) pattern composed of a bit-wide invalid 0 (in NRZ format) followed by a Manchester-encoded 0 and then an invalid 1 (in NRZ format). The purpose of the sync pattern is to clearly identify the beginning of a message and to provide a timing reference for decoding the rest of the message. The Manchester transmitter is designed to send a half-bit high before a sync pattern to reduce the uncertainty about the duration of the first half of the sync pattern. The sync pattern is followed by the payload section, which is the data word submitted by the source process to the Manchester transmitter. The most significant bit (msb) of the data is sent first. The last part of the message is the redundancy check, consisting of a single even-parity bit (i.e., set to 1 if the number of 1's in the payload is odd, and set to 0 if otherwise). After a message, the Manchester transmitter sends a constant-phase clock signal until the beginning of the next message.

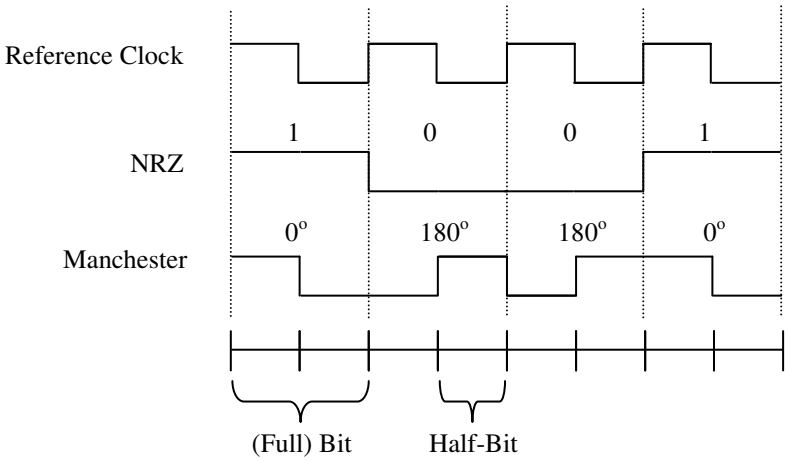


Figure 6: Manchester Bit Encoding

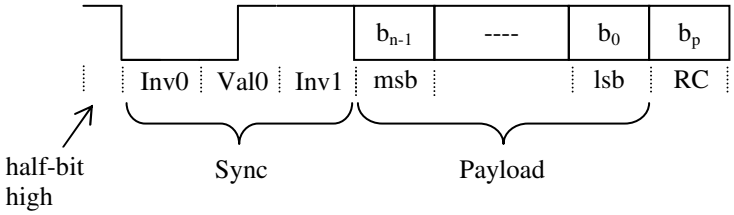


Figure 7: Data Link Message Format

The data link design used to implement the SUTs is the result of a series of design cycles where various parameters like clock rates, bit rates, message format, and encoding and decoding algorithms were tested and compared. The chosen design has a bit rate of 33 Mbps (megabits per second) with the serial transmitter running at (TxClk =) 66 MHz (i.e., twice the bit rate) and the receiver operating at (RxClk =) 198 MHz (i.e., 6 times the bit rate). Although a range of payload widths can be implemented, for the systems developed for the HIRF test, all the data links use a payload width of 17 bits. The error checks performed at the receiving end include a Manchester encoding check (i.e., Do the first and second half-bits have opposite binary values?) applied to the payload and redundancy check bits, a message parity check, and a waveform shape check, which is applied to the received signal itself rather than to individual messages. The design of the reception data synchronizer allows for a maximum message rate of one message every two ticks of the destination process clock (DClk). The transmission data synchronizer is designed to ensure that this data rate is never exceeded.

The internal designs of the Manchester transmitter and receiver, as well as the waveform monitoring function, are described next.

4.1. Manchester Serial Transmitter

Figure 8 shows the organization of the serial transmitter, which has a data-path-and-controller design. The data path has two major sections: the NRZ stream generator produces the message bits in non-return-to-zero format, and the Manchester encoder converts the NRZ bits to Manchester format using the two-value clock phase encoding approach. The multiplexer in the Manchester encoder is used to output the invalid bits of the sync pattern by passing NRZ bits directly to the output. The controller coordinates the timing of all data path activities. The transmitter processes one message at a time triggered by the

Strobe_In input signal. The Ready signal is asserted only after full processing of a message is complete.

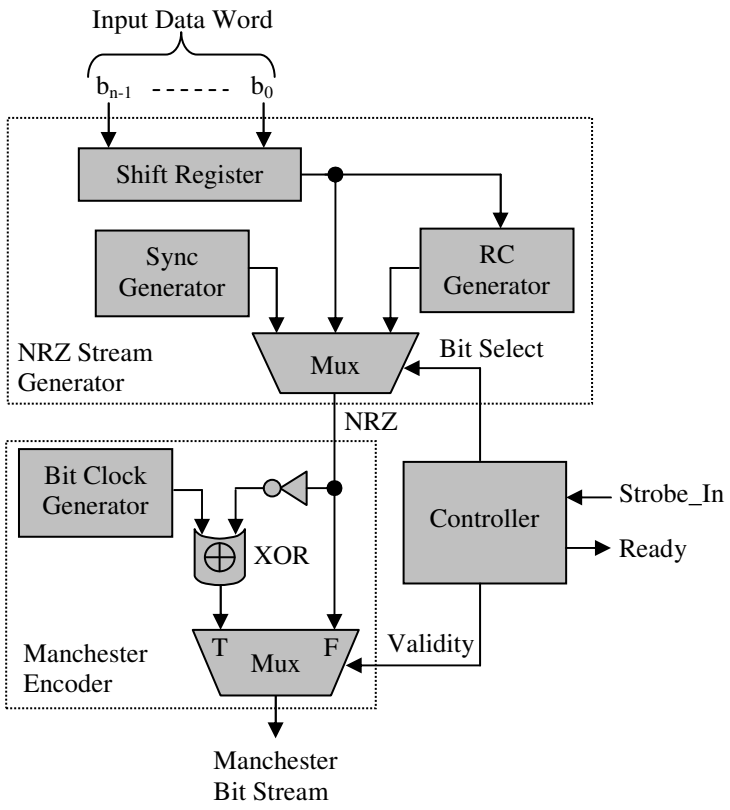


Figure 8: Organization of the Manchester Serial Transmitter

This serial transmitter requires a minimum separation between messages of one full bit. For a 17-bit payload message, the maximum efficiency supported by this design is 77.3% (i.e., 25.5 Mbps effective data rate), or 17 payload bits every 22 line bits (3 sync bits + 17 payload bits + 1 parity bit + 1 message spacing bit).

4.2. Manchester Waveform Monitor

The waveform monitor is a supporting function of the serial receiver meant to increase the error detection coverage. Like the serial receiver, the waveform monitor operates at a frequency of 198 MHz, which under ideal conditions results in 6 samples per full Manchester data bit, or 3 samples per half-bit. The monitor samples the input bit stream and compares the samples against the characteristics of a properly formatted Manchester encoded signal. The operation of the monitor is based on the concept of a symbol, which is a sequence of samples with a constant level (or value). As shown in Table 1, four valid symbols are defined. The allowed range of a symbol's sequence duration is specified taking into consideration the effects of a phase difference between the input stream and the sampling clock, and also possible slight variations in the frequencies of these signals. The symbols are used in conjunction with a set of rules, listed in Figure 9, that specify the allowed symbol transitions for a properly encoded Manchester bit stream. A violation of these transition rules results in an immediate declaration of the input signal as invalid. The monitor redeclares the input signal as valid after four consecutive valid

symbols are received. This four-deep rule was empirically derived and is the result of a trade-off between the error coverage (waiting for more consecutive valid symbols before declaring the signal valid may increase the coverage), the recovery delay (waiting too long to redeclare the input signal valid can result in the loss of good messages), and the complexity of the implementation (a higher-complexity implementation has smaller timing margins, which increases the likelihood of random processing errors due to internal timing violations). Notice that there is no transition rule constraining the time spacing between observed sync patterns. Such a rule could increase the error detection coverage, but it would also increase the complexity of the implementation.

Table 1: Symbol definitions for Manchester Waveform Monitor

Symbol Label	Level	Duration (samples)	Comments
S0	Low (0)	8 to 10	Sync pattern low section (A sync pattern is immediately preceeded by a half-bit high.)
S1	High (1)	8 to 13	Sync pattern high section and possibly first half-bit high
D0	Low (0)	2 to 7	Manchester-encoded data; up to two consecutive half-bits are low
D1	High (1)	2 to 7	Manchester-encoded data; up to two consecutive half-bits are high

1. S0 is followed by S1.
2. S1 is followed by D0.
3. D0 is followed by D1.
4. D1 is followed by D0 or S0.

Figure 9: Valid symbol transition rules

4.3. Manchester Serial Receiver

Figure 10 shows the organization of the serial receiver. The Manchester bit stream (MBS) and Signal Error inputs pass through a sampler meant to ensure that metastability [10] is not a concern further down in the processing path. The Signal Error input is driven by an OR gate connected to the output of the Manchester Waveform Monitor supporting the receiver and the logically inverted signal-detect output of the fiber-optic receiver. When the Sync Detector receives a valid sync pattern, the Sampling Clock Generator aligns the phase of its sampling clocks to the timing of the sync pattern, and the main controller begins the process of recovering the message. The Signal Check, Encoding Check, and Redundancy Check modules inspect the bit stream as the shift register collects the payload bits. The Strobe_Out signal is asserted for one clock tick when a new message is ready. The high clock rate driving this receiver enables it to process message streams with a minimum message spacing of less than one half-bit. That gives the receiver a higher maximum throughput than the transmitter, which generates message streams with a spacing of at least one full bit between messages.

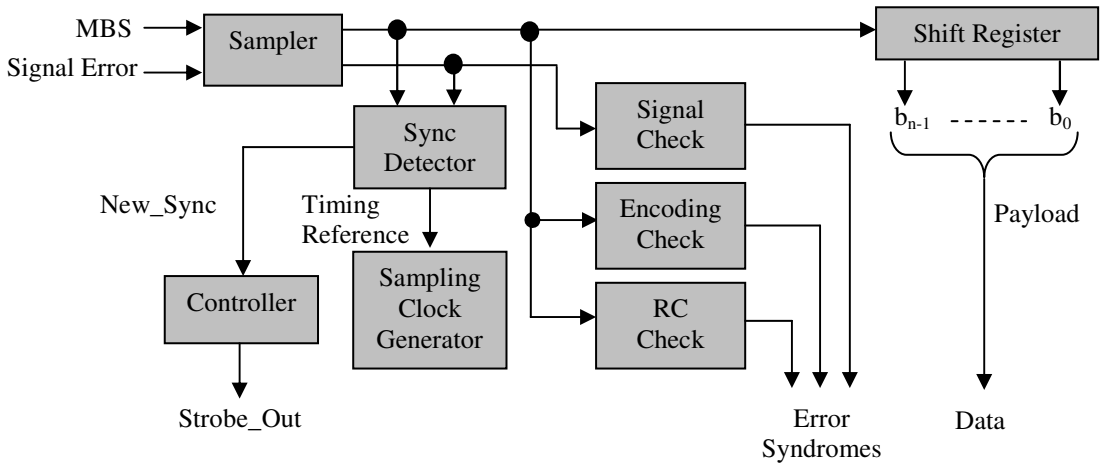


Figure 10: Organization of the Manchester Serial Receiver

4.4. Link Errors

The main sources of data link errors are related to sync pattern detection and signal sampling.

A problem observed early on in the development of the data link was the sporadic reception of false messages when there was no light input to the fiber-optic receiver. One obvious conclusion from these occurrences was that the signal-detect output from the fiber-optic receiver, which is supposed to be deasserted when there is no light input, was not being read reliably by the serial receiver. The cause of that problem was found to be a signal-standard incompatibility between the fiber-optic receiver output and the FPGA input. After consultation with DSI, it was determined that there was no suitable way to circumvent this problem and that the designs of the waveform monitor, the serial receiver, and the destination process where the messages are delivered would have to accommodate this undesirable behavior. The waveform monitor greatly reduced the rate of false messages for no-light conditions, but it did not eliminate the problem completely. This is probably due to the fiber-optic receiver's data output being allowed to float, which essentially injects a random input sequence on the signal that occasionally is interpreted by the waveform monitor as valid symbol sequences. As stated previously, the design of the waveform monitor was the result of a trade-off between error detection coverage and complexity, and it was not possible to reduce the missed-detection rate to an insignificant level without compromising the internal timing margins of the monitor. However, extensive testing has revealed that, although false-message reception is still possible, it appears that the combination of the waveform monitor and the error detection within the serial receiver is highly effective in detecting errors in these false messages. With the current data link design, none of these false messages has been observed without an error syndrome being asserted.

During normal operation with all components working as designed, the reliability of the link is limited by errors in decoding the Manchester signal at the serial receiver. Figure 11 illustrates how the sync pattern is used to obtain a timing reference S_{Ref} for sampling the half-bits of the message, and also how the accuracy of the S_{Ref} sample with respect to the midpoint transition of the sync pattern affects the timing of the half-bit samples $S_{Half\text{-}Bit}$. Ideally, the half-bits would be sampled exactly at their midpoints, which would result in the smallest possible decoding error rate.

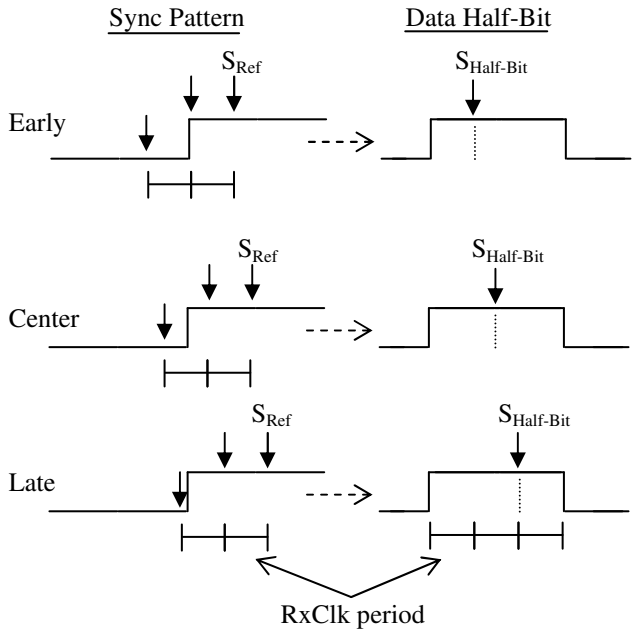


Figure 11: Sampling of half-bits using the sampling of the sync pattern as a timing reference

Two possible causes of decoding errors need to be considered: drift and jitter. The serial transmitter at the sending node and the serial receiver at the destination node are driven by clock signals generated by independent clock multipliers that use inputs from physical oscillators as frequency and phase references. In general, the clock signals generated by physical oscillators are characterized by their drift rate with respect to real time, which is a bound on the error of the oscillator in measuring the passage of time. The oscillators of the RSPP nodes have a drift rate bound of ±100 parts per million (ppm). Given the size of the messages (18 data bits), the bound on the drift rate of the clocks, and the sample timing margins (nominally, one RxClk tick on either side of the sample), the drift rate of the transmitter and receiver clocks by itself is not a significant contributor to the decoding error rate.

Jitter, defined as the deviation of the significant instances of a signal from their ideal location in time [11], can be the dominant source of decoding errors. Jitter has two components: deterministic (or bounded) jitter and random (or unbounded) jitter [12]. Deterministic jitter is bounded in amplitude and is characterized by a peak-to-peak value. Deterministic jitter is due to systemic causes like electromagnetic interference (EMI), crosstalk, and grounding problems. Random jitter is described by a Gaussian probability distribution and is characterized by its standard deviation (or RMS = Root Mean Square) value. The causes of random jitter include thermal noise and shot noise, among others [13]. Assuming a scenario in which the timing margin of a half-bit sample is always at its minimum (i.e., the early or late case in Figure 11) with the clocks of the serial transmitter and receiver having maximum but opposite drift rates (i.e., one runs at the fastest rate and the other at the slowest rate) and taking into consideration the jitter contributions of the oscillator clocks, clock multipliers, and fiber-optic transceivers, it was found that the error rate of the data link is 10^{-9}, or one error per billion half-bits. Given that the timing of the half-bit samples is likely to have a uniform distribution around the center of the half-bits, which results in bigger timing margins for the average case, the actual error rate of the link should be several orders of magnitude better than the analyzed case. To date, the data links have been tested with hundreds of billions of transmitted bits without a single error ever being detected while operating in a benign (i.e., no HIRF) environment.

Having reliable data links with extremely low intrinsic error rates is critical for the systems described in this report, as knowledge of such reliability is the basis for assuming that the root cause of errors detected during testing is the HIRF environment.

5. ROBUS-2 SUT

The design of ROBUS-2 at the conceptual and implementation levels is described in [2, 3]. For the HIRF tests, the behavior and implementation of ROBUS-2 were slightly modified to suit the test requirements. To reduce the SUT development burden, it was desired that a single implementation of ROBUS-2 be able to support all the ROBUS-2 test configurations of interest, including 4x2, 4x3, and 4x4, with one RMU, one BIU, or all BIUs and RMUs as radiation targets. Figure 12 illustrates the general system configuration for the ROBUS-2 tests. All the communication links in Figure 12 use the custom data links with the payload formatted as a ROBUS-2 message (RM) [2] consisting of a tag bit and a 16-bit RM payload. The parameters of the ROBUS protocol processors are set for a 4x4 system, but the automatic real-time reconfiguration capability allows it to support the 4x2 and 4x3 configurations without change. As shown, in addition to PE-BIU and BIU-RMU data links, an RPP must support connections to a state monitor and to the Node Fault Analyzer (NFA). For this purpose, the RPP was augmented with a test port that extracts state data from the internal units (Figure 13), including: from the Mode Control Unit, the control command and reset signals; from the Input Unit (IU), the schedule assessment results and the input message timing syndromes; from the Input Diagnostics Unit (IDU), the content and timing pulses of input messages, and the message content syndromes; from the Route and Vote Unit (RVU), the disagreement-with-the-result-of-the-vote syndromes from the message content voter and the synchronization voters; from the Node Diagnostics Unit (NDU), the diagnostic accusations and convictions; and from the Status Monitoring Unit (SMU), various node and system status assessment signals, including diagnostic error codes reporting the cause of unexpected RPP resets. The SMU error codes are described in Appendix B.

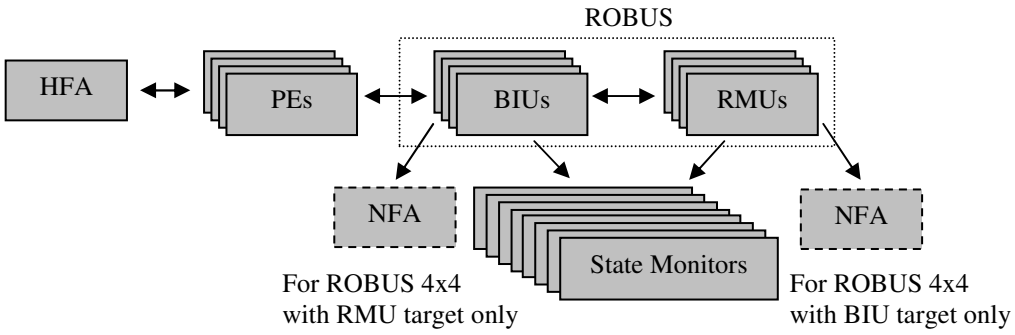


Figure 12: System configuration for ROBUS-2 SUT tests

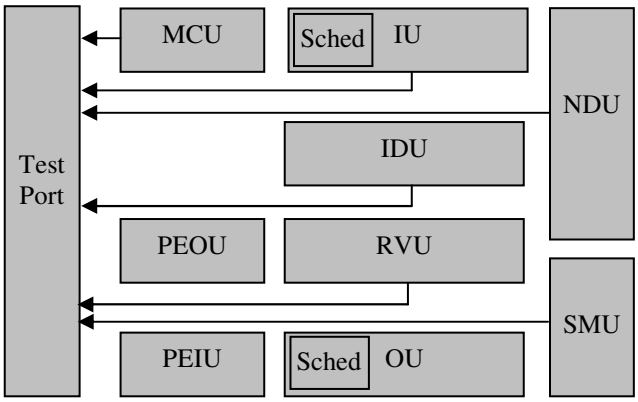


Figure 13: ROBUS protocol processor with state-sampling test port

The behavior of ROBUS-2 and the RPP was modified to support the Node Fault Analyzer function, including the requirement that the same NFA implementation be able to receive observations from either BIUs or RMUs without needing an explicit indication of which kind is the direct observer of the target node. To simplify the design of the NFA, the observing RPPs need to send to the NFA for each expected message separate indications of whether the received message from the target node is correct in the content and timing domains. The RPPs can do this assessment by comparing the content and timing of the target's message against vote results of all the inputs. Given that the NFA will only be used with the ROBUS 4x4 configuration with only one targeted node, there will always be three nodes of the same kind as the target that are not radiated, and if their outputs can be made to agree, they will constitute a correct majority at the voters. To make this possible, the BIUs (and the PEs) were modified such that during the PE Broadcast communication mode of operation, all the BIUs (and their PEs) transmit all the time independently of the source indicated by the active communication schedule. The PEs were designed to always agree on the value they send, which ensures that RMU observers always have good references to determine the correctness of the messages received from a radiation targeted BIU. Changes to other phases of the ROBUS-2 operation were not required because the original protocols ensure that under error-free conditions all BIUs and all RMUs always transmit at the same time and agree on their messages, and, in particular, that all the good nodes of the same kind as the target will always send agreeing correct messages. Accordingly, the RPPs were modified to apply inline error checks that are input lane independent.

As described in [2], Appendix B, subsection B.5.3, ROBUS-2 can operate in two different reception modes for point-to-point communication: overlapping and non-overlapping reception intervals. A reception interval is a timing window used as a reference for message arrival error checks. The size of a reception interval window is determined by factors like the uncertainty in the message transmission delay and the precision of the time synchronization between sender and receiver. The synchronization precision is dependent on the precision achieved by the synchronization protocol, the drift rate of the physical oscillators, and the duration of the resynchronization period. For this implementation, ROBUS-2 is set to operate with a cycle rate of 300 Hz (or 3.3 ms per cycle), corresponding to approximately 10,000 local-clock ticks per cycle with the RPPs driven by 3 MHz clock signals. Using the data links with 60-meter (or 197-foot) optical fibers, the size of a synchronous reception interval was found to be 15 RPP clock ticks. Two versions of ROBUS-2 are available for this test: a high-speed, overlapping-reception-interval version with a data introduction interval (DII) of 4 clock ticks, and a low-speed, non-overlapping-reception-interval version with a DII of 16 ticks. The purpose of testing these two versions is to

determine the relative impact of the message rate and the reception interval duration on the susceptibility to the HIRF environment. For the 16-tick DII version, there is one tick between reception intervals during the PE Communication and Schedule Update operation modes. A 4-tick DII version was selected over a 2-tick version to allow enough time between ROBUS messages to send the necessary observation data to the NFA. Recall that all point-to-point communication uses the same data link design with a maximum message rate of 33 Mbps. With the RPP driven by a 3 MHz clock and the serial transmitter running at 66 MHz, the minimum DII allowed by the data link is 2, at which the link operates with 100% utilization (i.e., 25.5 Mbps effective data rate). With the DII set to 4, the link operates with 50% utilization (i.e., 12.8 Mbps effective data rate). A DII of 16 reduces the utilization to 12.5% (i.e., 3.2 Mbps effective data rate).

To perform its analysis, the NFA needs visibility into the inline error syndromes generated by the observing RPPs for each expected input message from the target node. These syndromes are divided into three groups: communication, timing, and content. The communication syndromes of a channel are generated by the data link. The timing checks are generated by the Input Unit and include the following: empty buffer (asserted if the input buffer is empty when it is time to process an expected message), input overrun (asserted if the input buffer is not empty after the last expected message is read from the buffer), and buffer overload (asserted if at any time the number of currently buffered messages exceeds a maximum value computed based on known bounds for the input and output message rates of the buffer). The content syndrome is generated by the Input Diagnostics Unit and consists of a comparison of the received message content against a protocol-dependent reference.

The ROBUS-2 RPP was designed with a built-in PE interface based on a simple first-in first-out (FIFO) buffer-interfacing model [3]. For the system depicted in Figure 12, in which the BIUs and the PEs are in separate nodes and communicate only through fiber-optic data links, the RPP must be coupled to an interfacing module that can better handle the timing aspects of communication with a remote PE. Figure 14 illustrates how the PE Interface Unit (PEXU) is interconnected between the RPP and the Manchester transmitter (MTx) and receiver (MRx). The PEXU consists of an Output Buffer section that ensures that the MTx timing constraints are met while preserving the determinism of the BIU-to-PE messages, and an Input Buffer section that stores the messages from the PE until the RPP is ready to read them. The function of the Output Buffer section is simplified by the time-triggered operation of the RPP, which guarantees a deterministic message stream from the RPP during normal error-free operation. The Input Buffer section is required to store a flag with each received message payload indicating if the MRx detected any errors with the arrival of the message or if the received message had an unexpected tag. The PEs send only DATA tagged messages [2]. When the RPP's PEIU requests the next PE message, the PEXU Input Buffer asserts an output error flag if either the stored error flag or the empty flag from the buffer is asserted. The Output Buffer section monitors the message stream from the RPP to trigger a reset of the Input Buffer when the RPP outputs a synchronization message (INIT or ECHO), which indicates that all the expected messages from the PE for the current cycle have been read and that the buffer can be cleared in preparation for the next cycle [2].

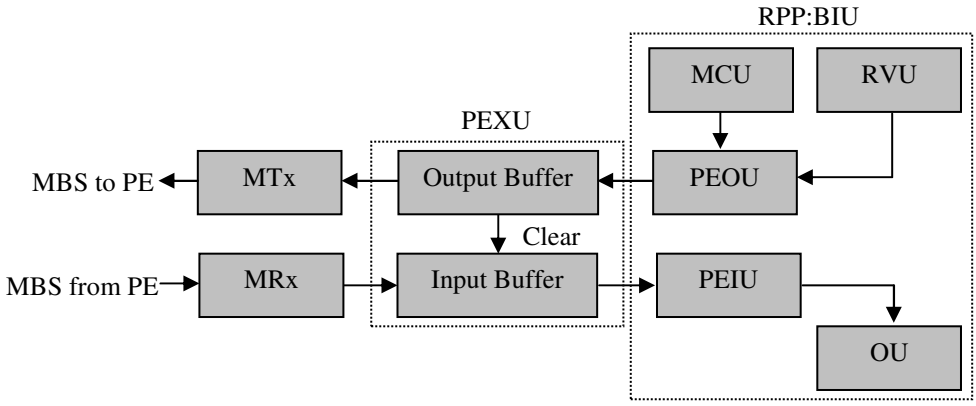


Figure 14: PE Interface Unit (PEXU) for an RPP configured as a BIU

6. Simplex Hub SUT

The purpose of the simplex hub is to provide a reference system for assessing the susceptibility and effects of the HIRF radiation on the ROBUS-2 configurations. The main requirement for the hub is to be functionally equivalent to ROBUS-2 at the PE interfaces. This enables the reuse of the PE design, albeit with slightly different interface timing due to the simplex, non-distributed nature of the hub. Document [2] provides a full description of the behavior of ROBUS-2. There is no need in the hub design for the Clique Detection, Clique Join, and Clique Initialization modes, as those modes are meaningless in the simplex system. The hub is designed to be equivalent to ROBUS-2 operating in Clique Preservation mode. Document [2], subsection 3.5, describes the message exchange pattern between ROBUS-2 and the PEs in Clique Preservation mode. Immediately after an internal reset (from power-up or a detected failure), the hub broadcasts an INIT message to synchronize the PEs and establish a timing reference for time-triggered operation. The diagnosis update messages about the status of BIUs and RMUs are always set not to indicate convictions since those nodes are not present in the hub system. The mode message always indicates that the hub is in Clique Preservation mode. For the Id message, the hub sends a different message to each PE corresponding to the port a PE is connected to (e.g., the PE on port 1 receives an Id of 1). For Schedule Update, the PEs download their desired schedule the same way as for ROBUS-2, then the hub processes the messages following the algorithm for process P1 of the ROBUS-2 Schedule Update protocol and broadcasts the accepted schedule to the PEs. After analyzing the new schedule, the hub sends the assessment result to the PEs. For the PE broadcast messages, the hub implements process P1 of the PE Broadcast protocol. At the end of this mode, the hub sends another INIT message and the cycle is repeated. The duration of a cycle is hard coded into the hub to match the average cycle duration of the ROBUS-2 implementation.

Figure 15 shows the internal organization of the simplex hub. The implementation uses modules and design ideas from the RPP where possible. The input buffers, including their error detection and processing logic, are taken from the PE Interface Unit (PEXU) designed for the RPP (see Figure 14). The input buffer module replaces the output of the buffer with a PE_ERROR message if the corresponding stored error flag is asserted or if the buffer is empty when the next PE message is read. The voter, used in the execution of the Schedule Update protocol, was taken directly from the word voter in the RVU of the RPP. Input lanes with detected errors are ineligible voters. The Content Transformation module replaces the output of the voter with a PE_ERROR message if there are no eligible voters or the result of the vote is NO_MAJORITY. The Schedule Processor, which is a slightly modified version of the design used in

the RPP, loads the new schedule and analyzes it for validity using the same rules as ROBUS-2. A default schedule is activated if the new schedule is invalid. The Schedule Processor drives the multiplexer according to the current schedule for PE Broadcast. The Output Unit has two sections: the individual Id Generators (IdG), which generate the Id message based on their corresponding port number, and the Common Output Generator, which generates all the other messages that are identically broadcast to the PEs. The Watchdog monitors the outputs and signals a failure to the controller if the time since the last INIT exceeds the maximum cycle duration. The Watchdog timer is reset when all the outputs simultaneously send an INIT message. This simple hub failure detection mechanism is the only internal-error detector. Like ROBUS-2, the hub is designed to be an independent system that provides services to the PEs and is PE-fault tolerant. Faulty PEs can cause the loading of an undesired PE communication schedule, but they cannot interfere with the proper execution of any of the protocols, including the Schedule Update protocol. The hub's Test Port samples the internal state of the hub for remote monitoring.

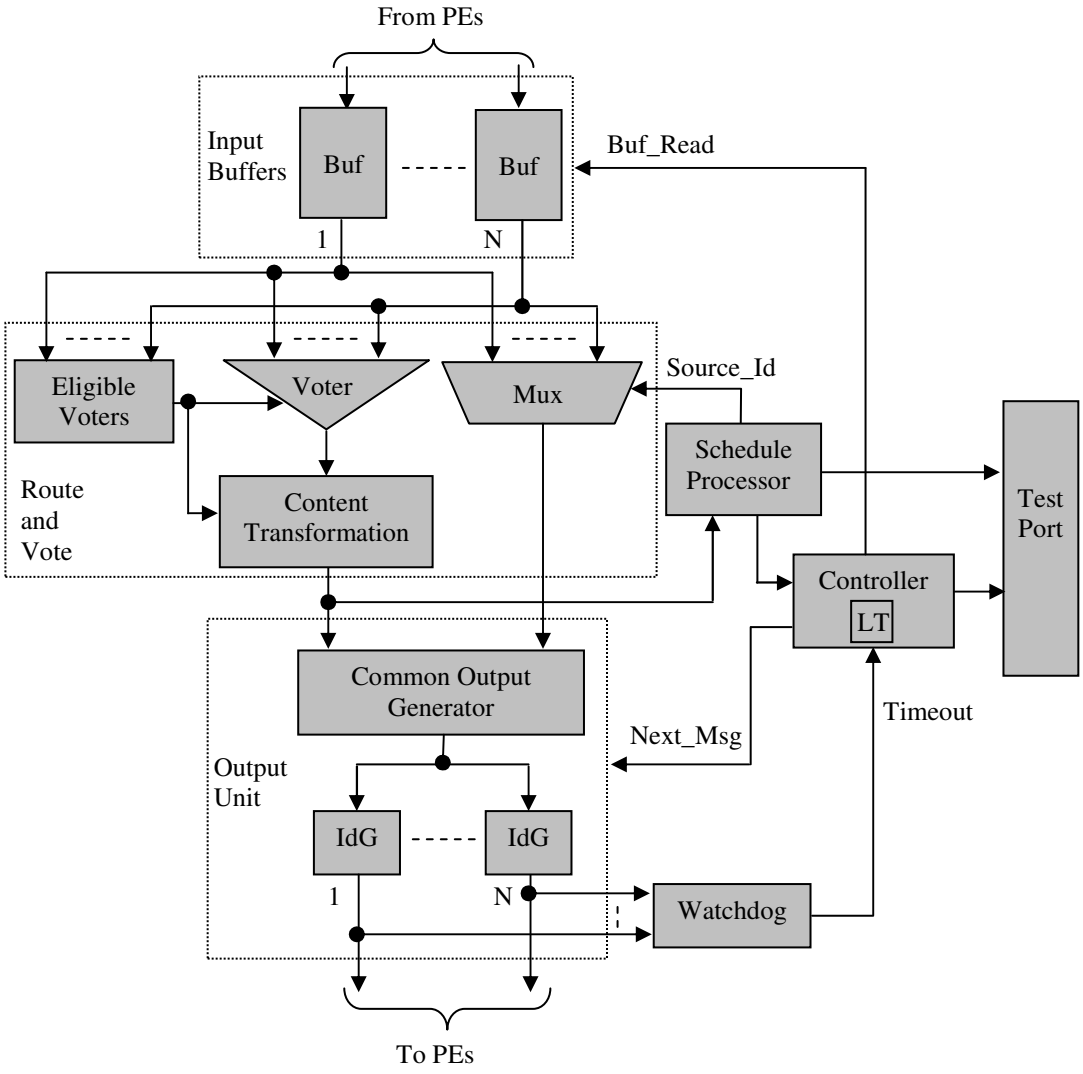


Figure 15: Internal organization of the Simplex Hub

7. Processing Element

The Processing Element (PE) designed for the HIRF test generates data for the SUT to operate on and observes and assesses the output of the SUT. The PE is required to provide a communication schedule to the SUT and then send messages to the SUT according to that schedule, but it is not required to perform any computation other than what is needed for PE-BIU interaction. To support real-time monitoring and post-test analysis of SUT behavior, the PE has to collect data on its observations during each communication cycle and provide a real-time observation data stream to the Hub Fault Analyzer (HFA) (see Figure 12) for integrated SUT analysis.

After a reset, the PE enters the initialization mode where it first waits to receive a synchronization message. Once synchronized, the operation of the PE is time-triggered and cyclic, with an expected resynchronization once per cycle. Normal operation begins when a Clique Preservation mode message is received. The first expected message of a cycle is the mode message. If there is a mode message error, the PE aborts the cycle immediately and executes a full reset. If no errors are detected, the remainder of the cycle will be completed independently of any detected errors for the other expected messages. If there is a synchronization message error, the PE will continue the cycle assuming perfect synchronization (i.e., no local time correction is applied) until all the cycle messages are processed, at which time a synchronization failure is declared, followed by a reset.

In essence, the PE is an active monitor that sends stimulus to the SUT and compares the response against a reference. To simplify the design of the PE (as well as the other SUT monitoring modules), the PE sends messages only when the SUT is in Clique Preservation mode. The PE uses a static communication schedule by which each PE is allowed to broadcast the same number of messages, and the total number of scheduled messages is the maximum allowed by the SUT for its particular data introduction interval (DII) setting. The PE message payload is generated by a pseudo-random number generator (specifically, a linear feedback shift register (LFSR) with parallel outputs) [14, 15] that is restarted at the beginning of each cycle, thus ensuring the same sequence of messages every cycle. Given the deterministic behavior of ROBUS-2 and the simplex hub, this repetitive input sequence generated by the PE ensures that under error-free conditions the SUT outputs the same sequence every cycle. During normal operation, the PE expects to receive the message pattern described in [2], subsection 3.5, with time-referenced message-arrival intervals. The PE is designed to operate with either overlapping or non-overlapping reception intervals to support both of these ROBUS-2 communication modes. Error detection is based on the assumption that the PE does not experience internal faults since it will always be isolated from the HIRF radiation. Because of this, all observed errors are attributed to the SUT. However, the PE does not adjust its reception expectations based on detected input errors. If the received Schedule Update results do not match the submitted schedule or the assessment is different than expected, the PE still expects to receive the same message sequence during the PE Broadcast communication phase as when the SUT is operating properly. As part of its error detection function performed for the HFA observation data stream, the PE changes its expected message content only during PE Broadcast communication to reflect knowledge provided by the HFA about which PEs are in normal mode. If a PE is not in normal mode, then it is not sending messages to the SUT and the expected response from the SUT is to deliver PE_ERROR messages during the schedule slots corresponding to that inactive PE. Note that the PE schedule is the same as the default schedule of the SUT, which is executed when the result of the schedule update is invalid. This PE behavior greatly simplifies the implementation and is consistent with the assumption that only the SUT can be the culprit of observed errors.

Figure 16 presents a simplified view of the PE's organization. The data processing modules of the Reception Unit (RU) are derived from the ones in the implementation of the RPP's IU. The Sync Pulse

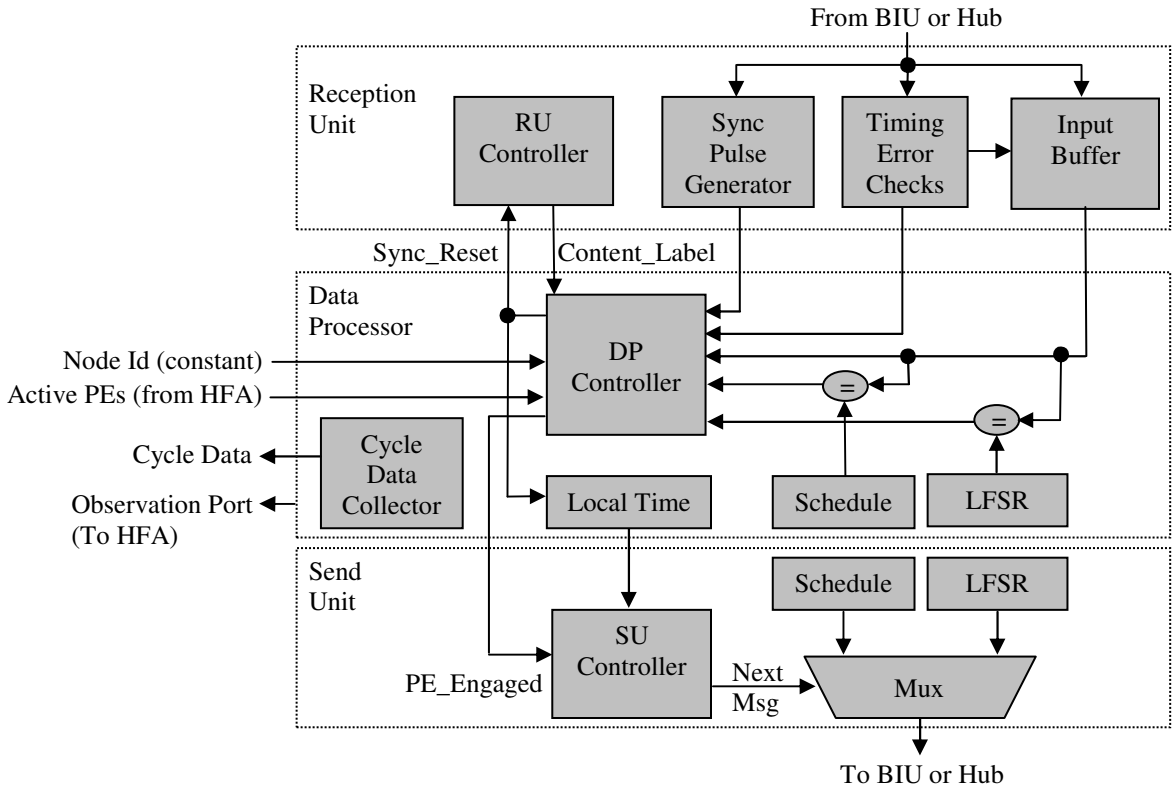


Figure 16: Simplified view of the Processing Element's organization

Generator outputs a timing signal with a predetermined delay after the arrival of a synchronization message (INIT or ECHO). The Timing Error Checks module performs the same timing checks as the RPP, including empty buffer, buffer-overrun (total number of received messages is higher than expected) and buffer overload (at a particular instant, the buffer is holding more messages than is expected based on the bounds for the input and output rates). Another module, not shown in Figure 16, counts the number of messages received during a cycle outside of the expected reception arrival intervals (i.e., the number of unexpected messages). The Input Buffer stores the received messages, the communication error syndromes generated by the data link receiver, and the buffer-overload timing syndrome until it is time to process them. The RU Controller has two modes of operation: unsynchronized (used during initialization to capture the first synchronization message) and synchronized (when its operation is time-driven). The RU Controller is programmed with the expected message sequence (including the communication schedule) and the time of the reception arrival intervals. At the end of each arrival interval, the RU Controller extracts the data from the Input Buffer and informs the Data Processor (DP) about the nature of the expected message. The Data Processor is responsible for determining the status of the PE, synchronizing the PE to the SUT, performing content error checks on the expected messages, collecting data about the observations during each cycle, and generating the observation data stream for the HFA. The central component of the Data Processor is the DP Controller, whose processing, including the coordination of activities by other DP components, is mostly determined by the Content_Label from the RU Controller. The Data Processor receives from the HFA a vector indicating which PEs are active, and it supplies observations to the HFA. The DP Controller generates the Sync_Reset signal that resets the local time and is used as a timing reference by the RU Controller. The DP Controller also generates the PE_Engaged signal to indicate when the PE is in normal operation mode (i.e., synchronized and with the SUT in Clique Preservation mode). The Send Unit (SU) Controller monitors the Local Time to trigger

the transmission of the schedule and the data messages. Appendix A provides a full description of the cycle data collected by the PE. The section on the HFA describes the observation data stream generated by the PE.

8. Hub Fault Analyzer

The Hub Fault Analyzer (HFA) classifies the observations of the SUT by the PEs according to the omissive-transmissive hybrid (OTH) fault model described in Table 2. (See [16] and [17] for an overview of hybrid fault models.) The observations are with respect to the expected messages for the pattern described in [2], subsection 3.5. The "data" of an observation is the content and/or the timing of an expected message. Each observation is classified independently. An omissive fault manifestation at an observer is one that the observer detects with its fault detection mechanisms. A transmissive fault manifestation at an observer is not detected. As an alternative to the terms omissive and transmissive, we use the concept of validity to capture the aspect of error detection at an observer. The data at a particular observer is valid (i.e., transmissive) if it passes all the input error checks, and it is invalid (i.e., omissive) if it fails any of the checks. The fault model can be expressed in terms of the four conditions listed in Table 3, which take into account the concepts of validity (i.e., error detection applied independently by each observer), symmetry (i.e., consistency or agreement of observations), and correctness (i.e., no errors) of the data at the observers. The PE error checks are specified to ensure that correct data are never declared invalid. However, valid data may not be correct. The coverage of the error checks applied at the PEs measures the percentage of incorrect received data that a PE can detect as invalid. Figure 17 presents the decision logic used by the HFA to classify an observation by the PEs.

Table 2: Description of the omissive-transmissive hybrid fault model

Fault Mode	Description
Correct	All observers receive the same correct data.
Omissive Symmetric	All observers declare invalid their received data.
Transmissive Symmetric	All observers accept the same incorrect data.
Strictly Omissive Asymmetric	Some observers receive the same correct data and others declare invalid their received data.
Single-Data Omissive Asymmetric	Some observers accept the same incorrect data and others declare invalid their received data.
Transmissive Asymmetric	The observers have other patterns of disagreeing observations .

Table 3: Conditions used in the decision logic for classifying faults

Condition	Description
Consistently Invalid Data	All observers declare invalid their received data.
Consistently Valid Data	All observers declare valid their received data.
Symmetric Valid Data	All observers that declare valid their received data also agree on the data.
Correct Valid Data	All observer that declare valid their received data also have correct data.

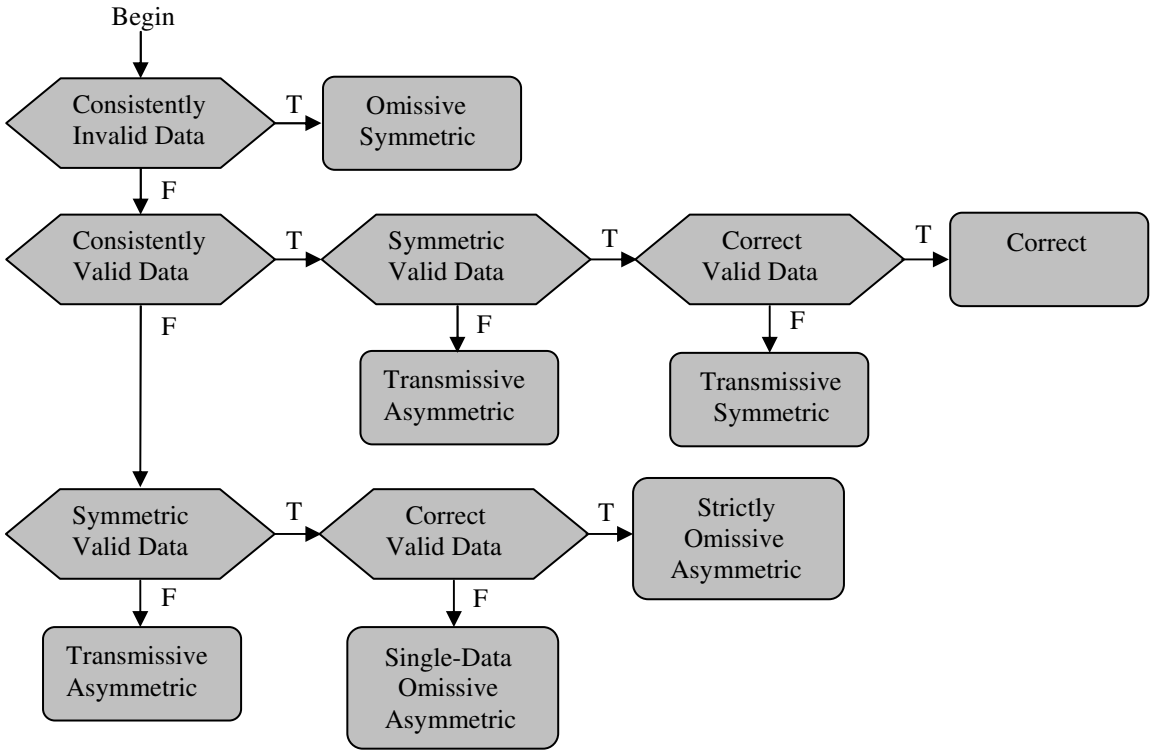


Figure 17: Decision logic used by the HFA for the classification of observation data

The observation data streams from the PEs cover the following expected messages: mode, Id, Schedule Update results, Schedule Update assessment, PE Broadcast, synchronization, BIU convictions, and RMU convictions. The error syndromes available from the PEs for each of these messages include communication (signal error, Manchester encoding error, and redundancy check error), timing (empty buffer, input overrun, and buffer overload), and expected content (compared against the valid content for the corresponding protocol). The HFA classifies the observations using two different validity criteria: one includes the communication and empty-buffer syndromes, and the other includes the communication and timing syndromes. The empty-buffer timing syndrome is used in both of these validity criteria since it is the one most likely to be present even in the simplest PE designs, since the most basic error check when expecting a message is whether any message is received at all.

For most expected messages, symmetry is determined by comparing the content of the PEs that receive valid data according to the applicable omission criteria. The only exceptions to this are the Id and synchronization messages. A direct content comparison is meaningless for the assessment of symmetry for Id messages since each PE expects to receive a different Id. Instead, symmetry for valid Id messages is replaced with a check of agreement that the valid Id messages are also individually correct for their respective PEs. The PEs determine correctness of the Id message by comparing the received Id against their hard-coded Id (see Figure 16). For the synchronization messages, the assessment of symmetry is expanded to include timing symmetry. When an engaged PE expects to receive a synchronization message, it generates a timing pulse upon arrival of a message. The timing of a synchronization message is symmetric if the precision of the pulses (i.e., the time between the earliest and the latest) from PEs with valid received messages is within the bound established by system timing analysis. The synchronization message is symmetric if it is symmetric in content and timing. Note that an expected message is always

declared symmetric if only one PE has a valid message.

The PEs perform a content-correctness check on every expected message and send the syndromes to the HFA as part of their observation data streams. The correct content for a message is determined based on the expected output when the SUT is working properly, which in the case of ROBUS-2 must take into consideration its fault-tolerant design with automatic reconfiguration and recovery mechanisms that enable it to operate even with faulty or recovering BIUs, RMUs, or PEs. Table 4 specifies the correct content for each expected SUT message. The *Node Id* is the value of the hard-coded Id given to each PE. The *Default Number of Messages per PE* is the predetermined number of messages per PE submitted during a schedule update. As stated previously, the PEs are designed to work with a static broadcast schedule in which all the PEs send the same number of messages. For synchronization and conviction messages, the HFA applies additional conditions to refine the definition of correctness. At the HFA, a valid synchronization message is correct if its content is correct and its timing is symmetric. In addition, a valid conviction message is correct if its content is correct (as per Table 4) and symmetric.

Table 4: Specification of correct content for expected messages at the PEs

Expected Message	**Correct ROBUS Message Content : (Tag, Payload)**
Mode	(SPECIAL, CLIQUE PRESERVATION)
Id	(DATA, *Node Id*)
Schedule Update Result	(DATA, *Default Number of Messages per PE*)
Schedule Update Assessment	(SPECIAL, VALID)
PE Broadcast	(DATA, *LFSR value*) for active source PEs; or (SPECIAL, PE_ERROR) or (SPECIAL, SOURCE_ERROR) for inactive source PEs
Synchronization Preservation	(SPECIAL, INIT)
BIU Convictions	Tag = DATA, Payload = {Four leftmost payload bits unspecified; remaining bits asserted}
RMU Convictions	Tag = DATA, Payload = {Four leftmost payload bits unspecified; remaining bits asserted}

The HFA is designed to operate with whichever PEs are active (i.e., engaged) with the SUT in Clique Preservation mode. It is assumed that the active PEs form a single, mutually synchronized clique. Immediately after an active PE receives a mode message indicating that its SUT port is in Clique Preservation mode, the PE outputs a timing message that enables the HFA to determine which PEs are active and mutually synchronized. The HFA will proceed with the classification of observations if it determines that there is a single PE clique. If at any time during a cycle the HFA detects the presence of active PEs that are not part of the same group, it aborts the current cycle and waits for a clique to appear. This feature of the HFA allows it to collect data in cycles where only a subset of the PEs is active. Note that the PEs are designed such that, once they have achieved synchronization with the SUT, they always complete a full cycle after receiving the expected Clique Preservation mode message at the beginning of the cycle. The PEs abort a cycle only at the beginning or at the end of a cycle.

Figure 18 shows a simplified view of the HFA's organization. The Active Port (AP) Monitor is enabled by the controller after a reset or at the beginning of a cycle to determine which PEs are active. When the active ports are ready, the fault classifiers begin reading observations from the buffer. The omission (i.e., validity) criteria for the first classifier include empty-buffer and all the communication syndromes. The second criteria add the remaining timing syndromes. The unexpected content syndrome

is not used for the current implementation. The controller reads the content labels and the end-of-cycle (EoC) flags sent by the PEs to enable the corresponding classification rules and to output the collected cycle data at the end of the cycle. The data collected by a fault classifier includes the number of observations for each category in the fault model. The number of unexpected messages received by each PE is stored directly in a separate dedicated register. If the AP Monitor detects a PE clique failure or the controller determines that the active ports do not agree on the content label or the end-of-cycle flag, then the current cycle is terminated immediately, the collected cycle data is output, and HFA is reset to restart its execution. Note that the HFA outputs the set of active ports for use by the PEs in their correct-content checks. Appendix A has a detailed description of the data collected by the HFA.

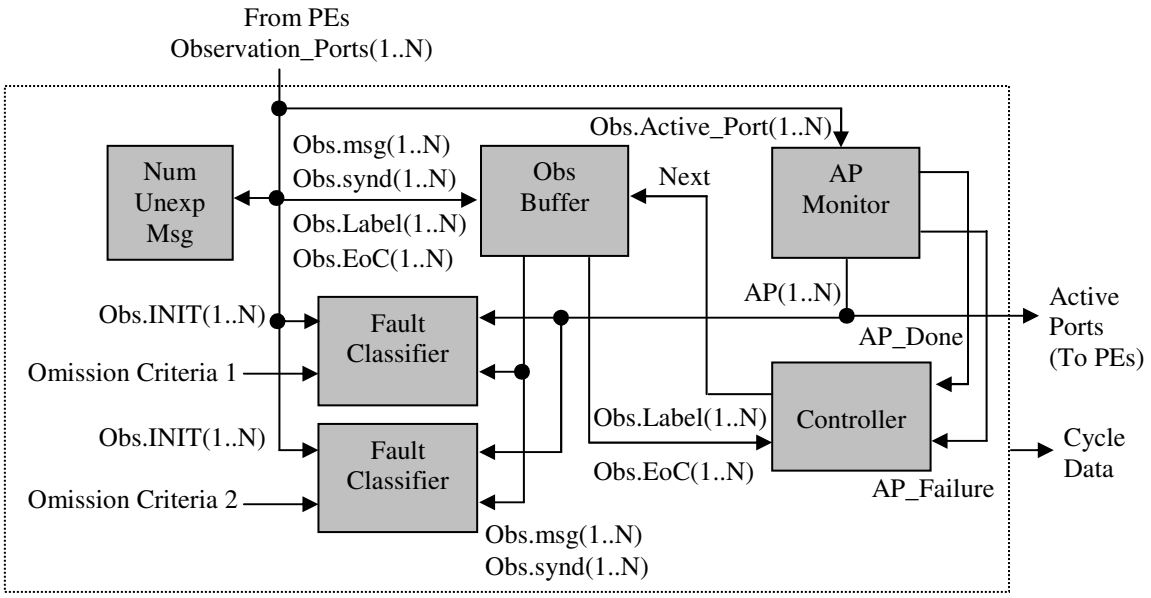


Figure 18: Simplified view of the Hub Fault Analyzer's organization

9. Node Fault Analysis System

Figure 19 illustrates the organization of the node-fault analysis system. This system is used only with the ROBUS 4x4 configuration with a single radiated node. The target node is either an RMU or a BIU, and the observers are the nodes of the opposite kind, either BIUs or RMUs, respectively. An observing RPP passes its observation data for each of the target node's expected message to the local Direct Node Observer (DNO), which packages the data into message frames that are sent over to a dedicated Node Observation Receiver (NOR). The NORs send their received observation records to the NFA, where the observations are integrated and classified according to the fault model described in Table 2. The criteria for omission, symmetry and correctness are described below.

The observers and the fault analyzer are in separate physical nodes with communication between DNOs and NORs via the custom fiber-optic data links. This arrangement unavoidably introduces a timing error component that must be taken into consideration when the NFA assesses timing symmetry for messages received by the observers. To reduce the size of this error, the NORs and the NFA are

driven by a faster clock signal than the observers (33 MHz for NORs and NFA vs. 3 MHz for the RPPs and DNOs). This faster clocking rate has the effect of increasing the timing resolution at the NFA, which enhances the time measurement accuracy.

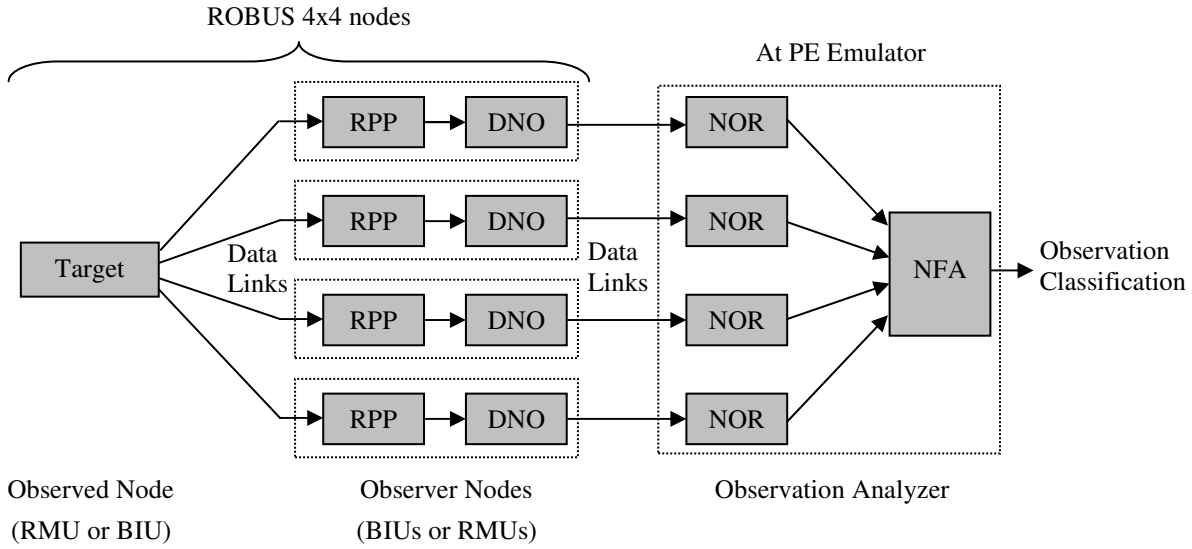


Figure 19: Node-fault analysis system

9.1. Direct Node Observer

The function of the DNO is to relay to its corresponding NOR the observation data produced by the RPP for each expected message from the target node. The DNO does this only when the RPP is in Clique Preservation mode. The DNO is programmed to track the commands from the RPP's Mode Command Unit (MCU) and the control signals of the RPP's data processing pipeline, from which it can determine the cycle boundaries and the type and timing of each expected message. The DNO packages the RPP's received message content and error syndromes into a two-message frame that also includes control information like cycle begin and end flags and a content label (which identifies the expected message as Schedule Update, PE Broadcast, INIT, ECHO, or Collective Diagnosis). For expected synchronization protocol messages, the DNO adds a third message whose relevant data is its timing and is sent a fixed delay after the message arrival time at the RPP.

The previous section on the ROBUS-2 SUT describes the modifications made to ROBUS-2 and the RPP to support the node-fault analysis function. In addition to the error syndromes generated by the RPP's inline checks (including communication, timing, and content syndromes), the RPP generates content and timing correctness syndromes by comparing the observations against the results of content and timing voting operation. The DNO is programmed assuming that the target node is connected to input lane 1 of the RPP.

Figure 20 shows a diagram of the DNO's organization. The MCU command and the timing of the RPP data processing units drive the controller. The first frame message (MSG0) is identified by a SPECIAL tag, and it carries control flags, error syndromes and the message tag received from the target

node. The second message (MSG1) has a DATA flag and contains the payload of the received message. These two messages are sent with the minimum allowed DII of 2 clock ticks. Therefore, the DNO requires that the RPP process data with a DII of at least 4 clock ticks. For INIT and ECHO synchronization messages, the content of the third message is the same as the content of the second one, as the only data of interest there is the timing of the message. The delay for transmitting the third message is set such that the NFA has had enough time to process the first two messages from all the observers before any third message arrives. If the RPP does not receive an expected synchronization message, the DNO sets the MSG0 control flags to indicate that the third message will not be sent. Although all the modules of the node-fault analysis system (i.e., RPP, DNO, NOR and NFA) have been implemented with the resources to process ECHO messages, the DNO's controller was revised to bypass sending that observation frame after timing analyses revealed that it is not possible to guarantee that there will not be a processing timing conflict at the NFA between the INIT and ECHO observations.

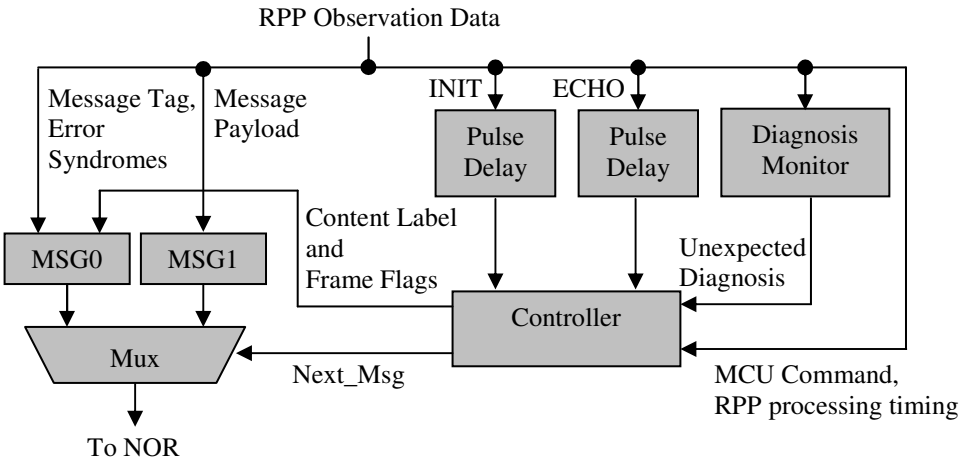


Figure 20: Organization of the Direct Node Observer

9.2. Node Observation Receiver

The purpose of the Node Observation Receiver (NOR) is to recover the RPP observations sent by the DNO. Figure 21 shows the organization of the NOR. The NOR synchronizes to the message stream from the DNO by monitoring the received message tags, where a SPECIAL tag signals the beginning of an observation frame. If a received frame indicates that it is the first one of a cycle, the NOR sends a pulse to the NFA signaling that the observation port (or channel) is active. For an INIT or ECHO frame with a third message, a corresponding timing pulse is generated upon arrival of the third message. If any of the data link error flags is asserted for a received message, the NOR asserts an observation-error signal declaring that a DNO frame is erroneous, to which the NFA responds by terminating processing of the current cycle observations and resetting the NORs.

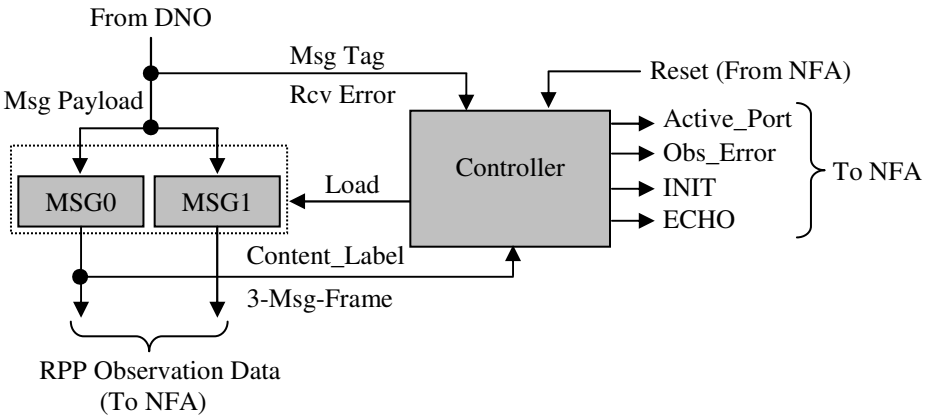


Figure 21: Organization of the Node Observation Receiver

9.3. Node Fault Analyzer

The Node Fault Analyzer (NFA) applies the same fault model to the observations of a node as the HFA does to the observations of an SUT. Figure 22 illustrates the NFA organization. The AP Monitor is different than the one in the HFA in that it requires that all ports be active and synchronized. A failure is declared if at any time there is any indication that the ports are not coordinated. The NFA has three fault classifiers with different omission criteria: 1) communication and empty-buffer syndromes; 2) communication and timing syndromes; and 3) communication, timing and content error syndromes. An observation is content correct if all the observers with valid data agree that the message content they received is correct. Synchronization messages are timing correct if all the observers with valid data agree that the timing of the message they received is correct and the timing of those messages is symmetric (i.e., in approximate timing agreement) across those observers. The worst-case precision for symmetric timing is determined by timing analysis. A synchronization message is correct if it is content and timing correct.

Appendix A describes the cycle data generated by the NFA.

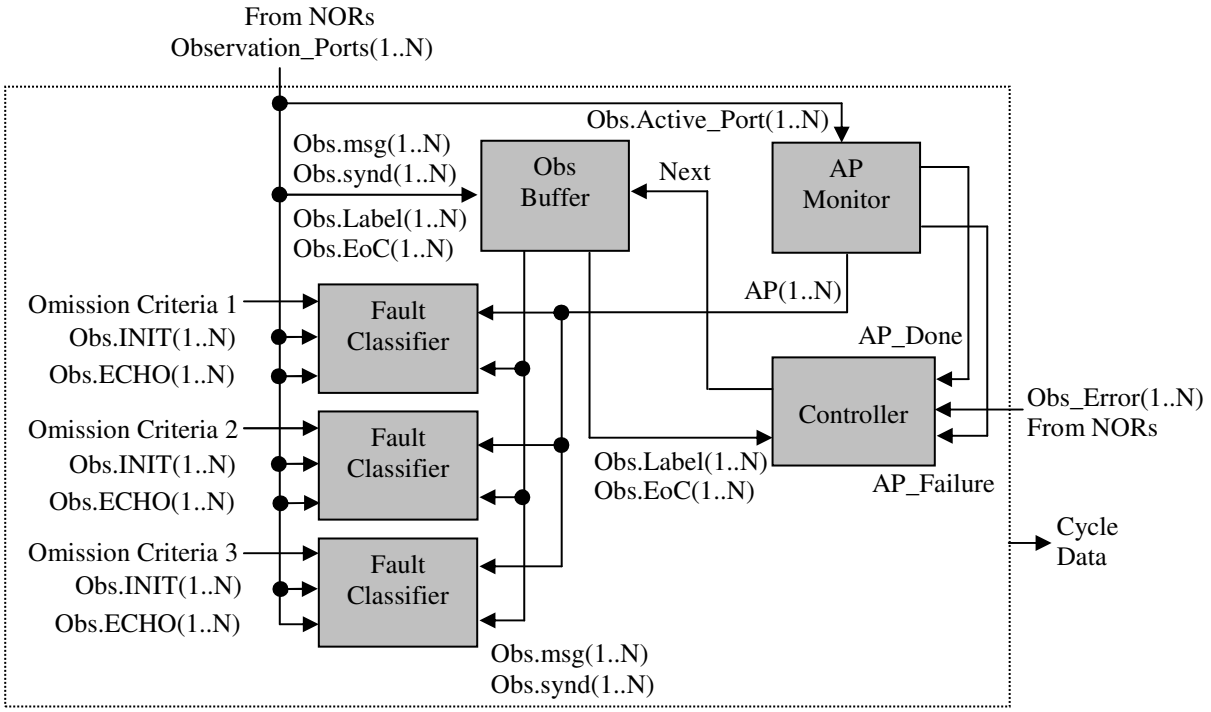


Figure 22: Organization of the Node Fault Analyzer

10. ROBUS-2 State Monitoring System

The ROBUS-2 state monitoring system consists of independent RPP state monitoring channels for each BIU and RMU node. Figure 23 shows the interconnection of components for monitoring one RPP. Each ROBUS-2 node has an Embedded Node Monitor (ENM) that takes samples of the state through the RPP's test port and relays them to the Remote Node Monitor (RNM) in the form of three-message frames. All the RNMs are located at the Bus Monitor computer, where the state updates are collected for real-time system monitoring and post-test analysis.

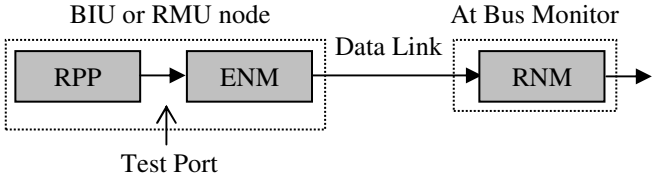


Figure 23: RPP state monitoring system

10.1. Embedded Node Monitor

Figure 24 shows a simplified representation of the ENM's organization. The collected state data includes error syndromes from the RPP's Status Monitoring Unit (SMU), the command generated by the

Mode Control Unit (MCU) and the node accusations and convictions from the Node Diagnostics Unit (NDU). This state record is packaged into a three-message frame, where a SPECIAL tag identifies the first message and the other messages are given DATA tags. The messages are sent with the minimum DII of 2 clock ticks allowed by the fiber-optic data link. The controller is triggered by the MCU commands or by a failure detected by the SMU. The ENM generates four state data frames per cycle during normal operation with the RPP in Clique Preservation mode. Because the MCU commands and SMU failure detection are not mutually exclusive in time, there is the possibility of both of these state snapshot triggers coinciding. In that case, the first one asserted will trigger a frame transmission, and the second one will be ignored.

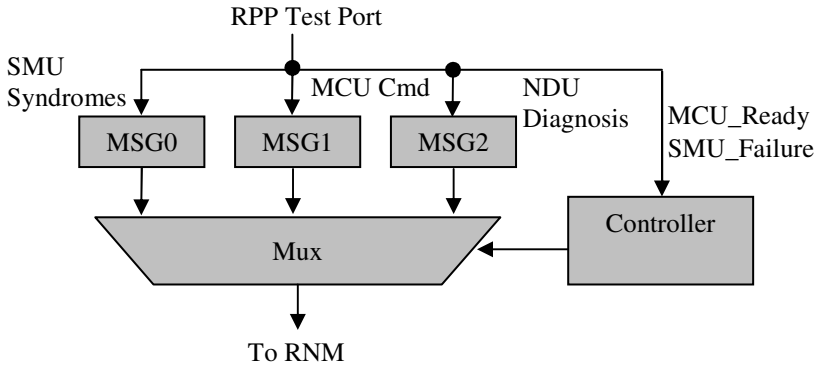


Figure 24: Organization of the Embedded Node Monitor

10.2. Remote Node Monitor

The RNM recovers the state data frames. Figure 25 shows a simplified view of the organization. The controller inspects the received message tag to identify the beginning of a frame. This allows the RNM to resynchronize to the incoming data stream after disruptions due to communication errors or the monitored node temporarily going offline. The received message payloads are stored as they arrive. The state data record includes a time tag, a frame sequence number and frame diagnosis information. The controller performs error checks while a data frame is being received, including monitoring for data link errors and measuring the time interval between intra-frame messages. The frame diagnosis information includes the Halt Code indicating the condition under which the recovery of the frame was ended (e.g., normal, timing violation, etc), and the actual number of messages received. Appendix A describes in detail the content of the RPP state record.

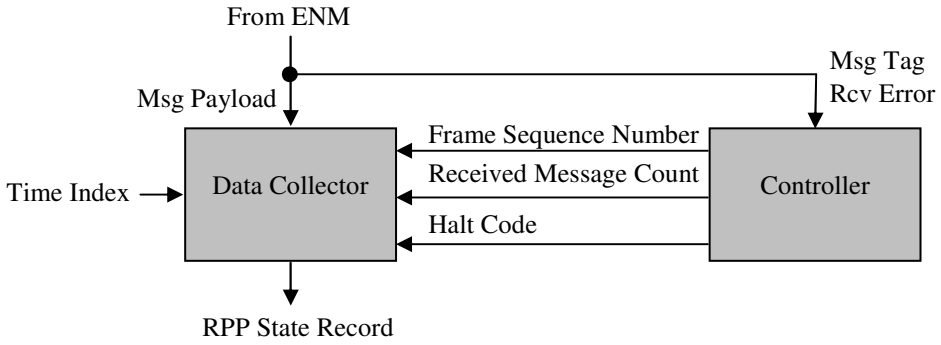


Figure 25: Simplified view of the Remote Node Monitor's organization

11. Hub State Monitoring System

The hub state monitoring system has the same structure as the ROBUS-2 state monitoring system, including an Embedded Hub Monitor (EHM) that takes samples of the internal state of the hub through its test port and relays that data to a Remote Hub Monitor (RHM) on the Bus Monitor computer. The EHM samples the hub's state triggered by the controller commands or a failure indication. During normal, error-free operation, the EHM samples the hub state four times per cycle. An EHM data frame consists of a single message. The RHM recovers the state sample message and adds a time index and a sequence number to form a state record. Appendix A describes the content of a hub's state-sample record.

12. System Health Monitor

The System Health Monitor (SHM) is a generic module that provides a real-time assessment of the status of an SUT. The SHM has a reusable design that leverages the highly deterministic and cyclic operation of the SUTs and knowledge of their recovery characteristics. The SHM is instantiated at the output of every module collecting data on the SUT behavior, including PEs, HFA, NFA and state monitors. When the SUT is in steady state (i.e., in Clique Preservation mode) without internal errors, each of those monitoring modules outputs a repetitive data record sequence with known record content and periodicity. The SHM can detect the activation of faults by comparing the actual records against known-good reference records. Figure 26 illustrates the SHM state transition graph. The design of the SHM covers the scenario in which the SHM is enabled before the SUT, in which case the SHM will wait for a period of time before declaring the SUT failed. When the SHM receives a record matching one of the known-good reference records, it transitions to the Recovering state, where it waits until it has confirmation that the SUT is working normally. Confirmation consists of receiving only records that match the reference records in content and timing for a given minimum time duration, referred to as the stable recovery delay. When normal operation is confirmed, the SHM transitions to the Trusted state, where it will remain as long as new records match the references. Every time the SHM receives an unexpected record, it transitions to the Recovering state and starts the failed recovery delay timer. If the SHM remains continuously in the Recovering state, it eventually times out and declares the SUT failed. The duration of the inactive, stable recovery and failed recovery delays is based on the operation of the Test Control System (TCS) [1] and the recovery characteristics of the SUTs.

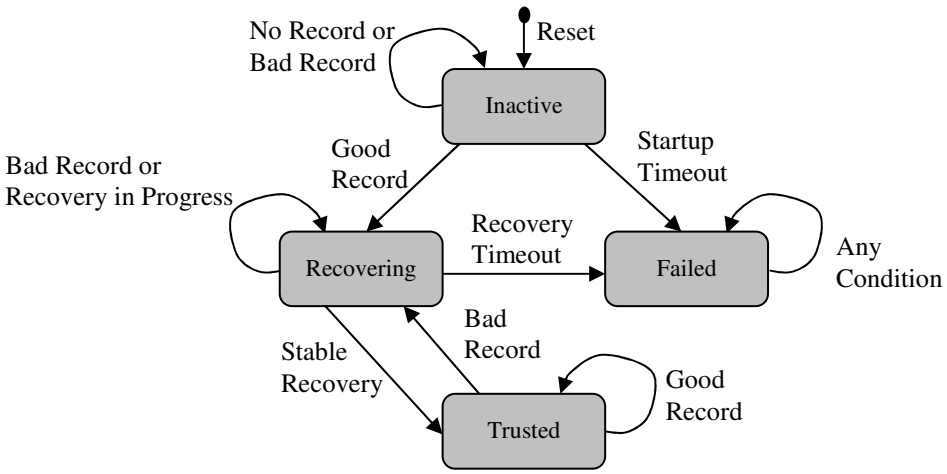


Figure 26: State transition graph of System Health Monitor

Figure 27 provides a simplified view of the SHM organization. The content and timing reference records and the content comparison functions are customized for the type of module that the SHM instance is attached to. The System Diagnostics Unit (SDU) and Timing Check module are the reusable components. The status output indicates the current state of the SHM. The Failure flag is asserted when the controller transitions to the Failed state. Note that there are no exit branches from that state, and only a reset can return the SHM to operation. The outputs of the SHM are appended to each data record for use by the TCS and for post-test analysis. Wherever the SHM is used, the assertion of the SHM Failure output flag triggers the generation of a new record with the latest SHM data and arbitrary content for the fields controlled by the monitored system.

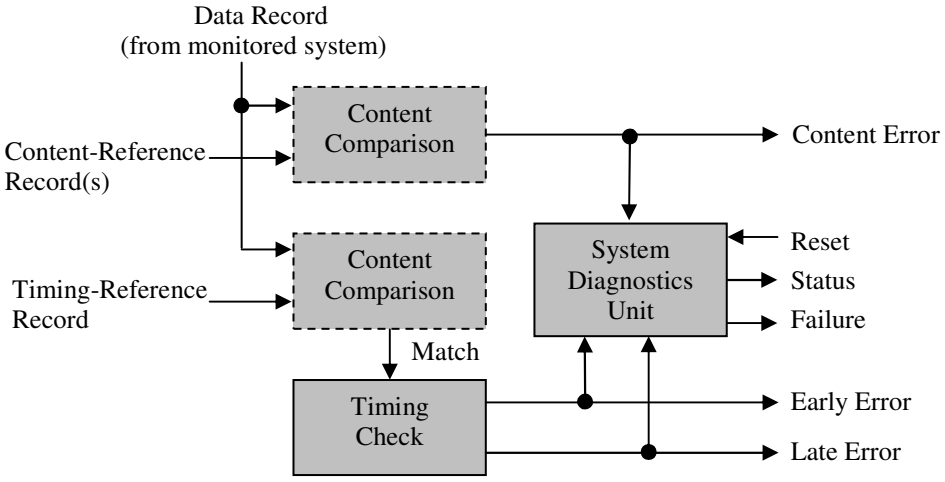


Figure 27: Organization of the System Health Monitor

13. Concluding Remarks

The SUTs and the monitoring system are implemented on a number of FPGA configuration programs for the RSPP nodes. For ROBUS-2 SUTs, two different FPGA programs were created for each node kind (i.e., BIU and RMU) to support the high- and low-speed communication system versions. There are also two FPGA programs for the two message rate versions of the simplex hub. The PE Emulator, which hosts the PEs, HFA and NFA functions, requires a different FPGA for each combination of SUT configuration (including ROBUS 4x2, 4x3, and 4x4, and the simplex hub) and communication speed. Each of the ROBUS-2 configurations requires a different PE Emulator FPGA program because they have different sets of SHM reference records. In total, 8 FPGA programs were created for the PE Emulator. The Bus Monitor requires one FPGA program for each SUT configuration, but the same FPGA can be used with both communication speed versions of an SUT. The reason for this is that the RPP and the simplex hub core are driven at a clock rate (3 MHz) that is independent of the actual message rate, which is determined by the message transmission and processing DII setting. As in the case of the PE Emulator, the reason for having a different FPGA program for each ROBUS-2 configuration is that each one has different SHM reference records. Thus, a total of 4 Bus Monitor FPGA programs were created (for 3 ROBUS-2 SUTs and the hub). Overall, including the SUTs, the PE Emulator and the Bus Monitor, 16 FPGA programs have been created for the HIRF test.

The ROBUS-2 RPP is described using the VHDL language, and its design is parameterized in the behavioral and structural domains. This flexible description enables its use in a wide range of physical implementation platforms, including reconfigurable FPGA-based platforms like the RSPP. This style of description was preferred as it adds portability to the design and reduces the failure risk of the development process by providing increased assurance that unexpected “quirks” of the physical platform can be accommodated without the need for large design changes.

The simplex hub and all the SUT monitoring systems for the HIRF test were designed following the same parameterization principle used for the RPP. A MATLAB script, originally developed to analyze the system-level parameters of a ROBUS-2 system and derive corresponding parameter values for the RPP, has been expanded to include the analysis of all the new systems and to compute the behavioral and structural parameter values for all the components. This script is instrumental in tuning the implementation to support the test plan described in [1] and ensuring that all the configurations of this complex communication and monitoring system perform flawlessly.

The design and implementation of the monitoring system followed a spiraling process of discovery of requirements and continuous design refinement supported with various small-scale concept-testing activities intended to gain insight into the operation of the physical hardware and available RSPP node software. This process minimized the development risk and allowed the creation of a design with a high degree of assurance of correctness. This approach also simplified the process of final design implementation and verification by reducing that step to a mapping from one design description to another supported by a high certainty that any observed errors in the implementation had their origin in the implementation description itself and not in the design. The allocation of processes between hardware and software at the PE Emulator and Bus Monitor was influenced by the observed execution timing uncertainty of the software, the high degree of available timing control at the FPGA hardware level, the large available hardware real-estate on the FPGAs, and the limited available bandwidth on the ISA bus connecting the CPU and FPGA hardware module of the RSPP nodes (see Figure 4). The timing of execution of software running on the available version of Linux, which was not a real-time-operating-system (RTOS) version, was observed to be highly irregular with frequent relatively long lapses in which the CPU would not be available to interact with the FPGA module, especially when updating the operator

monitoring display. This problem of loose software timing margins and the limited hardware-software interface bandwidth was handled by allocating all the extremely timing critical processes to hardware, by compacting the hardware data records as tight as possible (without sacrificing the quality of the data; see Appendices A and B), and by providing large output record buffers in the hardware to ensure that, even in the worst case, none of the generated records would be lost due to uncertainty in the interval between data transfers to the software. The insight gained with the spiraling development process also helped in the design and implementation of the CCP software processes as it was found that a simple top-level finite-state-machine (FSM) design significantly reduced the complexity of the solution by introducing modularity in the description and the execution. This solution mapped the problem into a series of small problems that were easier to reason about, easily accommodated local (i.e., within a particular computer) and global (i.e., at the TCS level) exception handling, simplified the CCP implementation verification, simplified the introduction of modifications to the CCP (by localizing the impact of design changes), simplified the integration of the distributed CCP processes (at the Test Controller, PE Emulator, and Bus Monitor), and simplified the local interaction between the software and hardware processes. This structured and simple design provides a strong degree of confidence about the safety (for equipment and personnel) of this highly automated HIRF test. The automation also enables the execution of a large multi-part test plan [1] with relative ease.

Appendix A. Description of Collected Data Records

The PE Emulator and the Bus Monitor collect the data records generated by their SUT monitoring hardware modules and store them in files for post-test analysis. This section describes the content of these data records.

A.1. PE Emulator Data

The PE Emulator collects data from the PEs, the HFA, and NFA.

A.1.1. PE Data Record

Table A.1 describes the content of the PE data record, which is generated by the PE after initial synchronization and at the end of every cycle.

Table A.1: Data record generated by a Processing Element

Record Field	Data Type	Range	Description
Time Index	Integer	0 to 2^{31} - 1	Driven by a 3 MHz clock (333 ns period); Rolls over after reaching the maximum value (in approximately 11.9 minutes)
Sequence Number	Integer	0 to 2^{8} - 1	Incremented with every new record; Rolls over after reaching the maximum value
SHM Health Status	Enumerated	{Inactive, Recovering, Trusted, Failed}	See Figure 26 for SHM state transition graph
SHM Content Error	Boolean	{False, True}	An error indicates that the record content from the PE does not match any of the reference records.
SHM Early Error	Boolean	{False, True}	An error indicates that the record content from the PE matches the timing reference record but it was generated earlier than expected with respect to the previous record matching the timing reference.
SHM Late Error	Boolean	{False, True}	An error indicates that the record content from the PE matches the timing reference record but it was generated later than expected with respect to the previous record matching the timing reference.
SHM Failure	Boolean	{False, True}	Indicates that the SHM has transitioned to the Failed state.
PE Id	Integer	1 to 4	Identifies the source PE for the record
Cycle Code	Enumerated	{Normal_Cycle, Sync_Cycle, Sync_Failure, Mode_Error}	- Normal_Cycle = Received valid mode and synchronization messages; - Sync_Cycle = Successful initial synchronization; - Sync_Failure = Received invalid synchronization message; - Mode_Error = Received invalid or unexpected mode message (“unexpected” means other than Clique Preservation mode after PE_Engaged)

Record Field	Data Type	Range	Description
PE Engaged	Boolean	{False, True}	False = PE is initializing; True = PE is in normal operation mode (Indicates that since the last reset the PE has synchronized to the SUT and received a Clique Preservation mode message.)
Mode	ROBUS Message	(Tag, Payload) Valid Tag: SPECIAL Valid Payload: { 0 = Self Test, 1 = Clique Detection, 2 = Clique Join, 3 = Clique Initialization, 4 = Clique Preservation}	Mode message received from the SUT
Correct BIU Id	Boolean	{False, True}	True = Received valid Id message (DATA, Id) with payload matching the hardcoded Id given to the PE

Schedule Update Result 1	ROBUS Message	(Tag, Payload)
		Valid Tag: DATA
		Valid Payload: {
		bits 5-16: Integer in range 0 to 2^{12} - 1}
		or
		Valid Tag: SPECIAL
		Valid Payload: {
		bits 5-16:
		$10_{decimal}$ = PE_ERROR}
			DATA tag indicates that the payload value is the number of scheduled messages for PE 1;
			(SPECIAL, PE_ERROR) message indicates that the SUT did not receive a valid number of scheduled messages for PE 1;
			Any other message content for bits 5-16 indicates an error;
			Bit 1 = leftmost payload bit.
			---------- 	
			Extra payload content:
			bit 1 = Unexpected-content syndrome;
			bit 2 = Combined timing error syndrome, except empty-buffer;
			bit 3 = Empty-buffer timing syndrome;
			bit 4 = Combined communication error syndrome
			
			The additional information carried by the Payload field are the reception syndromes for this message. Bit 2 is the result of OR'ing the buffer-overload and input-overrun timing syndromes. Bit 4 is the result of OR'ing the data link syndromes: signal error, Manchester coding error, and redundancy check error.

Schedule Update Result 2	ROBUS Message	(Tag, Payload)
		Valid Tag: DATA
		Valid Payload: {
		bits 5-16: Integer in range 0 to 2^{12} - 1}
		or
		Valid Tag: SPECIAL
		Valid Payload: {
		bits 5-16:
		$10_{decimal}$ = PE_ERROR}
			DATA tag indicates that the payload value is the number of scheduled messages for PE 2;
			(SPECIAL, PE_ERROR) message indicates that the SUT did not receive a valid number of scheduled messages for PE 2;
			Any other message content for bits 5-16 indicates an error;
			Bit 1 = leftmost payload bit.
			---------- 	
			Extra payload content:
			bit 1 = Unexpected-content syndrome;
			bit 2 = Combined timing error syndrome, except empty-buffer;
			bit 3 = Empty-buffer timing syndrome;
			bit 4 = Combined communication error syndrome
			
			The additional information carried by the Payload field

Record Field	Data Type	Range	Description
			are the reception syndromes for this message. Bit 2 is the result of OR'ing the buffer-overload and input-overrun timing syndromes. Bit 4 is the result of OR'ing the data link syndromes: signal error, Manchester coding error, and redundancy check error.

Schedule Update Result 3	ROBUS Message	(Tag, Payload)
		Valid Tag: DATA
		Valid Payload: {
		bits 5-16: Integer in range 0 to 2^{12} - 1}
		or
		Valid Tag: SPECIAL
		Valid Payload: {
		bits 5-16:
		$10_{decimal}$ = PE_ERROR}
			DATA tag indicates that the payload value is the number of scheduled messages for PE 3;
			(SPECIAL, PE_ERROR) message indicates that the SUT did not receive a valid number of scheduled messages for PE 3;
			Any other message content for bits 5-16 indicates an error;
			Bit 1 = leftmost payload bit.
			---------- 	
			Extra payload content:
			bit 1 = Unexpected-content syndrome;
			bit 2 = Combined timing error syndrome, except empty-buffer;
			bit 3 = Empty-buffer timing syndrome;
			bit 4 = Combined communication error syndrome
			
			The additional information carried by the Payload field are the reception syndromes for this message. Bit 2 is the result of OR'ing the buffer-overload and input-overrun timing syndromes. Bit 4 is the result of OR'ing the data link syndromes: signal error, Manchester coding error, and redundancy check error.

Schedule Update Result 4	ROBUS Message	(Tag, Payload)
		Valid Tag: DATA
		Valid Payload: {
		bits 5-16: Integer in range 0 to 2^{12} - 1}
		or
		Valid Tag: SPECIAL
		Valid Payload: {
		bits 5-16:
		$10_{decimal}$ = PE_ERROR}
			DATA tag indicates that the payload value is the number of scheduled messages for PE 4;
			(SPECIAL, PE_ERROR) message indicates that the SUT did not receive a valid number of scheduled messages for PE 4;
			Any other message content for bits 5-16 indicates an error;
			Bit 1 = leftmost payload bit.
			---------- 	
			Extra payload content:
			bit 1 = Unexpected-content syndrome;
			bit 2 = Combined timing error syndrome, except empty-buffer;
			bit 3 = Empty-buffer timing syndrome;
			bit 4 = Combined communication error syndrome
			
			The additional information carried by the Payload field are the reception syndromes for this message. Bit 2 is the result of OR'ing the buffer-overload and input-overrun timing syndromes. Bit 4 is the result of OR'ing the data link syndromes: signal error, Manchester coding error, and redundancy check error.

Record Field	Data Type	Range	Description
Schedule Update Assessment	ROBUS Message	(Tag, Payload) Valid Tag: SPECIAL Valid Payload: { 5 = Valid, 6 = Zero, 7 = Invalid}	Any other message content indicates an error.
Number of Correct PE Messages	Integer	0 to 2^{16} - 1	Total number of messages during the PE Broadcast phase that matched the expected content for error-free operation

BIU Convictions	ROBUS Message	(Tag, Payload)
		Valid Tag: DATA
		Valid Payload: {
		bits 1-4 = Conviction BIUs 1-4}
			Any other message content indicates an error;
			Bit 1 = leftmost bit;
			0/1 = False/True.
			---------- 	
			Extra payload content:
			- For BIU Convictions message:
			bit 5 = Unexpected-content syndrome;
			bit 6 = Combined timing error syndrome, except empty-buffer;
			bit 7 = Empty-buffer timing syndrome;
			bit 8 = Combined communication error syndrome;
			
			- For Id message:
			bit 9 = Unexpected-content syndrome;
			bit 10 = Combined timing error syndrome, except empty-buffer;
			bit 11 = Empty-buffer timing syndrome;
			bit 12 = Combined communication error syndrome;
			
			- For Mode message:
			bit 13 = Unexpected-content syndrome;
			bit 14 = Combined timing error syndrome, except empty-buffer;
			bit 15 = Empty-buffer timing syndrome;
			bit 16 = Combined communication error syndrome;
			
			The additional information carried by the Payload field are the reception syndromes for messages BIU Convictions, Mode, and Id. Bits 6, 10, and 14 are the result of OR'ing the buffer-overload and input-overrun timing syndromes. Bits 8, 12, and 16 are the result of OR'ing the data link syndromes: signal error, Manchester coding error, and redundancy check error.

RMU Convictions	ROBUS Message	(Tag, Payload)
		Valid Tag: DATA
		Valid Payload: {
		bits 1-4 = Conviction BIUs 1-4}
			Any other message content indicates an error;
			Bit 1 = leftmost bit;
			0/1 = False/True.
			---------- 	
			Extra payload content:
			- For RMU Convictions message:
			bit 5 = Unexpected-content syndrome;
			bit 6 = Combined timing error syndrome, except empty-buffer;
			bit 7 = Empty-buffer timing syndrome;
			bit 8 = Combined communication error syndrome;

Record Field	Data Type	Range	Description

			- For INIT message:
			bit 9 = Unexpected-content syndrome;
			bit 10 = Combined timing error syndrome, except empty-buffer;
			bit 11 = Empty-buffer timing syndrome;
			bit 12 = Combined communication error syndrome;
			
			- For Schedule Update Assessment message:
			bit 13 = Unexpected-content syndrome;
			bit 14 = Combined timing error syndrome, except empty-buffer;
			bit 15 = Empty-buffer timing syndrome;
			bit 16 = Combined communication error syndrome;
			
			The additional information carried by the Payload field are the reception syndromes for messages RMU Convictions, Mode, and Id. Bits 6, 10, and 14 are the result of OR'ing the buffer-overload and input-overrun timing syndromes. Bits 8, 12, and 16 are the result of OR'ing the data link syndromes: signal error, Manchester coding error, and redundancy check error.

A.1.2. HFA Data Record

The HFA generates a data record as described in Table A.2 at the end of every SUT cycle if at least one PE is engaged.

Table A.2: Data record generated by Hub Fault Analyzer

Record Field	Data Type	Range	Description
Time Index	Integer	0 to 2^{31} - 1	Driven by a 3 MHz clock (333 ns period); Rolls over after reaching the maximum value (in approximately 11.9 minutes)
Sequence Number	Integer	0 to 2^{8} - 1	Incremented with every new record; Rolls over after reaching the maximum value
SHM Health Status	Enumerated	{Inactive, Recovering, Trusted, Failed}	See Figure 26 for SHM state transition graph
SHM Content Error	Boolean	{False, True}	An error indicates that the record content from the HFA does not match any of the reference records.
SHM Early Error	Boolean	{False, True}	An error indicates that the record content from the HFA matches the timing reference record but it was generated earlier than expected with respect to the previous record matching the timing reference.
SHM Late Error	Boolean	{False, True}	An error indicates that the record content from the HFA matches the timing reference record but it was generated later than expected with respect to the previous record matching the timing reference.

Record Field	Data Type	Range	Description
SHM Failure	Boolean	{False, True}	Indicates that the SHM has transitioned to the Failed state.
Cycle Code	Enumerated	{Normal_Cycle, AP_Failure, AP_Buf_Timeout, AP_Label_Asymmetry, AP_End_Asymmetry}	- Normal_Cycle = Cycle completed without unexpected events; - AP_Failure = Active PEs do not form a single, synchronized clique (detected by monitoring beginning of PE cycles); - AP_Buf_Timeout = Active PEs failed to output an observation within the expected timing precision (detected by monitoring PE observations during a cycle); - AP_Label_Asymmetry = Active PEs disagree on the content label of an observation; - AP_End_Asymmetry = Active PEs disagree on the end of a cycle
Active Port 1	Boolean	{False, True}	True = PE 1 is active and included in the analysis of observations
Active Port 2	Boolean	{False, True}	True = PE 2 is active and included in the analysis of observations
Active Port 3	Boolean	{False, True}	True = PE 3 is active and included in the analysis of observations
Active Port 4	Boolean	{False, True}	True = PE 4 is active and included in the analysis of observations
Number of Unexpected Messages 1	Integer	0 to 2^{16} - 1	Number of messages received by PE 1 outside of the expected reception intervals during the cycle
Number of Unexpected Messages 2	Integer	0 to 2^{16} – 1	Number of messages received by PE 2 outside of the expected reception intervals during the cycle
Number of Unexpected Messages 3	Integer	0 to 2^{16} – 1	Number of messages received by PE 3 outside of the expected reception intervals during the cycle
Number of Unexpected Messages 4	Integer	0 to 2^{16} - 1	Number of messages received by PE 4 outside of the expected reception intervals during the cycle
Omission Criteria: C, CORR value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Correct content
Omission Criteria: C, OS value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Omissive Symmetric content
Omission Criteria: C, TS value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Transmissive Symmetric content
Omission Criteria: C, SOA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Strictly Omissive Asymmetric content

Record Field	Data Type	Range	Description
Omission Criteria: C, SDOA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Single-Data Omissive Asymmetric content
Omission Criteria: C, TA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Transmissive Asymmetric content
Omission Criteria: C, Timing Classification	Enumerated	{CORR, OS, TS, SOA, SDOA, TA}	Omission Criteria: Communication and empty-buffer syndromes; Classification of the synchronization message (INIT); CORR = Correct; OS = Omisssive Symmetric; TS = Transmissive Symmetric; SOA = Strictly Omissive Asymmetric; SDOA = Single-Data Omissive Asymmetric; TA = Transmissive Asymmetric
Omission Criteria: CT, CORR value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Correct content
Omission Criteria: CT, OS value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Omissive Symmetric content
Omission Criteria: CT, TS value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Transmissive Symmetric content
Omission Criteria: CT, SOA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Strictly Omissive Asymmetric content
Omission Criteria: CT, SDOA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Single-Data Omissive Asymmetric content
Omission Criteria: CT, TA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Transmissive Asymmetric content
Omission Criteria: CT, Timing Classification	Enumerated	{Corr, OS, TS, SOA, SDOA, TA}	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Classification of the synchronization message (INIT); CORR = Correct; OS = Omisssive Symmetric; TS = Transmissive Symmetric; SOA = Strictly Omissive Asymmetric; SDOA = Single-Data Omissive Asymmetric; TA = Transmissive Asymmetric

A.1.3. NFA Data Record

The NFA, used only with the ROBUS 4x4 SUT, generates the data record described in Table 3.1 at the end of every Clique Preservation mode cycle.

Table A.3: Data record generated by Node Fault Analyzer

Record Field	Data Type	Range	Description
Time Index	Integer	0 to 2^{31} - 1	Driven by a 3 MHz clock (333 ns period); Rolls over after reaching the maximum value (in approximately 11.9 minutes)
Sequence Number	Integer	0 to 2^{8} - 1	Incremented with every new record; Rolls over after reaching the maximum value
SHM Health Status	Enumerated	{Inactive, Recovering, Trusted, Failed}	See Figure 26 for SHM state transition graph
SHM Content Error	Boolean	{False, True}	An error indicates that the record content from the NFA does not match the any of the reference records.
SHM Early Error	Boolean	{False, True}	An error indicates that the record content from the NFA matches the timing reference record but it was generated earlier than expected with respect to the previous record matching the timing reference.
SHM Late Error	Boolean	{False, True}	An error indicates that the record content from the NFA matches the timing reference record but it was generated later than expected with respect to the previous record matching the timing reference.
SHM Failure	Boolean	{False, True}	Indicates that the SHM has transitioned to the Failed state.
Cycle Code	Enumerated	{Normal_Cycle, AP_Failure, AP_Buf_Timeout, AP_Field_Asymmetry, NOR_Obs_Error}	- Normal_Cycle = Cycle completed without unexpected events; - AP_Failure = Active observers do not form a single, synchronized clique (detected by monitoring beginning of observation cycles); - AP_Buf_Timeout = Active observers failed to output an observation within the expected timing precision (detected by monitoring observations during a cycle); - AP_Field_Asymmetry = Active observers disagree on the value of one or more observation control fields (which include start-of-cycle, end-of-cycle, and content label); - NOR_Obs_Error = Observation reception error detected by one or more NORs
Active Port 1	Boolean	{False, True}	True = Observer 1 (BIU or RMU) is active and included in the analysis of observations
Active Port 2	Boolean	{False, True}	True = Observer 2 (BIU or RMU) is active and included in the analysis of observations
Active Port 3	Boolean	{False, True}	True = Observer 3 (BIU or RMU) is active and included in the analysis of observations
Active Port 4	Boolean	{False, True}	True = Observer 4 (BIU or RMU) is active and included in the analysis of observations

Record Field	Data Type	Range	Description
Omission Criteria: C, CORR value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Correct content
Omission Criteria: C, OS value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Omissive Symmetric content
Omission Criteria: C, TS value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Transmissive Symmetric content
Omission Criteria: C, SOA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Strictly Omissive Asymmetric content
Omission Criteria: C, SDOA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Single-Data Omissive Asymmetric content
Omission Criteria: C, TA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and empty-buffer syndromes; Number of messages with Transmissive Asymmetric content
Omission Criteria: C, INIT message Classification	Enumerated	{CORR, OS, TS, SOA, SDOA, TA}	Omission Criteria: Communication and empty-buffer syndromes; Classification of the INIT message; CORR = Correct; OS = Omisssive Symmetric; TS = Transmissive Symmetric; SOA = Strictly Omissive Asymmetric; SDOA = Single-Data Omissive Asymmetric; TA = Transmissive Asymmetric
Omission Criteria: C, ECHO message Classification	Enumerated	{CORR, OS, TS, SOA, SDOA, TA}	Omission Criteria: Communication and empty-buffer syndromes; Classification of the ECHO message; CORR = Correct; OS = Omisssive Symmetric; TS = Transmissive Symmetric; SOA = Strictly Omissive Asymmetric; SDOA = Single-Data Omissive Asymmetric; TA = Transmissive Asymmetric
Omission Criteria: CT, CORR value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Correct content
Omission Criteria: CT, OS value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Omissive Symmetric content

Record Field	Data Type	Range	Description
Omission Criteria: CT, TS value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Transmissive Symmetric content
Omission Criteria: CT, SOA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Strictly Omissive Asymmetric content
Omission Criteria: CT, SDOA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Single-Data Omissive Asymmetric content
Omission Criteria: CT, TA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Number of messages with Transmissive Asymmetric content
Omission Criteria: CT, INIT message Classification	Enumerated	{CORR, OS, TS, SOA, SDOA, TA}	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Classification of the INIT message; CORR = Correct; OS = Omisssive Symmetric; TS = Transmissive Symmetric; SOA = Strictly Omissive Asymmetric; SDOA = Single-Data Omissive Asymmetric; TA = Transmissive Asymmetric
Omission Criteria: CT, ECHO message Classification	Enumerated	{CORR, OS, TS, SOA, SDOA, TA}	Omission Criteria: Communication and Timing Syndromes including empty-buffer; Classification of the ECHO message; CORR = Correct; OS = Omisssive Symmetric; TS = Transmissive Symmetric; SOA = Strictly Omissive Asymmetric; SDOA = Single-Data Omissive Asymmetric; TA = Transmissive Asymmetric
Omission Criteria: CTC, CORR value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication, Timing and Content Syndromes including empty-buffer; Number of messages with Correct content
Omission Criteria: CTC, OS value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication, Timing and Content Syndromes including empty-buffer; Number of messages with Omissive Symmetric content
Omission Criteria: CTC, TS value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication, Timing and Content Syndromes including empty-buffer; Number of messages with Transmissive Symmetric content

Record Field	Data Type	Range	Description
Omission Criteria: CTC, SOA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication, Timing and Content Syndromes including empty-buffer; Number of messages with Strictly Omissive Asymmetric content
Omission Criteria: CTC, SDOA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication, Timing and Content Syndromes including empty-buffer; Number of messages with Single-Data Omissive Asymmetric content
Omission Criteria: CTC, TA value count	Integer	0 to 2^{16} - 1	Omission Criteria: Communication, Timing and Content Syndromes including empty-buffer; Number of messages with Transmissive Asymmetric content
Omission Criteria: CTC, INIT message Classification	Enumerated	{CORR, OS, TS, SOA, SDOA, TA}	Omission Criteria: Communication, Timing and Content Syndromes including empty-buffer; Classification of the INIT message; CORR = Correct; OS = Omisssive Symmetric; TS = Transmissive Symmetric; SOA = Strictly Omissive Asymmetric; SDOA = Single-Data Omissive Asymmetric; TA = Transmissive Asymmetric
Omission Criteria: CTC, ECHO message Classification	Enumerated	{CORR, OS, TS, SOA, SDOA, TA}	Omission Criteria: Communication, Timing and Content Syndromes including empty-buffer; Classification of the ECHO message; CORR = Correct; OS = Omisssive Symmetric; TS = Transmissive Symmetric; SOA = Strictly Omissive Asymmetric; SDOA = Single-Data Omissive Asymmetric; TA = Transmissive Asymmetric

A.2. Bus Monitor Data

The Bus Monitor collects data records from the RPP state monitors or the simplex-hub monitor.

A.2.1. RPP State Record

The RPP (i.e., BIU or RMU) state records are triggered by the RPP's Mode Control Unit (MCU) issuing a command or by the detection of a failure by the RPP's Status Monitoring Unit (SMU).

Table A.4: RPP state data record

Record Field	Data Type	Range	Description
Time Index	Integer	0 to 2^{31} - 1	Driven by a 3 MHz clock (333 ns period); Rolls over after reaching the maximum value (in approximately 11.9 minutes)
Sequence Number	Integer	0 to 2^{8} - 1	Incremented with every new record; Rolls over after reaching the maximum value
SHM Health Status	Enumerated	{Inactive, Recovering, Trusted, Failed}	See Figure 26 for SHM state transition graph
SHM Content Error	Boolean	{False, True}	An error indicates that the record content from the RNM does not match any of the reference records.
SHM Early Error	Boolean	{False, True}	An error indicates that the record content from the RNM matches the timing reference record but it was generated earlier than expected with respect to the previous record matching the timing reference.
SHM Late Error	Boolean	{False, True}	An error indicates that the record content from the RNM matches the timing reference record but it was generated later than expected with respect to the previous record matching the timing reference.
SHM Failure	Boolean	{False, True}	Indicates that the SHM has transitioned to the Failed state.
Monitor Id	Integer	1 to 8	Identifies the source of the record Id = 1..4 => BIU 1..4; Id = 5..8 => RMU 1..4
Message Count	Integer	0 to 2	Number of messages received from the ENM for this record
Halt Code	Integer	0 to 7	1 = Message reception timeout; 2 =Data link error; 3 =Wrong message tag; 4 = Valid frame; Others unused
ENM Message 0	ROBUS Message	(Tag, Payload) Valid Tag: SPECIAL	Payload content (Bit 1 = leftmost bit): Bit 1 = SMU_Failure; Bit 2 = SMU_No_Clique; Bit 3 = SMU_Accusation_Against_Self; Bit 4 = SMU_OK_Distrusted; Bit 5 = SMU_SK_Distrusted; Bit 6 = SMU_Timeout; Bit 7 = SMU_Protocol_Error; Bits 8-12 = SMU_Protocol_Error_Code (see Appendix B); Bits 13-16 = Communication status for ROBUS ports 1-4 (as indicated by the corresponding Manchester Waveform

Record Field	Data Type	Range	Description
			Monitors)
ENM Message 1	ROBUS Message	(Tag, Payload) Valid Tag: DATA	Payload content (Bit 1 = leftmost bit): Bit 1 = IU Zero_Schedule; Bit 2 = IU Invalid_Schedule; Bit 3 = MCU Node Kind 0 = RMU 1 = BIU Bits 4-6 = MCU Node Id; Bits 7-9 = MCU Major Mode 0 = Self Test 1 = Clique Detection 2 = Clique Join 3 = Clique Initialization 4 = Clique Preservation Others unused Bit 10 = MCU Diagnostic Cycle (meaningful only in Clique Join mode); Bits 11- 13 = MCU Minor Mode 0 = Reset 1 = Self Test 2 = Local Diagnosis and Synchronization Acquisition 3 = Initial Diagnosis and Synchronization 4 = Collective Diagnosis 5 = Schedule Update 6 = PE Broadcast/Communication 7 = Synchronization Preservation Others unused Bits 14-15 = MCU Schedule Status 0 = Valid 1 = Zero 2 = Invalid Others unused Bit 16 = MCU Output Enable
ENM Message 2	ROBUS Message	(Tag, Payload) Valid Tag: DATA	Payload content (Bit 1 = leftmost bit, OK = Opposite Kind, SK = Same Kind): Bit 1 = NDU Accusation OK node 1; Bit 2 = NDU Accusation OK node 2; Bit 3 = NDU Accusation OK node 3; Bit 4 = NDU Accusation OK node 4; Bit 5 = NDU Accusation SK node 1; Bit 6 = NDU Accusation SK node 2; Bit 7 = NDU Accusation SK node 3; Bit 8 = NDU Accusation SK node 4; Bit 9 = NDU Conviction OK node 1; Bit 10 = NDU Conviction OK node 2; Bit 11 = NDU Conviction OK node 3; Bit 12 = NDU Conviction OK node 4; Bit 13 = NDU Conviction SK node 1; Bit 14 = NDU Conviction SK node 2; Bit 15 = NDU Conviction SK node 3; Bit 16 = NDU Conviction SK node 4

A.2.2. Hub State Record

The generation of simplex hub state records are triggered by the transmission of a synchronization or mode message, by the beginning of the Schedule Update mode, the beginning of the PE Broadcast mode, or a detected failure.

Table A.4: Hub state data record

Record Field	Data Type	Range	Description
Time Index	Integer	0 to 2^{31} - 1	Driven by a 3 MHz clock (333 ns period); Rolls over after reaching the maximum value (in approximately 11.9 minutes)
Sequence Number	Integer	0 to 2^{8} - 1	Incremented with every new record; Rolls over after reaching the maximum value
SHM Health Status	Enumerated	{Inactive, Recovering, Trusted, Failed}	See Figure 26 for SHM state transition graph
SHM Content Error	Boolean	{False, True}	An error indicates that the record content from the RHM does not match any of the reference records.
SHM Early Error	Boolean	{False, True}	An error indicates that the record content from the RHM matches the timing reference record but it was generated earlier than expected with respect to the previous record matching the timing reference.
SHM Late Error	Boolean	{False, True}	An error indicates that the record content from the RHM matches the timing reference record but it was generated later than expected with respect to the previous record matching the timing reference.
SHM Failure	Boolean	{False, True}	Indicates that the SHM has transitioned to the Failed state.
EHM Message	ROBUS Message	(Tag, Payload) Valid Tag: SPECIAL	Payload content (Bit 1 = leftmost bit): Bits 1-5 = 0 each (unused); Bit 6 = Failure (from Watchdog); Bit 7 = Invalid_Schedule (from Schedule Processor); Bit 8 = Zero_Schedule (from Schedule Processor); Bits 9-12 = Next_Message (output mux control signal) 0 = Route (PE Broadcast) 1 = Vote (Schedule Update) 2 = Synchronization 3 = Mode 4 = Id 5 = Valid_Schedule (Schedule Assessment) 6 = Zero_Schedule (Schedule Assessment) 7 = Invalid_Schedule (Schedule Assessment) 8 = Diagnosis (BIU or RMU convictions) Others unused Bits 13-16 = Communication status for PE ports 1-4 (as indicated by the corresponding Manchester Waveform Monitors)

Appendix B. Protocol-Error Codes for the RPP Status Monitoring Unit

Table B.1 describes the protocol-error codes generated by the Status Monitoring Unit (SMU). Error codes not listed are unused. See [2] for a full description of the ROBUS-2 protocols.

Table B.1: SMU protocol-error codes

Error Code	Error Description
1	No-Eligible-Voters after the completion of the Frame Synchronization phase of the Synchronization Acquisition minor mode in the Clique Detection major mode. This implies that all nodes of the opposite kind are accused.
2	No-Majority result for the Accept(ECHO) function during the Synchronization Acquisition phase of the Clique Detection major mode.
3	Invalid-Eligible-Voters detected after the Accept(ECHO) function asserts its output during the Synchronization Acquisition phase of the Clique Detection major mode. This error occurs when fewer than a majority of the eligible voters latched at the beginning of the Accept(ECHO) function remain valid after the function asserts its output. This implies that there was a violation of the assumption that a majority of the eligible voters used by the Accept(ECHO) function were trustworthy.
4	No-Eligible-Voters after the Initial Diagnosis in the Clique Initialization major mode. This implies that all nodes of the opposite kind are accused.
5	No-Majority result for the Accept(ECHO) function during Initial Synchronization in the Clique Initialization major mode.
6	Invalid-Eligible-Voters detected after the Accept(ECHO) function asserts its output during Initial Synchronization in the Clique Initialization major mode. This error occurs when fewer than a majority of the eligible voters latched at the beginning of the Accept(ECHO) function remain valid after the function asserts its output. This implies that there was a violation of the assumption that a majority of the eligible voters used by the Accept(ECHO) function were trustworthy.
7	No-Eligible-Voters during process P1 of the Collective Diagnosis protocol. This implies that all nodes of the opposite kind are accused.
8	No-Eligible-Voters during process P2 of the Collective Diagnosis protocol. This implies that all nodes of the opposite kind are accused.
9	No-Eligible-Voters, No-Majority word vote result, or all defendants convicted during process P3 of the Collective Diagnosis protocol.
10	No Conviction-Against-Self during process P3 of the Collective Diagnosis protocol in Clique Detection mode or the first diagnostic cycle in the Clique Join mode.
11	Conviction-Against-Self during process P3 of the Collective Diagnosis protocol in a major mode other than Clique Detection mode or the first diagnostic cycle in the Clique Join mode.
12	No-Eligible-Voters, No-Majority word vote result, or all defendants convicted during process P4 of the Collective Diagnosis protocol.
13	Result of process P4 of the Collective Diagnosis protocol does not match the result of process P2 in a major mode other than Clique Detection mode.
14	No-Eligible-Voters during process P2 of the Schedule Update protocol. This implies that all nodes of the opposite kind are accused.
15	No-Eligible-Voters or No-Majority word vote result during process P3 (for RMUs) or P4 (for BIUs) of the Schedule Update protocol.
16	No-Eligible-Voters or No-Majority Accept(INIT) result during process P1 (for RMUs) or P2 (for BIUs) of the Synchronization Preservation protocol.
17	No-Eligible-Voters or No-Majority Accept(INIT) result during process P3 (for RMUs) or P4 (for BIUs) of the Synchronization Preservation protocol.
18	No-Eligible-Voters during process P1 of the PE Broadcast protocol. This implies that all nodes of the opposite kind are accused.
19	No-Eligible-Voters during process P2 of the PE Broadcast protocol. This implies that all nodes of the opposite kind are accused.

Error Code	Error Description
20	The result of process P2 of the PE Broadcast protocol does not match the message sent at the source BIU.

References

[1] Torres-Pomales, Wilfredo; Malekpour, Mahyar R.; Miner, Paul S.; and Koppen, Sandra V.: *Plan for the Characterization of HIRF Effects on a Fault-Tolerant Computer Communication System.* NASA-TM-2008-215306, 2008.

[2] Torres-Pomales, Wilfredo; Malekpour, Mahyar; and Miner, Paul S.: *ROBUS-2: A Fault-Tolerant Broadcast Communication System.* NASA TM-2005-213540, 2005.

[3] Torres-Pomales, Wilfredo; Malekpour, Mahyar; and Miner, Paul S.: *Design of the Protocol Processor for the ROBUS-2 Communication System.* NASA TM-2005-213934, 2005.

[4] Miner, Paul S.; Malekpour, Mahyar; and Torres, Wilfredo: *A Conceptual Design for a Reliable Optical Bus (ROBUS).* Presented at the 21st Digital Avionics Systems Conference (DASC), Irvine, California, October 27-31, 2002.

[5] Fuller, Gerald L.: *Understanding HIRF – High Intensity Radiated Fields.* Aviation Communications, Inc., Leesburg, VA, 1995, p. 7-2.

[6] Hess, Richard: *Computing Platform Architectures for Robust Operation in the Presence of Lightning and Other Electromagnetic Threats.* Presented at the 16th Digital Avionics Systems Conference (DASC), Irvine, California, October 26-30, 1997

[7] Xilinx Application Note XAPP230: *The LVDS I/O Standard.* Version 1.1, Xilinx Corporation, November 1999.

[8] Xilinx Application Note XAPP339: *Manchester Encoder-Decoder for Xilnx CPLDs.* Version 1.3, Xilinx Corporation, October 2002.

[9] MIL-STD-1553B: *Aircraft internal time division command/response multiplex data bus.* United States Department of Defense, September 1987.

[10] Xilinx Application Note XAPP077: *Metastability Considerations.* Version 1.0, Xilinx Corporation, January 1997.

[11] Agilent Application Node 1448-1: *Measuring Jitter in Digital Systems.* Agilent Technologies, Inc, June 2003.

[12] Maxim Application Note 1916: *An Introduction to Jitter in Communication Systems.* Maxim Integrated Product, Dallas Semiconductor, March 2003.

[13] Hancock, Johnnie: Jitter – Understanding it, Measuring it, Eliminating it: Part 1: Jitter Fundamentals. High Frequency Electronics, April 2004.

[14] Xilinx Application Note XAPP052: Efficient Shift Registers, LFSR Counters, and Long Pseudo-Random Sequence Generators. Version 1.0, Xilinx Corporation, July 1996.

[15] Xilinx Application Note XAPP210: *Linear Feedback Shift Registers in Virtex Devices.* Version 1.3, Xilinx Corporation, April 2007.

[16] Azadmanesh, M.H.; and Kieckhafer, R.M.: *Exploiting omissive faults in synchronous approximate agreement.* IEEE Transaction on Computers, Vol. 49, No. 10, October 2000, pp. 1031–1042.

[17] Weber, Paul J.: *Dynamic Reduction Algorithms for Fault Tolerant Convergent Voting with Hybrid Faults*. Doctoral Dissertation, Michigan Technological University, May 2006.

Acronyms

AP	Active Port
BIU	Bus Interface Unit
CCP	Controller Coordination Protocol
COTS	Commercial Off-The-Shelf
CPLD	Complex Programmable Logic Device
CW	Continuous Wave
DC	Direct Current
DClk	Destination-Process Clock
DII	Data Introduction Interval
DNO	Direct Node Observer
DP	Data Processor
DSI	Derivation Systems, Inc.
EHM	Embedded Hub Monitor
EM	Electromagnetic
EMI	Electromagnetic Interference
ENM	Embedded Node Monitor
EoC	End of Cycle
FHSTC	Fine HIRF Susceptibility Threshold Characterization
FIFO	First In First Out
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
HC	Hardware Configuration
HEC	HIRF Effects Characterization
HFA	Hub Fault Analyzer
HIRF	High Intensity Radiated Field
HSTC	HIRF Susceptibility Threshold Characterization
IdG	Id Generator
IDU	Input Diagnostics Unit
IU	Input Unit
IVHM	Integrated Vehicle Health Management
LFSR	Linear Feedback Shift Register
LT	Local Time
LUF	Lowest Usable Frequency
LVDS	Low Voltage Differential Signalling
Mbps	Mega-bits per second
MCU	Mode Control Unit
MTx	Manchester Transmitter
NDU	Node Diagnostics Unit
NFA	Node Fault Analyzer
NIST	National Institute of Standards and Technology
NOR	Node Observation Receiver
NRZ	Non-Return-To-Zero
OTH	Omissive-Transmissive Hybrid
OU	Output Unit
PE	Processing Element
PE OU	PE Output Unit
PEIU	PE Input Unit
PEXU	PE Interface Unit
ppm	Parts Per Million
RC	Reverberation Chamber
RF	Radio Frequency
RHM	Remote Hub Monitor
RM	ROBUS Message

RMS	Root Mean Square
RMU	Redundancy Management Unit
RNM	Remote Node Monitor
ROBUS	Reliable Optical Bus
RPP	ROBUS Protocol Processor
RSPP	Reconfigurable SPIDER Prototyping Platform
RTCA	Radio Technical Commission for Aeronautics
RTOS	Real-Time Operating System
RTx	Manchester Receiver
RU	Reception Unit
RVU	Route and Vote Unit
Rx	Receiver
RxClk	Receive Clock
SBIR	Small Business Innovation Research
SClk	Send-Process Clock
SDU	System Diagnostics Unit
SHM	System Health Monitor
SMU	Status Monitoring Unit
SPIDER	Scalable Processor-Independent Design for Extended Reliability
SU	Send Unit
SUT	System Under Test
TCS	Test Control System
TDMA	Time Division Multiple Access
Tx	Transmitter
TxClk	Trasmit Clock

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
01- 07 - 2008	Technical Memorandum	

4. TITLE AND SUBTITLE

Design of Test Articles and Monitoring System for the Characterization of HIRF Effects on a Fault-Tolerant Computer Communication System

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Torres-Pomales, Wilfredo; Malekpour, Mahyar R.; Miner, Paul S.; and Koppen, Sandra V.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

NASA Langley Research Center
Hampton, VA 23681-2199

8. PERFORMING ORGANIZATION REPORT NUMBER

L-19473

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSOR/MONITOR'S ACRONYM(S)

NASA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

NASA/TM-2008-215322

12. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 62
Availability: NASA CASI (301) 621-0390

13. SUPPLEMENTARY NOTES

An electronic version can be found at http://ntrs.nasa.gov

14. ABSTRACT

This report describes the design of the test articles and monitoring systems developed to characterize the response of a fault-tolerant computer communication system when stressed beyond the theoretical limits for guaranteed correct performance. A high-intensity radiated electromagnetic field (HIRF) environment was selected as the means of injecting faults, as such environments are known to have the potential to cause arbitrary and coincident common-mode fault manifestations that can overwhelm redundancy management mechanisms. The monitors generate stimuli for the systems-under-test (SUTs) and collect data in real-time on the internal state and the response at the external interfaces. A real-time health assessment capability was developed to support the automation of the test. A detailed description of the nature and structure of the collected data is included. The goal of the report is to provide insight into the design and operation of these systems, and to serve as a reference document for use in post-test analyses.

15. SUBJECT TERMS

FPGA; HIRF; Avionics; Data bus; Fault tolerance

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			STI Help Desk (email: help@sti.nasa.gov)
					19b. TELEPHONE NUMBER *(Include area code)*
U	U	U	UU	59	(301) 621-0390