4Links™

Monitoring and Analysis of SpaceWire Links

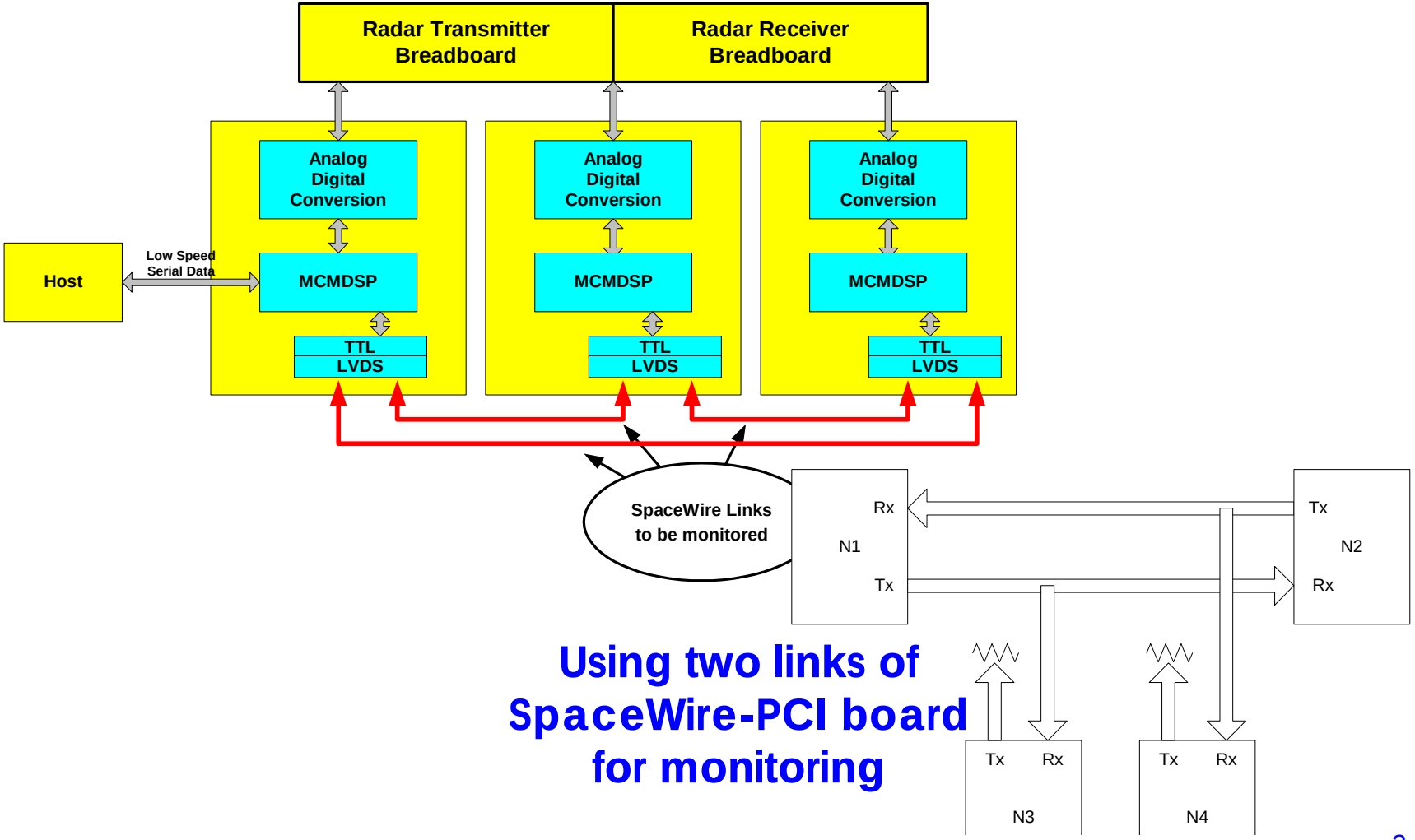
James Lux, Frank Loya
Jet Propulsion Laboratory
4800 Oak Grove Avenue,
Mailstop 161-213
Pasadena CA 91109
USA
james.p.lux@jpl.nasa.gov

Barry M Cook, Paul Walker
4Links Limited
The Mansion, Bletchley Park,
Milton Keynes,
MK3 6ZP, UK
barry@4Links.co.uk
paul@4Links.co.uk

Jet Propulsion Laboratory
California Institute of Technology

4Links™

Block Diagram of DSP Scatterometer Breadboard



Using two links of SpaceWire-PCI board for monitoring

4Links™

Introduction

A breadboard scatterometer (a type of orbiting radar used for measuring vector ocean winds) was constructed at Jet Propulsion Laboratory to investigate the feasibility of using general purpose programmable Digital Signal Processors (DSP) to replace the special purpose hardware used in the previous successful instruments[1]. The architecture used multiple processors to reach the required computational performance with the available space qualified processors (Analog Devices 21020 family). Three Astrium MCMDSP modules were used in the breadboard, each of which integrates the CPU, memory, peripheral controls, and a SMCS332 high speed interface. The latter was used to implement SpaceWire links between the processors. The breadboard successfully demonstrated a scalable multiprocessor approach for this type of instrument.

Early in the development effort it was realized that monitoring of the messages being passed over the SpaceWire links would be essential, particularly for development of the low level message passing drivers, as well as the higher level algorithms. In particular, there was a need to match individual messages with specific Radio Frequency (RF) pulses transmitted or received by the breadboard radar, requiring a real-time capability that could not be conveniently met with software based solutions in the limited resources of the DSP. The need to make real-time performance and timing measurements required that the monitoring approach be entirely passive (as opposed to a decode and retransmit approach). Slide 2 shows a block diagram of the breadboard, together with the circuit used for monitoring.

SpaceWire properties enabling passive monitoring and decoding

SpaceWire continually transmits bits at a constant rate (this is accomplished by filling in gaps with Null characters).	allows us to derive formulas that can determine when a character occurred. For example, given the character sequence $\langle C_i \rangle$, $i = 0 \cdots n$, and the time that C_0 occurred (call it T_0), then the time that C_i occurred is given by: $T_i = T_0 + \tau \sum_{j=1}^{i} \#_b(C_j)$ where τ is the bit time, and $\#_b$ is a function that yields the number of bits in C_j
If either the D or S lines on either end (receive, transmit) of the link fails to make a transition within 850 ns., the link will be broken, and both sides of the link will reset (i.e. stop transmitting for a preset time interval).	assures us that property 1 will be enforced.
The SpaceWire bit stream contains no framing or error-correcting bits	imposes a “zero tolerance” condition on bit error. Since no characters are used for framing or error correction, it is assumed that each bit will contribute either to the content or attributes of a specific character.
The SpaceWire bit stream always contains enough error detection bits to detect any single bit error.	assures us that if a single bit error does occur, it will be detected. The SpaceWire data stream always contains between 10-25% odd parity bits (as attributes of a character). When a parity error is detected link is reset. This assures us that no corrupt data will gain acceptance by the receiving node (i.e. a detected reset implies that the incoming data packet should be discarded).
The separation of the SpaceWire bit stream into characters can be done in a single pass	allows the separation of the bit stream into characters with the application of “straight forward” hardware.

4Links™

Monitoring with Standard Lab Instruments

The initial efforts to monitor the links used differential high impedance probes and a digital storage oscilloscope/logic analyzer, which was satisfactory at first (it is easy to recognize NUL and FCT tokens after a bit of practice). When the monitoring need moved to needing higher level information beyond "is the link running?", the limited capture memory, the high speed signals, and the lack of a high level interface prompted further development. One of the authors (Loya) identified 5 characteristics of the SpaceWire protocol design that made possible a simple hardware implementation using a small Finite State Machine to decode the messages. Table 1 lists the properties and their implications.

SpaceWire-PCI as a Monitor Receiver

The SpaceWire-PCI card, in a suitable realtime programming environment, can provide a significant fraction of the monitoring functionality that would be provided by the fully idealized monitor embodied in the state machine described above. This approach had the significant advantage that it required almost no hardware development and there was a suitable set of API calls available to integrate the card into a realtime computing environment.

4Links™

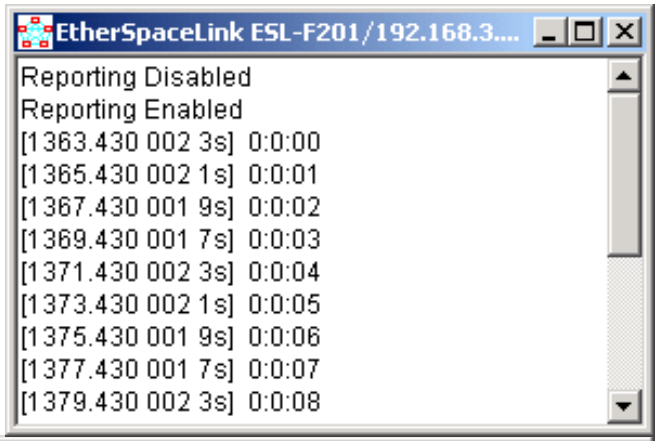
The performance limitations of the Labview/SpaceWire-PCI monitor approach easily keeps up with the usual breadboard traffic, consisting of messages every few milliseconds consisting of about a thousand bytes. Unfortunately, the Labview implementation does not allow accurate unambiguous time-stamping (with an accuracy of milliseconds), mostly because of the limitations inherent in the NT driver model.

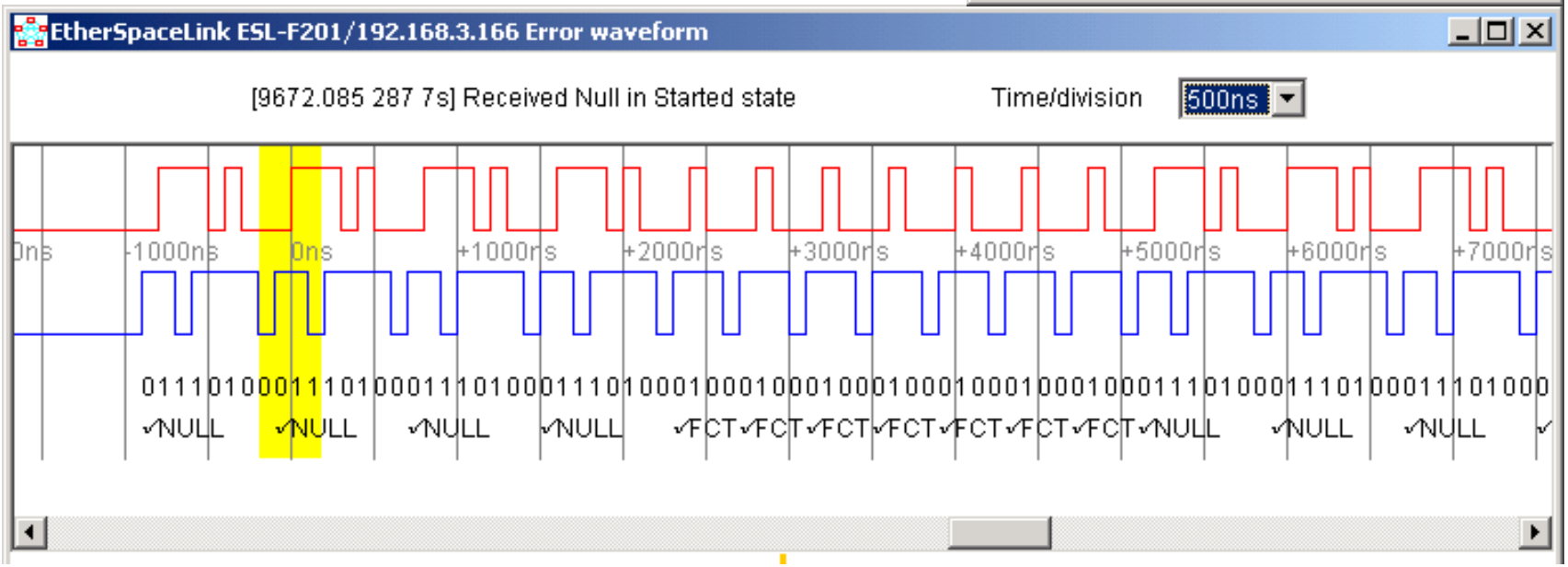
4Links have developed a series of FPGA implementations of SpaceWire, and use their latest implementation for monitoring SpaceWire links. With access to the data when it arrives, it is possible to give very precise time-of arrival information. The data log below shows time stamp reports, such as `[11798.205 633 7s]`, as shown by the 4Links EtherSpaceLink ESL-F201 with the Time-Stamp option. The stamp is a free-running 40-bit count with the resolution of 100ns.

Jet Propulsion Laboratory
California Institute of Technology

4Links™

The Time-Code report includes a time stamp of when the time code arrived, and the time code in the format specified in the SpaceWire standard, with two reserved bits and then a six-bit value. Building waveform capture into the monitoring system allows the data to be presented already decoded, and co-ordinated with a report of the triggering event. The waveform below is captured during SpaceWire initialization.





Jet Propulsion Laboratory
California Institute of Technology

4Links™

The waveform captured below is the same event as was reported on Slide 6. Note the time stamp of when the error occurred, the type of error, the state when the error occurred, the decode of individual bits, and the further decode of those bits into characters.

These reports were generated by the EtherSpaceLink ESL-F201, with options for time codes, time stamps, error reporting and error waveforms, and were displayed by the SpaceWireUI program. The EtherSpaceMon ESM-F200 will bring these capabilities to monitoring both directions of links.

EtherSpaceLink ESL-F201