Open-Source Hardware Implementation of a SpaceWire Router

Kristi Stefa, Thomas Filarski, Daniel Llamocca
kstefa@oakland.edu, tfilarski@oakland.edu, llamocca@oakland.edu
Department of Electrical and Computer Engineering
School of Engineering and Computer Science, Oakland University

Introduction

The aim of the project was to implement a SpaceWire Router, documented by the European Space Agency in 2003. The SpaceWire protocol is an effective data transport system that details high frequency data transfer from the wire to the network level, with the aim of operating effectively on spacecraft hardware. The SpaceWire Router design, described in VHDL, encompassed several protocol layers (Character, Exchange, and Network). Internal to the architecture are SpaceWire Nodes which are directed by the router and must establish a link before passing data characters. Most of the work done focused on the design of these nodes and the correct implementation of the nodal "handshake" required by the protocol. Each node contains a Transmitter that interfaces with a host, a Receiver that reads the Data and Strobe lines to interpret data characters, and a State Machine that manages the node operation and interprets signals from the network on correct state and operation.

Methodology

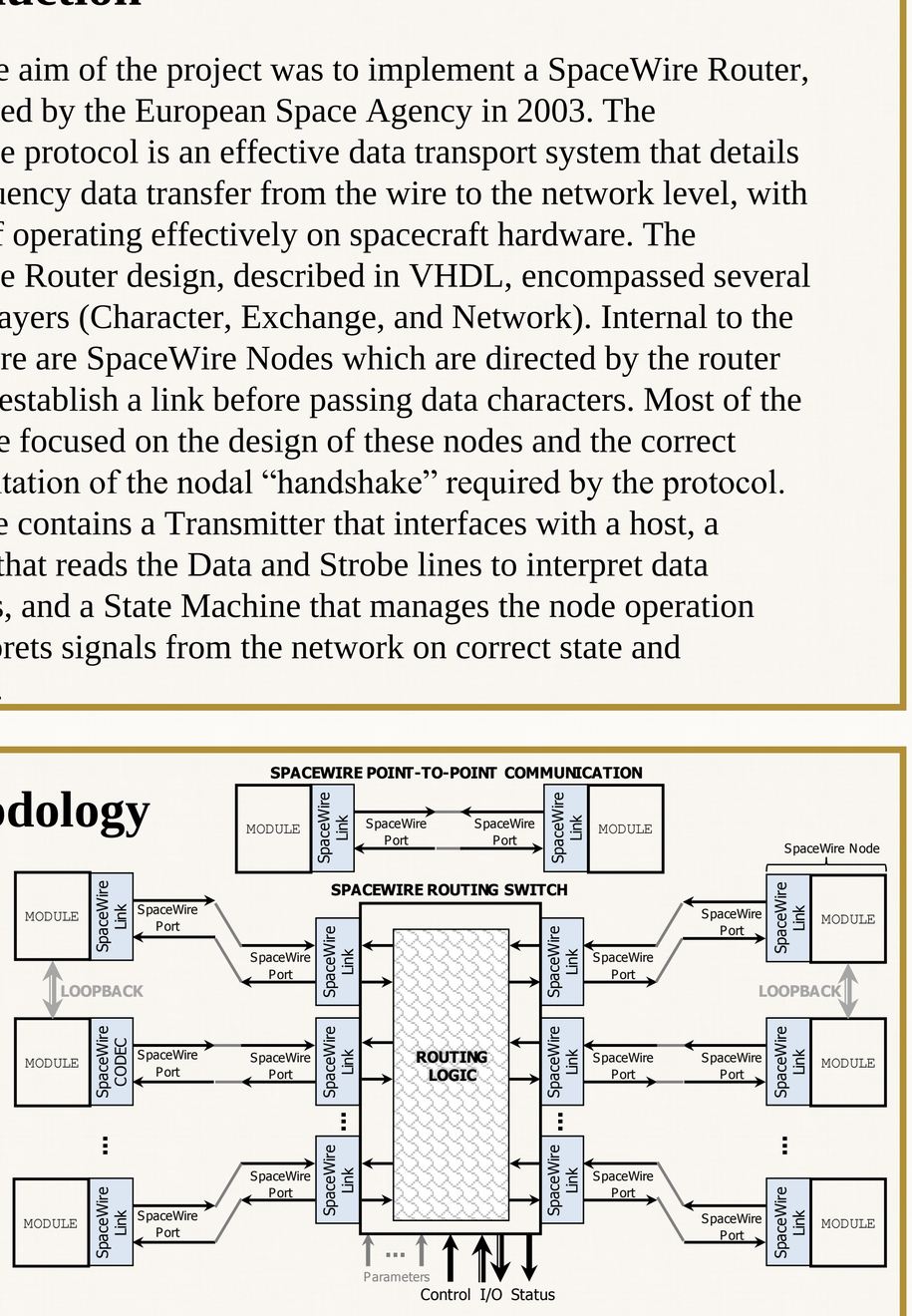

Figure 1. SpaceWire Router with nodes

Figure 1 depicts the SpaceWire Router along with several SpaceWire nodes. This was the final goal of the project and the version that was successfully assembled consisted of a Router and two node blocks with respective host stimuli blocks.

Figure 2 shows a SpaceWire Node which includes the Receiver, Transmitter, and main State Machine. The node has to interface with a host system and generate data bits or decoding incoming data bits into data characters. The two components of the node will establish a handshake protocol and begin data transfer with another node. This is how we can include the same node design into all the relevant systems that need data transfer or reception via the SpaceWire protocol.

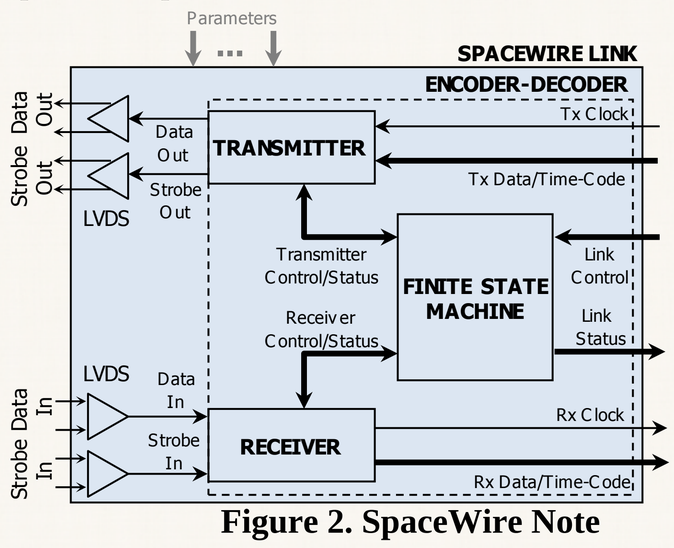

Figure 2. SpaceWire Note

By using a Timer, the main State Machine runs through a signal check before initiating node connection and ending in continuous data transfer until a relevant error occurs and transmission is aborted.

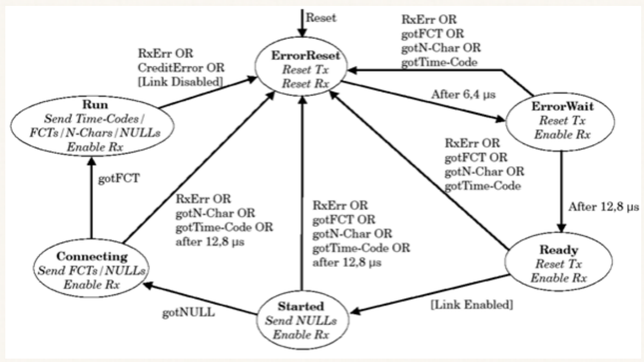

Figure 3: Operation Logic (*Source:ESA*)

Design

Within a SpaceWire node, Figure 4 depicts the design of the Receiver.

By using these components, the process of reading serial data and converting it to the correct parallel data was accomplished. The receiver is also able to communicate with the host side to transfer the parallel data to the modules or through the router.

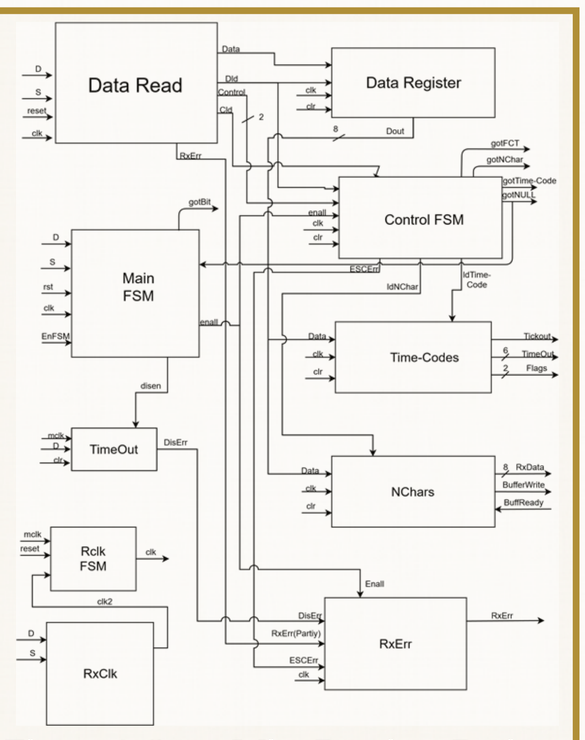

Figure 4. SpaceWire Receiver Design

Figure 5 depicts the router with an accompanying host block. The design was based around having these two communicate with each other. The host serves as the buffer for both the Transmitter and Receiver portions associated with a node. This allows for two data streams to be passed through the router to opposite node locations.

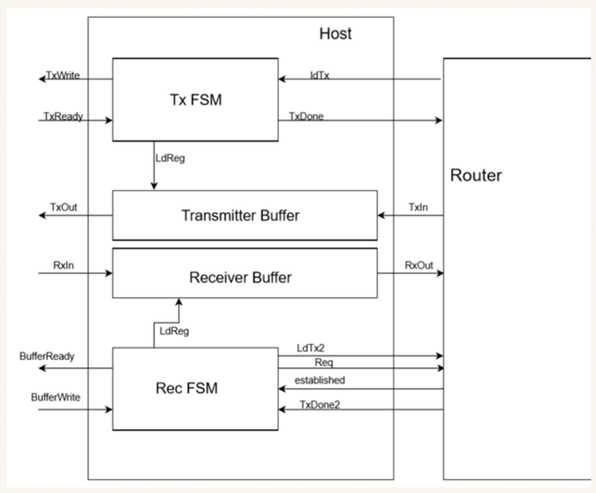

Figure 5. SpaceWire Router with a Host

Results

Simulation -Case 1: Data with Even Number of 1s
The initial link is successful and data flows until a TimeCode is required. This flips the parity of the data stream for a cycle and if captured by the Receiver, a parity error is tripped and communication ends.

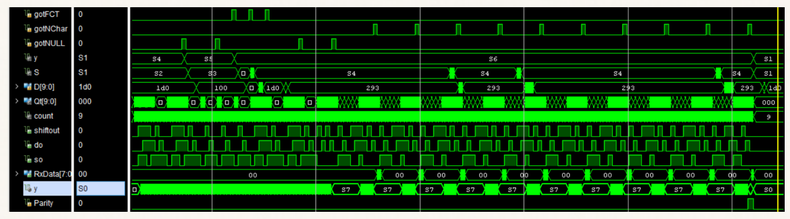

Figure 6. Simulation (even # of 1s). Error Occurrence

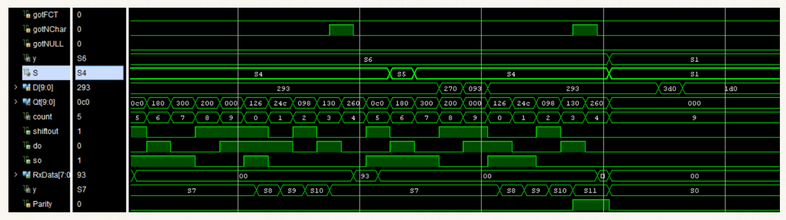

Figure 7. Simulation (even # of 1s). Closer look at Error Occurrence

Simulation - Case 1: Data with Odd Number of 1s
The first NULL to start the link is successful but the system needs to immediately grab the next character before the parity changes which it only has one cycle to do. Another parity error thrown by the Receiver.

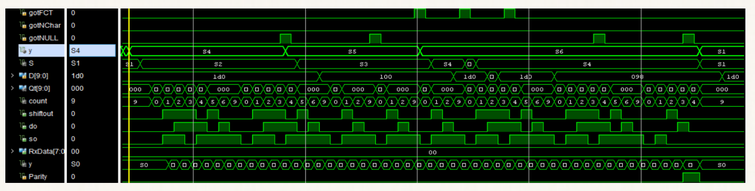

Figure 8. Simulation (odd # of 1s). Error Occurrence

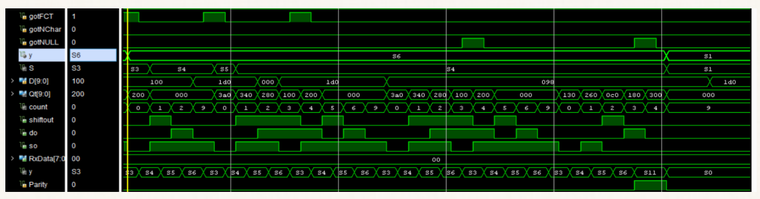

Figure 9. Simulation (odd # of 1s). Closer look at Error Occurrence

Conclusion

A SpaceWire Link and and a Router were designed in dedicated hardware and verified on an FPGA (Field Programmable Gate Array). The design can be incorporated into other systems that employ the SpaceWire protocol. With a complete Exchange to Physical implementation, it would prove effective in the spacecraft applications it was intended for.

This work was sponsored the Michigan Space Grant Consortium Research Seed Grant 2017.