NASA/BAE SYSTEMS SpaceWire Efforts
International SpaceWire Seminar (ISWS 2003)
4-5 November 2003, ESTEC Noordwijk, The Netherlands

Glenn Parker Rakow [1], Richard G. Schnurr [2], Paul Kapcio [3]

[1] *NASA Goddard Space Flight Center*
NASA GSFC, Code 561
Greenbelt, MD 20771 USA
Glenn.P.Rakow@nasa.gov

[2] *NASA Goddard Space Flight Center*
NASA GSFC, Code 560
Greenbelt, MD 20771 USA
Richard.G.Schnurr@nasa.gov

[3] *BAE SYSTEMS*
BAE SYSTEMS 9300 Wellington Road
Manassas, VA 20110 USA
Paul.Kapcio@baesystems.com

ABSTRACT

This paper discusses the state of the NASA and BAE SYSTEMS developments using SpaceWire. NASA has developed intellectual property that implements SpaceWire in Register Transfer Level VHDL for a SpaceWire link and router. This design has been extensively verified using directed tests from the SpaceWire Standard and design specification, as well as being randomly tested to flush out hard to find bugs in the code. The high level features of the design will be discussed, including the support for multiple time code masters, which will be useful for the James Webb Space Telescope electrical architecture. This design is now ready to be targeted to FPGA's and ASICs. Target utilization and performance information will be presented for some spaceflight qualified FPGA's and a discussion of the ASIC implementations will be addressed. In particular, the BAE SYSTEMS ASIC will be highlighted which will be implemented in their .25µm rad-hard line. The chip will implement a 4-port router with the ability to tie chips together to make larger routers without external glue logic. This part will have integrated LVDS drivers/receivers, include a PLL and include skew control logic. It will be targeted to run at greater than 300 MHz and include the implementation for the proposed SpaceWire transport layer. The need to provide a reliable transport mechanism for SpaceWire has been identified by both NASA and ESA, who are attempting to define a transport layer standard that utilizes a low overhead, low latency connection oriented approach. The Transport layer needs to be implemented in hardware to prevent bottlenecks.

INTRODUCTION

The need for a suitable high-speed standard for on-board satellite applications has been a sizable one for electrical system engineers for some time. System engineers have looked at commercial standards to fill this need with little success. Custom interfaces are too expensive for programs and offer little return on their investment. SpaceWire has started to fill this void in the satellite community, but still has some hurtles to climb in terms of validation & test support equipment, redundancy issues and reliable transport. The in-roads SpaceWire has started to make in the NASA community continue to build support for the standard and will help solve some of the issues.

BACKGROUND

NASA began investigating IEEE-1355-1995 as a possible high-speed standard back in the fall of 1998 during the phase A study for the Gamma-Ray Large Area Telescope (GLAST). At that time, it looked attractive except that the standard was not well defined and there were not many designs that could be purchased.

As a result, NASA began an effort to further define IEEE-1355-1995 and design a link. The design was coded in Register Transfer Language (RTL) VHDL for portability. About the same time the early Draft versions of the SpaceWire standard were being released. Many of the assumptions between the NASA design and the SpaceWire standard, including the use of LVDS, were the same. The major difference was the initialization state machine, which was changed to match the SpaceWire standard. This first design was selected by the NASA Swift mission to be used as a high-speed interface for the Burst Alert Telescope (BAT).

The router design was completed for the James Webb Space Telescope (JWST). The JWST implementation required the SpaceWire routing function for the baseline architecture. The router design used a modified version of the Swift link design. The Swift link design, which had one local clock domain, was broken up into two local clock domains, one for the serialization clock and the another slower clock used for core logic functions. This partition reduced the amount of high frequency clocking making the design easier to route and lower in power.

There are currently two link designs and one router design supported by NASA, although there may be many permutations of the router design, i.e., number of router ports, number of external interfaces, etc. Two link designs exist for size reasons. The modified Swift link design called the "Deluxe" version, was too large to fit into an Actel 54SX32 so it was modified for this purpose. This smaller design called the "Skinny" version, has one local clock domain and as a result it is not as fast as the "Deluxe" version.

All designs have been extensively verified. There are at least five different institutions that are being supported for use of the NASA core on the JWST. The router core is currently being targeted to two different .25µm ASICs, one for a technology development (BAE SYSTEMS) and another for JWST. Other NASA projects are using the NASA SpaceWire core including the National Oceanic and Atmospheric Administration (NOAA) Geostationary Operational Environmental Satellite (GOES) - R. Many other projects, some non-NASA have expressed interest. This is a testimony to the hard work and dedication that went into the development of the SpaceWire standard.

VERIFICATION

Much of the success of the NASA development may be attributed to the verification effort, which made the design a reliable product. The verification environment developed by Omar Haddad from QSS Inc.was is built around a C to VHDL gasket that allowed tests to be written in C and communicate to the VHDL where the Bus Functional Models (BFMs), written in behavioural VHDL and Unit Under Test (UUT), written in RTL VHDL reside (see Figure 1). The C portion generates and receives packets and compares them to the packet scoreboard.

Two types of tests are performed, directed tests and random tests. The directed tests are designed to verify a certain portion of the protocol as defined in the standard and the NASA Intellectual Property (IP) core specification. Random tests are use to ensure that no errors occur when a valid combination of test parameters is randomized. Random seeds used to randomize the test parameters are generated from the computer time. Random tests have been the most important aspect of the testing performed. They have uncovered many subtle errors that could not have been anticipated. Every time that the RTL design is modified or fixed a set of regression tests is performed. The set of regression tests comprises all directed tests and all of the random seeds that exposed previous bugs. The regression tests are executed in batch mode by using a perl script. The perl script compiles source files if necessary and calls the simulator. Individual tests may be run in an interactive mode that displays the waveforms. The verification environment consists of a freeware program called Cywin, which creates an Unix-like environment for Windows including a Perl interpreter and modelsim VHDL simulation environment.

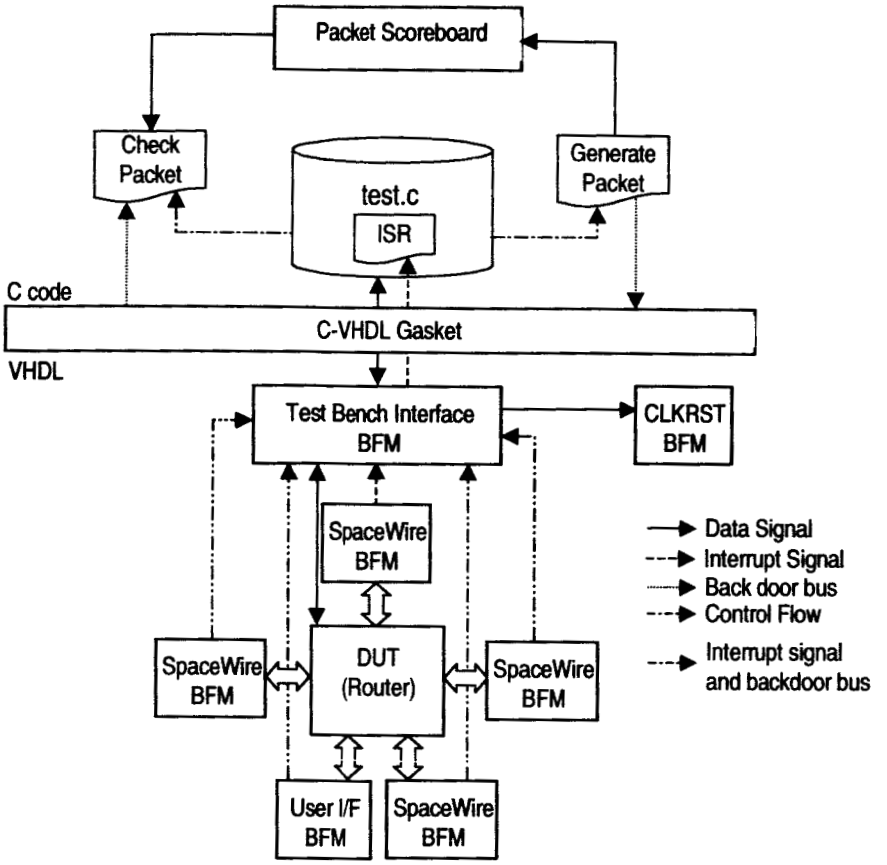

Fig. 1. SpaceWire Router Verification Environment

DESIGN

The link and router designs are fully compliant with the final version of the SpaceWire standard (ECSS-E-50-12A), yet have unique interfaces, features and extensions inherent in any implementation.

Interfaces

Unlike the example in the standard, which shows a 9-bit FIFO, the NASA core uses a 10-bit FIFO with the extra two bits encoding the EOP flag and the EOP type. The FIFO also uses 2 clocks to relieve the synchronization issues between the interface. Lastly, the FIFO has hand-shaking signals between the user and the FIFO to simplify the state machine communicating with the FIFO so that it can burst data easily. This also relieves the design from technology induced timing constraints. These features make the interface logic to the NASA Core flexible and simple.

The NASA core also has an external clock divide value that is used to program the 10 Mbps data rate independent of the external clock.

Time-Code

The router core expanded the SpaceWire Time-Code to also allow multiple time masters on the network, although the design may be set in a mode that makes it compatible with the standard, allowing only one time master. This addition to the standard was added for the JWST mission that will have two time masters, the spacecraft Command Telemetry Processor (CTP) and the Instrument Command Data Handling (ICDH). In order for the two time masters not to interfere with each other a mode had to be added to break the standard. In this mode the Time-Code control bits are used to identify one of four different Time-Codes. The design keeps four separate time-code counters corresponding to the time-code control bits. There are also configuration register bits that either enable or disable the transmission or the receipt of a particular Time-Code on a per port basis. This control allows the design to communicate with designs not supporting multiple time-codes.

Configuration 0 Format

Since the standard does not describe the exact format of the configuration 0 packet to program the router, our implementation is described. The configuration 0 space is memory mapped to a 16 bit address space (64K) of which only a small portion is used (less than 400) with the exact number dependent upon the size of the router. The memory width is 16 bits per address location. There are three different configuration 0 packet formats; Write Command Packet, Read Command Packet, Response Packet.

The Write Command Packet has the first byte zero as defined in the standard for addressing configuration 0 space. The second byte is the Router ID, which is a hardwired 8-bit value used to uniquely identify the router from all others on the switched fabric. This value must match the targeted router or the command will be ignored. The third byte is the Packet Type, which has currently 4 defined types: write, read-no-clear, read-clear, read-response. For this packet write is selected. The next two bytes are the Address Least Significant Byte (LSB), followed by the Address Most Significant Byte (MSB). This address corresponds to the location of the data to be written in the configuration space, i.e., routing table, port configuration, etc. The Data LSB and then the Data MSB follow the address bytes. The last byte of the Write Command Packet is the checksum packet, which is an arithmetic sum of the Destination Address (always 0 for configuration space), Router ID, Packet Type, Address LSB, Address MSB, Data LSB and Data MSB. If the checksum value does not match or if the packet has less than eight bytes or the checksum is not the last byte, than the command will be ignored. Note: other packet formats may have more than eight bytes.

The Read Command Packet has a similar format with the Write Command Packet except for several differences. The Packet Type is either set to Read-No-Clear or Read-Clear. The difference being the Read-Clear will clear the status value being read if applicable and the Read-No-Clear will not clear the status value. Also the 2 Data bytes are missing, and the next byte after the Address MSB is the Count byte, which indicates the number of 16-bit address locations to read. Note: the Count byte must be greater than zero or the command will be considered invalid and ignored. The final field after the Count byte is a number of Return Address bytes. This must be at least one byte and may be more depending upon the switched fabric topology to return the Response Packet to the proper location and the addressing scheme used. Finally, the last byte is the checksum, which covers all the described bytes beginning with the Destination byte equal to zero. If the checksum does not match the command will be ignored.

The response to the Read Command packet is the Response Packet. This packet unlike the previous two does not have a Destination Address equal to zero but one or more physical or logical address corresponding to the Return Address field in the Read Command Packet. These bytes are followed by the Router ID, Packet Type, which is set to Read Response, the starting Address LSB, the starting Address MSB followed by one or more Data byte pairs (16-bits) corresponding to the Count byte value in the Read Command Packet. The last byte is the Checksum byte, which is calculated starting with the Router ID through to the lass Data byte Pair.

Network discovery may be performed by a special read in the Configuration 0 memory map to the Router ID Register. This particular register does not require a valid Router ID field to access. The Router ID register holds the unique Router Identification number, and the command port physical address. A read command to this register does not require a valid RouterID field and the first byte of the Return Address Header will be overwritten with the hardware address of the incoming Read Command Packet. Note that although the first header byte will be overwritten, it must not be address zero. By using this special Read Command Packet the Router ID and the port it is attached to may be determined

Routing features

There are several features that have been added to the router to make it more flexible or easier to test. One feature that will make the NASA router core easier to use is the addition of a "By-pass" mode. This mode is on a per port basis. When this mode is enabled for a particular port then all the packets coming into that port will be routed to the desired external interface (if more than one). This makes the port effectively a point-to-point link. This will be very useful when the router is connected to another SpaceWire device that only transmits raw data and can not program the header information.

Another feature that will make the router easier to test is a "Loop-back" mode that when enabled will allow the port to connect back to itself. This will allow the router to be able to route packets within itself without having to connect external cables to the ports.

Redundancy

In order for SpaceWire to be more than just a high-speed science bus there are several issues that will have to be resolved. One of them is the need for reliable transport layer (discussed in next section) and the other is a redundancy scheme that will meet the needs of the types of systems that SpaceWire designed to support. This latter concern may be handled at different points depending upon the level of protection required. It may be handled with redundant physical layers or with redundant SpaceWire Cores and a transport layer controlling which core will communicate. The latter case is more robust but often it is desirable for instruments and other non-critical applications to use redundancy at the physical layer, which is easier to implement and requires less resources (area, power, testing, etc.).

For JWST a method has been developed to perform redundancy at the physical layer. Chris Dailey from QSS Inc., proposed this method. The concept is based upon a master-slave approach. One side of the link will be the master to determine whether the primary or redundant side of the physical layer is active. The other side of the link is the slave and will search for which of the two physical layer sides is active and follow. The slave determines if a side of the physical layer is active based upon how long it takes for it to get into the Run State. If it does not reach the Run State after a certain amount of time it will listen to the other side of the link for that amount of time. It will oscillate back and forth until the Run State is reached.

TRANSPORT LAYER

NASA has been working since Spring of 2002 to define a transport layer for the SpaceWire standard. It grew out of a requirement for JWST that was later dropped. The NASA team has consulted with Steve Parkes so as to have a basic agreement on some of the tenets for the layer with the hope that a standard may be developed. The NASA team has pushed forward with an implementation of the Transport Layer for the BAE SYSTEMS ASIC being developed under NASA technology funds.

The Transport Layer is implemented at the source and destination of a packet, not at the routers along the path. The Transport Layer is backward compatible and will allow raw SpaceWire packets to pass. Packets that use the Transport Layer are sent over channels that are established by the Transport Layer. There are two types of packets that may use the Transport Layer, reliable packets and un-reliable packets. Each reliable packet is acknowledged. Unreliable packets are not acknowledged but have sequence number information so that the receiver can determine if a packet is missing or out of order. The Transport Layer protocol allows transmitting multiple packets within a sliding window while the acknowledgment is outstanding. When the window limit is reached, another packet may not be transmitted until the first packet in the window is acknowledged. The window is moved according to the number of acknowledgements received starting from the first packet of the window and the number of contiguous adjacent acknowledgments from the first. This provides more efficient use of the link over stop-and-wait approach. Retransmission is performed by a time-out mechanism at the transmitter if an acknowledgement is not received in the allowed time. Duplicate packets are identified as having a repeated sequence number.

In order to open a channel and move Transport Layer packets a special Configuration Packet is sent. This packet has all the information to define the channel parameters. A channel is for unidirectional data transfer. In order for two nodes to send data back and forth two channels are required. This simplifies the protocol for establishing a channel The channel is opened by the transmit side.

This layer defines a packet structure that is within the SpaceWire packet definition. There are three different packet structures defined for the Transport Layer, Data Packet, Configuration Packet and Acknowledgement Packet. All three packets have the same first four bytes. The first byte is the Destination Byte per the SpaceWire standard. The second byte is the Source byte of the node that sent the packet. This if followed by the Packet Control byte which has three fields. The first field of the Packet Control byte is a 2 bits that is called Type. Type identifies the packet as a transport packet or another undefined packet type. The second field is called the mode field and identifies the packet as an acknowledgement packet or not. It also identifies it as the leading segment of a message or last segment of a message if it is not an acknowledgement packet. Lastly, it flags a transport error in all packet formats. The last field for the Packet Control byte is the sequence number. This is used for reliable packets to correlate the acknowledgement with the data packet. It is also used for both reliable and unreliable for segmentation. The fourth byte is the Channel ID. This byte is the channel ID number for the Data packet and Acknowledgement packet but always zero for the Configuration

packet. Following the Channel ID byte is the payload, which is different for each packet format. The last byte of all Transport Layer packets is an eight-bit Cyclic Redundancy Check (CRC).

The details of the implementation of the Transport Layer are being completed through the architecture of the BAE SYSTEMS ASIC, which will implement most of the Transport Layer is a micro-controller. However, some aspects such as the CRC checking will be done in hardware for speed purposes.

TARGET INFORMATION

The following is targeting information for some flight worthy Field Programmable Gate Arrays (FPGAs):

Actel 54SX32 – std	Actel 54SX72 - std	Actel AX1000 – std	Actel AX1000 - std
Link "Skinny" version	Link "Deluxe" version	Link "Deluxe" version	Router 4 port 1 External IF
49 MHz	100 MHz	127 MHz	73 MHz
C cells - 59%	C cells - 33%	C cells - 9%	C cells – 58%
S cells - 93%	S cells - 68%	R cells – 16%	S cells – 95%
Total Cells - 73%	Total cells - 45%	Total cells – 12 %	RAM/FIFO – 11 out of 36
			Total cells – 73%

Note: These are initial routes and there have been no attempts to optimize them at this point.

For the JWST ICDH an ASIC will be built for a SpaceWire router. It will have twelve ports with four external ports multiplexed onto the same interface. At the time this paper was written, the final requirements were being resolved, but it is expected to have the redundancy scheme previously mentioned with external LVDS drivers and receivers. It may have another external port that will be used to connect to a FPGA that will implement 2:1 lossless data compression. This function was at one time considered to be part of the ASIC. The BAE ASIC was considered an alternative for this application but because it is being developed in parallel and for a different customer, it was decided to have independence. It is expected that this will be a 0.25µm ASIC and the vendor has not been selected at the time this paper was authored.

BAE SYSTEMS ASIC

BAE SYSTEMS was selected for a NASA research award in the Space Communications Project (SCP). The contract is to design and build a SpaceWire 0.25µm ASIC in their rad-hard technology. The effort is funded through NASA's Glenn Research Center. The proto type part will be available late in 2004. The BAE SYSTEMS point-of-contact for the effort is Paul Kapcio, a co-author of this paper.

Goddard is the co-investigator on the proposal and is working with BAE SYSTEMS to supply the SpaceWire router and assist in the architecture of the chip. The participants are the authors of this paper along with Christopher Dailey from QSS Inc., a contractor to NASA Goddard Space Flight Center. The chip will implement a four-port router with two external (parallel) interfaces. One of the two external interfaces may be used to attach another chip to expand the number of ports on the router without the need for external logic. It will also implement the Transport Layer previously described through a combination of hardware and software. A number of hardware IP cores will be part of the chip design including a micro-controller and multiple DMA engines to handle much of the Transport Layer protocol. Since the Transport Layer standard is still evolving, a software-based solution was selected to keep the design flexible. The part is expected to run at least 300 Mbps per link with integrated Low Voltage Differential Signalling (LVDS) drivers and receivers. The LVDS cores will be cold spareable and Electro Static Discharge (ESD) protected. The receivers will also have skew control to realign the Data Strobe (DS) signals. Figure 2 shows the high level view and Figure 3 shows an internal view BAE SYSTEMS ASIC.

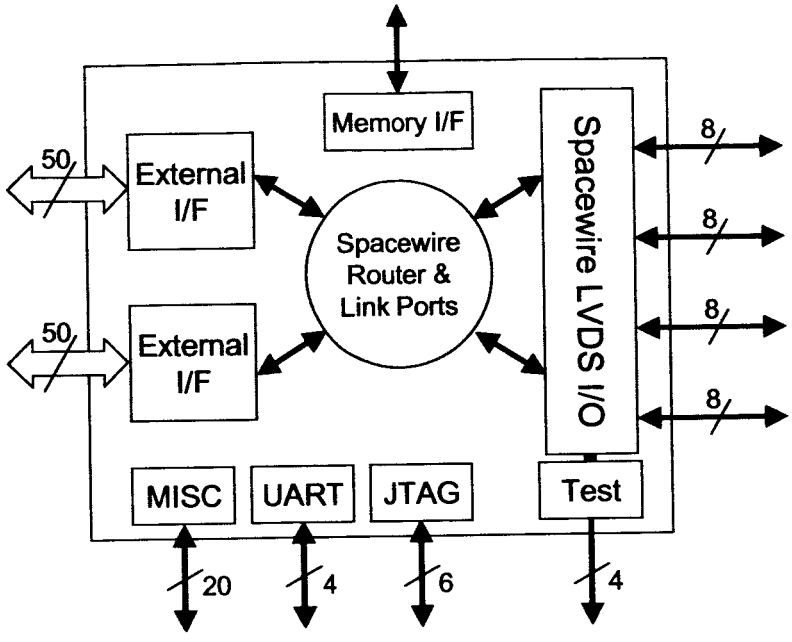

Fig. 2. High Level View BAE SYSTEMS ASIC

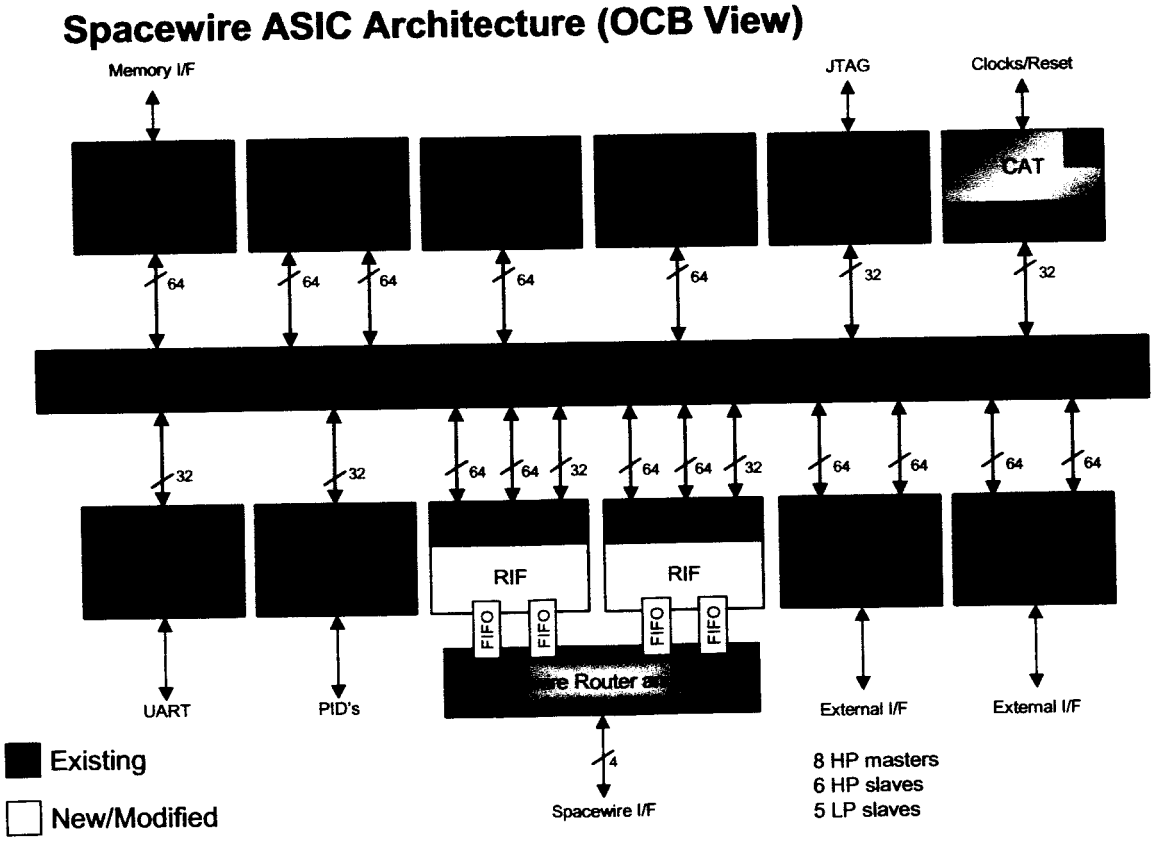

Fig. 3. BAE SYSTEMS Architecture

AVAILABILTY

Due to International Traffic in Arms Regulations (ITAR), the NASA SpaceWire cores are only available to United States (US) entities working on US government contracts or US companies interested in commercialization. A Software Usage Agreement (SUA) is required for release of the design and commercialization licenses are currently being explored for several companies.

FUTURE

SpaceWire has now been accepted in NASA as a potential high-speed data bus. It's use on Swift paved the way for its use on JWST. The next NOAA GOES satellite has now adopted SpaceWire as a result of the JWST efforts. These successes will pave the way for other programs, large and small.

Currently most satellite architectures use a high-speed bus for science and the Mil-Std-1553 bus for commands. Hopefully, the Transport Layer and a redundancy scheme will allow SpaceWire to eliminate the need for two buses, creating a very attractive standard for space flight applications.