Implementation of Infiniband for 1x Trancevier

R.Vijaya Ragavan
M.Tech VLSI Design Scholar
Department of ECE
Kalasalingam University

N.Sivasankari
Assistant Professor II
Department of ECE
Kalasalingam University

ABSTRACT

Infiniband is a new system interconnection protocol that provides high bandwidth, expandability, and scalability. This paper presents the Implementation of the Link layer of Infiniband for 1x Transmitter and Receiver. The Link Layer provides a number of services for the upper layers including data verification and error detection, flow control and buffering among others. In the InfiniBand HCA (Host Channel Adapter). Six VLs(Virtual Lanes),three for transmit and three for receive, are used to communicate data between the Transport Layer and the Link Layer. This paper describes the distribution and Translation of the Link Layer packet byte stream to the physical lanes.

General Terms

Link layer design, data formate, crccalculation,

Keywords

Infiniband specification,packet receiver machine.

1. INTRODUCTION

Infiniband is a new system interconnection technology that improves the interconnectivity between servers and I/O devices. As any fast Link technology, IB is attractive in the framework of High Energy Physics(HEP) to set-up the fabric for the data acquisition. Infiniband architecture is a well-defined technology with a specification initially announced by the Infiniband Trade Association (IBTA). Since its a rich and complex technology designed to overcome existing barriers and , at the same time ,to provide the framework for new and enhanced features and capabilities. Infiniband is being widely accepted as a future system interconnect technology to serve as the universal enterprise fabric for cluster, network and storage traffic classes. The design and implementation of the link layer of infiniband for 1x trancevier. The Link layer is responsible for sending and receiving data across the fabric at the packet – level.it also handles flow control(transmit/receive buffer management) at the physical link-level between the transmitter on one end and the receiver on the other end of the physical link. Additional Link layer's functions include buffering, error detection, packet routing within the local subnet, and address decode For the efficient implementation of the link layer this paper deals with practical issues such as resource management, flow control, buffer architecture, high speed packet processing. In this a new FIFO architecture is introduced this results in reduction of memory size and controller complexity. Additionaly the New data format introduced for time by the is significantly reduced

2. ARCHITECTURE FOR THE INFNIBANDLINKLAYER

Fig. I shows the proposed hardware architecture for the InfiniBand Link layer. The Link layer consists of three main blocks, the transmit block, the receive block, and the central control block. , the link layer is responsible for providing error detection, flow control, buffering, and other services to the upper layers. This section describes how these services are designed and implemented. A top-level design of the link layer is shown in **Error! Reference source not found.**1. There are main components of the link layer Link State Machine, Packet Receiver State Machine, Data Packet Check Machine, Link Packet Check Machine, Virtual Lanes, Virtual Lane Selector, Link P The link state machine is used to control which state the link layer is in. In order for either link packets or data packets to be processed, the link state machine must be initialized and pass all checks (enables, triggers, etc) before processing a packet.the state transitions and what functionality is enabled when the link layer is brought to each state. For a complete description of the link state machine packet Generator Packet Priority Selector.

2.1 LINK STATE MACHINE INPUTS

Reset – When asserted, link layer will return to initialize state. RemoteInit – A link packet with the flow control initialize OP code has been received and has passed the checks of the link packet check machine. CPortState – A value that indicates commands from management to change the port state. There are three valid states. Down – Connection between physical layer and link layer current inactive. Arm – Connection between physical layer and link layer has been initialized. Active – Ongoing connection between physical layer and link layer. ActiveEnable – Prevents Active state of CPortState from occurring too quickly. ActiveEnable will be asserted when a link packet with a normal OP code is received. ActiveTrigger – Allows the transition between Arm and Active states. ActiveTrigger only deals with non- VL15 packets that pass the VCRC check.LinkDownTimeout – A timeout that indicates that the Link State Machine has been down for a period of time that causes the link to transition to the LinkDownState.PhyLink – When asserted, notified link layer that the physical layer is active. When not asserted, link layer stays in initialstate.

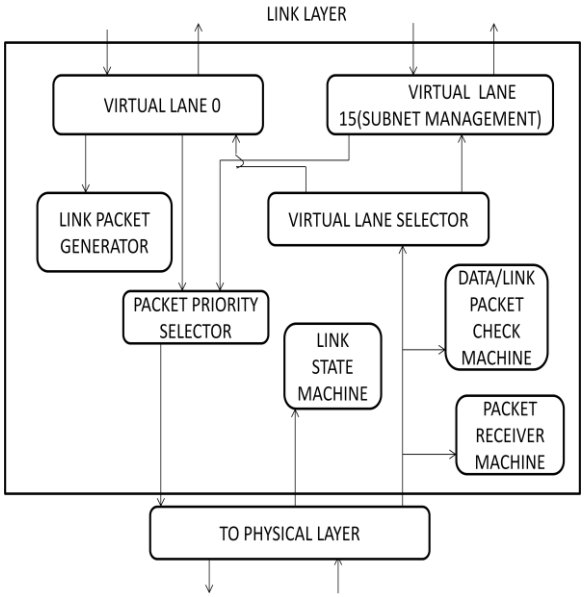


Fig-1 Architecture of linklayer

2.2 LINK STATE MACHINE OUTPUTS

DataPktXmitEnable - Indicates the link layer's action with respect to transmission of non Subnet Management Packets (SMP) data packets. When true, transmission of non-SMP data packets is enabled. When False, non-SMP data packets submitted to link layer for transmission are discarded. DataPktRcvEnable - Indicates the link layer's action with respect to reception of non-SMP data packets from the physical layer. When True, reception of non-SMP data packets is enabled. When false, non-SMP data packets received from the physical layer are discarded. SMP Enable - Indicates the link layer's action with respect to transmission and reception of subnet management packets. When true, transmission and reception of SMPs are enabled. When false, SMPs submitted to link layer for transmission or reception is discarded. LinkPktEnable - Indicates the link layer's action with respect to transmission and reception of link packets. When true, transmission and reception of link packets are enabled. When false, link packets are not generated by the link layer and any link packets received are discarded. PortState - The value of the PortState component of the PortInfo attribute. Valid values are "Down", "Initialize", "Arm" and "Active".

2.3 PACKET RECEIVER STATE MACHINE

The packet receiver state machine determines the state of the incoming data stream from the physical layer. The module determines the type of packet incoming so that the packet check machines and other modules can respond to the packet type. If an error occurs in the data stream or a stream is marked bad by the physical layer, an error is indicated and the incoming packet is dropped. For a complete description of the packet receiver state machine.

2.4 PACKET RECEIVER STATE MACHINE INPUTS

Reset – An internal signal to reset the interface, PhyLink – The physical link status from the physical layer, RcvStream – Input from physical layer with packet type information, PortState – Value of PortState component of PortInfo Down – Indicates to the link layer that the physical layer is down Initialize – Indicates there are packets to be received and initializes all starting values Arm – Responsible for receiving packet info Active – Responsible for processing packet info

2.5 DATA PACKET CHECK MACHINE

The data packet check machine performs checks on incoming data packets. The first check is the variant and invariant CRC to determine if the link packet maintained data integrity during transmission. The next check determines if a packet marked for virtual lane 15 is in fact a subnet management packet. The length check determines if the packet length matches the length field in the header. Finally, the destination local identifier check determines if the packet was destined for the current port. If any errors are indicated by the module, the system will drop the packet. For a complete description of the data packet check machine.

2.6 LINK PACKET CHECK MACHINE

The link packet check machine performs checks on incoming link packets. The first check is the CRC to determine if the link packet maintained data integrity during transmission. The next check is the length check to see if the packet is six bytes long. Finally, the virtual lane check determines if the destination virtual lane of the packet exists on the port. If any errors are indicated by the module, the system will drop the packet. For a complete description of the link packet check machine.

2.7 VIRTUAL LANES

A virtual lane (VL) provides buffering for the link layer. There can be up to 16 virtual lanes in an InfiniBand port, 15 data virtual lanes numbered 0-14 and one reserved for subnet management packets numbered 15. There must be a minimum of one data virtual lane and VL15. A virtual lane consists of a circular buffer with three indexes; read, write, and temp write. The temp write index indicates where bytes from the incoming data stream are written to. The write index is set to the temp write index once the incoming packet is confirmed valid by the data packet check machine. If the packet has an error then the temp write index is set to the write index, dropping the packet. When a virtual lane receives a link packet, it is stored on a separate buffer until the packet is validated and the packet can be consumed by the virtual lane. Flow Control is implemented in the link layer to ensure that no packets need to be retransmitted due to buffer overflow on the receiving end. VL15 is not subject to flow control. Each virtual lane keeps track of the Adjusted Blocks Received (ABR) and Flow Control Total Blocks Sent (FCTBS). FCTBS is the number of blocks (64 bytes) sent by the virtual lane since link initialization. ABR is set to the value of FCTBS for each link packet received by the virtual lane, and then incremented by number of blocks received for each data packet received by the virtual lane. When a link packet is transmitted, the Flow Control Credit Limit (FCCL) is calculated by taking the ABR plus the number of empty blocks left on the virtual lane up to 2048. Let CR represent the total blocks sent since link initialization plus the number of blocks in the data packet to be transmitted. Let CL represent the last FCCL received in a link packet. If (CL-CR) modulo 4096 <= 2048, then the data packet may be transmitted by the virtual lane. If not, the packet must wait until the condition becomes true. For a complete description of flow control

2.8 LINK PACKET GENERATOR

The link packet generator creates a flow control packet to be sent to the device on the other end of the link. Each data

virtual lane has a link packet generator. A link packet is generated whenever indicated to by the associated virtual lane, which is at least every 65,536 clock cycles. The generator receives the FCCL and FCTBS fields from the associated virtual lane along with the VL number. The link packet is then appended with a 16 bit CRC. Once a packet is generated the module indicates to the packet priority selector that a packet is ready to be transmitted. Link packets are not generated if not enabled by the link state machine.

2.9 PACKET PRIORITY SELECTOR

The packet priority selector feeds transmitting packets to the physical layer from multiple sources. If no sources have data to transmit the module remains idle. If one or more sources have data to transmit then the packet priority selector first selects which source to service. Link packets have first priority followed by subnet management packets and finally data packets. If more than one virtual lane exists in the architecture, then they are serviced in a round robin manner. Once a source is selector the module sends a start packet delimiter to the physical layer. The module then sends the source data to the physical

3. EFFICIENT PACKET BUFFERING ARCHITECTURE AND A FIFO CIRCUIT

An InfiniBand packet consists of three fields, which are the header field, the data field and the error control field. The InfiniBand standard specification states that data can be stored only after no errors are detected in checking the error control field. Although the data field is received ahead of the error control field, the data field cannot be buffered into the corresponding FIFO until all data of the error control field is received and checked [SI. To meet the InfiniBand spec., a packet receiver, in general, is designed **As** indicated by the arrow *a* in the figure, all fields of a packet are stored in a temporary buffer, which consists of memory cells and its controller. Then, the packet is analyzed and error-checked by the inspection logic as shown by the arrow of *0.*If t here is any error, it is indicated by the signal *0* from the inspection logic and the corresponding packet is discarded. If no error is detected, the packet is sent to the corresponding FIFO toward upper layer as the arrow of 6 and the packet is sent to the upper lay

er **as** the arrows of *6).*

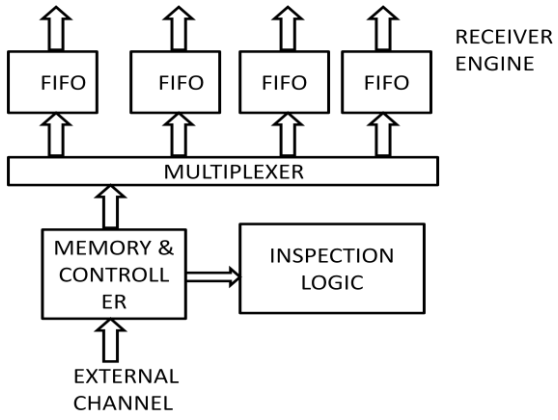


Fig.2 Architecture For Packet Checking & Buffering Engine

For current packet-based network communications and *VO* standards, which deal with relatively slow data stream, the above architecture can process the data stream in time ***[9,*** 101.However, it may be inefficient or even impossible to apply the architecture to new standards that require very high bandwidth ranging from several Gbps to several tens of Gbps. This is because the Memory and Controller block in Fig. 2 cannot afford to process all incoming packets rapidly and becomes the bottleneck of the receiver resulting in the increase of the processing latency. This section proposes an efficient high-speed packet buffering architecture and a FIFO circnit supporting high-speed network or *U 0* standards. The proposed architecture is shown in Fig 3. The Memory and Controller block, bottleneck in the conventional approach, is removed in the proposed architecture. The packet data is stored immediately in the corresponding FIFO as soon as received from external channels. This is possible because the packet header has the corresponding **FIFO** address. When the input packet is being stored, the inspection logic performs real-time checks about various fields of the packet and accumulates the CRC value calculated at every clockcycle. If there is an error at the end or in the middle of packet receiving, the packet data being stored into the FIFO is immediately discarded. The upper layer **also** detects errors while pulling out data from the FIFO. If an error is detected, thewhole packet is discarded from the FIFO.

4. PROPOSED PACKET BUFFERING ARCHITECTURE

When compared to the conventional architecture of Fig. 2, the proposed architecture reduces the latency because the input packet is directly stored into the FIFO. In addition, the proposed architecture does not conlain any bottleneck, and therefore, it can efficiently support the bandwidth available in the external channel.

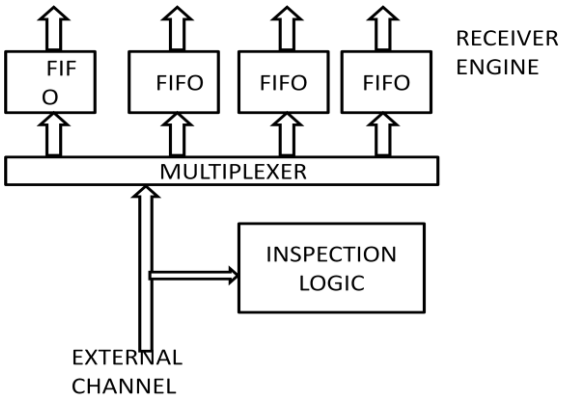


Fig. 3. Proposed Packet Buffering Architecture

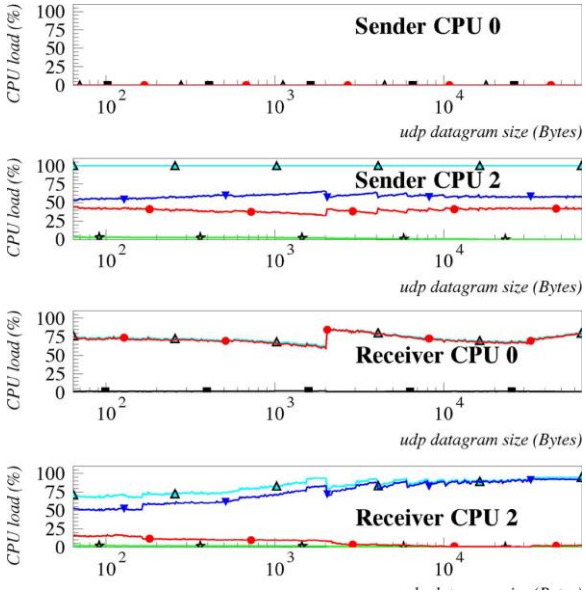
In order to implement the above architecture, the FIFO circuit needs to be capable of discarding packets immediately (within one clock cycle) by the signals indicating error detection. These signals can be generated by both the input side and the output side, i.e., the current layer and the upper layer. A novel FIFO circuit represented in Fig. 4 **is** proposed to handle the packet discard. The data input and output method is the same as a conventional FIFO. The 'write-data-in' and the 'read-data-out' are the input data bus and the output data bus, respectively. The input data is latched into the FIFO when the'write-enable-in' signal is asserted 'high' while the 'read-enable-in' signal is used to enable data read from the FIFO. The 'full-out' signal is asserted 'high' when the FIFO is full and the 'empty-out' signal indicates that the FIFO is empty. and the 'wad&-bound' points to the address of the fust data of the current packet. If any error is detected by the inspection logic, 'wad&-load' is asserted. Then, the value of the 'wad&-

bound' is loaded into the 'waddr' register. The change of the 'waddr' value means that the input data Gom 'wad&-bound' to 'waddr' is all discarded. Since the load of 'waddr' takes place in one cycle, the data discard is performed in one cycle, too. If the packet has no error, the value ***of*** the 'wad&' register is loaded into the 'wad&-bound' register enabled by the assertion of the 'waddr-bound-load' signal. In the output side, a packet can also be immediately discarded while data being transmitted to the upper layer by using the 'rad&-plus-1' register. The register points to the address of the current output data and is set to the new address to discard thewhole packet when error is detected. The 'upper-layer-discard' signal indicates the occurrence of errors. The initial value of a register at system boot-up is indicated by 'set to 0 **or** 'set to 1'denoted just below each register in Fig. 4. The 'waddr-plus-1','waddr-bonndglus-1'. 'waddr-bound','waddr', 'rad&-plus-**l',** and 'rad&-plus-1-reg' registers are used to expressaddresses of the 'dual-port-sync-sram' memory cell in the FIFO circuit. These registers are updated according to the three-hit control signals, write-allow, wad&-bound-load, and waddr-load. .

5. IMPLEMENTATION

For the implementation of the InfiniBand Link layer core, the 0.18-pm Faradayw standard cell libraty (FSAOOOA) is used for gate-level logic synthesis. The gate count and propagation delay are optimized by the Xilinx Tool. The BIST (Built-In Self-Test) wrappers for the memory cells are inserted before synthesis and the scan insertion for DFT (Design For Testability) after synthesis is performed by the Xilinx tools. The design includes seven dual-port synchronous SRAM cells, five SBllOO4Os ***(2048*** **x** 64-bit) and two SB104040s ***(5*** 12**x** @-bit). The total gate count of the Link layer core is 106,970,including BIST wrappers while excluding the memory cells and the DFT scan logic. With memory cells included, the total gate count is 1,328,229. The normal operating clock speed is 125 MHz and the critical path delay is about 7.211s on the data output path of a SB 110040 through a BIST wrapper in worst-case from the static timing analysis with the SynopsysN F'rimeTime.

6. RESULT



7. CONCLUSION

This paper presents an implementation of the InfiniBand Link layer and proposes a high-speed packet buffering architecture with **a** new **FIFO** circuit. The proposed architecture and **FIFO** remove temporary buffers and consequently reduce hardware cost and power consumption. In addition, the Link layer core can efficiently utilize theavailable bandwidth of InfiniBand. The proposed architecture and FIFO are also applicable to the state-of-the-an ***U0*** standards that have high-speed switched fabric architectures such as HyperTransport, RapidIO, PCI Express, Fibre Channel, Gigabit Ethernet, etc.

8. REFERENCES

[1] JNICorp.(2001,November)An Introduction to InfiniBand Bridging I/O up to Speed. *White Papers* [Online] Available: http:www,jni.comlProductslibsfm

[2] InfiniBand Trade Association official Homepage [Online] Available: http:l/www.infinibandtaorg

[3] W. T. Futral, *InfniBand Archilecture Development ond Deployment: A Strolegic Guide to Server I/O Solurion.* Santa Clara, CA Intel Press,2001.

[4] T. Shanley. *InfniBand Network Archilecture.* Boston, MA Addison-Wesley,2002

[5] VIEO Inc. (2W1, July) InfiniBand Architecture: Evolution/ Revolution - InfiniBand Adoptionin the Enlerprise Datacenter. *While Papers* [Online] Available: http://www.vieo.comlvieo_ whit.html

[6] InfiniBand Trade Association (2001, June) hfiniBandm Architecure Release I .Oa Volume 1. *InfiniBand Specificorion* [Online] Available:http://www.infiniband.org/specs

[7] R. F. Hobson and K. L. Cheung, "A high-performance CMOS 32-bitparallel CRC engine." *IEEE J. Solid-Slate Circuits.* vol. 34. no. 2.pp 233-235, February 1999.

[8] S. K. A. Yaklaf, B. M. Ali, V. Prakash, S. Khatun. and A. Gasim, "Data link control layer performance for variable packet size and fined packet size(wireless ATM packet)," in *PIDC. IEEE TENCON,* vol. 3. Sep. 2000 pp. 404-409.

[9] Y. C. Park, C. S. Yoon, K. M. Jung, and S. W. Min. "Design of the core modules for multimedia packet processing in a novel home gateway," in*Proc. IEEEInt. Con$ on Commun. Syst. (ICCS' 2002).* vol. *2,* Nov. 2002, pp. 952-956.

[10] J. Chen, P. Lanmer. and J. Kumar. "A flexible design of Dackets over SONET or directly wer fiber," in *Proc. IEEE Int. Symp. Circuirs Syn.(ISCAS'ZWO).*vo l. 1. May 2000, pp. 375-378.