MUG5: Modeling of Universal Chiplet Interconnect Express (UCIe) Standard Based on gem5

Xiaoyan Li
Department of Micro/Nano Electronics
Shanghai Jiao Tong University
Shanghai, China
lixiaoyan981103@sjtu.edu.cn

Zizheng Dong
Department of Micro/Nano Electronics
Shanghai Jiao Tong University
Shanghai, China
dongzzh@sjtu.edu.cn

Shuaipeng Li
Department of Micro/Nano Electronics
Shanghai Jiao Tong University
Shanghai, China
sjtuer_lsp@sjtu.edu.cn

Sai Gao
Department of Micro/Nano Electronics
Shanghai Jiao Tong University
Shanghai, China
gsw_96@sjtu.edu.cn

Jianfei Jiang
Department of Micro/Nano Electronics
Shanghai Jiao Tong University
Shanghai, China
jiangjianfei@sjtu.edu.cn

Guanghui He
Department of Micro/Nano Electronics
Shanghai Jiao Tong University
Shanghai, China
guanghui.he@sjtu.edu.cn

Zhigang Mao
Department of Micro/Nano Electronics
Shanghai Jiao Tong University
Shanghai, China
maozhigang@sjtu.edu.cn

2023 IEEE 15th International Conference on ASIC (ASICON) | 979-8-3503-1298-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/ASICON58565.2023.10396422

***Abstract*—In the post-Moore era, chiplet heterogeneous integration technology is gaining attention as a way to address the scaling limitations of monolithic chips. The Universal Chiplet Interconnect Express (UCIe) standard defines a complete stack for inter-chiplet communication, ensuring interoperability among chiplets. This paper introduces MUG5, a UCIe link model that enables accurate latency estimation through gem5-based simulation. The targeted mode of operation in UCIe is PCIe 6.0 with standard 256B flit (flow control unit). The model focuses on flit packing and Ack/Nak-based retry mechanism. We validate the model on two commonly used system topologies, and the results demonstrated that its deviation is within 0.04ns.**

***Keywords*—UCIe, PCIe, flit mode, gem5**

I. Introduction

The traditional scaling of monolithic chips is encountering challenges in terms of physical limitations and economic considerations. Chiplet heterogeneous integration technology has garnered significant attention as a promising solution [1]. Chiplet technology, integrating multiple modular dies into a single package, addresses issues related to scalability and costs while enhancing development speed, reusability, and yields.

The spread of chiplet technology needs an interoperable and universal interconnect standard. A consortium comprising industry leaders such as Intel, AMD, and ARM collaborated to develop the UCIe standard at the package level in March 2022. UCIe offers high-bandwidth, low-latency, power-efficient, and cost-effective connectivity between chiplets [2]. To ensure broad interoperability, the UCIe standard provides multiple options across various aspects. For instance, at the protocol layer, it supports widely adopted PCIe (PCI Express) and CXL (Compute Express Link), as well as user-defined streaming protocols. The standard also defines two types of packaging technologies: the cost-effective standard (2D) package and the power-efficient advanced package (2.5D). In terms of usage models, it supports both package-level integration and off-package connectivity. Moreover, it offers flexibility in terms of data rates, widths, bump-pitches, and channel reach, catering to diverse requirements. Another significant challenge in chiplet technology is the need for upgrading EDA tools to accommodate the new chip design process. Chiplet-based systems can be conceptualized as dividing the functionalities of a single SoC into multiple chiplets. Therefore, during the early stage of system development, careful attention must be paid to architectural design and function partitioning. In architectural design, an essential factor is the communication overhead between chiplets, highlighting the importance of a simulator that provides communication latency information.

gem5 [3] is a widely used open-source system simulation platform in computer architecture research. Renowned for its versatility, gem5 has amassed over 5,000 citations and has gained significant adoption across various industrial research labs, including ARM, AMD, Google, among others. One of gem5's notable strengths is its event-driven mechanism and comprehensive full-system simulation capabilities, making it well-suited for modeling protocols and their complex behaviors.

Based on gem5, this study focuses on the modeling of the UCIe link, specifically examining its functionality and latency characteristics. To our knowledge, this research represents the first endeavor to successfully model and simulate the UCIe link using gem5. The developed link model not only establishes a solid foundation but also enables gem5 to serve as a powerful simulator for architectural design exploration of chiplet-based systems.

The rest of the paper is organized as follows. Section II introduces the detailed modeling implementation, focusing on the event scheduling methodology and the retry mechanism. Section III presents the validation of the link model, assessing its functionality and latency characteristics. Section IV draws the conclusion and provides an outlook on the extensible capabilities of the proposed link model.

II. Model Implementation

This work models UCIe link of PCIe 6.0 with standard 256B flit mode from the Universal Chiplet Interconnect Express Specification Revision 1.0 [4] and the PCI Express Base Specification Revision 6.0.1 [5]. The layout of the standard 256B flit mode packet is illustrated in Fig.1. The packet consists of 236B from the PCIe protocol layer, along with 6B DLP (Data Link Packet) and 4B CRC (Cyclic Redundancy Check) from the UCIe die-to-die adapter layer.

979-8-3503-1298-0/23/$31.00 ©2023 IEEE

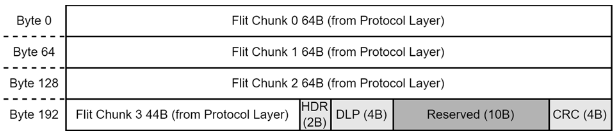
Byte 0
Flit Chunk 0 64B (from Protocol Layer)
Byte 64
Flit Chunk 1 64B (from Protocol Layer)
Byte 128
Flit Chunk 2 64B (from Protocol Layer)
Byte 192
Flit Chunk 3 44B (from Protocol Layer)
HDR (2B)
DLP (4B)
Reserved (10B)
CRC (4B)

Fig. 1. Layout of Standard 256B Flit Mode Packet

In the UCIe electrical physical layer, two types of packaging technologies are supported. The standard package has a relatively large bump pitch and channel reach, which results in a reduced number of die-to-die (D2D) lanes that can be accommodated within the same physical dimensions[6]. In the mainband group of the interface, the standard package contains one-fourth the number of data lanes compared to the advanced package. Specifically, the standard packaging contains 16 data lanes in the transmission side of each module, while the advanced packaging includes 64 data lanes. Both packaging types support data rates of 4, 8, 12, 16, 23, and 32 GT/s.

In the model, we choose the most basic electrical layer of the standard package interconnects with 16-bit data width and 4 GT/s data rate, resulting in the PHY operating at 64 GT/s. To ensure the consistency between the speed of the digital portion and the analog PHY, the digital data path is configured to operate with 256 bits at 250 MHz. This configuration ensures compatibility and coherence between the speed of the digital portion and the analog PHY.

A. Overview of UCIe Link Model

In gem5, communication between objects is achieved using packets which are transmitted across ports. There are two types of port in gem5: request ports and response ports. The link between these two port types is unidirectional. The master port is responsible for sending requests and receiving responses, while the slave port receives requests and sends responses. Considering the bidirectional UCIe link, two gem5 links are required to model communication in both directions.

Fig. 2 illustrates the overview of UCIe link model. On the left side of the figure, the scope of the model is depicted within a real chiplet-based system, indicating the communication link between UCIe controller 1 and controller 2 via physical wires. The UCIe controller in the model encompasses the protocol layer, die-to-die adapter layer, and PHY layer. On the right side of Fig.2, a detailed implementation diagram of the model is presented. The UCIe link model consists of two link interfaces, each comprising a RequestPort and a ResponsePort.

The link interface can be connected to any chiplet component in gem5. When any port receives a gem5 Packet, the internal flit packer will pack the TLP into the nearest available flit(s) and transmit it to the peer link interface.

Additional data paths in the link model are for the retry mechanism, wherein the flit sequence number is denoted by the time tick. Within this framework, the flit packer enqueues duplicates of the packed flit packets into the retry buffer. Upon the reception of an Ack (Acknowledgement), the flit packets of the ack sequence number and all antecedent ones are freed from the retry buffer. Whereas when a Nak (Negative Acknowledgement) is received, the link interface blocks the reception of other TLPs and resends the flit packets pop from the retry buffer.

B. gem5 Event Schedule of Flit Mode

Fig. 3 depicts the packing process of two consecutive active TLPs in flit mode. The horizontal axis is in time sequence, with the unit of the cycle number of flit packer. It takes 8 cycles, equivalent to 32ns, to pack a 256B flit packet due to the 256-bit data width and 250MHz frequency of data path. In the Fig.3 scenario, the active TLP1 is 332B in size and is received at cycle 14. TLP1 spans three flits and occupies 20B of the 32B data width at cycle 25. This example demonstrates several corner cases such as the positioning of the TLP header at any DWORD (32-bits) aligned location, the concurrency of multiple TLPs within the data path of a single cycle, and the extension of a single TLP across multiple flits. It is important to remember that the following illustration of event scheduling and the retry mechanism is based on the same scenario in Fig.3.

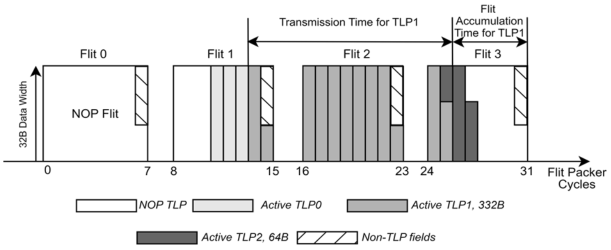

Fig. 3. Packing Process of Two Consecutive Active TLPs

Fig. 3 also shows the latency breakdown of TLP1 in the top row. The latency of the UCIe link model contains both transmission time and flit accumulation time. It denotes the time interval from the initiation of a valid packet on die1 on-

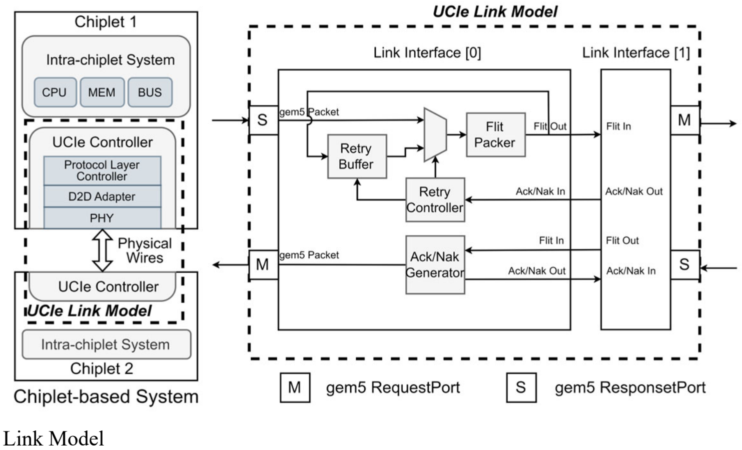

Fig. 2. Overview of gem5 UCIe Link Model

chip bus to its reception on die2 bus. It is noteworthy that the flit accumulation time is applicable only on the receiver side, as the transmitter side operates in a pipelined manner.

The latency of the link is modeled using event-driven programming in gem5 [7]. In this approach, each event is associated with a callback function or handler. When an event is scheduled, it is inserted into the event queue corresponding to the designated time tick. gem5 employs a global clock object to track the progression of simulation time. Within each cycle, gem5 schedules the events based on their priority until the event queue becomes empty. Only then does it proceed to the next time tick and repeats the event scheduling work.

Fig. 4 demonstrates the temporal sequence of event scheduling for the two consecutive TLPs in Fig. 3. The event scheduling of FlitPacker and FlitSender occurs in parallel, with arrows lines representing the scheduling relationship. Within the Link class, a txQueue is utilized to store TLPs which are awaiting transmission. FlitSender schedules sendFlit event every 8 cycles. The corresponding event handler function traverses the txQueue and transmits the packet(s) to the peer link interface. When there is no active TLP, FlitPacker schedules packNop event for each cycle. Upon the reception of a packet, the packTlp event is scheduled at the same time tick instead of packNop event. packTlp handler enqueues the packet into the txQueue after the transmission time, which is calculated based on the size of the TLP packet and vacancy of the current flit. Subsequently, at the 8-cycle-aligned time boundary, the event handler within the FlitSender dequeues and dispatches packet. In scenarios where txQueue.push and sendFlit are coincidently scheduled at the same time tick, it is imperative to assign the priority of events to ensure the successful transmission of the packet. Also, to effectively respond to recvPkt() within the same time tick, it is crucial that the priority of the packTlp event set to be the highest.

C. Modeling of Retry Mechanism

The UCIe protocol provides multiple mechanisms to ensure reliable transmission over the link. For example, CRC (Cyclic Redundancy Check) is employed to detect transmission errors, while a retry mechanism is leveraged to rectify the errors. Both mechanisms are implemented at the UCIe adapter layer. The UCIe link model designed in this paper supports the retry mechanism by using Ack/Nak DLLP (Data Link Layer Packet) at the granularity of flit.

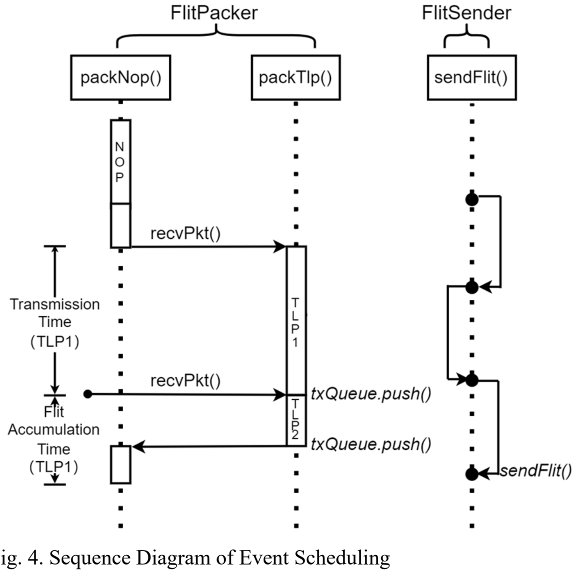

Fig. 4. Sequence Diagram of Event Scheduling

Regarding the operational principle of Ack/Nak DLLP in UCIe, on the sender's side, when flit packer assembles 236B from the protocol layer, it inserts an 8-bit flit sequence number into the corresponding bit field of the flit header. While the sequence number is theoretically incremental, in this model, the time tick is employed as the sequence number. The duplication of the flit packet is stored in the retry buffer before transmitting the packed packet to the peer link interface. The retry buffer retains a flit until an Ack DLLP confirms its error-free reception. In contrast, if the receiver identifies a flit with error, it sends a Nak DLLP containing the sequence number corresponding to the last accurately received flit. The sender subsequently removes and frees flit packets in the retry buffer with sequence numbers less than or equal to the sequence number in the Nak packet and retransmits all flit packets stored in the buffer. During the implementation, it is important to note that the transmitter can choose to skip over the NOP flits in the replay, the sequence number "0" is solely reserved for Idle Flit, and NOP Flit does not consume flit sequence number.

Fig.5 is the schematic diagram of the retry buffer and flit descriptor. When a new flit is packed, the corresponding flit descriptor, realized as a linked list, is enqueued into the retry buffer. The descriptor linked list comprises tuples, with each tuple containing two elements: a pointer targeting a segment of a specific TLP payload data and the length of this segment.

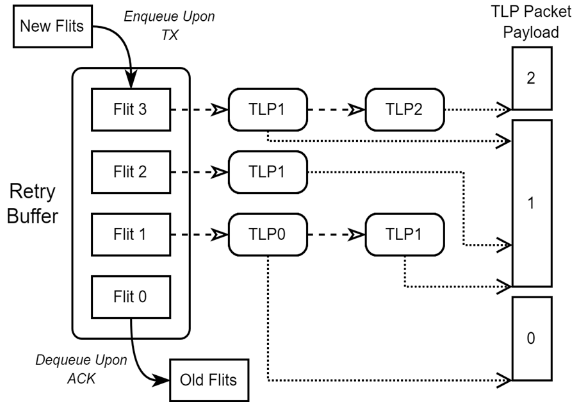

Fig. 5. Schematic Diagram of the Retry Buffer and Flit Descriptor

The use of a linked list is advantageous due to the variable number of TLPs in the flit packet and the retry buffer's sequential element fetching nature. By storing pointers in the tuples, the data duplication for TLP packet payloads is eliminated, resulting a zero-copy implementation, which in turn saves memory space and enhances simulation performance. In the model, Nak DLLP is generated randomly based on the modeled physical wire BER (Bit Error Rate), which is ssupplied by the user as a parameter. The link interface accumulates the count of received flit packets and generates the Nak with the probability derived from the BER. Upon the reception of a Nak at the transmission end, the retry controller activates the retransmission path (which in turn blocks the normal transmission path), sending the flits stored in the retry buffer.

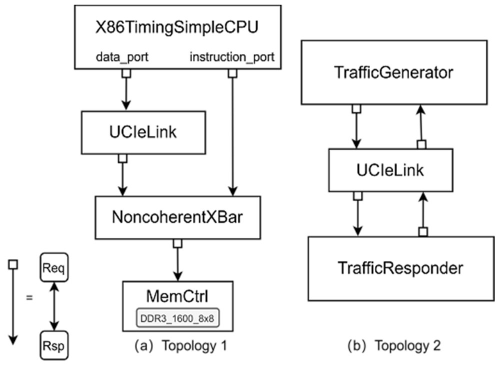

Fig. 6. Validation Typologies

III. Model validation and Discussion

gem5 takes a Python script as its input, which sets up the simulation system before the execution of simulation. To validate the UCIe link model, we constructed two topologies, as depicted in Fig.6. Topology 1 employs an x86 CPU model loaded with test programs, and it targets the functional validation. The CPU's data port is connected to an XBar through our UCIe link. This configuration facilitates the validation of the flit packing mechanism as well as the message transmission functionality within the model.

The next step is the validation of latency scheduling. As shown in Fig. 6(b), a TrafficGenerator is employed to dispatch test packets of various predefined lengths. The choice of stimulus packets refers to [8]. The theoretical latency excluding retry is outlined in Tab.1. (1) provides the latency formula, utilizing the average accumulation time. Accumulation time refers to the time interval between the reception of the last byte of TLP and the completion of the current flit packet on the receiver side due to the CRC mechanism. Theoretically, accumulation time is dependent on the size of the TLP packet and the frequency of the data path. Assuming the data path frequency is f GHz, the possible values for accumulation time can be represented as {nT}, where n is in the range 0, 1, 2, ..., (32f - 1).

$$\bar{t}_{accumulation}\ (ns) = \frac{\sum_{n=0}^{32f-1} nT}{32f} = 16 - \frac{1}{2f}$$

$$t_{latency}\ (ns) = t_{transmission} + \bar{t}_{accumulation} \qquad (1)$$

$$= \frac{TLPSize(B)*8}{64\ Gb/s} + 16 - \frac{1}{2f}$$

Considering a data path frequency (f) set at 0.25 GHz, the theoretical latency is listed in Table 1. The simulated latency is calculated as the average value across 100,000 simulation runs. In the test system, the TrafficGenerator issues memory write requests by invoking the sendTimingReq function, utilizing gem5 TimingRequest to mimic memory transactions at the protocol layer. Latency measurement starts from the moment link interface 0's ResponsePort receives the TimingReq and ends when the RequestorPort on link interface 1 sends the corresponding TimingResp. This setup effectively captures the latency involved in transmitting data across the UCIe link without overhead of the OS. The outcomes exhibit that simulated latency of the UCIe link model is close to the theoretical values, and the average error is within 0.04ns, thereby affirming the accuracy of the latency scheduling in the model.

TABLE I. Latency Measurement

Data Size (B)	*TLP Size (B)*	*Theoretical Latency (ns)*	*Simulated Latency (ns)*
16	32	18	17.9188
48	64	22	22.0180
80	96	26	26.0212
112	128	30	30.1364
240	256	46	45.9988
496	512	78	77.9988
880	896	126	126.1364
1008	1024	142	141.9988
2032	2048	270	269.9988
4080	4096	526	525.9988

IV. Conclusion

In this paper, we present a gem5-based modeling of the UCIe link, targeting the "PCIe 6.0 with standard 256B flit mode" defined in [4]. The proposed link model implements the flit packing logic, leveraging gem5's event scheduling feature and the retry mechanism. Through simulation data, we validate the functionality of the model and verify its accuracy in terms of latency. More importantly, this work explores the feasibility of employing gem5 as an architectural simulator in the early-stage design of chiplet-based systems.

In future research, the developed link model in gem5 can serve as a foundation for integrating and interconnecting various chiplet models, enabling system-level simulations. However, it is crucial to acknowledge that although the UCIe link model demonstrates functional correctness, further rigorous evaluation against real interconnects is necessary. This highlights the need for continued research and testing to ensure the reliability and expand the functionality of the model such as a more detailed model for the UCIe PHYwhich covers various physical parameters.

References

[1] T. Li, J. Hou, J. Yan, R. Liu, H. Yang, and Z. Sun, "Chiplet heterogeneous integration technology—Status and challenges," Electronics, vol. 9, no. 4, p. 670, April 2020.

[2] D. Das Sharma, G. Pasdast, Z. Qian, and K. Aygun, "Universal chiplet interconnect express (UCIe): an open industry standard for innovations with chiplets at package level," IEEE Trans. Compon., Packag. Manufact. Technol., vol. 12, no. 9, pp. 1423–1431, September 2022.

[3] N. Binkert et al., "The gem5 simulator," SIGARCH Comput. Archit. News, vol. 39, no. 2, pp. 1–7, May 2011.

[4] "Universal chiplet interconnect express (UCIe) specification revision 1.0." February, 2022. [Online]. Available: https://www.uciexpress.org/specification

[5] PCI Special Interest Group, "PCI Express base specification revision 6.0, version 1.0." Jan. 11, 2022. [Online]. Available: https://pcisig.com/specifications

[6] D. D. Sharma, "System on a package innovations with universal chiplet interconnect express (UCIe) interconnect," IEEE Micro, vol. 43, no. 2, pp. 76–85, March 2023.

[7] A. Butko, R. Garibotti, L. Ost, and G. Sassatelli, "Accuracy evaluation of gem5 simulator system," in 7th International Workshop on Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC), July. 2012, pp. 1–7.

[8] D. Das Sharma, "PCIe 6.0 specification: a high-performance I/O interconnect for advanced networking applications," 2022 OFA Virtual Workshop, April, 2022. [Online]. Available: https://www.openfabrics.org/wp-content/uploads/2022-workshop/2022-workshop-presentations/206_DDasSharma.pdf