2019 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Use of Message Bus Interface to Verify Lane Margining in PCIe

Ankita Vashisht, R&D Engineer, Synopsys India Pvt Ltd, New Delhi, India
(vashisht.ankita4@gmail.com)

Narasimha Babu G V L, Director PCIe, Synopsys India Pvt Ltd, Bengaluru, India
(NarasimhaBabu.GVL@synopsys.com)

***Abstract*— The purpose of this paper is to demonstrate the usage of Message Bus Interface (MBI) to verify Lane Margining at the Receiver (commonly referred to as "Lane Margining") introduced in the PCIe 4.0 specification to address the problem of signal degradation. The Message Bus interface provided a way to participate in verifying non-latency sensitive PIPE operation without adding any additional wires. In Gen4, it was basically used to control Lane Margining, to measure the available margin in a standardized manner. The usage suggested accommodates different design parameters of different DUTs and is hence a generalized usage without any specific assumptions.**

Keywords—Message Bus Interface; Lane Margining; no-latency; standardized

I. INTRODUCTION

With the Introduction of Gen4 PCIe, there was a new challenge which designers faced. They had to address the demands for higher bandwidth (double as compared to Gen3) while still maintaining the quality of the signal transmitted. They also had to take into account transmission losses over a larger distance which would be evident while using the DUT.

PCI-SIG addressed this challenge with the introduction of the PCI Express 4.0 (i.e. Gen4) lane margining at the receiver end feature. It allowed system designers to assess the performance variation tolerance (in terms of timing and voltage margins) of their system. Lane margining allowed designers to measure the available electrical margin in each system and also provided a medium to calculate the tolerable limits within which the signal quality wouldn't be degraded in transmission. [1]

PCI express supports both internal & external connections (between VIP & Physical Layer of DUT, and, between VIP & Controller DUT). Due to the variations in manufacturing, different components have different performance indexes. All performance degrading factors like noise, cross talk, humidity etc. must be considered while designing any component [2]. To avoid issues due to these factors, verification of lane margining is mandatory at all receiver ports (for PCIe 4.0 onwards i.e. Gen4 onwards).

The actual implementation of the lane margining feature in both the PHY and the controller DUT is design-specific. Presented in the paper is our employed approach of using MBI to verify the lane margining requirements of the PHY DUT. Local PHY lane margining (i.e. by using MBI) is supported in PCIe VIP (PIPE Mode) using transmission of lane margin command over the MBI interface signals (i.e. p2m & m2p) defined in PIPE v4.4.1 specifications.

To evaluate the margins available, some designs utilize data or error samples while others use jitters in the PHY. The usage illustrated in the paper elaborates on the two possible topologies that can be used for verification purposes. It also elaborates on the different kinds of error and normal scenarios possible for the verification purposes. Further, it also talks about the performance variations of DUT due to this feature.

The need for this methodology was recognized to expand the scope of using Message Bus Interconnect in verifying the various PHY & MAC characteristics (like Status & Control Registers, design parameters.) Also, MBI has much of scope in the subsequent PIPE 5 specifications & using this methodology would be the stepping stone for its further enhancement.

The Controllers could process the data differently by using different timing & voltage offsets, voltage, and timing steps among others. Also the error tolerance could be different for different evaluations.

II. BACKGROUND

A. Need for margining

Technology is moving towards Big Data, IoT, Machine Learning, Large Storage, networks amongst other upgradations. These require high performance systems to be in place. In order to meet the requirements, either the bandwidth or the speed needed to be increased. [3]. Increasing the lane width was not feasible due to the additional

cost in adding connections, lanes. Also, it would require some changes in the existing architecture (in terms of disabling the unused lanes).

Thus, in PCIe 4.0 specification, bandwidth was increased from 8 GT/s to 16 GT/s per Lane while maintaining backwards compatibility with previous versions of the specifications. This would ensure better performance without compromising on speed. The problem with higher data rate was it limited the distance over which a signal could be transmitted without channel loss or any other degradation in the signal. Thus, a concept of lane margining at the receiver was devised to assess the tolerance limits in the system. Lane margining provided a mechanism to measure available electrical and timing margins which could be leveraged for transmitting a signal without much degradation in quality.

The arrows in the below diagram are the margins available beyond the minimum eye width requirement.

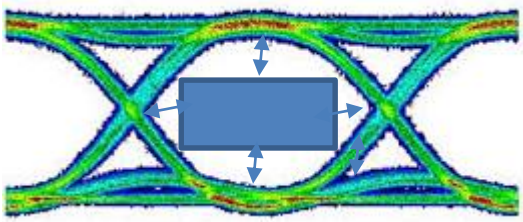

Figure 1: Eye Diagram for Lane Margining

B. Message Bus Interface (MBI)

MBI is a smaller portion of the PIPE interface. It enables additional PIPE functions to be added without using any additional wire. The use of this interface requires that the device is in power state and has a PCLK running [4]. It consists of 2 major signals: m2p (MAC to PHY) message bus & p2m (PHY to MAC) message bus. The data on these buses in coordination with the clock & reset signal is responsible for lane margining. With the help of m2p message bus, we schedule the commands/data etc. onto the PHY and then wait for appropriate responses from PHY within certain specification defined timings.

C. Description of Control and Status register usage

In PIPE 4.4.1, there were 2 categories of registers: PHY & MAC registers. The configuration & usage is as described:

i. PHY Registers

   a. RX Margin Control0: Controls RX Margining by providing commands like sample count reset, error count reset, margin voltage or timing, start margin

   b. RX Margin Control1: Controls RX Margining by providing commands like setting the margin offset & margin direction.

   c. Elastic Buffer Control: Controls the number of write committed (ones which require a Write Ack) and write uncommitted (not requiring an ACK) can be scheduled before writing them on the bus.

ii. MAC registers

   a. RX Margin Status0: Provides responses from PHY including Write Nak, Margin Status updates.

   b. RX Margin Status1: Provides sample count update from PHY while margining is in progress.

   c. RX Margin Status2: Provides error count update from PHY while margining is in progress.

   d. Elastic Buffer Status: Provides status of elastic buffer depth adjustment.

To initiate margining, Margin offset, Margin type & direction, start margin bit is written on by the MAC. When the PHY receives such a request, it responds with a write_ack followed by updating the margin status. The subsequent writes, reads etc determine what parameters will be tested & verified.

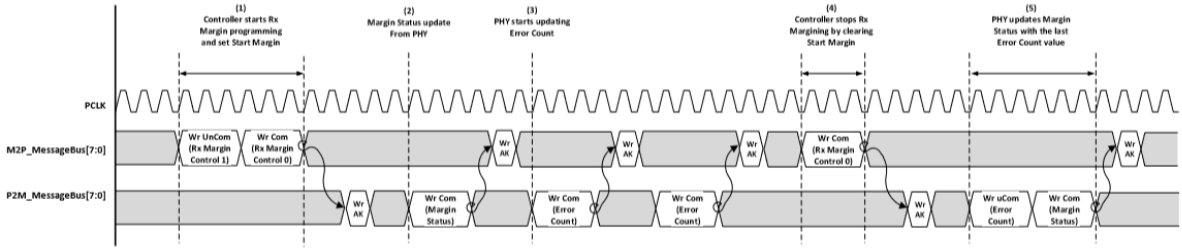

Figure 2: A sample RX Margining Sequence

D. Difference between Gen5 & Gen4 MBI

In Gen4, MBI was only used for lane margining verification. In Gen5, the scope has been expanded to verify receiver & dynamic equalization, elastic buffer control, TxDeemph, FS, LF settings, polarity inversion & block align control.

E. Challenges in margining & approach to overcome those

1. PCIe is used in a wide variety of devices and platforms thus leading to large variations in voltage, timing, temperature and other environmental factors. This results in a variance in actual signal performance. Thus different assumptions would be required for different DUTs.
   To cater to this problem, different parameters were provided as test suite defines, controllable test suite & sequence configurations so that the users could set the values as per their DUTs.
2. Some systems would require retime integration to address the challenges of signal integrity. To overcome the same, controls were also provided to enable retimer and specific scenarios targeting the functionality of retimers were also provided.

III. IMPLEMENTATION & WORKING

A. Setup requirement

The ideal setup for verification of lane margining is an environment running at 16 GT/s or higher speed where the VIP is either a Downstream or Upstream port initiating the Rx Margining requests to its own PHY. Rx Margining can be initiated using the serial interface via CTRL SKP Ordered Set or by using the PIPE interface via Message Bus Interface as per the requirement of the testing to be carried out. This paper dwells on the use of MBI to verify lane margining.
To use MBI, signal connections between DUT & VIP's m2p and p2m message bus should be in place. The device should be in power on mode and should have a running PCLK.

B. Different topologies possible

1. EP on PIPE

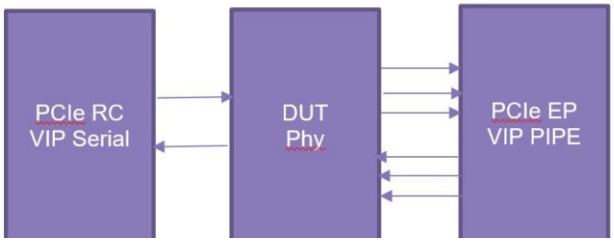

Figure 3: RX Margining when EP on PIPE

In this topology, the interactions happen between the EP on PIPE and the PHY DUT. EP initiates lane margining onto the PHY and waits for an acknowledgement from PHY. The RC on serial has not much role in the verification of lane margining with the use of MBI in such a scenario.

* Identify applicable sponsor/s here. If no sponsors, delete this text box (*sponsors).*

2. RC on PIPE

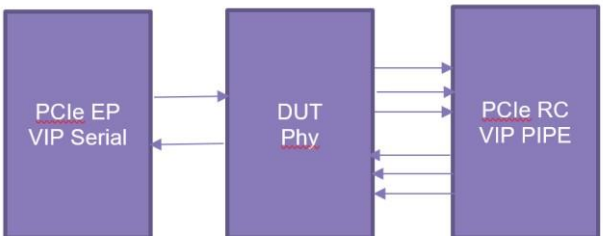

Figure 4: RX Margining when RC on PIPE

In this topology, the interactions happen between the RC on PIPE and the PHY DUT. RC initiates lane margining onto the PHY and waits for an acknowledgement from PHY. The EP on serial has not much role in the verification of lane margining with the use of MBI in such a scenario.

C. Elaboration on a typical scenario

A typical setup for margining is that both the serial & pipe VIPs are at 16 GT/s or higher speed. The link is also supposed to be in L0 state. After the above two have been reached, MAC (the PIPE partner) initiates margining.

Data is written onto the Control0 & Control1 registers via write uncommitted/ committed followed by write committed respectively. A user control was provided to determine direction & steps of margining either to a configurable value or randomly.

This is followed by receiving an ACK from PHY. This ACK can be received in configurable delays as per the DUT. A control was provided to configure the waiting time before declaring that ACK wasn't received for write committed.

Write ACK is succeeded by an update in Margin Status Register by the PHY, to declare that margining is in progress. This update has to be also received in a configurable time.

In our tests, this was followed by a mandatory wait to check margining for certain samples (by updated values of Sample count & error count.)

The above steps were followed by the scenario to be tested, the same is illustrated in next subheading.

As the verification was completed, stop margining was initiated from MAC to PHY by instructing to reset value of margin status.

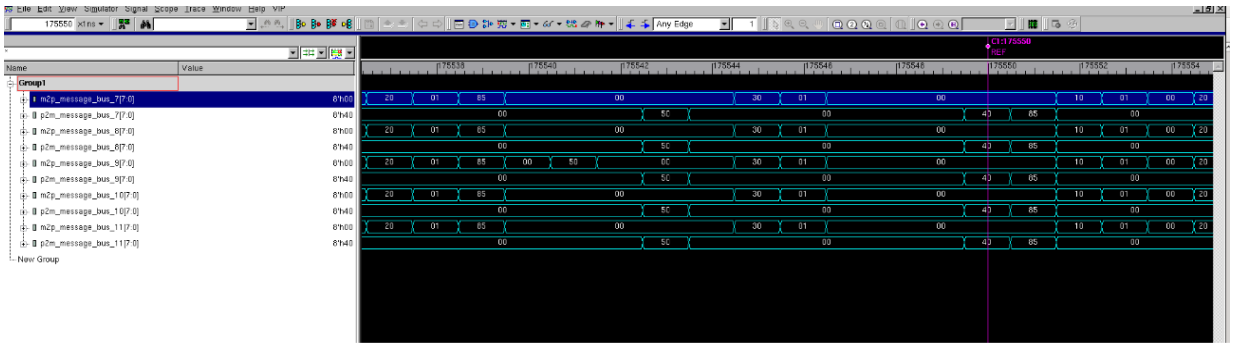

Figure 5: Waves corresponding to RX Margining

D. Examples of scenarios tested

A kaleidoscope of normal and error scenarios was verified. Some examples are illustrated below:

1. Timing margin left/right, voltage margining up/down – Tested deviations to be in prescribed limits.
2. Timing margin left to right/ right to left / voltage margin up to down/ down to up – PHY shouldn't change direction of margining while margining is in progress. The approach used was to read the PHY registers for margin direction and determine if they were correctly updated.
3. Timing margin to voltage margin/ voltage margin to timing margin: PHY shouldn't change type of margining while margining is in progress. The approach used was to read the PHY registers for margin type and determine if they were correctly updated

4. Reading PHY registers: Read was initiated onto the PHY registers to check values of Sample Count, Error Count, Margin type, Margin Direction, Margin offset. These were required to be same as ones written in start of margining. Only the sample count & error count could be as per the scenario.
5. Reading 1 cycle and level attributes in the PHY registers
6. Incrementing offset in each subsequent writes that were spaced by an interval of time.
7. Changing Data rate while margining: Margining can only progress at 16 GT/s or higher. In case, data rate change request is below this value; margining needs to be stopped. This request is initiated by MAC while device is in recovery state after initiating the speed change request.
8. Initiating Voltage margin when it isn't supported: PHY is expected to send a NAK.
9. Setting offset limit to value above allowed limits: In case of Timing margin, it should discard the request & in case of voltage margining, it should send a NAK.
10. Clearing Error and Sample count: While changing offset & while not changing offset: the error and sample count field in MAC registers should have a value less than previous read value.

E. Dut bugs caught

1. The value of delays in providing status updates, WRITE ACK were determined for various DUTs.
2. Certain ambiguities related to stopping margining on initiating margin direction or type change were cleared. (on what procedure to be accepted after such a request is obtained.)
3. Certain DUT capability related ambiguities were cleared. (Eg: Sample Count can be updated using frequency and count, there were issues in its definition which were cleared).
4. Certain DUTs didn't send write NAKs as specification. These had to be rectified.

IV. ABBREVIATIONS USED

A. Abbreviations and Acronyms

VIP: Verification Intellectual Property

DUT: Design Under Test

PHY: Physical Layer

MBI: Message Bus Interface

MAC: Media Access Layer

ACKNOWLEDGMENT

We would like to thank our employer, Synopsys India Pvt. Ltd. for supporting this paper. We would also like to thank the organizers of the Design and Verification Conference (DVCON) for their constructive feedback and the opportunity to share our work.

REFERENCES

[1] Rita Horner,https://www.synopsys.com/designware-ip/technical-bulletin/pci-express-4-lane-margining.html

[2] Dr. DebendraDas Sharma, https://pcisig.com/pushing-limits-understanding-lane-margining-pcie%C2%AE

[3], [4] PHY interface for PCIe, SATA, USB 3.1, Display Port, and Converged I/O Architectures Version 5.1