

PCIe GEN5 DATA LINK LAYER

For years PCIe has served as a standard for interconnects being widely accepted throughout the industry. This piece of writing aims to showcase the working of the PCIe Gen5 Data Link Layer, guarding data exchange between the components on a link.

By:

Amritanshu Rai & Adarsh Prakash Pandey

Contents

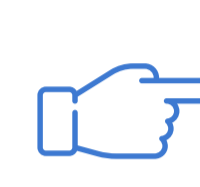
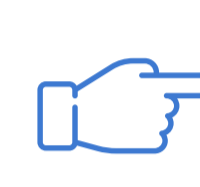
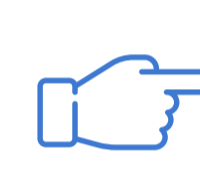
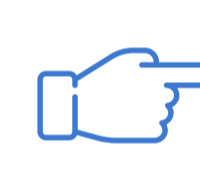
Introduction.....	3
What Does The Data Link Layer Do?.....	3
Data Link Layer Packets (DLLPs).....	3
Structure of DLLPs.....	4
ACK/NAK.....	4
Flow Control DLLP.....	5
Power Management DLLP.....	6
Feature DLLP.....	6
TLP & DLLP Processing in Data Link Layer.....	6
Structure of TLP (O/P of DLL).....	7
Link Management And Status State Machine.....	7
States Involved.....	8
States Output.....	8
Flow Control in DLL.....	8
Scaled Flow Ccontrol.....	10
Retry Mechanism in DLL.....	10
ACK/NAK Examples.....	11
Definition And Acronyms.....	13
References.....	14
About Us.....	14
LFT And PCIe.....	14

INTRODUCTION

PCIe facilitates the connection between two devices and is responsible for smooth data exchange between them. To handle the transfer mechanism appropriately PCIe has 3 layers, which are Transaction layer, Data link layer, and Physical layer. The Data link layer is the intermediate layer, which supervises link management, data integrity of packets in transmission and re-transmission.

WHAT DOES THE DATA LINK LAYER DO?

Data Link Layer administers the following tasks:

-  **Supervises the link management state machine (DLCMSM):** manages flow control initialization, conveys link status to adjacent layers of the PCIe and changes link activity.
-  **Data exchange:** DLL receives data from the Transaction Layer as TLPs, and concatenates it with sequence number and LCRC. The modified TLPs are sent to the physical layer, where they are encoded and transmitted onto the link to the receiver. At the receiver, it ensures data goes through proper scrutiny before it is received by the transaction layer.
-  **Error detection in transferred data:** LCRC ensures incoming data, that does not contain error, is transferred to the upper layers, thus maintaining data integrity. In case of link failure or data error, DLL also initiates backup functions such as Link re-training, or Retry Mechanisms.
-  **Flow Control:** Trade credit information (buffer space) available on both sides of the link which helps the transaction layer to plan TLP generation accordingly.

DATA LINK LAYER PACKETS (DLLPs)

DLLP is data output of DLL, and is 8 bytes wide, which comprise 2 bytes of LCRC and 2 bytes of framing fields. TLPs have route information which helps it to reach its intended receiver after passing through numerous ports. Unlike TLPs, DLLPs are not routed and do not contain data payload and are used for nearest neighbour communication. DLLPs convey information such as ACK, NAK, Init, Power management and flow control information.

If received DLLPs have errors, then it gets discarded. DLLP are sent periodically across the link and with successive occurrences they update the missing information in earlier DLLPs, allowing DLL to continue working without raising error for corrupted DLLPs. All DLLPs such as ACK/NAK, InitFC1, InitFC2, UpdateFC and power management DLLPs are structured to update the information with successive iterations. For example, If the receiver receives ACK for sequence number five, then it is expected that acknowledgement is for all TLPs before sequence number six, allowing DLL to work without sending ACK for every sequence number.

STRUCTURE OF DLLPs

All DLLPs follow a standard structure, 8byte wide data, made from 2 bytes of LCRC, 2 bytes of framing fields and reserved bits.

Byte 1	Byte 2	Byte 3	Byte 4
DLLP Framing Information	DLLP Type	DLLP Type Specific Information	
Byte 5	Byte 6	Byte 7	Byte 8
DLLP Type Specific Information	LCRC		Framing end Indication Byte

Figure 1: Byte-wise information of 8-byte DLLP Structure in Gen1/2

Byte 1	Byte 2	Byte 3	Byte 4
DLLP Framing Information 11110000 10101100	DLLP Type		DLLP Type Specific Information
Byte 5	Byte 6	Byte 7	Byte 8
DLLP Type Specific Information		LCRC	

Figure 2: Byte-wise information of 8-byte DLLP Structure in Gen3/4/5

Further in this section, we will discuss Byte 1 to 6 in the structural description of Gen1/2, since framing information is common across all DLLPs.

ACK/NAK

ACK is used by DLL to Acknowledge successful reception of TLP, and NAK is used to inform about unsuccessful reception of TLP. A TLP is considered “accepted”, if transmitted LCRC matches with calculated LCRC (at the receiver) and transmitted sequence number matches with expected sequence number (at the receiver).

Byte 1	Byte 2	Byte 3	Byte 4
ACK/NAK TYPE	12-bit Reserved	12-bit ACKNAK_SEQ_NUM	
Byte 5	Byte 6		
16-bit LCRC			

Figure 3: ACK/NAK Structure

FLOW CONTROL DLLP

In this category 3 types of DLLPs are scheduled Init1DLLP, Init2DLLP, and UpdateDLLP. All three type of DLLPs further tells about category of transactions {P(posted),NP(non-posted),Cpl(completion)}

Init1 DLLP

Byte 1			Byte 2		Byte 3		Byte 4	
P/NP/CPL type	0	3-bit VC-ID	2-bit Hdr Scale		8-bit HdrFC		2-bit Data Scale	
Byte 5		Byte 6						
16-bit LCRC								

Figure 4: Init1FC Structure

****HdrScale:** Scale factor for Header

HdrFC: Header credit value

DataScale: Scale Factor for data credits

DataFC: Data credit value

VC-ID: Virtual Channel Number: here zero to seven for 3 bits allotted

Init2 DLLP

Byte 1			Byte 2		Byte 3		Byte 4	
P/NP/CPL type	0	3-bit VC-ID	2-bit Hdr Scale		8-bit HdrFC		2-bit Data Scale	
Byte 5		Byte 6						
16-bit LCRC								

Figure 5: Init2FC Structure

Note: InitFC1 and InitFC2 have the same type of procedure for initialization, where transition from initFC1 to initFC2 confirms that the available credit values for P, NP and Cpl have been recorded for VCs, and InitFC2 is there to make sure that the connected device receives initial credit values (as it could still be in InitFC1 state) because it could take more time to record those shared credit values. When both devices transition to InitFC2, it represents initial credit exchange has happened successfully.

UpdateFC DLLP

UpdateFC is sent by the receiver to update buffer credits to the transmitter side during runtime.

Byte 1			Byte 2		Byte 3		Byte 4	
P/NP/CPL type	0	3-bit VC-ID	2-bit Hdr Scale	8-bit HdrFC	2-bit Data Scale	12-bit DataFC		
Byte 5		Byte 6						
16-bit LCRC								

Figure 6: UpdateFC Structure

POWER MANAGEMENT DLLP

Specifies the change of power management states depending on the link activity.

Byte 1		Byte 2		Byte 3		Byte 4	
PM TYPE		24-bit Reserved					
Byte 5		Byte 6					
16-bit LCRC							

Figure 7: Power Management DLLP Structure

FEATURE DLLP

Specifies the features supported by the transmission port.

Byte 1		Byte 2		Byte 3		Byte 4	
00000010		1-Bit Feature Ack		23-bit Feature Support			
Byte 5		Byte 6					
16-bit LCRC							

Figure 8: Feature DLLP Structure

TLP & DLLP PROCESSING IN DATA LINK LAYER

Data link layer handle two types of data packet:

- By modifying the TLPs received from Transaction Layer. DLL receives data payload from Transaction Layer and adds 2 bytes of header(12 bits of which are sequence number) and 32 bits of LCRC to the end to frame the TLPs. TLPs are routed and contain route information.

☞ DLLPs are created in this layer itself and are used in link control, flow control and error management. DLLPs are non-routed data and are meant for the nearest neighbour. While DLLPs are formed based on the type of DLLP, with 2 bytes of LCRC and 2 Bytes of framing information. Major DLLPs used are ACK/NAK, Flow Control DLLP and Power Management DLLPs.

STRUCTURE OF TLP (O/P OF DLL)

The TLP received from the Transaction layer is framed with 12 bit sequence number and a 32 bit LCRC. The structure of TLP is:

8 bit STP	12-bit TLP SEQUENCE NUM.	TLP From Transaction layer	32-bit LCRC	8 bit END
-----------	--------------------------	----------------------------	-------------	-----------

Figure 9: TLP Structure

Before PCIe starts operating, it needs to configure the link to work at best supported settings, by allowing receiver and transmitter to exchange information such as buffer size, local port supported feature, etc. This task is handled by DLCMSM (data link control and management state machine).

LINK MANAGEMENT AND STATUS STATE MACHINE

In PCIe, data travels through link between two connected ports. When the link is active, the DLCMSM will also go to active by exchanging the InitFC DLLPs. The DLL may ask the link to retrain if either the replay timeout occurs or the update fc timeout occurs which indicates that link is inactive and so needs to retrain. This process is supervised by DLL through a state machine DLCMSM. This has four states DL_inactive, DL_feature, DL_init, and DL_active. This machine produces two types of output DL_up and DL_down. An UpdateFC timer is also implemented at received side, which keeps track of interval between reception of UpdateFC DLLP. If the timer expires, then link is retrained. UpdateFC are sent when TLP are processed at receiver side, which creates credit space.

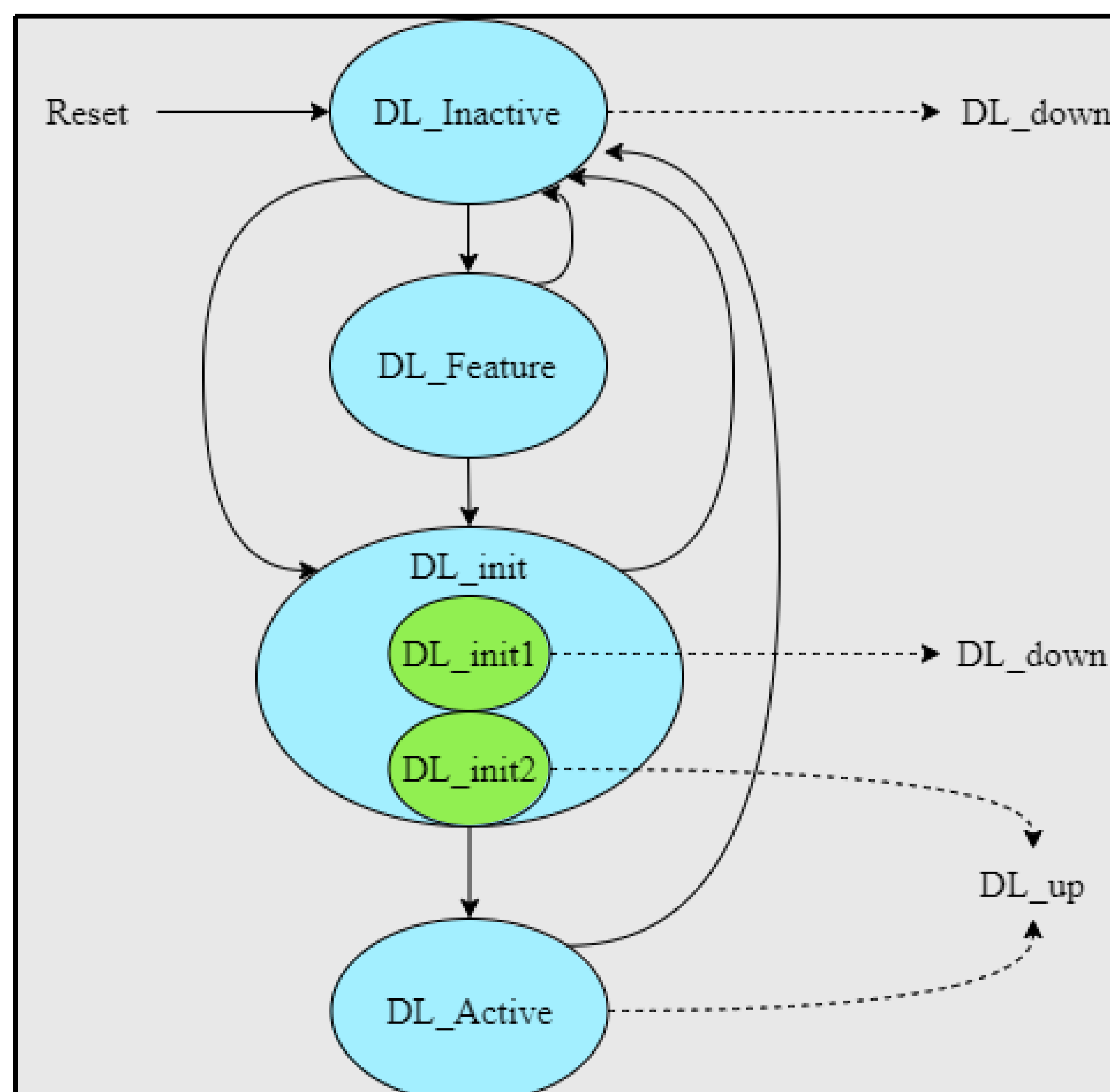


Figure 10: The DLCMSM

States Involved:

- ✓ **DL_Inactive:** The Physical Layer is reporting that the Link is not operational. Either the device is not connected on the other side or, the PHY layer is establishing the link with another end.
- ✓ **DL_Feature (optional):** Physical Layer reports Link is operational. In this state, the Data Link Feature Exchange protocol transmits a Port's Local Feature Supported information to the Remote Port and captures that Remote Port's Feature Supported information. This is required for Ports that support 16.0 GT/s and higher data rates.
- ✓ **DL_Init:** Physical Layer is reporting that the link is operational. Initialize Flow Control for the default Virtual Channel (VC0). The initialization process has two states, Init1 and Init2. At the time of initialization, the receiver advertises available buffer credits for each type of request PH, NPH, CplH, PD, NPD, and CplD. In this state, devices continuously send InitFC1 sequences in P, NP, and Cpl in order to advertise their buffer space to other devices' transmitters. These are sent frequently so the transmitter can properly see and record buffer credits information. The transaction Layer must block the transmission of TLPs from all VCs.
- ✓ **DL_Active:** Normal operations such as the transmission of TLPs can be performed.

States Output:

- ✓ **DL_Down:** this state signifies that the link between receiver and transmitter is not active and communication between receiver and transmitter is off. For states DL_Inactive and DL_Init1 state output is DL_Down, to show the link is paused.
- ✓ **DL_Up:** It is used to show that the link between receiver and transmitter is active. For states DL_Active and DL_Init2, the DLL has an output state at DL_up to show that the link is active.

FLOW CONTROL IN DLL

Before sending a packet, the transmitter needs information on whether the receiver has enough buffer space to accept it. The receiver stores the incoming data in buffers as processing data packets require time. If buffers are filled, then incoming data will be lost and will result in numerous retries by the transmitter. This will simply waste time since REPLAY_NUM value will roll over and the link will go into Re-training. Therefore the link efficiency will be affected severely, thus it requires a flow control credit-based mechanism. At the time of initialization, the receiver reports the size of its buffer to the transmitter on the other side of the link. During normal operations, available buffer space is kept updated by a DLLP called Flow control DLLP. The transmitter halts the TLPs exchange if sufficient buffer space is not available at the receiver of the connected device. Transmitter uses the formula

$$(\text{Credit Limit} - (\text{Cumulative Credit Req.})) \bmod 2^{[\text{Field Size}]} \leq 2^{[\text{Field Size}]} / 2$$

...Equation 1

where,

CL is credit limit, it gets incremented as more buffer space is made available at the receiver.

Cumulative Credit Required is the total credit used by previous TLPs and pending TLP.

Field Size is 8 for header and 12 for data.

Let's try to understand Flow control with an example. Consider a buffer size for posted header at the receiver is 1kB, and every credit is 5DW (20 bytes) in size. Then credit available is $1024/20=51d$ credits or 33h, thus Credit Limit=33h, and let's say 20h credits have already been consumed by earlier TLPs, then Credit Consumed=20h, and next TLP sent will take 1 credit, thus credit for Pending TLP=01h.

According to the formula

$$\begin{aligned}(33h - (20h + 01h)) \bmod 2^8 &\leq 2^8/2 \\ (12h) \bmod 256 &\leq 128\end{aligned}$$

Since the equation is true, then transmitter sends the pending TLP(PTLP).

Considering the receiver is taking time to consume the data available in the buffers, and now we have reached a state where the receiver buffer is full and not a single credit space is emptied. Thus, Credit Limit would remain as 33h but consumed credits will approach the value of credit limit, i.e. Credit Consumed=33h. Now, transmitter want to send one more TLP of size one credit unit, i.e. Pending TLP = 01h. In this case,

According to the formula

$$\begin{aligned}(33h - (33h + 01h)) \bmod 2^8 &\leq 2^8/2 \\ (FFh) \bmod 256 &\leq 128\end{aligned}$$

The equation is not true, and the transmitter stops further transmission on the channel until receiver empties some buffer space. Now, if we receive update FC DLLP showing two credit space is made available, then, Credit Limit = 33h + 02h = 35h. Then

According to the formula

$$\begin{aligned}(35h - (33h + 01h)) \bmod 2^8 &\leq 2^8/2 \\ (01h) \bmod 256 &\leq 128\end{aligned}$$

The equation is true, thus transmitter resumes transmission on the channel.

Due to use of 2's complement method for calculations, rollover of Credit Limit and Credit Consumed does not cause a problem. It will automatically takes care of the condition when Credit Limit rolls over to 00h. If, say, Credit Limit has rolled over to 08h and Credit Consumed is still F8, Equation 1 will still satisfy.

SCALED FLOW CONTROL

If the available buffer size is insufficient to account for link round trip, then link efficiency will be proportionally affected. This effect is even more pronounced when the link is operating at higher speed which requires more buffer size.. Also the bits in InitFC DLLP is fixed, thus can only represent upto 127 for headers and 2047 for data. Ports that support data rates above 16.0 GT/s must support scaled flow control. With scaled flow the same number of bits are able to represent a larger credit value, by right shifting the bit representation, scaling the value that can be represented by multiple of 4 or 16. The scale in the value of buffer size is depicted by the scale factor in the DLLP.

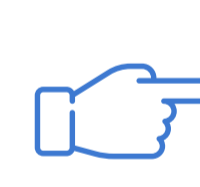
Let's say we have only 3 bits for credit value, then the highest value that can be represented is 7('111') but with a scale factor of 4 the same '111' will represent '11100' i.e. 28(4 times 7).

RETRY MECHANISM IN DLL

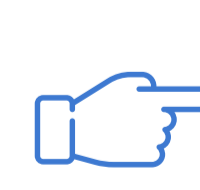
If, in any case, data integrity across the link is not intact, it means the information (TLPs) got compromised, and TLP is required to be sent again, and DLL initiates retry procedure. The retry mechanism is triggered in two cases: the first is when a NAK signal is generated, and the second is when RETRY_TIMER expires.

The DLL stores the sequence number and Data packets (transmitted as TLPs) in the retry buffer. The receiver has a job to periodically send acknowledgement for the packets received, which can be either positive or negative and are known as ACK and NAK respectively. When ACK is received then packets having the same or lower sequence number as shown in ACK DLLP are removed from the retry buffer. If NAK is received, then the packets with higher sequence number than shown in NAK DLLP are resent.

During communication across PCIe, one device transmits the DLLP and other receives the DLLP. Further in this section, the transmitter will be referred as device A and receiver will be device B.

 **REPLAY_TIMER:** it is the maximum time the transmitter will wait for ACK/NAK before replaying everything in the retry buffer. If the timer expires, then it will assume that data sent earlier was not received by device B and will initiate a process for retry. The timer is reset every time it receives either ACK or NAK. The timer also starts counting when the last symbol TLP is transmitted after receiving Acknowledgement.

When device A receives the NAK signal, the replay timer which was reset starts counting after TLPs are resent. However, upon receiving an ACK, the reset timer starts counting, if TLPs are present in the retry buffer. Otherwise, the timer is kept at 0 until the last byte of the current outgoing TLP is transmitted.

 **ACKNAK_latency_timer:** It is the time the receiver waits before sending ACK to the transmitter. Sending ACK for every received TLP will require more bandwidth, decreasing the link efficiency. ACKNAK_latency_time ensures that Acknowledgement is sent for a group of TLPs, at a constant interval. This timer starts from 0 when an ACK is scheduled, ACK DLLP will be sent once this timer expires. Every time ACK or NAK is sent from B, this timer will be reset to 0.

- ☞ **REPLAY_NUM:** this counter is used to keep track of number of retries attempted by the DLL. If there are more retries, then it may raise issues within the link itself. When REPLAY_NUM rolls over from 11 to 00, then link is re-trained.
- ☞ **NEXT_RCV_SEQ:** Stores the packet sequence number expected on the receiver side.
- ☞ **ACKD_SEQ:** Stores the sequence number Acknowledged in the most recently received ACK or NAK DLLP.
- ☞ **NAK_SCHEDULER:** It is a flag used by DLL to mark NAK has been generated, and no further ACK/NAK are to be generated until former NAKs have been resolved. At the end of ACKNAK_latency_timer receiver sends ACK with the sequence number set as ACKD_SEQ, but when NAK_SCHEDULER is set then NAK is sent in place of ACK with sequence number set as ACKD_SEQ+1.
- ☞ **ACKNAK_SEQ_NUM:** keeps track of the sequence number of the last accurately received TLP.

If TLP has erroneous bits, then NAK will have ACKNAK_SEQ_NUM point to the last TLP correctly received.

- ☑ If Good TLP is received means incoming sequence number = NEXT_RCV_SEQ, then ACK is scheduled with

ACKNAK_SEQ_NUM = NEXT_RCV_SEQ

- ☑ If Duplicate TLP has received means incoming sequence number < NEXT_RCV_SEQ, then ACK is scheduled with

ACKNAK_SEQ_NUM = NEXT_RCV_SEQ - 1

If Bad TLP has received means incoming sequence number > NEXT_RCV_SEQ, then NAK is scheduled with

ACKNAK_SEQ_NUM = NEXT_RCV_SEQ - 1

ACK/NAK EXAMPLES

Scenario 1: ACK DLLP

Device A transmits TLPs with Sequence Numbers starting from 0, and reaches 6, 7, 8, 9, and 10.

- ☑ Device B successfully receives TLP6, and the NEXT_RCV_SEQ counter is incremented from 6 to 7. When ACK5 was sent to A, then the receiver ACKNAK_LATENCY_TIMER was reset, and the TLPs in retry buffer (of device A) till five is removed. Now, TLP6 has been received by the receiver but not Acknowledged, so ACKNAK_LATENCY_TIMER starts running. (This is the same as scheduling an ACK)

- ④ Before ACKNAK_LATENCY_TIMER expires, device B successfully receives TLP 7 and 8. Though receiving TLP7 and eight does not reset ACKNAK_LATENCY_TIMER.
- ④ When ACKNAK_LATENCY_TIMER expires, device B sends an ACK with TLP 8, the last good TLP received. ACKNAK_LATENCY_TIMER reset and put on hold until TLP 9 is received successfully.
- ④ Device A receives ACK with TLP 8; since forward progress is made, REPLAY_TIMER and REPLAY_NUM get reset. Device A purges all the TLP stored in the buffer with TLP 8 or less.
- ④ Now, device B receives TLPs 9 and 10, then ACKNAK_LATENCY_TIMER expires, and ACK is sent to device A, Which again reset REPLAY_TIMER and REPLAY_NUM and purges TLP 9 and 10 from the replay buffer.

Scenario 2: NAK DLLP

- ④ Device A transmits TLPs with Sequence Number 4095, 0, 1, 2, and 3.
- ④ Device B receives TLP 4095 successfully, and NEXT_RCV_SEQ is incremented from 4095 to 0 (counter roll-off) further ACKNAK_LATENCY_TIMER starts.
- ④ Device B detects a CRC error in TLP 0, NAK_SCHEDULED flag is set, which will send a NAK with sequence number 4095(NEXT_RCV_SEQ -1). Since NAK is generated, device B does not wait for ACKNAK_LATENCY_TIMER to expire. NAK will be sent immediately on the boundary of the next packet. Since NAK is sent so ACKNAK_LATENCY_TIMER reset to 0.
- ④ Device B will be looking for TLP 0. Since device A did not know about CRC fail of TLP 0, it had sent TLP 1,2,3, which device B will receive but not accept even if they are good TLPs (they pass the CRC check). This happens because TLPs should be accepted in order, and since TLP 1,2,3 is considered out of order, the receiver simply drops these without generating any NAK signal.

NOTE: Even if TLPs 1,2,3 fail CRC check, additional NAK will not be generated because the NAK_SCHEDULED flag is still set (TLP0), which will be cleared when device B will receive good TLP with sequence number 0.

- ④ Device A receives NAK 4095, all the TLPs with sequence number 4095 and less than that will be purged from the replay buffer. Since forward progress is made, it reset REPLAY_NUM and REPLAY_TIMER.
- ④ Device A will replay all remaining TLPs in the replay buffer (TLP 0,1,2,3) once again, REPLAY_TIMER restarts, and REPLAY_NUM will be incremented to 1.

NOTE: REPLAY_NUM is a 2-bit counter, it will count 00,01,10,11 and then rolls back to 00 when it rolls back, a fatal error is generated and the link will go into the inactive state and link retraining should be initiated.

- ✔ Device B receives TLP 0 successfully, NAK_SCHEDULED flag will be cleared, NEXT_RCV_SEQ will be incremented and ACKNAK_LATENCY_TIMER starts to count.

Scenario 3: Lost TLP

- ✔ The device A transmits TLPs starting with sequence number 0.
- ✔ Receiver has Sent ACK for TLPs till sequence number 15 and has set Next_SEQ_NUM as 16.
- ✔ TLP with sequence number 16 is lost during transmission and the receiver receives TLP with sequence number 17. Since Next_SEQ_NUM is not the same as the received TLP sequence number, the receiver schedules a NAK with seq number 15 and rejects the TLP with the sequence number 17.
- ✔ This allows the transmitter to understand that TLPs till 15 were received correctly and TLP with sequence number 16 was not exchanged.
- ✔ Transmitter sends a TLP with sequence number 16 and data exchange continues without problems.

DEFINITION AND ACRONYMS

- 👉 **DLCMSM (Data Link Control Management and state machine):** takes care of flow control initialization, and keeps track of link status while verifying all requirements for the link to remain functional.
- 👉 **Posted Transaction:** These are the transactions in which the requester does not receive any completion packet from the receiver side. Ex Memory Writes and Messages.
- 👉 **Non-Posted Transactions:** These are the transactions in which the requester receives a completion packet from the receiver side. Ex Memory Reads, Configuration Reads and Writes, and I/O Read and Writes.
- 👉 **Completions:** These transactions are made in reply to non-posted transactions. Ex Read and Write Completions.
- 👉 **VC (Virtual channel):** Buffers are used as queues for outgoing packets based on their priority of transmission. They are considered highly effective for maintaining data traffic on the channel since two different VCs function independently, thus mitigating the performance degradation caused by the slowest signal. Each port contains a maximum of 8 VCs numbered from zero to seven. Virtual Channel numbered as “zero”, i.e VC0 should mandatorily be activated by each port, and remaining VCs should be initialized based on further requirements.

☞ **LCRC (Link Cyclic Redundancy Code):** is calculated using data bytes, thus is unique to each packet. It helps DLL to check for data integrity of received packets. LCRC is only available to DLL, and transaction layer cannot access it.

☞ **ACK:** positive Acknowledgement received for Good TLP from receiver.

☞ **NAK:** negative Acknowledgement sent by receiver to transmitter for corrupted/lost TLPs.

☞ **Init:** Flow control initialization performed between Data Link Layer of both connected device.

REFERENCES

PCIe Gen5 Specification, NCB-PCI_Express_Base_5.0r1.0-2019-05-22

ABOUT US

At Logic Fruit, we specialize in development and Validation of high-quality real-time high throughput FPGA/SoC embedded solutions, and Developing Proof-of-concept (PoC) designs/prototypes with real-time data generation, acquisition and analysis.

Our engineers have expertise in many high speed protocols and interfaces, including 1G/10G/40G/100G Ethernet, PCIe(Gen1-Gen6), USB3.0/4.0, CPRI/ORAN, DisplayPort, ARINC818.

The team also has deep expertise in Signal processing for wireless and Imaging based solution development, software-defined radio (SDR), as well as encryption, protocol compliance, signal generation, data analysis, IoT technology, and multiple image processing techniques.

LFT AND PCIe:

We have worked on RTL IP and sub-system design for Latest generation of PCIe and other high speed serial protocols, and development of device drives. Verification is done using the latest methodologies like UVM and RTL-SW co-simulation. We also perform FPGA prototyping, validation, and testing with real DUTs. Development of reference hardware is done with different kind of PCIe devices and links.

Thank You!

Does anyone have any questions?

Contact Us



Gurugram (Headquarter)

806, 8th Floor
BPTP Park Centra Sector-30,
NH-8 Gurgaon - 122001
Haryana (India)

info@logic-fruit.com

+91-0124 4643950



Bengaluru (R&D House)

Sy. No 118, 3rd Floor,
Gayathri Lakefront,
Outer Ring Road, Hebbal,
Bangalore - 560 024

sales@logic-fruit.com

+91 80-69019700/01



United States (Sales Office)

Logic Fruit Technologies
INC 691 S Milpitas Blvd Ste
217 (Room 9) Milpitas CA
95035

info@logic-fruit.com

+1-408 338 9743