SIEMENS

Faculty of Engineering, Cairo University

Department of Electronics and Electrical Communications

DisplayPort Link Layer

AUX Services Block Interface Spec Document

Authors:

Mohamed Alaa Elden Atfat Mostafa

Mohamed Magdy Mabrouk Nada

Mohamed Hesham Hassan Tersawy

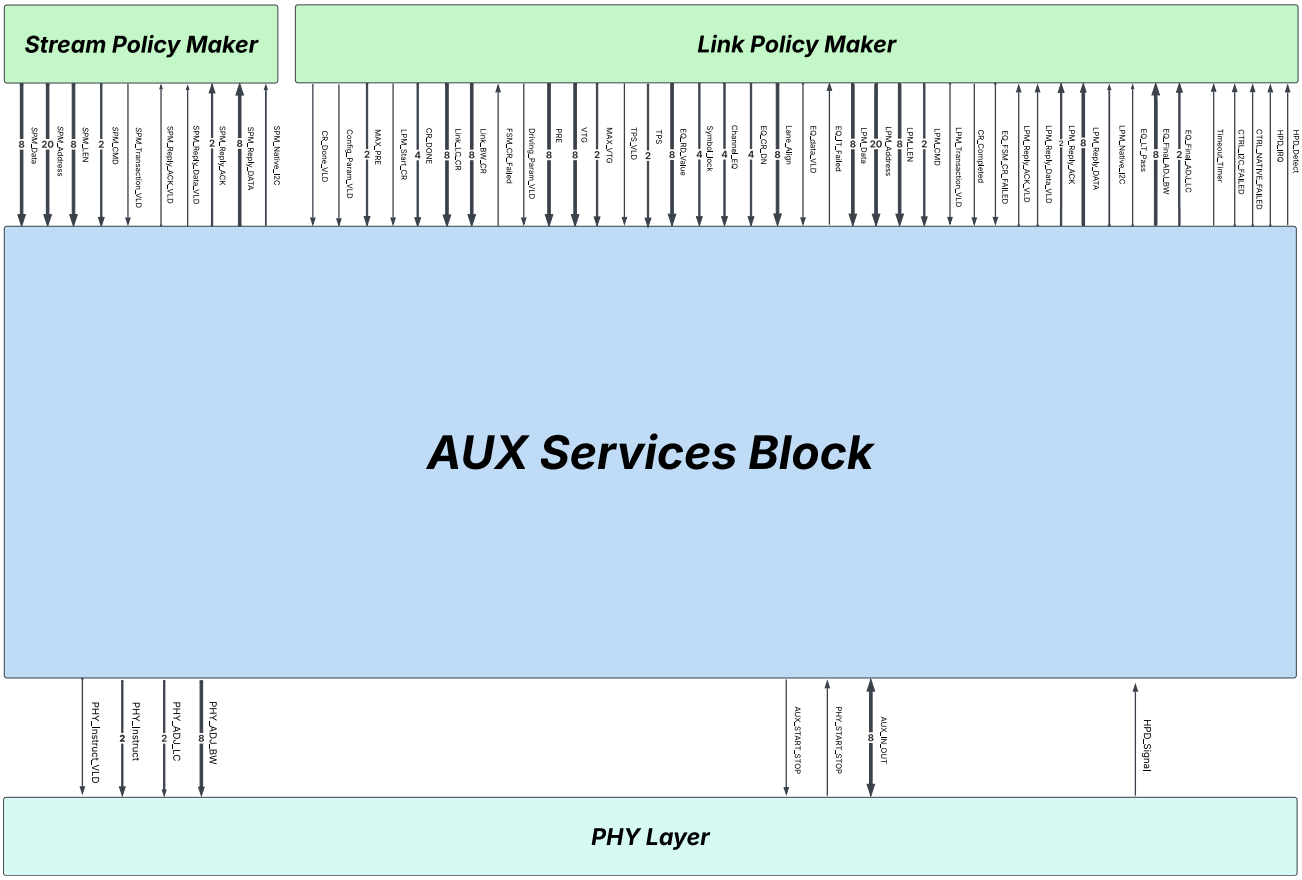
Supervised by:

Dr. Kareem Osama - Dr. Abdelrahman Abutaleb - Eng. Sara Nassef

Table of Contents:

System Overview

- High-Level Block Diagram:



- System Description:

The system is designed to perform the two primary AUX services: Device Services and Link Services. It supports two main transaction types: Native AUX Transactions and I2C-over-AUX Transactions. The I2C-over-AUX transaction is typically used for operations like reading the EDID, while the Native AUX transaction is used for reading or writing DPCD registers.

The process begins by detecting whether a sink device is connected to the source through the HPD signal. Once detected, the system reads the sink's characteristics by retrieving its EDID (128 bytes) using an I2C-over-AUX transaction. Next, it reads the sink's capabilities from the DPCD registers (addresses 0x00000 to 0x000FF) through a Native AUX read transaction. After gathering this information, the system initiates the link training process using the optimal parameters obtained from the sink's capabilities. If the training process fails, the system adjusts specific parameters and retries the training until it successfully establishes a stable connection. This adjustment process involves multiple Native AUX read and write transactions. Once link training is completed successfully, video transmission begins via the Isochronous Transport Services block. Meanwhile, the AUX Services system continuously monitors for HPD_IRQ events, which indicate that the sink is requesting retraining due to performance degradation or any event happened in the sink device.

- Interface Signal Description:

Signals	Interface	Type	Width *(Bits)*	Description
SPM_Reply_Data	*Stream Policy Maker*	*Output*	8	I2C-over-AUX Reply Transaction (Data Part) for I2C-over-AUX Request Read Transaction, using I2C Transaction Method 1 (transfer 1 byte per transaction)
SPM_Reply_ACK			2	This signal represents the status of the reply transaction whether ACK, NACK or DEFER.
SPM_Reply_ACK_VLD			1	A valid signal indicating that the status of the reply transaction is ready.
SPM_Reply_Data_VLD			1	A valid signal indicating that the data is ready (set to one when receiving data from an I2C-over-AUX reply transaction).
SPM_Native_I2C			1	Signal representing the most significant bit in the command field to determine the type of transaction whether I2C or Native
SPM_Data		*Input*	8	Data written by Stream Policy Maker when initiating a write transaction (byte-by-byte)
SPM_Address			20	Address sent by Stream Policy Maker to indicate register to be written to or read from when initiating a transaction
SPM_LEN			8	Length of data, in bytes, send by Stream Policy Maker to be written or read for request transaction
SPM_CMD			2	The command field sent to specify the transaction type (Read or Write).
SPM_Transaction_VLD			1	A valid signal activated by the Stream Policy Maker during a request transaction.
HPD_Detect	*Link Policy Maker*	*Output*	1	The signal is activated when the HPD (Hot Plug Detection) signal remains asserted for a period exceeding 2ms, which indicates that a sink device has been successfully connected.
HPD_IRQ			1	The signal is active when the HPD (Hot Plug Detection) signal is set to low for a duration of 0.5ms to 1ms before returning to high, indicating an interrupt request from the sink.
LPM_Reply_Data			8	Native Reply Transaction (Data Part) for a Native Request Read Transaction.
LPM_Reply_Data_VLD			1	A valid signal indicating that the data is ready (set to one when receiving data from a Native reply transaction).
LPM_Reply_ACK			2	Native Reply Transaction (Command Part) for a Native Request Transaction.
LPM_Reply_ACK_VLD			1	A valid signal indicating that the acknowledge data is ready (set to one when receiving reply acknowledge data from a Native reply transaction).
LPM_Native_I2C			1	Signal representing the most significant bit in the command field to determine the type of transaction whether I2C or Native

FSM_CR_Failed	*Link Policy Maker*	*Output*	1	A Signal indicating the failure of the Clock Recovery phase during link training, meaning the sink failed to acquire the clock frequency during the training process
EQ_LT_Failed			1	Signal indicating the failure of the Channel Equalization phase during link training.
EQ_LT_Pass			1	This signal represents a successful channel equalization phase which indicates successful link training process
EQ_Final_ADJ_BW			8	The adjusted link BW after successful link training used for sending main video stream.
EQ_Final_ADJ_LC			2	The adjusted number of lanes after successful link training, used for sending main video stream.
Timer_Tiomout			1	This signal is asserted if the transaction is delayed long enough to cause a timeout while waiting for a reply.
CTRL_Native_Failed			1	This signal is asserted if the reply to the Native Request transaction is deferred seven consecutive times.
CTRL_I2C_Failed			1	This signal is asserted if the reply to the I2C Request transaction is deferred seven consecutive times.
CR_Completed			1	This signal is asserted once the Clock Recovery Phase of link training has successfully completed.
EQ_FSM_CR_Failed			1	This signal is asserted when the Channel Equalization Phase of link training fails because the CR_Done bits were not all ones, requiring the Clock Recovery Phase to restart.
LPM_Start_CR		*Input*	1	After reading the DBCD capability register of the sink, the Link Policy Maker starts link training by asserting this signal, which marks the beginning of the Clock Recovery phase.
CR_Done			4	Four bits indicating whether clock recovery has been completed for the four lanes (with the least significant bit representing Lane 0). These bits are read from registers 00202h and 00203h in the DBCD registers of the sink device.
CR_Done_VLD			1	This signal is a valid indicator for the CR_Done bus, it is asserted when the bits become valid on the CR_Done bus.
LPM_Link_LC			2	The maximum number of lanes that the sink can support, based on its capability, which are specified in the sink DPCD capability registers.
LPM_Link_BW			8	Maximum Bandwidth that the sink can support, based on its capability, which are specified in the sink DPCD capability registers.
PRE			8	The minimum pre-emphasis level, which is either retrieved from the sink DPCD capability registers at the beginning of the link training process or adjusted during the link training

VTG			8	The minimum voltage swing level, which is either retrieved from the sink DPCD capability registers at the beginning of the link training process or adjusted during the link training
Driving_Param_VLD			1	Flag signal asserted by link policy maker indicating the arrival of voltage swing and pre-emphasis data.
Config_Param_VLD			1	Flag signal asserted by link policy maker indicating the arrival of Link BW, Lane Count, Max VTG, Max PRE, and EQ_RD_Value for the CTR.
EQ_RD_Value			8	The time the source waits during the link training process before checking the status update registers, which are read from the Capability Register (0000Eh).
EQ_CR_DN			4	Clock recovery done bits that read from register 202h-203h and sent to link layer via link policy maker through this signal, if all ones indicating the successful clock recovery from different lanes
Channel_EQ	*Link Policy Maker*	*Input*	4	Channel Equalization bits read from register 202h-203h and sent to link layer via Link Policy Maker through this signal, if all ones indicating the successful channel equalization of different lanes
Symbol_Lock			4	Same concept as the both signals EQ_CR_DN and Channel_EQ
Lane_Align			8	The result of the Equalization phase during link training for the four lanes, found in the Status Update registers (00204h), indicating whether Lane Align achieved.
EQ_Data_VLD			1	Valid signal indicating the arrival of these data from link policy maker
TPS			2	This signal indicates the maximum supported TPS Pattern Sequence
TPS_VLD			1	This signal is a valid indicator for the TPS bus, it is asserted when the bits become valid on the TPS bus, Max VTG, Max PRE.
MAX_VTG			2	Maximum level of voltage swing supported by the sink capability (Level 2 or 3)
MAX_PRE			2	Maximum level of pre-emphasis supported by the sink capability (Level 2 or 3)
LPM_Data			8	Data written by Link Policy Maker when initiating a write transaction
LPM_Address			20	Address sent by Link Policy Maker to indicate register to be written to or read from when initiating a transaction
LPM_LEN			8	Length of data, in bytes, send by Link Policy Maker to be written or read for request transaction
LPM_CMD			2	The command field sent to specify the transaction type (Read or Write).
LPM_Transaction_VLD			1	A valid signal activated by the Stream Policy Maker during a request transaction.

AUX_IN_OUT	*PHY Layer*	*Inout*	8	A request/reply transaction where each byte is transmitted or received during every individual clock cycle, ensuring that the data transfer occurs sequentially, one byte at a time, ensuring a continuous, byte-by-byte data exchange.
PHY_START_STOP		*Input*	1	A single bit indicating the beginning and end of the reply transaction. It is set to one during the reception of the transaction and set to zero when no transaction is occurring.
AUX_START_STOP		*Output*	1	A single bit indicating the beginning and end of the request transaction. It is set to one during the transmission of the transaction and set to zero when no transaction is occurring.
PHY_Instruct		*Output*	2	A signal sent by CR FSM block which instructs the physical layer to begin sending a specific link training pattern (TPS1, 2, 3, 4) during the link training process.
PHY_Instruct_VLD			1	This signal is a valid indicator for the PHY_Instruct bus, it is asserted when the bits become valid on the PHY_Instruct bus.
PHY_ADJ_BW			8	Maximum Bandwidth that the sink can support, based on its capability, which are specified in the sink DPCD capability registers in the beginning of the link training or the adjusted value of the BW during the link training.
PHY_ADJ_LC			2	Maximum Lane Count that the sink can support, based on its capability, which are specified in the sink's DPCD capability registers in the beginning of the link training or the adjusted value of the Lane Count during the link training.
HPD_Signal		*Input*	1	The HPD signal indicates the connection status based on its duration, If set to one for more than 2ms, it indicates the start of a connection, If set to zero for more than 2ms, it indicates the end of a connection, If it toggles between high and low for 0.5 to 1ms it represents interrupt request IRQ

- Flow of Operation:

When a sink device is connected, the **Link Policy Maker** detects it through the ***HPD_Detect*** signal. The first step is to gather information about the sink's characteristics by reading its **EDID**.

To do this, the **Link Policy Maker** informs the **Stream Policy Maker** of the detected sink. In response, the **Stream Policy Maker** initiates an **I2C-over-AUX read request transaction** by activating the ***SPM_Transaction_Valid*** signal. The address for the read request is provided via the ***SPM_Address*** signal, while the command type, indicating a read operation, is sent through ***SPM_CMD.***

Once the transaction is initiated, the **AUX Control Unit** enters the **Talk Mode** state to transmit the data. It asserts the ***TR_VLD*** signal to indicate that a request is initiated. The data is then passed to the **DE-MUX**, which determines whether it should be forwarded to the **I2C Message Encoder** or the **Native Encoder**, based on the ***AUX_CTRL_I2C_Native*** signal. Since this is an I2C-over-AUX transaction, the DE-MUX directs the data to the **I2C Message Encoder** while also signaling the **I2C FSM** through the ***I2C_TR_VLD*** signal, prompting it to begin the I2C-over-AUX process.

The **I2C-over-AUX transaction** consists of three key phases:

1. **Address-Only Transaction:** The first transaction sends the address.
2. **Data Transfer:** After receiving an acknowledgment (ACK), the actual read request is sent, retrieving one byte per transaction (following **Method 1** of I2C-over-AUX).
3. **Completion Transaction:** Once all required bytes are received, a final transaction is sent to complete the process.

The generated transaction is then passed to the **Bi-Directional AUX PHY Interface**, which relays it to the **PHY Layer** while asserting the ***START_STOP*** signal to indicate the beginning of the transfer. A **Timeout Timer** starts counting simultaneously—if no response is received within **400 µs**, the transaction must be retransmitted, so the timer asserts the ***Timeout*** signal. However, if a response arrives in time, the ***Timer_RESET*** signal resets the timer.

Once the sink device responds, the **Reply Decoder** extracts the ***Reply_ACK*** field to determine whether the response is an **ACK (acknowledgment), NACK (negative acknowledgment), or DEFER (delay request).**

- If an **ACK** is received, the **I2C FSM** instructs the **I2C Message Encoder** to proceed with the next transaction.
- If a **NACK** or **DEFER** is received, the transaction is resent.
- For **Native transactions**, if the **AUX CTRL Unit** receives an **ACK**, it returns to its idle state. If a **NACK** or **DEFER** is received, the request is retransmitted via the ***Native_Retrans*** signal.
- If the ***Timeout*** signal is asserted, it functions similarly to a **NACK** or **DEFER**, requiring retransmission regardless of whether the transaction is **I2C-over-AUX** or **Native**.

Once the EDID read is successfully completed, the **I2C FSM** asserts the ***I2C_Complete*** signal, indicating that the **LPM** can proceed to the next step—reading the **Sink Capabilities** stored in the **DPCD register (addresses 0x00000 – 0x000FF).** Since the maximum burst size for a single transaction is **16 bytes**, multiple native transactions are required to read the sink capabilities. Unlike I2C-over-AUX, native transactions complete in a single step without requiring an FSM.

After reading the sink's capabilities, the **Link Policy Maker** moves on to the **Link Training Process**, which ensures proper signal integrity between the source and sink. This process consists of two main phases *(Descripted later in details):*

1. **Clock Recovery Phase:** Handled by the **CR FSM** and **CR_ERR_CHK** blocks.
2. **Channel Equalization Phase:** Managed by the **EQ FSM** and **EQ_ERR_CHK** blocks.

Upon successful completion of link training, the **EQ FSM** asserts ***EQ_LT_Pass*** and provides the final number of lanes and bandwidth achieved via ***EQ_Final_ADJ_LC*** and ***EQ_Final_ADJ_BW*** signals. At this stage, video transmission begins.

While video is being transmitted, the ***HPD_IRQ*** signal may be asserted, indicating either a connection degradation that requires retraining or another event on the sink side. In response, the **Link Policy Maker** performs a read transaction to determine the cause and takes corrective actions accordingly.

System Scenarios

1- Connection Detection:

1. The DisplayPort Transmitter (DPTX) identifies a connection to the DisplayPort Receiver (DPRX) when the ***HPD_Signal*** remains high for **more than 2ms**.
2. Once detected, the ***HPD_Detect*** signal is asserted high and sent to the **Link Policy Maker**.

2- I2C-over-AUX transaction

The I2C-over-AUX transaction typically involves multiple transactions and can be carried out using two methods. Method 1 transmits only one byte of data per transaction, while the other method allows multiple bytes to be sent within a single transaction. ***This system utilizes Method 1.***

The process begins with an **Address-only** transaction, where the address is sent without any accompanying data. Once the address is transmitted, the system waits for an ACK (Acknowledge) response before proceeding. Each transaction should be forward by ACK.

EDID Read at I2C address of 1001000

1. The Stream Policy Maker initiates an I2C-over-AUX Read transaction to access the EDID registers in the sink device after detecting a connection.
2. The Stream Policy Maker configures ***SPM_CMD*** as **01** for read, assigns the EDID address to ***SPM_Address***, and assign ***SPM_LEN*** with (00000000) as system utilizes method 1 with always 1 byte length, leaves ***SPM_Data*** as don't care, and asserts ***SPM_Transaction_Valid*** high to validate the transaction.
3. Once encoded, the ***AUX_IN_OUT*** bus transmits an **Address-only** I2C-over-AUX transaction (0101|0000 -> 00000000 -> 01001000) while ***START_STOP*** is high. This signifies an I2C read request with MOT = 1 and the target I2C address 1001000.
4. The DisplayPort Transmitter (DPTX) waits for an acknowledgment (ACK) from DPRX. A response of 0000|0000 confirms (I2C ACK | AUX ACK) while ***START_STOP*** is high while ack reception.
5. Upon receiving ACK, DPTX sends (0101|0000 -> 00000000 -> 0|1001000 -> 0000|0000) on the ***AUX_IN_OUT*** bus while ***START_STOP*** remains high while sending valid messages on the ***AUX_IN_OUT*** bus. This represents an I2C read request with MOT = 1, the same I2C address, and a read length of **1 byte**.
6. DPTX waits for an acknowledgment and the first data byte, expected as (0000|0000 -> Data0), where 0000|0000 confirms (I2C ACK | AUX ACK) and Data0 is the received byte. while ***START_STOP*** is high.
7. The Stream Policy Maker receives the data on the ***SPM_Reply_DATA*** bus when ***SPM_Reply_Data_VLD*** is high and receives if it **ACK, NACK** on the ***SPM_Reply_ACK*** bus when ***SPM_Reply_ACK_VLD*** is high and an indicator for that is an I2C reception not a native one by receiving 0 (I2C) on ***SPM_Native_I2C***.
8. Steps 5 to 7 are repeated **127 more times** to retrieve the entire **128-byte EDID memory**.
9. Once the full EDID data is received, DPTX sends (0001|0000 -> 00000000 -> 01001000) on ***AUX_IN_OUT*** with ***START_STOP*** high, indicating an **Address-only** I2C read with MOT = 0, signaling an **I2C STOP** to DPRX.
10. In response, DPRX sends (0000|0000), confirming (I2C ACK | AUX ACK) while **START_STOP** is high which mean the end of the whole EDID read transactions.

3- Native AUX Transaction

The Native AUX transaction consists of a single transaction, unlike the I2C-over-AUX transaction, which involves multiple transactions. This transaction has two types:

1. **Request Transaction** – Sent from the source to the sink.
2. **Reply Transaction** – Sent from the sink to the source in response to a request.

Each of these transactions can be either a **read** or **write** transaction.

Read Request Transaction:

1. The Link Policy Maker asserts the ***LPM_Transaction_VLD*** signal and sends the address, command (1001b), and the expected data length through ***LPM_Address, LPM_CMD***, and ***LPM_LEN*** to the AUX Services Block.
    - The source expects the sink to reply with (***LPM_LEN*** **+ 1)** bytes of data (ranging from 1 to 16 bytes).
2. The AUX Services Block transmits the request in the following order: command, address, and length, sending each byte sequentially through ***AUX_IN_OUT*** to the PHY Layer. The ***START_STOP*** signal is asserted during transmission.

Reply to Read Request Transaction:

1. The PHY Layer transmits the reply byte-by-byte through ***AUX_IN_OUT***, asserting ***START_STOP*** during the transmission.
    - The response includes a **command field** that can be **ACK (Acknowledge), NACK (Negative Acknowledge), or DEFER**.
    - If **ACK** is received, the sink sends the requested data (either fully or partially).
2. The AUX Services Block captures the command field and received data in case of an ACK and asserts the valid signal through ***LPM_Reply_ACK_VLD*** and ***LPM_Reply_Data_VLD*** signals and report the command and data via ***LPM_Reply_ACK***, ***LPM_Reply_DATA,*** and set the ***LPM_Native_I2C*** signal to indicate the type of receiving transaction.

Write Request Transaction:

1. The Link Policy Maker asserts the ***LPM_Transaction_VLD*** signal and sends the address, command (1000b), expected data length, and the data to be written through ***LPM_Address, LPM_CMD***, ***LPM_LEN***, and ***LPM_Data*** to the AUX Services Block.
2. The AUX Services Block transmits the request in the following order: command, address, length, and data, sending each byte sequentially through ***AUX_IN_OUT*** to the PHY Layer. The ***START_STOP*** signal is asserted during transmission.

Reply to Write Request Transaction:

1. The PHY Layer transmits the reply byte-by-byte through ***AUX_IN_OUT***, asserting ***START_STOP*** during the transmission.
    - The response includes a **command field** that can be **ACK, NACK, or DEFER**.
    - If **NACK** is received, the sink must indicate the number of successfully written bytes.

2. The AUX Services Block reports the command field and data (in case of NACK) through ***LPM_Reply_ACK_VLD***, ***LPM_Reply_Data_VLD***, ***LPM_Reply_ACK***, and ***LPM_Reply_DATA,*** and set the ***LPM_Native_I2C*** signal to indicate the type of receiving transaction.

4- Display Port Link Training Flow:

When a Hot Plug event is detected, the DisplayPort source initiates link training to configure the connection. The source device communicates with the sink device via the AUX channel to access the sink's DPCD register block, determining its capabilities and status before starting the Link Training process. Below is the sequence of steps after HPD assertion:

1. The source's link policy maker initiates a read transaction to retrieve the sink's DPCD capabilities (offsets 0x00000–0x000FF) includes the **TRAINING_AUX_RD_INTERVAL** value from offset 0x0000E.
2. The source link policy maker asserts the ***LPM_Start_CR*** signal to begin the clock recovery sequence. It writes to the Link Configuration field (offsets 0x00100–0x00101) to set the Link Bandwidth and Lane Count these values are send to link layer via ***Link_BW_CR***, ***Link_LC_CR*** based on the sink's capabilities. The source also clears register 0x00102 to 00h in the same write transaction.

After configuring the link, the source starts **Link Training Clock Recovery Phase:**

1. The source writes to offset 0x00102 to enable Training Pattern 1 and disable scrambling. Simultaneously, it instructs the PHY layer to transmit TPS1 via the ***CR_PHY_instruct*** signal which indicates the training pattern sequence number, sending the adjusted bandwidth and lane counts to physical layer via ***CR_ADJ_BW*** and ***CR_ADJ_LC*** (PHY layer uses these values of link bandwidth and lane count to send the training patterns) **.**
2. The source writes the initial driving settings (minimum voltage and 0 dB pre-emphasis) to offsets 0x00103–0x00106 to configure Link Training Control for each lane.
3. The source reads the **TRAINING_AUX_RD_INTERVAL** value from offset 0x0000E, which indicates the wait time required for the sink to update its status registers. This value is passed to the link layer via the ***EQ_ED_Value*** signal.
4. The source waits for the specified interval before reading the Link Status registers (0x00202–0x00207).
5. The link policy maker extracts the clock recovery status (CR_DONE) from registers 0x00202 and 0x00203 and passes it to the link layer via the ***CR_Done*** signal.
6. If clock recovery (CR_DONE) fails in any lane:
    - The source reads the Link Driver setting adjust request registers (0x00206–0x00207) and updates the settings via the ***PRE*** and ***VTG*** signals.
    - If Training Pattern Sequence 1 has been repeated five times for the same voltage and pre-emphasis or the sink reaches its maximum capable voltage swing passes to it by link policy maker after reading sink capabilities via **MAX_VTG** or it repeats this loop 10 times (these are three checks performed each loop), the source reduces the Link Bandwidth (e.g., from HBR2 to HBR to RBR) and restarts the process.
    - If the bandwidth is already at the lowest rate (RBR), the source reduces the number of lanes (from 4 to 2 to 1).
    - If only one lane remains and recovery fails, link training is aborted by asserting ***FSM_CR_Failed.***

For **Link Training Channel Equalization Phase:**

1. The source writes to offset 0x00102 to enable Training Pattern 2 and instruct the phy layer the same way as mentioned in clock recovery phase and disable scrambling. It also configures Link Training Control for each lane (offsets 0x00103–0x00106).
2. The source reads the **TRAINING_AUX_RD_INTERVAL** value from offset 0x0000E and passes it to the link layer via ***EQ_ED_Value***.
3. After waiting, the source reads the Link Status registers (0x00202–0x00207) and passes the results (CR_DONE, SYMBOL_LOCKED, and CHANNEL_EQ_DONE) to the link layer via **Lane_Allign** , **Channel_EQ** , **Symbol_Locked** , **EQ_CR_DN** .
4. If CR_DONE fails in any lane, the source reduces the lane count and adjust for max bandwidth, aborts Sequence 2, and restarts Sequence 1.
5. If CR_DONE fails in all lanes, the source reduces the bandwidth and adjust for max no. of lanes supported by sink capability, aborts Sequence 2, and restarts Sequence 1.
6. If CR_DONE passes, the source checks for:
    - CHANNEL_EQ_DONE (via **Channel_EQ**)
    - SYMBOL_LOCKED (via **Symbol_Locked**)
    - INTERLANE_ALIGN_DONE (via **Lane_Allign**).
7. If any of these parameters fail:
    - The source adjusts the Link Driver settings based on registers 0x00206–0x00207.
    - If Sequence 2 has been repeated five times, the source reduces the lane count, aborts Sequence 2, and restarts Sequence 1.
    - If only one lane remains and the lowest bandwidth (RBR) is reached, link training fails, and ***EQ_Failed*** is asserted.
8. If Sequence 2 succeeds, link training is complete then asserting the ***EQ_LT_Pass*** to the link policy maker to indicate successful link training then passing the adjusted bandwidth and No. of lanes to the link policy maker to use them in sending the main video stream via ***EQ_Final_ADJ_LC*** and ***EQ_Final_ADJ_BW.***
9. The source writes 00h to offset 0x00102 to disable Link Training.

Note: If both source and sink support HBR2, Training Pattern Sequence 3 replaces Sequence 2. **Refer to link training flowcharts for further details.**

In the **LANEx_CR_DONE sequence**, the transmitter must wait at least 100 µs before reading the **LANEx_CR_DONE** bits from the following registers:

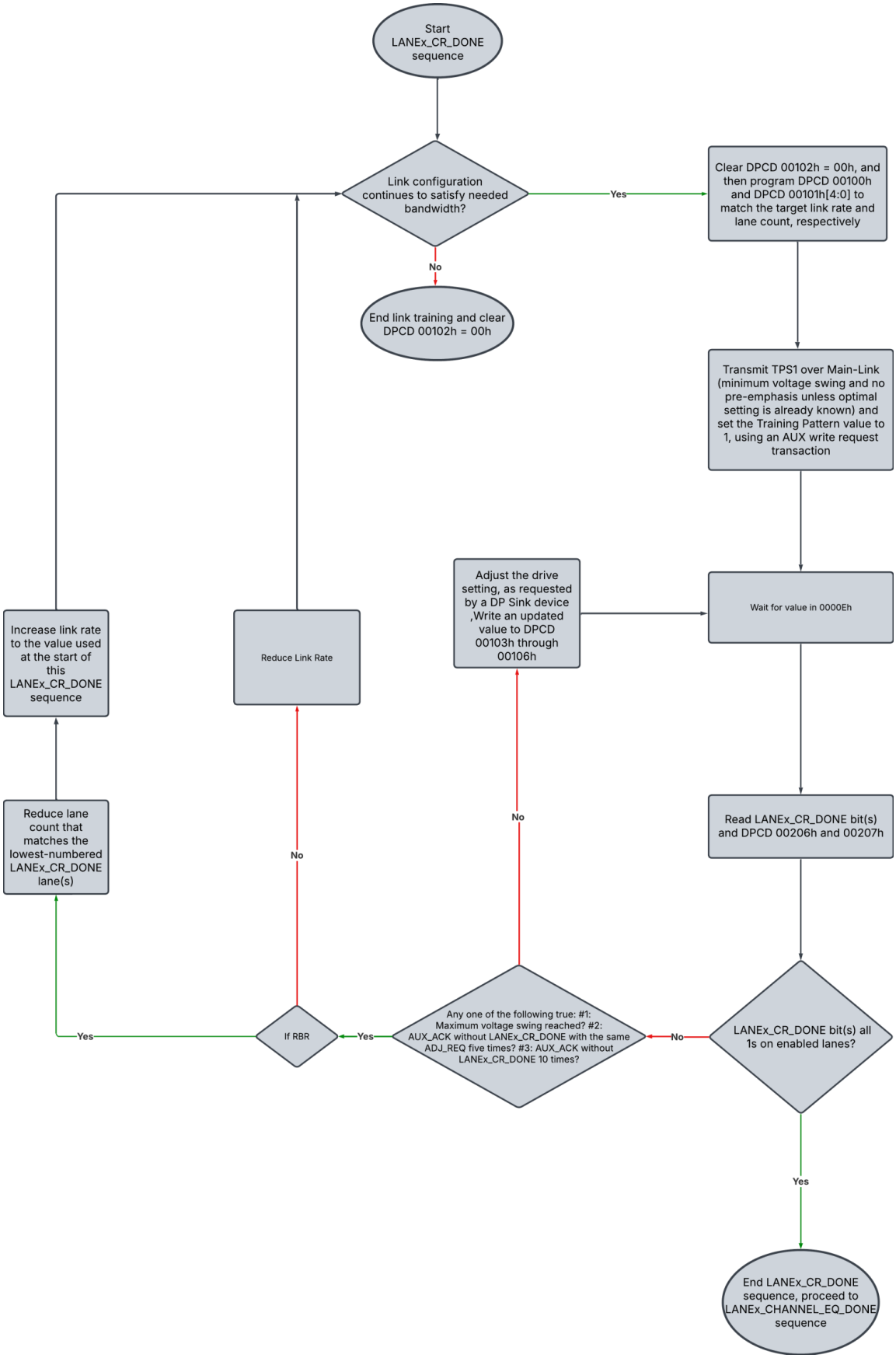
- LANEx_y_STATUS (DPCD 00202h [0 and 4] and DPCD 00203h [0 and 4])
- LANEx_y_STATUS_ESI (DPCD 0200Ch [0 and 4] and DPCD 0200Dh [0 and 4]).

If any LANEx_CR_DONE bits are not set, the transmitter must:

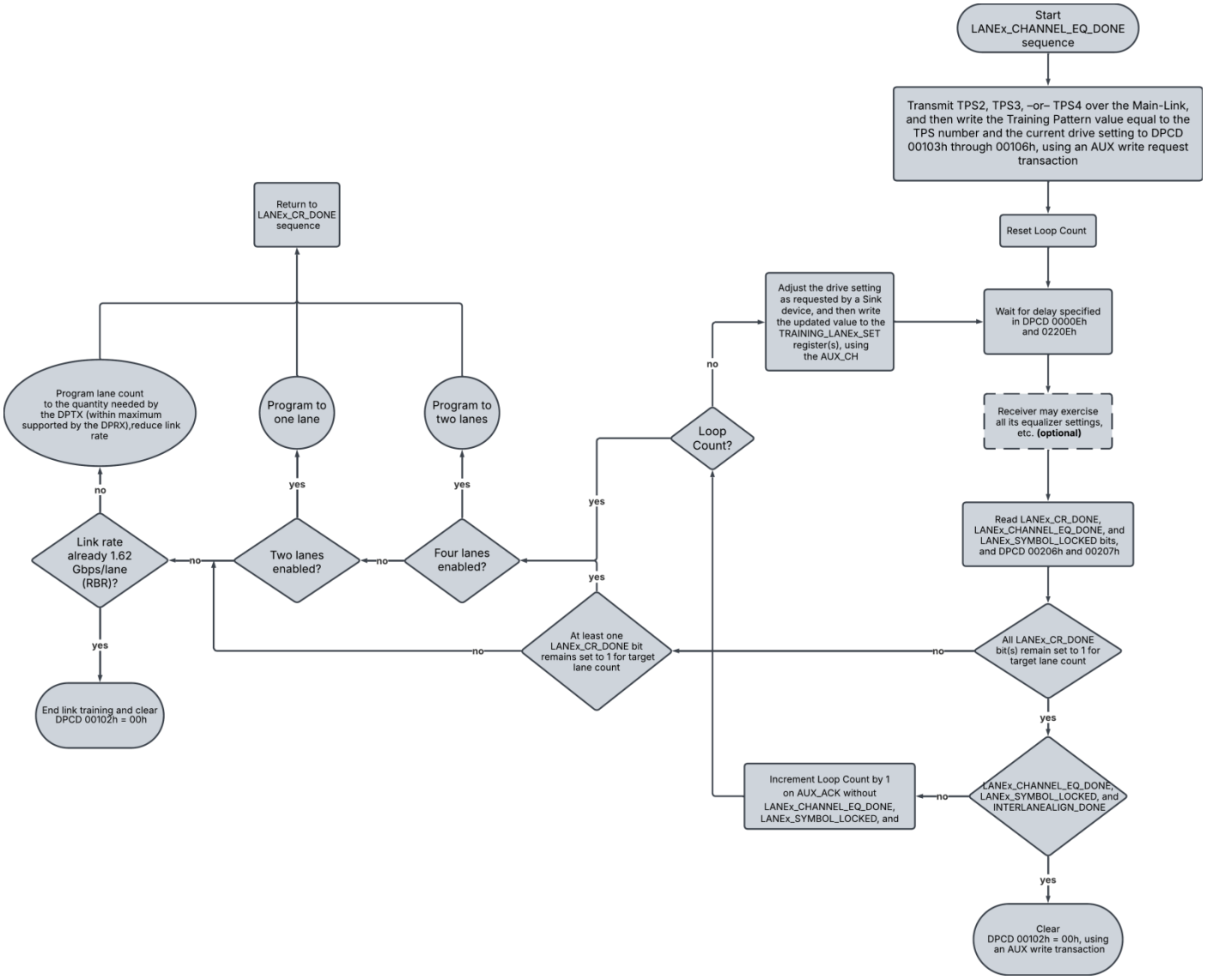
1. Read the ADJUST_REQUEST_LANEx_y registers (DPCD 00206h and 00207h).
2. Adjust the voltage swing and/or pre-emphasis levels as requested.
3. Update DPCD 00103h–00106h with the new settings.

Flow charts of the training process:

- *Clock Recovery:*



- ***Channel Equalization***



5- Interrupt Request:

1. DPTX recognizes an interrupt request from DPRX if the ***HPD_Signal*** **de-asserted for a duration between 0.5ms and 1ms and then asserted again**.
2. Upon detection, the ***HPD_IRQ*** signal is asserted high for **one CLK cycle** and received by the **Link Policy Maker.**

AUX Clock Calculation

In DisplayPort, the AUX channel operates at a standard **1 MHz** bit rate at the PHY-to-PHY serial link. Since the PHY layer employs **8b/10b encoding**, every 8 bits of data are mapped to 10 transmitted bits. As a result, the effective data rate at the Link Layer is **100 kHz** (1 MHz ÷ 10). By using a pipelined architecture where each internal block processes data in a single cycle, the Link Layer clock should also run at **100 kHz** to ensure synchronized data processing.