Remote Fault Handling

12 March 2025

Problem Statement

- Assuming Murray_3dg_01_03122025 option B adopted
- Murray_3dg_01_03122025 slide 11 shows a linkup sequence of

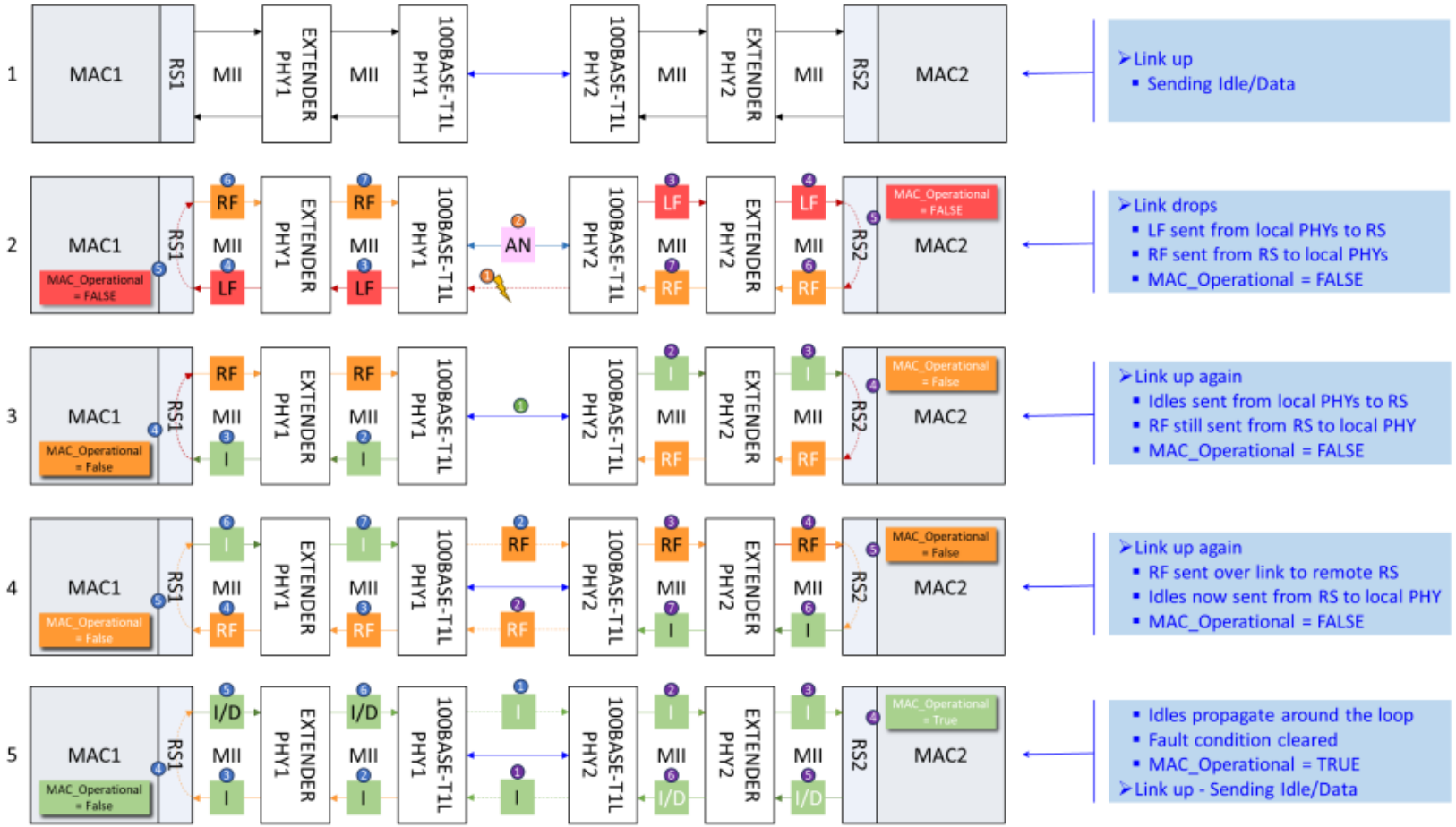
  Local Fault ➔ Idle ➔ Remote Fault ➔ Idle
- For XGMII the linkup sequence is a bit cleaner

  Local Fault ➔ Remote Fault ➔ Idle
- This presentation makes some suggestions to alter Murray_3dg_02_03122025 to make MII behave a little cleaner

  Local Fault ➔ Remote Fault ➔ Idle

Murray_3dg_01_03122025 Slide 11

- Local Fault ➔ Idle ➔ Remote Fault ➔ Idle



How Reconciliation Sublayer Handle Local and Remote Faults

- See Local Fault on RXD ➔ Sent Remote Fault on TXD
- See Remote Fault on RXD ➔ Sent Idles on TXD
- Else TXD free to transmit data

How XGMII Works

- PHY A RXD (LF) → PHY A TXD (RF) → (not linked up) → PHY B RXD (LF)
  PHY A RXD (LF) ← (not linked up) ← PHY B TXD (RF) ← PHY B RXD (LF)

- Assume PHY A links up first
- PHY A RXD (RF) → PHY A TXD (Idle) → (not linked up) → PHY B RXD (LF)
  PHY A RXD (RF) ← (linked up) ← PHY B TXD (RF) ← PHY B RXD (LF)

- PHY A RXD (Idle) → PHY A TXD (Idle) → (linked up) → PHY B RXD (Idle)
  PHY A RXD (Idle) ← (linked up) ← PHY B TXD (Idle) ← PHY B RXD (Idle)

Either LF → RF → Idle or LF → Idle. RF is an indicator that link partner is not ready.

How 100BASE-T1L Works

- Does not rely on directly passing remote fault sequence ordered set to indicate link partner is not ready.

- Uses loc_phy_ready = NOT_OK or OK in the block code instead of a sequence ordered set

Proposed Modifications

- No change to the encoding to Murray_3dg_01_03122025 slide 10.

Table 22–1—Permissible encodings of TXD<3:0>, TX_EN, and TX_ER

TX_EN	TX_ER	TXD<3:0>	Indication
0	0	0000 through 1111	Normal inter-frame
0	1	0000	Reserved
0	1	0001	Assert LPI
0	1	0010	PLCA BEACON request
0	1	0011	PLCA COMMIT request
0	1	0100	Assert remote fault
0	1	0101 through 1111	Reserved
1	0	0000 through 1111	Normal data transmission
1	1	0000 through 1111	Transmit error propagation

Table 22–2—Permissible encodings of RXD<3:0>, RX_ER, and RX_DV

RX_DV	RX_ER	RXD<3:0>	Indication
0	0	0000 through 1111	Normal inter-frame
0	1	0000	Normal inter-frame
0	1	0001	Assert LPI
0	1	0010	PLCA BEACON indication
0	1	0011	PLCA COMMIT indication
0	1	0100	Assert remote fault
0	1	0101	Assert local fault
0	1	0110 through 1101	Reserved
0	1	1110	False Carrier indication
0	1	1111	Reserved
1	0	0000 through 1111	Normal data reception
1	1	0000 through 1111	Data reception with errors

Proposed Modifications

- Change the encoding to Murray_3dg_02_03122025 slide 10.
- NOT_RDY map to RF and not to Idle.

  (i.e. tx_enable = 0, tx_error = 1, TXD = 0100)

Table 199–1— MII transfer categories

Category	tx_enable	tx_error	TXD<3:0>	Description
	loc_phy_ready = FALSE			
NOT_RDY	–	–	–	PHY not ready for MII transfer
	loc_phy_ready = TRUE			
DAT	1	0	–	Normal data transmission
ERR	1	1	–	Transmit error propagation
NIF	0	0	–	Normal inter-frame
ALPI	0	1	0001	Assert LPI
ARF	0	1	0100	Assert remote fault
IDL	0	–	–	

Proposed Optional Modification

- /Q/ Set to reserved
- /Ix/ Normal Inter-Frame, loc_phy_ready = NOT_OK or assert remote fault
- No more control symbols left
- Save /Q/ for future development

Table 199–3— TOCT symbol to TOCT value mapping

TOCT Symbol	Definition	TOCT Value
/Q/	Assert remote fault	0x00
/E/	Transmit Error Propagation	0x10
/I/	Normal Inter-Frame, loc_phy_ready = OK	0x08
/Su/	Start of packet on odd nibble	0x18
/Tp/	End of packet after odd nibble	0x04
/L/	Assert LPI	0x14
/Ix/	Normal Inter-Frame, loc_phy_ready = NOT_OK	0x0C
/Sp/	Start of packet on even nibble	0x1C
/Tu0/	End of packet after even nibble, last data nibble = 0x0	0x01
/Tu8/	End of packet after even nibble, last data nibble = 0x8	0x11
/Tu4/	End of packet after even nibble, last data nibble = 0x4	0x09
/TuC/	End of packet after even nibble, last data nibble = 0xC	0x19
/Tu2/	End of packet after even nibble, last data nibble = 0x2	0x05
/TuA/	End of packet after even nibble, last data nibble = 0xA	0x15
/Tu6/	End of packet after even nibble, last data nibble = 0x6	0x0D
/TuE/	End of packet after even nibble, last data nibble = 0xE	0x1D
/Tu1/	End of packet after even nibble, last data nibble = 0x1	0x03
/Tu9/	End of packet after even nibble, last data nibble = 0x9	0x13
/Tu5/	End of packet after even nibble, last data nibble = 0x5	0x0B
/TuD/	End of packet after even nibble, last data nibble = 0xD	0x1B
/Tu3/	End of packet after even nibble, last data nibble = 0x3	0x07
/TuB/	End of packet after even nibble, last data nibble = 0xB	0x17
/Tu7/	End of packet after even nibble, last data nibble = 0x7	0x0F
/TuF/	End of packet after even nibble, last data nibble = 0xF	0x1F

How MII Works with Modification

- PHY A RXD (LF) / PHY A TX PCS (not ready) ➔ (not linked up) ➔ PHY B RXD (LF)
  PHY A RXD (LF) ➔ (not linked up) ➔ PHY B TX PCS (not ready) / PHY B RXD (LF)

- Assume PHY A links up first
- PHY A RXD (RF) ➔ PHY A TXD (Idle) ➔ (not linked up) ➔ PHY B RXD (LF)
  PHY A RXD (RF) ➔ (linked up) ➔ PHY B TX PCS (not ready) / PHY B RXD (LF)

- PHY A RXD (Idle) ➔ PHY A TXD (Idle) ➔ (linked up) ➔ PHY B RXD (Idle)
  PHY A RXD (Idle) ➔ (linked up) ➔ PHY B TXD (Idle) ➔ PHY B RXD (Idle)

How MACs that don’t support LF and RF

- Unrecognized codes are interpreted as idles
- So mechanism is backwards compatible

How MII Works with Modification – MACs without RS Support

- PHY A TX PCS (not ready) ➔ (not linked up) ➔ PHY B RXD (LF)
  PHY A RXD (LF) ➔ (not linked up) ➔ PHY B TX PCS (not ready)

Assume PHY A links up first

- PHY A TXD (Idle) ➔ (not linked up) ➔ PHY B RXD (LF)
  PHY A RXD (RF) ➔ (linked up) ➔ PHY B TX PCS (not ready)

- PHY A TXD (Idle) ➔ (linked up) ➔ PHY B RXD (Idle)
  PHY A RXD (Idle) ➔ (linked up) ➔ PHY B TXD (Idle) ➔ PHY B RXD (Idle)

THANK YOU