Canova Tech

*The Art of Silicon Sculpting*

**PIERGIORGIO BERUTO**
**ANTONIO ORZELLI**

*IEEE802.3cg TF*

# *PHY-Level Collision Avoidance – Addendum #2*

*August 30th, 2017*

# Overview

- PHY Level Collision Avoidance (PLCA)
  - Media access multiplexing protocol
  - Proposal for multi-drop, half-duplex, short reach PHY
- Objectives
  - Interworking with standard CSMA/CD MAC
    - No modifications to MAC, everything done at PHY level
  - Beat CSMA/CD performance, especially at high bus loads
  - Keep complexity low

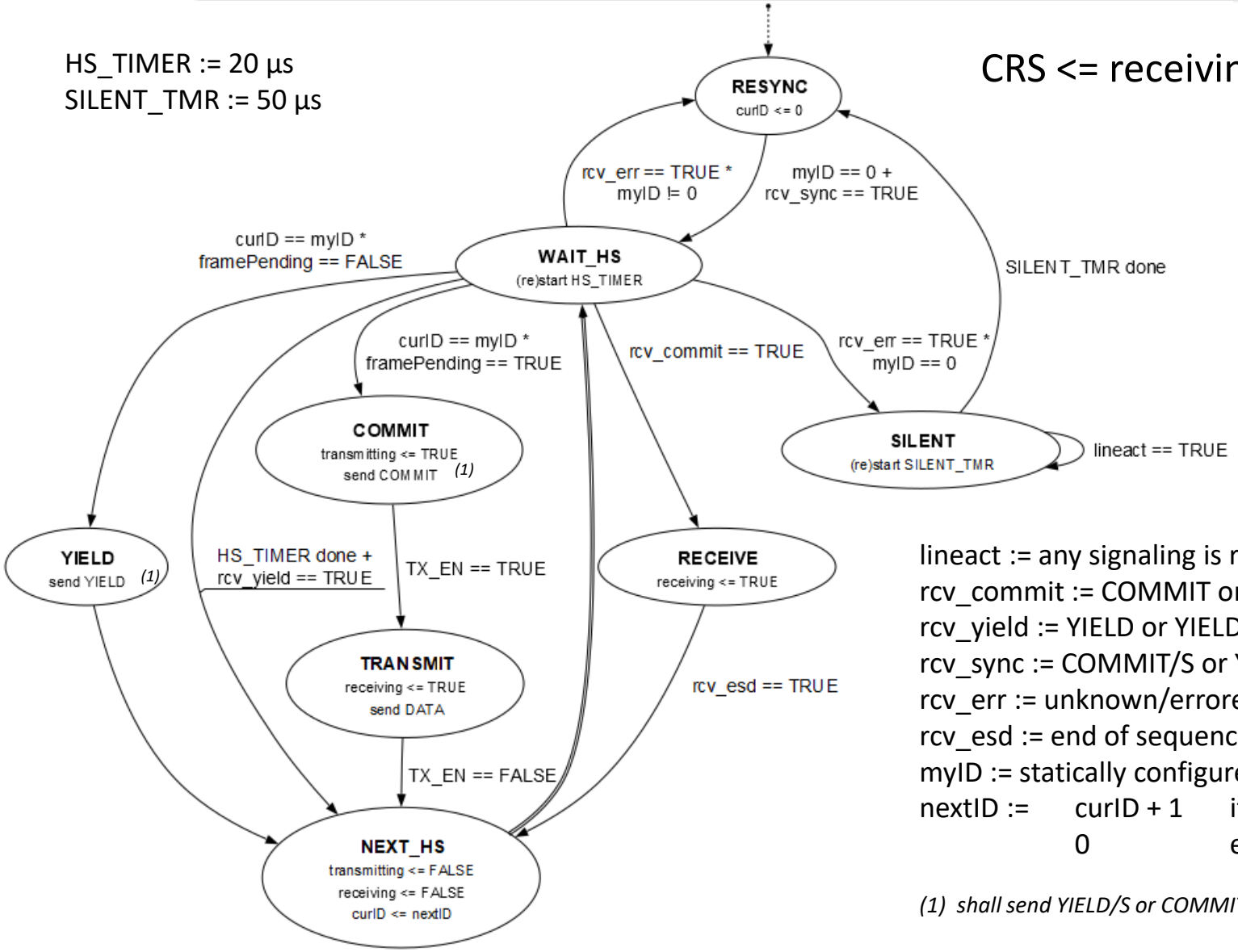
# PLCA definition

- PLCA can be represented with two parallel processes
  - TX process
    - implements handshaking protocol
  - COL process
    - implements logic to synchronize with the MAC
- Shared variables:
  - framePending
  - transmitting
  - receiving

# TX-PROCESS

HS_TIMER := 20 µs
SILENT_TMR := 50 µs

CRS <= receiving OR forceCRS



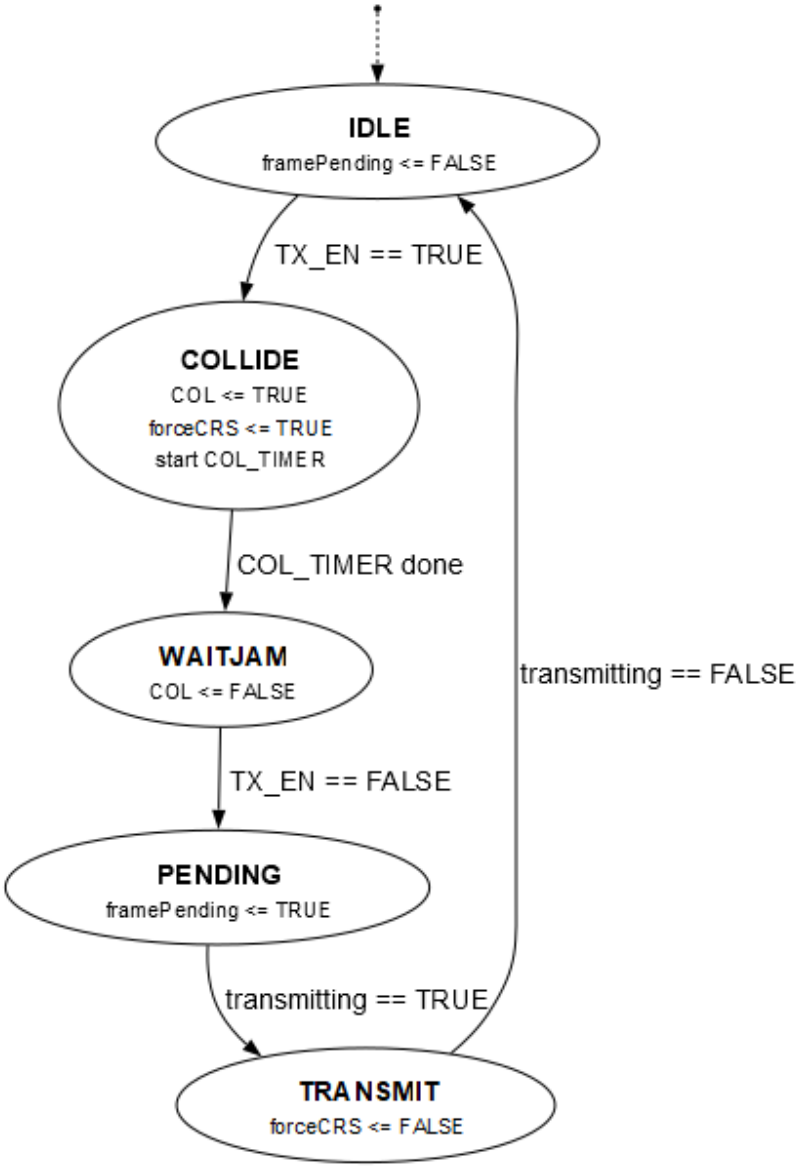

lineact := any signaling is received
rcv_commit := COMMIT or COMMIT/S received
rcv_yield := YIELD or YIELD/S received
rcv_sync := COMMIT/S or YIELD/S received
rcv_err := unknown/errored signaling received
rcv_esd := end of sequence delimiter received
myID := statically configured PHY ID

| | | |
|---|---|---|
| nextID := | curID + 1 | if curID < MAX_ID |
| | 0 | else |

*(1) shall send YIELD/S or COMMIT/S if myID == 0.*

# COL-PROCESS

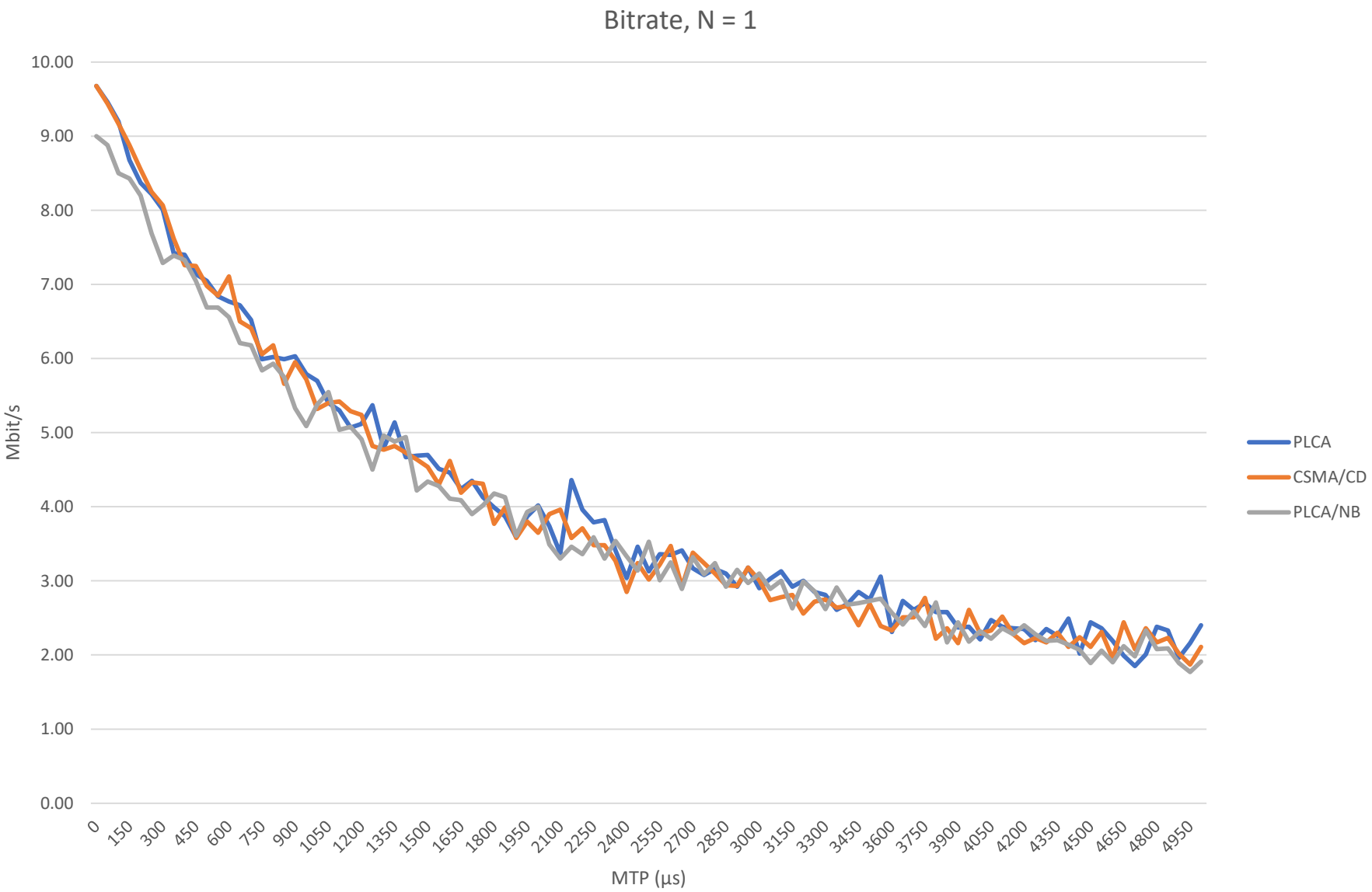
COL_TIMER := 1.6 µs



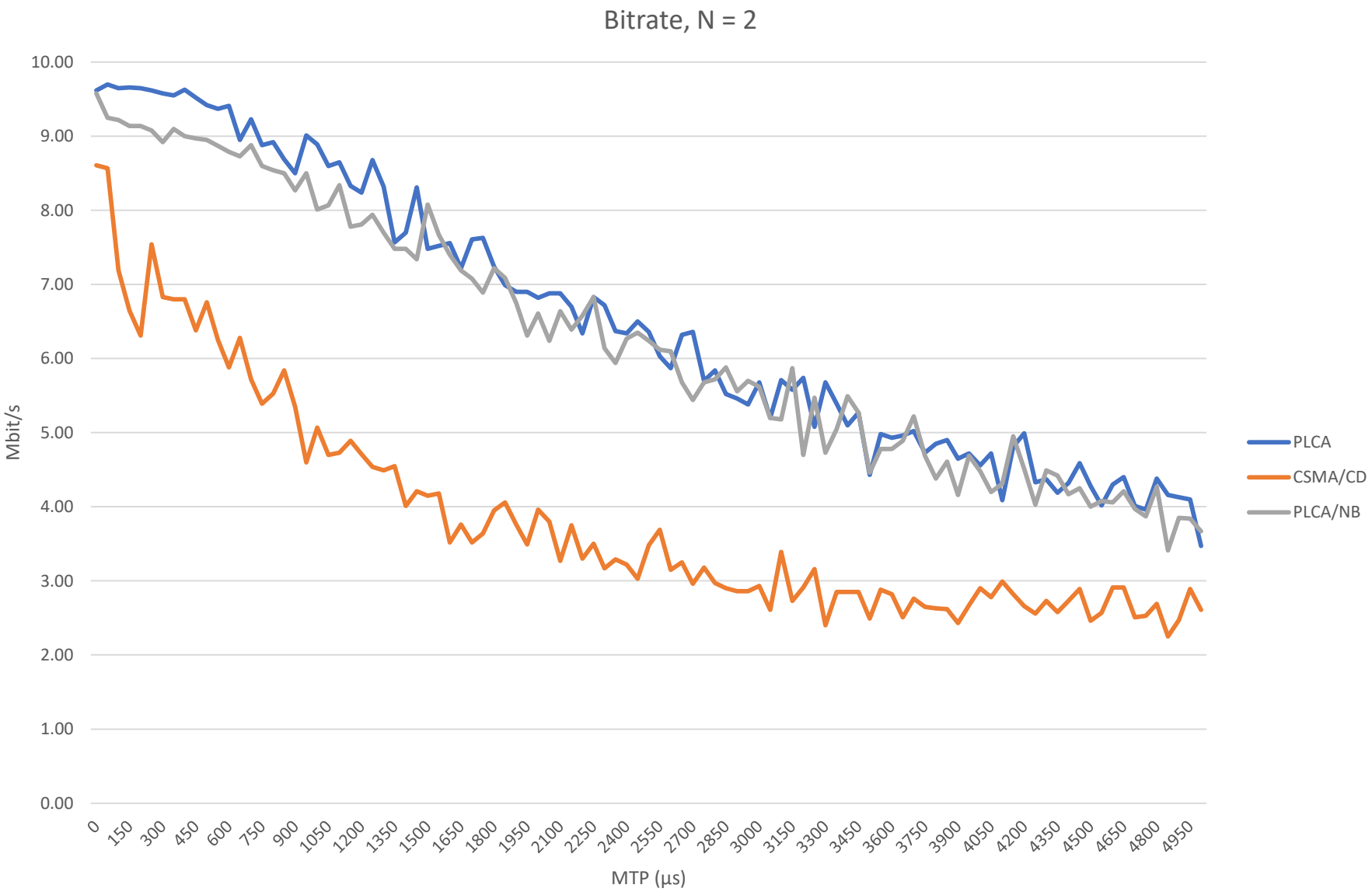
# Notes

- Unlike previous presentations, no buffering is required
- This proposal, however, does not preclude specific implementations to use buffering to further improve performance
- Throughput and latency simulations have been re-run to compare PLCA with and without buffering.

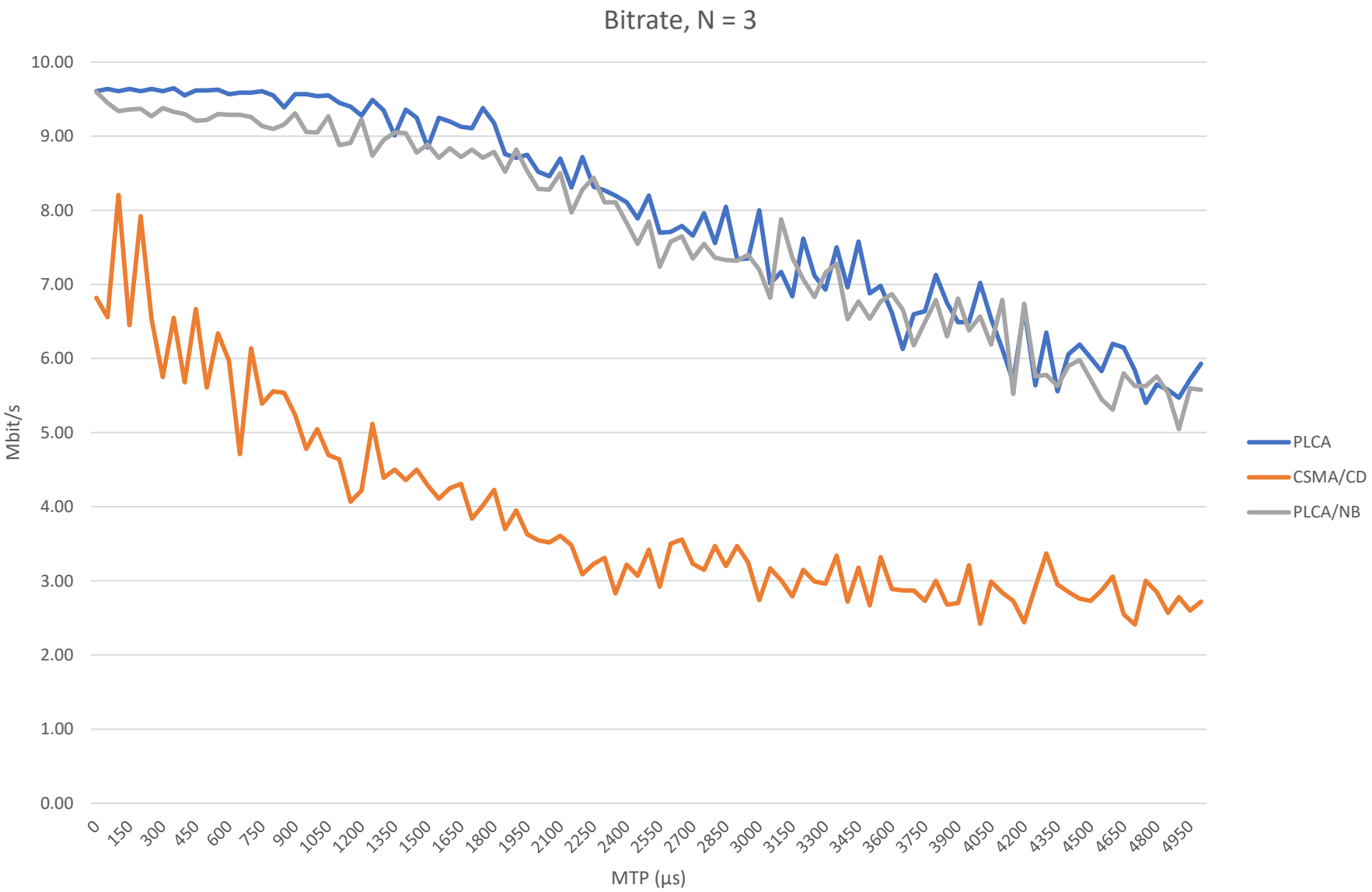
# Simulations



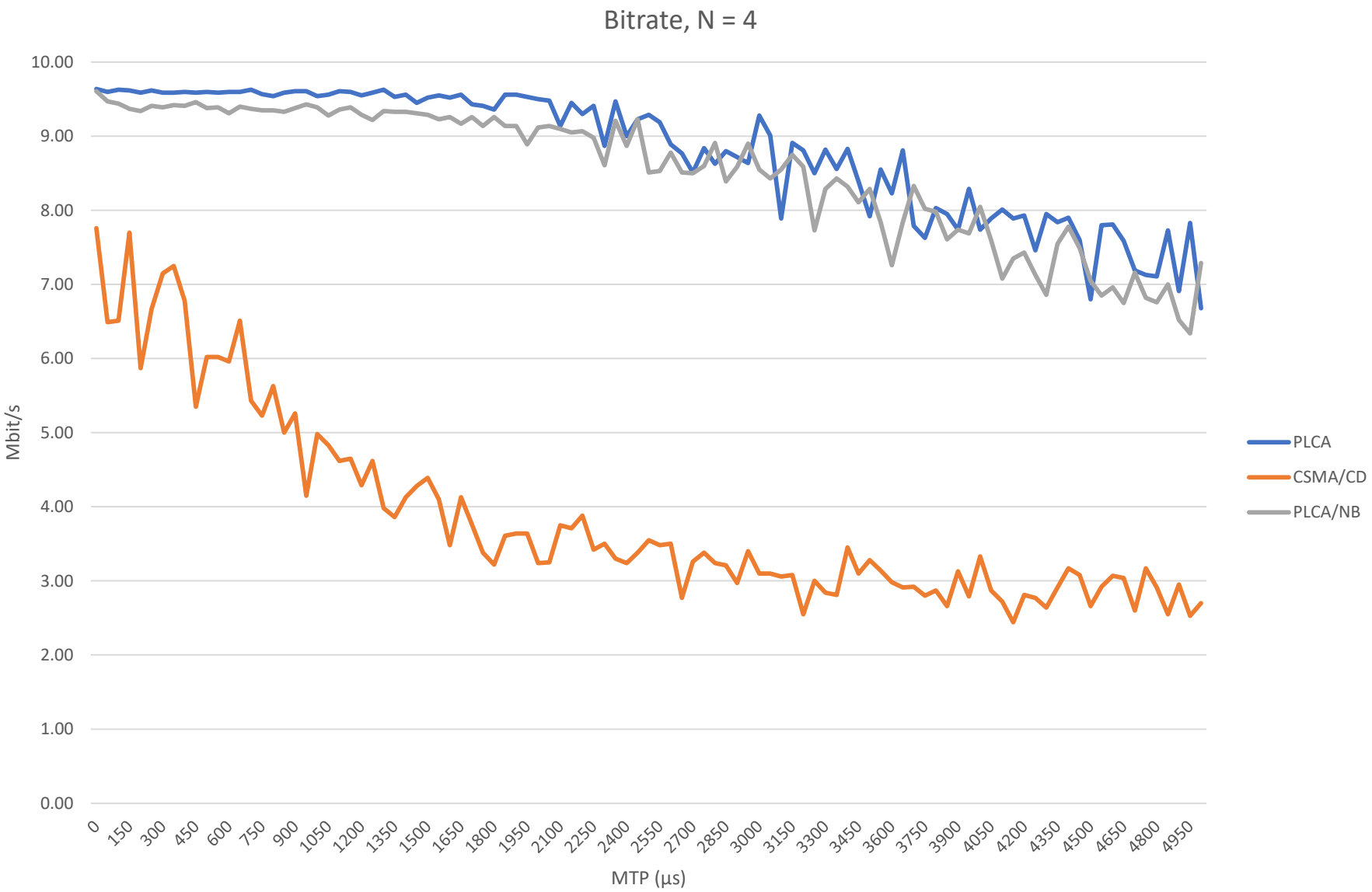
# Simulations



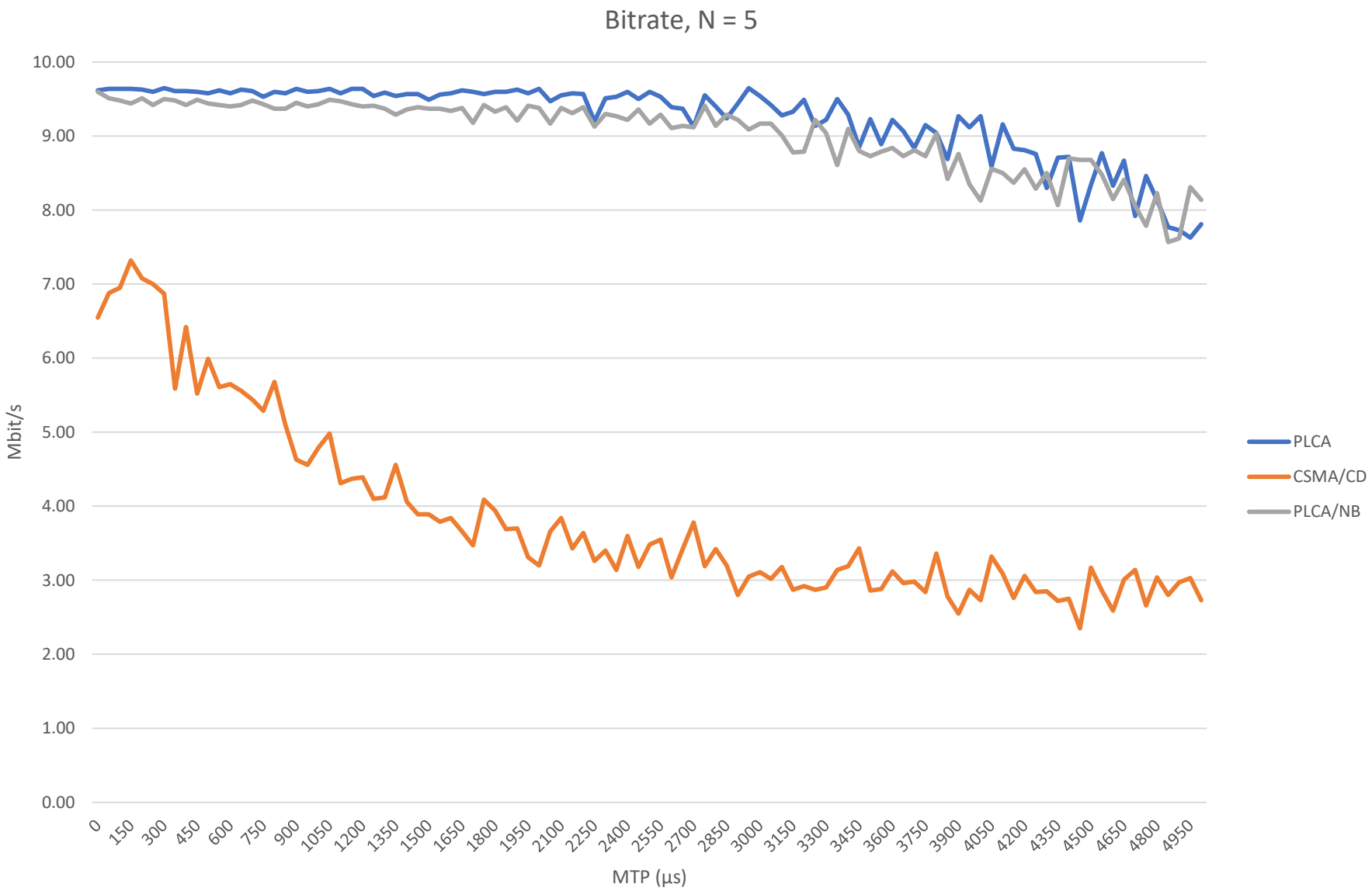
# Simulations



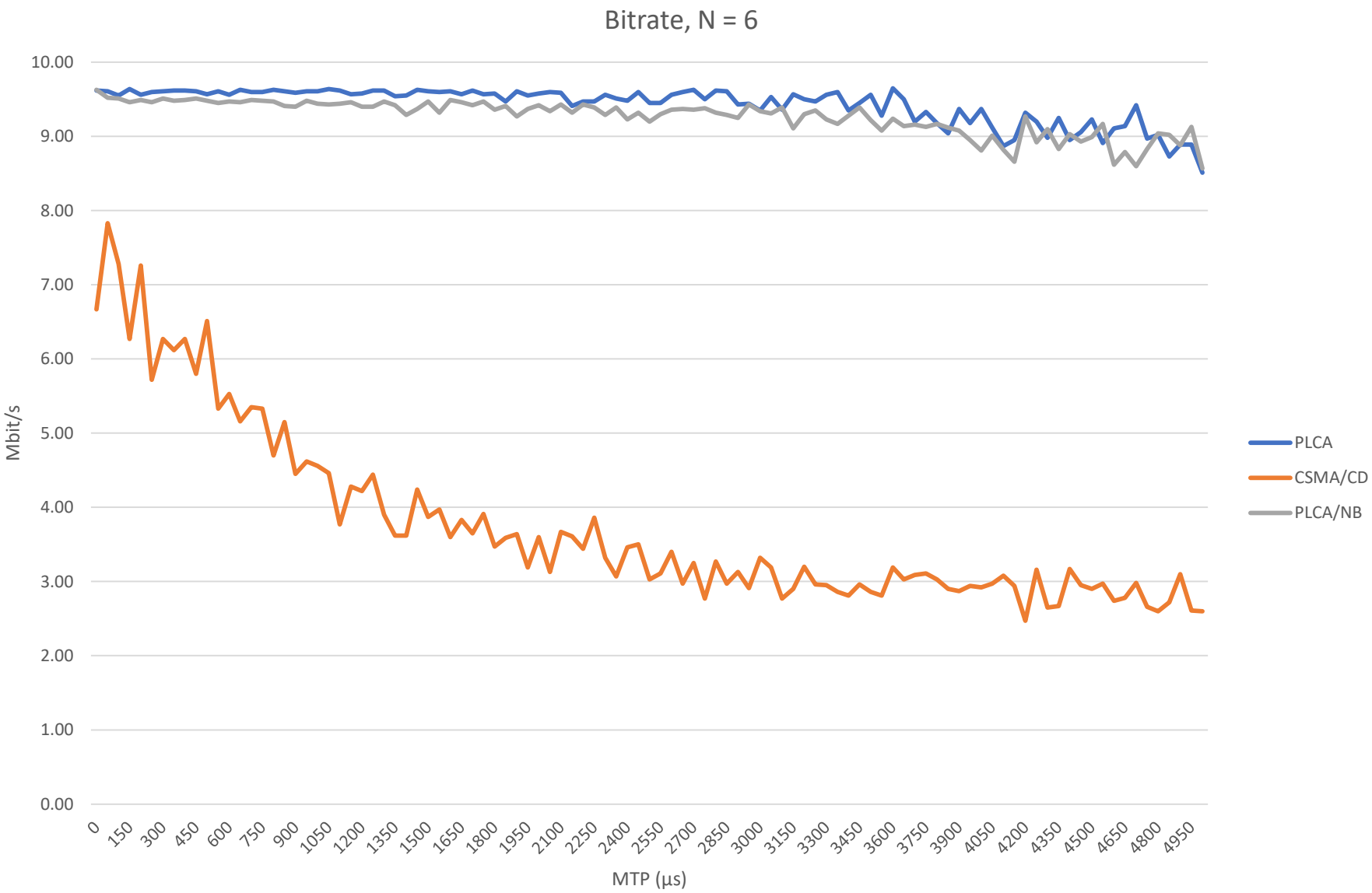
# Simulations



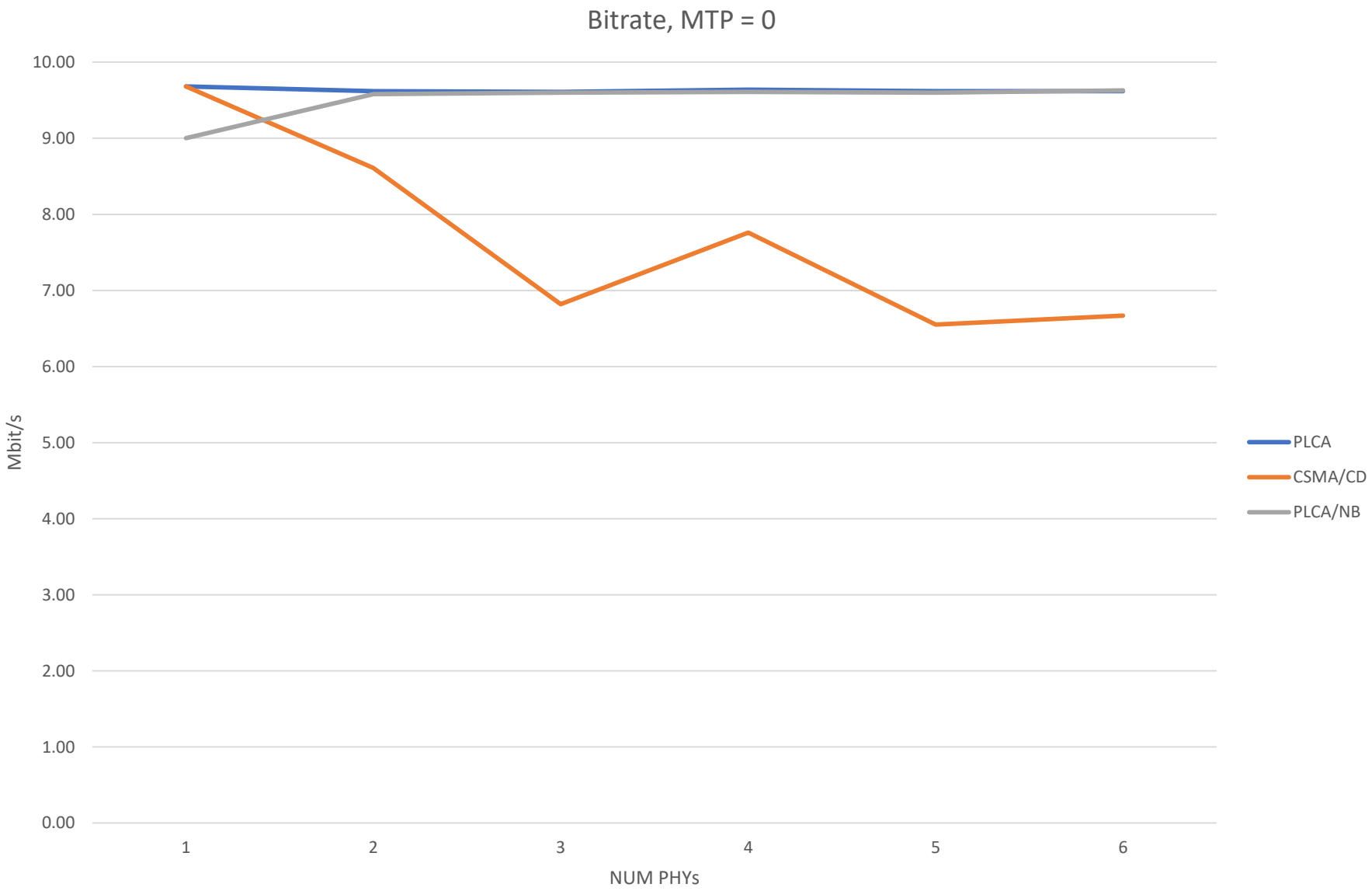
# Simulations



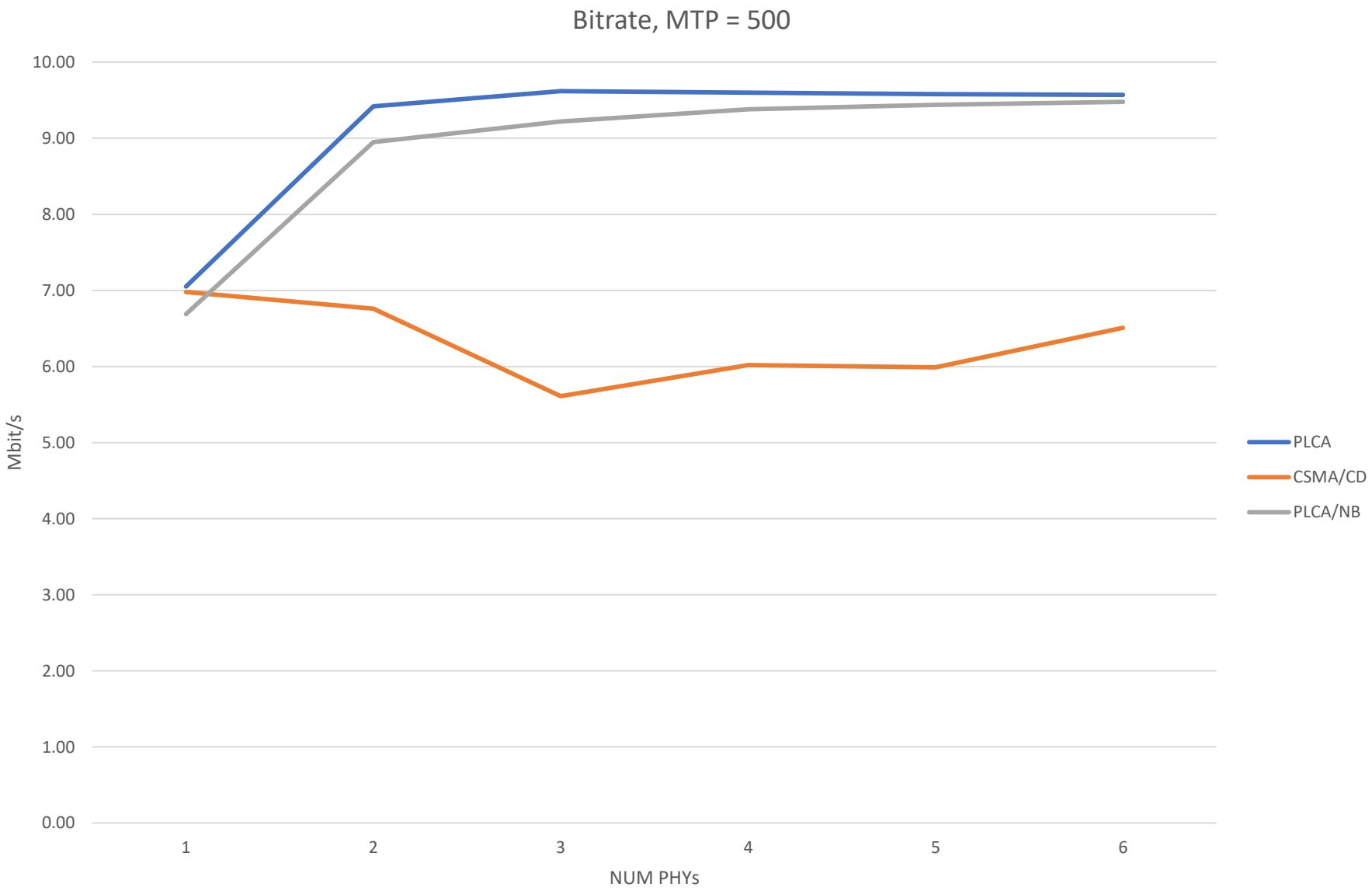
# Simulations



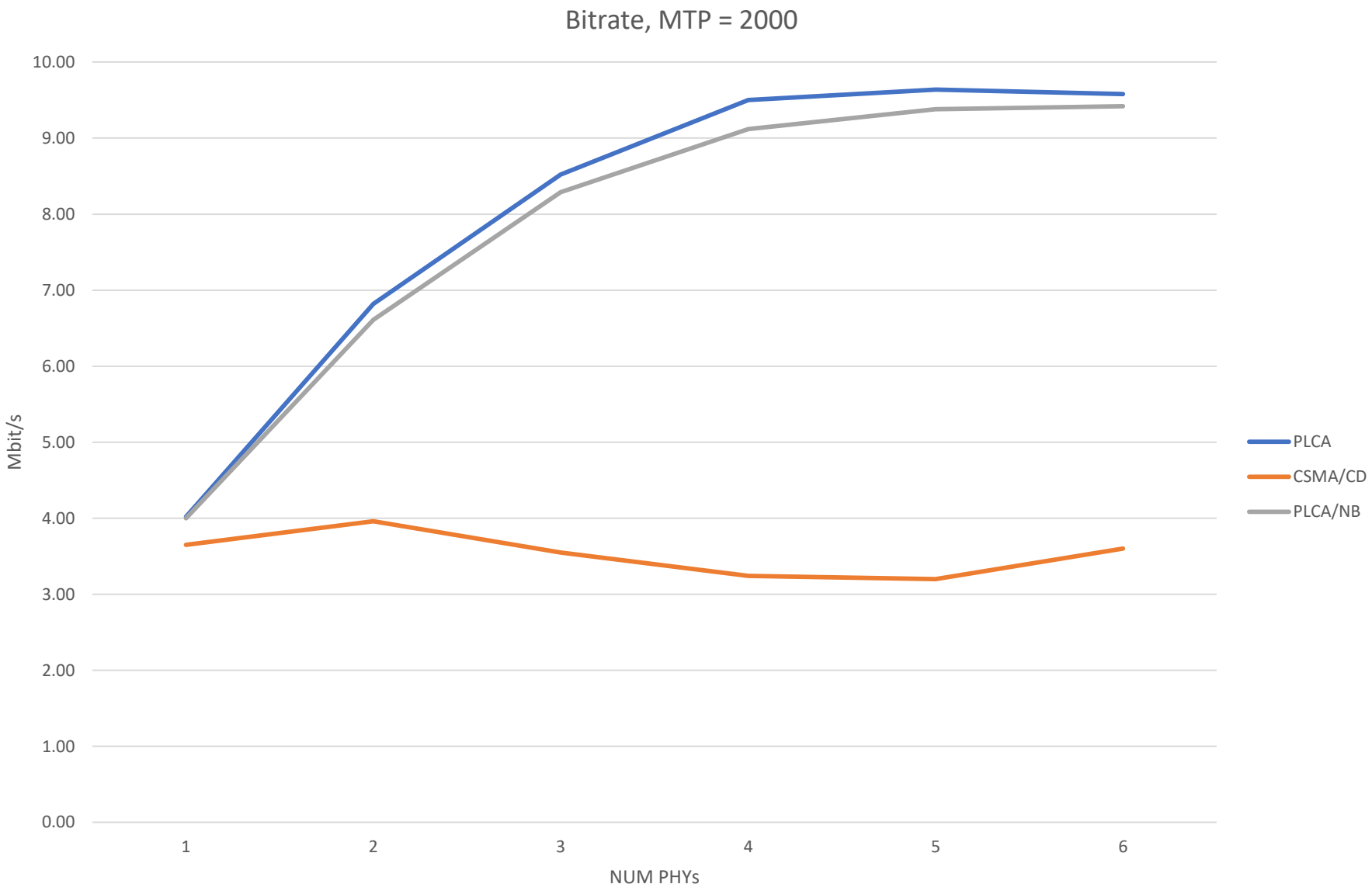
# Simulations



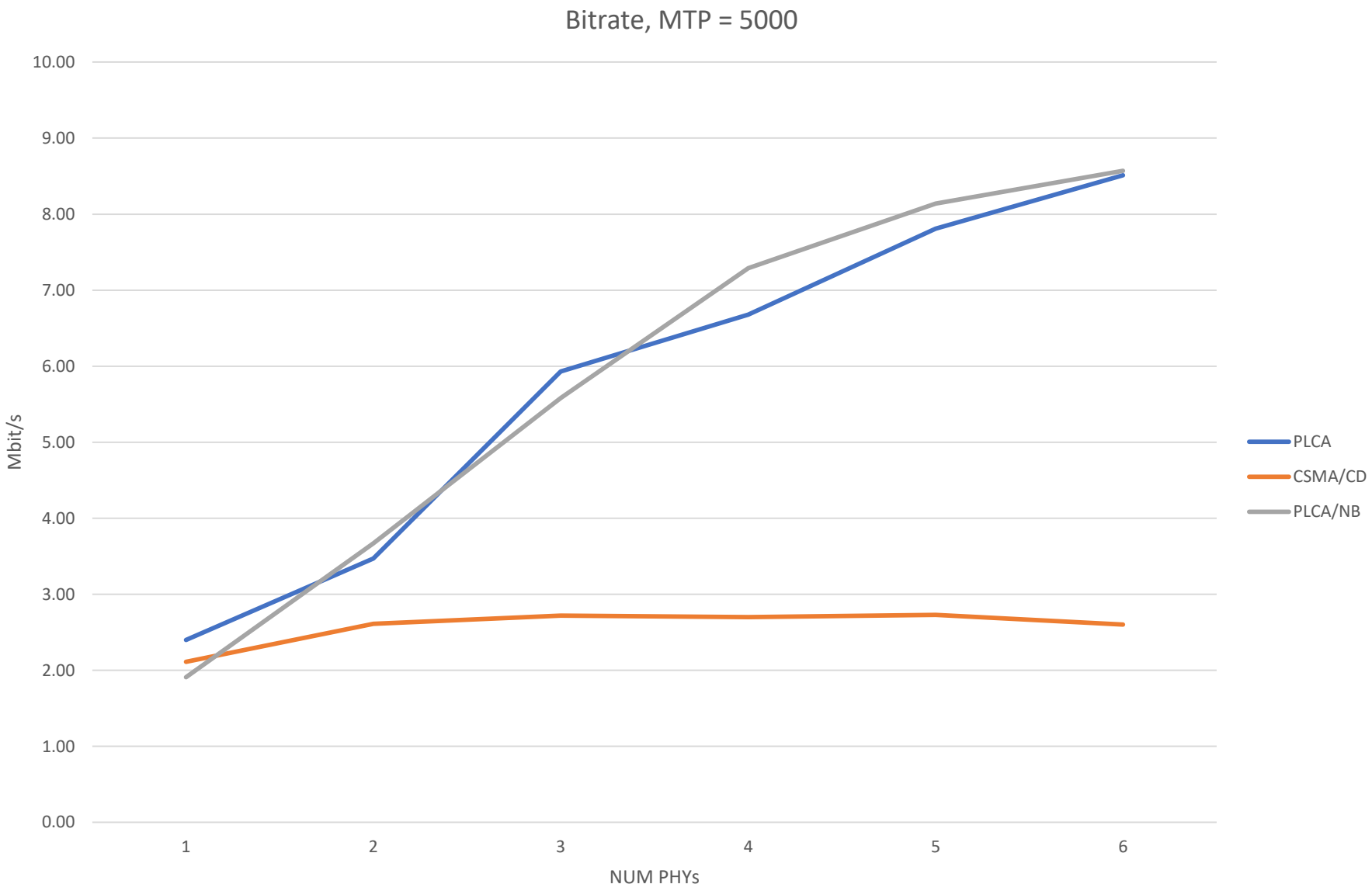
# Simulations



# Simulations



# Simulations



# Latency

- 500 pkts, size = 60B, variable MTP, 6 nodes. Latencies in µs.
- Comparison between 10base-T (simple CSMA/CD), PLCA and PLCA/NB

| MTP | MAX_LAT | AVG_LAT | STDEV |
|---|---|---|---|
| 0 | 57595.6 | 553.3 | 4826.0 |
| 500 | 59692.8 | 1034.2 | 4637.4 |
| 2000 | 29387.5 | 618.9 | 2298.2 |
| 5000 | 19645.4 | 264.0 | 1035.7 |

CSMA/CD

| MTP | MAX_LAT | AVG_LAT | STDEV |
|---|---|---|---|
| 0 | 443.4 | 441.1 | 26.2 |
| 500 | 546.4 | 186.4 | 90.7 |
| 2000 | 269.2 | 74.8 | 31.6 |
| 5000 | 223.7 | 64.0 | 17.8 |

PLCA

| MTP | MAX_LAT | AVG_LAT | STDEV |
|---|---|---|---|
| 0 | 448.4 | 441.4 | 24.0 |
| 500 | 631.2 | 275.8 | 109.3 |
| 2000 | 379.6 | 138.5 | 55.5 |
| 5000 | 296.8 | 113.8 | 37.7 |

PLCA/NB

- Latency somewhat increased in PLCA/NB
  - Expected as backoff is always triggered

# Next Steps

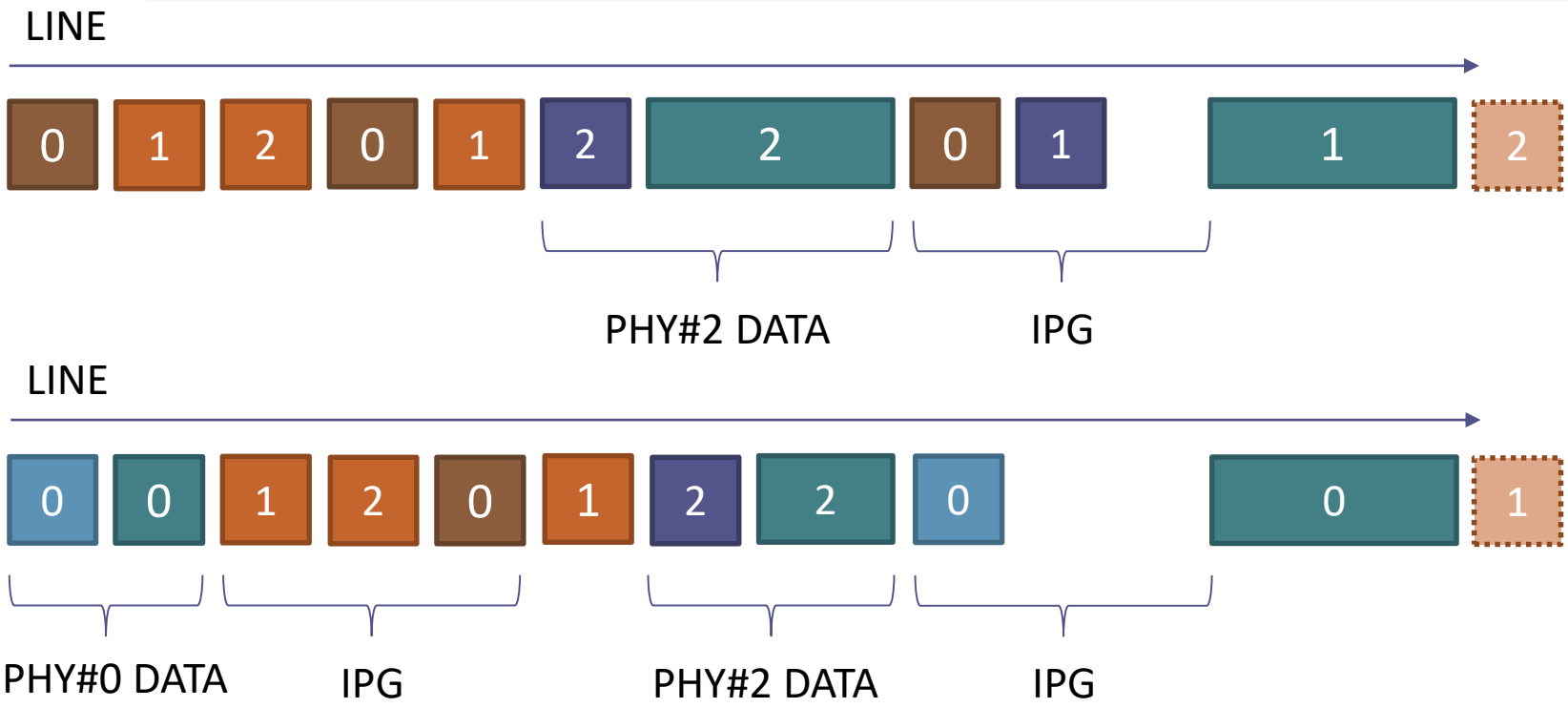
- PCS definition
  - Ok with 4b5b?
    - "JK" → SSD
    - "TR" → ESD
    - "S" → COMMIT, "T" → COMMIT/S
    - "R" → YIELD, "H" → YIELD/S
- Channel simulations
  - 4b5b + DME = 25 MHz
  - Line topology, 6 PHYs

# Thank You !

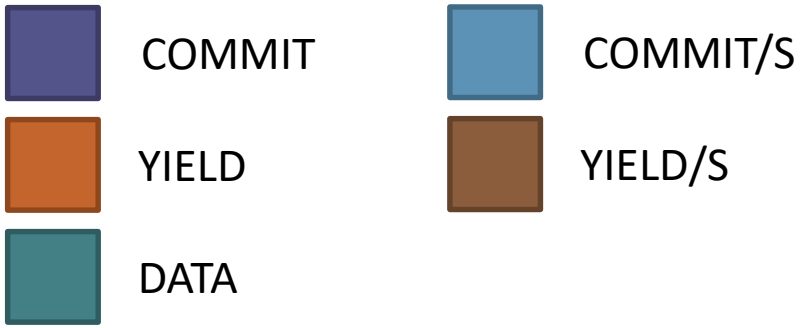
# In a nutshell

- Assumptions
  - Up to 6 nodes with pre-configured unique IDs (0-5)
    - ID = 0 is the "master" node
  - Interface to standard 10Mbit half-duplex CSMA/CD capable MAC
  - PCS encoding shall allow more signaling (SSD, ESD, COMMIT, YIELD, COMMIT/S, YIELD/S)
- Basics
  - PHYs send packet in sequence (no collisions)
    - Also good for fairness! (no starvation)
  - Starting from master, each PHY sends COMMIT + DATA or YIELD in turn, following unique ID order.
    - no additional overhead as long as handshake time < IPG
    - no waste of bandwidth (PHYs with no data to send just "skip the turn")
  - Implicit YIELD after timeout to handle absent/link-down nodes
    - Constraint on maximum allowed TX/RX latency
      - Trade-off with max achievable throughput

# Control signals

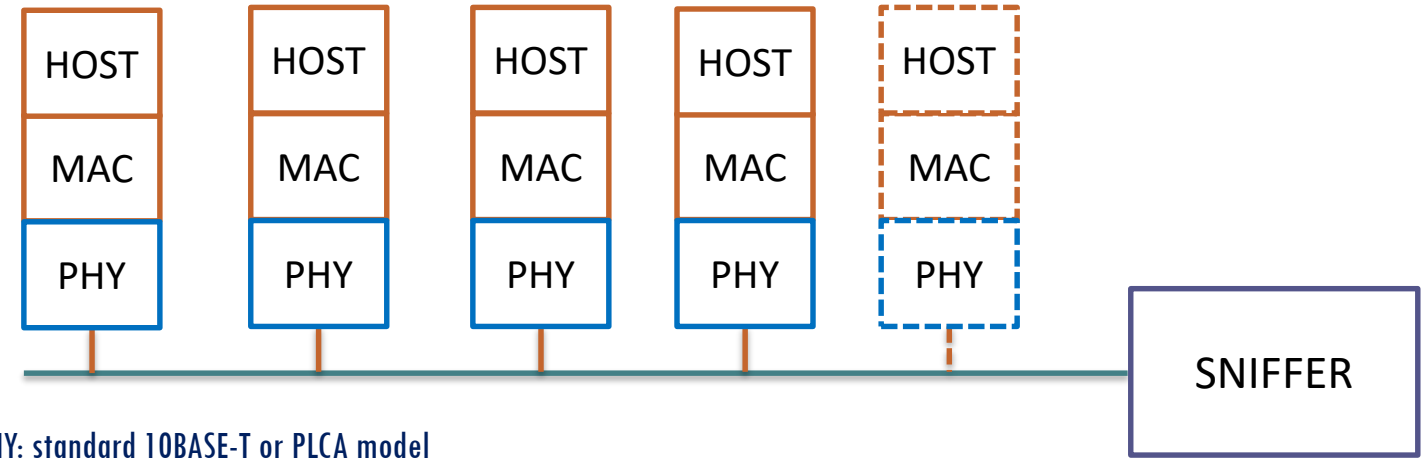



- **Control signals:**
  - COMMIT → take time slot to send data
  - YIELD → leave time slot for others
  - COMMIT/S → as COMMIT, sent by master to allow synchronization
  - YIELD/S → as YIELD, sent by master to allow synchronization



# Simulations

- Full digital simulation (verilog)
  - 4b/5b encoding + DME (25Mhz)
  - Use 5b codes S, R, T, H for COMMIT, YIELD, COMMIT/S, YIELD/S signaling




- PHY: standard 10BASE-T or PLCA model
- MAC: standard CSMA/CD capable MAC (802.3 clause 4)
  - host interface: DPRAM (one frame) + busy indication + size + trigger
  - PHY interface: MII (txd, txclk, txen, txer, rxd, rxclk, rxdv, rxer, col, crs)
- HOST: simple transmitter
  - wait for MAC BUSY = 0
  - wait random time between 0 and MTP (sim parameter, 0 = MAX speed)
  - write random data in DPRAM of size PKTSZ (sim. parameter 60 < PKTSZ < 1500) or random size
- SNIFFER: measuring throughput, latency
  - throughput: number of received bytes (excluding FCS, PREAMBLE) / total simulation time
  - latency: time between BUSY = 1 and BUSY = 0 for each node

# Interface with CSMA/CD MAC

- CSMA/CD MAC transmit process (from IEEE 802.3, clause 4)
  - If line is busy (CRS = 1) → wait (defer transmission)
  - Wait IPG (at least 96 bits)
  - Start transmitting, despite line becoming busy again
    - If a collision is detected (COL = 1) → backoff:
      - Send jam for 32 bit times, stop transmission
      - Retry after random(0, ATTEMPTS) * 512 bit times
      - If ATTEMPTS > attemptLimit → give up (discard packet)
- CRS / COL can be used to have the MAC defer transmission until next handshaking
  - Use CRS to have the MAC defer transmission
  - Use COL at most once and only at beginning of a packet
    - MAC is ready to re-send in at most 32 + 512 = 544 bit times
      - Less than minimum packet size (576 bits)
      - COLLISION AVOIDED, NO WASTE OF BUS TIME!

# Join/ Leave

- Still static configuration of the IDs
- How to handle "disabled" nodes?
  - COMMIT/YIELD is expected to be sent within a determined time (HS timer).
  - On timeout an implicit YIELD is assumed and next node will resume handshaking
  - HS timer shall be long enough to accommodate for PHY latency but as short as possible not to degrade performance
  - Simulations performed with HS_TIMER = 20µs
    - Allows for ~16 µs of HS latency
    - Simulated with random missing nodes
      - Negligible impact on performance
- PHY with ID = 0 (master) use COMMIT/S and YIELD/S for handshaking.
  - Slave nodes synchronize on xxx/S signals before joining the HS mechanism
    - link_status shall not be signaled until first sync
  - Nodes receiving "bad" COMMIT/YIELD shall not transmit and re-synchronize
    - if this happens to the master, this one shall wait until there is no data on the line for a certain amount of time, then re-start handshaking.