MAC/PHY Delay Variability:
Corner Cases and Solutions

glen.kramer@teknovus.com

Introduction

- An action item from September meeting required to verify the PHY Delay Variability through simulation and to report the results back to TF in November (see comment 2414)

- In simulation, three corner cases were identified that cause increased delay variability

  Case 1: Deficit Idle Counter changes FEC codeword boundary

  Case 2: Under light load, a frame arrives between column boundaries

  Case 3: Under light load, a frame arrives during parity block

Case 1

- This case is related to misalignment between IPG assumed at the MPCP and the actual IPG allocated by the MAC/RS, with or without the Deficit Idle Counter mechanism (see 46.3.1.4)

- DIC gives the RS an efficient mechanism to align Start control characters to lane 0 without always inserting extra idle
- DIC allows for minimum inter-frame gap to vary from 9 to 15 bytes with the average gap value remaining at 12 bytes

	Current DIC = 0		Current DIC = 1		Current DIC = 2		Current = 3	
Packet Length Modulo 4	IPG Length	New DIC value	IPG Length	New DIC value	IPG Length	New DIC value	IPG Length	New DIC value
n+0	12	0	12	1	12	2	12	3
n+1	11	1	11	2	11	3	15	0
n+2	10	2	10	3	14	0	14	1
n+3	9	3	13	0	13	1	13	2

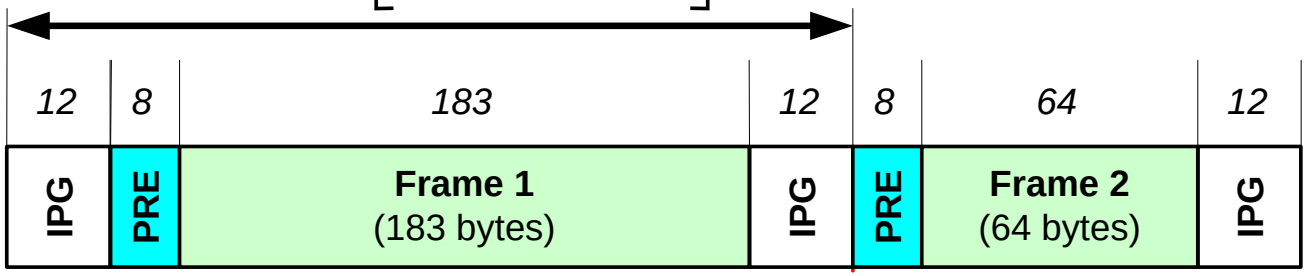
- Without DIC, IPG can take only the values of 12, 13, 14, and 15, depending on packet size

Illustration of Case 1

MPCP: Control Multiplexor

MPCP Calculation:
Initial ByteTime = 0
FEC_Overhead = 32*⌊(12+8+183+12)/216⌋ = 0

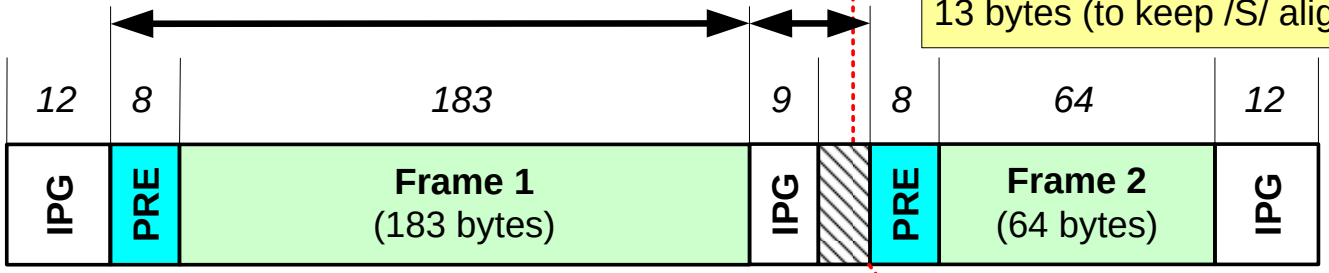
1. MPCP does not allocate parity overhead before frame #2



RS: Deficit Idle counter

RS Calculation:
Old DIC = 0 → New DIC = 3
191 **mod** 4 = 3 → Min.IPG = 9

2. DIC calculates Min IPG = 9, but after 9 bytes, frame #2 is not yet available from MPCP/MAC. The actual IPG increases to 13 bytes (to keep /S/ aligned to lane 0).

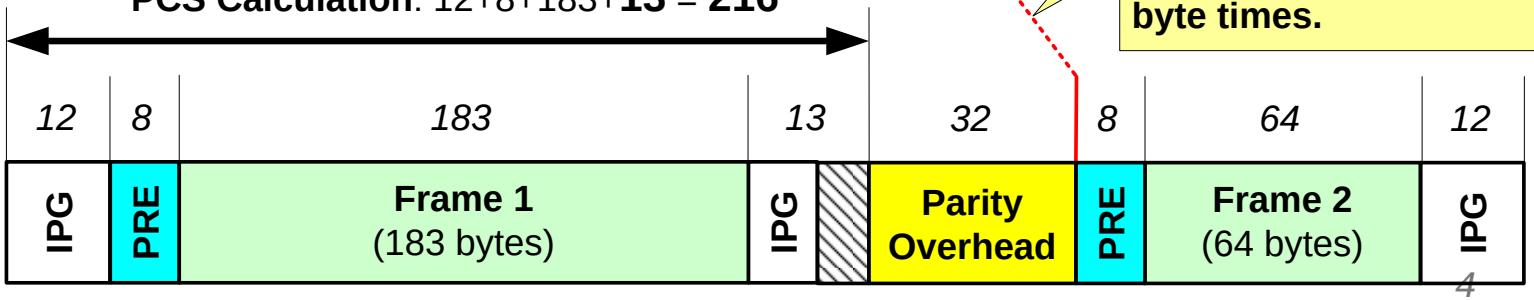


PCS: Data Detector

3. Complete FEC payload is ready before frame #2

PCS Calculation: 12+8+183+**13** = **216**

4. FEC Encoder inserts 4 parity blocks before frame #2. **Frame #2 is delayed by 32 byte times.**



Notes on Case 1

- **Case 1 may happen when**
  1. Deficit Idle Counter allocates smaller IPG (9, 10, 11 octets) and next frame is not available
  2. Deficit Idle Counter allocates larger IPG (13, 14, 15 octets) and next frame is available
- **Case 1 happens because MPCP is trying to estimate how long the MAC will take to transmit the frame**
  - The cleanest solution is to allow MAC to tell the MPCP when it is ready for the next packet (as it was in 802.3ah).
  - Unfortunately, due to some global changes done by other TFs, the MAC unexpectedly lost this ability. This problem is being fixed right now, but it is unknown when the changes will be added to 802.3.
  - We have to have a working interim solution.
- **Whether the Deficit Idle Counter will try to reduce the IPG or increase it, the actual IPG increases in most cases. Case 1 will be solved if MPCP will consider that fact when it calculates packet_initiate_delay values.**

Proposed Solution for Case 1 [1]

- The simplest solution for Case 1 is to modify FEC_Overhead() function to account for the increased IPG.

$$FEC\_overhead(length)$$

{

$$length = COLUMN\_SIZE \times \left\lceil \frac{length}{COLUMN\_SIZE} \right\rceil$$

$$return \;\; length + FEC\_PARITY\_SIZE \times \left\lfloor \frac{byteTime + length}{FEC\_CODEWORD\_SIZE} \right\rfloor$$

}

- Length is rounded up to column boundary
- FEC_Overhead() returns not only the size of the overhead, but the entire transmission length, including the frame, preamble, and the IPG. (This also will simplify the state diagrams as shown below)

Proposed Solution for Case 1 [2]

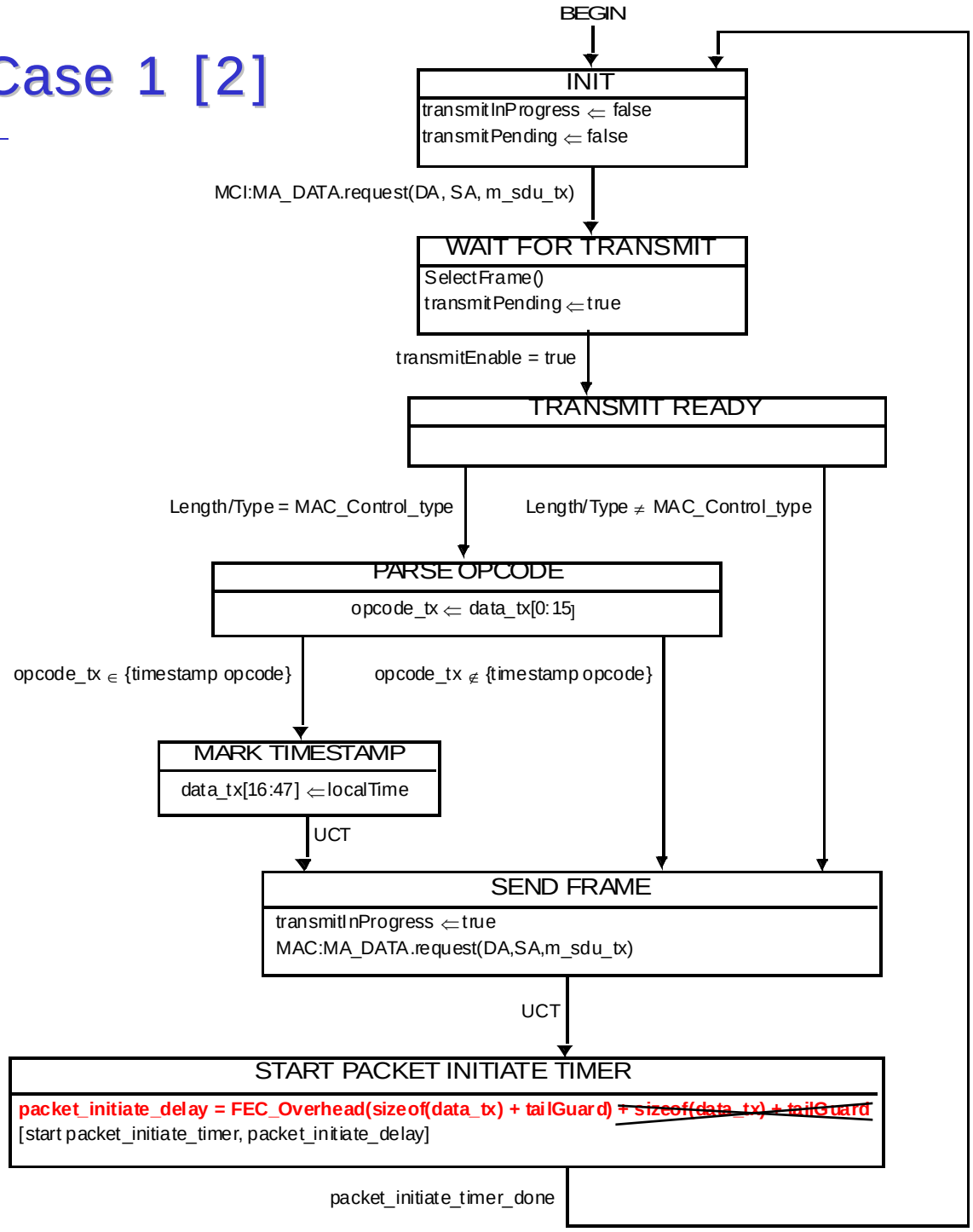
- In state START PACKET INITIATE TIMER, replace

```
packet_initiate_delay =
    FEC_Overhead(
        sizeof(data_tx) +
        tailGuard) +
    sizeof(data_tx) +
    tailGuard
```

with

```
packet_initiate_delay =
   FEC_Overhead(
        sizeof(data_tx) +
        tailGuard)
```

- **OLT Control Multiplexor (Figure 77-13) →**

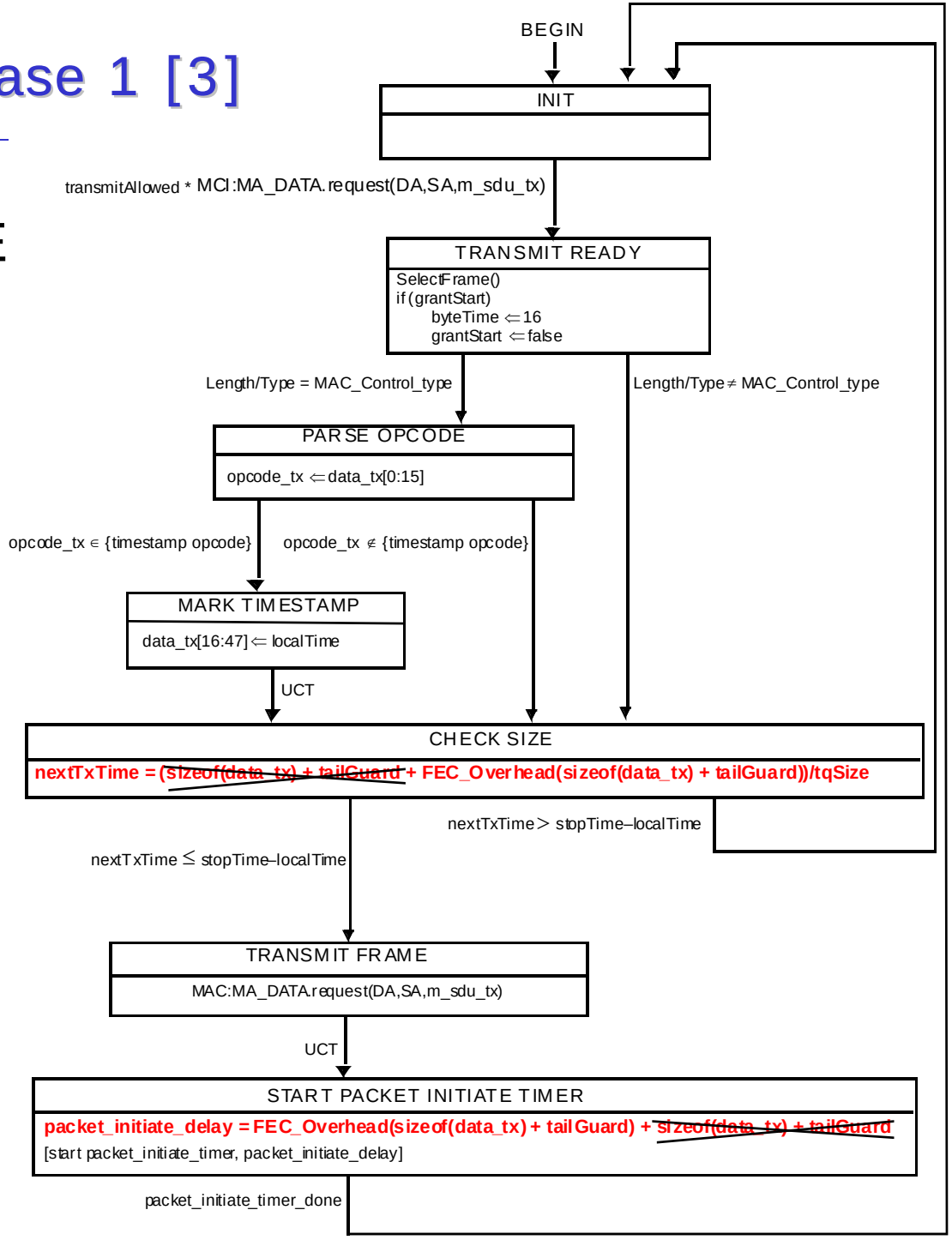


Proposed Solution for Case 1 [3]

- In states CHECK SIZE and START PACKET INITIATE TIMER, delete

  sizeof(data_tx) + tailGuard

  as shown



- **OLT Control Multiplexor (Figure 77-13) →**

Case 2

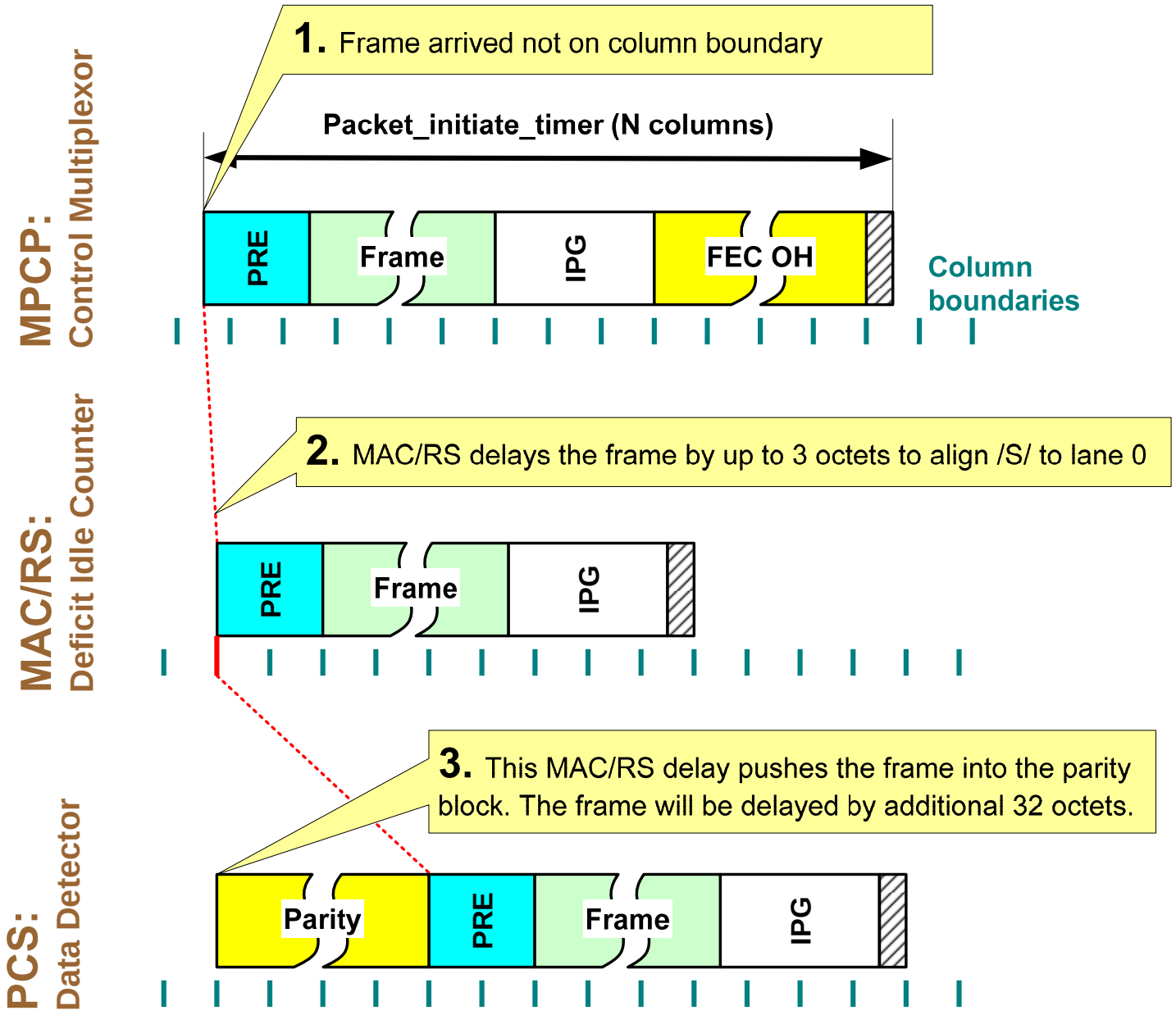
- The value of packet_initiate_delay (as fixed for case 1) is aligned to the next column boundary, to account for possible IPG increase in MAC/RS.

- When frames are available from MAC Client back-to-back (under heavy load), the calculation of packet_initiate_delay is done at the time when the previous packet_initiate_timer just expired, i.e., on column boundary. That assures that packet_initiate_timer expired on packet boundary.

- However, under light load, the next frame may not be available from the MAC Client when the previous packet_initiate_timer just expired. Instead, it may become available between column boundaries.

- In this case, the MAC/RS will additionally delay the frame to align /S/ character to lane 0. This delay will increase the transmission interval before the frame and may cause parity data to be inserted in front of the frame, causing further delay by additional 32 byte times.

Illustration of Case 2



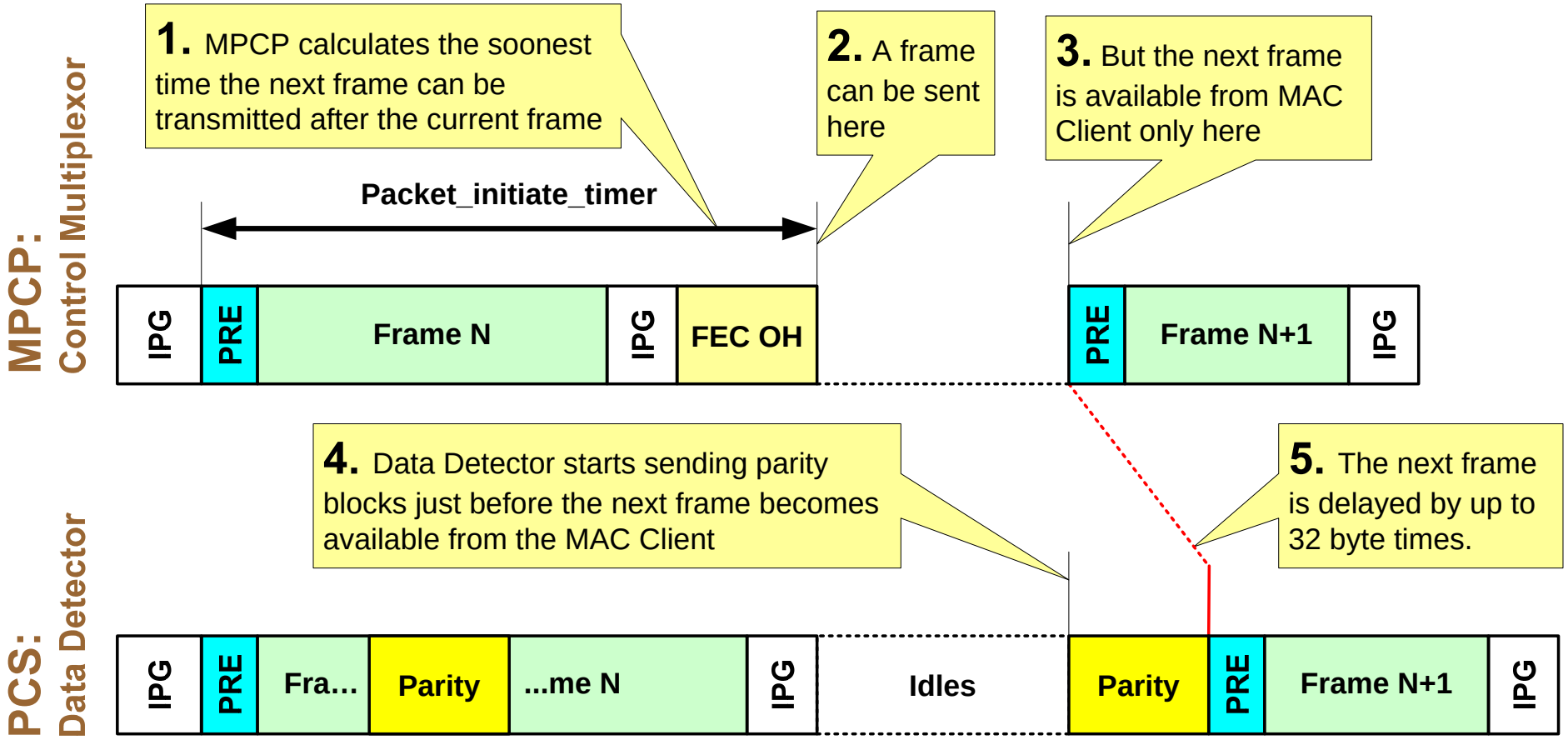
Case 3

- When MPCP Control Multiplexor sends a frame, it also calculates the time when the next frame can be transmitted (see Figures 77-13 and 77-14).

- Under the light load, the next frame may not be available from the MAC Client at the time when MPCP Control Multiplexor expects it.

- The next frame may become available at a time when Data Detector is inserting parity data. Thus, the next frame will experience a delay of up to 32 byte times.

Illustration of Case 3



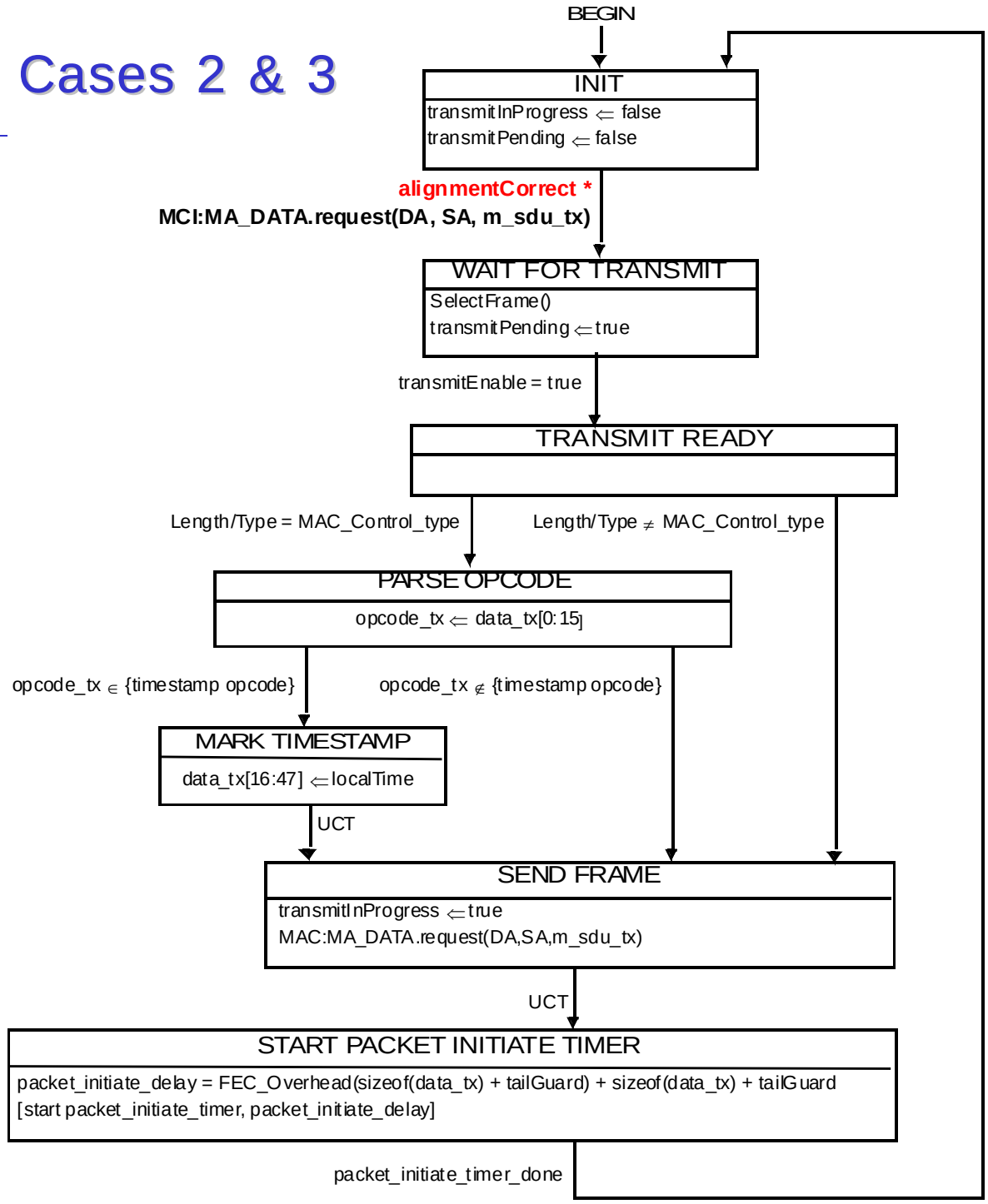
Proposed Solution for Cases 2 & 3

- Modify transition from INIT to WAIT TO TRANSMIT to allow frames through only
  - On column boundary

  and
  - During payload portion of FEC codeword.

- MAC Control frames that arrive during the parity portion will be delayed **before timestamping** until the parity part completes and column boundary is aligned.

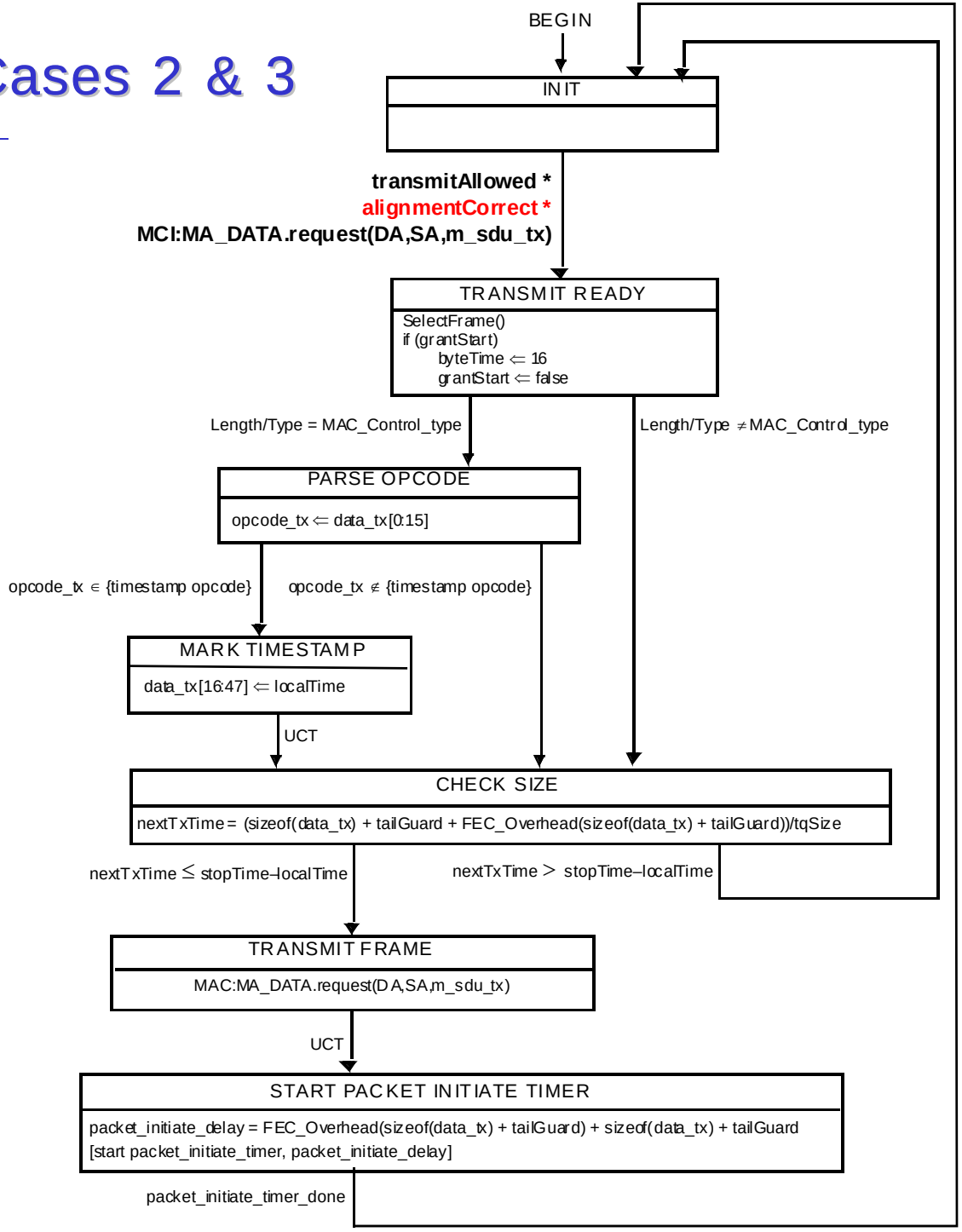
- **OLT Control Multiplexor (Figure 77-13) →**



Proposed Solution for Cases 2 & 3

- Similar modification should be made to ONU Control Multiplexor.

- **ONU Control Multiplexor (Figure 77-14) →**



Proposed Solution for Cases 2 & 3

- A new variable alignmentCorrect should be defined as follows:

alignmentCorrect

TYPE: boolean

At the OLT, this variable is an alias for the condition byteTime[1:0] == 0 AND byteTime < FEC_PAYLOAD_SIZE.

At the ONU, this variable is an alias for the condition byteTime[1:0] == 0 AND (byteTime < FEC_PAYLOAD_SIZE OR grantStart).

This variable is set true on column boundaries (XGMII transfers) occurring during the payload part of an FEC codeword. It is reset to false with the next increment of byteTime.

Summary

- Three corner cases are identified that can add up to 32 byte times of frame delay variability each (the three cases are exclusive)
  - Downstream variability: 32 byte times, or 1.6 TQ
  - Round-trip variability: 64 byte times or 3.2 TQ.
- These conditions are not detrimental. Current guard bands support 8 TQ downstream and 12 TQ round-trip.
- But removing this overhead is also not difficult
  - Few changes to state diagrams as shown above
  - With the proposed fixes, the frame delay variability is 0 in the downstream and 0.2 TQ in the upstream (due to /S/ character alignment)

Straw Poll

- The three corner cases should be fixed as suggested on slides 6-8 and 13-15.

  _____

- The delay variability due to the three corner cases should be considered a part of expected transmission overhead. No changes to state diagrams should be made.

  _____