Loop Aggregation Baseline

Klaus Fosmark

FirstMile Systems

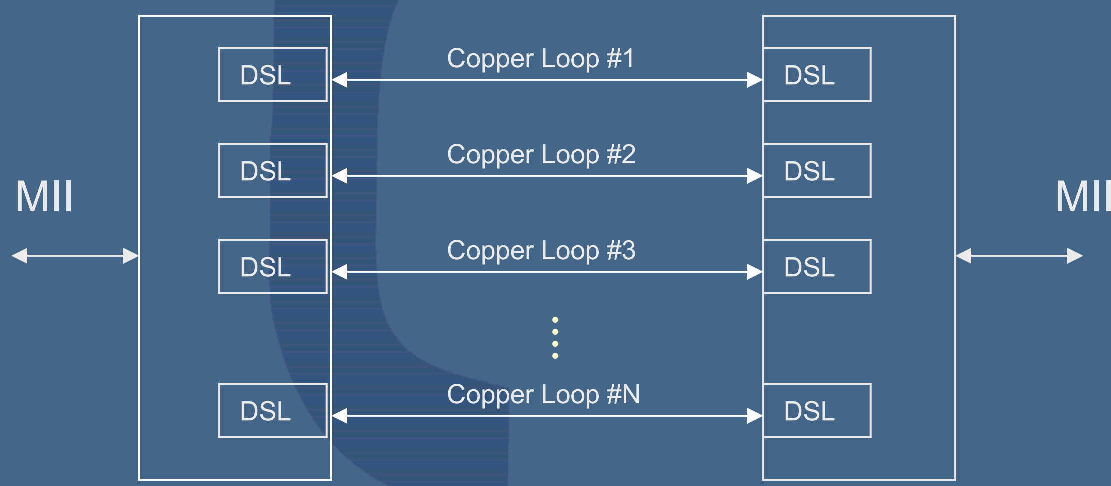
klaus@firstmilesystems.com

Summary of Agreed Issues

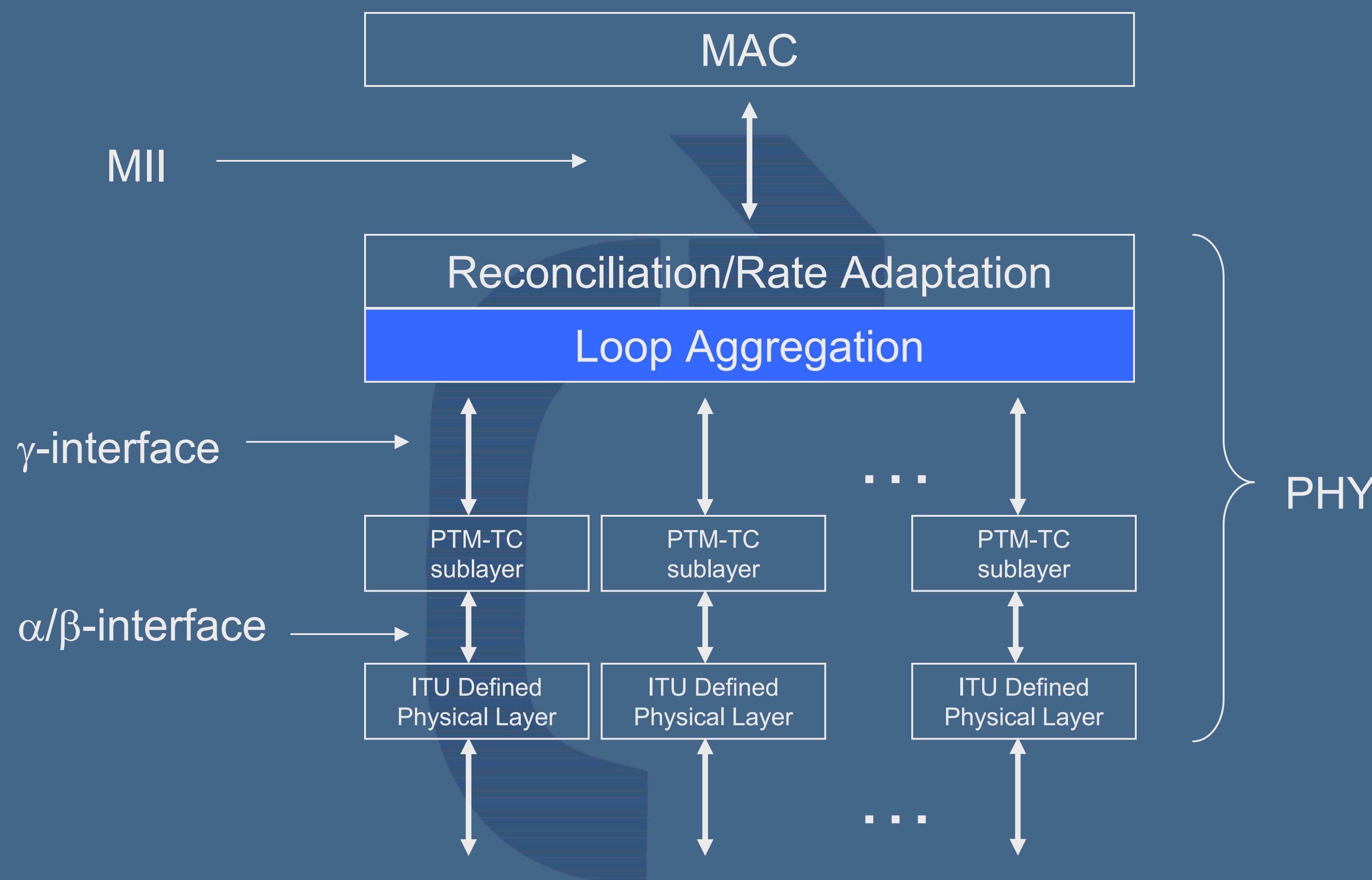
March, 2002

What is Loop Aggregation?

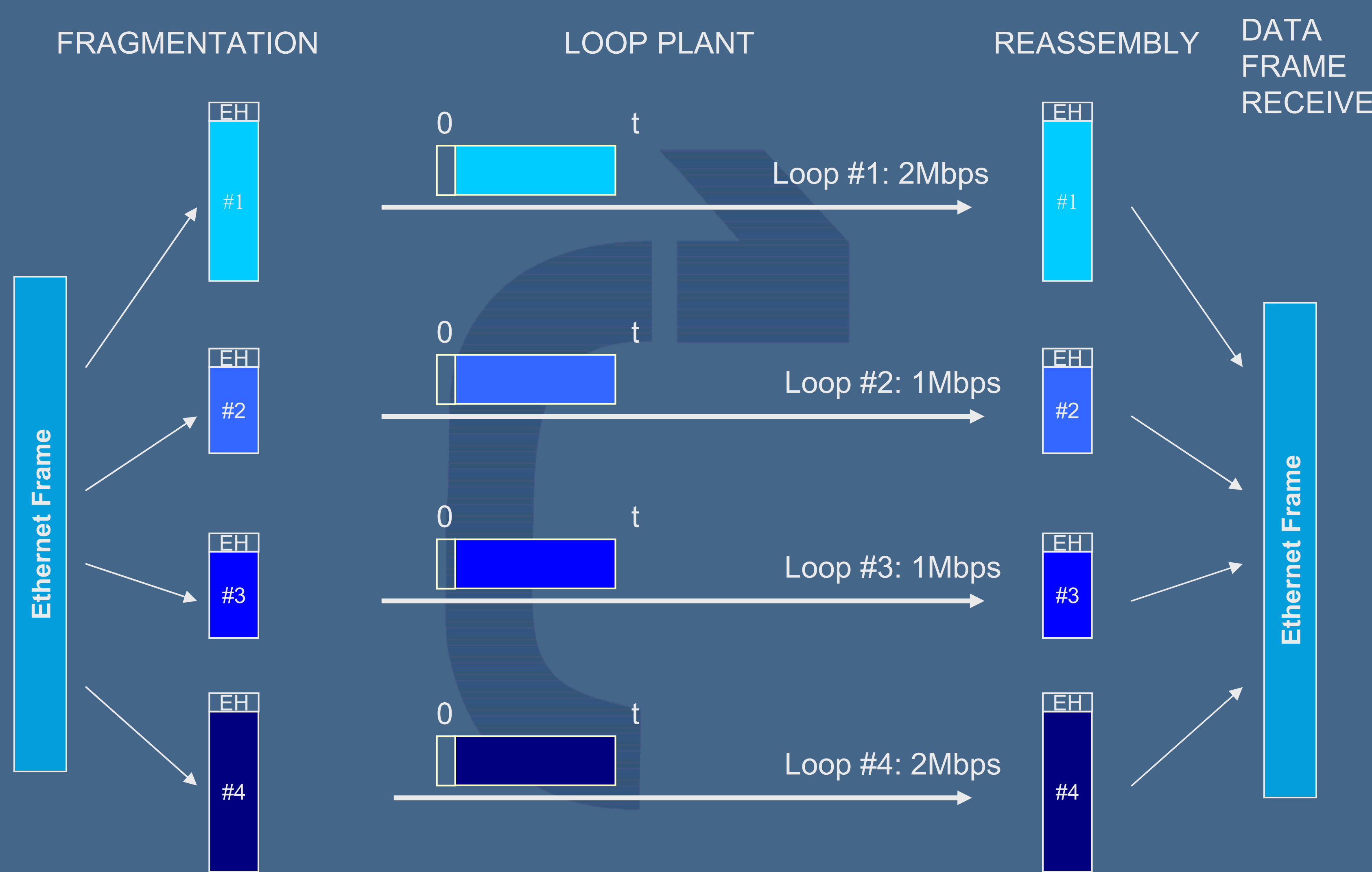
- Meets objective: “Include an optional specification for combined operation on multiple copper pairs”
- PHY Layer protocol for aggregation of up to 32 copper loops into one logical Ethernet link
- Independent of PMD layer flavor of DSL
- Scalable and resilient to loop failures



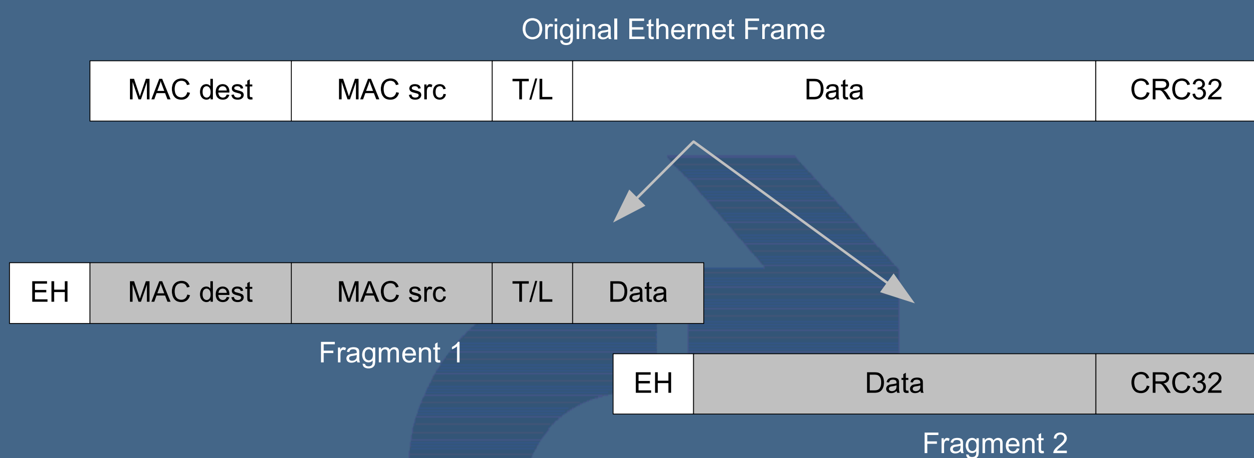
Protocol stack



Fragmentation & Reassembly

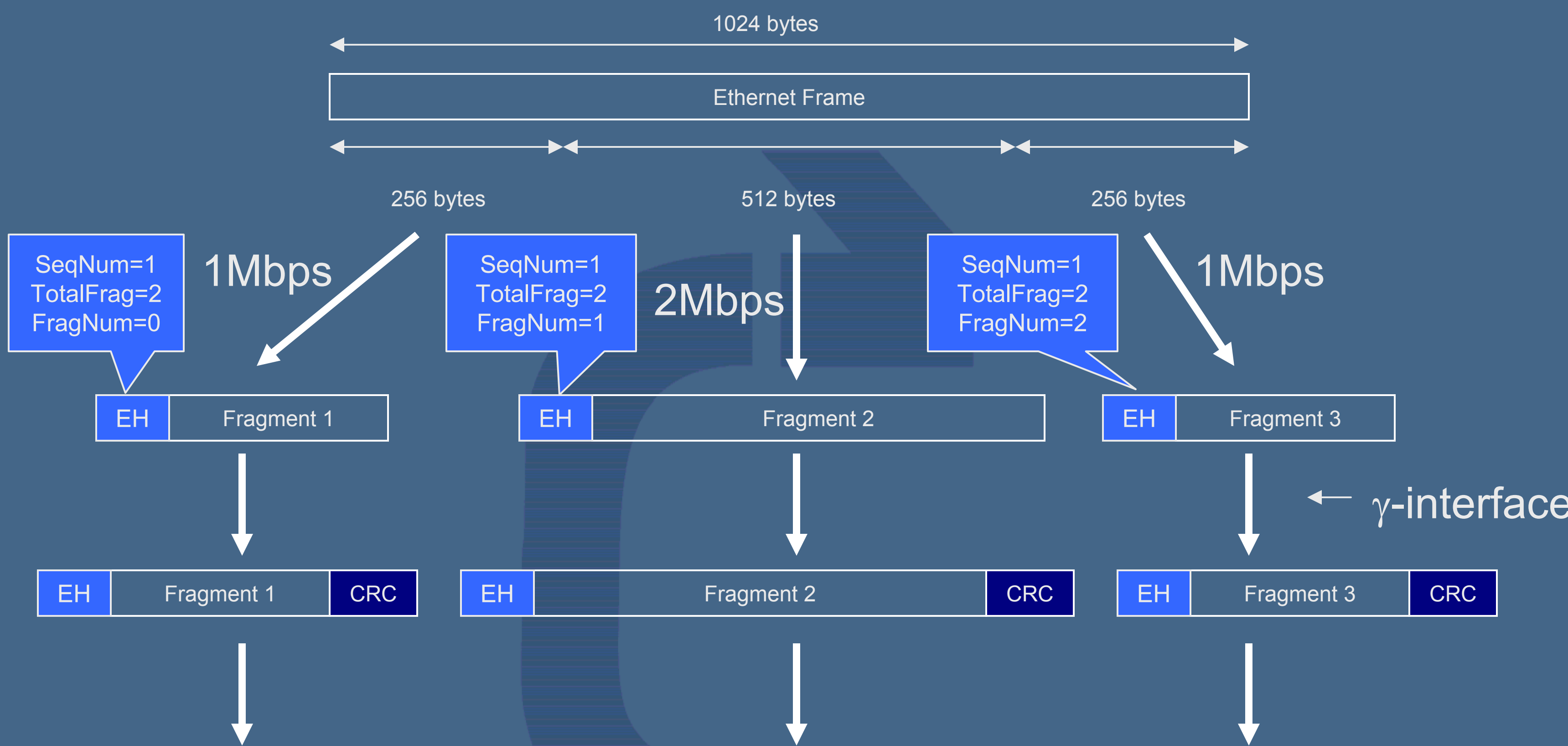


EFM Protocol Encapsulation



- EFM Header (EH)
  - SeqNum - frame sequence number (10bit)
  - TotalFrag - # of other fragments that belongs to this Ethernet frame (5bit)
  - FragNum - fragment number (5bit)
- Underlying PTM-TC sublayer (if applicable) provides
  - HDLC framing
  - 0xFF 03 header (Could be used?)
  - CRC checksum (Some error protection is a requirement).

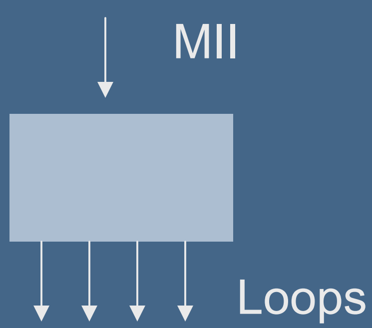
Example



- It does not matter which ports are connected to which, the protocol header implicitly determines how they are to be reassembled

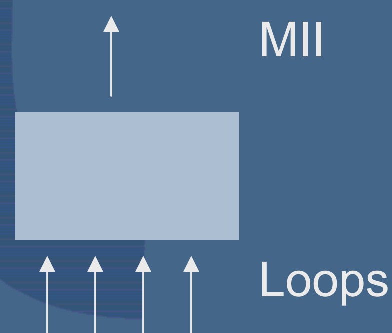
Fragmentation Procedure

- Ethernet frame from MAC layer
  - Determine N, the number of currently functional loops (if some loops are down, N will be smaller than the number of ports)
  - Slice up the frame into N fragments, each with a length according to line rates
  - Add EFM Header to all N fragments
  - Set SeqNum to SeqNum+1 from last frame sent
  - Set TotalFrag to one less that the number of loops (N-1)
  - Set FragNum to indicate fragment number of each fragment (it does not matter which fragment of the frame is sent on which loop)
  - Hold off transmission until no backpressure from any PTM-TCs, then send all N fragments in parallel across the N loops
  - In PTM-TC sublayer, CRCs are calculated and inserted on all N loops



Reassembly Procedure

- CRC is checked on each loop (PTM-TC)
  - if error, fragment is discarded
- Original Ethernet frame is reassembled
  - Using FragNum, TotalFrag, and SeqNum in the EFM Headers
- If a fragment is received with SeqNum out of sequence the fragment is discarded



Resiliency

- A transmitter can in real time determine which of the connected loops are to be used (based on DSL link failures or bit error levels)
- The EFM header allows the fragmentation to only take place on a subset of the connected loops. The EFM header implicitly defines how many and which loops were used.
- The reassembly process can determine how many loops were used on a packet by packet basis

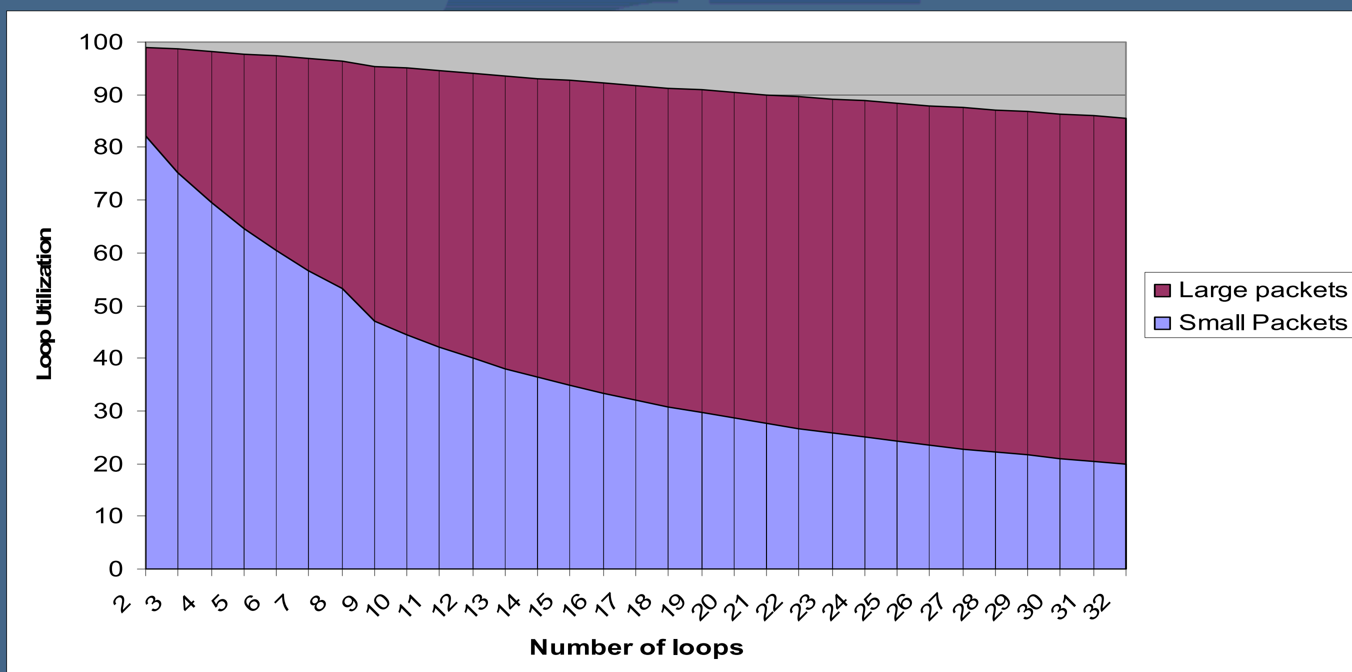
Issues

- Depending on underlying packet encapsulation scheme:
  - If HDLC, fragmentation can optionally compensate for HDLC skew (covered in backup slides)
  - If something else, some form of error protection on (at least) the EFM header is a requirement
- Number of supported loops
  - Consensus in Raleigh showed support for 32 loops. (Backup slide addresses overhead with less loops).
- Differential Latency supported
  - Size of SeqNum parameter, amount of (other) overhead in fragments, and top speed of loops determine how large a differential latency can be supported

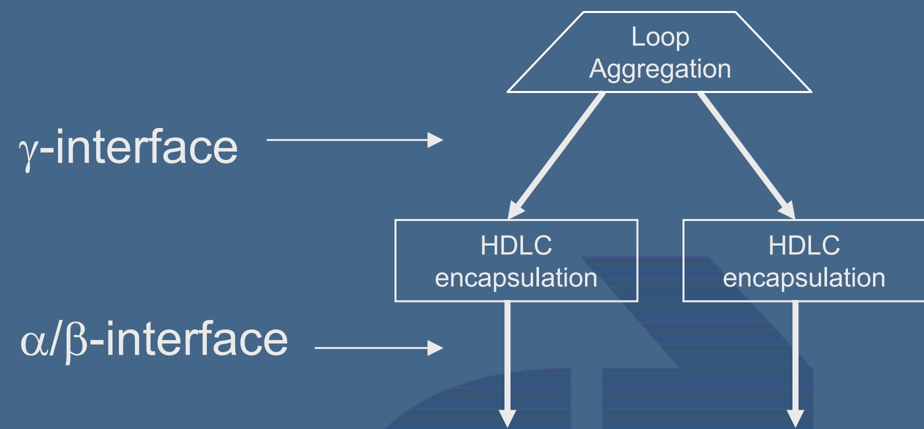
Backup Slides

How Many Loops?

- Maximum number of loops that can be aggregated is implementation specific, but we need to pick a protocol limit!
- 8-32 loops, what does it cost?
  - More loops means smaller payloads per loop. I.e. more relative overhead.
  - More loops mean more bits needed in EFM Header:
  - N <= 8 loops means 2 bytes EFM Header
  - 8 < N <= 64 means 3 bytes EFM Header
  - N>32 should not be considered! (MDIO support)
  - No buffer cost (other than linear scale)



The “HDLC Skew” issue



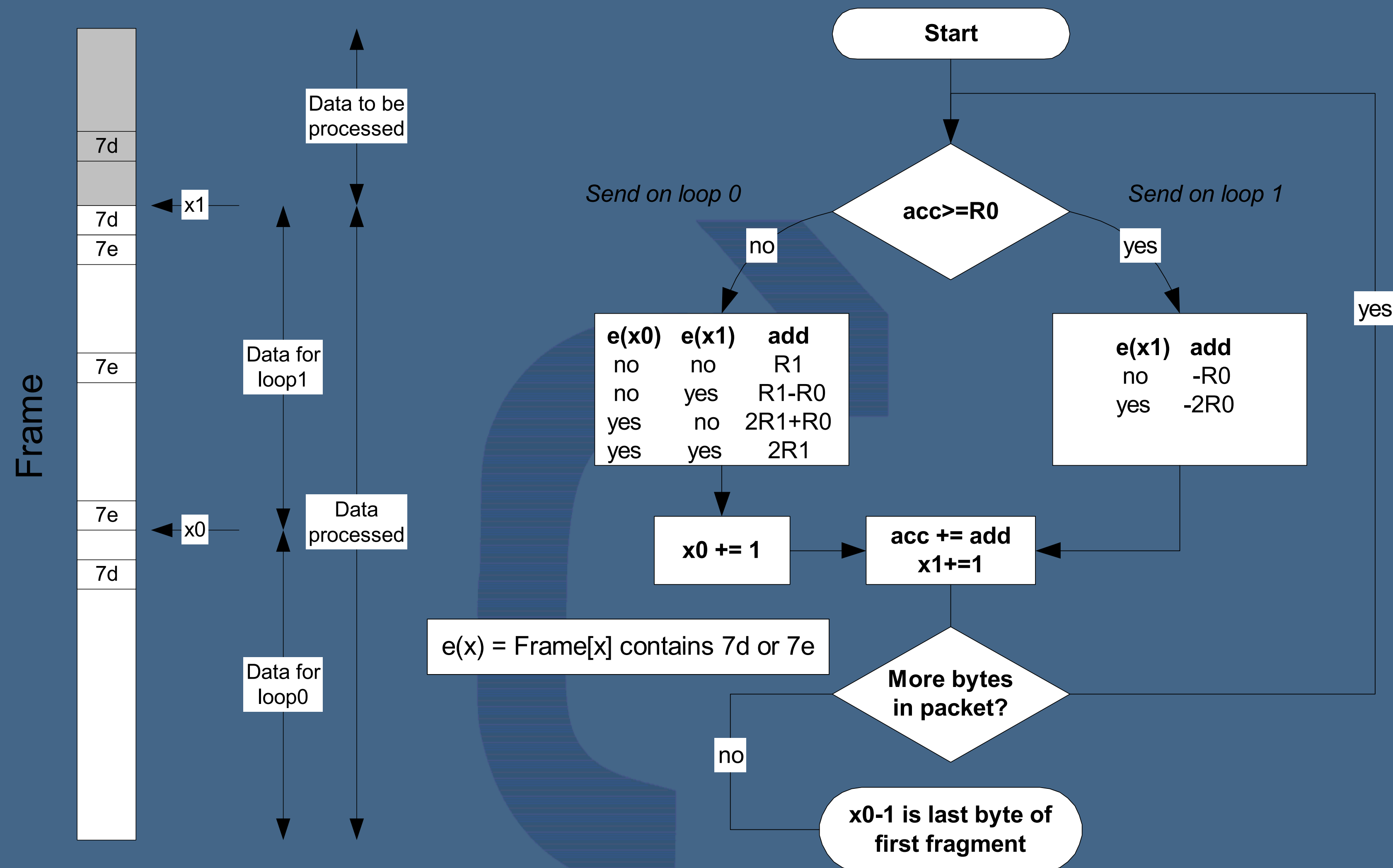
- ITU defined Packet Transfer Mode (PTM) defines use of byte synchronous HDLC encapsulation
- HDLC encapsulation makes the data stream longer than it was:
  - Data byte 0x7E is encoded as 0x7D-5E (two bytes)
  - Data byte 0x7D is encoded as 0x7D-5D (two bytes)
- Skew is dependant on content of packets
- Unless HDLC skew is compensated for, Loop Aggregation layer will not know the real transmission rates
- Can lead to lower loop utilization

Packet Mux, Fragmentation Algorithm

- Fragmentation algorithm can be vendor specific, does not need to be defined in standard
- Fragmentation algorithm can optionally compensate for HDLC skew
- Fragmentation algorithm does not need to be known at receiver, it does not affect interoperability
- The following are examples of possible algorithms that do compensate for HDLC skew and are simple to implement

HDLC Skew Compensated Frag. Algorithm

Loop Aggregation Baseline



- Incremental calculation (only TX end)
  - One pointer parameter (x0, x1,…) per loop

HDLC Skew Compensated Frag. Algorithm

- Algorithm that works for N loops:
  - Initially, and each time a line rate changes:

```
for i=1 to N do
   C[i] = G/R[i]
```

  - where G is the least common multiple (LCM) of R[1], R[2],…R[N].
  - For every packet:

```
Clear all A[i], x[i] ,  i = 1,2,,N
for each byte in the frame do
{
  Find k where A[k] = min(A[i]) ,  i = 1,2,,N
  for i=k to N do
  {
    if frame content in x[i] contains 0x7e or 0x7d
      f=2
    else
      f=1
    A[i]   += f * C[i]
    A[i+1] -= f * C[i+1], if i<N
    x[i]   += 1
  }
}
```

  - where the x's are the intersection pointers