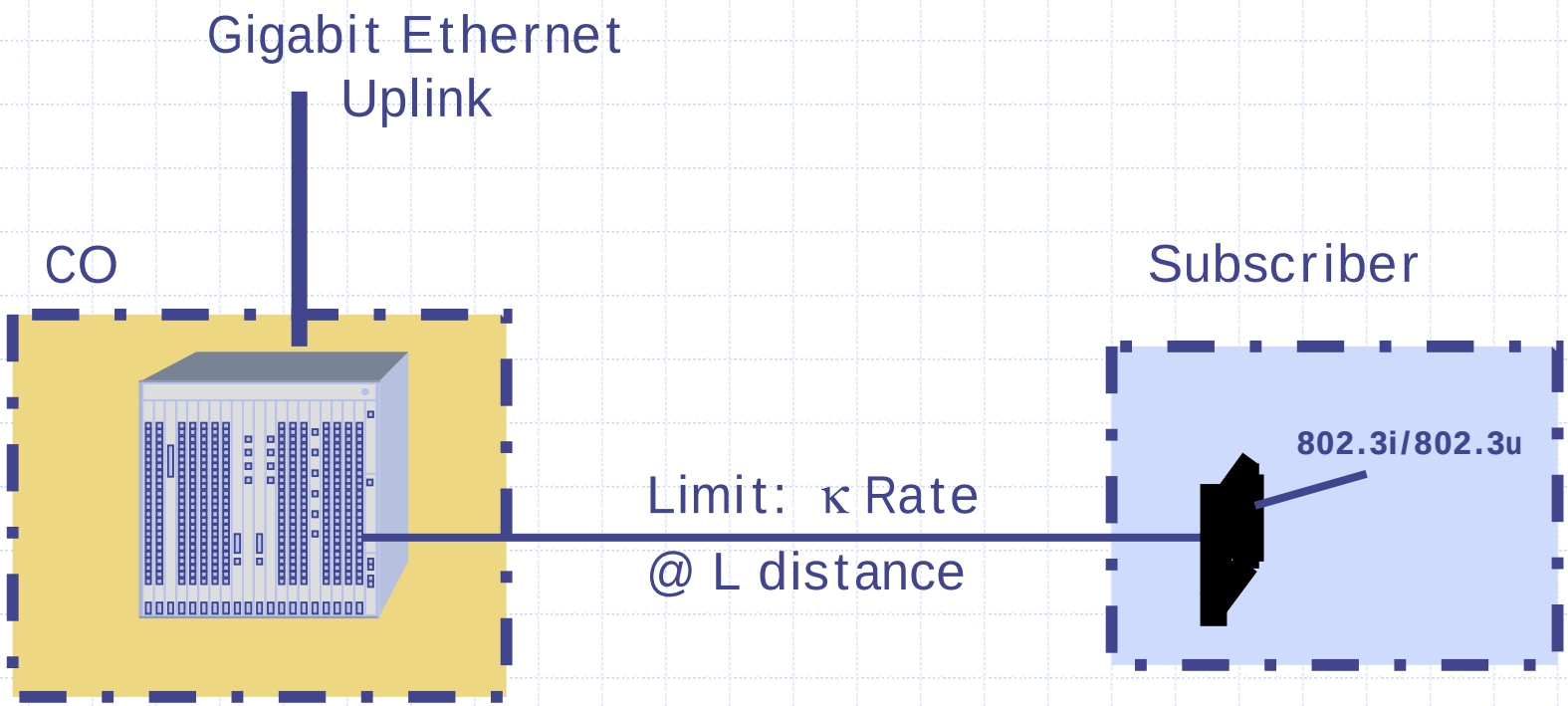
Multi-pair Bonding Enhancement via Improved Sub-packet Numbering

Steve Jackson

HATTERAS NETWORKS

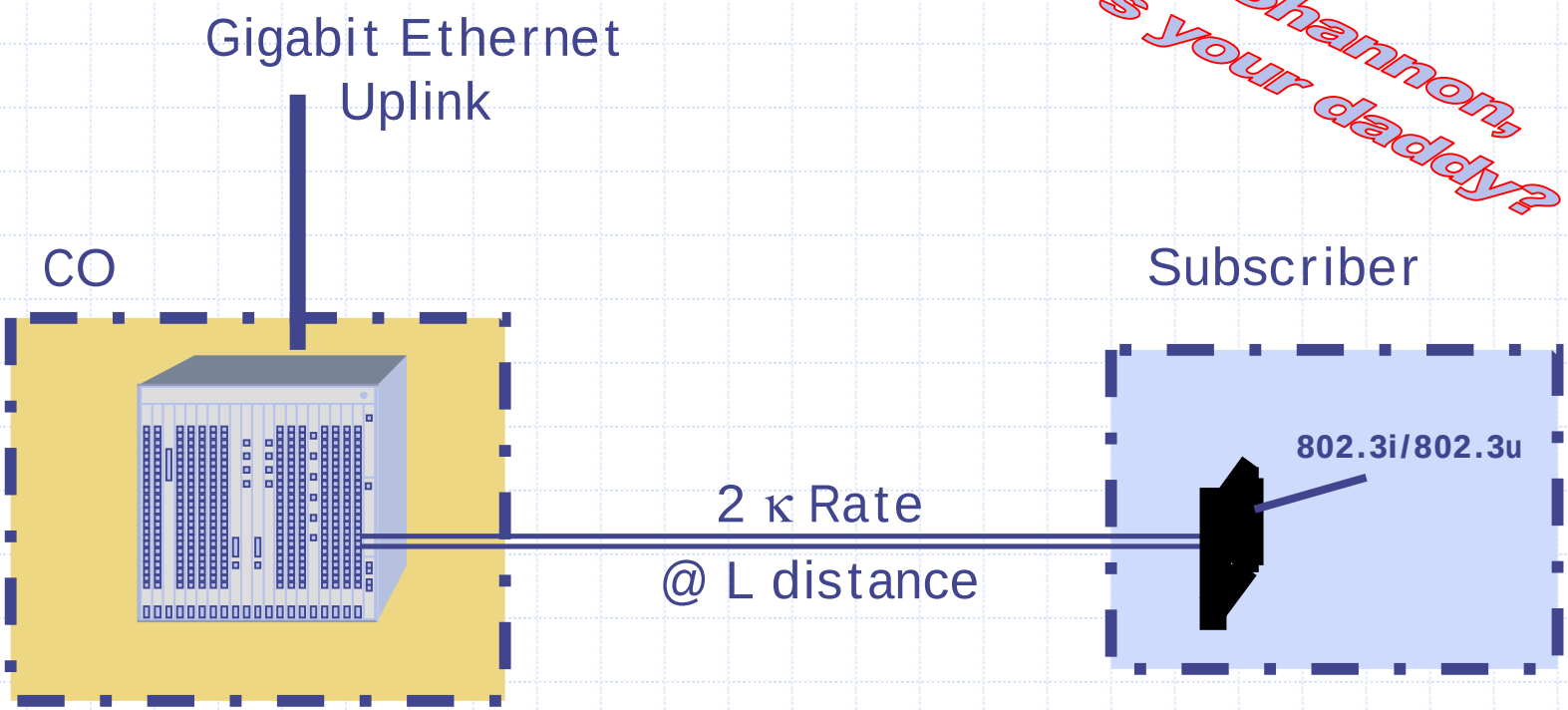
Single Copper Pair



Not always “Ethernet” speed …

Two Copper Pairs

hey Shannon,
who's your daddy?



Real “Ethernet” speed!

Great Idea!

A path consists of a working and enabled Cu loop

If you take N known paths, you can treat them as separate links by adding a simple “bonding layer” to coordinate their intelligent use

This is a description of such a “bonding layer” on the transmit side

There is a matching process on the receive side

One ‘way’ of the full-duplex link is shown, for clarity

HATTERAS NETWORKS

Sub-packet Multiplexing Works!

- **Sub-packet multiplexing scales well**
  - Enables ‘hitless’ add and drop of PHY links
  - Addresses requirement to bond from 2 to 32+ pairs
  - Independent of link specifics; no need to ‘sync’ to new rate
- **Even better …**
  - Simpler sequence numbering makes for simpler system
  - Simplified fragment header provides CRC for itself
  - Lower overhead than variable-length “EFM Header” + CRC32
  - Latency minimized through use of managed FIFOs
  - Minimizes the MII boundary limitation (streaming data)
  - Allows for vendor differentiation while maintaining interoperability

HATTERAS NETWORKS

Fosmark Transmit Proposal

Fosmark Sub-packet Bonding Transmit:

- Determine the number of loops (N)
- Partition Frame into N parts depending on link speeds
- Determine sequence number and fragment number for each part
- Set sequence number & fragment number in EFM Header
- Hold off on transmission til no back-pressure
- Transmit to PTM-TC layer
- PTM-TC layer responsible for CRC on sub-packet (CRC32 for whole sub-packet)

HATTERAS NETWORKS

Fosmark Receive Proposal

Fosmark Sub-packet Bonding Receive:

- Check validate CRC of sub-packet at PTM-TC
- Determine next sequence number expected on any active loop
- Grab sub-packet with that sequence number from all loops with it, waiting if nec.
- Figure out if entire frame received by keeping track of number of fragments
- When all fragments available reassemble in order of fragment number
- Pass frame to MAC after reassembly

Fosmark Proposal

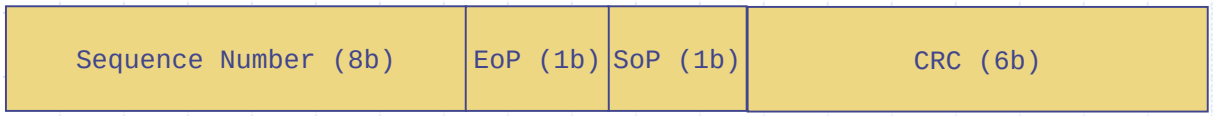
Good points:

- Receive doesn't have to know about transmit, nor the # of lines used
- Allows vendor specific algorithms for product differentiation

Bad points:

- Hard limit on the number of loops supported (protocol header)
- Hold and wait strategy
  - must hold transmission until no backpressure on any loop
- Complexity of two sequence number management
  - per packet, per fragment
- Must compute when all fragments received
- Redundant CRC protection for payload
  - once per sub-packet, once per packet
- Extra overhead!
- Requires CRC to be in PTM-SC to cover HDLC encapsulation

Alternate Proposal (1) Transmit

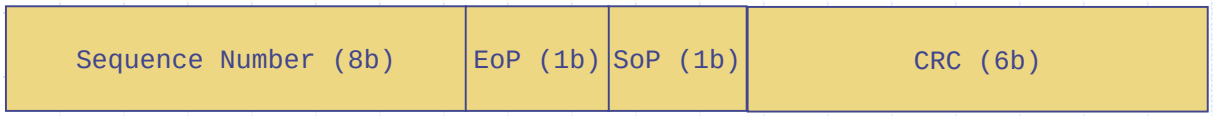


Alternate Sub-packet Bonding Transmit:

- Choose a loop
  - algorithm need not be specified
- Choose number of N bytes to transmit on that loop
  - algorithm need not be specified
- Increment and set fragment sequence number in EFM Header
- Set EOP & SOP in EFM Header as appropriate
- Set CRC in EFM header
- Transmit to PTM-TC layer

HATTERAS NETWORKS

Alternate Proposal (1) Receive



Alternate Sub-packet Bonding Receive:

- Validate CRC of header above PTM-TC
- Determine next sequence number expected on any loop
- Wait if necessary
- Grab that fragment
  - If EoP, then pass up to MAC and expect SoP next
  - If unexpected SoP, then previous frame lost; reset buffer

Alternate Proposal

Good points:

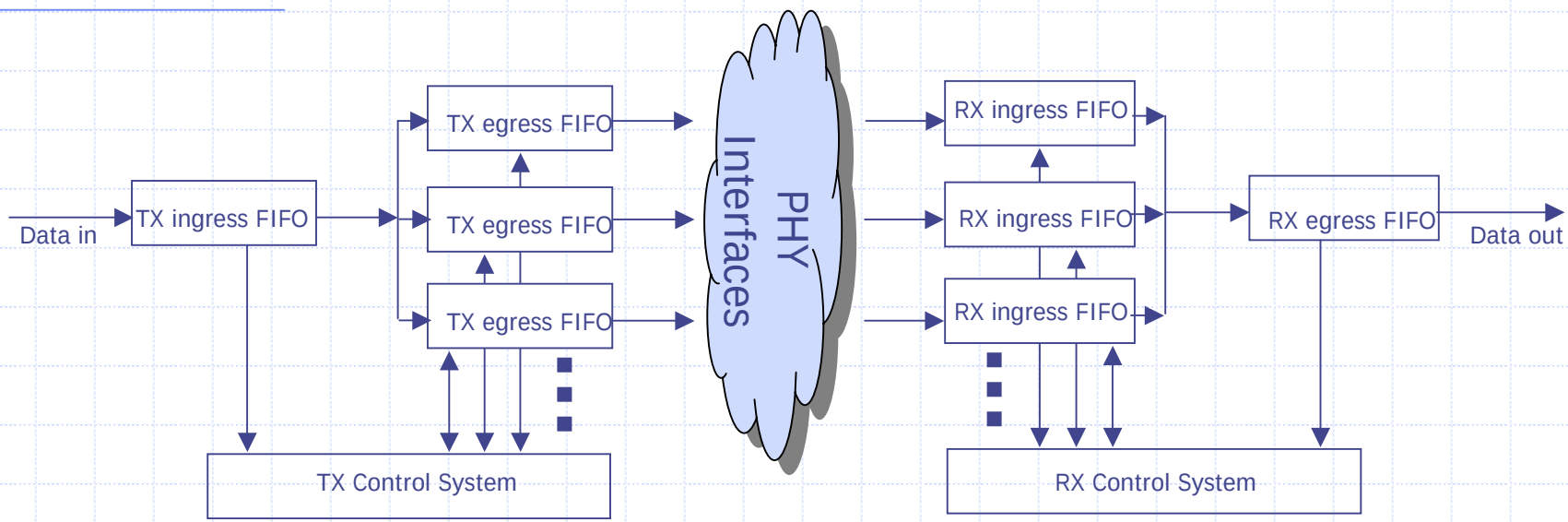
- Receive doesn't have to know about transmit, nor the # of lines
- Allows vendor specific algorithms for product differentiation
- Supports greater # of loops limited only by sequence "wrap"
- No backpressure-induced hold and wait latency
- Less complexity with single sequence number
- Efficient CRC protection on header only
  - Ethernet payload protected by CRC on frame
- Less overhead
  - 2B header per sub-packet + CRC per frame vs.
  - 3B header per sub-packet + 2B CRC per frame

Bad points:

- ?

HATTERAS NETWORKS

System example



Control Systems:

- **Generate/decode fragment header (and CRC for header)**
- **Generate/decode unique sequence number for each packet fragment**
- **Appends/strips fragment header in FIFO**
- **Monitors FIFO status, controlling latency**