Back to the future: using TAS and preemption for deterministic distributed delays

Michael Johas Teener

mikejt@broadcom.com / m@jot.us

Agenda

- **Objectives of this proposal**
- **A bit of history**
  - A bit more history
- **Using TAS for peristaltic transport**
  - Optimizing peristaltic transport using preemption
- **Scalable delays with link speed**
- **What needs to be done**
- **So, what did I get wrong this time?**

Objectives

- **Deterministic distributed delays for all streams**
  - really, this time I mean it!
  - queues distributed between bridges evenly
- **Scalable delays with link speed**
  - 10x shorter delays for Class A traffic over links with a 10x speed increase
- **Multiple traffic classes**
  - Equivalent to Gen 1
  - This time we will make sure the "observation interval" is programmable!
- **Use Gen 1 SRP or future SRP/IS-IS**

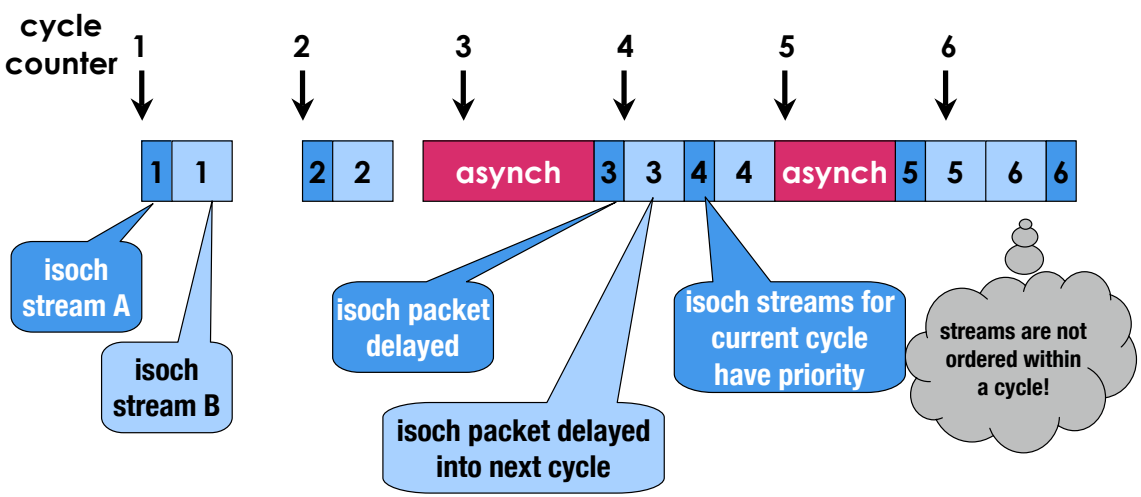
A bit of history*

- **The original "Residential Ethernet" proposal**
  - http://www.ieee802.org/802_tutorials/05-March/tutorial_1_0305.pdf
  - http://www.ieee802.org/3/re_study/public/200409/teener_2_0904.pdf

Synchronous pacing

- **Transmitter sends all packets labelled with cycle "n" as soon as possible in that cycle**

I am NOT proposing this, so don't get excited!



Synchronous pacing in the RE proposal

- **All bridges and talkers know the time, and schedule transmissions on "cycle" boundaries**

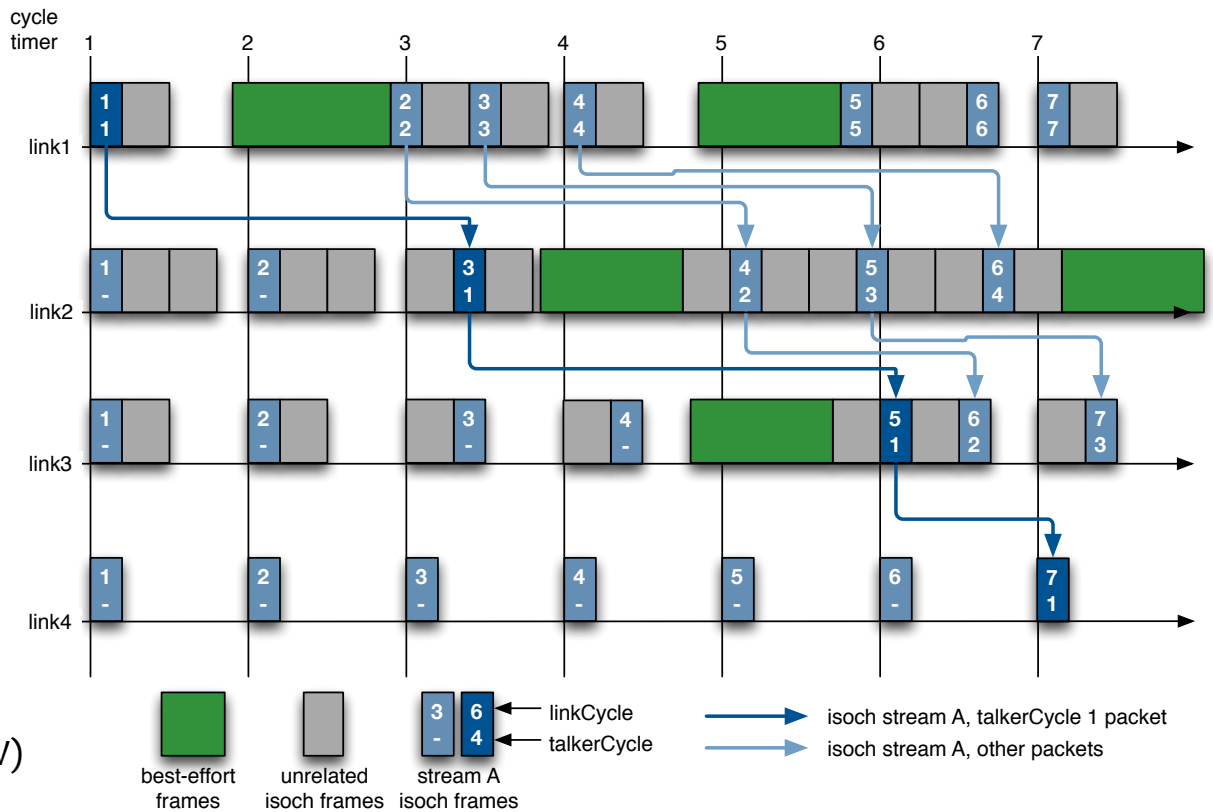
frames are labeled with a talker cycle timestamp and a scheduled cycle

frames are launched as soon as possible after the scheduled cycle

at a bridge, the received scheduled cycle in incre-mented by 2 for 100Mbs links, by 1 for 1G+ if no jumbos

delay is between 1+#hop*2 and 4+#hop*2 cycles for 100Mbs links, 1+#hop and 2+#hop for 1G

(all assuming max 75% isoch BW)



Problems, problems

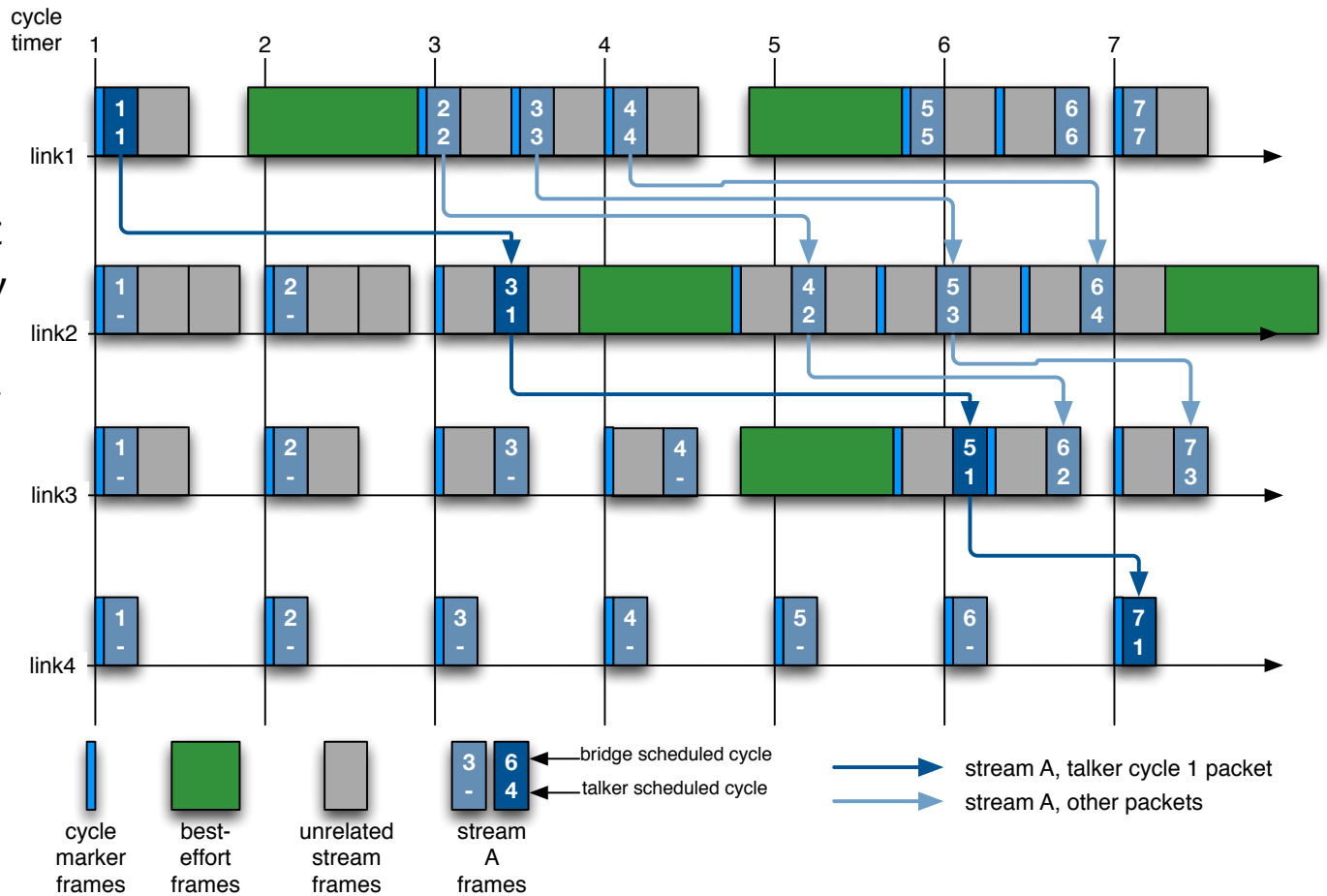
- **Required a new tag with embedded cycle time**
  - NOT acceptable in 2005 (or, most likely, even now)
- **Required hop-by-hop relabeling of frames**
  - time stamps of a crude nature
  - NOT acceptable in 2005
- **Required putting a frame in a different queue based on new label**
  - hmm, that's been done for a long time, hasn't it?
- **Required different queues to be enabled for transmit at different times**
  - hmm, that's time-aware shaping, isn't it?
    - (yeah, yeah, I know I'm elaborating on the concept a bit)
  - NOT acceptable in 2005

Another way to label packets

- **Use an explicit cycle marker frame?**

now frame timer labels are virtual/implicit

same delays, but small inefficiency due to min size cycle marker per cycle

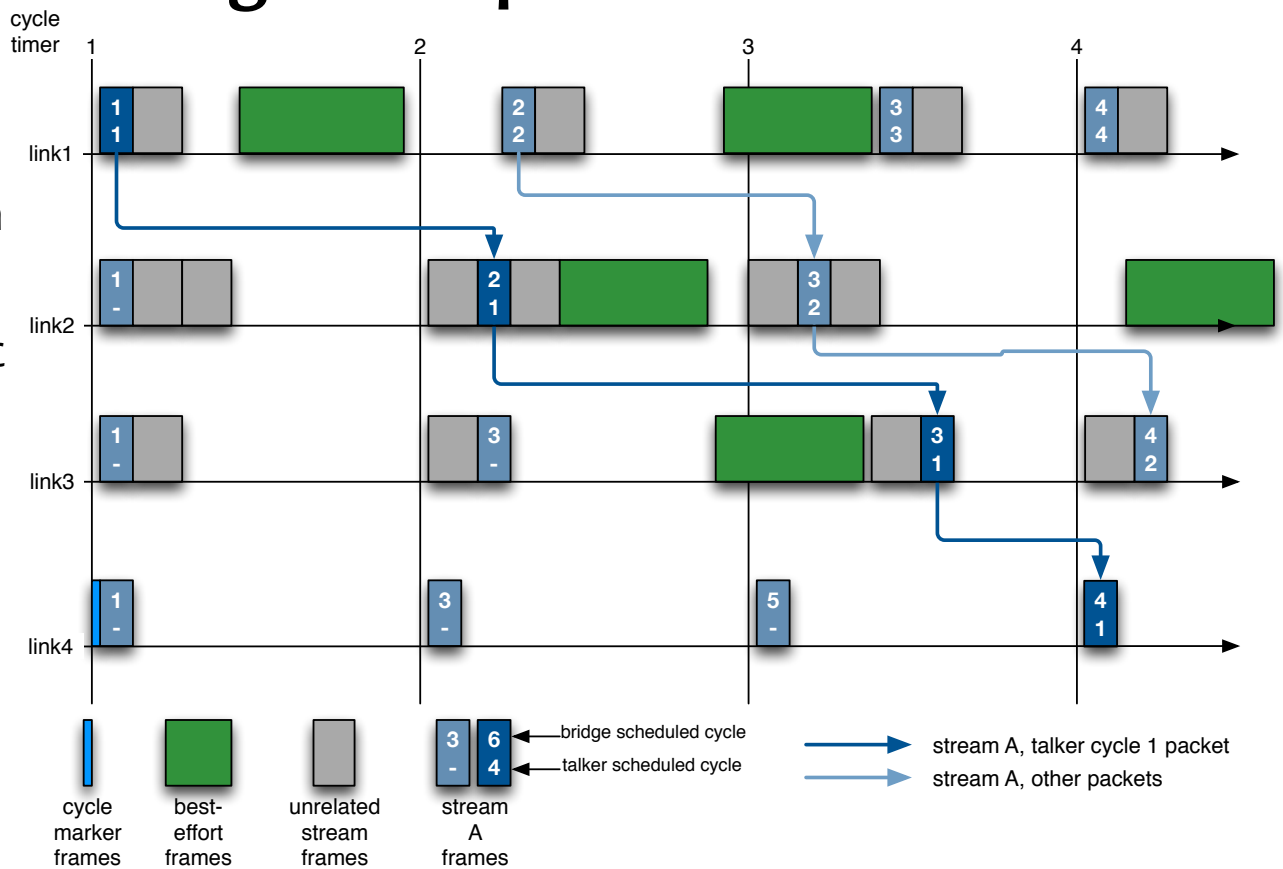


Totally implicit time labeling

- **Ensure that the cycle time is greater than the sum of the longest interfering frame plus all the isochronous traffic**

  now all isochronous traffic will arrive within the same cycle

  if, of course, this traffic is complete queued at the start of a cycle and is the highest priority



Using TAS for "peristaltic transport"

I could have called this "isochronous", but that's got a bad name in 802, so ...

- **Take some of the concepts of the RE approach, but remove the most objectionable parts:**
  - Frames are normal Q-tagged frames, nothing special
  - Distinguished by traffic class, exactly like current scheduled/ shaped traffic
  - Time aware shapers on egress

... but ... (new stuff alert) ...

  - Time aware internal tagging on ingress
  - Two egress queues per traffic class

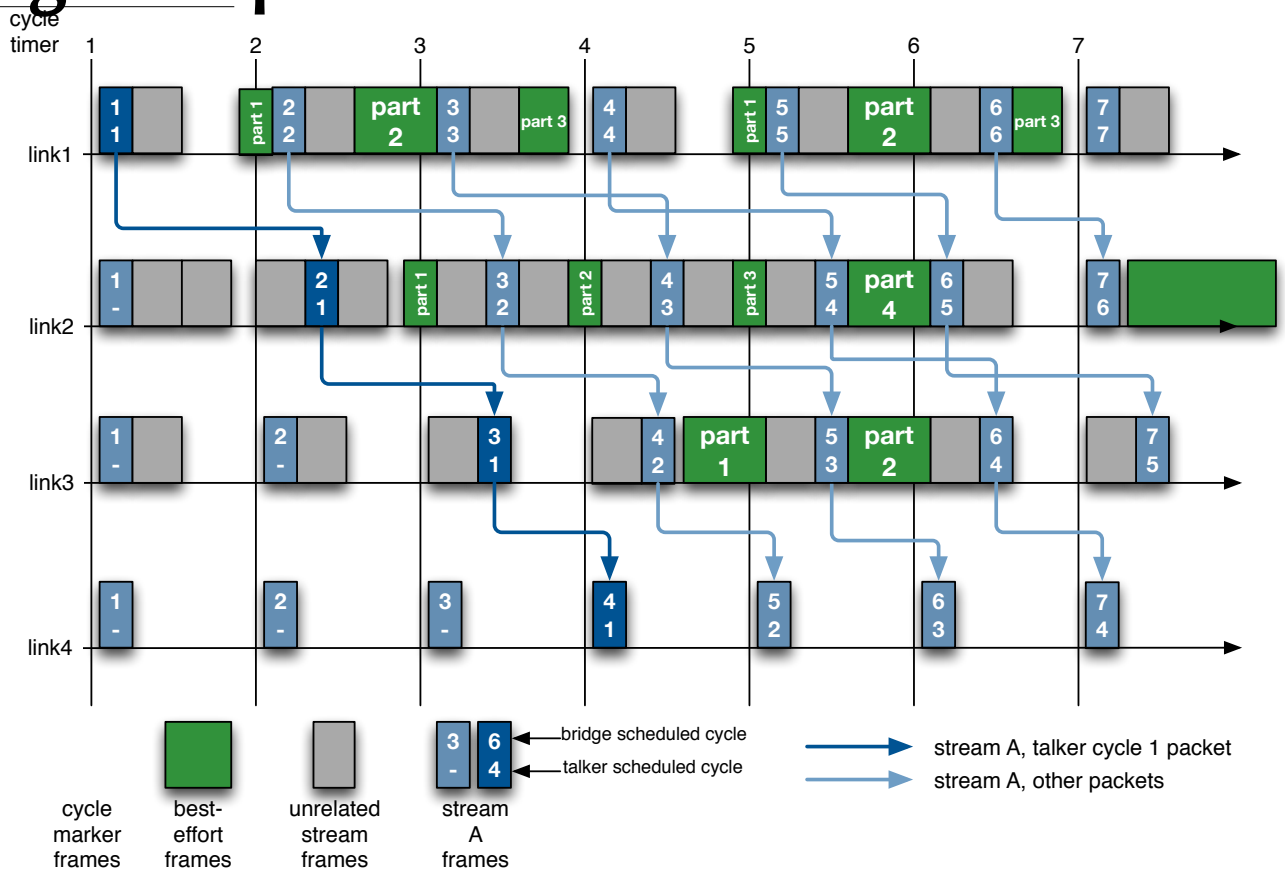
Time-aware tags … two egress queues?

- **Use current time to queue frames into different queues**
  - put frames received in a odd cycle in the even queue
    - and the frames received in the even cycle in the odd queue
- **Even queue has a window the opens up at the beginning of the even cycle**
  - duration of opening could be almost a cycle time
  - odd queue window opens during the start of the odd cycle
- **Many implementation options, but this uses the Qbv concept**

Improve efficiency with preemption

- **Now the cycle time must be greater than the longest interfering *fragment* plus all the isochronous traffic**

if the max isoch traffic is 75% of the available BW, then the fragment could be almost 400 bytes for 100Mbs links



Scalable delays with link speed

- **Scale the cycle time inversely with link speed …**
  - if the 100Mb/s link cycle time is 125µs, then the 1G link cycle time could be 12.5µs … could be any integer divisor
  - this preserves the "bytes per cycle" number, and potentially reduces delays
  - but there is a big complication:
    - if a stream is less than 75Mb/s then SRP may have to assign it to a particular "sub-cycle" (remember the integer divisor?) so that it can be carried on a 100Mb/s link
    - "first come, first served" subcycle assignment may work just fine if lower speed packets are internally assigned a lower priority within a switch, but may add delays for faster streams … needs more thought!
  - I'm sure there's an easier way to do this … but it's late and I'm tired …

What needs to be done?

- **Is this worth while to continue the investigation?**
  - deterministic delays … always
  - delays distributed evenly among switches, so queue sizes are topology invariant
  - worst case delays will always be lower
  - delivery jitter will be MUCH lower and topology invariant
  - SRP operational requirements unchanged for single-speed networks
  - paths with multiple link speeds will require more thought, but scalable delays look reasonably possible
- **Who else is interested?**

Conclusion

- **What's wrong with this idea?**
  - I've been wrong before …
- **Who's willing to help make it right?**
- **Who's willing to blow it away?**

Thanks!