How Many Transmission Selection Algorithms Do We Need?

Christian Boiger
christian.boiger@hdu-deggendorf.de
IEEE 802.1 Interim
May 2013
Victoria, BC

DEGGENDORF UNIVERSITY OF APPLIED SCIENCES

Current Situation

Three Transmission Selection Algorithms

Strict Priority	Credit Based Shaper	Enhanced Transmission Selection
no latency guarantees	latency guarantees (in combination with SRP)	no latency guarantees
no bandwidth guarantees	bandwidth guarantees over a relatively short measurement interval	bandwidth guarantees over a relatively long measurement interval
best effort traffic (only low bandwidth high priority traffic)	bandwidth intensive, loss and latency sensitive stream traffic	bandwidth intensive and loss sensitive traffic

Future Situation

- Four Transmission Selection Algorithms

Strict Priority	Credit Based Shaper	Enhanced Transmission Selection	Time Aware Shaper
no latency guarantees	latency guarantees (in combination with SRP)	no latency guarantees	latency guarantees (engineering necessary)
no bandwidth guarantee	bandwidth guarantee over a short measurement interval	bandwidth guarantee over a relatively long measurement interval	bandwidth guarantee over a short measurement interval
best effort traffic (only low bandwidth high priority traffic)	bandwidth intensive, loss and latency sensitive stream traffic	bandwidth intensive and loss sensitive traffic	extremely latency and loss sensitive (stream) traffic

Additionally 802.3 DMLT/IET might improve latency and/or convergence of the four algorithms

Do We Need Additional Traffic Shaper?

Proposals So Far

- Peristaltic Shaper
  - Similar to Time Aware Shaper
  - http://www.ieee802.org/1/files/public/docs2012/new-avb-mjt-back-to-the-future-1112-v01.pdf

- Burst Limiting Shaper
  - Similar to Credit Based Shaper
  - http://www.ieee802.org/1/files/public/docs2012/new-goetz-CtrDataScheduler-0712-v1.pdf

Why Are Theses Shapers Better?

- Peristaltic Shaper
  - Topology independent latency guarantees
  - Shaper failures easier to detect
  - Scales with speed
  - Addresses current AVB type of traffic

- Burst Limiting Shaper
  - Lower latency than CBS
  - Only low bandwidth high priority traffic
  - Limits the bandwidth
  - Less engineering than Scheduled Traffic (?)
  - Addresses control traffic with latency requirements between Reserved Traffic (~250µs/hop) and Scheduled Traffic (~3µs/hop)

This assumes that the proposed concepts work!!!

Expressed Needs/Goals

- Less engineering (knowing that this increases the latency and reduces the possible topologies)
- Higher transmission periods than AVB Gen1 (with low latency)
- "Shaper" for DMLT mechanism
- Replacement for CBS
- What to do with asynchronous traffic?

How do the proposed solutions perform?

Burst Limiting Shaper

- The following slides show some examples with the BLS

- My assumptions on how this shaper works might be wrong, I used the following equations. They are slightly different than the one in this presentation (http://www.ieee802.org/1/files/public/docs2012/new-goetz-CtrDataScheduler-0712-v1.pdf) (no margin, slope calculations seemed to be wrong in the presentation)

  Parameters:

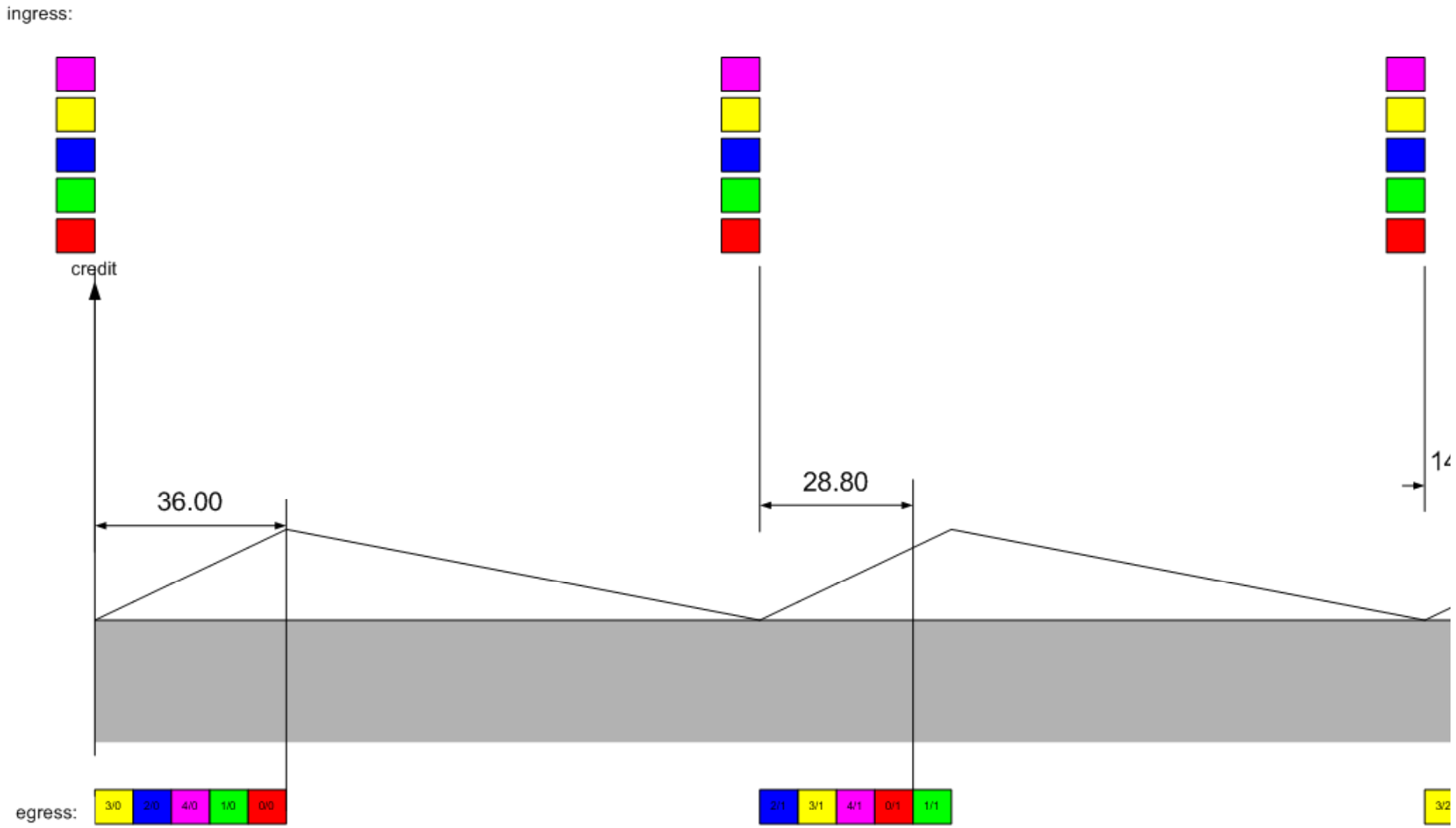
  - leakRate = allocatedBytes * 8 / interval;
  - idleSlope = -leakRate
  - sendSlope = (transmitrate - leakRate)
  - maxlevel = leakRate * interval
  - resumelvl = 0.1*maxlevel;
  - credit + (sendSlope * (packet+IPG) / transmitrate)
  - credit + (idleSlope * (packet+IPG) / transmitrate)

Burst Limiting Shaper

Additional parameters used in the following simulations:

- All streams have a 125 µs transmission period
- Transmission rate = 100 Mbit/s
- Packet size = 70 byte (+ 20 byte)
- White frames = interfering non high priority frames

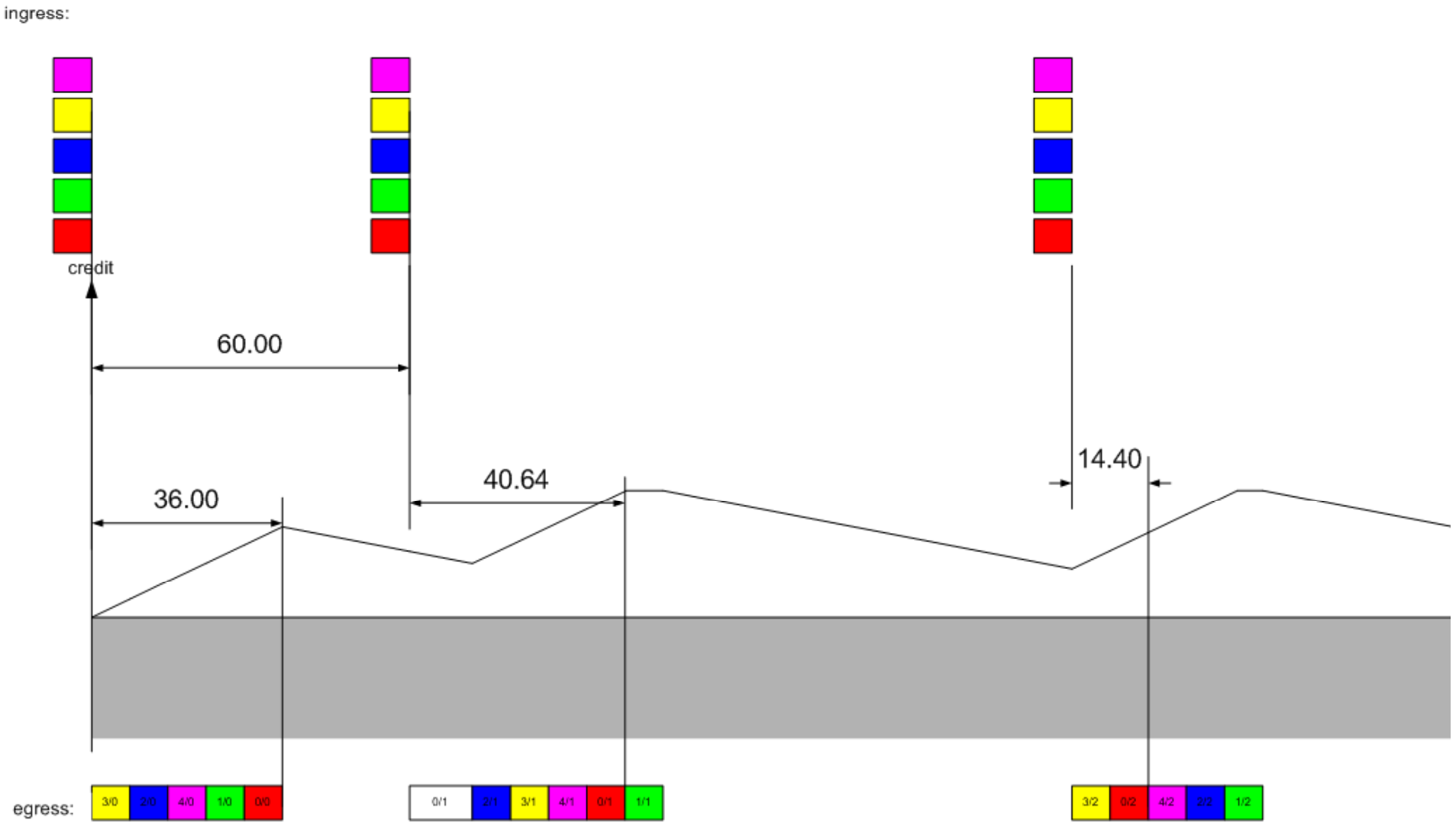
Burst Limiting Shaper (Example 1)



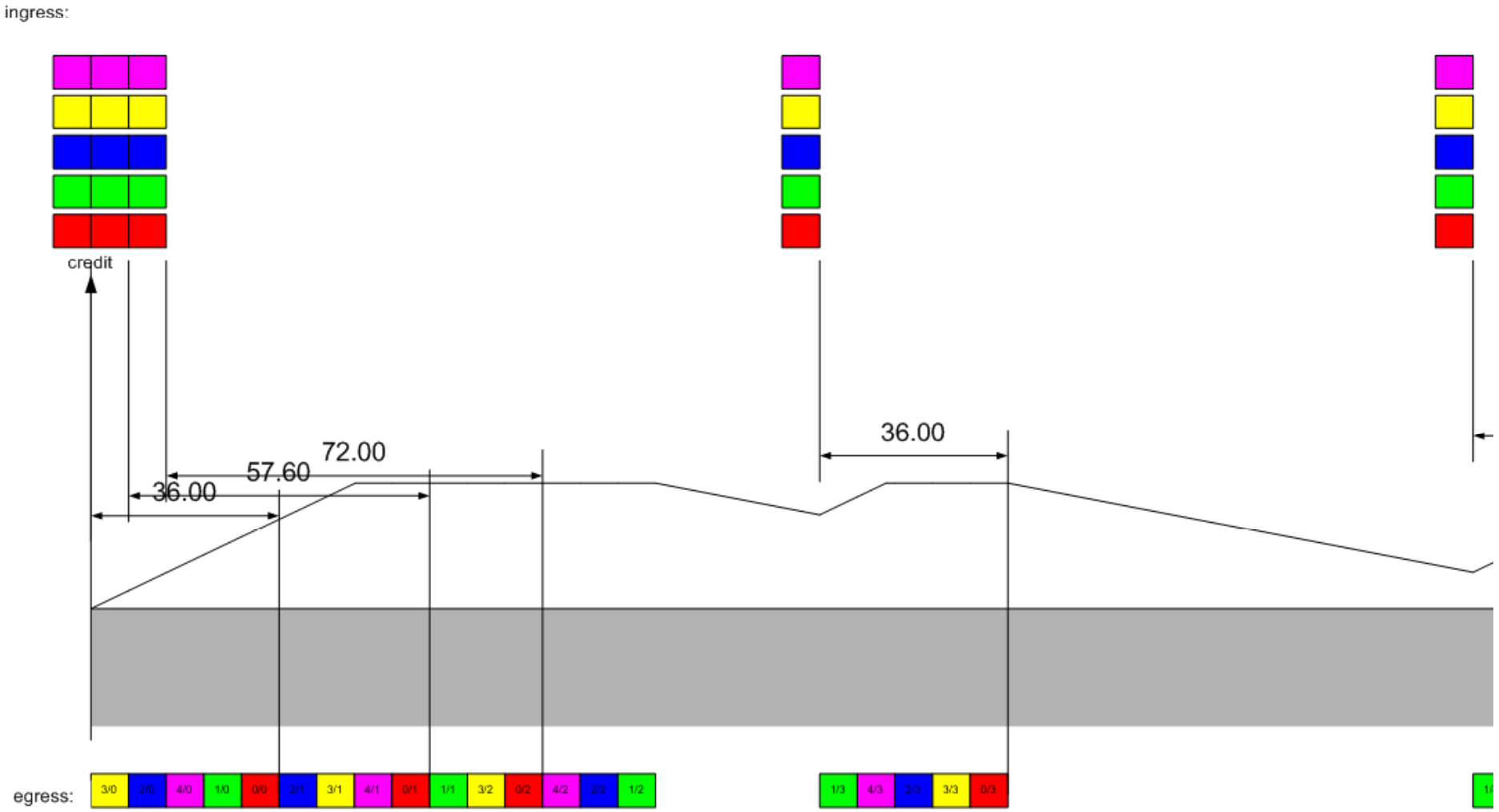
Burst Limiting Shaper (Example 1)

- The result in example 1 shows that there is no difference between the BLS and strict priority in a “normal” operation mode
- The shaper has no effect on the traffic
- So when does the shaper start to spread out the frames?

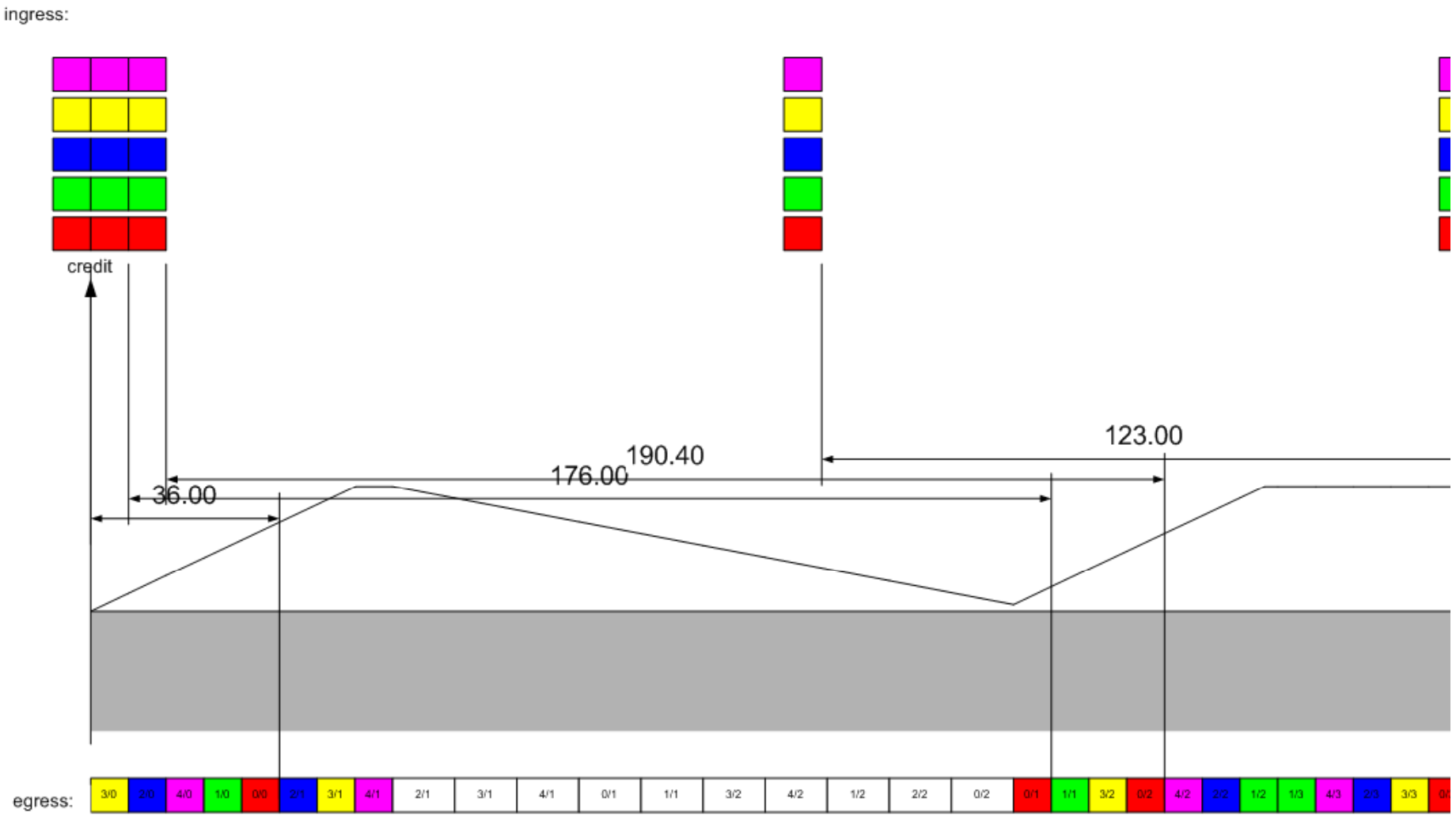
Burst Limiting Shaper (Example 2)



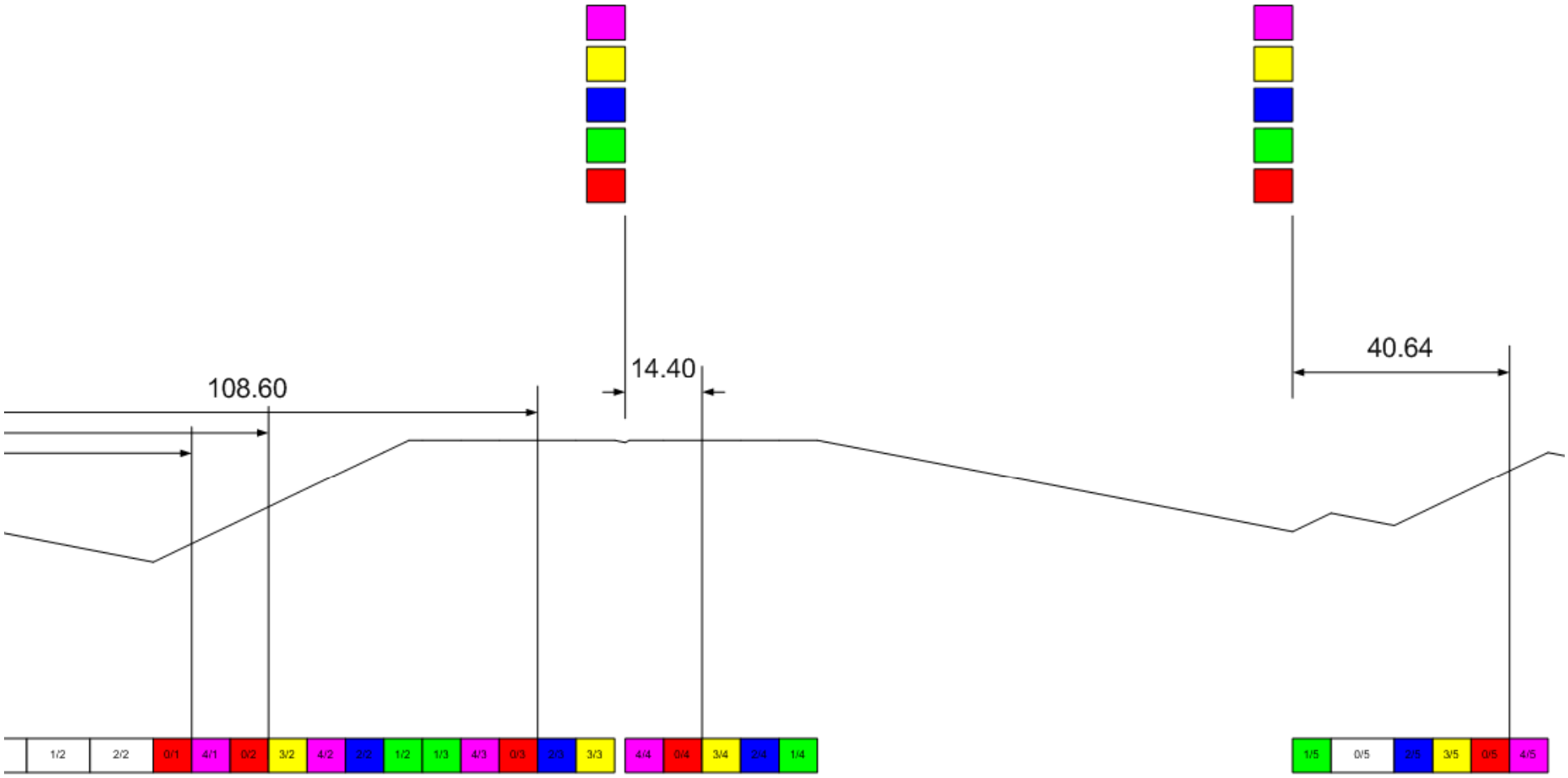
Burst Limiting Shaper (Example 3)



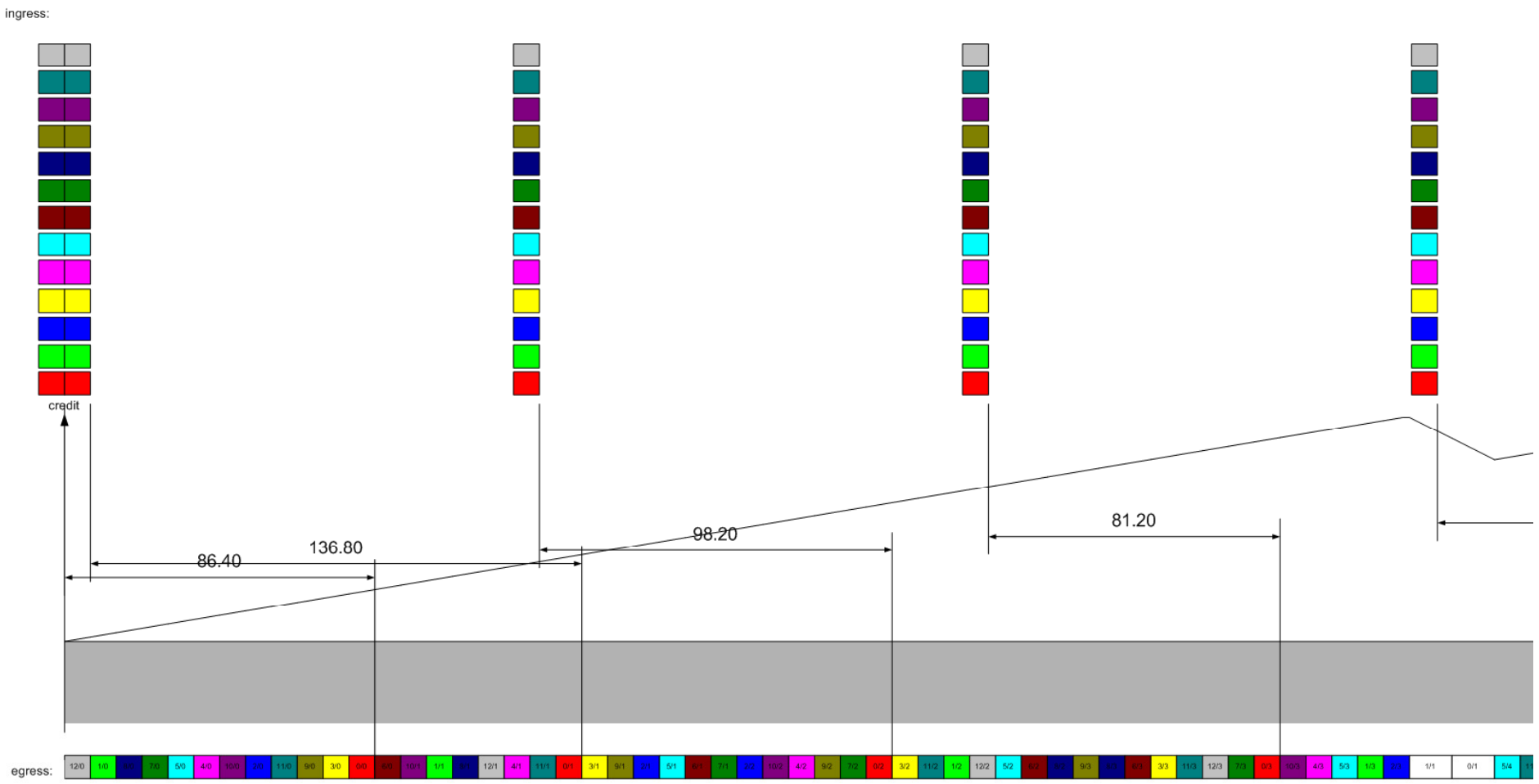
Burst Limiting Shaper (Example 4)



Burst Limiting Shaper (Example 5)



Burst Limiting Shaper (Example 6)



Inverse Priority Mode

- During the „inverse priority mode“ the high priority traffic has the lowest priority.
- But the traffic is not blocked, i.e. the shaper is not limiting the bursts as long as there is no other traffic.
- Therefore the down stream bridges are not protected from high priority streams especially other high priority streams which interfere with such a bursting bridge are not protected.

Conclusion

My understanding so far is:

- The operation in the "inverse mode" should not happen during the normal operation (I don't think that this assumption is right, but assumed it is).
- The "inverse mode" therefore should protect from malfunctions.
- But I think the "inverse mode" makes the whole concept completely unpredictable (e.g. if there is not enough low priority traffic, the port stays in the "inverse mode").
- Additionally the "inverse mode is not protecting the downstream bridges in the absence of low priority traffic.

Would this meet our goals?

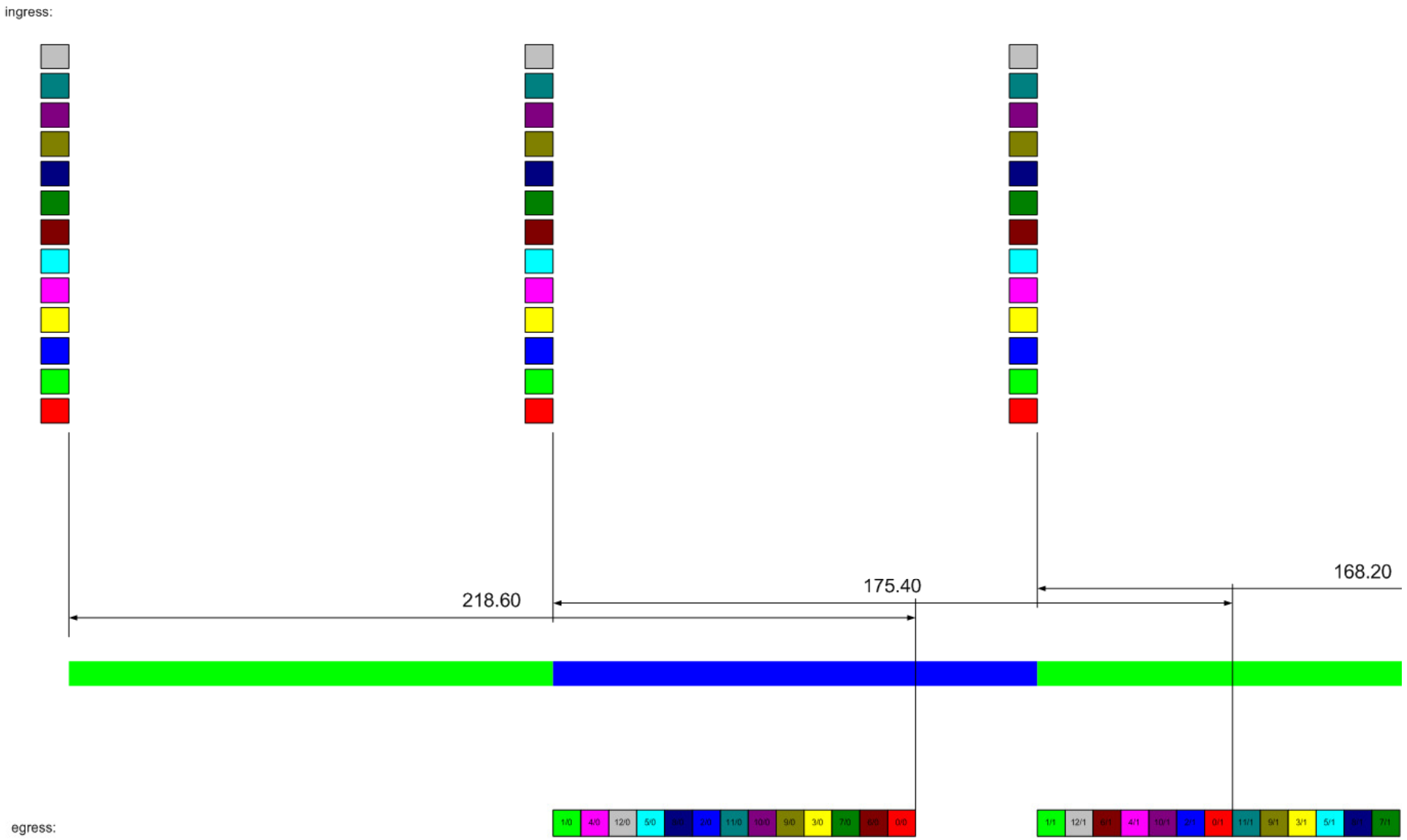
- It was intended that the BLS protects networks in overload situations (this was my impression so far).
- It seems that for this goal, the shaper should be much more restrictive, e.g.:
  - Block traffic in order to really protect the downstream bridges from an upstream overload.
  - Delete frames in an overload situation to get back into the normal operation mode

Peristaltic Shaper

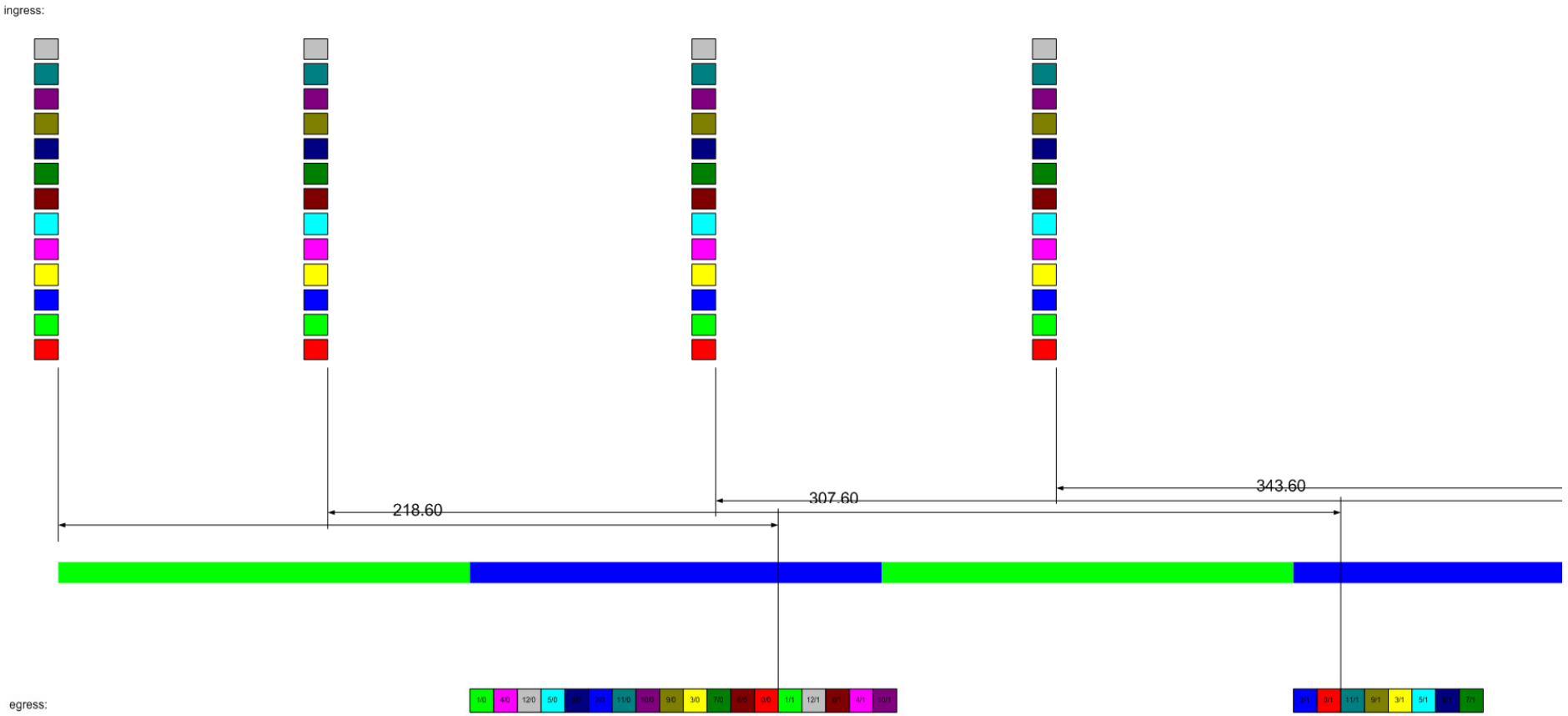
- The following slides show some example simulations of the Peristaltic Shaper

- Again, my assumptions on how this shaper works might be wrong, so please correct me, if I misunderstood something.

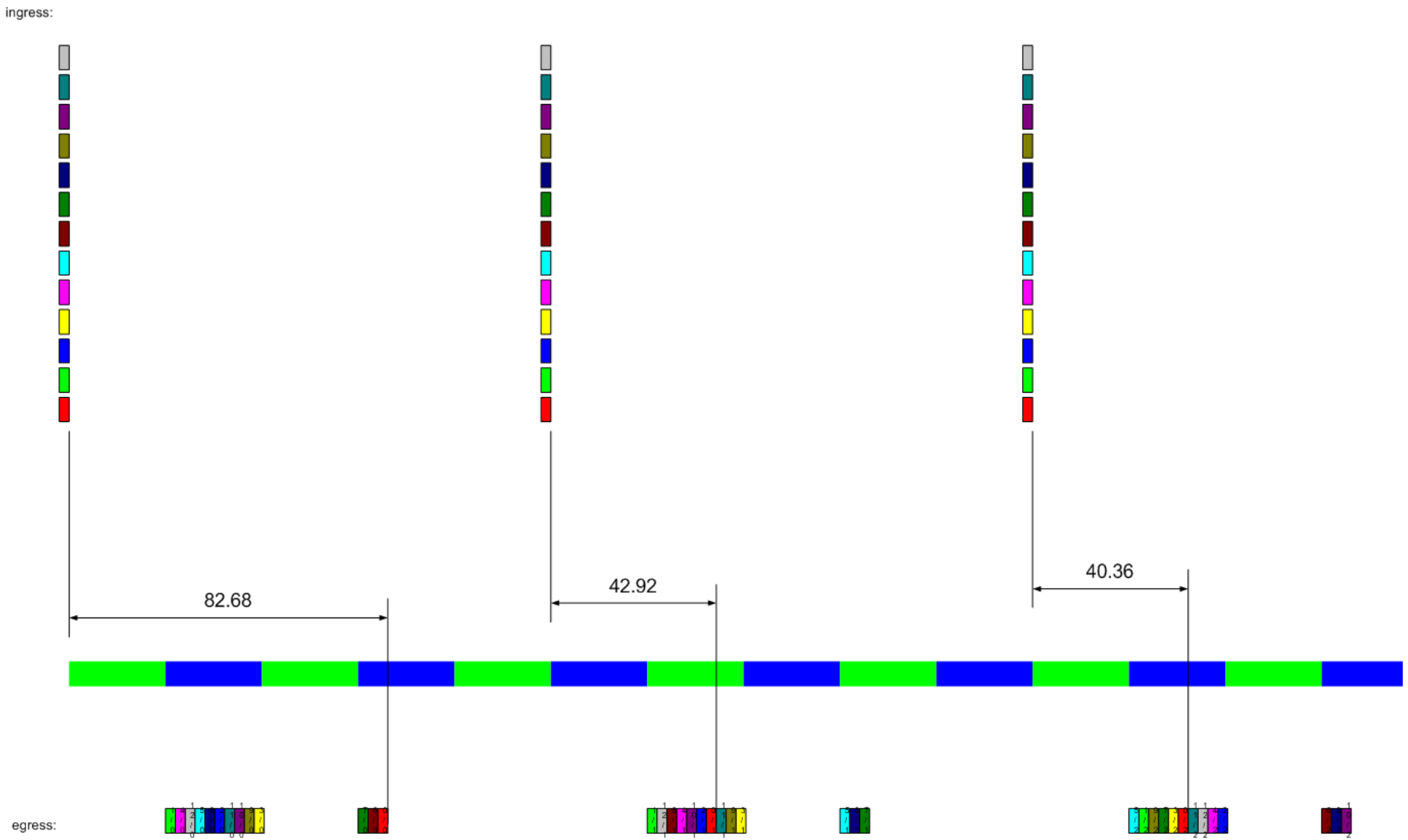
Peristaltic Shaper (Example 1)



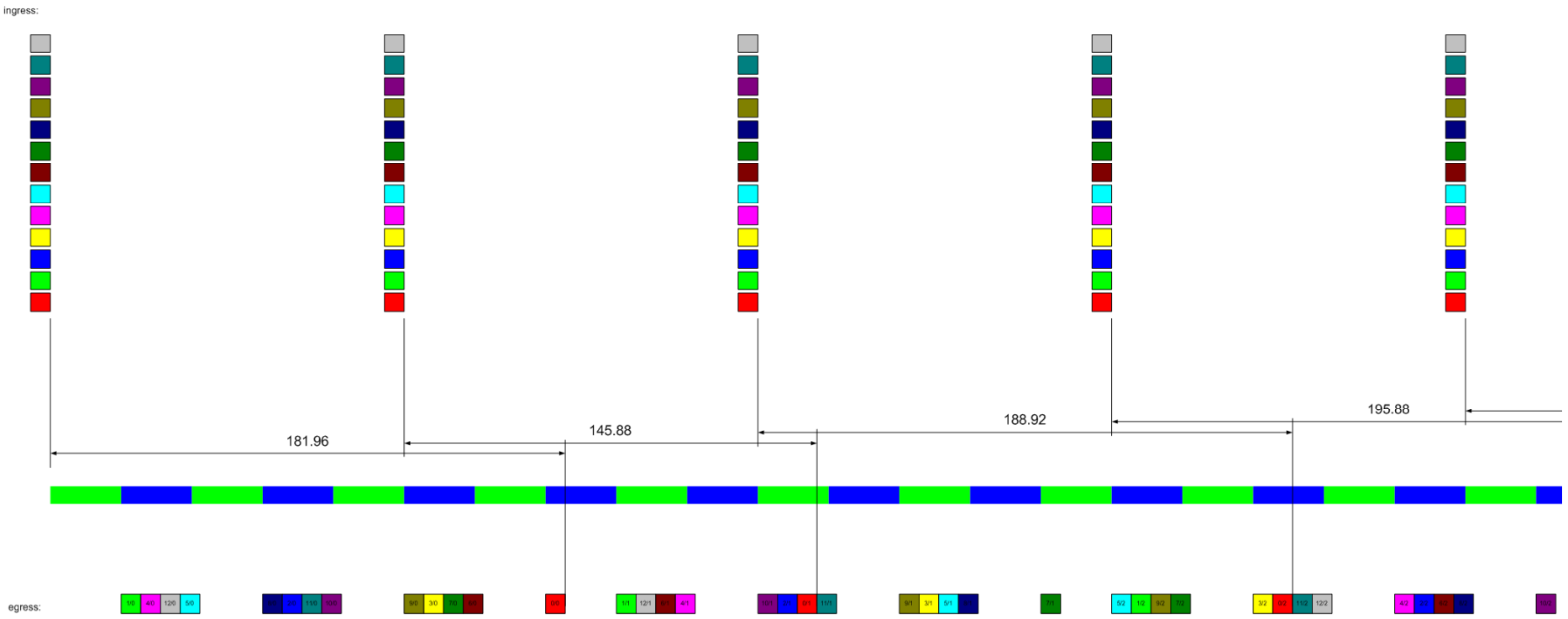
Peristaltic Shaper (Example 2)



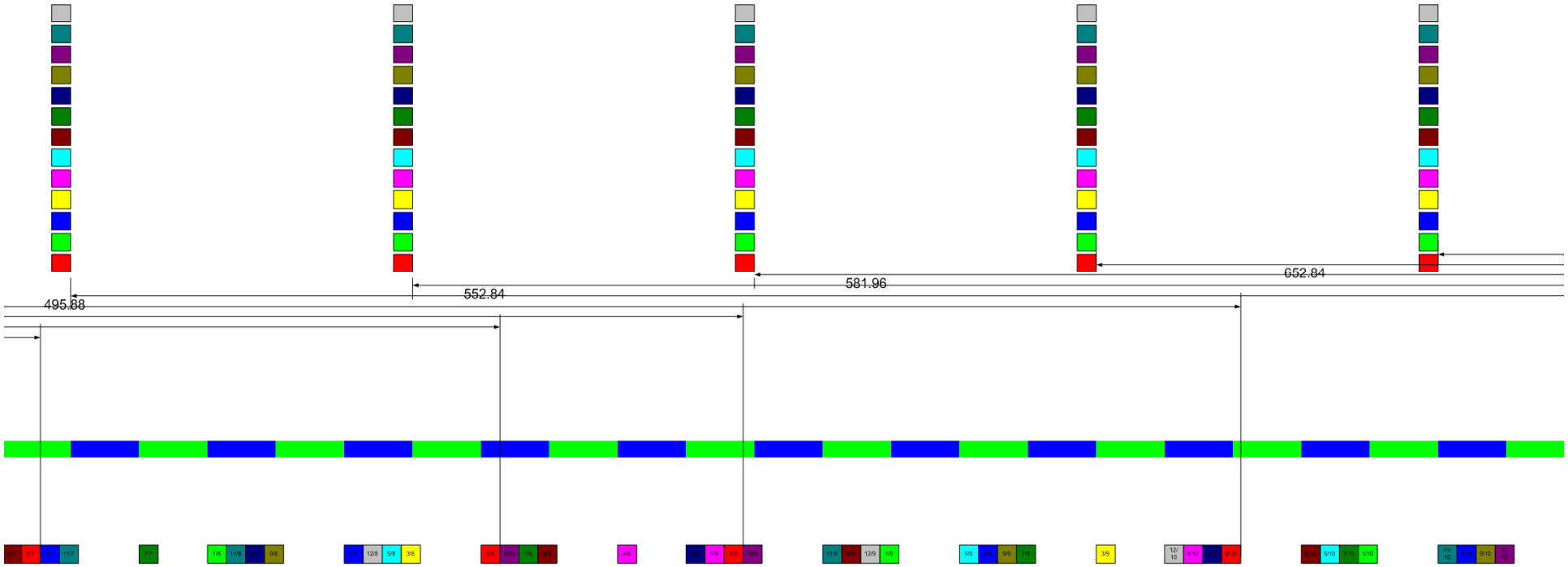
Peristaltic Shaper (Example 3)



Peristaltic Shaper (Example 4/1)



Peristaltic Shaper (Example 4/2)



Peristaltic Shaper Conclusion

- This concept might work
- The basic idea behind this shaper seems to be, that everything [!] that is received in an interval is transmitted in the next interval
- This requires that:
- The (reserved) transmission period is equal to the shaper interval
- The shaper of the bridges are synchronized (similar to TAS – perhaps a closer look on delays is necessary)
- This shaper is not backward compatible to the CBS!

Thank You