# Input Synchronization for Cyclic Queueing and Forwarding

Norman Finn
nfinn@nfinnconsulting.com
Huawei Technologies Co. Ltd
18 September, 2021

# Introduction

The Shape of Things To Come:

1. The Problem: Synchronizing a CQF input port to a transmitter.
2. The apparent solution found in IEEE 802.1Q: Synchronized time, link delay measurement, and rotating clock-driven input gates.
3. A better way to think about the problem and its solution.
4. A proposed protocol for solving the problem that has advantages over the current apparent solution.

# The Problem

# Basic  Cyclic Queuing and Forwarding

As  described in Annex T of IEEE Std 802.1Q-2018, each output port has two CQF queues.

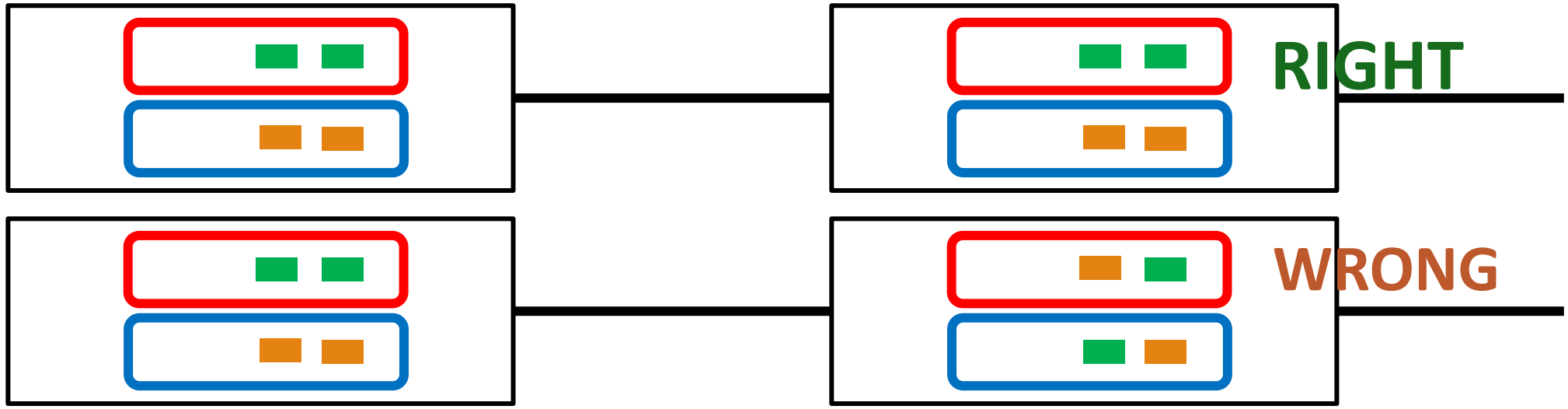
At any given moment, one queue is taking input frames and storing them, but not offering any frames for transmission.  At that same moment, no input frames are being sent to the the other queue; it is, however, offering frames to the MAC layer for transmission.

Every $T_C$ seconds, the roles of the two queues are interchanged.

Admission control guarantees that no queue will receive more data than it can hold, and that all of its data can be transmitted within time $T_C$.

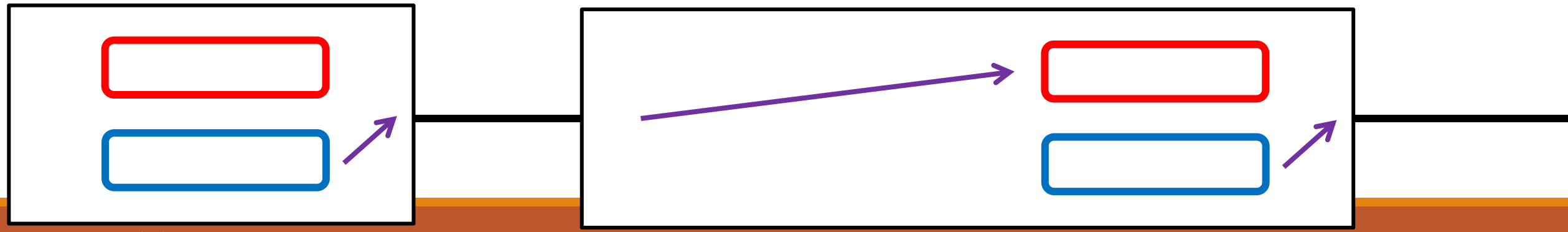
# The core assumption/requirement of Cyclic Queuing and Forwarding

If two frames are assigned to the same cycle in one transmission port, and they both are to be transmitted from the same port at the next hop, they must be transmitted in the same cycle at that next hop; they cannot be separated into different cycles.

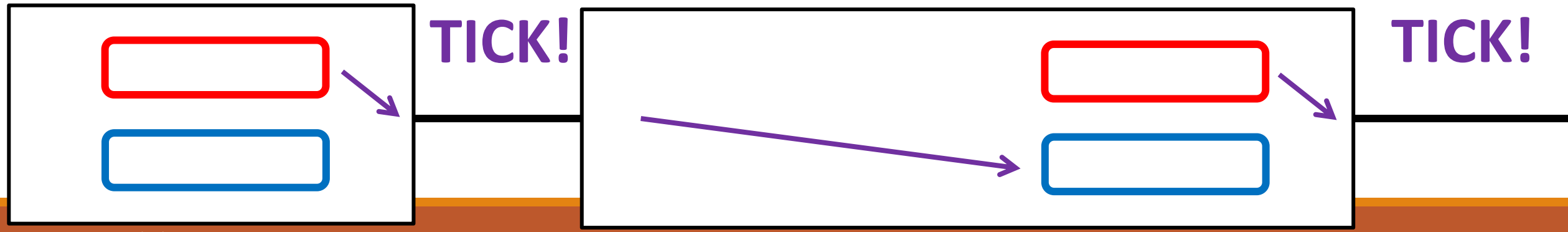


# Basic Cyclic Queuing and Forwarding: Two buffers

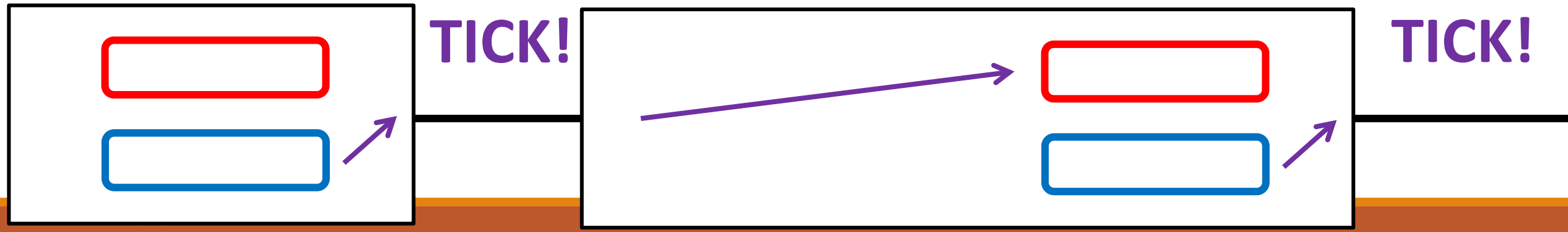
Two buffers per port. Input and output buffers swap at the same moment, once every cycle, period $T_C$. All bridges are synchronized and swap buffers at the same moment. A small guard band allows for link delay and forwarding delay. Cycle time $T_C$ > transit time + forwarding time + clock inaccuracy + max data transmit time.



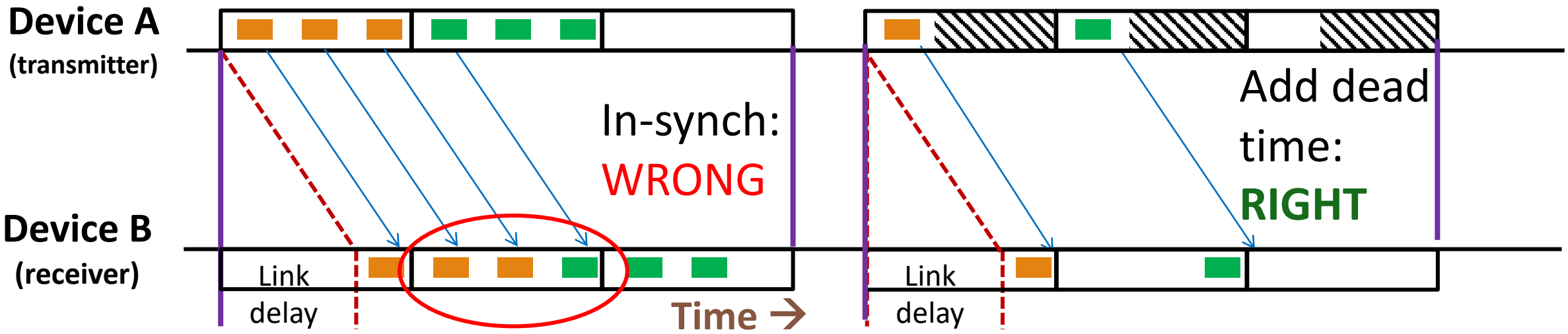
# Two buffers



# Two buffers



# Receiver synchronization with link delay

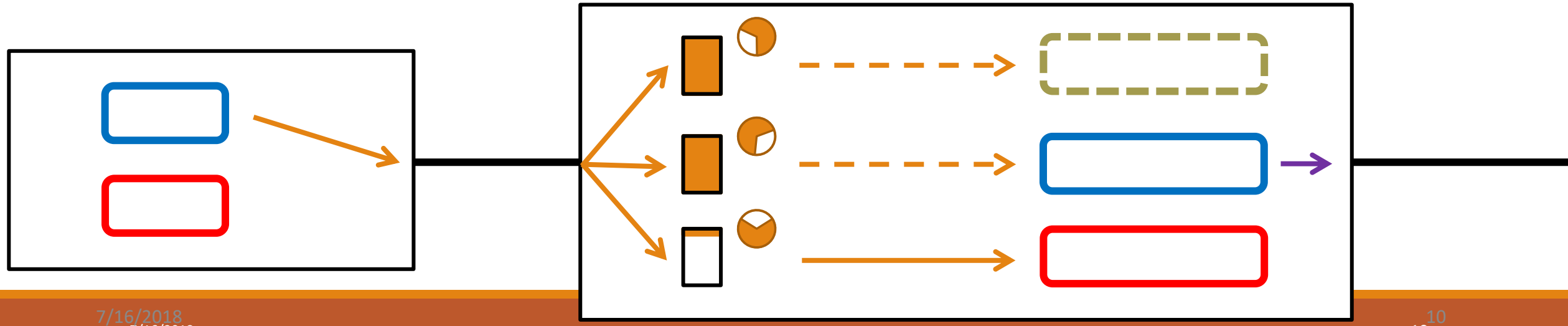



- Annex T of IEEE 802.1Q-2018 fixes long link delay by adding dead time, which reduces the bandwidth available for CQF streams and sets a minimum cycle time.

# Not-quite-Synchronous CQF: Two buffers sending to three buffers

Input buffer swap is out-of-phase with output buffer swap to allow for arbitrary link delay or out-of-phase output buffers.

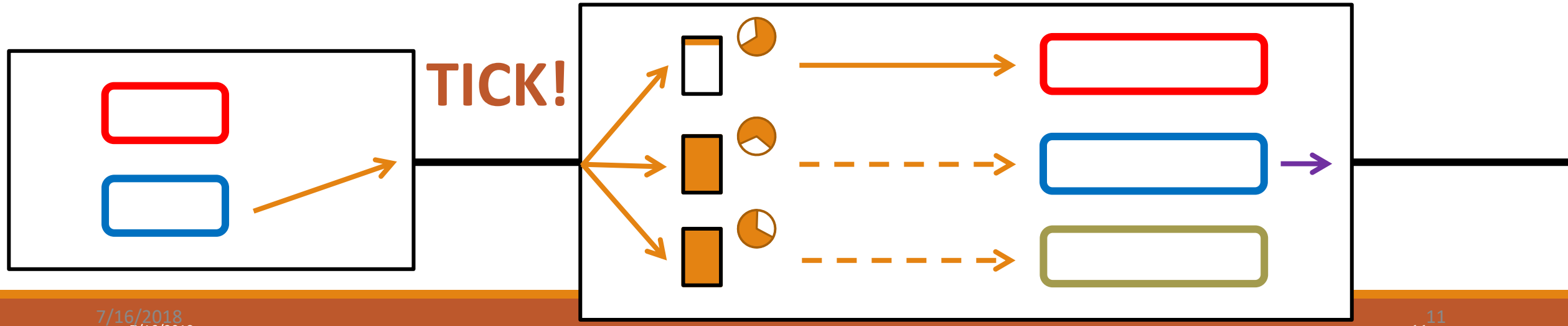
Each buffer cycles through four states: filling, full, draining, empty.




7/16/2018
10

# Two buffers sending to three buffers

- Three timed input gates, one for each buffer, turned on/off by the clock schedule.
- All frames are offered to all three gates; only one is open at any given moment.

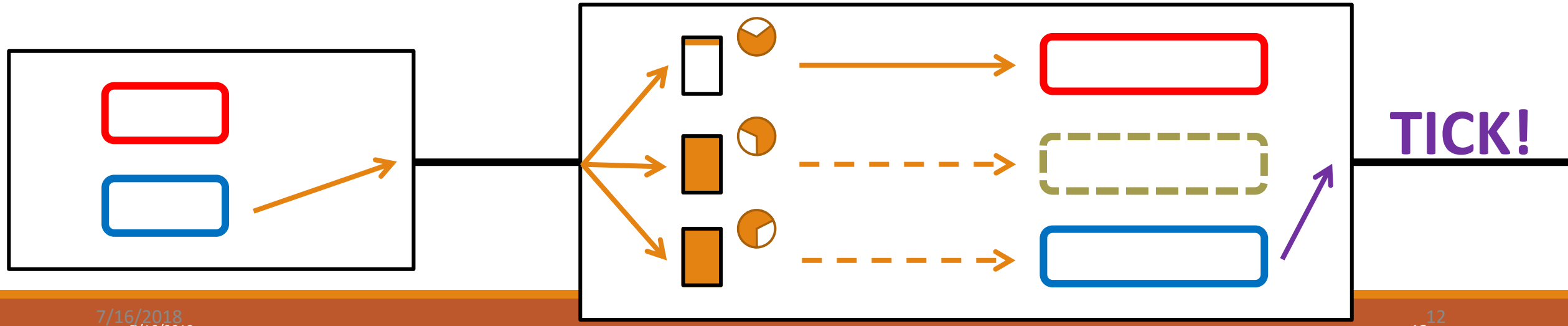




7/16/2018
11

# Two buffers sending to three buffers

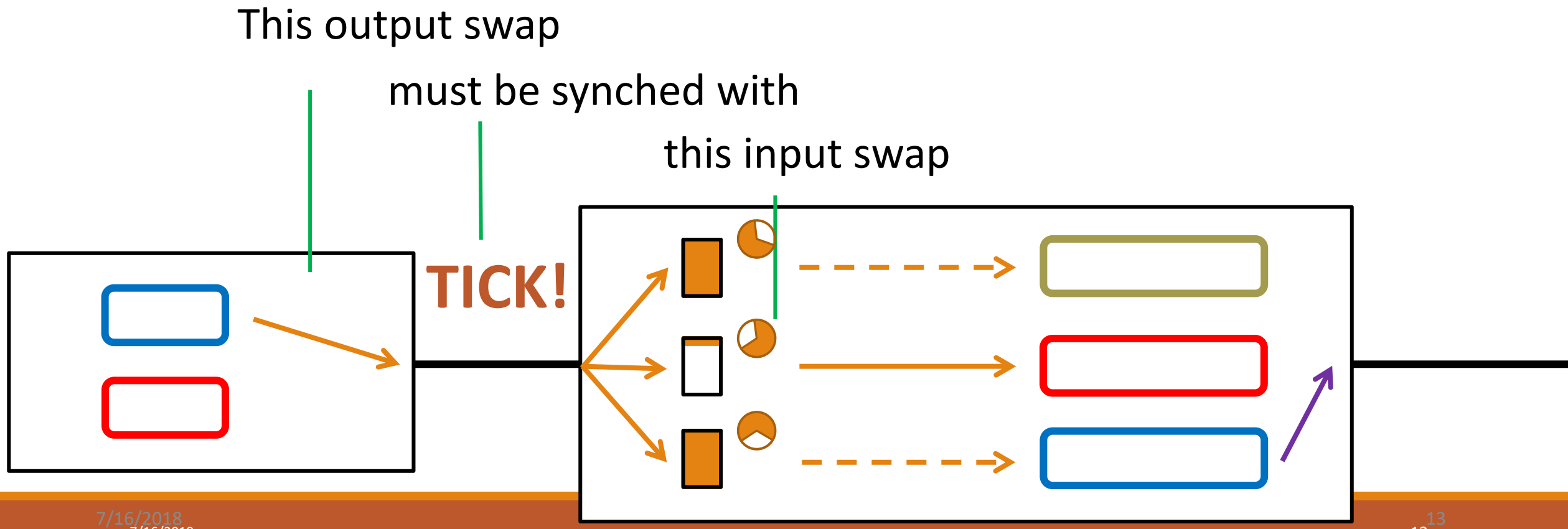
- Three timed input gates, one for each buffer, turned on/off by the clock schedule.
- All frames are offered to all three gates; only one is open at any given moment.





7/16/2018
12

# Two buffers sending to three buffers

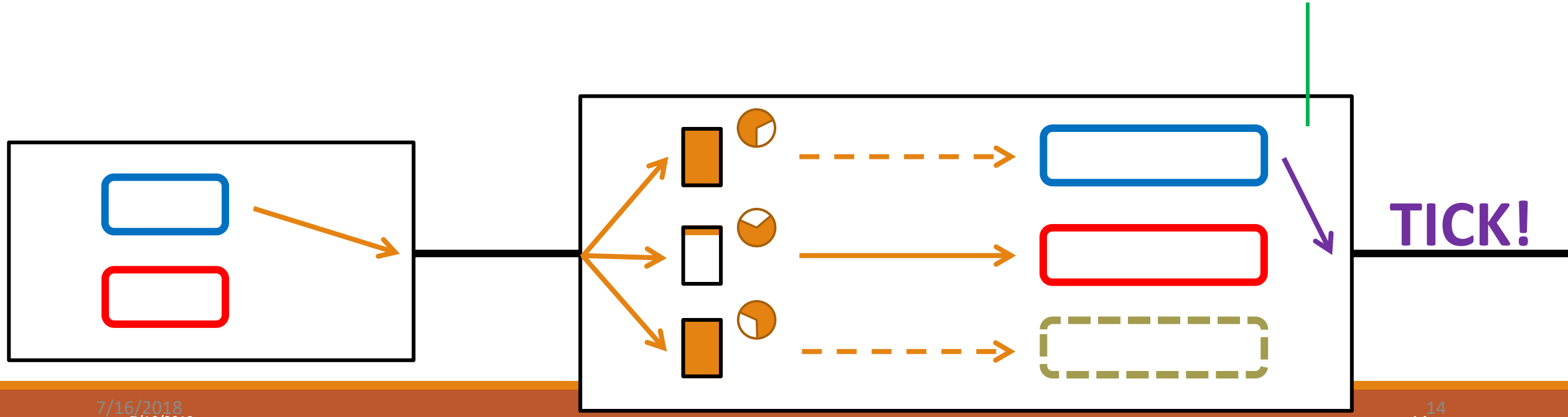




7/16/2018
13

# Two buffers sending to three buffers

This output swap can be out-of-phase with input swap, must be have same frequency

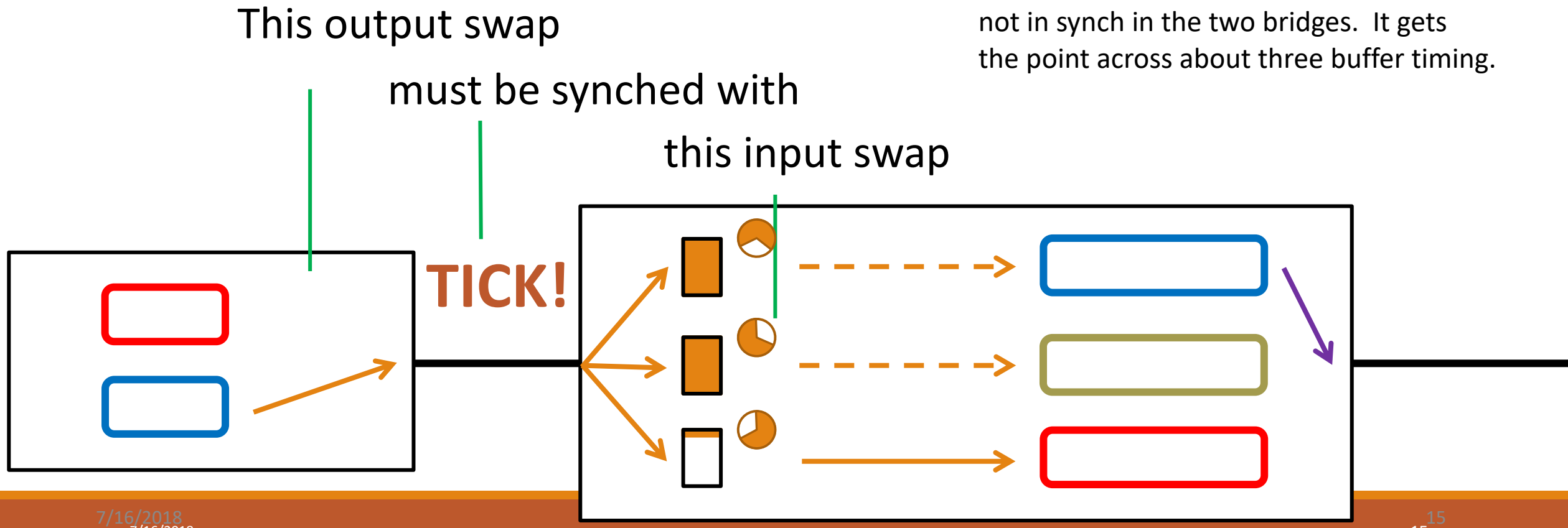




7/16/2018
14

# Two buffers sending to three buffers

(Link delay is not shown in this sequence. Instead, we show the output buffer cycles not in synch in the two bridges. It gets the point across about three buffer timing.

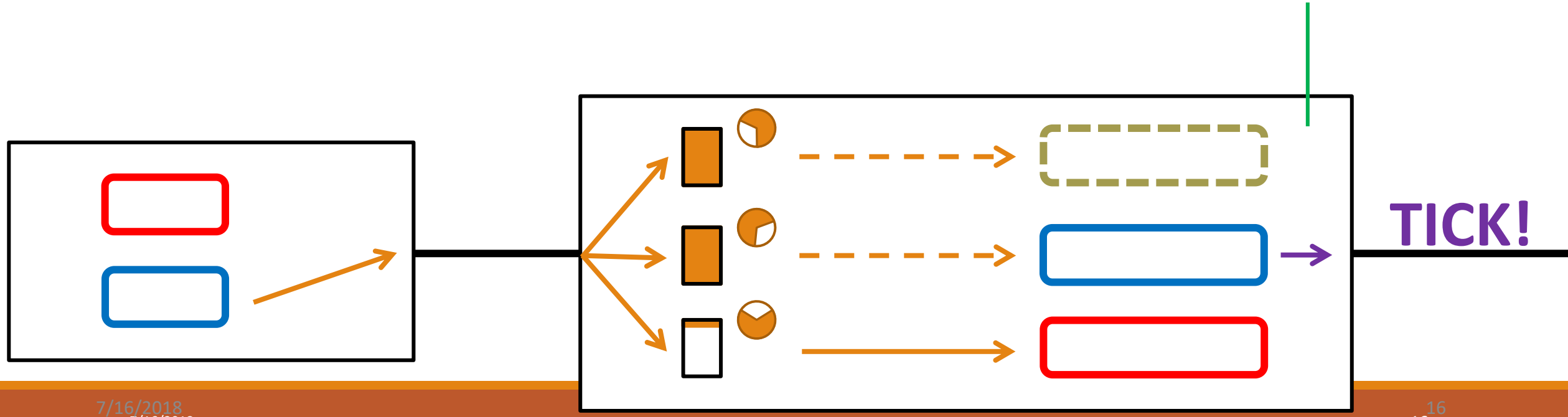




7/16/2018
15

# Two buffers sending to three buffers

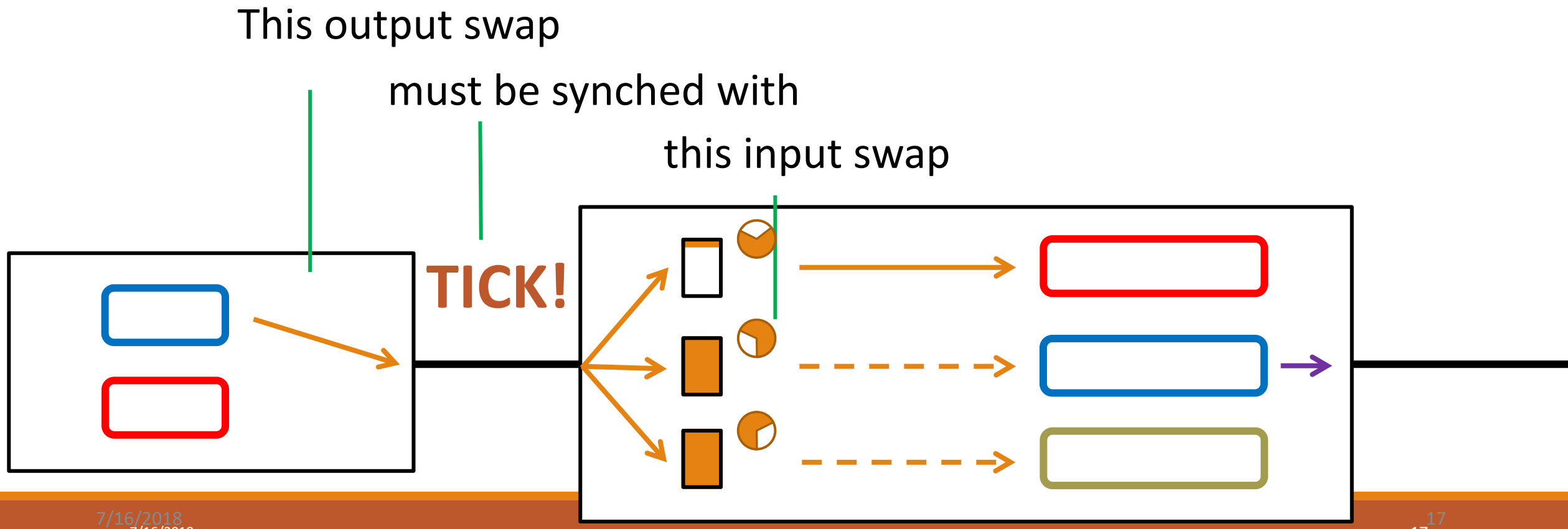
This output swap can be out-of-phase with input swap, must be have same frequency





7/16/2018
16

# Two buffers sending to three buffers

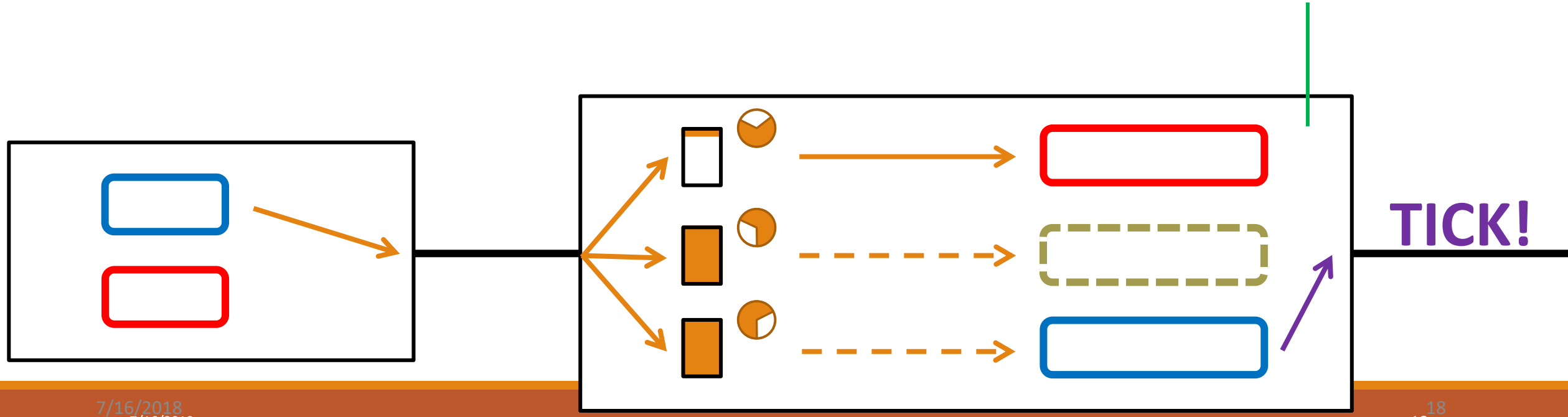




7/16/2018
17

# Two buffers sending to three buffers

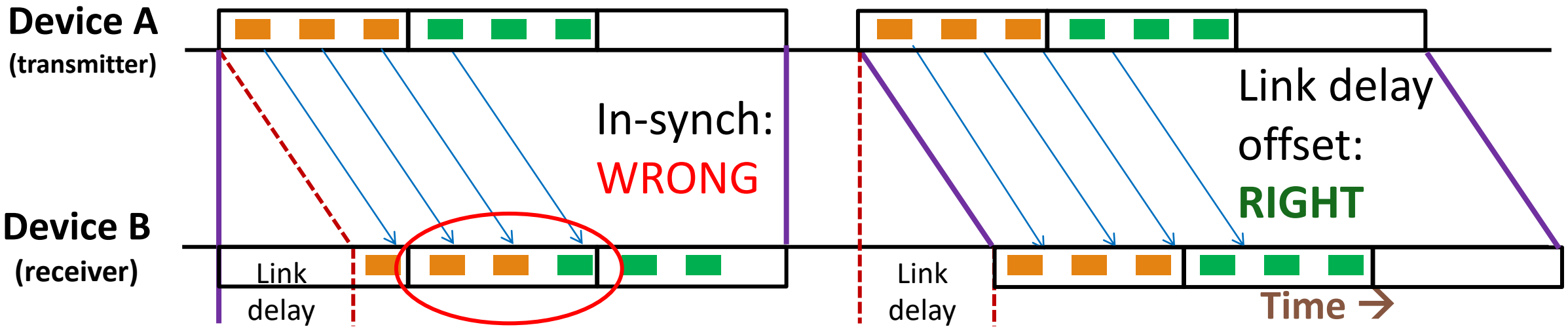
This output swap can be out-of-phase with input swap, must be have same frequency





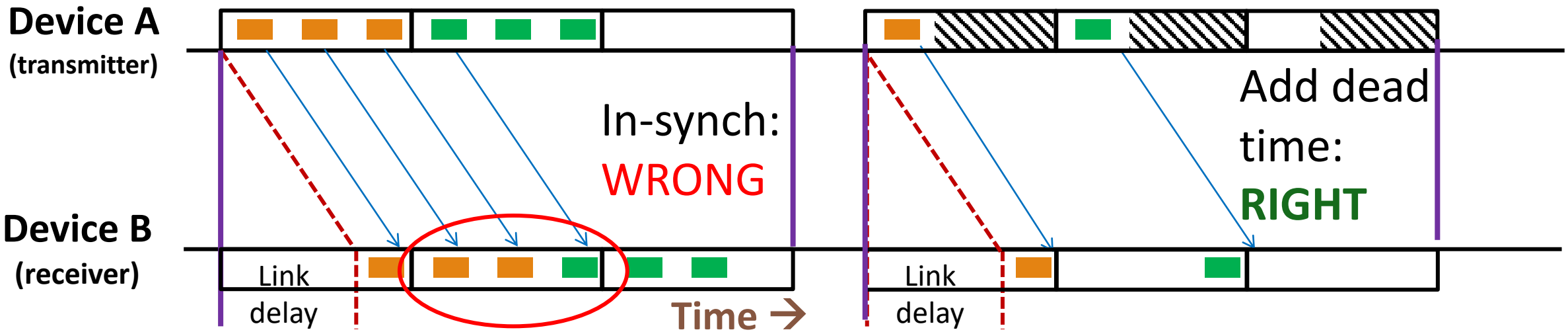
7/16/2018
18

# Receiver synchronization: Three buffers




- Offsetting the input clock by the link delay means that link delay does not add to the required dead time. More bandwidth is available for CQF.
- This looks, to the receiver, just like the animation just shown.

# Receiver synchronization: Two buffers




- Annex T of IEEE 802.1Q-2018 fixes long link delay by adding dead time, which reduces the bandwidth available for CQF streams and sets a minimum cycle time.

# Is link delay important?

- In a car? **No.** 1 meter @ 100 Mb/s, = 0.3 bits.
- To a service provider? **Yes.** 100 km @ 10 Gb/s = 3M bits = 3000 frames = a very, very large CQF cycle.

# The problem

How do we sync the input port to the output port, offset by link delay?

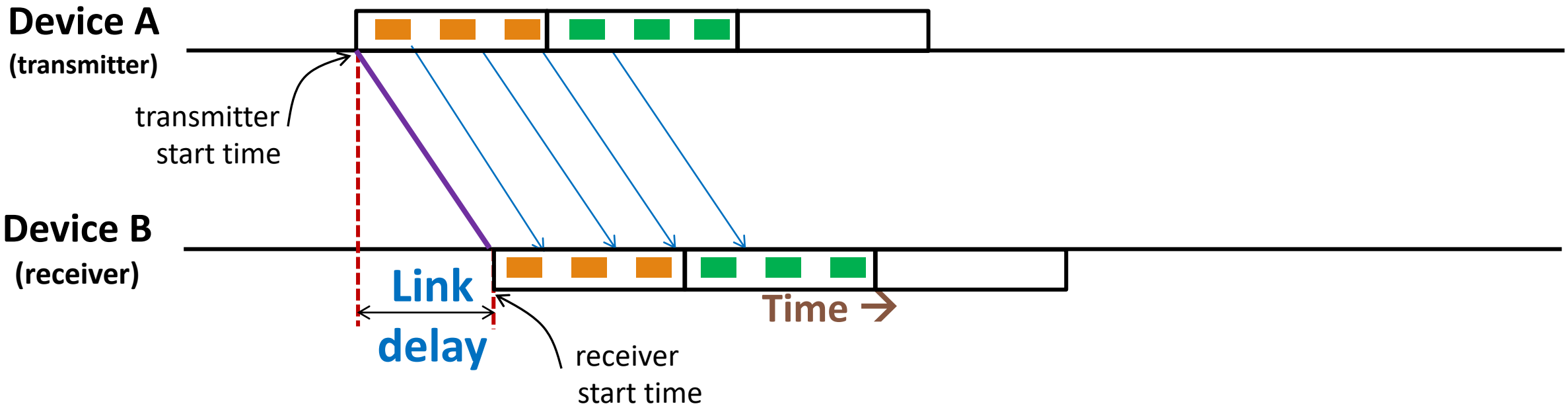
# The apparent solution

# How schedules are defined

An input or output schedule is defined by:

- A schedule of on/off events, with a total duration (in nanoseconds) which is less than or equal to the cycle time at which the schedule repeats.
- A cycle time, expressed as a rational number of seconds/cycle.
- A cycle start time, which is a moment in the past (or future) at which time the schedule did (or will) begin.

# The problem:
# Adjust receiver cycle start time by link delay




- We want offset the cycle start time of the receiving input gates from the transmitting output gates by an amount equal to the link delay.

# The apparent solution

- Use some form of the Precision Time Protocol (PTP), e.g. IEEE Std 802.1AS, to synchronize time between the transmitter and the receiver.
- Use half the two-way link delay, as measured by PTP, to offset the input cycle start time from the output cycle start time, which is synchronized among all bridges.

# Another way to think about the problem

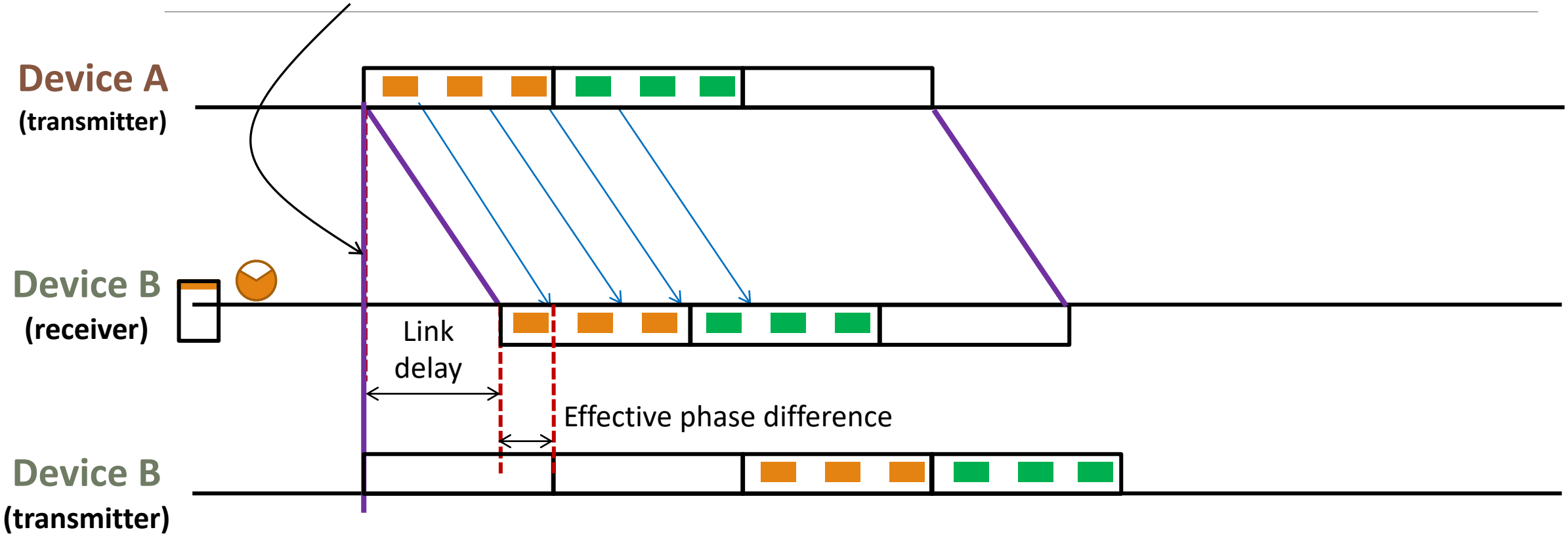
# What's wrong with the apparent solution?

- For some users, nothing is wrong with that scheme.
- But some use SyncE, instead of PTP, to get all bridges running at the same frequency.
- There is some asymmetry in the link delay:
  - Asymmetry cannot be measured easily; it is normally configured.
  - Asymmetry has to be accounted for by adding dead time to the cycle.
  - For long links, this can be a significant limit on the minimum cycle time.

# Is link asymmetry important?

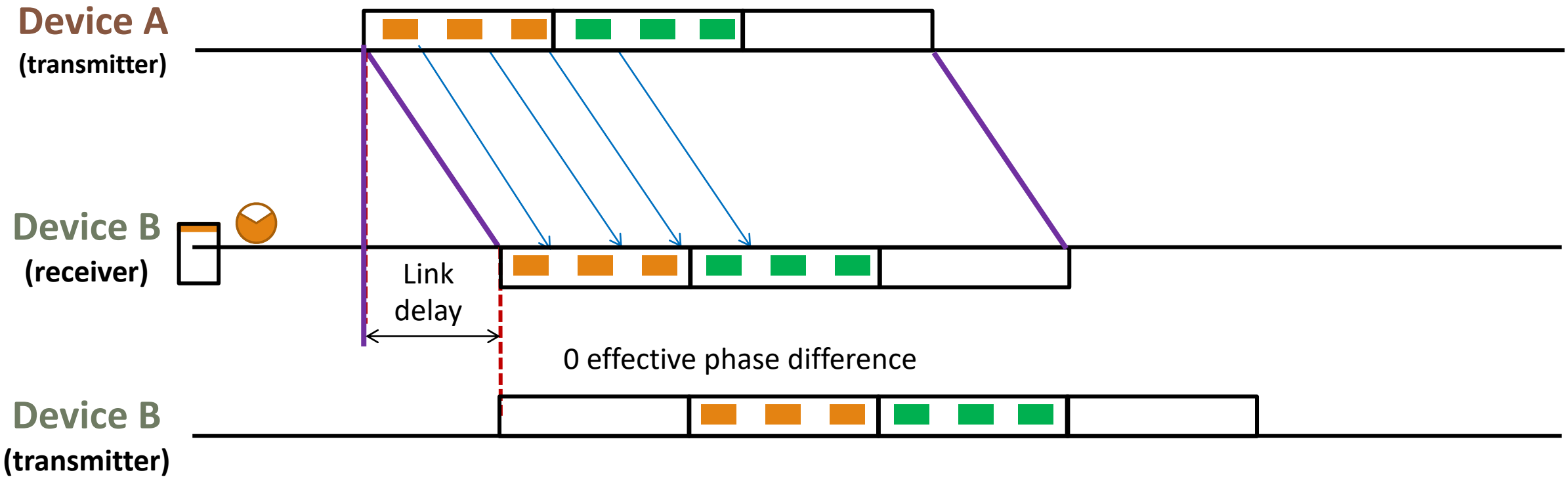
- In a car? No. 1 meter @ 100 Mb/s = 0.3 bits.
- To a service provider? Yes. 100 km @ 10 Gb/s = 3M bits = 5000 small frames. 1% asymmetry is not unusual, which is 50 frames dead time per cycle.
- This matters for a service provider offering a high-speed service for a few critical flows.

# Transmit cycles synchronized



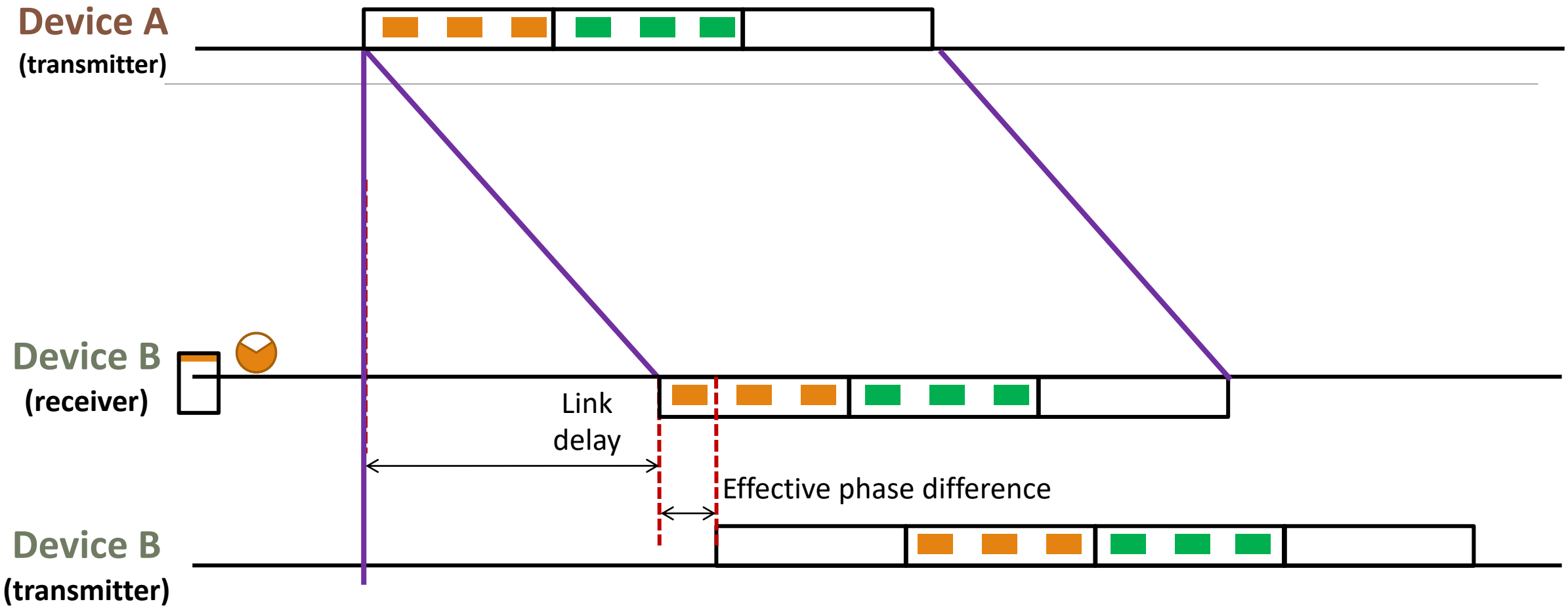

- The “effective phase difference” determines the CQF buffering delay in device B. If it is small, two buffers and a little dead time can be used.

# Transmit cycles not synchronized




- The “effective phase difference” determines the CQF buffering delay in device B. If it is small, two buffers and a little dead time can be used.

# Long link delay



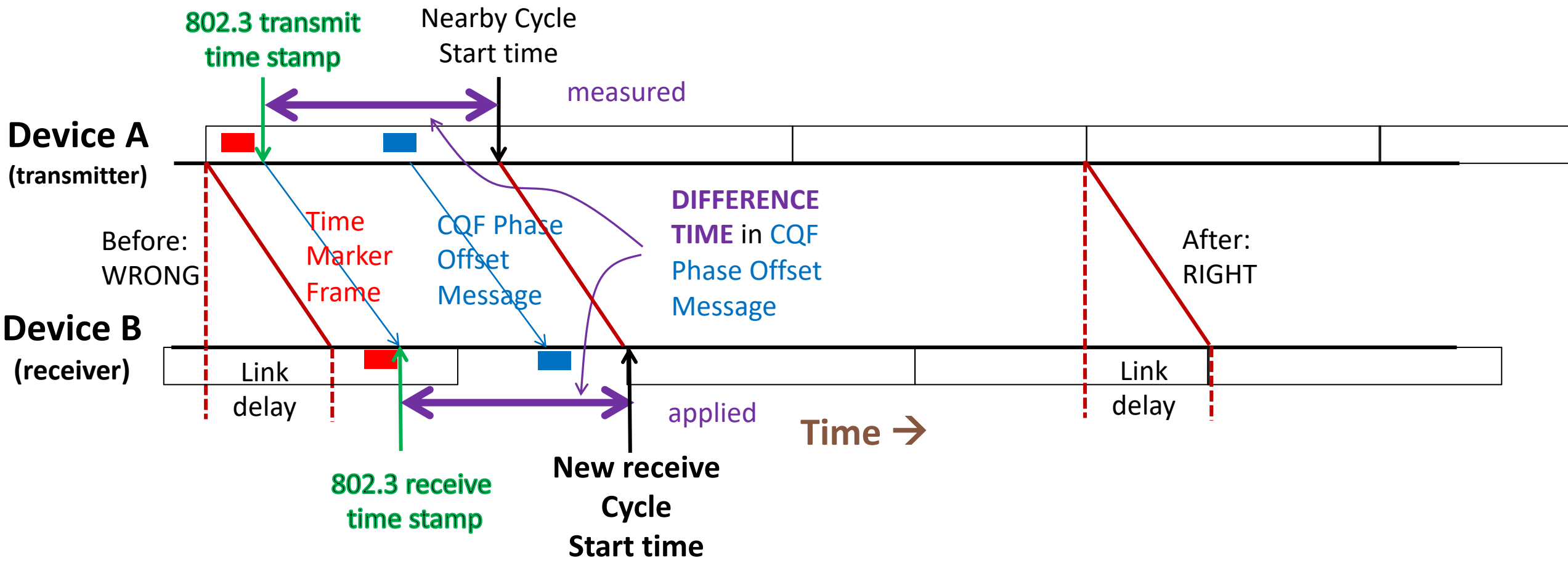

- Link delay can be multiple cycles long, but still yield a small effective phase difference.

# What do we (Device B's receiver input gates) **really** care about?

- We **don't** care about the actual link delay in nanoseconds.
- We **do** care about setting the receive cycle start time so that we assign received frames to cycles using exactly the boundaries that the transmitter used.
- We **don't** care whether our output cycles are synchronized with the previous hop's output cycles.
- We **do** care about the "effective phase difference", because that determines how many buffers we need (2 or 3).
- And, critically, we know that we are frequency-locked with the transmitter.

# A new proposed solution

# The proposed solution



# The proposed solution: transmitter's end

1. Device A transmits a Timing Marker Frame. This can be any frame, but it must carry an identification value (TMFID) that changes with each transmission.
2. After transmitting the Timing Marker Frame, Device A recovers the time at which the first bit of the frame was transmitted from the hardware. IEEE Std 802.3 Clause 90 specifies a method for accomplishing this.
3. Device A then transmits a CQF Phase Offset Message that contains a) the TMFID of a Timing Marker Frame, and b) the difference, in local time, between the start of a recent transmission cycle and the Timing Marker Frame transmission time recovered in step 2, above. (Specifically, [time of start of cycle] – [time of start of transmission].) Typically, this time difference would be expressed in increments of nanoseconds or finer. The cycle chosen may be such that the time difference is positive or negative.

# The proposed solution: receiver's end

1. Device B receives a Timing Marker Frame. It records the time of arrival of the first bit of the frame (again, IEEE Std 802.3 clause 90), and the TMFID of the frame.
2. Device B receives a CQF Phase Offset Message with a TMFID matching a recently-recorded Timing Marker Frame.
3. The time of reception of the Timing Marker Frame, plus the (signed) time difference carried in the CQF Phase Offset Message, is the local time at which the receive cycle that corresponds to the transmission cycle selected by the transmitter for reporting in the CQF Phase Offset Message, should have started (or should start, if in the future). This establishes the cycle start time for the receiver's input gates in local time.

# NOTE:

- **The actual time of flight of the Timing Marker Frame is not determined, and is not relevant**. All that matters is aligning the relative phases of the transmitter's cycle and the receiver's cycle. Thus, **link asymmetry is also irrelevant**.
  - (The actual link delay is important when figuring out the total end-to-end flight time of a frame, but that is not *this* problem.)
- The in/out difference time, and thus the internal buffering delay for Device B, can be determined trivially.
- All measurements are in local time; no synchronization is necessary beyond frequency locking.

# Notes

- The receiver knows the phase from the first Timing Marker / Phase Offset Message pair. If the transmitter sends further periodic Timing Marker Frames and Phase Offset Messages, the receiver can track the accuracy of its phase determination. This allows it to take appropriate action, such as:
  - Raising an alarm or demoting multi-CQF traffic to best-effort priority, if the phase drifts excessively.
  - Adjusting the phase, if it drifts slowly, e.g. due of diurnal temperature changes in a long optical fiber, which is made possible by using more than three buffers.
- The Timing Marker Frame can be a new protocol (a new EtherType), or an existing suitable frame, such as one used by the Precision Time Protocol (PTP, IEEE Std 1588, IEEE Std 802.1AS, or others), or by Connectivity Fault Management (CFM, IEEE Std 802.1Q clauses 18-22).
- The Phase Offset Message can be a new protocol (a new EtherType), or the information can be added as an additional information element in an existing timing protocol such as PTP or CFM.

# Conclusion

# Improvements over current apparent solution for service providers

- By using three buffers instead of two, CQF can eliminate the dead time imposed link delay.  This is important to service providers.
- By using a protocol based on IEEE Std 802.3 Clause 90 timestamps, the receiver's timed input gates can placed in phase with the transmitter's timed output gates.
- This protocol removes dead time caused by link delay asymmetry.
- This protocol supports slow changes in link delay.
- This protocol supports CQF in a SyncE environment without PTP.

# Discussion

# Thank you