Low-Power System Design

227-0781-00L

Fall Semester 2019

Jan Beutel

ETH zürich

Plan for Today

- Network Time Synchronization
  - Basics, Fundamental Effects
  - Algorithm Examples
  - Time-of-Flight Aware Time Sync

- Slides contain material from R. Wattenhofer and R. Lim

Low-Power System Design

NETWORK TIME SYNCHRONIZATION BASICS

Clock Synchronization in Networks?

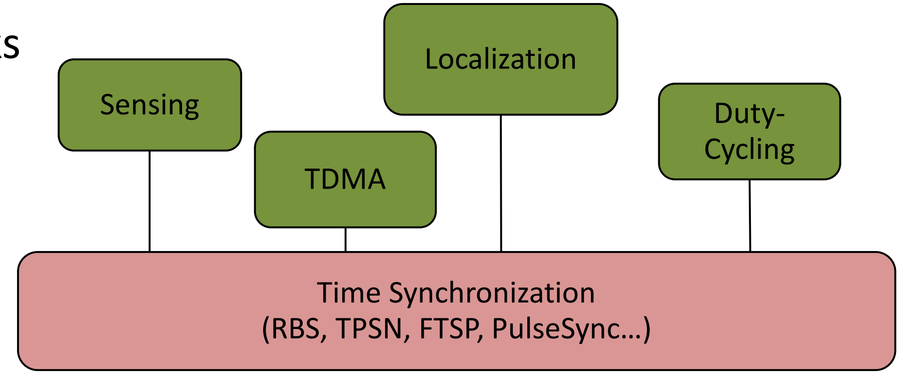
- *Time, Clocks, and the Ordering of Events in a Distributed System.* L. Lamport, Communications of the ACM, 1978.
- ***Internet Time Synchronization: The Network Time Protocol (NTP).*** **D. Mills, IEEE Transactions on Communications, 1991**
- *Reference Broadcast Synchronization (RBS).* J. Elson, L. Girod and D. Estrin, OSDI 2002
- *Timing-sync Protocol for Sensor Networks (TPSN).* S. Ganeriwal, R. Kumar and M. Srivastava, SenSys 2003
- ***Flooding Time Synchronization Protocol (FTSP).*** **M. Maróti, B. Kusy, G. Simon and Á. Lédeczi, SenSys 2004**
- and many more ...

Time in Sensor Networks

- Synchronizing time is essential for many applications
  - Coordination of wake-up and sleeping times (energy efficiency)
  - TDMA schedules
  - Ordering of collected sensor data/events
  - Co-operation of multiple sensor nodes
  - Estimation of position information (e.g. shooter detection)

- Goals of clock synchronization
  - Compensate *offset** between clocks
  - Compensate *drift** between clocks

  *terms are explained on following slides



Properties of Clock Synchronization Algorithms

- External versus internal synchronization
  - External sync: Nodes synchronize with an external clock source (UTC)
  - Internal sync: Nodes synchronize to a common time
    - to a leader, to an averaged time, or to anything else

- One-shot versus continuous synchronization
  - Periodic synchronization required to compensate clock drift

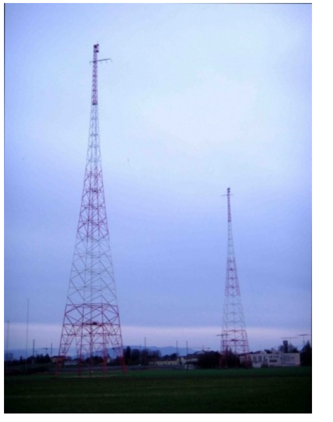
- A-priori versus a-posteriori
  - A-posteriori clock synchronization triggered by an event

- Global versus local synchronization

- Accuracy versus convergence time, Byzantine nodes, …

Global Clock Sources

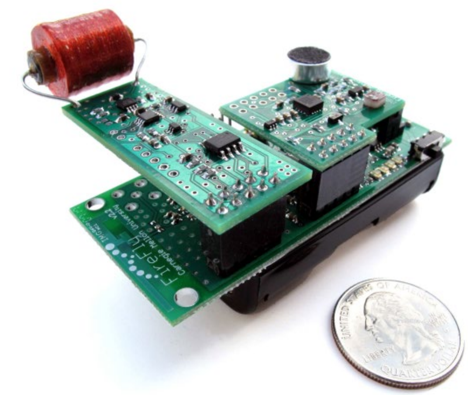
- Radio Clock Signal
  - Clock signal from a reference source (atomic clock) is transmitted over a long wave radio signal
  - DCF77 station near Frankfurt, Germany transmits at 77.5 kHz with a transmission range of up to 2000 km
  - Accuracy limited by the distance to the sender, Frankfurt-Zurich is about 1ms.
  - Special antenna/receiver hardware required



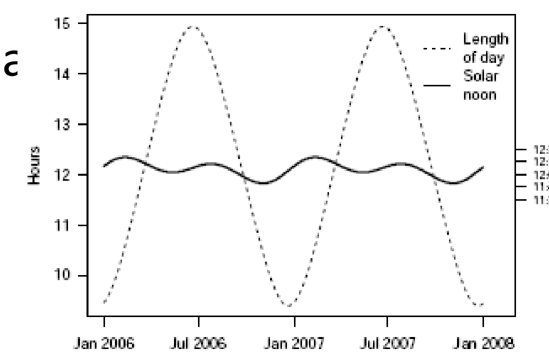
- Global Positioning System (GPS)
  - Satellites continuously transmit own position and time code
  - Line of sight between satellite and receiver required
  - Special antenna/receiver hardware required

Global Clock Sources (2)

- AC power lines
  - Use the magnetic field radiating from electric AC power lines
  - AC power line oscillations are extremely stable (10^{-8} ppm)
  - Power efficient, consumes only 58 µW
  - Single communication round required to correct phase offset after initialization



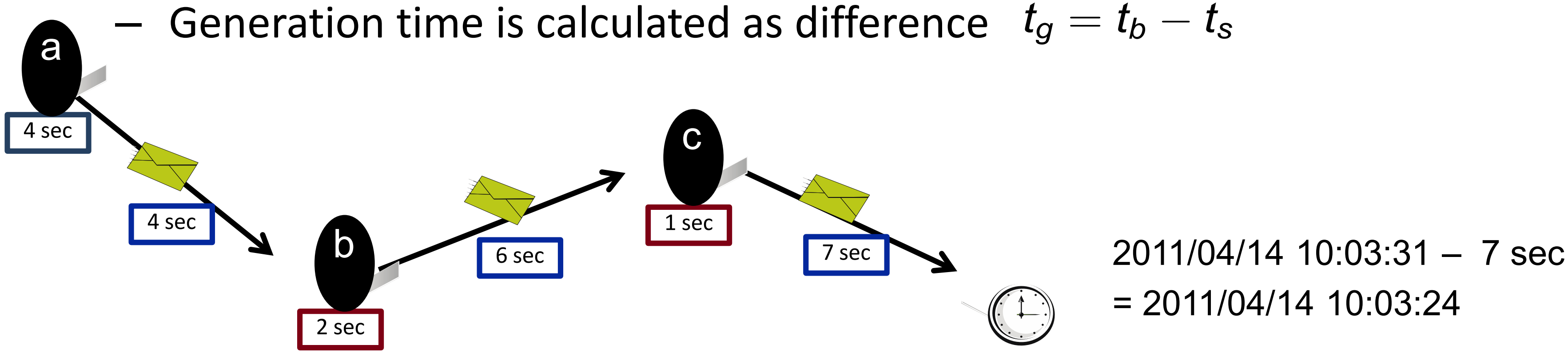
- Sensor Signals (Sunlight)
  - Using a light sensor to measure the length of a da
  - Offline algorithm for reconstructing global timestamps by correlating annual solar patterns (no communication required)



Global vs. Local Time Sync

- In cases where no network-wide time synchronization is available
  - Global time sync not available for network protocol control
  - Implications on data usage

- Solution: Elapsed time on arrival
  - Sensor nodes measure/accumulate packet sojourn time
  - Base station annotates packets with UTC timestamps
  - Generation time is calculated as difference $\tilde{t}_g = t_b - \tilde{t}_s$

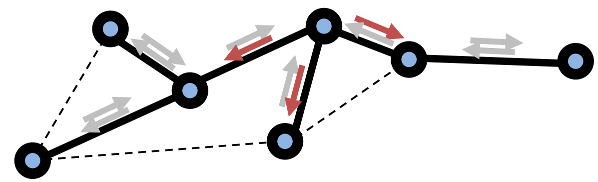


Network Time Synchronization

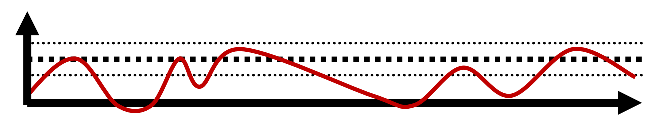
- Goal

  Send time information (beacons) across network to synchronize clocks

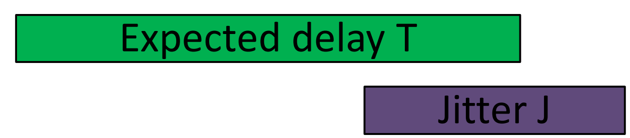
- Problems
  - –Network ensemble **interactions**



  - – Hardware clocks exhibit **drift**



  - – **Jitter** in the message delay

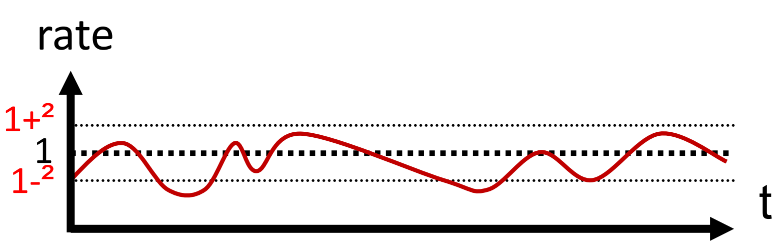


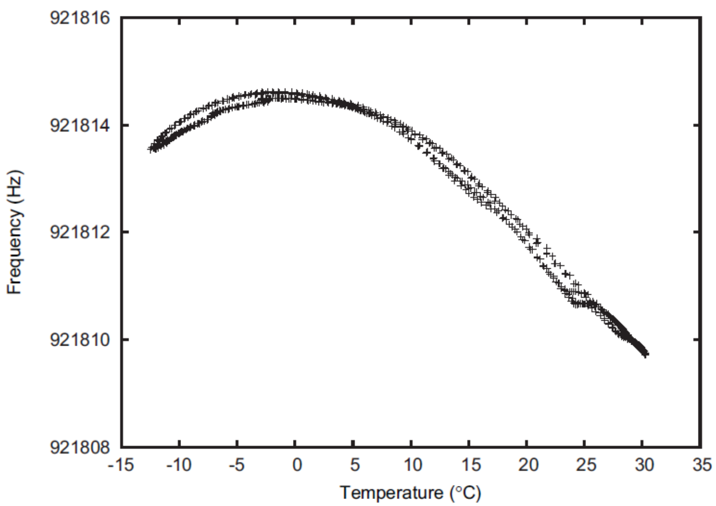
Hardware Clocks Experience Drift

- Hardware clock
  - Counter register of the microcontroller
  - Sourced by an external crystal (32kHz, 7.37 MHz)

- Clock drift
  - Random deviation from the nominal rate dependent on ambient temperature, power supply, etc. (30-100 ppm)

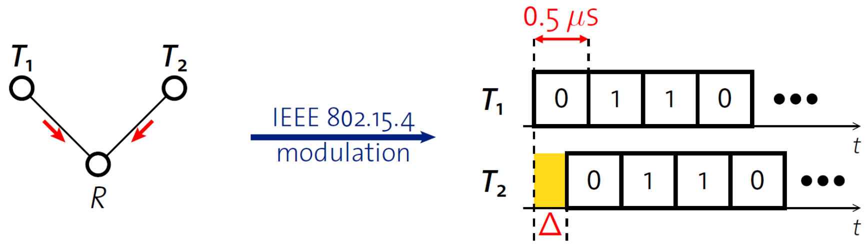
This is a drift of up to 50 µs per second or 0.18s per hour





Example Glossy and Timing

- Remember



- R receives packet with high probability if Δ ≤ 0.5 µs

- 32.768 kHz @ 10/20/50ppm -> $\frac{1}{32768\ kHz} = 30.5\ \mu sec$

- +/-20 ppm results in 32.7673 to 32.7687 kHz
  $32768\ \pm 20\ ppm$ is x 0.999980 to x 1.000020

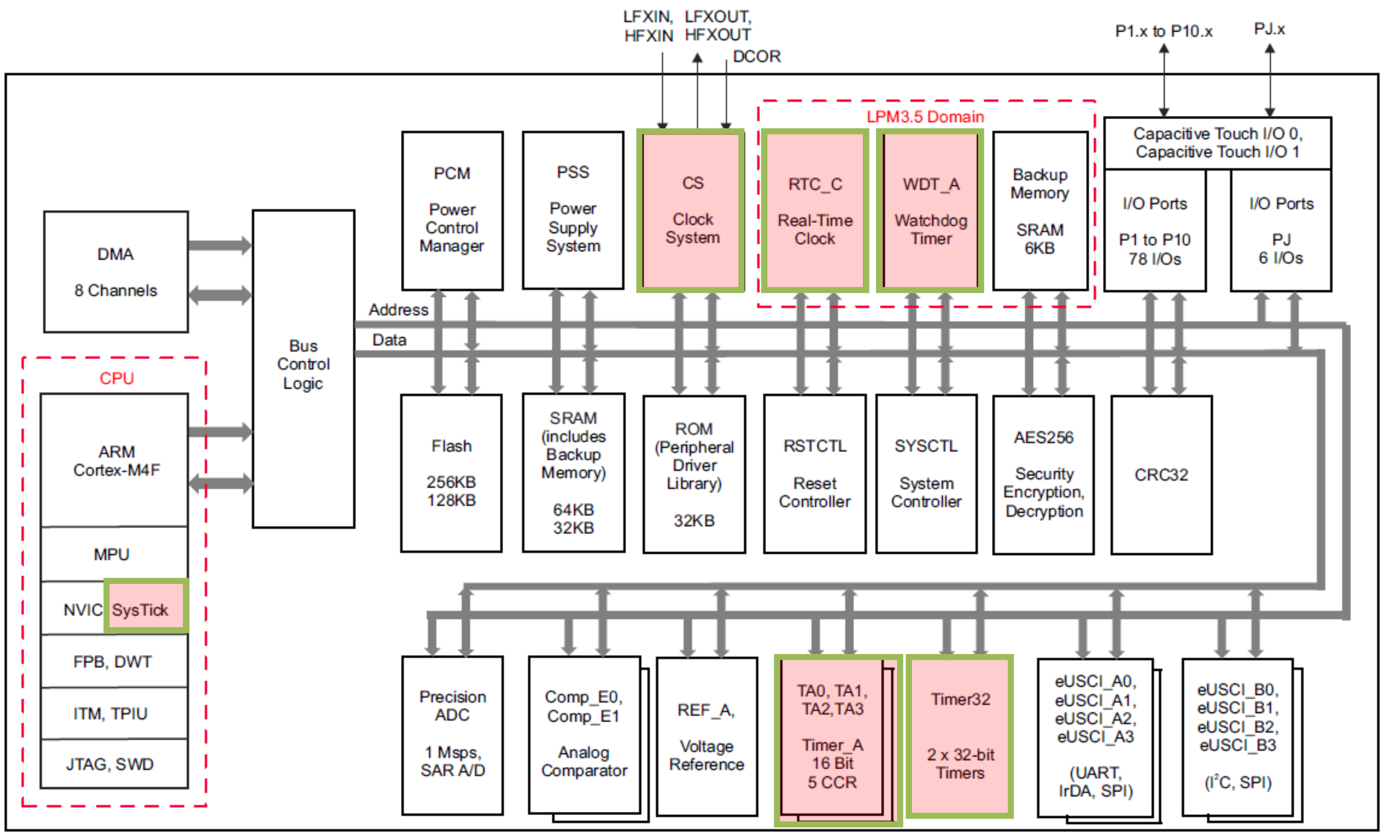
- 1 Month = 60x60*24*30 = 2.6 million seconds
  20 ppm crystal for wakeup results in error 1 min per month

Hardware Clocks

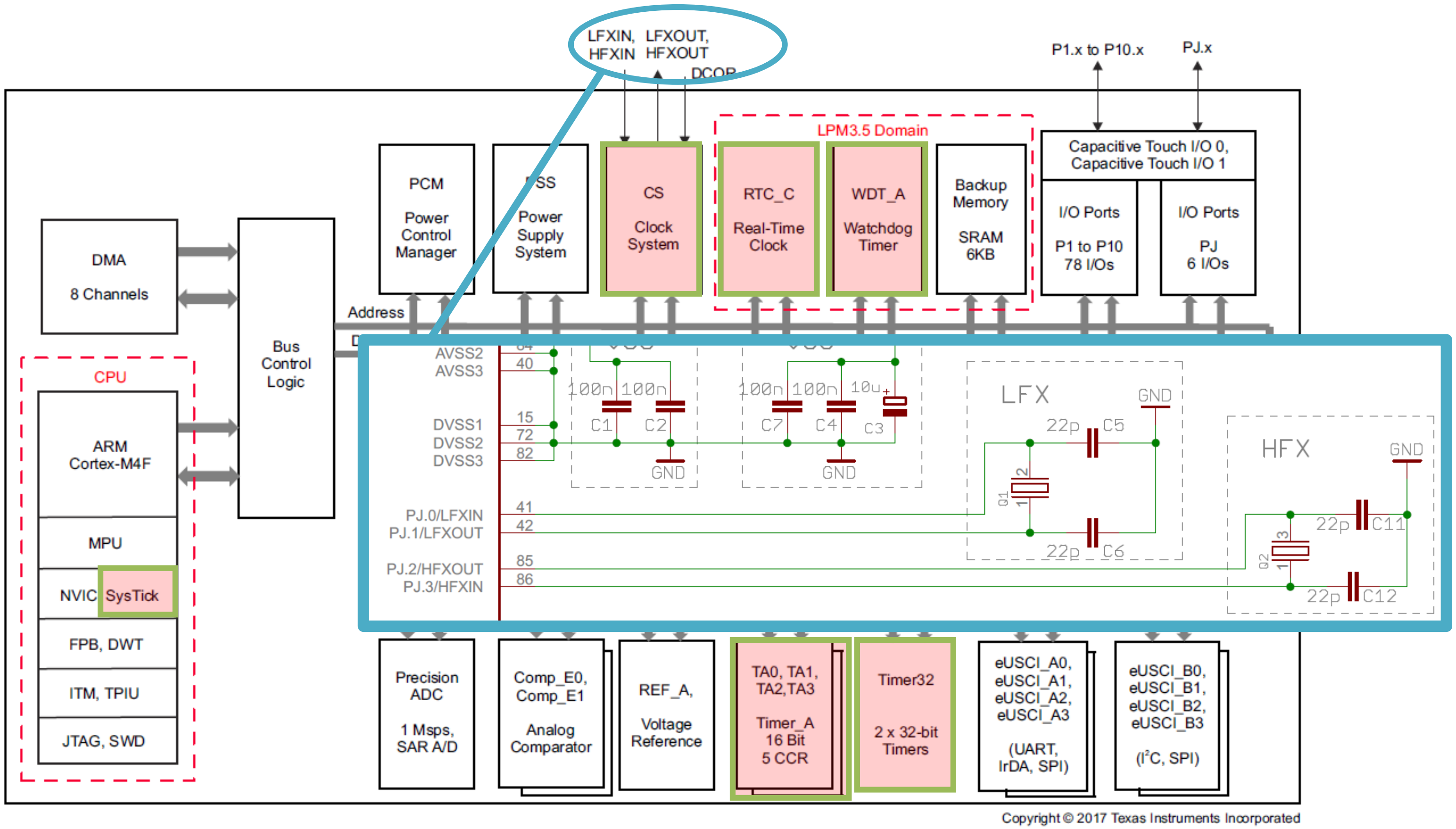
- Microcontrollers usually have different clock sources with varying
  - frequency (relates to precision)
  - energy consumption
  - stability, e.g., crystal-controlled clock vs. digitally controlled oscillator
- As an example, the MSP432 has the following clock sources:

	frequency	precision	current	comment
LFXTCLK	32 kHz	0.0001% / °C … 0.005% / °C	150 nA	external crystal
HFXTCLK	48 MHz	0.0001% / °C … 0.005% / °C	550 µA	external crystal
DCOCLK	3 MHz	0.025% / °C	N/A	internal
VLOCLK	9.4 kHz	0.1% / °C	50 nA	internal
REFOCLK	32 kHz	0.012% / °C	0.6 µA	internal
MODCLK	25 MHz	0.02% / °C	50 µA	internal
SYSOSC	5 MHz	0.03% / °C	30 µA	internal

Clocks and Timers MSP432



Clocks and Timers MSP432



ETH zürich

Best-in-class Real-time Clocks

em microelectronic
A COMPANY OF THE SWATCH GROUP

DATASHEET I EM3028

EXTREME LOW POWER RTC WITH I²C,

32-bit UNIX time counter, 43 bytes EEPROM, Battery Switchover and Trickle Charger

DESCRIPTION

The EM3028 engineered using the in-house analog low power (ALP) technology provides unmatched true ultra-low current consumption of typically 40nA while running on a standard 32'768 Hz tuning fork crystal. Thus allowing several hours of backup supply using cost effective MLCC capacitors.

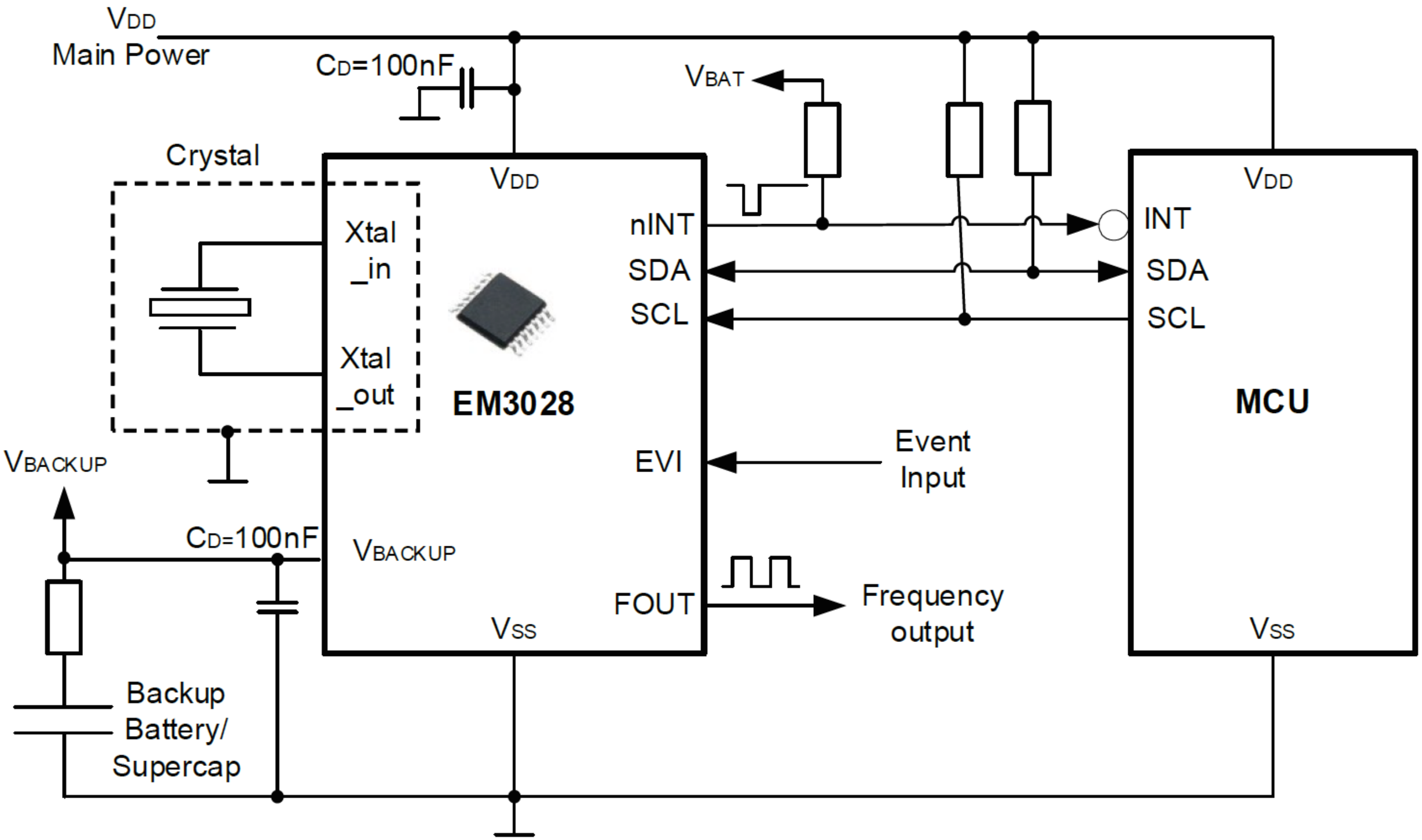
It provides full RTC function with programmable counters, alarm, selectable interrupt and clock output functions and also a 32-bit UNIX Time counter.

The internal EEPROM memory hosts all configuration settings and allows for additional 43 bytes of user memory.

FEATURES

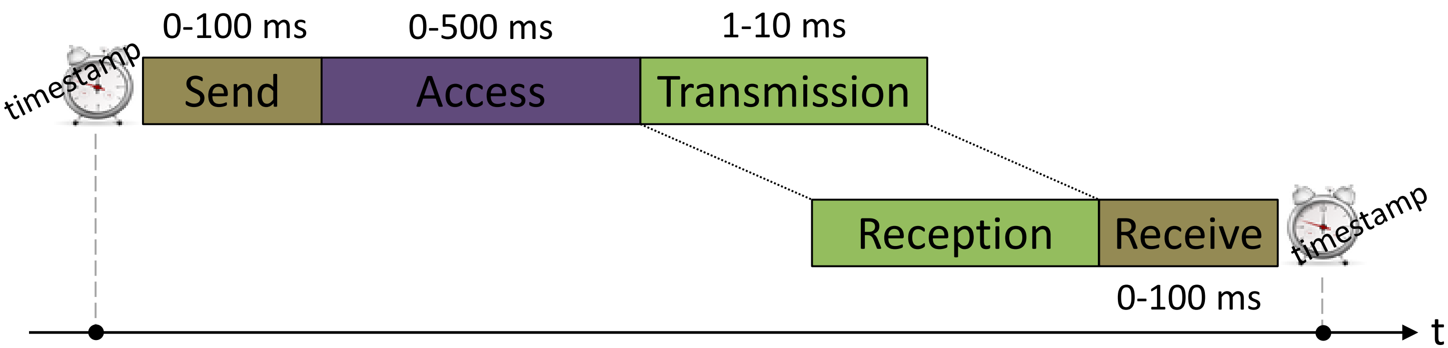
- Extreme low power consumption: **45 nA @ 3 V**.
- Wide operating voltage range: 1.2 V to 5.5 V.
- Built-in tuning Fork crystal at 32'768 Hz
- Time accuracy: possible to calibrate to **±1 ppm @ 25°C**
- Non-volatile configuration settings with user programmable offset value.
- Configuration stored in EEPROM and mirrored in RAM
- Password protection to secure configuration registers
- Backup Switch and Trickle Charger function.
- Provides year, month, date, weekday, hours, minutes and seconds.
- Automatic leap year correction; 2000 to 2099
- 32 bit UNIX time counter.
- Timer, alarm and external event functions with time stamp
- Clock output: 32.768 kHz, 8192 Hz, 1024 Hz, 64 Hz, 32Hz,1 Hz.
- 43 bytes non-volatile user memory, 2 bytes user RAM.
- I²C-bus interface: 400 kHz.
- Package: TSSOP14, 100% Pb-free, RoHS-compliant
- Also available in ultra-small SMD C7 package, factory calibrated and including the 32kHz crystal, part number EM3028-C7

Best-in-class Real-time Clocks

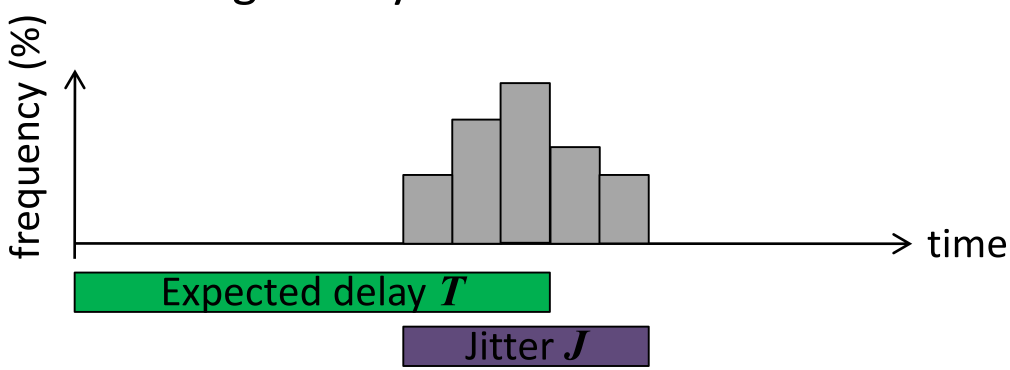


Messages Delays Experience Jitter

- Problem: Jitter in the message delay
  - Various sources of errors (deterministic and non-deterministic)



- Solution: Timestamping packets at the MAC layer
  - Jitter in the message delay is reduced to a few clock ticks



Messaging Delays Influence Factors

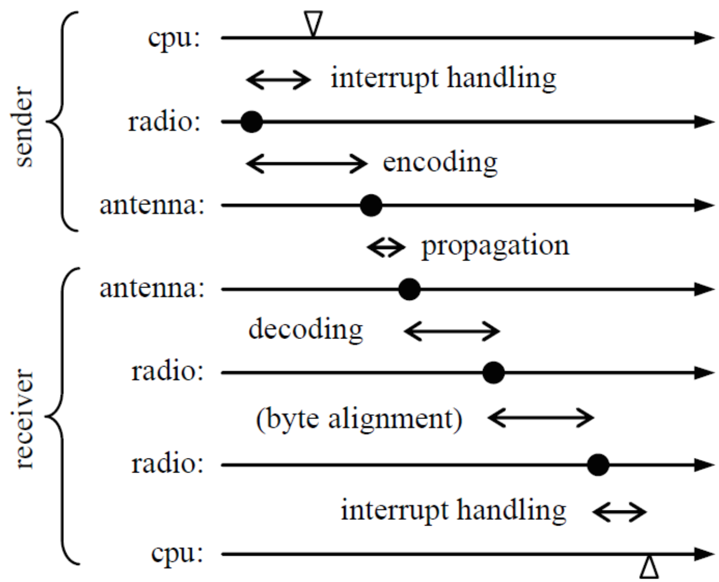


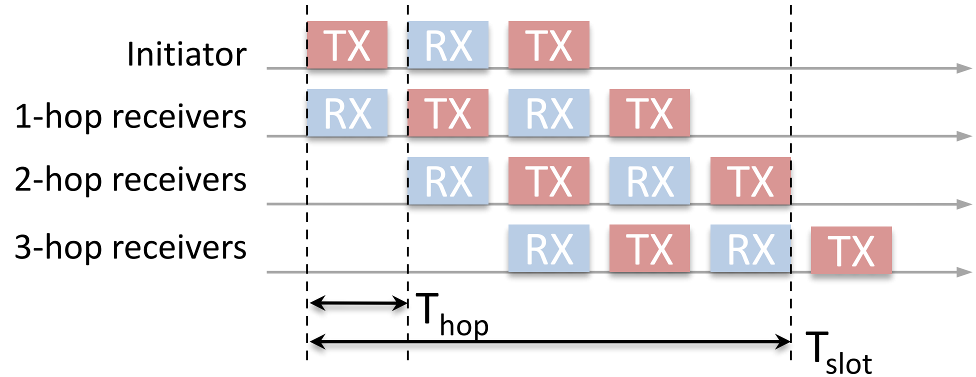
Table 1. The sources of delays in message transmissions

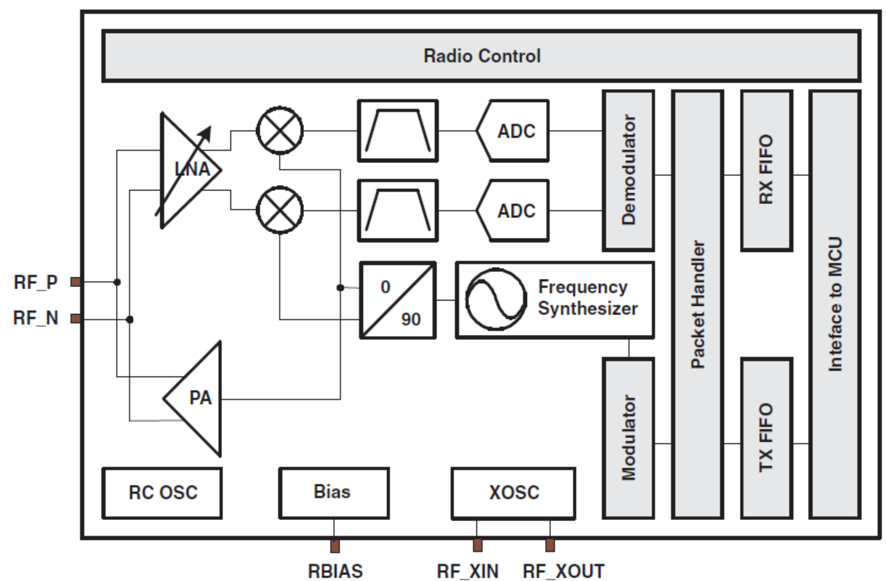
Time	Magnitude	Distribution
Send and Receive	0 – 100 ms	nondeterministic, depends on the processor load
Access	10 – 500 ms	nondeterministic, depends on the channel contention
Transmission / Reception	10 – 20 ms	deterministic, depends on message length
Propagation	< 1µs for distances up to 300 meters	deterministic, depends on the distance between sender and receiver
Interrupt Handling	< 5µs in most cases, but can be as high as 30µs	nondeterministic, depends on interrupts being disabled
Encoding plus Decoding	100 – 200µs, < 2µs variance	deterministic, depends on radio chipset and settings
Byte Alignment	0 – 400µs	deterministic, can be calculated

Critical Technique

MAC LAYER TIMESTAMPING

Details – MAC Layer Timestamping

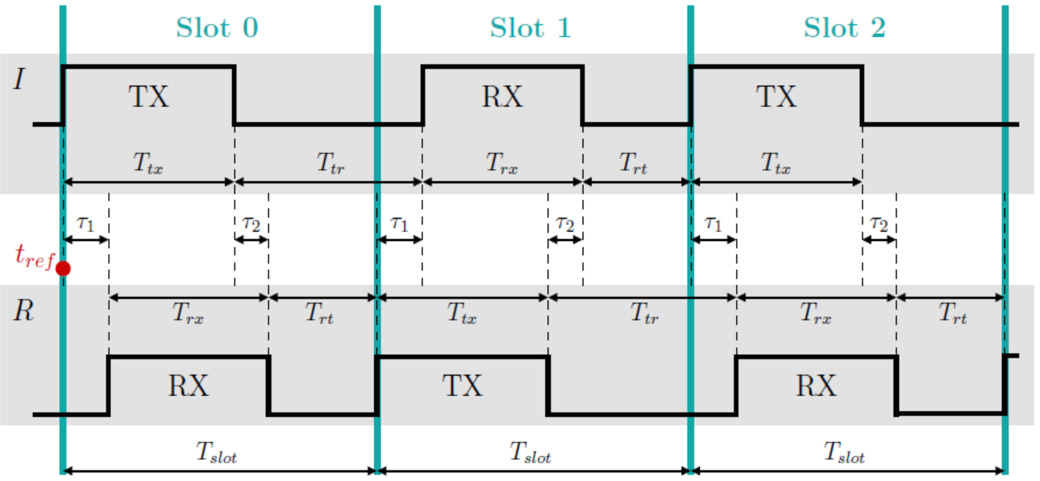




CC1101 radio module integrated into CC430

CC430 Implementation Details

- τ_1: time distance from when the sync word pin goes high at a transmitter (*i.e.*, beginning of the transmission) until when the sync word pin goes high at a receiver (*i.e.*, beginning of the reception).
- τ_2: time distance from when the sync word pin goes low at a transmitter (*i.e.*, end of the transmission) until when the sync word pin goes high at a receiver (*i.e.*, end of the reception).



$$\tau_1 = 13.54\,\mu\text{s}$$
$$\tau_2 = 11.86\,\mu\text{s}$$

→

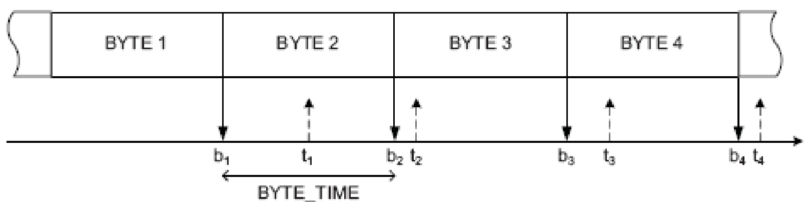
$$\tau_1 - \tau_2 = T_{tx} - T_{rx}$$
$$\tau_1 + \tau_2 = T_{tr} - T_{rt}$$

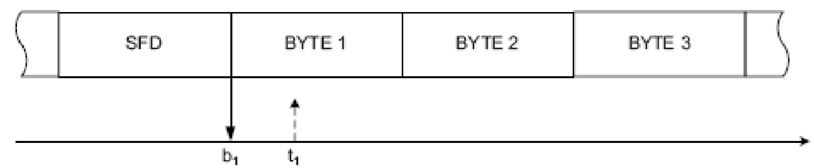
→

$$T_{slot} = T_{rx} + T_{rt} + \tau_1 = T_{tx} + T_{tr} - \tau_1$$

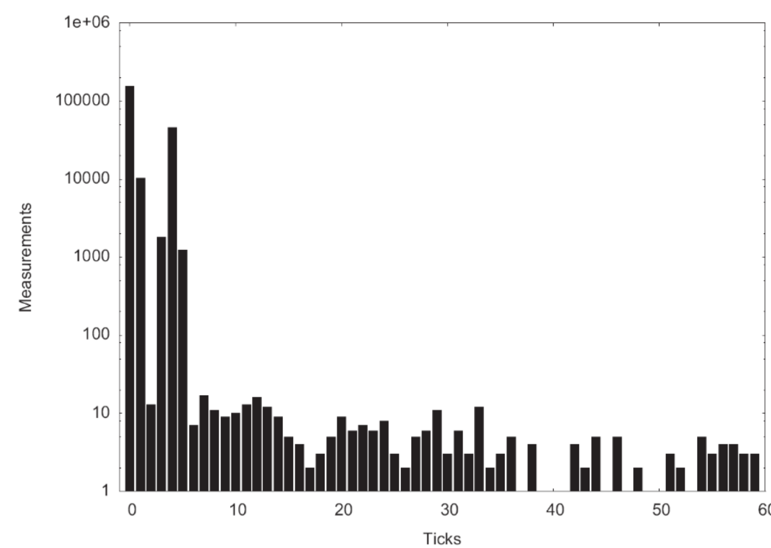
Interrupt Handling Causes Jitter

- Different radio chips use different paradigms
  - Left is a CC1000 radio chip which generates an interrupt with each byte
  - Right is a CC2420 radio chip that generates a single interrupt for the packet after the start frame delimiter is received

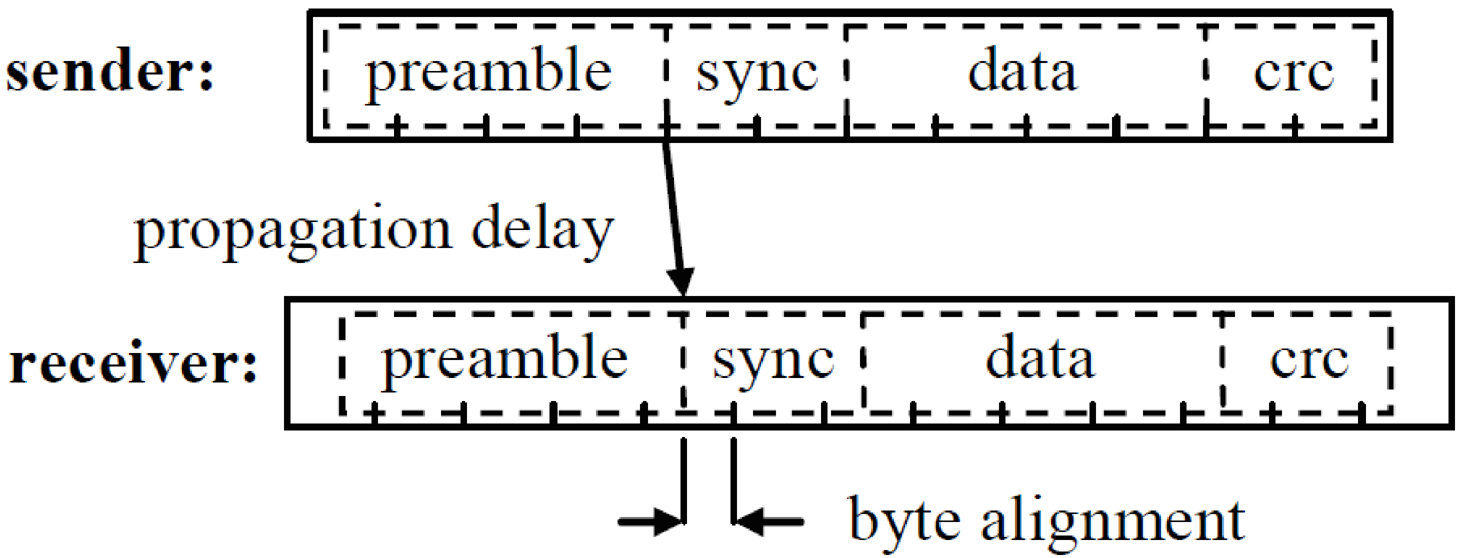




- Still there is quite some variance in transmission delay because of latencies in **interrupt handling**

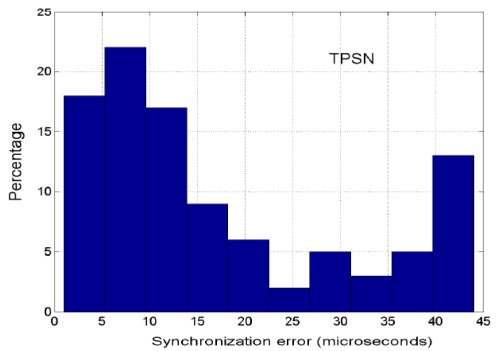


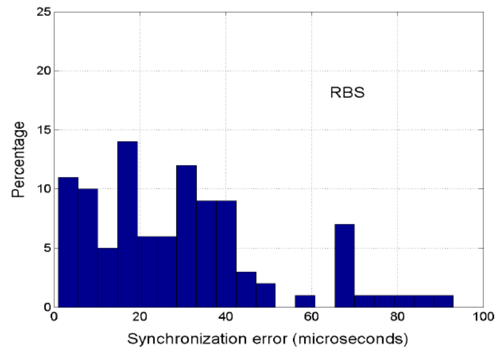
Radio Architectures Offer Sync Points

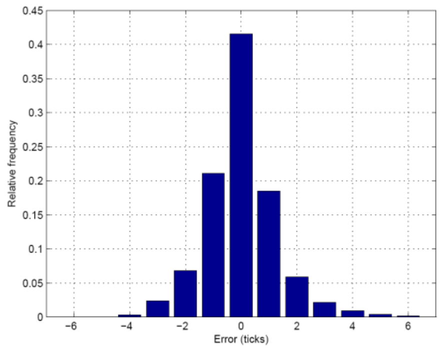


Symmetric Errors

- Many protocols don't even handle single-hop clock synchronization well. On the left figures we see the absolute synchronization errors of TPSN and RBS, respectively. The figure on the right presents a single-hop synchronization protocol minimizing systematic errors







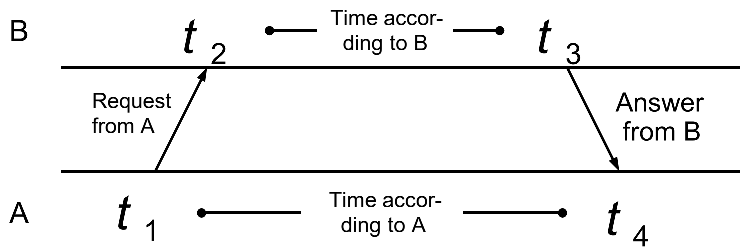
- Even perfectly symmetric errors will sum up over multiple hops
  - In a chain of *n* nodes with a standard deviation σ on each hop, the expected error between head and tail of the chain is in the order of $\textit{cumulative error} = \sigma\sqrt{n}$

Low-Power System Design

NETWORK TIME SYNCHRONIZATION ALGORITHMS

Sender/Receiver Synchronization

- Round-Trip Time (RTT) based synchronization



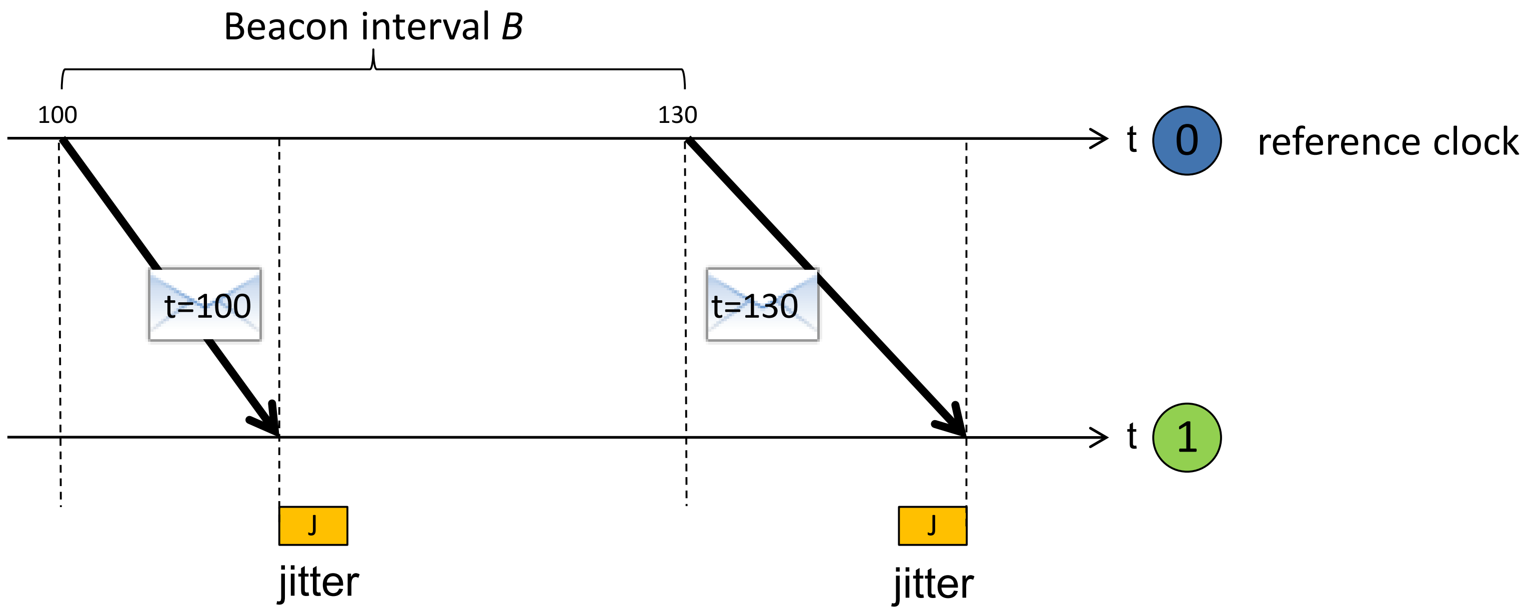
- Receiver synchronizes to the sender's clock
- Propagation delay δ and clock offset θ can be calculated

$$\delta = \frac{(t_4 - t_1) - (t_3 - t_2)}{2}$$

$$\theta = \frac{(t_2 - (t_1 + \delta)) - (t_4 - (t_3 + \delta))}{2} = \frac{(t_2 - t_1) + (t_3 - t_4)}{2}$$

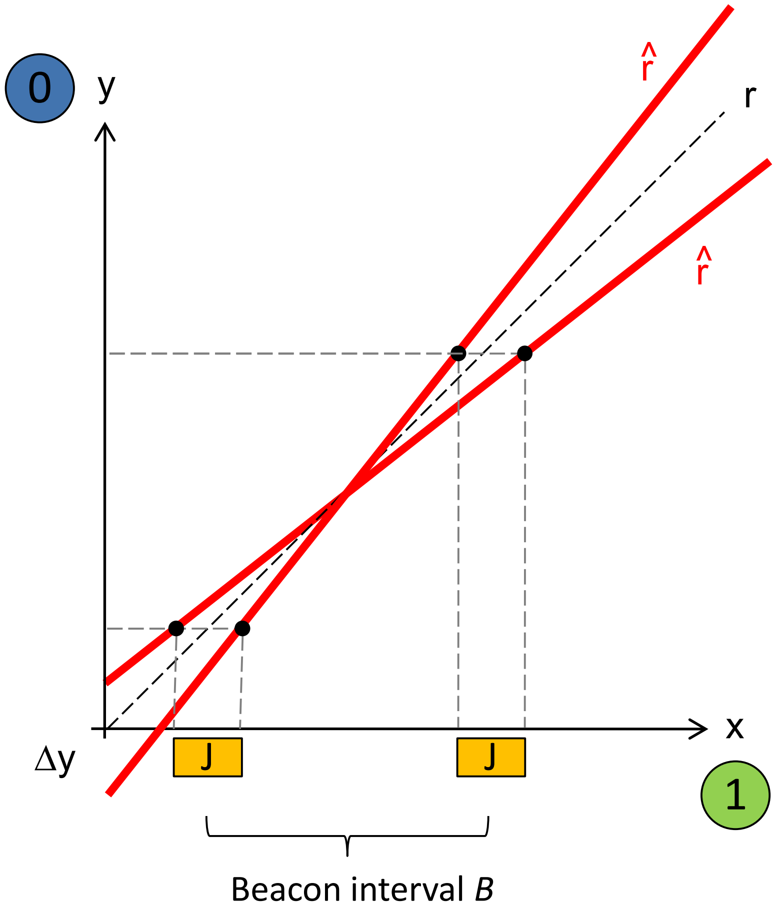
Synchronizing Nodes

- Sending periodic beacon messages to synchronize nodes
- Payload contains local time information



How Accurately Can We Synchronize?

- Message delay jitter affects clock synchronization quality



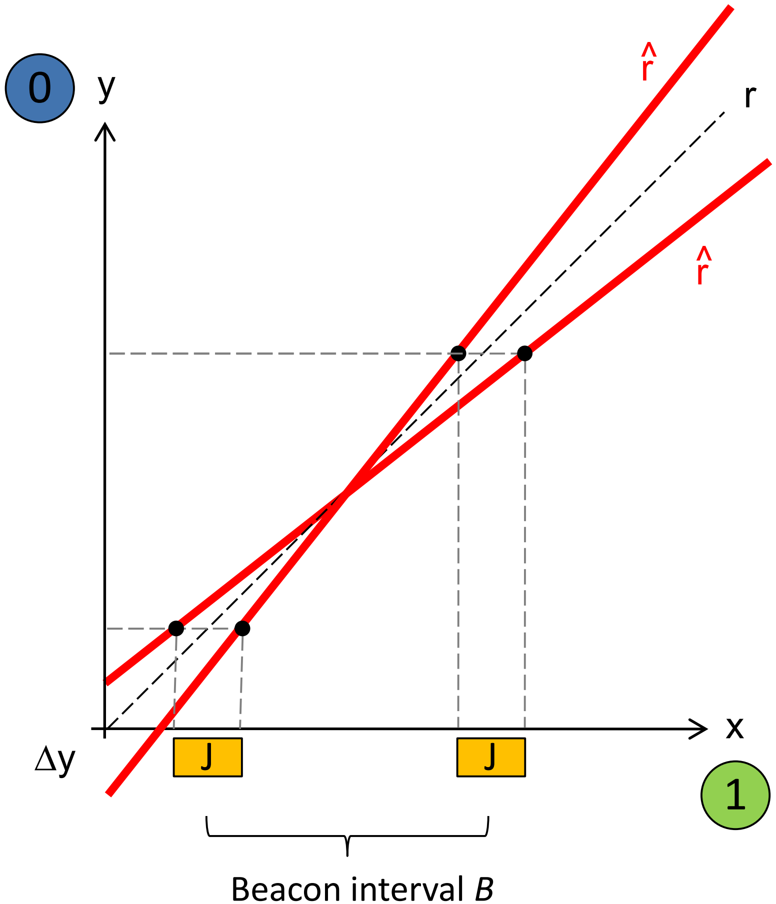
$$y(x) = \hat{r} \cdot x + \Delta y$$

clock offset

relative clock rate (estimated)

Clock Skew between two Nodes

- Lower Bound on the clock skew between two neighbors



Error in the rate estimation:

– Jitter in the message delay

– Beacon interval

– Number of beacons k

$$|\hat{r} - r| \sim \frac{J}{Bk\sqrt{k}}$$

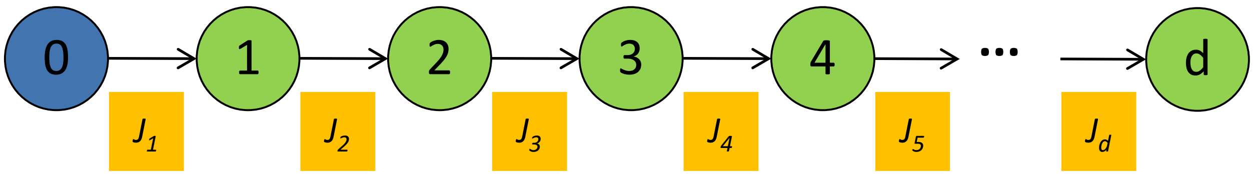
Synchronization error:

$$|\hat{y} - y| \sim \frac{J}{\sqrt{k}}$$

Multi-hop Clock Skew

- Nodes forward their current estimate of the reference clock

  Each synchronization beacon is affected by a **random jitter *J***



- Sum of the jitter grows with the square-root of the distance

$$stddev(J_1 + J_2 + J_3 + J_4 + J_5 + \ldots Jd) = \sqrt{d} \times stddev(J)$$

Single-hop:

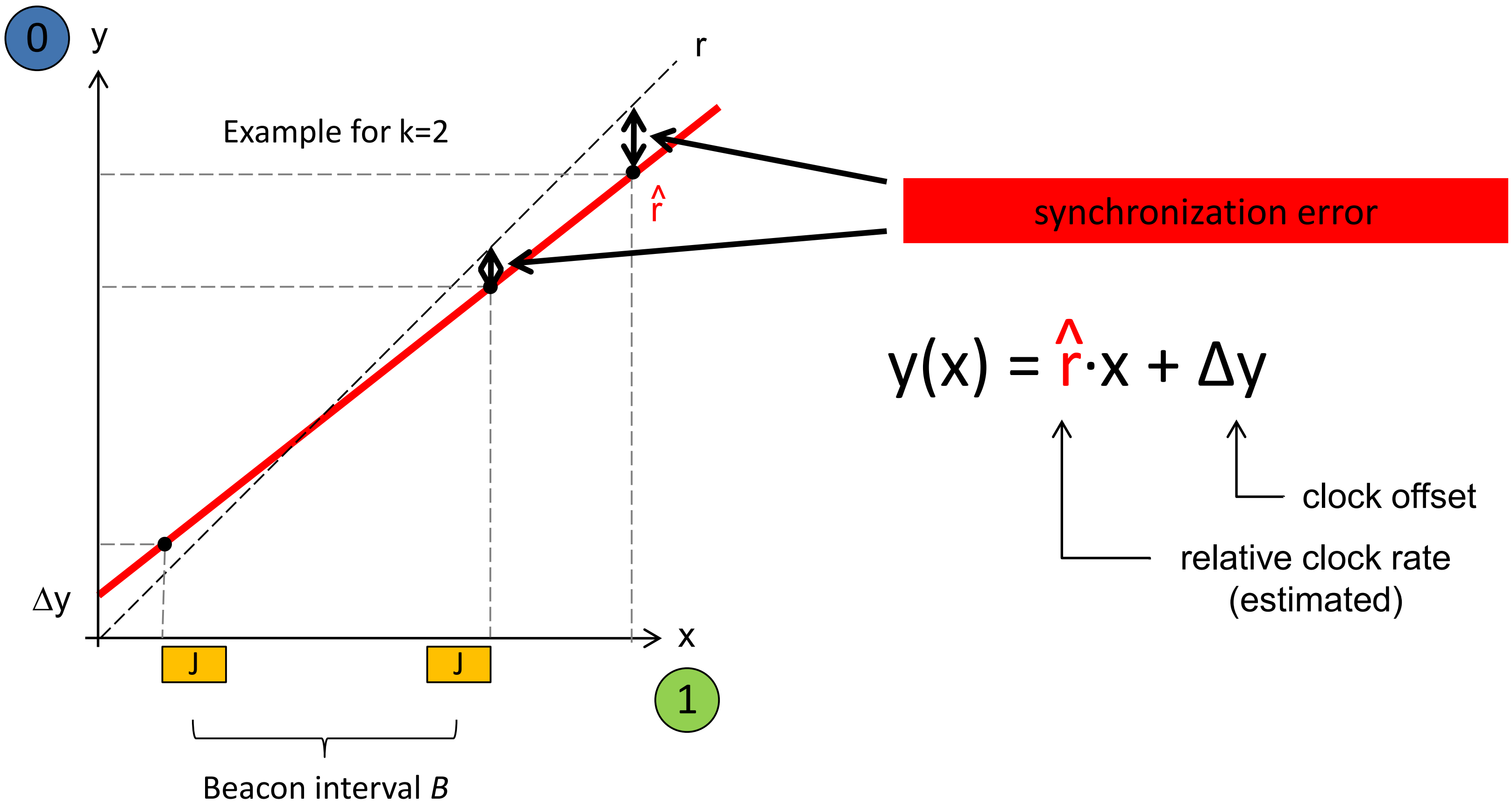
$$|\hat{y} - y| \sim \frac{J}{\sqrt{k}}$$

Multi-hop:

$$|\hat{y} - y| \sim \frac{J\sqrt{d}}{\sqrt{k}}$$

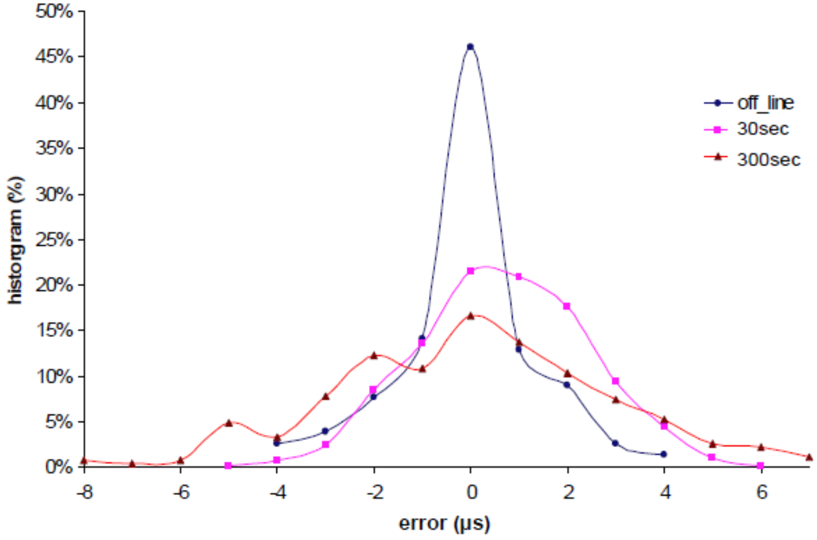
Error Mitigation: Linear Regression

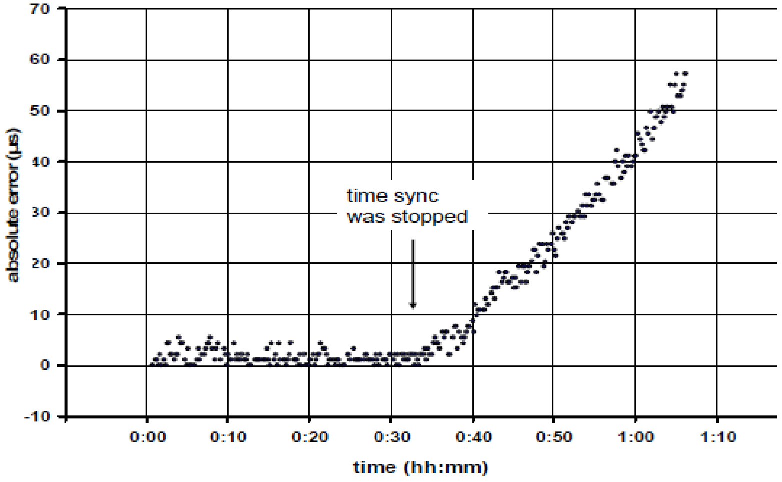
- FTSP uses linear regression to compensate for clock drift
  - Jitter is amplified before it is sent to the next hop



$$y(x) = \hat{r} \cdot x + \Delta y$$

FTSP Offline Regression Errors





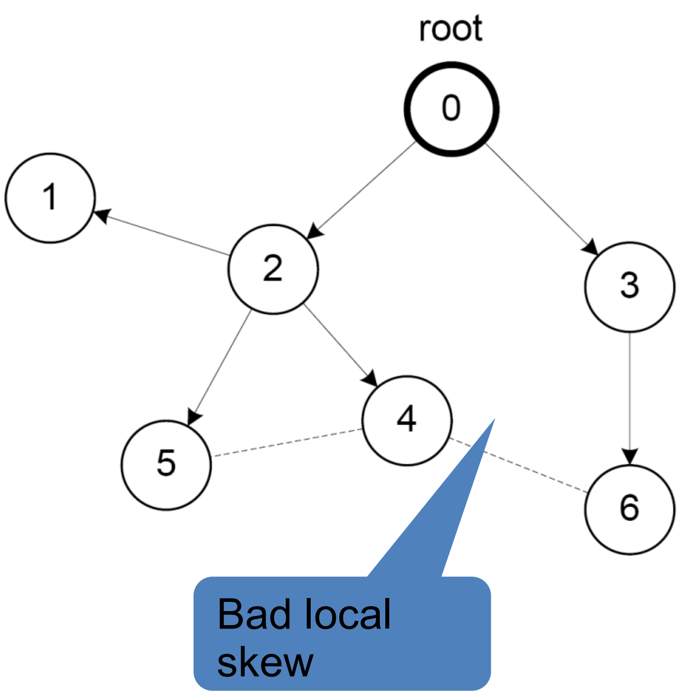
FTSP Time Message Handling

```
event Radio.receive(TimeSyncMsg *msg)
{
  if( msg->rootID < myRootID )
    myRootID = msg->rootID;
  else if( msg->rootID > myRootID
        || msg->seqNum <= highestSeqNum )
    return;

  highestSeqNum = msg->seqNum;
  if( myRootID < myID )
    heartBeats = 0;

  if( numEntries >= NUMENTRIES_LIMIT
     && getError(msg) > TIME_ERROR_LIMIT )
    clearRegressionTable();
  else
    addEntryAndEstimateDrift(msg);
}
```

Clock Synchronization Algorithms

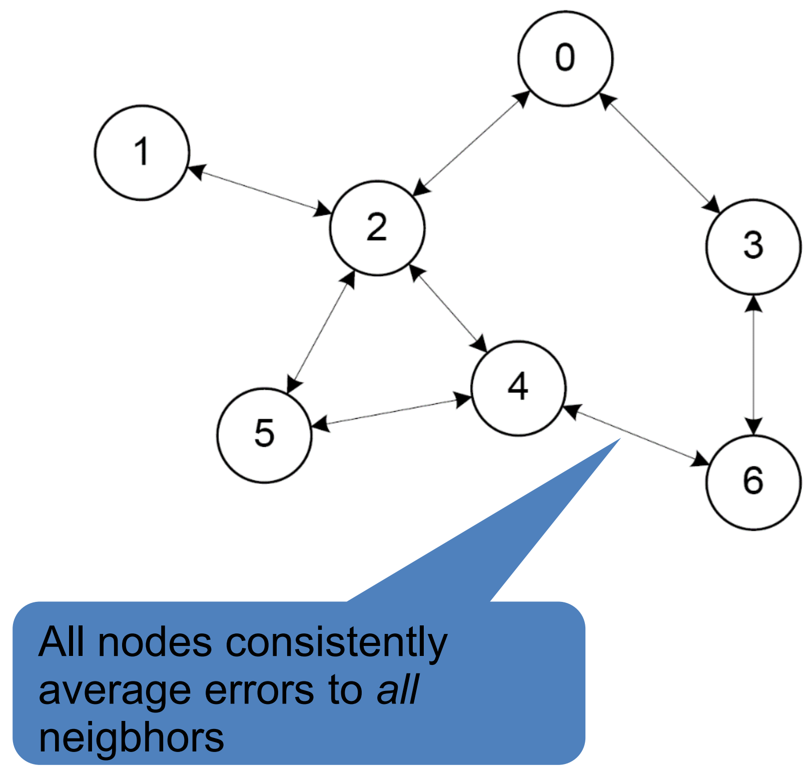
Tree-like Algorithms

e.g. FTSP



Distributed Algorithms

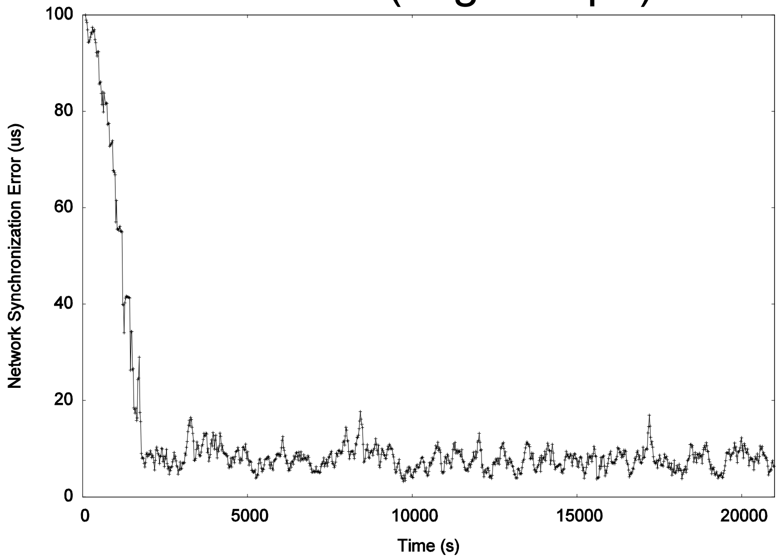
e.g. GTSP



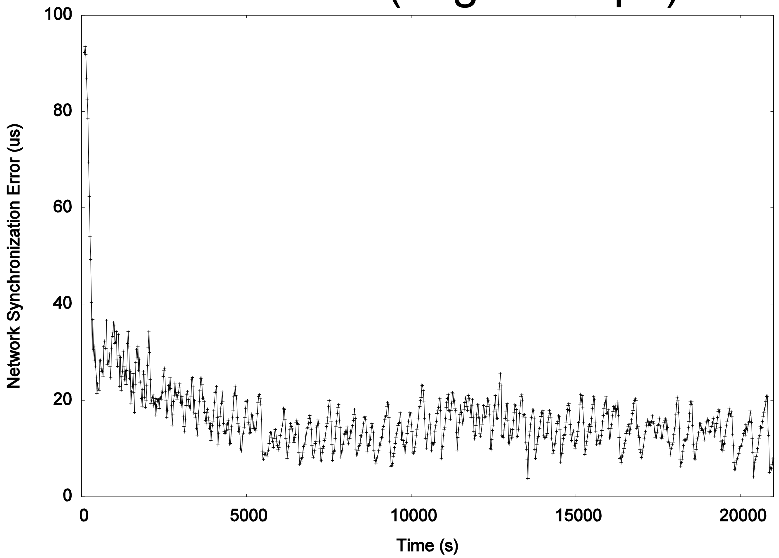
FTSP vs. GTSP: Global Skew

- Network synchronization error (global skew)
  - Pair-wise synchronization error between any two nodes in the network

FTSP (avg: 7.7 µs)



GTSP (avg: 14.0 µs)

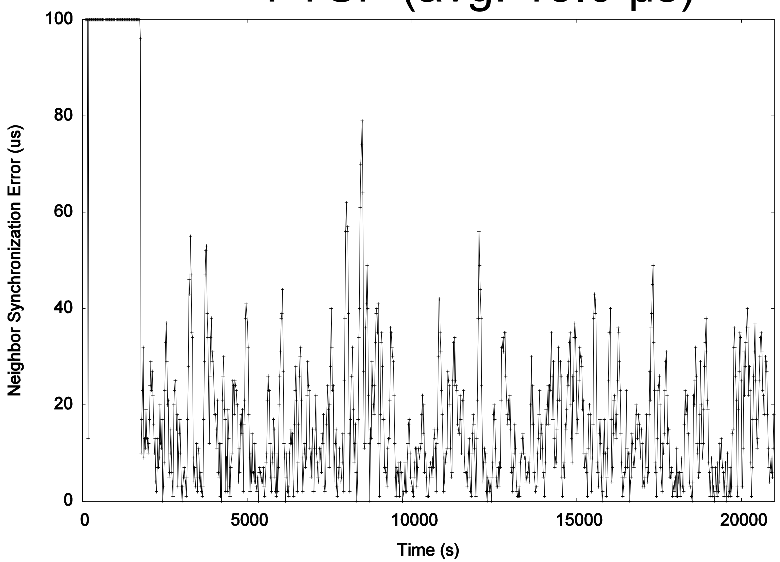


FTSP vs. GTSP: Local Skew

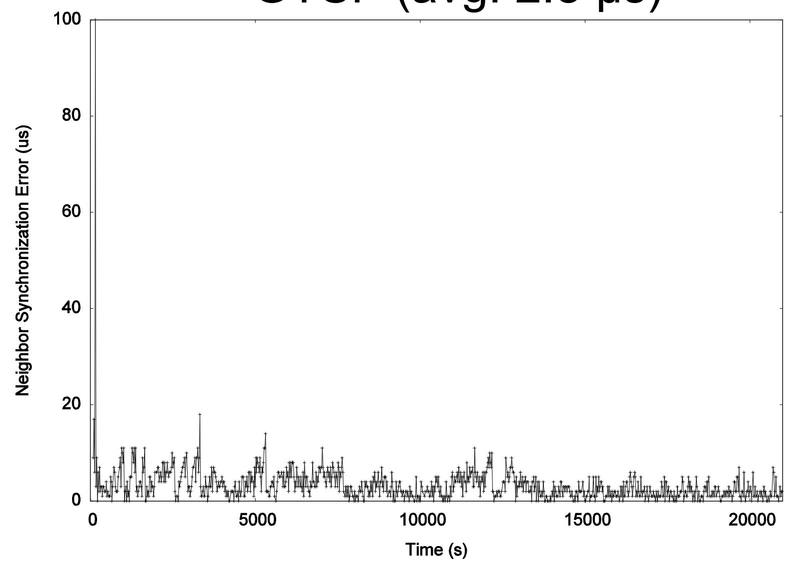
- Neighbor Synchronization error (local skew)
  - Pair-wise synchronization error between neighboring nodes

- Synchronization error between two direct neighbors:

FTSP (avg: 15.0 μs)



GTSP (avg: 2.8 μs)

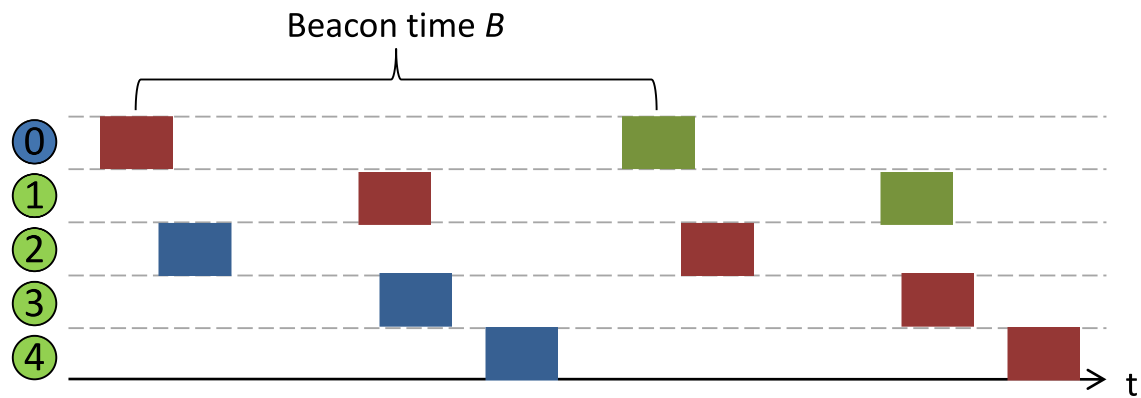


The PulseSync Protocol

- Send fast synchronization pulses through the network
  - Speed-up the initialization phase
  - Faster adaptation to changes in temperature or network topology

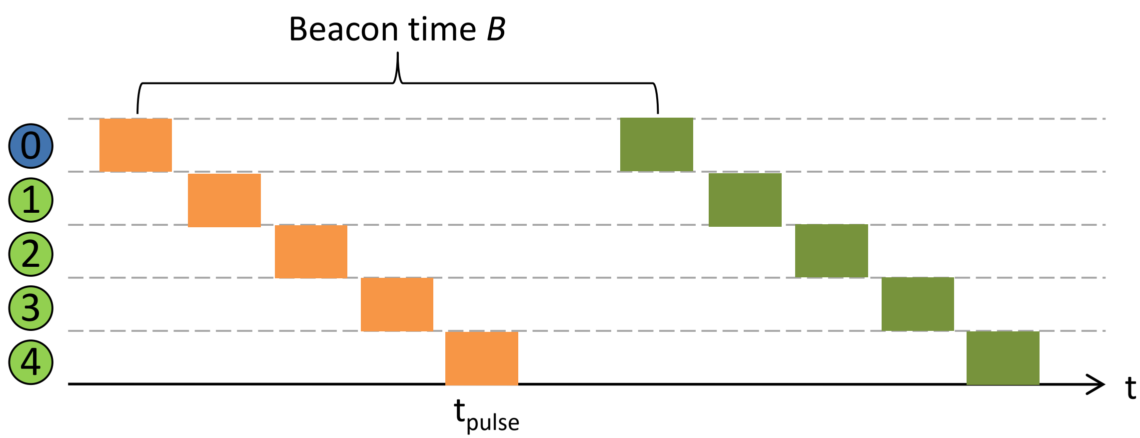
FTSP

Expected time
= D·B/2



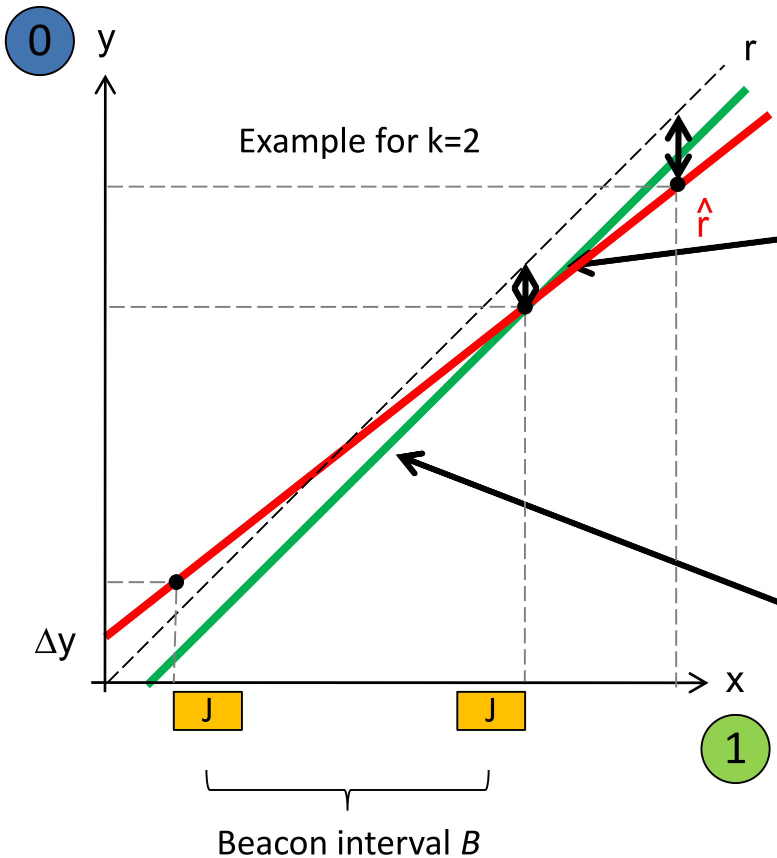
PulseSync

Expected time
= $D \cdot t_{pulse}$



The PulseSync Protocol (2)

- Remove self-amplification of synchronization error
  - Fast flooding cannot completely eliminate amplification



synchronization error

$$y(x) = \hat{r} \cdot x + \Delta y$$

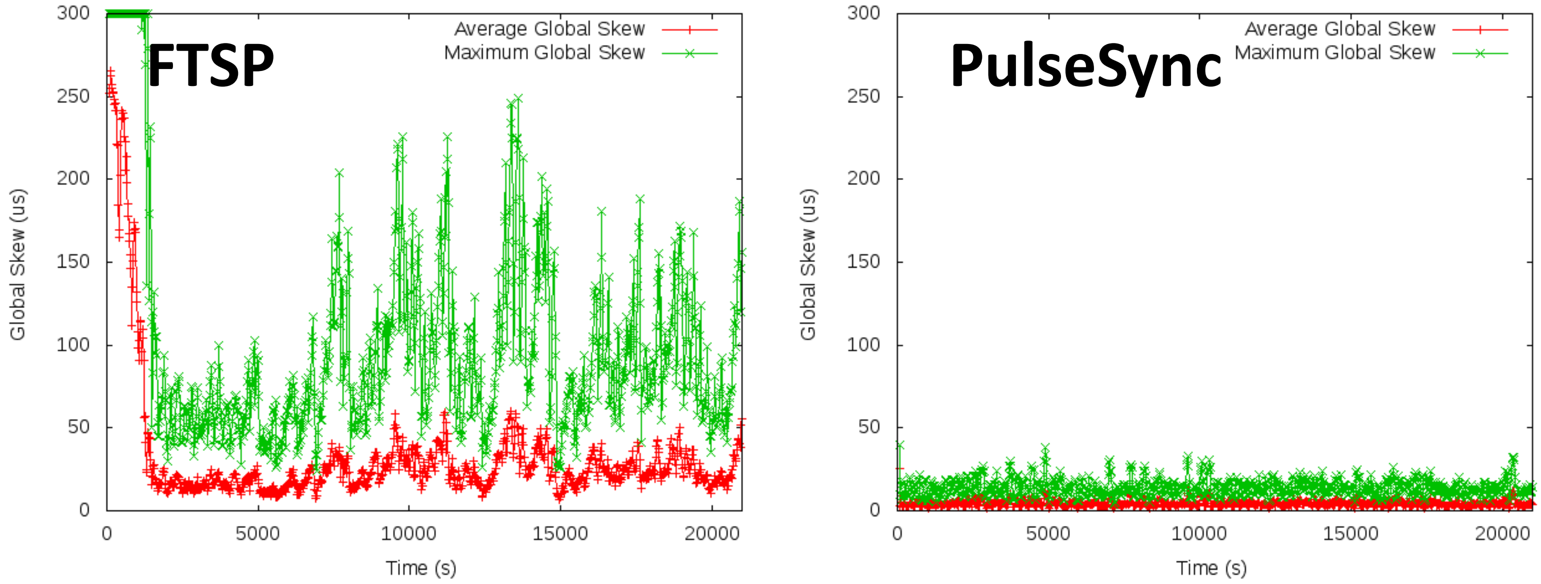
clock offset

relative clock rate (estimated)

The green line is calculated using k measurement points that are statistically *independent* of the red line.

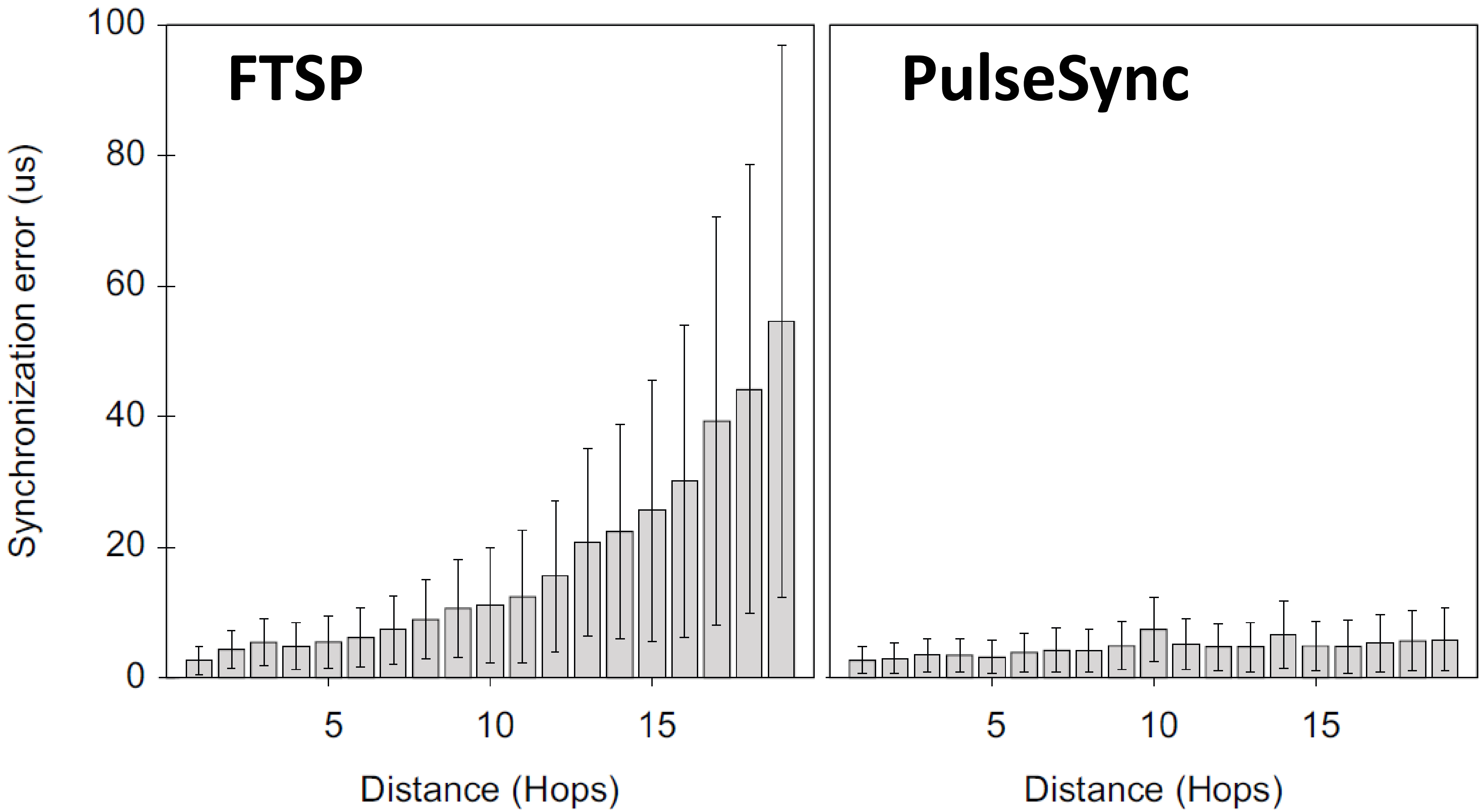
FTSP vs. PulseSync

- Global Clock Skew
  - Maximum synchronization error between any two nodes

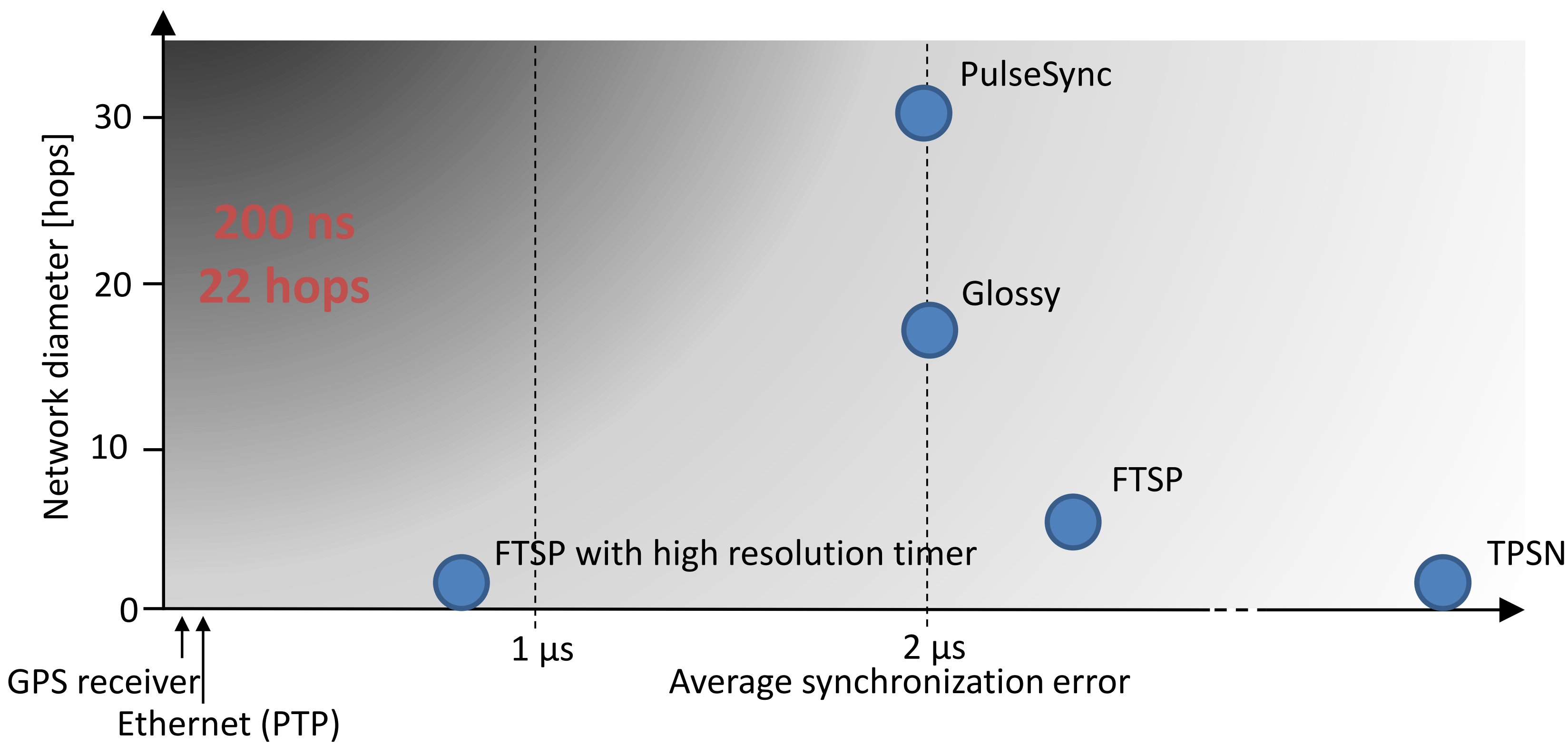


Synchronization Error	FTSP	PulseSync
Average (t>2000s)	23.96 µs	4.44 µs
Maximum (t>2000s)	249 µs	38 µs

FTSP vs. PulseSync



Wireless Multi-hop Time Synchronization



Low-Power System Design

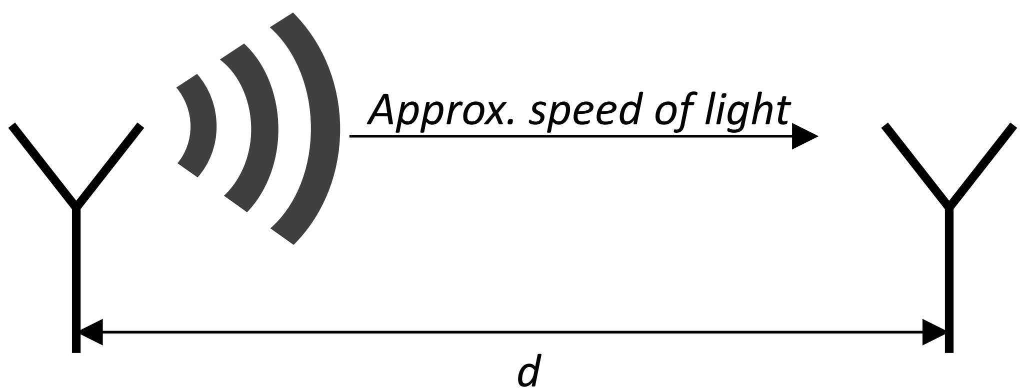
INCORPORATING TIME-OF-FLIGHT

Is Time-of-flight Really Negligible?

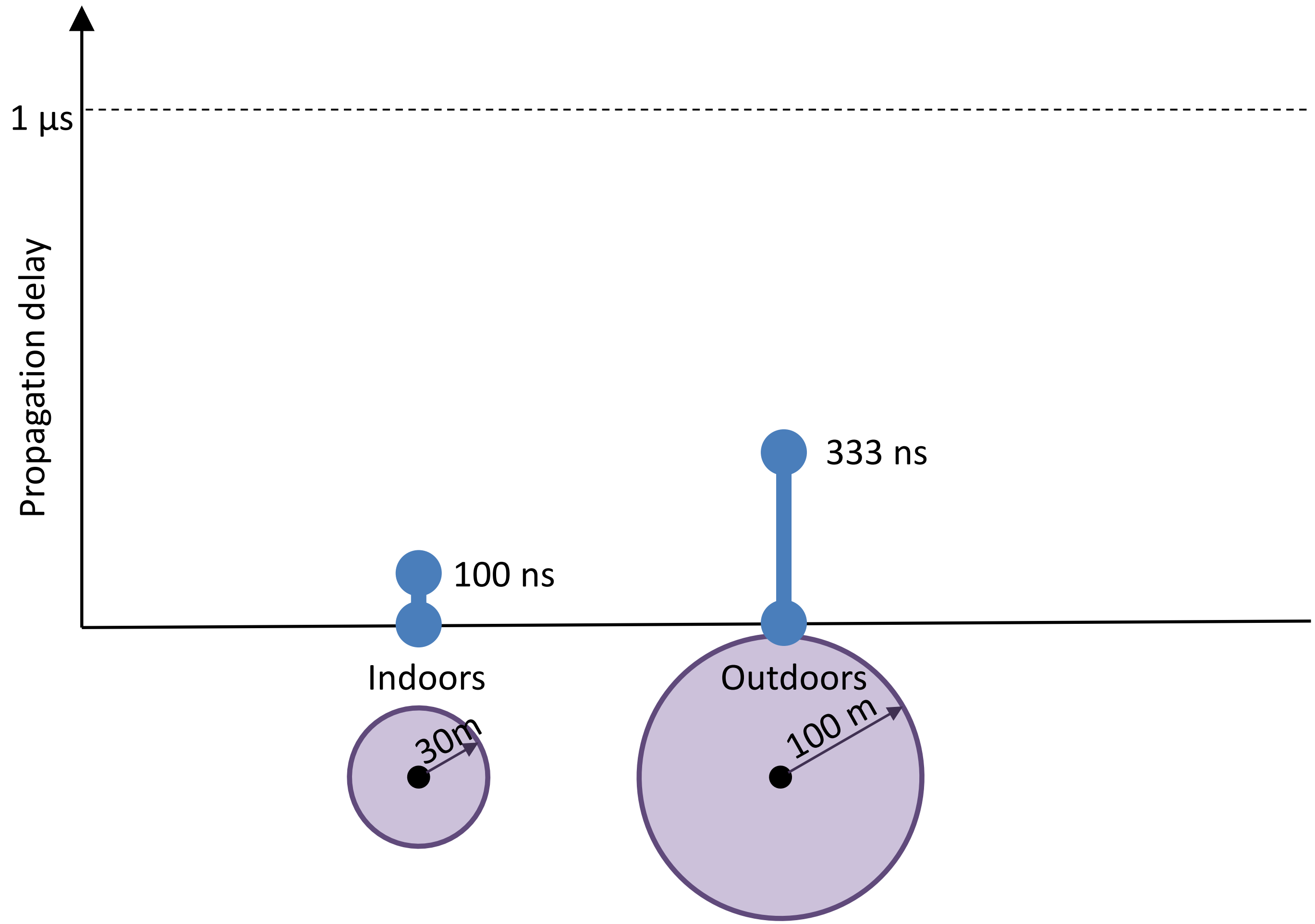
"The absolute value of this delay is negligible as compared to other sources of packet latency." [TPSN 2003]

"… it does not and cannot compensate for the propagation delay. This is not a major limitation of the approach in typical WSN…" [FTSP 2004]

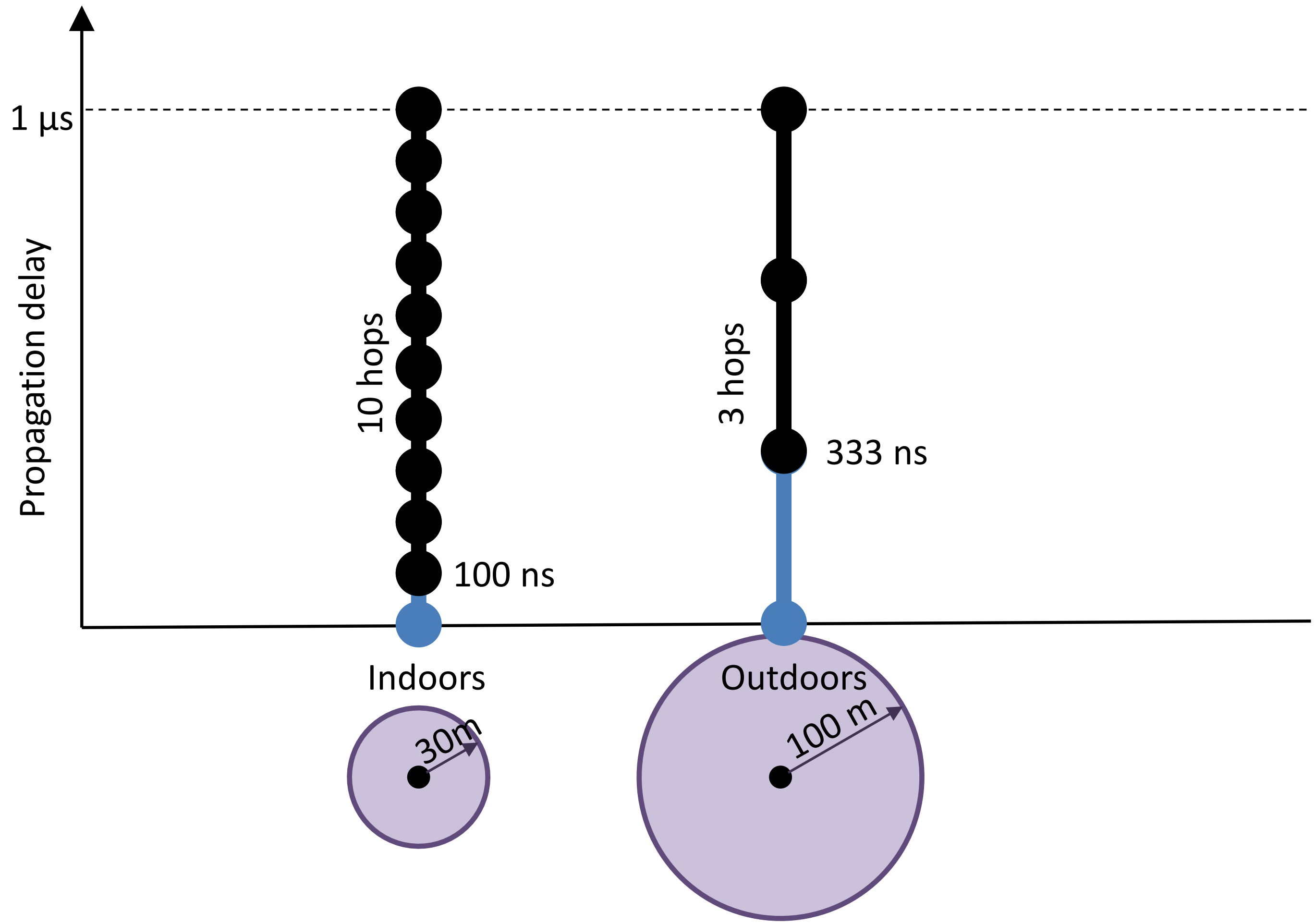
"… over short distances (less than 300 meters) its duration is negligible (less than one microsecond)." [RATS 2006]



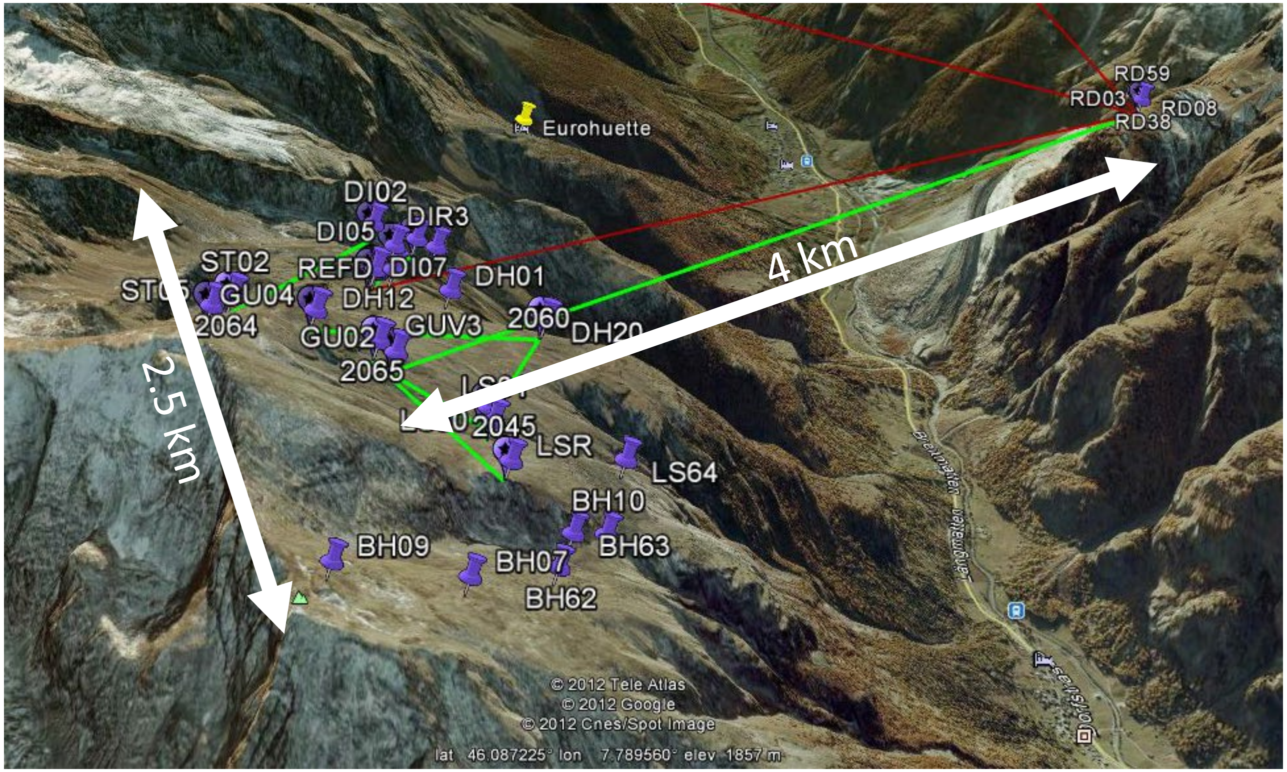
Time-of-flight Matters



Time-of-flight Matters



Outdoor Distances Might Be Long



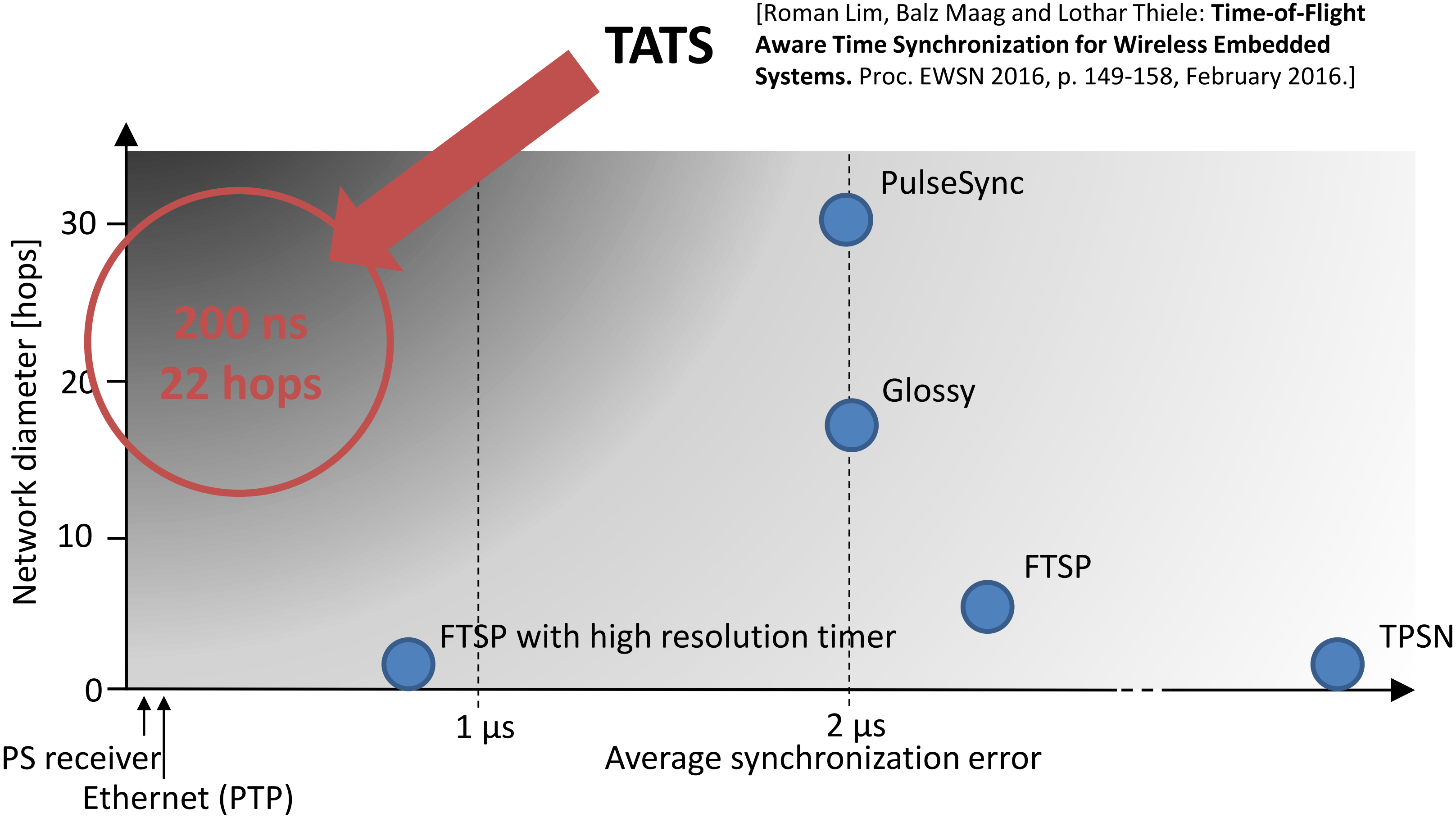
Deployment of the PermaSense Project [1] in the Swiss Alps

Propagation delay 13.33 µs

[1] www.permasense.ch

Wireless Multi-hop Time Synchronization

[Roman Lim, Balz Maag and Lothar Thiele: **Time-of-Flight Aware Time Synchronization for Wireless Embedded Systems.** Proc. EWSN 2016, p. 149-158, February 2016.]

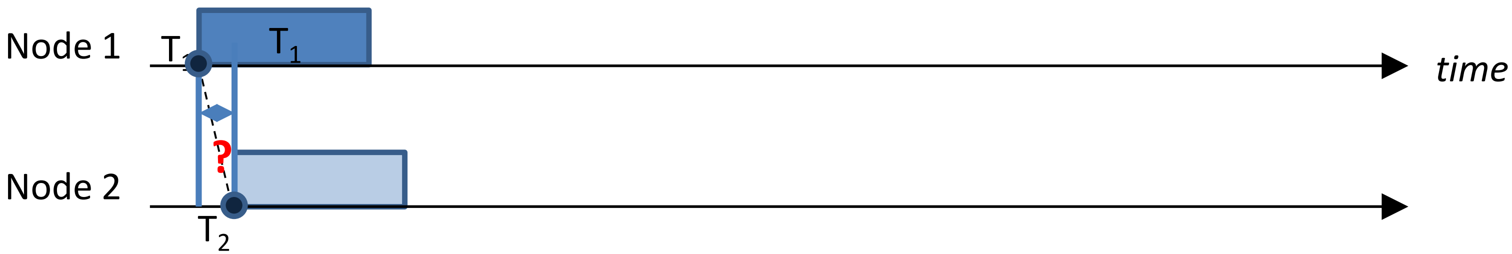


Ingredients for Accurate Synchronization

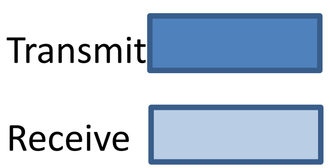
	TPSN	FTSP	PulseSync	Glossy	TATS *
MAC-layer timestamping	✓	✓	✓	✓	✓
Linear regression for offset and clock rate estimation		✓	✓		✓
Two-way delay measurements	✓				✓
Fast flooding			✓	✓	✓

* **T**ime-of-flight **A**ware **T**ime **S**ynchronization

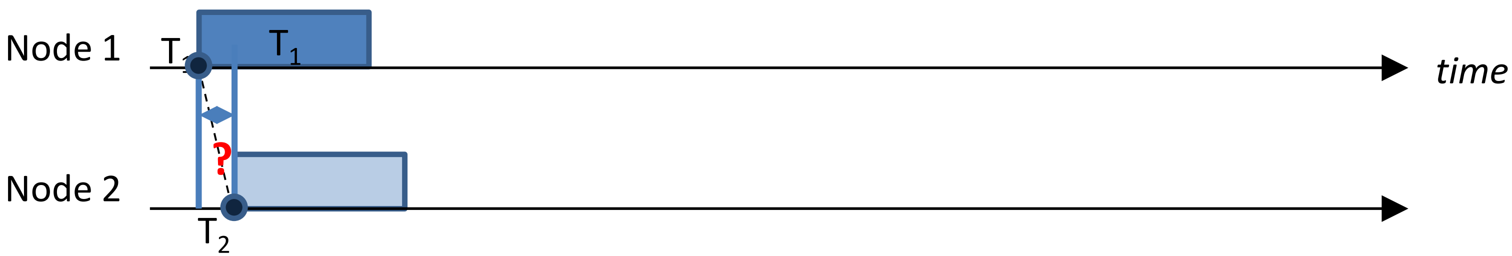
Synchronizing a Pair of Nodes



$Time_1(T_2) = T_1 + ?$

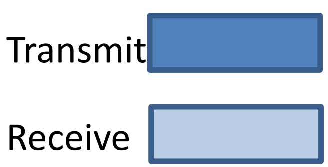


Synchronizing a Pair of Nodes

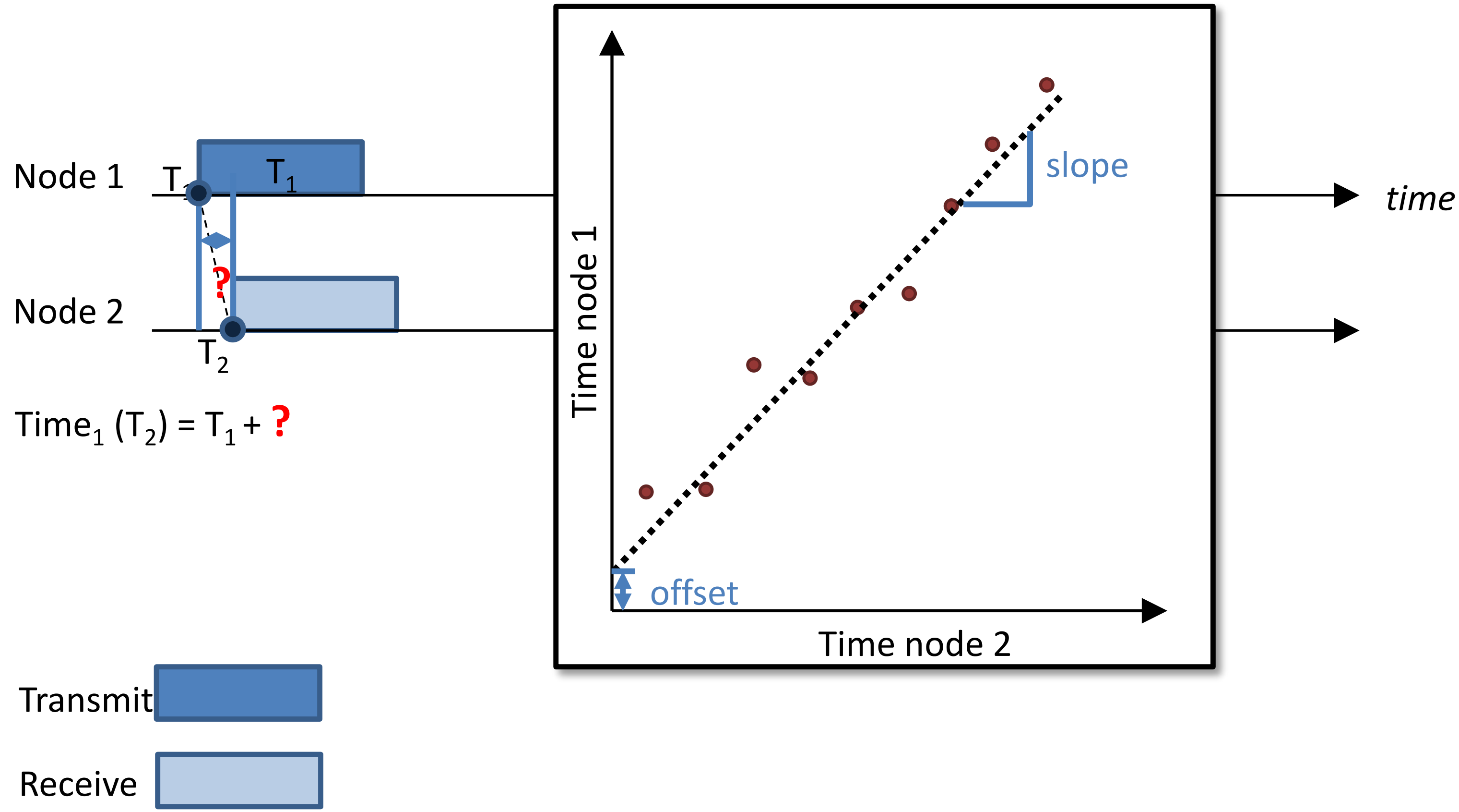


$Time_1(T_2) = T_1 + ?$

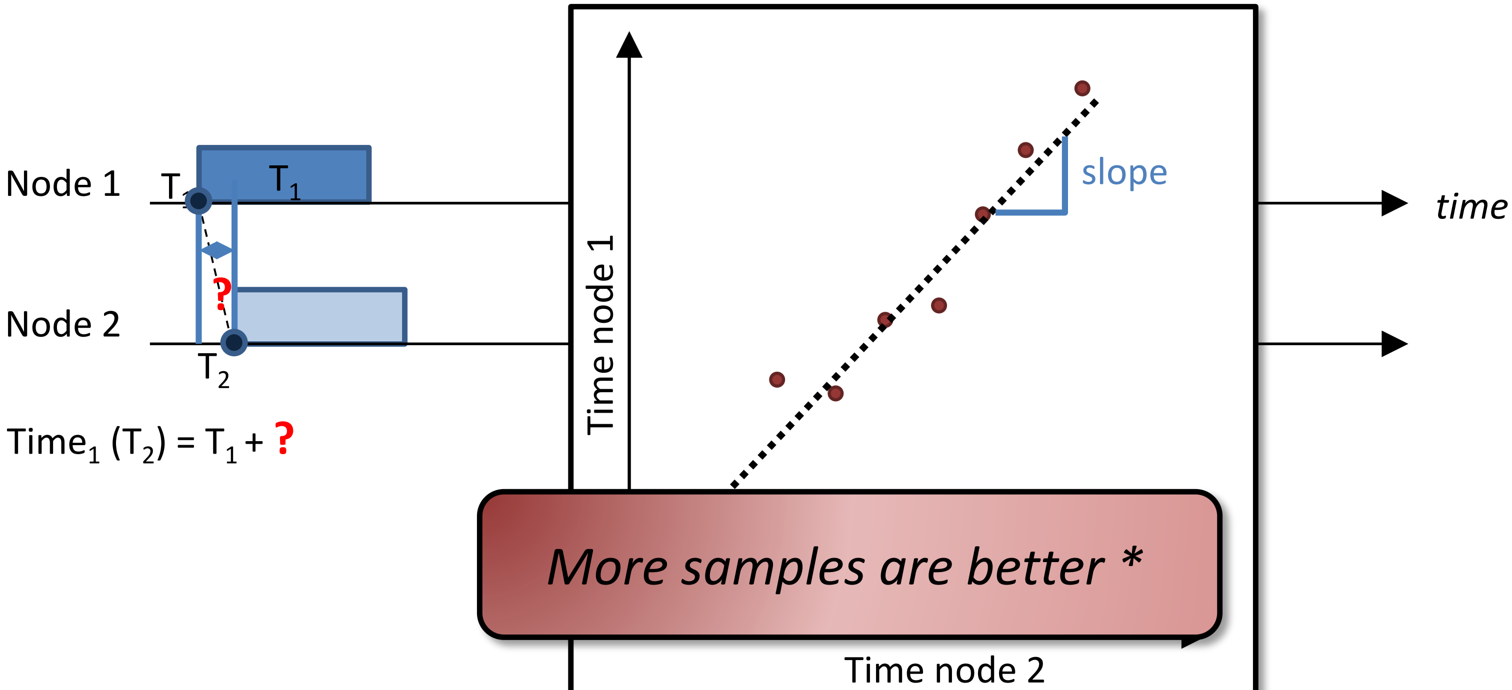
We need accurate packet timestamps with sufficient time resolution

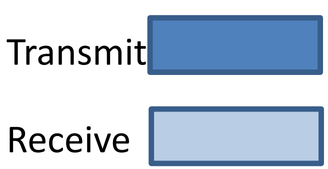


Synchronizing a Pair of Nodes



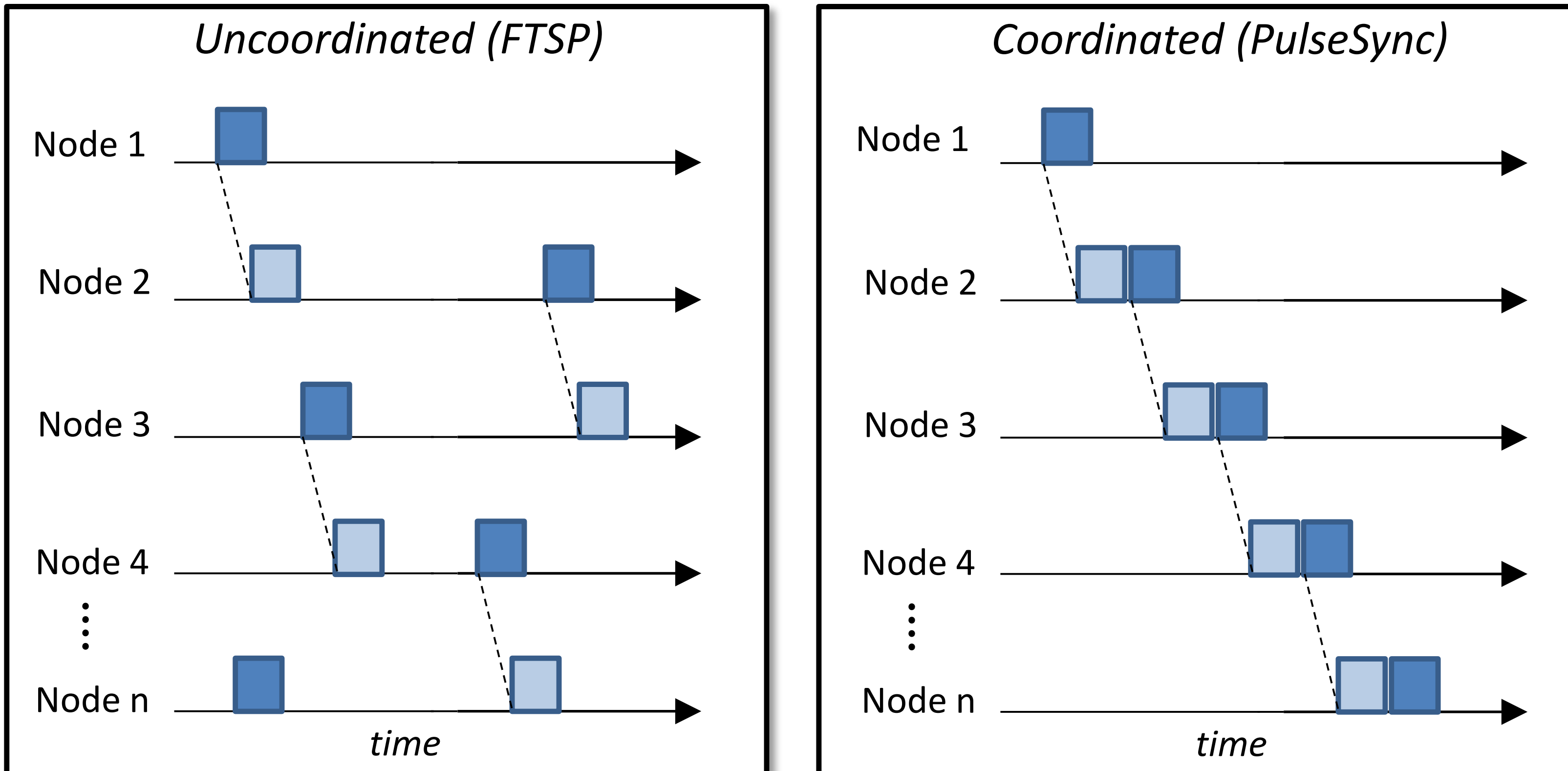
Synchronizing a Pair of Nodes





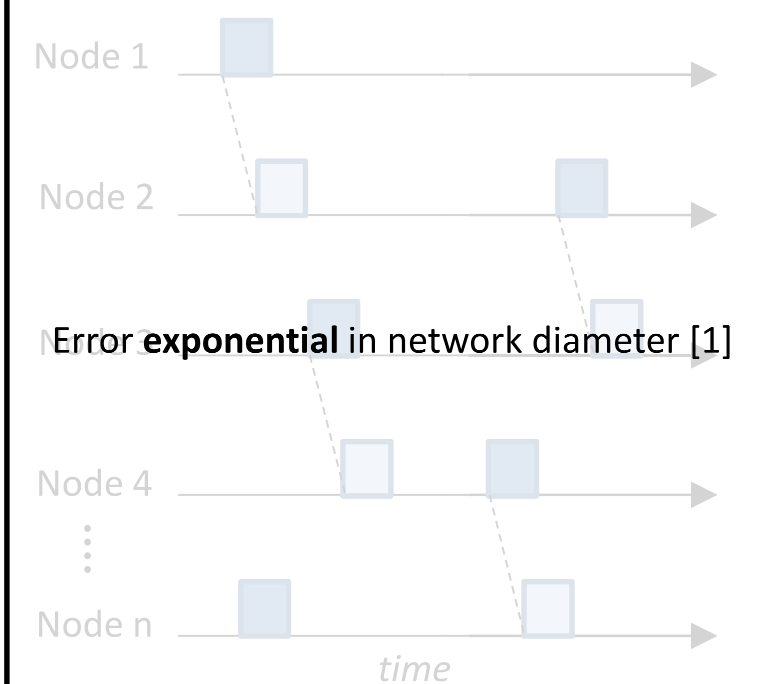
* Up to a time interval where clocks exhibit non-linear behavior

Synchronization Based on Network Flooding



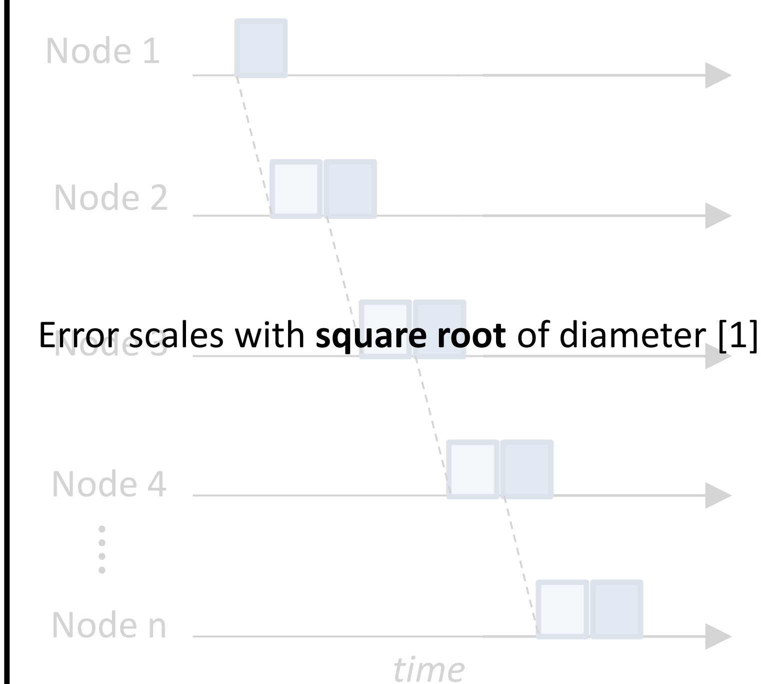
Synchronization Based on Network Flooding

Uncoordinated (FTSP)



Error **exponential** in network diameter [1]

Coordinated (PulseSync)

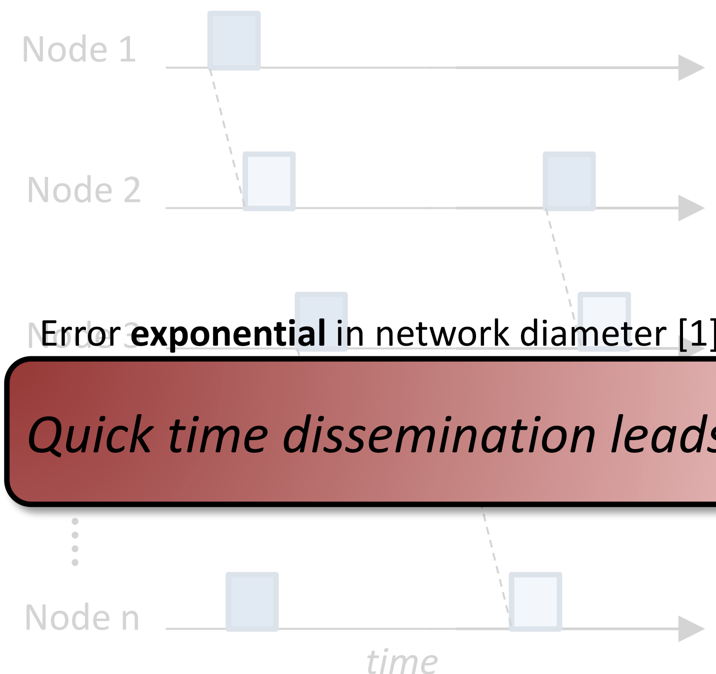


Error scales with **square root** of diameter [1]

[1] C. Lenzen et al, Optimal clock synchronization in networks, SenSys 2009

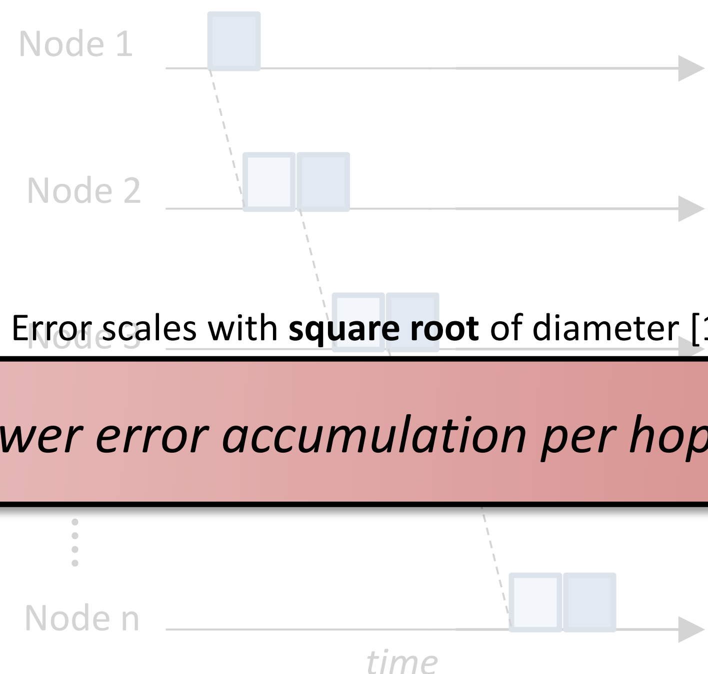
Synchronization Based on Network Flooding

Uncoordinated (FTSP)



Error **exponential** in network diameter [1]

Coordinated (PulseSync)



Error scales with **square root** of diameter [1]

Quick time dissemination leads to lower error accumulation per hop

[1] C. Lenzen et al, Optimal clock synchronization in networks, SenSys 2009

TATS MAC Layer Timestamping

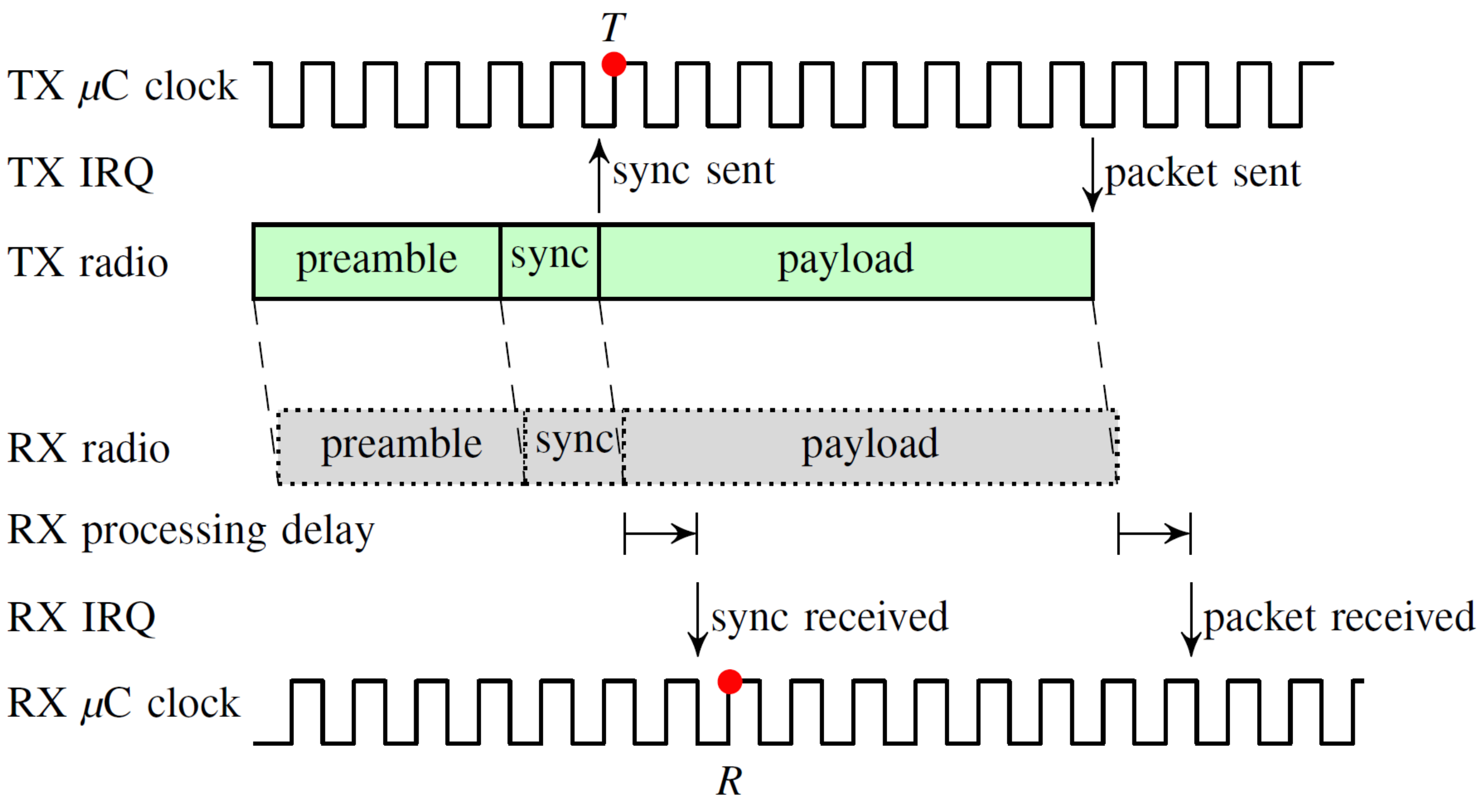
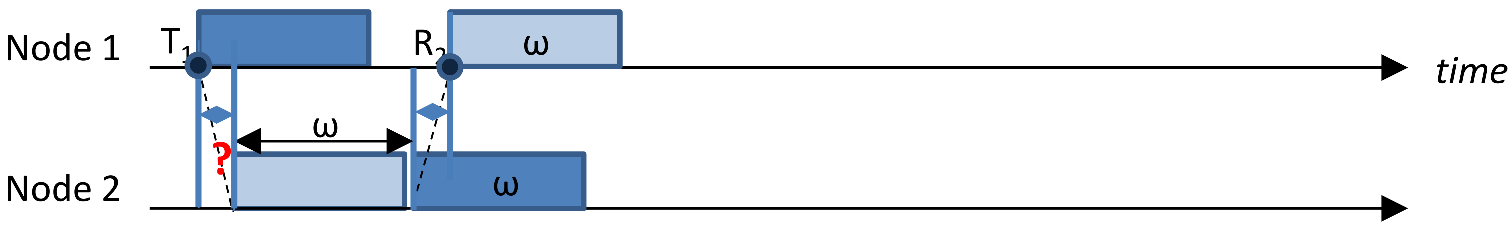
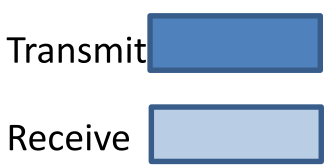


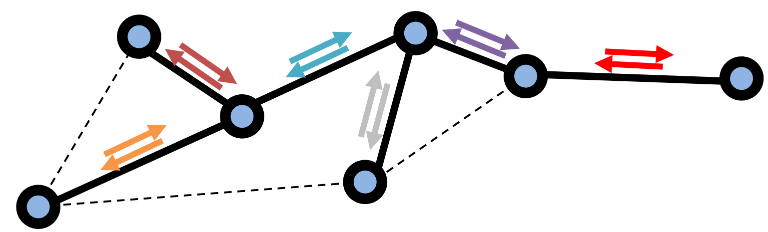
Figure 5. Timestamps for one message transmission. *Timestamps T and R are inaccurate due to asynchronous clocks and uncertainties introduced with radio modulation.*

Propagation Delay Measurement



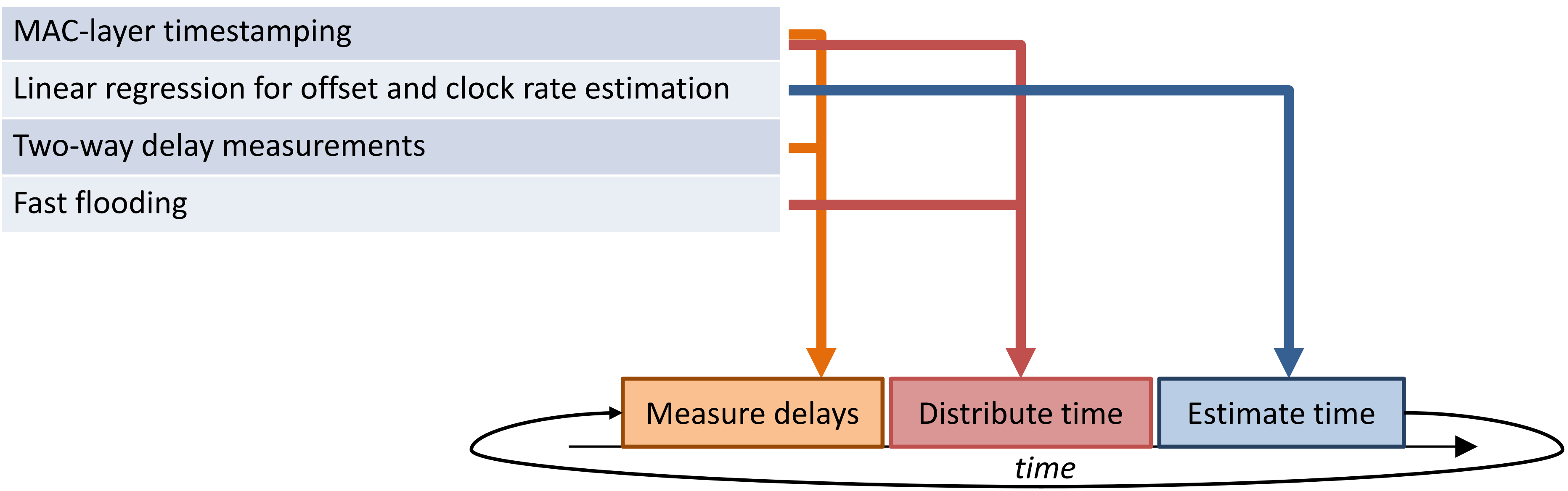
Propagation delay: $(R_2 - T_1 - \omega) / 2$



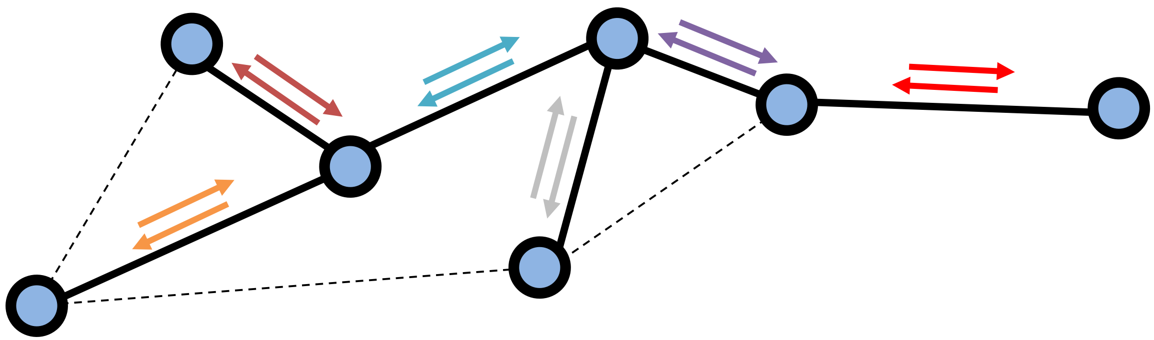


Two-way delay measurement on each link

Putting it Together

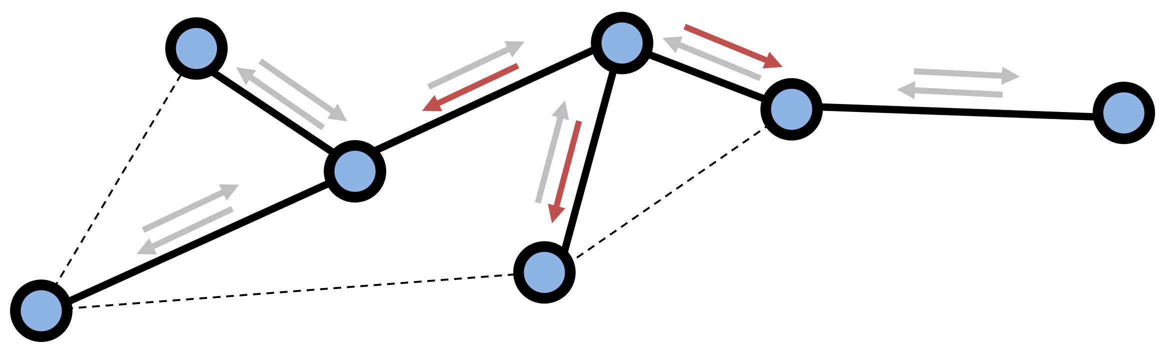


Can we measure delays using only ***one packet per node***?



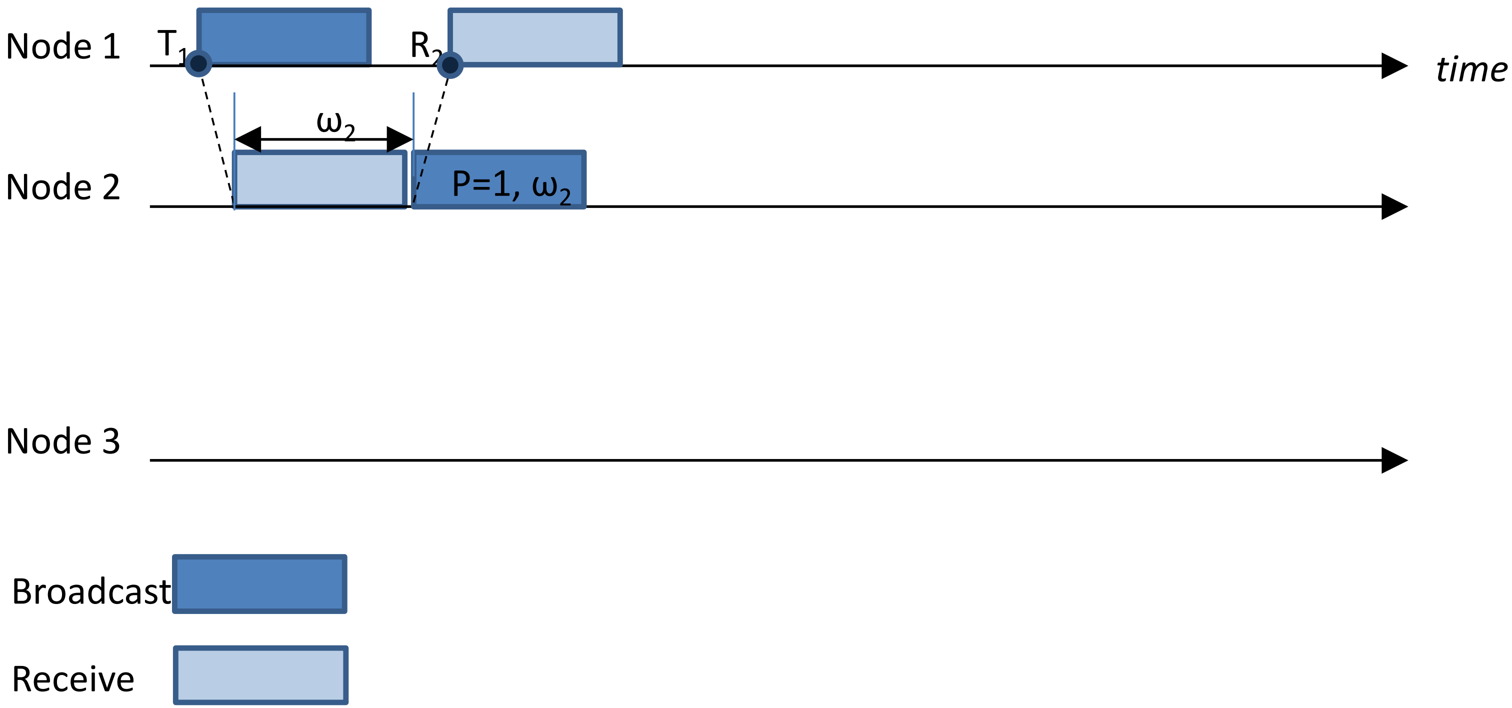
Less transmissions
Fits into existing flooding communication schemes
No need for explicit tree topology creation

Can we measure delays using only ***one packet per node***?

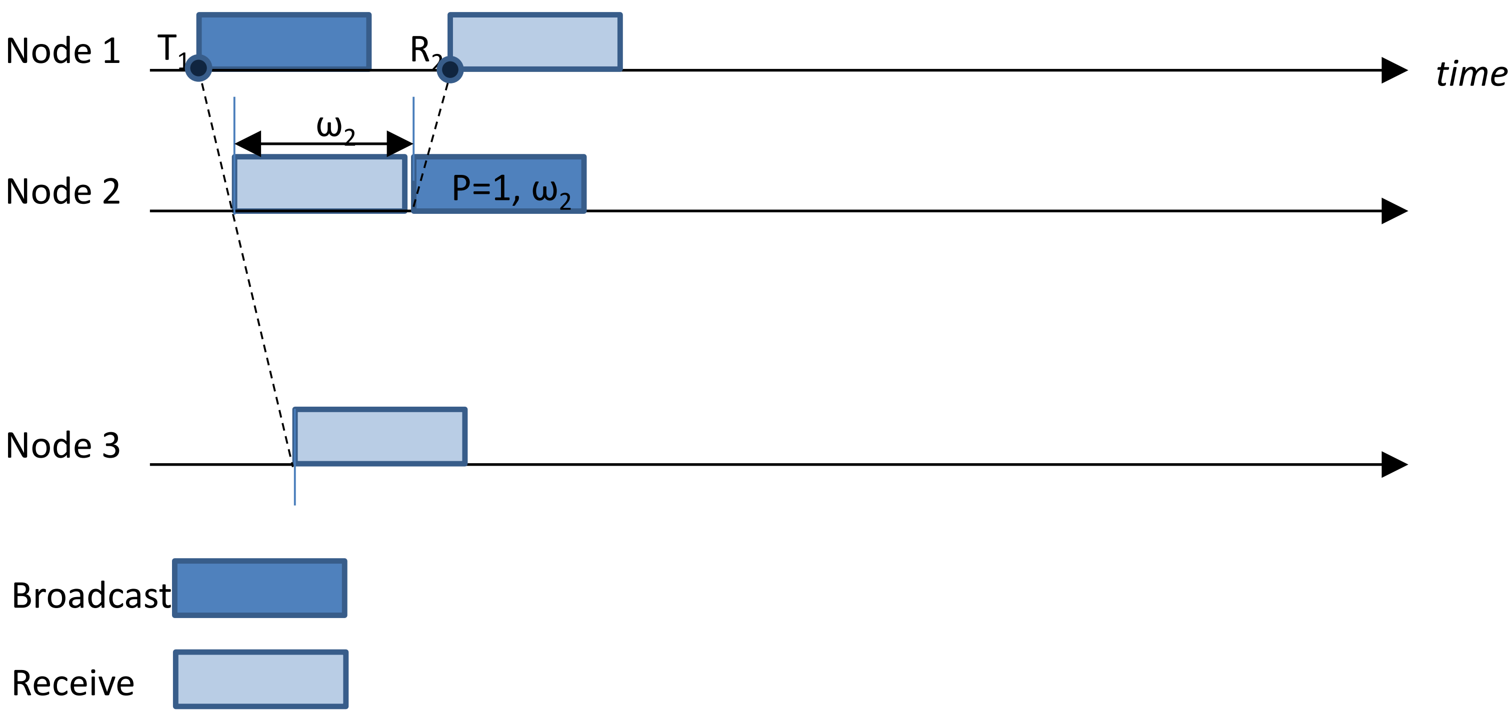


Less transmissions
Fits into existing flooding communication schemes
No need for explicit tree topology creation

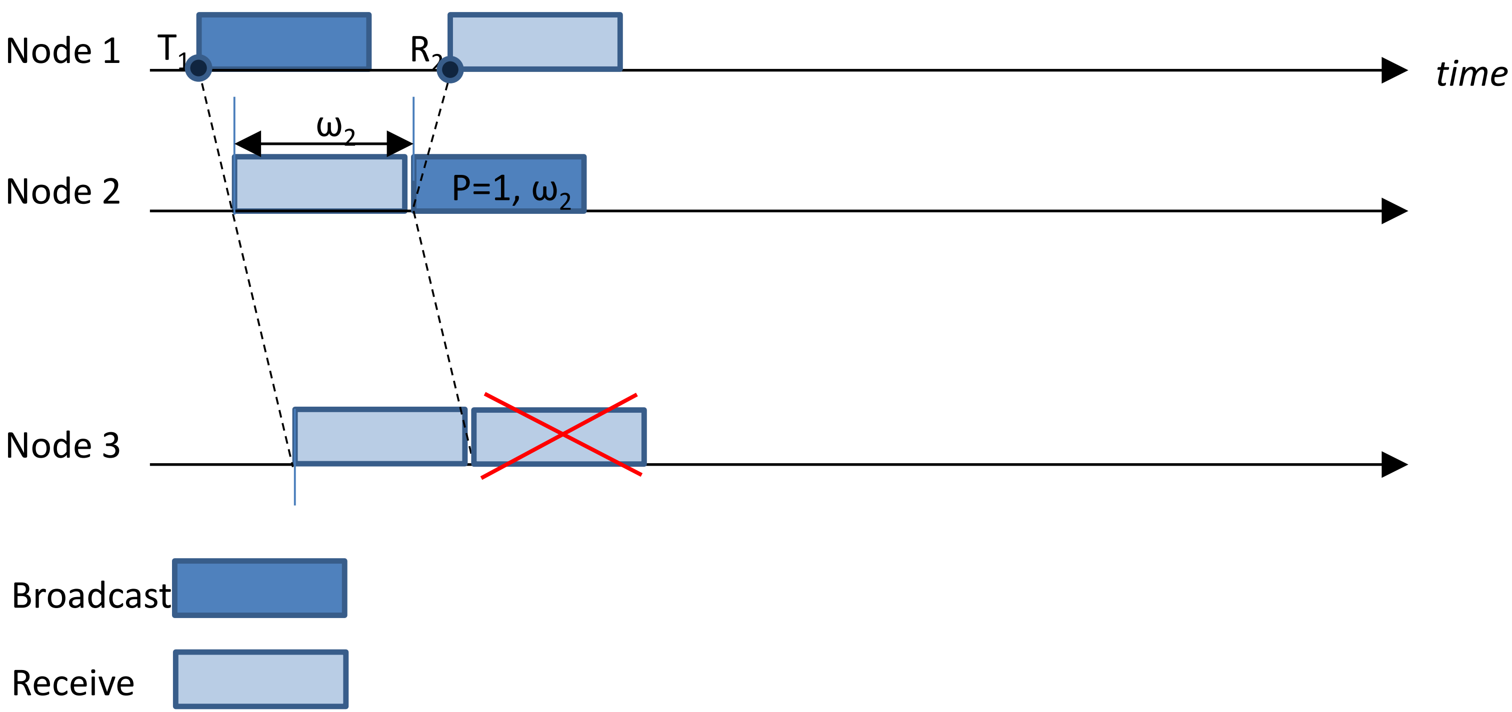
Propagation Delay Measurements in Floods



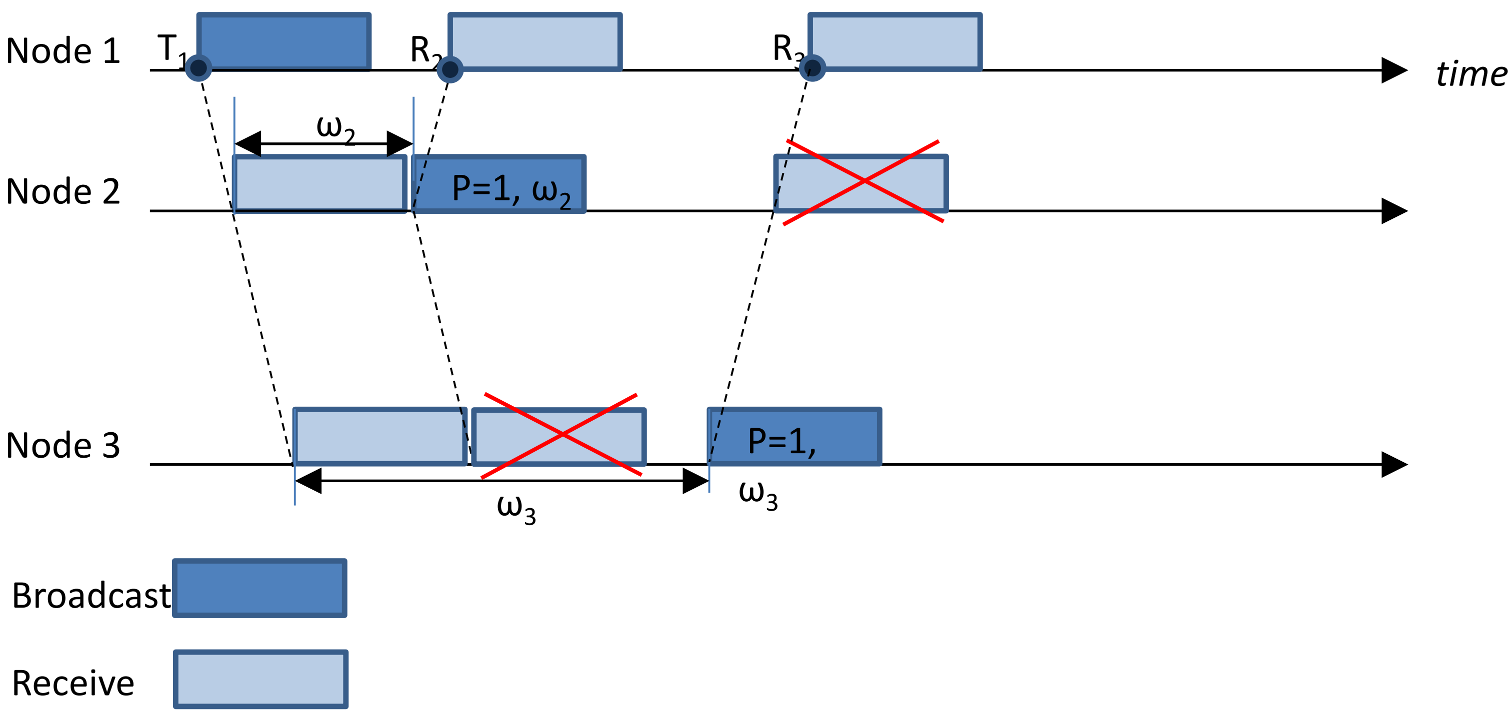
Propagation Delay Measurements in Floods



Propagation Delay Measurements in Floods

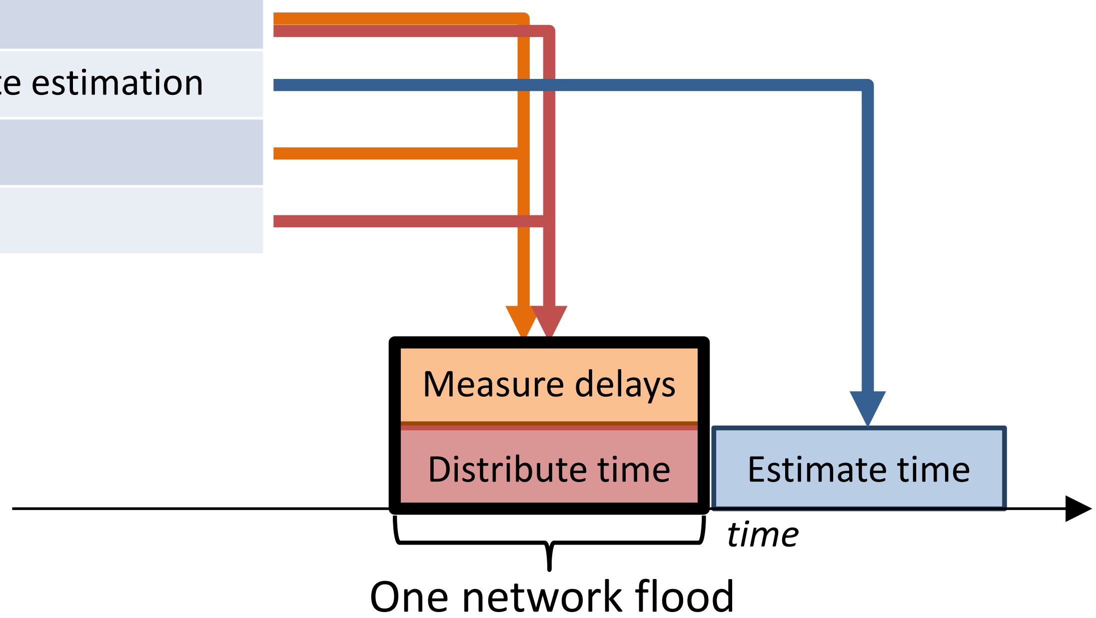


Propagation Delay Measurements in Floods

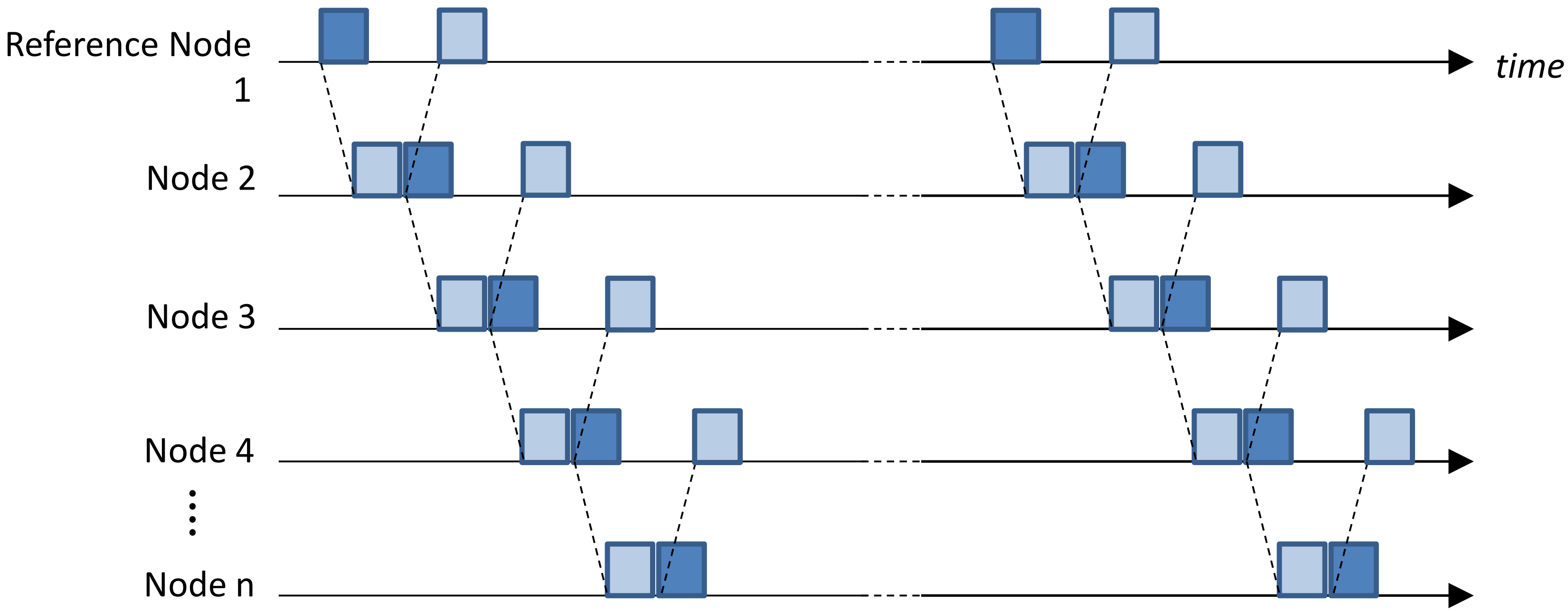


Putting it Together

- MAC-layer timestamping
- Linear regression for offset and clock rate estimation
- Two-way delay measurements
- Fast flooding

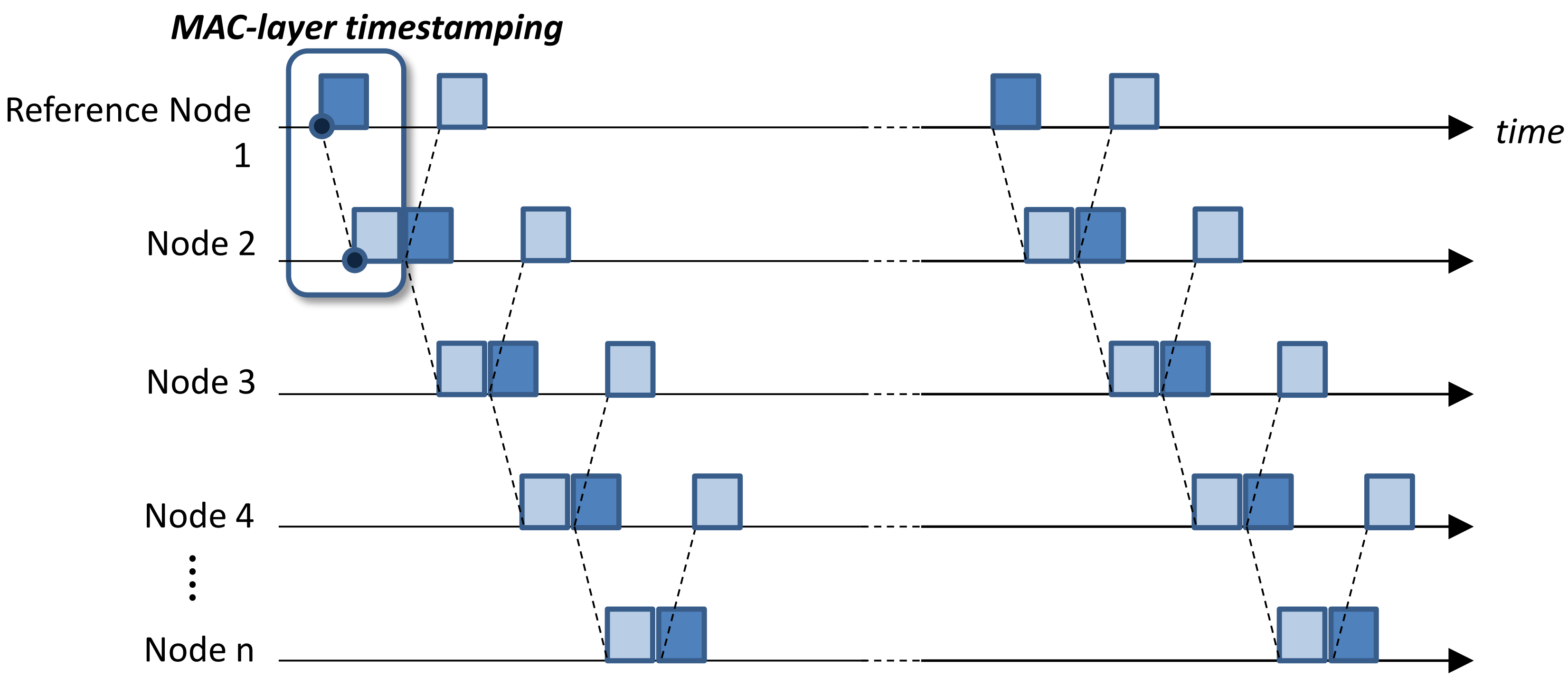


TATS in a Nutshell

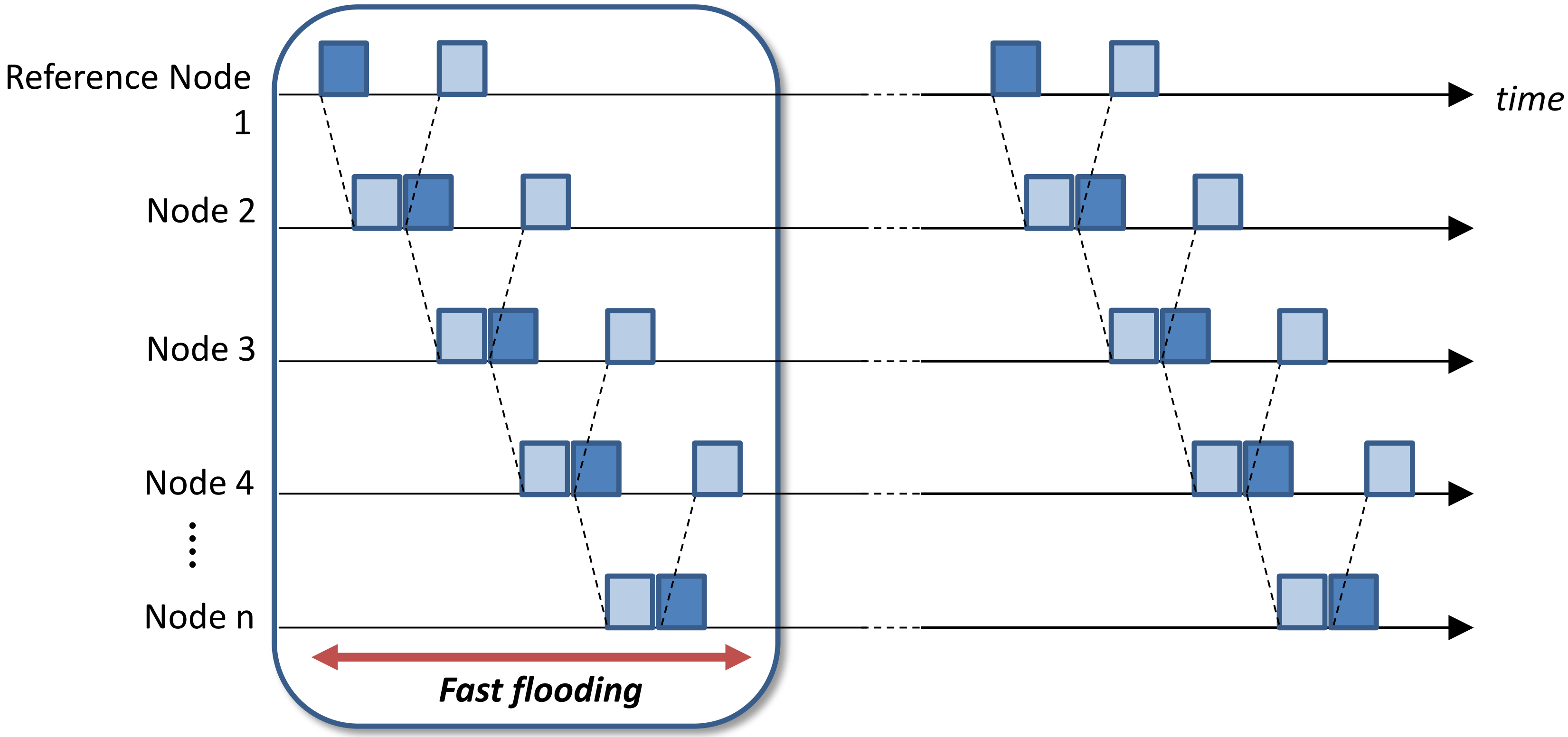


One broadcast packet per round and node, same as in FTSP and PulseSync

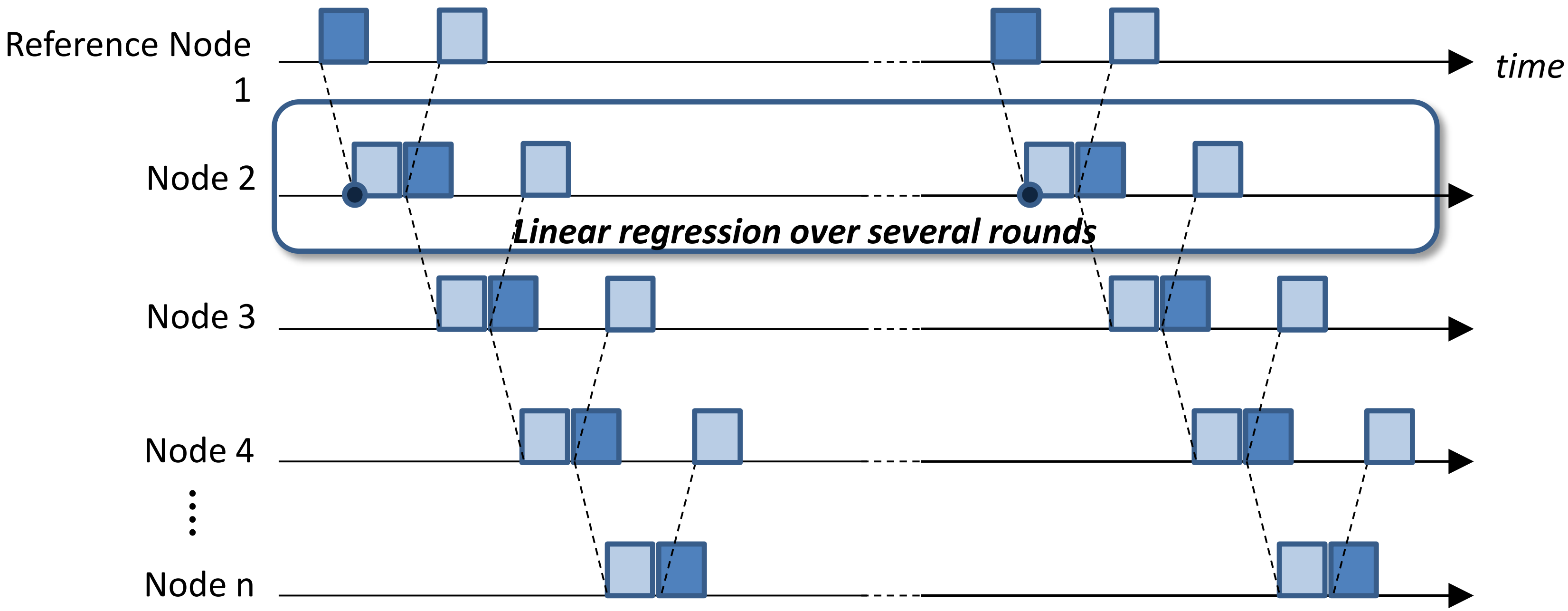
TATS in a Nutshell



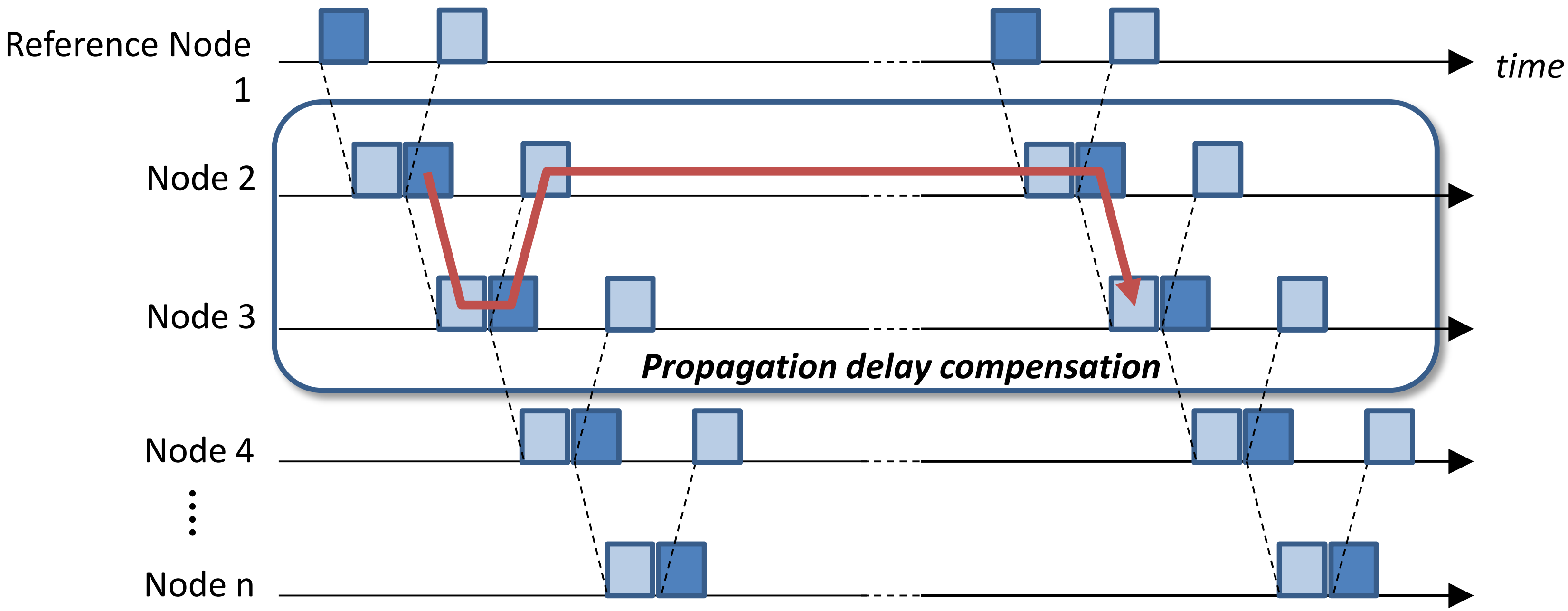
TATS in a Nutshell



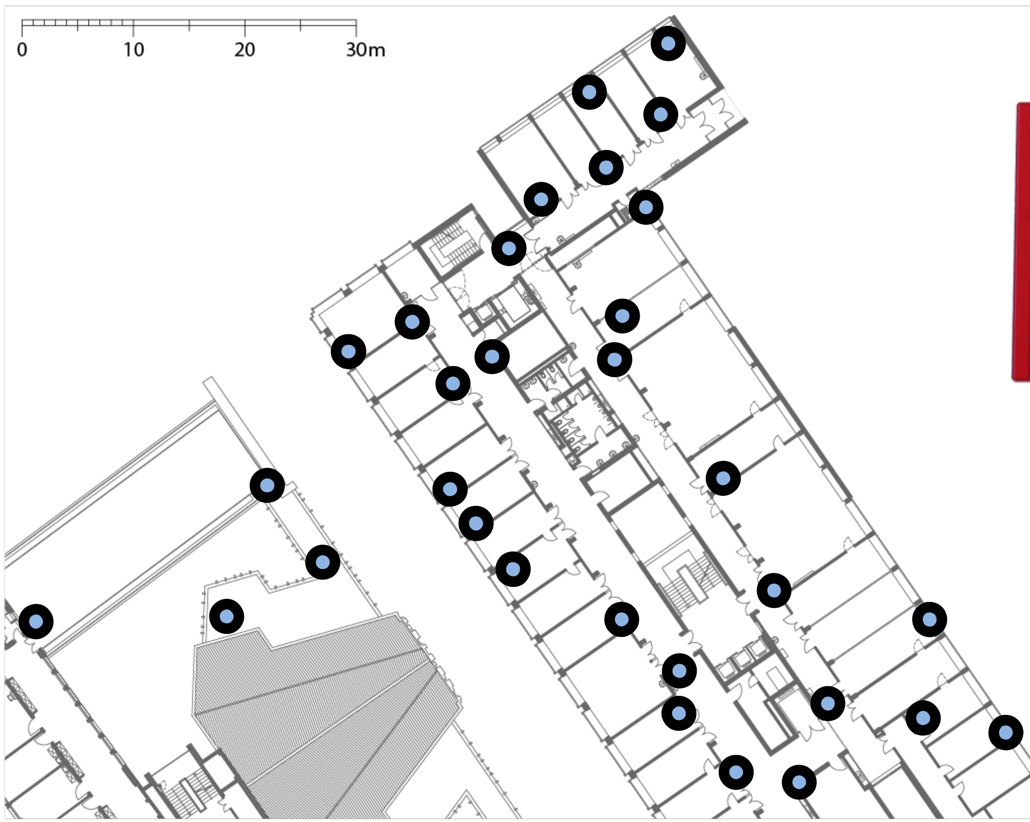
TATS in a Nutshell

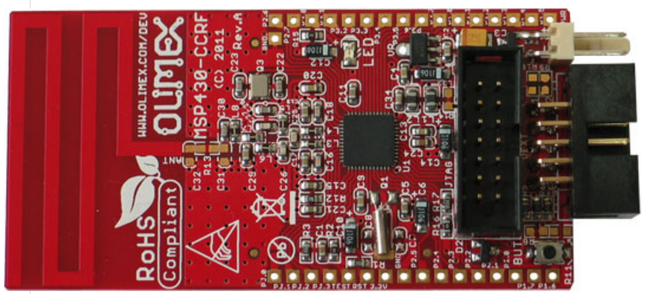


TATS in a Nutshell



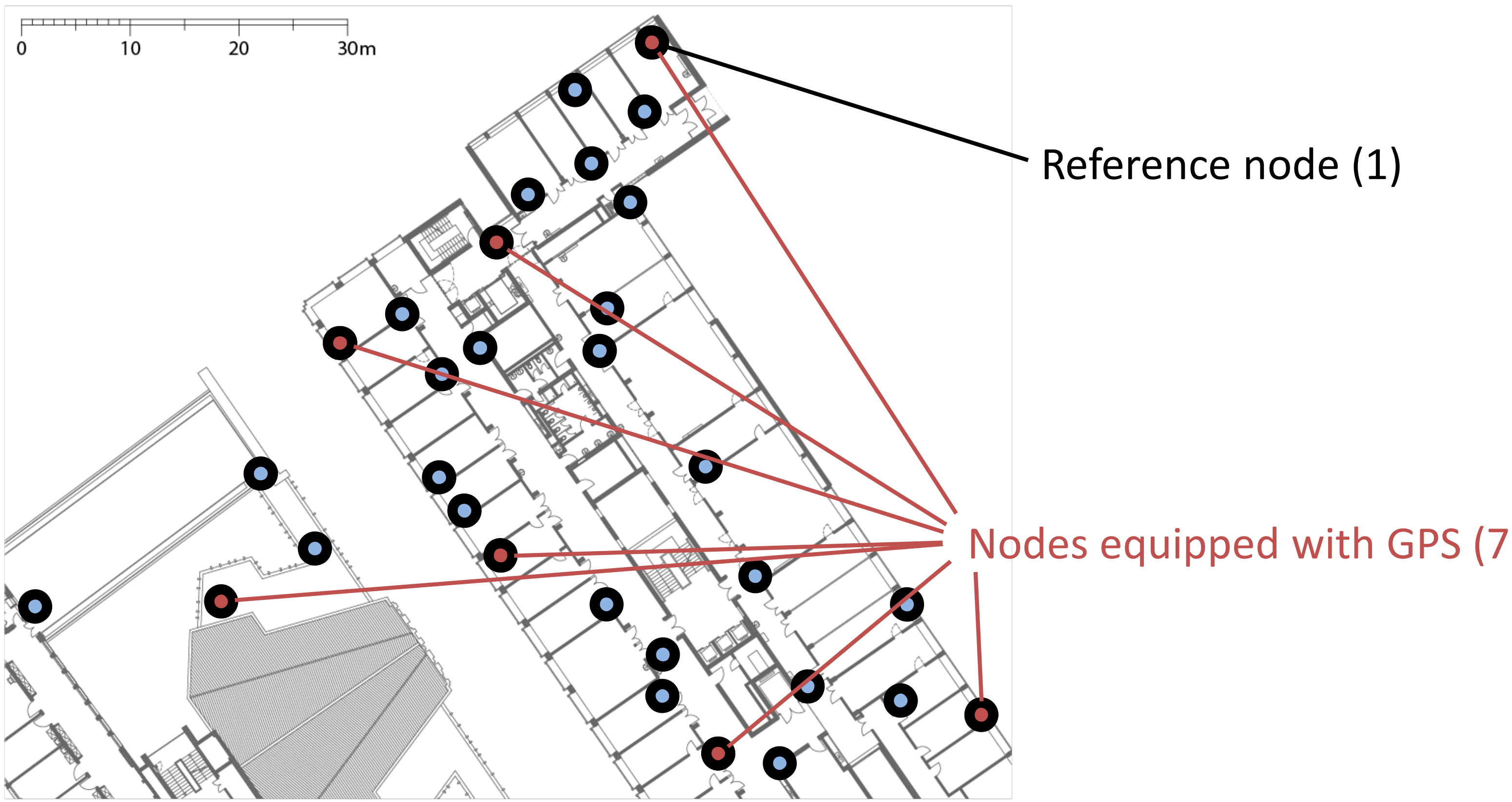
Experimental Evaluation on FlockLab



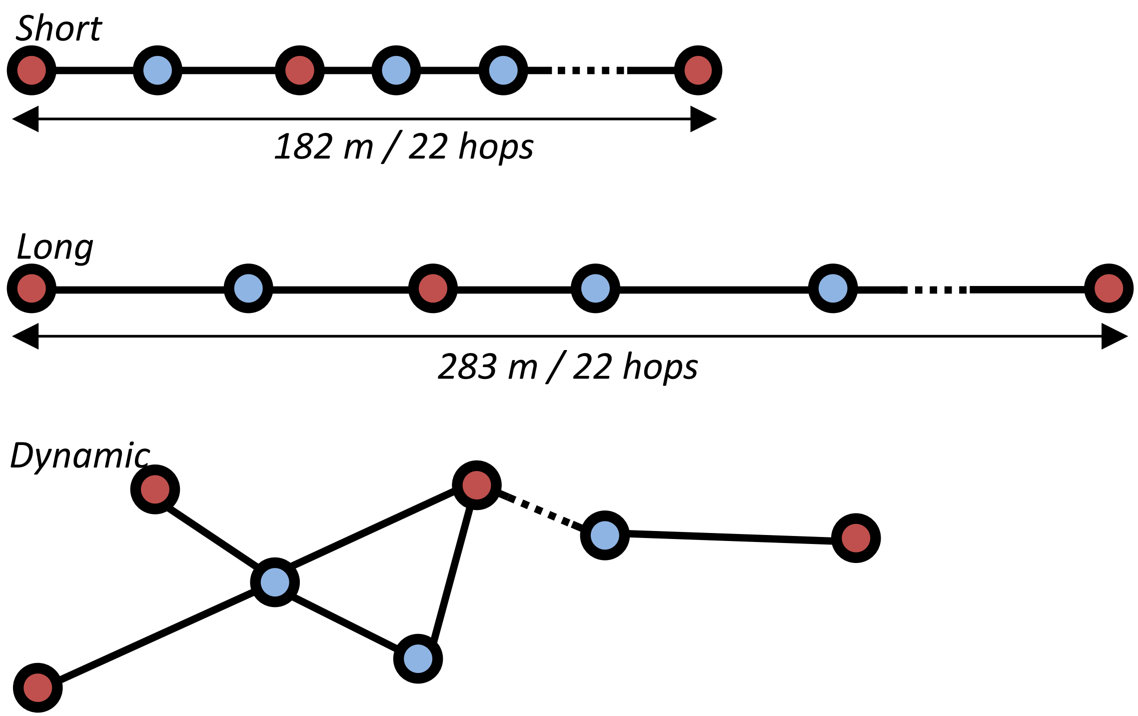


CC430 SoC, MSP430 + sub-1GHz radio
Use radio clock as system clock (13 MHz)

Comparison to PulseSync and Glossy

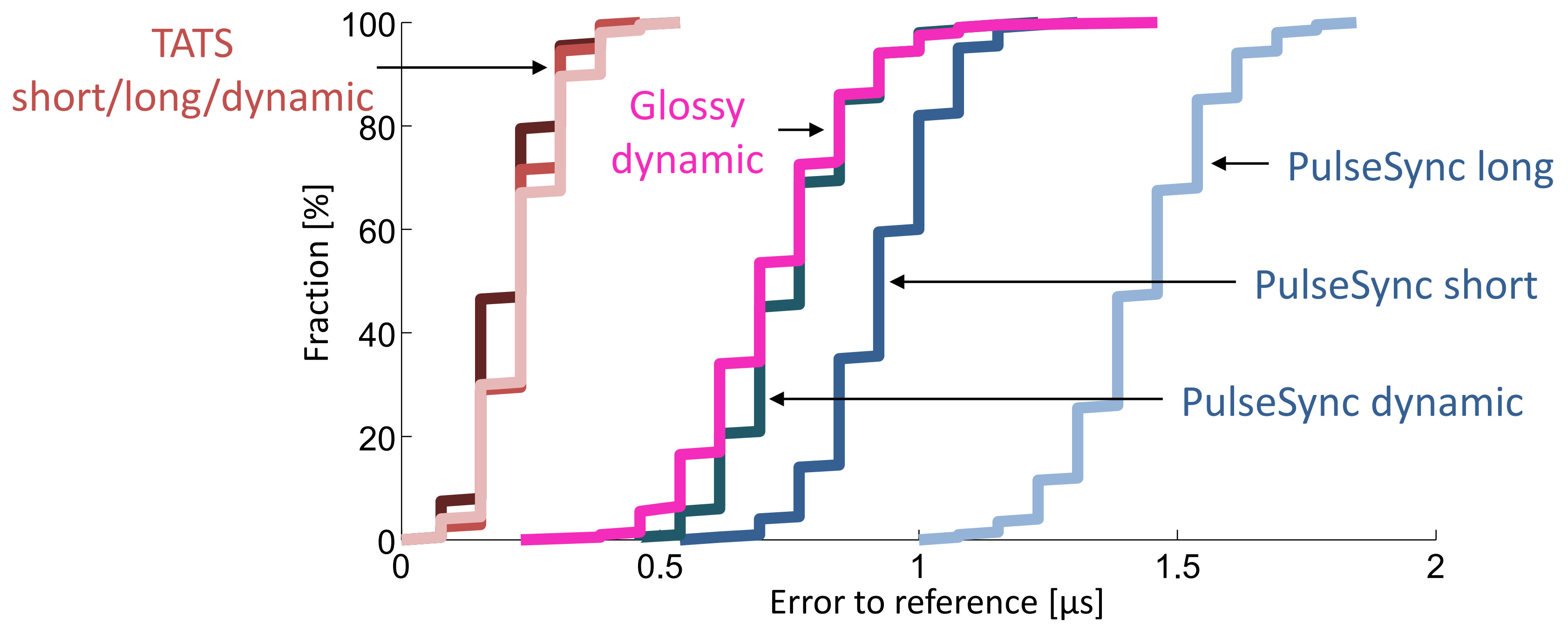


Comparison to PulseSync and Glossy

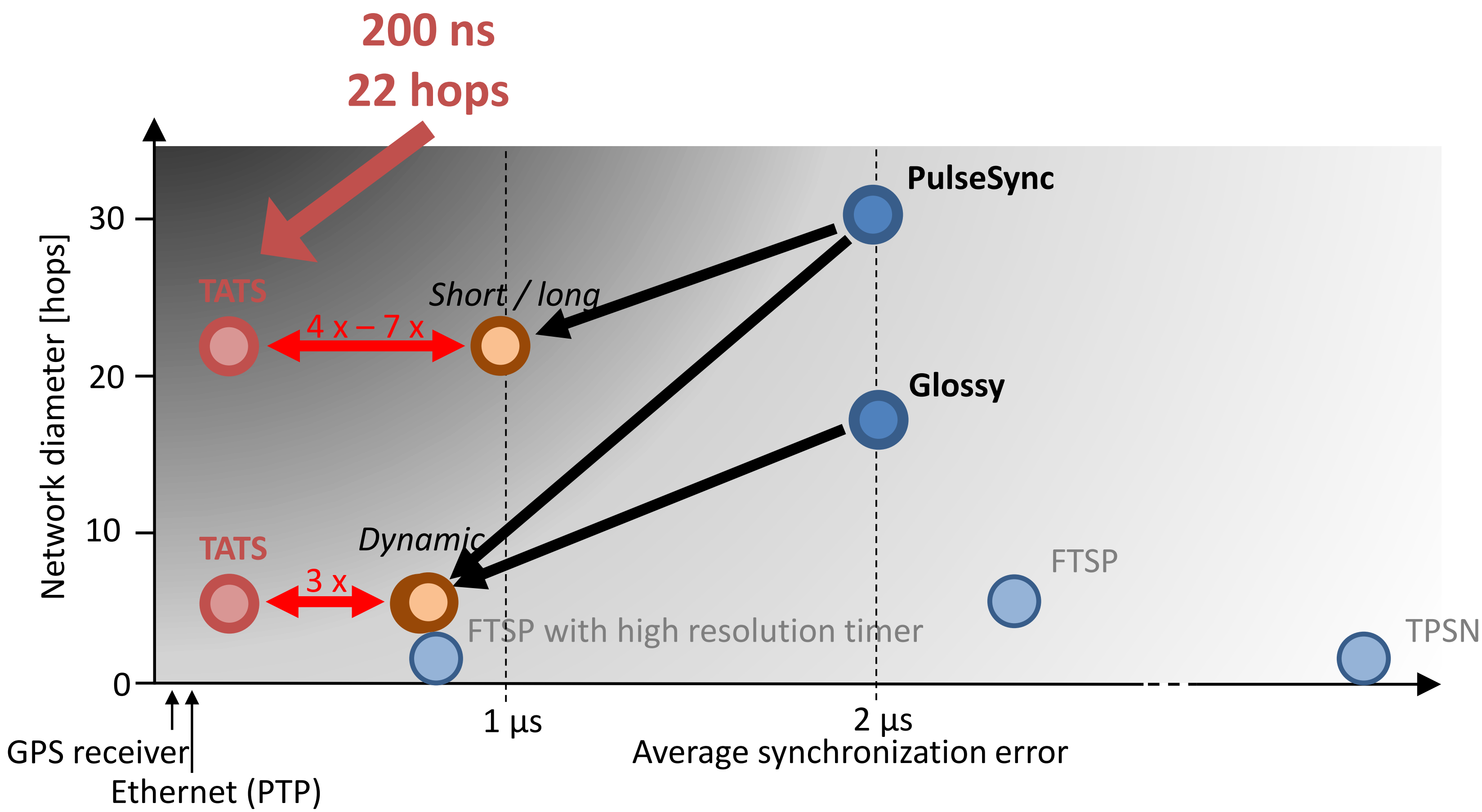


Parameters
- 1 s synchronization interval
- Regression over 80 samples
- Duration: 1 h

Head-to-head Comparison



Head-to-head Comparison



Today's Hot Researcher & Paper

- Deborah Estrin
  - Faculty at Cornell Tech (NY)
    formerly UCLA

- Founding Director of the NSF Center for Embedded Networked Sensing (CENS)
  - Pioneering the development of mobile and wireless systems to collect and analyze real time data about the physical world and the people who occupy it
- Now focusing on mHealth

Embedded, Everywhere. A Research Agenda for Networked Systems of Embedded Computers. *National Research Council,* NATIONAL ACADEMY PRESS, Washington, D.C. 200.