b
UNIVERSITÄT
BERN

b
UNIVERSITÄT
BERN

Internet of Things

V. Time Synchronization

Prof. Dr. Torsten Braun, Institut für Informatik

Bern, 22.03.2021 – 29.03.2021

Internet of Things: Time Synchronization

Table of Contents

1. Introduction

1. Time Synchronization

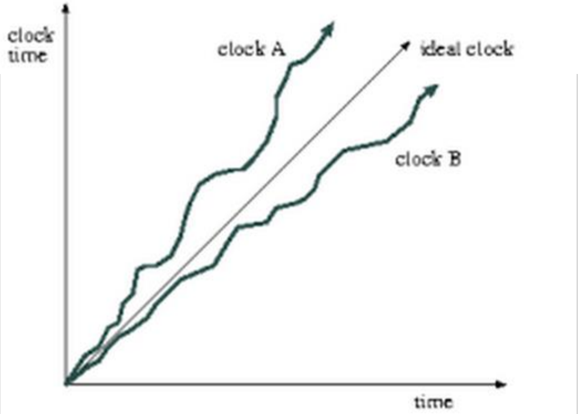
Clocks at sensor nodes should be synchronized in order to

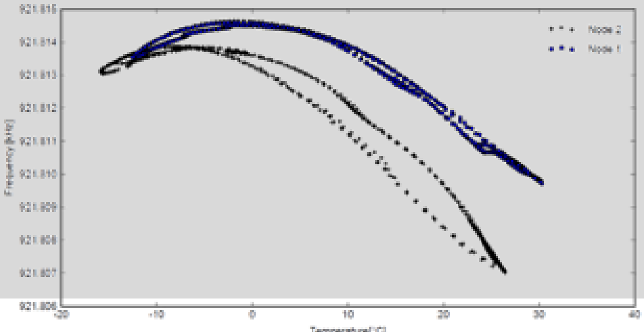
- transform events occurring at different sensor nodes into a common time frame.
- support synchronized sleep and duty cycles among nodes.
- enable localization.

1. Introduction

2.1 Clocks and Communication Delays

- Sensor clocks are based on oscillators.
- C(t): clock at time t
- Derivative dC(t)/dt is ideally 1.
  If not, the clock has a drift.
- ***Clock skew***:
  difference between readings of 2 clocks
- ***Clock drift***: difference in reading between a clock and a nominal perfect reference clock per unit of time of the reference clock





UNIVERSITÄT BERN

1. Introduction

2.2 Clocks and Communication Delays

– Clock drift depends on environmental conditions such as temperature and humidity.

$$1 - \rho \leq \frac{dC(t)}{dt} \leq 1 + \rho$$

– Typical values for ρ: $20 * 10^{-6}$ [s/s] = 20 s in 11.6 days

– If drifts $d_{i,j}$ and initial offsets $o_{i,j}$ of two clocks are known

$C_i(t) = d_i * t + o_i$ $C_j(t) = d_j * t + o_j$	$\Rightarrow$ $(C_i(t) - o_i) * d_j = (C_j(t) - o_j) * d_i$ $C_i(t) = (d_i/d_j) * C_j(t) - o_i * (d_i/d_j) + o_{jj}$

– Clock times can be "translated": for two nodes i, j:

$$C_i(t) = a_{i,j}\, C_j(t) + b_{i,j}$$

– Example:

$C_i(t) = 1.1 * t$

$C_j(t) = 0.9 * t$

$\rightarrow C_j(t) = 0.8181 * C_i(t)$

UNIVERSITÄT BERN

1. Introduction

3. Packet Delay

Send	Access	Transmission	Propa-gation	Reception	Receive

- Send time
  - Time needed by the packet from application to MAC layer
  - Variable because of software delays
- Access time
  - Delay resulting from MAC protocol
- Transmission time
  - Delay caused by bit-by-bit transmission
  - Can be calculated from packet length and radio speed, deterministic
- Propagation time
  - Time taken to traverse the wireless link from sender to receiver
  - Negligible
- Reception time
  - Time required to receive each bit of a packet
  - Deterministic
- Receive time
  - processing time between MAC and application layer
  - Variable

1. Introduction

4. Classes of Synchronization

- Internal vs. external
  - External: Nodes synchronize to an external master.
  - Internal: Nodes synchronize among each other.
- Scope: all nodes or sub-sets
- Rate vs. offset synchronization
- Time-scale transformation vs. clock synchronization
- Lifetime
  - Continuous: Network maintains synchronization at all times.
  - On-demand: No synchronization for a long time to save energy
    - Event-triggered: time-stamping of events and synchronization when event occurs, e.g., post-facto synchronization
    - Time-triggered: synchronization for a specific point of time

1. Introduction

5. Requirements for Time Synchronization Schemes

- Energy efficiency
- Memory usage
- Scalability
  - Deployment of large number of nodes
- Precision
  - Ordering of events vs. microsecond accuracy
- Robustness
  - Schemes should be robust to node failures.
- Lifetime
- Scope
  - Global vs. local
- Cost and size
  - GPS receivers are relatively large and costly.
- Immediacy
  - requires pre-synchronization

UNIVERSITÄT BERN

1. Introduction

6. Global Positioning System

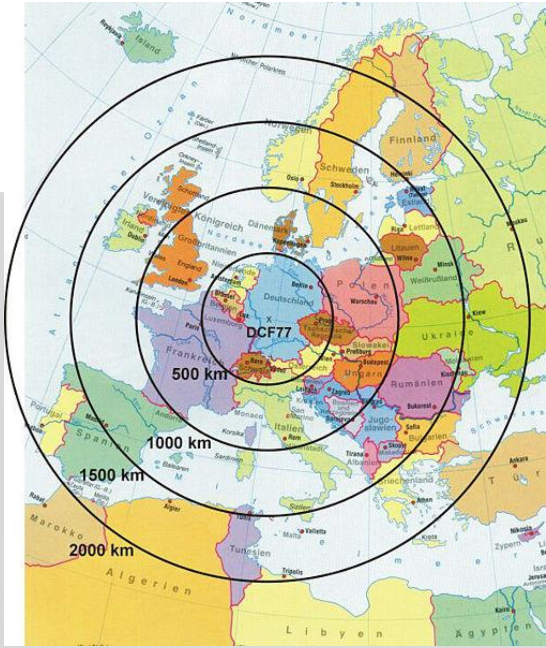
- consists of 32 operational satellites in 6 different planes broadcasting their exact location and precisely synchronized on-board clock time
- requires line-of-sight and does not work inside buildings.
- requires at least 4 visible satellites to calculate x, y, z, and Δt.
- Receiver consumes non-negligible power.

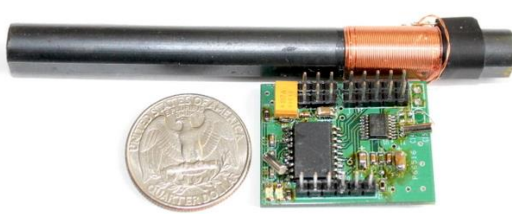
Component	Power (mW)
MCU	18
Radio (TX)	79.2
Radio (RX)	29.7
GPS	165
Switch-mode Regulator	6
Linear Regulator	3.3
Audio	3

1. Introduction

7. Time Signals

– Time signals transmitted by dedicated radio stations
– Example: DCF77
  – D=*Deutschland* (Germany), C=long wave signal, F=Frankfurt, 77=frequency: 77.5 kHz.
  – Frequency synchronous with controlling atomic clock
  – Time information by amplitude modulation, 59 bits per minute
– Prototype receiver consumes 0.266 mW for an average reporting frequency of 1Hz
– Synchronization error ~ 1 ms for 500 s update intervals on TelosB motes





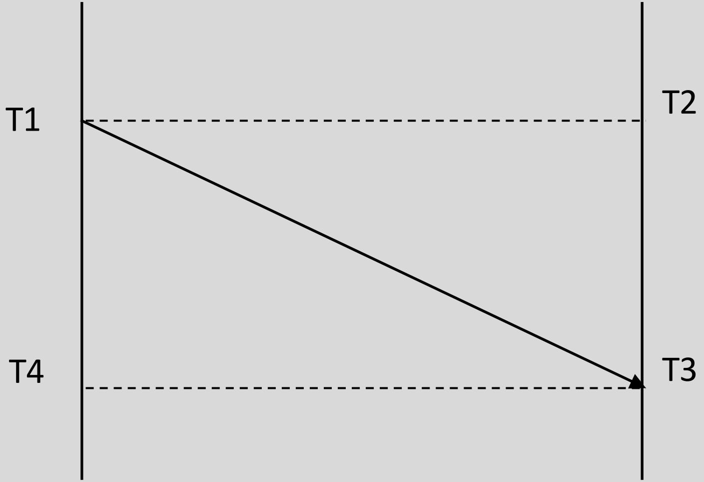
2. Synchronization Techniques

1. Taking 1 sample
   – Unidirectional synchronization
   – Round-trip synchronization
   – Reference broadcasting
2. Combining multiple estimates
3. Synchronization of multiple nodes

2. Synchronization Techniques

1.1 Unidirectional Synchronization

- Time-stamping of message
- Receiving node
  - knows T1 and T3
  - needs to estimate T2 or T4
  - Example:
    - estimated delay d
    - T2 ≈ T3 – d
- Example: DCF77

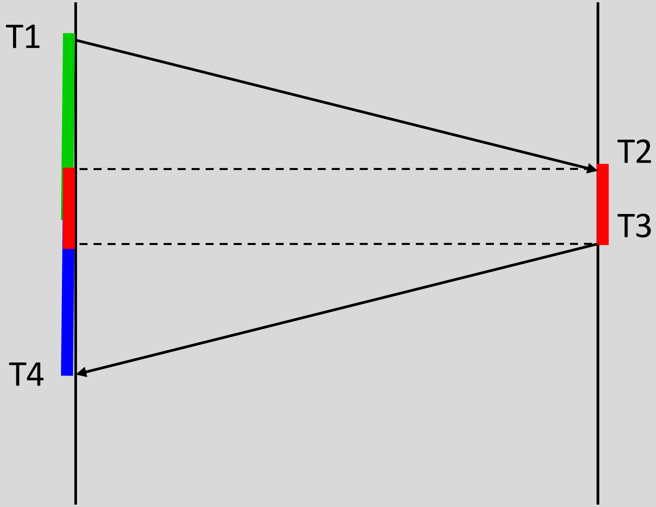


UNIVERSITÄT BERN

2. Synchronization Techniques

1.2 Round-Trip Synchronization

- Δ: skew (offset)
- d: communication delay
- T2 – T1 = d + Δ
- T4 – T3 = d – Δ
- Δ = ((T2 – T1) – (T4 – T3)) / 2
- d = ((T2 – T1) + (T4 – T3)) / 2
- Example
  - Δ = 10 s, d = 0.1 s
  - T1 = 0 s, T2 = 10.1 s
  - T3 = 11.1 s, T4 = 1.2 s
  - Δ = (10.1 s – (-9.9 s)) / 2 = 10 s
  - d = (10.1 s + (-9.9 s) / 2 = 0.1 s

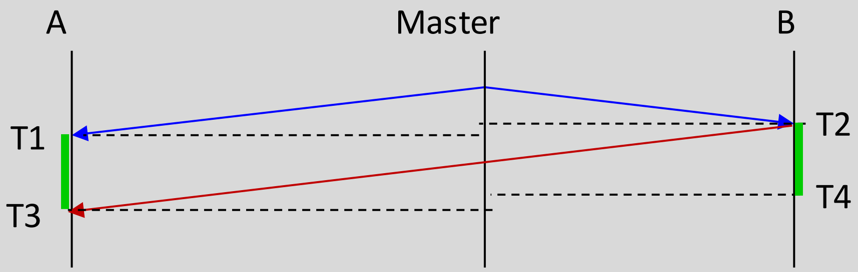


Example: NTP (Network Time Protocol)

2. Synchronization Techniques

1.3 Reference Broadcasting

- Broadcast of master received at T1 by A and at T2 by B with almost equal delay.
- B sends T2 to node A.
- A receives message from B with delay D = T3-T1 at T3 and estimates T3 = T1 + D ≈ T2 + D = T4
- Smaller synchronization error because of nearly simultaneous broadcast message reception
- Master reaches A and B but not vice versa.



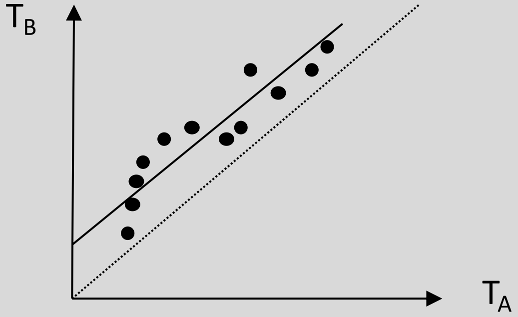
2. Synchronization Techniques

2. Combining Multiple Estimates

- Synchronization can be performed several times.
- Each point denotes a single estimate of the time relation between two nodes A and B.
- Multiple samples and interpolation techniques can decrease estimation error.
  - Example: linear regression: $T_B = a \cdot T_A + b$
    - Problem: requires large amount of memory, and much processing!
    - SS: sum of squares
- Examples
  - Tiny-Sync: Round-trip synchronization using multiple estimates
  - Reference Broadcast Synchronization: reference broadcasting using multiple estimates

$$b = \frac{SS_{xy}}{SS_{xx}} = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

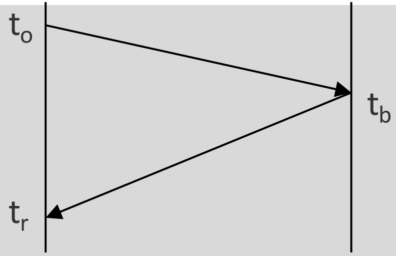
$$a = \bar{y} - b\bar{x} = \frac{\sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} x_i y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$$

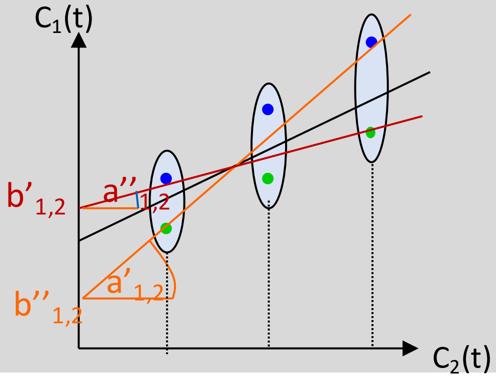


2. Synchronization Techniques

2.1 Tiny-Sync

- $C_1(t) = a_{1,2}\, C_2(t) + b_{1,2}$, (*)
- $t_o < a_{1,2}\, t_b + b_{1,2}$
- $t_r > a_{1,2}\, t_b + b_{1,2}$
- (t_o, t_b, t_r) = data point
- Several data points are collected:
- Line corresponding to (*) must lie between the vertical intervals of each data point.
- Two lines indicate upper and lower bound.
- $a''_{1,2} \leq a_{1,2} \leq a'_{1,2}$
- $b''_{1,2} \leq b_{1,2} \leq b'_{1,2}$
- The more data points the tighter are the bounds.
- Problem
  - Estimation is computationally expensive and requires significant memory.
- Solution
  - Keep only data points contributing to the bounds
  - Problem: Optimal solution might not be found.





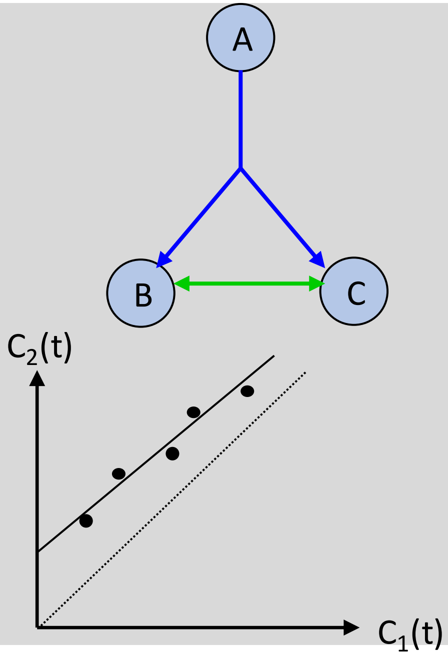
2. Synchronization Techniques

2.2 Reference Broadcast Synchronization

- Third party A transmits beacon message (without timestamp!) to neighbors.
- A may have unlimited power resources and reaches B, C, but not vice-versa!
- Neighbors timestamp the packet on reception and exchange timestamps.
- Neighbors can estimate the clock skew and clock drift using m beacons.
  - Each receiver i can calculate its offset to its neighbor j.

$$Offset_{i,j} = \frac{1}{m}\sum_{k=1}^{m}(T_{i,k} - T_{j,k})$$

  - Averaging for offset estimation
  - Least square linear regression for offset/drift estimation
- Case Study
  - Mutual synchronization of a 30 nodes network in the range of 5 µs



UNIVERSITÄT BERN

2. Synchronization Techniques

3. Synchronization of Multiple Nodes

– Typically, more than two nodes need to be synchronized.

– Not all nodes to be synchronized can directly communicate to each other.

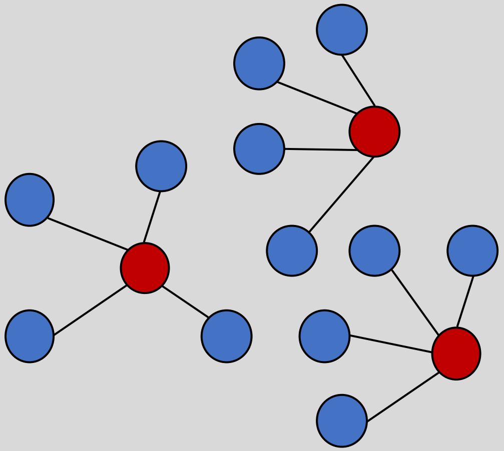
→ Multi-hop synchronization approaches

- Out-of-band synchronization
- Structured approaches
  - Clustering
  - Tree Construction
- Unstructured approaches
  - Diffusion-based Synchronization
  - Gradient Clock Synchronization

2. Synchronization Techniques

3.1 Out-of-Band Synchronization

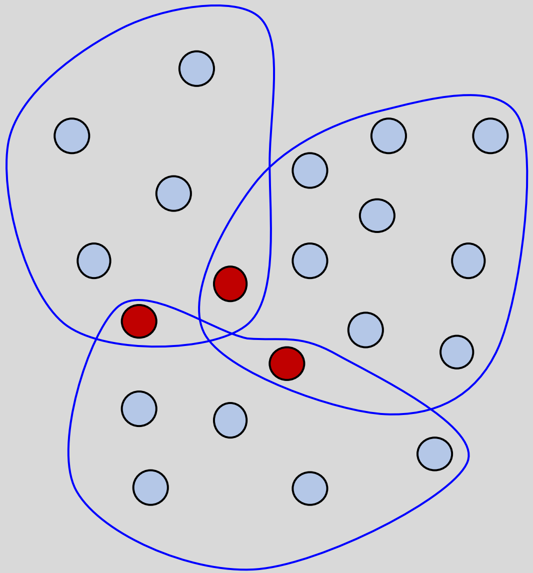
- Each node is connected to at least 1 master
- Masters are synchronized by out-of-band mechanisms, e.g., GPS.



2. Synchronization Techniques

3.2.1 Clustering

- All members of a cluster can synchronize, e.g., by reference broadcasting
- Time gateways belonging to different clusters can translate time-stamps between clusters.
- Tradeoff for cluster size
  - Many translations for small clusters
  - Higher energy consumption for large clusters
- Examples with RBS
  - Multi-Hop Synchronization
  - Time Routing in Multi-Hop Networks



2. Synchronization Techniques

3.2.1.1 Multi-Hop Synchronization with RBS

- Nodes A and B send synchronization beacons (pulses) P_A and P_B.
- Two events E_1 and E_7 are close to receivers 1 and 7.
- Receiver 1 observes E_1 2s after P_A.
- Receiver 7 observes E_7 4s prior to P_B.
- Receiver 4 observes P_A 10s after P_B.

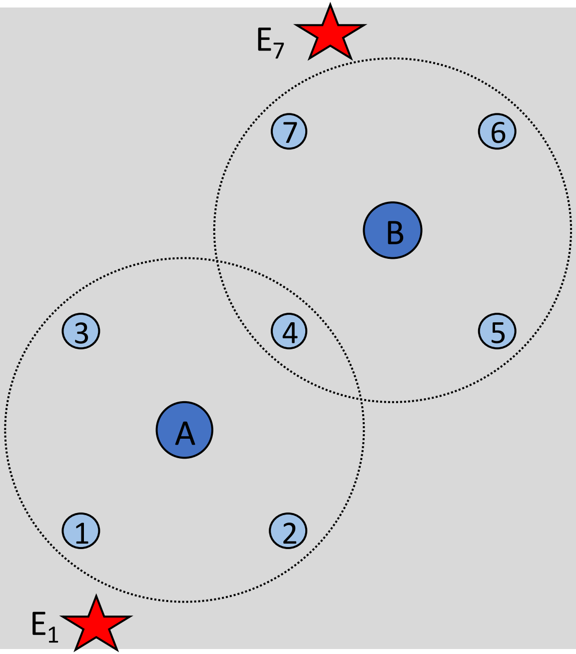
$$E_1 = P_A + 2$$

$$E_7 = P_B - 4;\ E_7 + 4 = P_B$$

$$P_A = P_B + 10$$

$$\rightarrow E_1 = P_A + 2 = P_B + 10 + 2$$

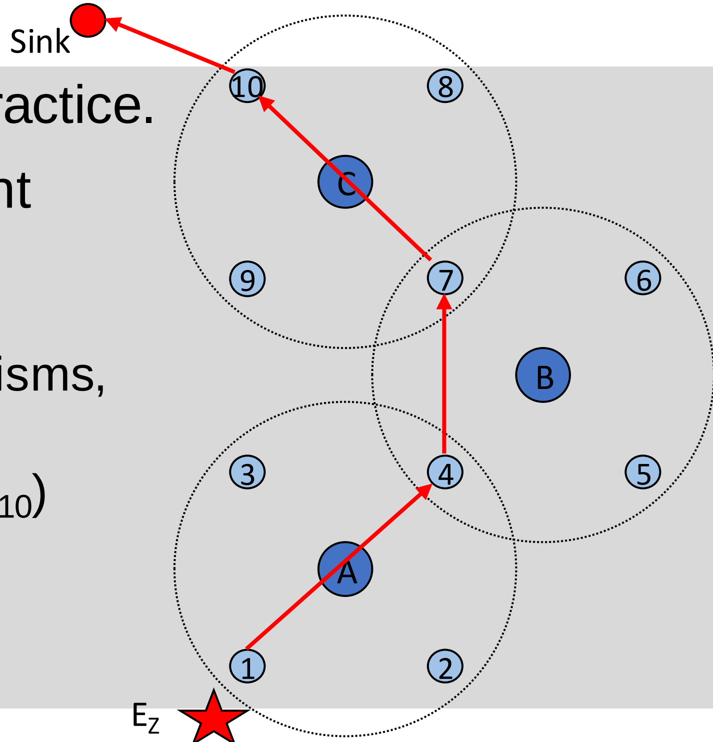
$$= E_7 + 4 + 10 + 2 = E_7 + 16$$



2. Synchronization Techniques

3.2.1.2 Time Routing in Multi-Hop Networks with RBS

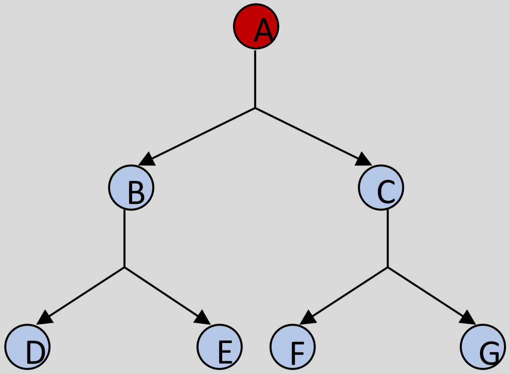
- Single time gateways are not desirable in practice.
- Approach: Dynamic time route establishment
  - Each node can convert time relationships (drift and offset).
  - Time route can be set up by routing mechanisms, e.g., shortest path routing
  - Example: $E_Z(R_1) \rightarrow E_Z(R_4) \rightarrow E_Z(R_7) \rightarrow E_Z(R_{10})$



2. Synchronization Techniques

3.2.2 Tree Construction

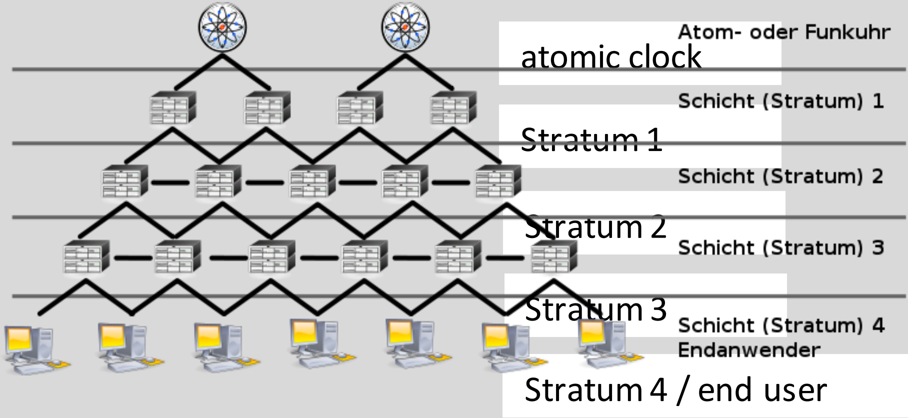
- Synchronization tree with master as the root
- Single-hop synchronization along tree
- Accuracy degrades with distance from root.
- Low tree level desired, but puts burden to nodes in order to synchronize other nodes.
- Tree formation is difficult in dynamic environments.
- Example: Timing-Sync Protocol for Sensor Networks



2. Synchronization Techniques

3.2.2.1.1 Timing-Sync Protocol for Sensor Networks

- TPSN inspired by Network Time Protocol
- NTP
  - Global time is injected to the network by time servers (Stratum 1).
  - Stratum 1 servers are synchronized in an out of band manner by atomic clocks (Stratum 0).
  - Nodes form a hierarchy. Stratum 1 server list publicly available at http://support.ntp.org.



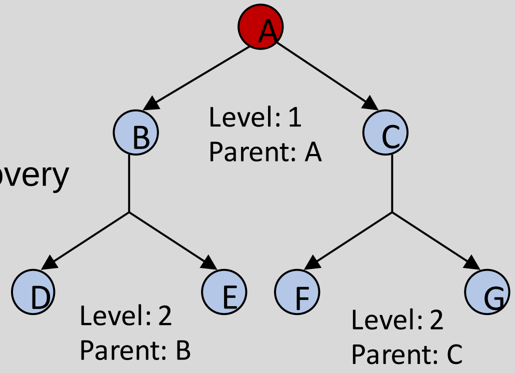
u^b UNIVERSITÄT BERN

2. Synchronization Techniques

3.2.2.1.2 Timing-Sync Protocol for Sensor Networks

Protocol Steps

- Level Discovery
  - Root node on level 0 broadcasts Level_Discovery packet.
  - Neighbors assign level 1 and rebroadcast
  - …
- Synchronization
  - Pair wise synchronization along hierarchy established during level discovery
  - Root node sends Time_sync packet.
  - Nodes back off randomly to avoid collisions and start message exchange (round trip synchronization).
  - Level n+1 nodes adapt clock to level n nodes.
- Implementation on MICA motes results in synchronization errors < 20 µs.

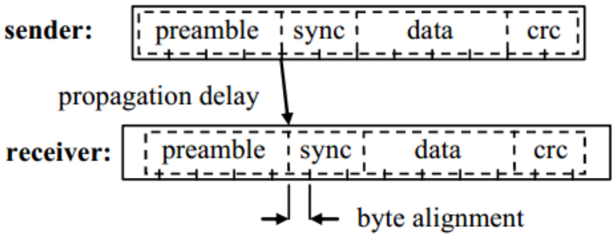


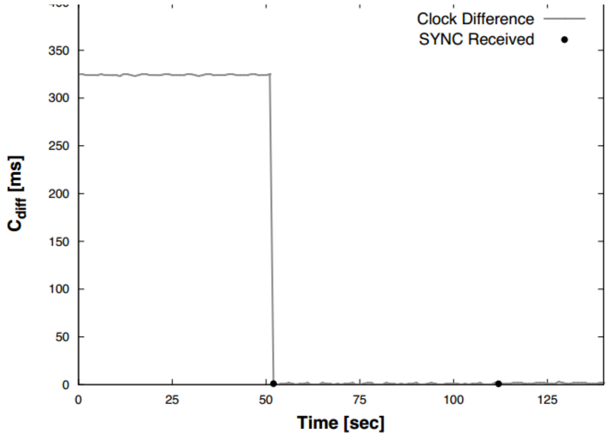
b UNIVERSITÄT BERN

2. Synchronization Techniques

3.2.2.2 Flooding Time-Synchronization Protocol

- Node with lowest ID is used to synchronize the network as a leader.
- A leader periodically floods the network with a synchronization message including the leader's current time (time stamp, recorded on MAC level).
- Nodes record time stamp of message and time of arrival.
- Nodes rebroadcast after updating the time stamp.
- Linear regression (typically with 8 data points) for estimating offset and rate difference to leader.
- Experiments with motes: < 2 µs synchronization error per node pair, synchronization messages every 30 s.





2. Synchronization Techniques

3.3 Unstructured Approaches

- Unstructured approaches do not establish a certain structure, but exchange time information between nodes that can communicate with each other.
- Completely localized solution
- Often: time information exchange using piggy-backing
  $\longrightarrow$ very low overhead

Examples:

- Diffusion-based Synchronization
- Gradient Clock Synchronization

2. Synchronization Techniques

3.3.1 Diffusion-based Synchronization

Rate-based Synchronous Diffusion

– achieves global synchronization, convergence takes longer in larger networks
– Diffusion rate r_{ij}, $\sum_{j \neq i} r_{ij} \leq 1$, r_{ij} random

```
for each sensor node nᵢ in the network {
   exchange clock time with nᵢ's neighbors
   for each neighbor nⱼ {
         cᵢ := time of nᵢ; cⱼ := time of nⱼ
         cⱼ := cⱼ + rᵢⱼ (cᵢ - cⱼ);}
   cᵢ := cᵢ - Σall nⱼ rᵢⱼ · (cᵢ - cⱼ);}
```

Asynchronous Diffusion

– achieves global synchronization; convergence takes longer in larger networks

```
for each node nᵢ with uniform
   probability {
   ask clock readings from nᵢ's
   neighbors;
   average clock readings;
   send back new value to
   neighbors;
}
```

2. Synchronization Techniques

3.3.2 Gradient Clock Synchronization

- Definitions
  - Hardware clock H_i with rate h_i and offset Φ
  - Logical clock L_i with rate l_i and offset θ
- Goal: precise time synchronization among neighbors
- No adaptation of hardware clock, but of logical clock (rate and offset), e.g., implemented by software timer
- No adaptation to reference clock, but completely distributed algorithm
- Nodes periodically broadcast L_i , l_i
- Evaluation on motes: 4 µs synchronization error between neighbors with 30 s synchronization interval

$$H_i(t) = \int_{t_0}^{t} h_i(\tau)d\tau + \phi_i(t_0)$$

$$1-\rho \le h(t) \le 1+\rho$$

$$L_i(t) = \int_{t_0}^{t} h_i(\tau)\cdot l_i(\tau)d\tau + \theta_i(t_0)$$

$$l_i(t_{k+1}) = \frac{\left(\sum_{j\in N_i} l_j(t_k)\right) + l_i(t_k)}{|N_i|+1}$$

$$\theta(t_{k+1}) = \theta(t_k) + \frac{\left(\sum_{j\in N_i} L_j(t_k)\right) + L_i(t_k)}{|N_i|+1}$$

Thanks

for Your Attention

Prof. Dr. Torsten Braun, Institut für Informatik

Bern, 22.03.2021 – 29.03.2021