The Pulse Coupled Phasor Measurement Units and the PulseSS Protocol

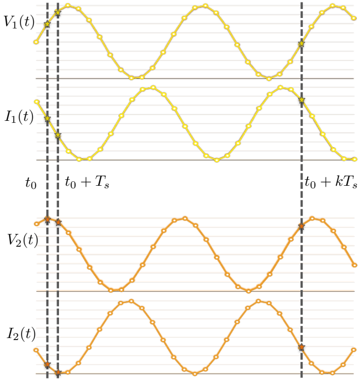
Reinhard Gentz, Lorenzo Ferrari, Anna Scaglione, Masood Parvania

Department of Electrical, Energy and Computer Engineering,
Arizona State University, Tempe, AZ, USA

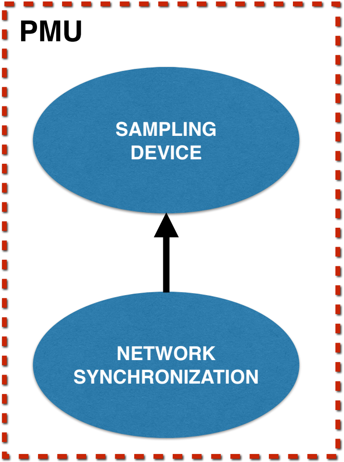
TCIPG 04/03/2015

Premise

- PMU: why we love them?
  - State information
  - Direction of power flow
- Concept of phasor invented by Steinmetz (1893)
- Vast literature (F. Chen et al "State estimation model and algorithm including PMU")
- C37.118 IEEE standard puts performance requirements (2005)



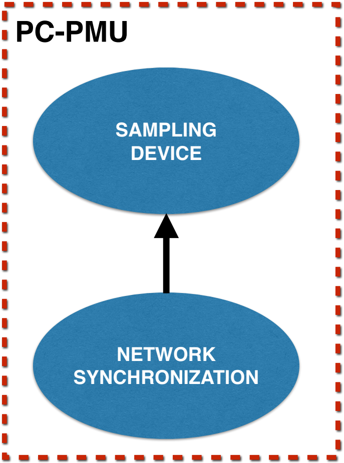
Standard PMU



$\equiv$ **GPS**

GPS + PTP

Idea we propose: Pulse Coupled PMU



Pulse Coupled Oscillator (PCO)

≡ ~~GPS~~ ~~GPS~~ + PTP

Main benefits

- Very scalable cross-layer communication scheme
- Higher security compared to the GPS (spoofing)
- Possible integration of sensor and radio on a single chip
  - PCO protocol comes with PHY-layer and can work over powerline

Question

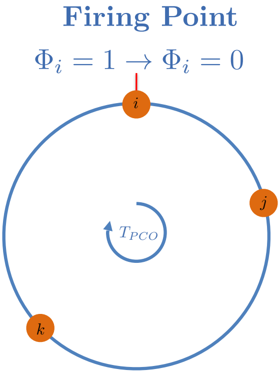
Can the PC-PMU's attain C37.118 IEEE requirements?

Overview

The Pulse Coupled Oscillator Protocol (PCO)

State of the PCO Clock

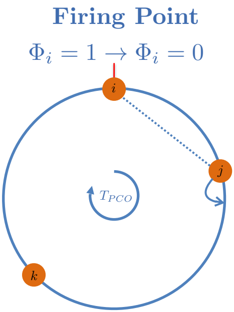
$$\Phi_i(t) = \left(\frac{t}{T_{PCO}} - \phi_i\right) \pmod 1$$



Node fires when $\Phi_i(t) = 1$

The Pulse Coupled Oscillator Protocol (PCO)

When node j hears node i firing at $t = t_i$:



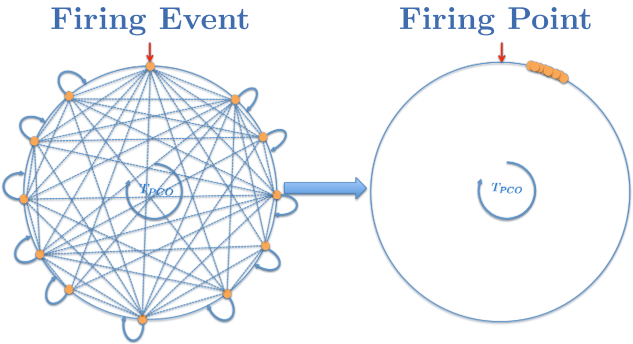
PCO Phase Update

$$\Phi_j(t_i^+) = \begin{cases} \min\{(1+\alpha)\Phi_j(t_i), 1\} & \text{if } \rho < \Phi_j(t_i) < 1 \text{ (refractory period)} \\ \Phi_j(t_i) & \text{else} \end{cases}$$

Previous work on PCO

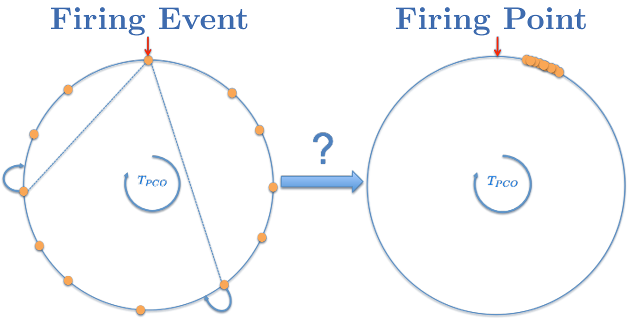
- "Mathematical aspects of heart physiology" (Peskin, 1975)
- Proposed as network synchronization protocol by:
  Frigui, Torikai, Nakano, Saito, Hong, Barbarossa, Celano, ...
- Performance studies on convergence and accuracy:
  - Mirollo, Strogatz (1990)
  - Lucarelli, Wang (2004)
  - Werner-Allen, Tewari, Patel, Welsh, Nagpal (2005)
- Most relevant:
  - Tyrrell, Auer, Bettstetter (2008)
  - Pagliari, Scaglione, Hong (2009)

The Pulse Coupled Oscillator Protocol (PCO)



- "Synchronization of pulse-coupled biological oscillators", R.Mirollo e S.Strogatz (1990)

The Pulse Coupled Oscillator Protocol (PCO)



- "Decentralized synchronization protocols with nearest neighbor communication", D. Lucarelli and I.-J. Wang (2004)
- Distribution grids are primarily radial: we studied PCO convergence on tree networks

Questions

- Can we still reach convergence?
- What if we have propagation delays?
- We will use the results to analyze the Mean Squared Error (MSE) of the PC-PMU

Timing errors in the PCO Protocol

We can modify the update equation taking into account:

- the propagation delay $\tau_{(i,j)}$
- the error $e_j(t_i)$ due to the noisy channel

PCO Phase Update with transmission delays and noise

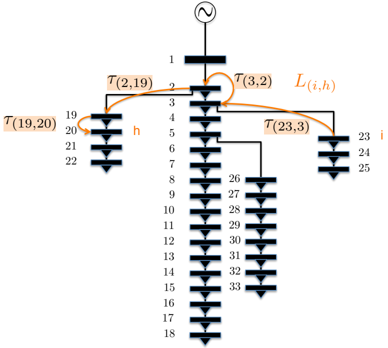
$$\Phi_j(t_i^+ + \tau_{(i,j)} + e_j(t_i)) = \begin{cases} \min\left\{(1+\alpha)(\Phi_j(t_i) + \tau_{(i,j)} + e_j(t_i)), 1\right\} \\ \text{ if } \rho < \Phi_j(t_i) + \tau_{(i,j)} + e_j(t_i) < 1 \\ \Phi_j(t_i + \tau_{(i,j)} + e_j(t_i)) \\ \text{ if } 0 < \Phi_j(t_i) + \tau_{(i,j)} + e_j(t_i) \leq \rho. \end{cases}$$

Theoretical Performance Analysis

- $\Delta\Phi_{(i,j)} = \Phi_j - \Phi_i$
- $L_{(i,j)}$ is the path that connects node i and j

Assumption:

- The refractory period $\rho > 2 \max \tau_{(i,j)}$



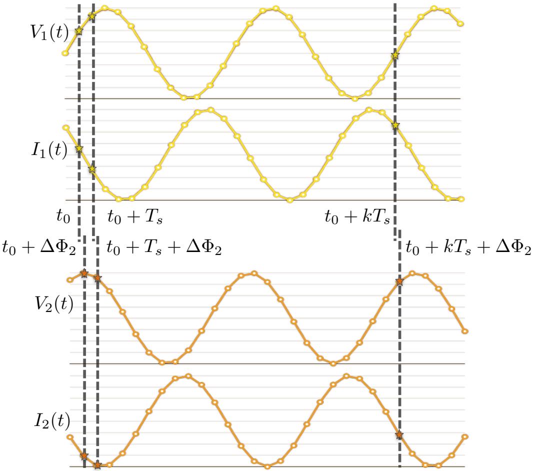
Lemma

The protocol converges and the fixed points are such that there is a node firing first (the head node) and the others are separated in time by:

$$\Delta\Phi_{(i,h)}(t) \underset{t\to\infty}{=} \sum_{(k,m)\in L_{(i,h)}} \tau_{(k,m)}$$

Pulse Coupled Phasor Measurement Error Model

$$\widehat{V}_i(t_0) \approx |V_i(t_0)| e^{j(\theta_{V_i}(t_0) + \omega_0 \Delta\Phi_i(t_0))}$$



Pulse Coupled Phasor Measurement Error Model

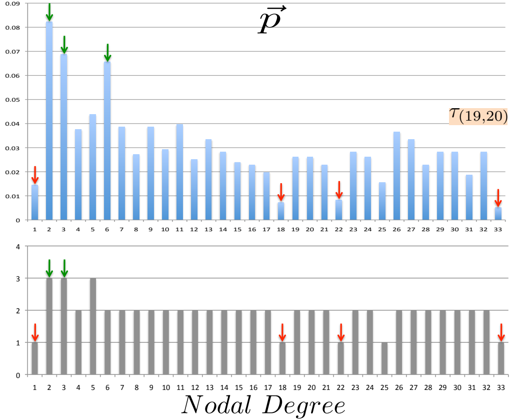
$$MSE = \mathbb{E}\{||\mathbf{V} - \hat{\mathbf{V}}||^2\}$$

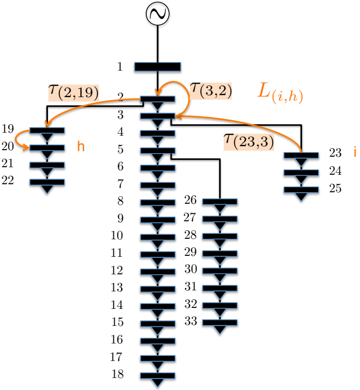
$$MSE \approx \omega_0^2 \mathbf{V}^H \mathbb{E}\{\boldsymbol{\Delta}^2(\Phi)\}\mathbf{V} = \omega_0^2 \sum_{i=1}^{N} \mathbb{E}\{\Delta\Phi_{(i,h)}^2\}|V_i(t_0)|^2$$

$$\mathbb{E}\{\Delta\Phi_i^2\} = \sum_{j=1}^{N} p_j \left(\sum_{(k,m)\in L_{(i,j)}} \tau_{(k,m)} \right)^2$$

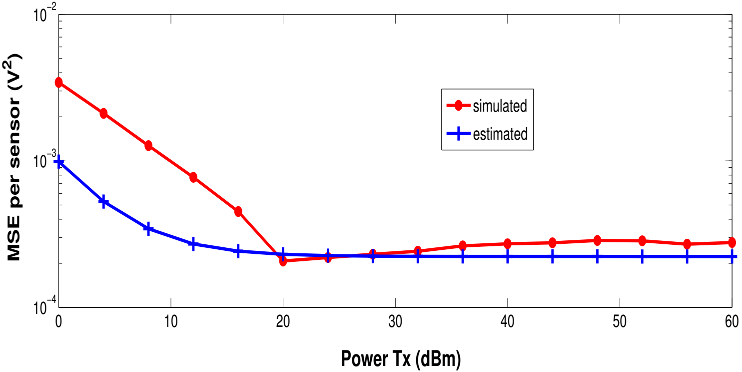
where $p_j = P\{\text{Node } j \text{ is the } \textit{head} \text{ node}\}$

Numerical estimation of *head* probability





Numerical Comparison of MSE and Theory



Simulation: *IEEE 33 Bus, 1 PC-PMU per branch. Power-line communication (band around 300kHz +/- 100kHz). Losses on the line 40dB/km, average distance 100m,Coupling factor α=0.04, 170 iterations, Noise Level= -103 dBm.*

PROs and CONs of the PC-PMU

- PROs
  - Fault tolerance
  - Cross-layer $\rightarrow$ complete integration
  - Potentially higher security
- CONs
  - Lack of transmission protocol
  - Propagation delays sensitivity

Fixing the Con's

Missing for large scale usage

- Data transfers (scheduling)
- Time of flight correction, so they do not add up

We propose decentralized scheduling algorithm as Extension of PCO we call Pulse Synchronizatoin and Scheduling Protocol (PulseSS)

Why decentralized scheduling?

Centralized required full knowlege of the Network

- Requires a lot of metadata communication
- Not scaleable (NP Complete problem)

Table: Comparison between WirelessHART and PulseSS

Protocol	WirelessHART	PulseSS
Medium Access Control	Central, by the Network Manager	Decentral
Knowledge of global network required	Yes	No
Maxiumum number of Nodes	600	unlimited
Source of Timing	Built in, but only basic mechanics defined	Build in
Timing Provided to Sensors	Yes	Yes
Timing Accuracy	few μs per hop	Simulated $<$5ns per hop
Network Layer Defined	Yes	No

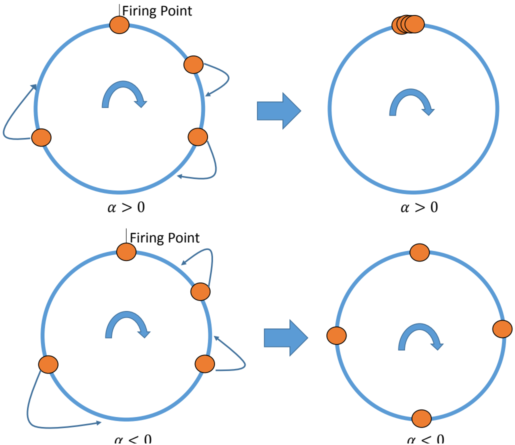
TDMA scheduling for fully connected network

Top Synchronization

-All nodes together

-Bottom Desynchronization

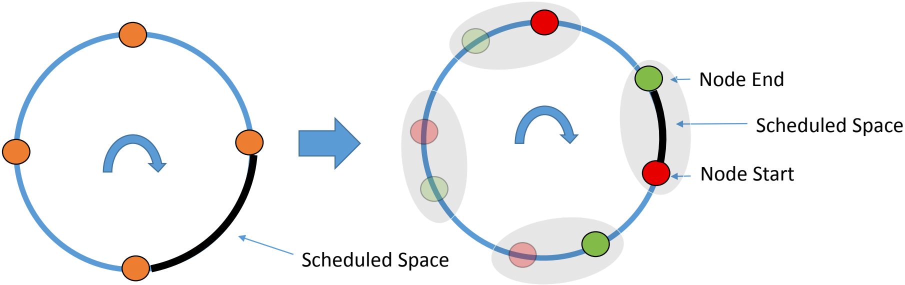
All nodes spaced appart equally



PCO Phase Update

$$\Phi_j(t_i^+) = \begin{cases} \min\{(1+\alpha)\Phi_j(t_i), 1\} & \text{if } \rho < \Phi_j(t_i) < 1 \\ \Phi_j(t_i) & \text{else} \end{cases}$$

Damand based share for Fully Connected Network



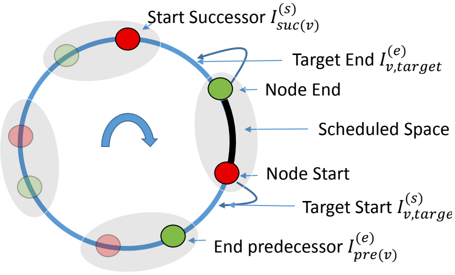
Introduce Start and End for each node

- One Node low demand D takes less
- One Node high demand D takes more

Each node v gets a share proportional its Demand $D_v / \sum_i D_i$

Proportional Fair scheduling for fully Connected Network

Damand D based share:



$$I^{(s)}_{v,\mathrm{target}} = \frac{D_v+\delta}{D_v+2\delta} I^{(e)}_{\mathrm{pre}(v)}\left(t^{(s)+}_{\mathrm{suc}(v)}\right) + \frac{\delta}{D_v+2\delta} I^{(s)}_{\mathrm{suc}(v)}\left(t^{(s)+}_{\mathrm{suc}(v)}\right)$$

$$I^{(e)}_{v,\mathrm{target}} = \frac{\delta}{D_v+2\delta} I^{(e)}_{\mathrm{pre}(v)}\left(t^{(s)+}_{\mathrm{suc}(v)}\right) + \frac{D_v+\delta}{D_v+2\delta} I^{(s)}_{\mathrm{suc}(v)}\left(t^{(s)+}_{\mathrm{suc}(v)}\right)$$

Proportional Fair Scheduling

For a single cluster network each node gets a portion of the frame proportional to its own demand D (Pagliari et al., 2009)

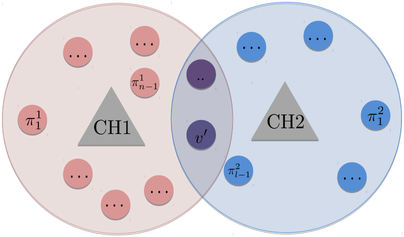
A guardspace proportional to δ is left empty.

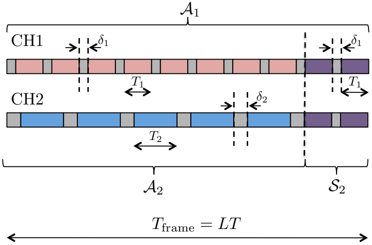
Multicluster Network

So far we had the network fully connected
Introducing Clusterheads (CH) to solve:

- Hidden Station Problem
- Aquire Time of flight information

Proportional Fair Scheduling works also with spase multicluster networks
Shared nodes are limited by denser cluster



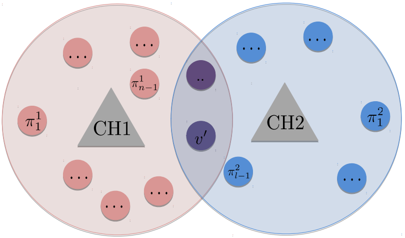


Signals used

Signal used: (idential for all nodes)

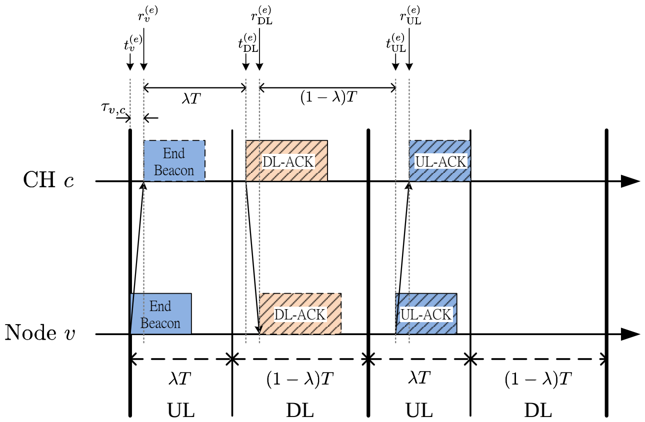
- Start and End Beacon from nodes
- Start Acknowlege and end Acknowlege from CH

Key idea: If multiple nodes send the same signal at the same time, they will overlap constructively (and receiver sees just 1)



Example Shared node gets acknowleged by both CH, but looks as one to a receiver

Measureing time of flight



Time of flight is measured with a back and forth sixnal exchange

- All timings except time of flight is known

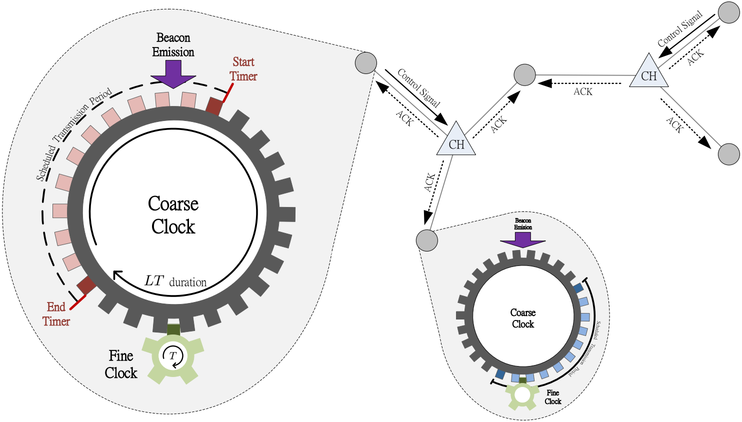
$$Time\_of\_flight = (time\_received - time\_sent - known\_delays)/2$$

We introduced:

- Synchronization
- Scheduling/Desynchronization
- Time of flight

Let's combine!

PulseSS Protocol



-One ‘cycle’ concist of multiple T_{PCO} periods ‘Fine Clock’

-Desynchronization with Start and end timer

-CH for Time of flight and hidden station

PulseSS Protocol

- The nodes fire two distinct Start/End beacons
- CH in range ack withing the PCO slot

Local variables (normalized to T_{PCO}) for each node v

$I_v^{(s)}, I_v^{(e)} \in (0, L]$ Discrete counters responsible for the scheduling

$\Phi_v(t) \in (0, 1]$ Continuous clock for the PCO synchronization

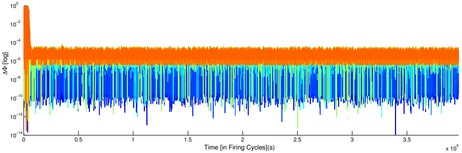
$$\Psi_v^{(s)}(t) \triangleq I_v^{(s)}(t) + \Phi_v(t)$$

$$\Psi_v^{(e)}(t) \triangleq I_v^{(e)}(t) + \Phi_v(t)$$

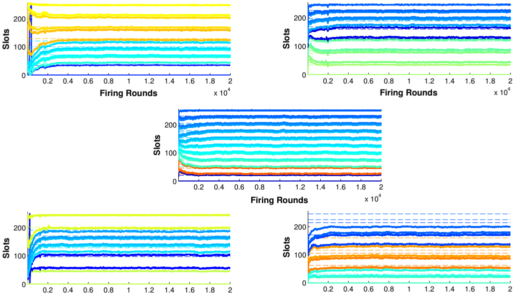
- Nodes fire Start (End) when $\Psi_v^{(s)}(t) = 1(\Psi_v^{(e)}(t) = 1)$
- When they hear other signals, they update $Phi_v(t)$ with the PCO-sync update
- The integer $I_v^{(s)}, I_v^{(e)}$ are updated with the scheduling update

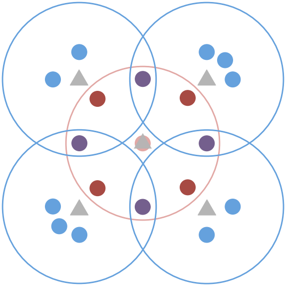
Simulation Results

Synchronization (log)



Scheduling Result



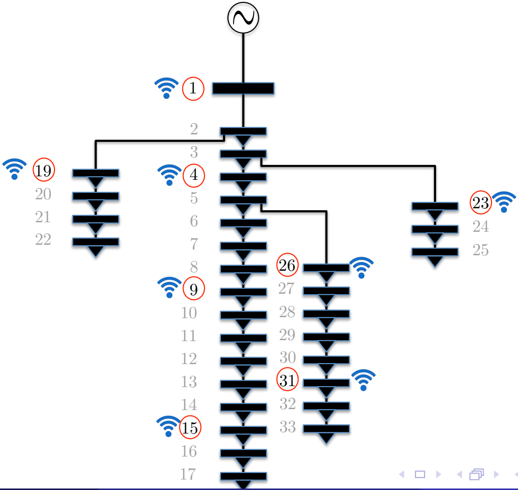


PulseSS-PMU

PulseSS applied to the IEEE-33Bus

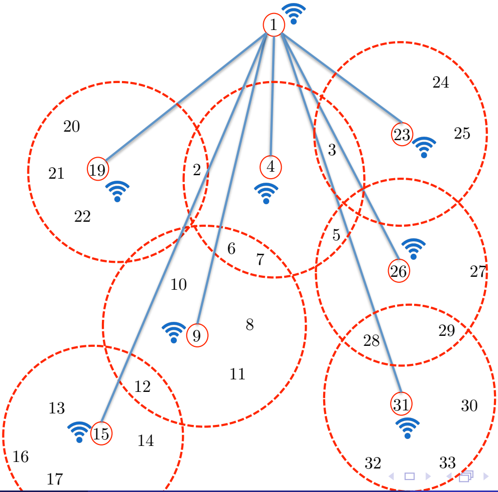
Some stations are typically gateways and collect data => our CHs

Node in the middle connected e.g. via powerline



PulseSS-PMU

PulseSS applied to the IEEE-33Bus
Some stations are typically gateways and collect data => our CHs
Node in the middle connected e.g. via powerline



Wireless Testbed Implementation

Implemented in MicaZ Motes
CPU: Atmel 128L, 8-Bit, 7.3728MHz
Memory: 128KB
Radio: CC2420, 2.4GHz, 250Kbit/s
Zigbee Protocol, Layer 2 access

Wireless Testbed Implementation

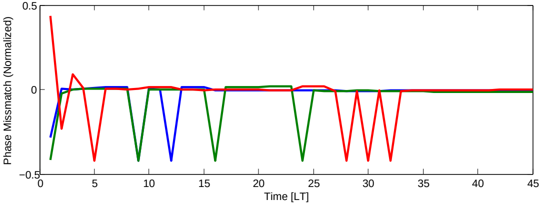
Real World Issues:

Transmissions are instantaneous, they are processed as packets, can fail, are noisy

Caculations take time, block the CPU

Why good timing matters

Why good timing matters:
Single cluster with 3 Nodes (should converge very fast)



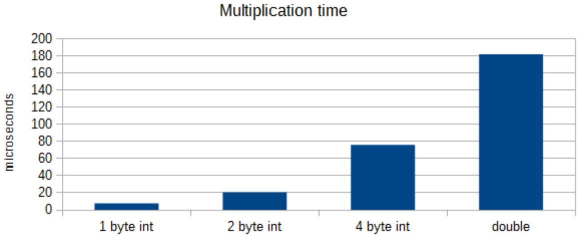
Wrong timing causes the PCO to ‘loose sync’
This negative example shows how its NOT done

Timing Wireless

Task for a single transmission	Time [us]
Set Tx_target on uC-SPI	0.136
Push first Byte on uC-SPI	0.136
Set up SPI to Tx-Buffer on CC2420	0.272
Transmit first Byte from uC-SPI to CC2420	1.180
Switch CC2420 from Rx to Tx	192
Calibration of Tx filters	192
Send Preamble+SPF	160
Calibration Rx	104
Send Rx_Byte via SPI to uC-SPI input	1.180
Push message from uC-SPI to uC-register	0.136
Send Headers (9 Byte)	288
Send Data (4 Byte)	4*32
Enable Interrupt (uC informed)	0.136
Stop Byte Transfered (t_{stop})	32
Total (t_{single})	1099.176us

Timing - Microcontroller

How long does it take to call and excute functions?



Atmel 128L, 8Bit CPU, no hardware multiplication
Nothing else can be executed while CPU is busy
Avoid multiplications where we can
PCO calculation: $\min\left\{(1+\alpha)(\Phi_j(t_i)+\tau_{(i,j)}-t_{Delay}),1\right\}$

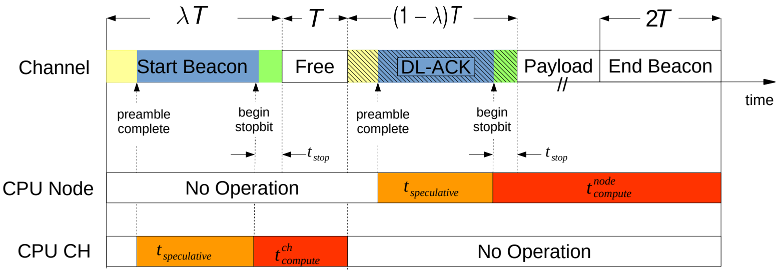
Set $\alpha = 0.125$ only one shift & add operation

Ignore time of flight

Ignore time of flight:
$\tau_{(i,j)} <$ 1 Clock cycle $(7.3728MHz \cdot c = 40.66m)$

Firing and acknowledgment implementation

Deadlines:



We need to be done before next operation.
CPU gets informed once preamble received
=> speculatively compute PCO

Overlapping multiple acknowledgements

Transmissions can fail:

Example: 1 Shared Node transmitts, multiple CH respond.

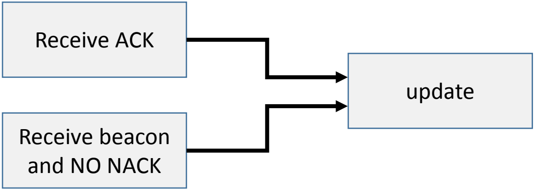
If perfectly alligned in time OFDM multipath transmission

If not alligned perfectly ACK collide mid air and none arrives.
=> not bad for PCO (when syncronized an update does nothing)
=> not tolerateable for scheduling as the node thinks the channel is busy and backs off

We have to make sure at least one CH acknowleges successfull so that the protocol can react.

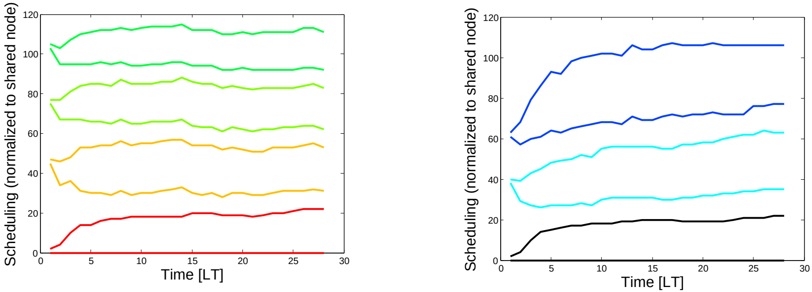
Solution of multiple ack problem

Nodes update when



Testbed scheduling Results

2 Clusters, 1 Shared Node, 3Nodes in Cluster 1, 2 Nodes in Cluster 2

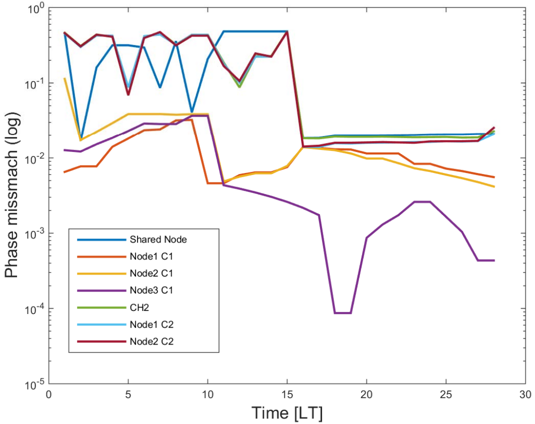


Shared node in red (l) and black (r)

Left: 4 Nodes with equal share
Right: Shared Node less than other 2 Nodes, as shared Node congested from left.

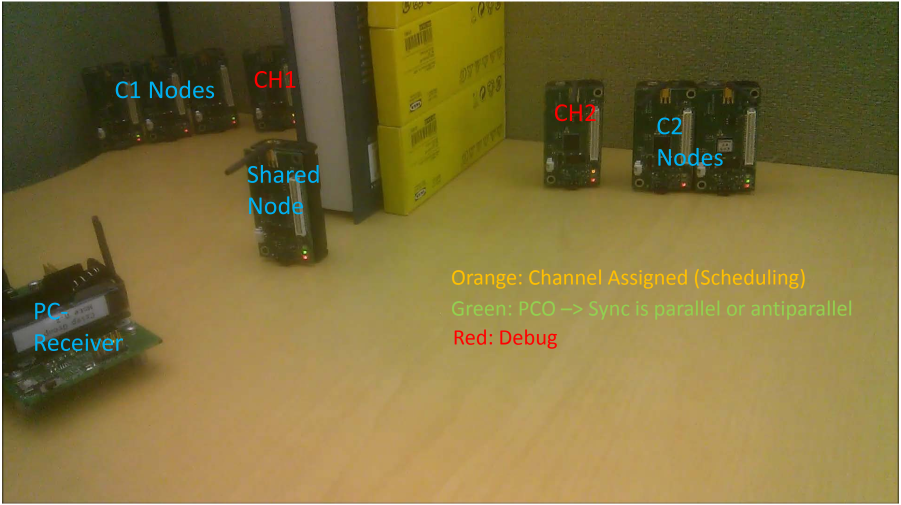
Synchronization Results

2 Clusters, 1 Shared Node, 3Nodes in Cluster 1, 2 Nodes in Cluster 2



Final Accuracy: 80us (40us/ cluster)

Results



https://youtu.be/diErISxxg-c

Conclusions and Future Work

- PC-PMU is a completely new device for low cost Smart Grid PMUs
- We analyzed its accuracy in non ideal synchronization conditions
- Implementation of PulseSS in microcrontroller

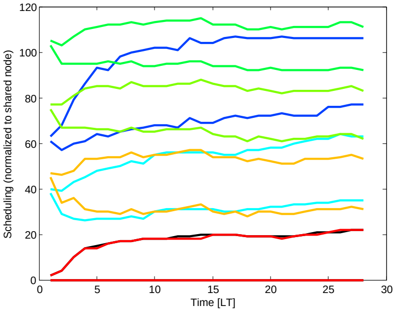
In future work we are going to focus on:

- More complex effects and network configurations
- Implementation with FPGA $->$ Physical Layer Access
- Compensation for propagation delays

Thank you for your attention

Note on Results

Scheduling Results
Overlay of scheduling Cluster 1 and 2:



Shared not is not identical.

Scheduling result recoded by each CH according to each CH clock
- Not rounded
- 'Time' progresses differently when not yet converged

PC-PMU Architecture