

Northumbria Research Link

Citation: Zong, Yan, Dai, Xuewu, Gao, Zhiwei, Binns, Richard and Busawon, Krishna (2018) Simulation and evaluation of pulse-coupled oscillators in wireless sensor networks. *Systems Science & Control Engineering*, 6 (1). pp. 337-349. ISSN 2164-2583

Published by: Taylor & Francis

URL: <http://dx.doi.org/10.1080/21642583.2018.1496043>
<<http://dx.doi.org/10.1080/21642583.2018.1496043>>

This version was downloaded from Northumbria Research Link:
<http://nrl.northumbria.ac.uk/id/eprint/36196/>

Northumbria University has developed Northumbria Research Link (NRL) to enable users to access the University's research output. Copyright © and moral rights for items on NRL are retained by the individual author(s) and/or other copyright owners. Single copies of full items can be reproduced, displayed or performed, and given to third parties in any format or medium for personal research or study, educational, or not-for-profit purposes without prior permission or charge, provided the authors, title and full bibliographic details are given, as well as a hyperlink and/or URL to the original metadata page. The content must not be changed in any way. Full items must not be sold commercially in any format or medium without formal permission of the copyright holder. The full policy is available online: <http://nrl.northumbria.ac.uk/policies.html>

This document may differ from the final, published version of the research and has been made available online in accordance with publisher policies. To read and/or cite from the published version of the research, please visit the publisher's website (a subscription may be required.)




**Northumbria
University**
NEWCASTLE



UniversityLibrary

Simulation and evaluation of pulse-coupled oscillators in wireless sensor networks

Yan Zong , Xuewu Dai, Zhiwei Gao, Richard Binns and Krishna Busawon

Department of Mathematics, Physics and Electrical Engineering, Northumbria University, Newcastle upon Tyne, UK

ABSTRACT

The clock in embedded systems usually is driven by a crystal oscillator and implemented via a *counter* register, such a crystal clock is non-identical and drifting due to the manufacturing tolerance and variation of working conditions. Thus, a common time among distributed wireless sensor nodes, also referred to as Time Synchronization, is required for many time-sensitive wireless applications, such as collaborative condition monitoring, coordinated control and localization. Inspired by fireflies' behaviour, the Pulse-Coupled Oscillators (PCO) has been proposed for synchronization in complex networks. Since the concurrent transmission of PCO's *Pulses* is impossible in Wireless Sensor Networks (WSNs), the *desynchronization* mechanism is adopted to ensure the implementation of PCO in WSNs. Moreover, due to the uncertainties in radio channels and the complexities of communication protocols and packet-exchange behaviours in wireless networks, it is challenging to have a closed-form solution to the performance of PCO synchronization in WSNs. The realistic software simulation, in particular, the discrete event simulator has been a powerful tool to exam the performance of communication protocols in various scenarios, since an order sequence of well-defined event in time is to represent the behaviour of a complex system. This paper presents the development of a pulse-coupled oscillators time synchronization simulator on the OMNeT++ platform for simulating and studying its behaviour and performance in sensor networks. A clock module with configurable phase and frequency noises, and adjustable and higher resolution is developed to mimic various crystal oscillators in embedded systems, for example, the 32.768 kHz real-time clock. The developed simulator also supports the full functions devices defined by ZigBee protocol, which allows realistic simulation of multi-hop IEEE 802.15.4 wireless networks. Finally, the intensive simulations of classical PCO with the refractory period in IEEE 802.15.4-based WSNs have been carried out to demonstrate the features and benefits of the developed simulator. It is shown that for the non-identical and time-varying PCO clocks in the WSNs, the achieved synchronization will lose gradually, and the time that maintained synchronization depends on the length of refractory period.

ARTICLE HISTORY

Received 28 February 2018
Accepted 29 June 2018

KEYWORDS

Time synchronization;
pulse-coupled oscillators;
desynchronization;
OMNeT++ simulation;
wireless sensor networks

1. Introduction

Time Synchronization (TS), aiming to provide a common sense of timing among distributed sensor nodes, is one of the enabling technologies for many Wireless Sensor Networks (WSNs) applications, such as collaborative condition monitoring, coordinated control, time-of-flight localization and underwater navigation and tactical surveillance. In such wireless applications, a network of distributed sensors is dedicated to cooperatively monitor physical or environmental variables such as location, sound and pressure at different locations, which requires precise timing among sensor nodes.

Even though the behaviour of complicated wireless networks is too complex to have a closed-form solution to the time synchronization performance in WSNs, the Discrete Event Simulator (DES) has been recognized as a

powerful simulation tool for studying complex systems (Huang et al., 2015).

1.1. Pulse-coupled oscillators and time synchronization in wireless sensor networks

Packet-exchange time synchronization is recognized as one of the commonly used synchronization methods in low-cost indoor or underwater WSNs. This is particularly true in those wireless applications where satellite-based time synchronization method is impossible due to the economic issue and the difficulty to receive the satellite's signal (e.g. Amazon robotic warehouse management systems (Shead, 2017) and underwater monitoring (Petrioli, Petrocchia, Potter, & Spaccini, 2015)). Typical examples of packet-exchange based time synchronization algorithms

are Precise Time Protocol (PTP) (IEEE Instrumentation and Measurement Society, 2008), Reference Broadcast Synchronization (RBS) (Elson, Girod, & Estrin, 2002) and Flooding Time Synchronization Protocol (FTSP) (Maroti, Kusi, Simon, & Ledeczi, 2004). In these methods, clock offsets among sensor nodes, observed by recording packet's transmission and received time (also known as the timestamp), are used to correct the drifting clocks of sensor nodes. Obviously, the precision of packet-exchange based time synchronization is dependent on the accuracy of the timestamp. Moreover, since the synchronization error of each hop is accumulated with the increase of hop distance, the performance of packet-exchange time synchronization is weakened in the multi-hop large-scale WSNs.

On the other hand, inspired by the synchronous flashing of fireflies observed in certain parts of southeast Asia, a bio-inspired mathematical model, namely, Pulse-Coupled Oscillators (PCO) has been proposed to enable synchronization in complex networks (Mirollo & Strogatz, 1990) (Mohammadpour & Binazadeh, 2018). In the pulse-coupled oscillators, the *Pulse* is broadcasted by individual oscillator periodically according to its state and oscillation frequency. Upon receiving a *Pulse* from neighbours, the oscillator adjusts its state to catch up the firing oscillators. Eventually, all oscillators in the network fire and broadcast the *Pulses* simultaneously and synchronization of the network is achieved. However, most research work in PCO literature is theoretical studies. The classical PCO by (Mirollo & Strogatz, 1990) was approved that synchronization can be achieved under the following assumptions: (i) All the internal dynamics of oscillators are identical (i.e. the frequencies of oscillators are same). (ii) There is no coupling delay during the pulse-exchange among the oscillators. All of the aforementioned assumptions are not true when it comes to any real-world environments, and classical PCO need be improved to implement it to realistic networks. It is worth noting that the term *oscillator* is used to describe the mathematical model, the term *sensor node* is adopted when implementing the mathematical model to wireless sensor networks, and the *Pulse* packet of WSNs is used to describe the physical *Pulse* signal in PCO.

Although PCO features at its simplicity and scalability, only a few studies are about applying PCO for time synchronization in WSNs. This is because some assumptions in PCO theories, such as identical and constant crystal oscillator frequency in all sensor nodes, concurrent transmission of firing *Pulse* packet and no transmission delay, are not feasible in any practical WSNs (Bojic & Nymoen, 2015). In embedded systems, the crystal oscillators' frequencies indeed are non-identical and time-varying due to the manufacturing tolerance and

changes of environment factors (e.g. temperature and power supply voltages). In Radio Frequency (RF) communications, packet transmission delay always exists and may be varying randomly depending on the length of the packet, channel conditions and the Media Access Control (MAC) scheme. When the delay is present in a system of pulse-coupled oscillators, the classical PCO algorithm of (Mirollo & Strogatz, 1990) fails to achieve synchronization (Ernst, Pawelzik, & Geisel, 1995), (Mathar & Mattfeldt, 1996), (Tyrrell, Auer, & Bettstetter, 2010). To address this challenge, the refractory period was introduced by (Mathar & Mattfeldt, 1996) to retain the stability of a network. (Mathar & Mattfeldt, 1996) and (Tyrrell, Auer, & Bettstetter, 2008) stated that the synchronization accuracy was bounded by the coupling delay (e.g. the propagation and transmission delays in WSNs) in the two-node network. (Tyrrell et al., 2008) also analysed the accuracy of PCO with the refractory period in fully-connected wireless networks, and it was concluded that neighbouring nodes are synchronized with an accuracy that is always equal or below the coupling delay (i.e. propagation delay in (Tyrrell et al., 2008)) in multi-nodes network. It is worth noting that oscillators analyzed and simulated in (Mathar & Mattfeldt, 1996) and (Tyrrell et al., 2008) are identical oscillators (i.e. same internal dynamics for all WSNs nodes' clock oscillators). Even though it was proved that the synchronization of PCO clocks with the different constant frequencies can be achieved under two conditions (i.e. instantaneous synchronization condition and synchronization maintaining condition) (An et al., 2010), these two conditions of (An et al., 2011) may not be met in practice.

Another issue on implementing PCO to practical WSNs is impossible concurrent transmission of PCO's firing *Pulse* packet. In wireless communications of single RF channel, only one packet transmission is allowed at any time. Otherwise, packets from concurrent transmitters interfere with each other (known as transmission collision) and no packet can be successfully received at the receiver. To address this issue, the concept of *desynchronization*, logical opposite of synchronization, is adopted. In this approach, the *Pulse* packets are transmitted in a uniformly distributed fashion (i.e. as far away as possible from all other nodes), rather than the *Pulse* packets of all nodes are broadcasted to the wireless channel simultaneously (Degesys, Rose, Patel, & Nagpal, 2007).

Since the fact that the assumption of identical oscillators, no transmission delay and concurrent transmission of classical PCO cannot be met in any real-world environments, the PCO dynamics are too complicated to have a closed-form solution if those assumptions are released. As a result, an analytic solution to PCO's performance in realistic WSNs is very difficult, and it is of importance to

evaluate PCO's behaviour and performance in WSNs by realistic simulation.

1.2. Discrete event simulator

With the rapid development of the computer science, discrete event simulator has become a powerful cutting-edge simulation tool for studying complex systems, such as complicated networks and protocols. In the DES, an ordered sequence of well-defined events in time (e.g. packet transmission and reception, interrupts and so on) is used to represent the complex behaviours of complicated networks and protocols (Huang et al., 2015). This feature ensures the DES be a good choice for simulating synchronization protocols in various networks. Widely used DESs for WSNs include the commercial OPNET (Kucuk, Bandirmali, Kavak, & Atmaca, 2013), open source simulator NS2/3 (NS3, 2018) and OMNeT++ (OMNeT++, 2018). It has been shown that the software simulator OMNeT++ is one of the best simulation platforms for WSNs in academia, featuring at rich simulation libraries, modular design and open structure, debugging and tracing capability, graphical user interfaces, and simulation efficiency (such as, delivery ratio, memory efficiency, CPU Utilization and Computational Time) (Khan, Hasbullah, & Nazir, 2014), (Sharif & Sadeghi-Niaraki, 2017), (Xian, Shi, & Huang, 2008). Many simulators also have been developed on the platform of OMNeT++, such as INET, Castalia, MiXiM and so on.

There have been some DESs for simulating time synchronization, such as Castalia with clocks (Ferrari, Meier, & Thiele, 2010) and realistic simulator TS2 (Huang et al., 2015). Since industrial WSNs usually work in a complicated electromagnetic environment, and a detailed radio channel model is required to simulate the packet transmission, environmental noise and packet loss. In addition, there are conflicts and interferences among multi-point communication, and the simulation of the physical layer has a significant impact on the accuracy of WSNs simulation (Huang et al., 2015). Several OMNeT++-based simulators have been developed in the last decade for WSNs simulation. Most of them are for the simulation of MAC and higher layers, and their treatment to the physical layer, hardware and wireless channel are superficial (Huang et al., 2015). On the OMNeT++ platform, MiXiM (Kopke et al., 2008) is a simulator that has detailed radio channel model and physical layer to achieve the high-fidelity simulation of physical and MAC layers. On the basis of MiXiM, a recent realistic time synchronization simulator for wireless sensor networks is TS2 of (Huang et al., 2015), which is developed to simulate and evaluate the PTP-like time synchronization in WSNs. However, only the Reduced Function Device (RFD) of ZigBee protocol is

simulated in TS2. Even though both (Ferrari et al., 2010) and (Huang et al., 2015) provide the realistic drifting clock model, in the literature, there still has a lack of a simulator that simulates the PCO synchronization with realistic crystal oscillator clocks, radio channel model and IEEE 802.15.4 standard.

The main contribution of this paper presents the development of a high fidelity PCO simulator for IEEE 802.15.4-based WSNs, including a PCO model implemented by crystal oscillators, *desynchronization* mechanism to implement transmission of PCO's *Pulse* packet without violating the non-concurrent transmission constraint in IEEE 802.15.4 wireless networks. Based on the *desynchronization* mechanism, a TS superframe is proposed that is compatible with the IEEE 802.15.4 superframe. The drifting PCO clock is modelled and simulated realistically with configurable phase and frequency noises and at adjustable and higher resolution, to mimic the crystal oscillators in embedded systems, for example, the 32.768 kHz Real-Time Clock (RTC) oscillator. Another main contribution of developed simulator developed on the basis of MiXiM features at a generic, flexible and modular architecture for future extension. The developed simulator also supports both Full Functions Devices (FFD) and RFD of ZigBee protocol, which is compatible with IEEE 802.15.4 standard in realistic distributed sensor networks and enables the high scalability for multi-node and multi-hop distributed sensor networks simulation. The intensive simulations of classical PCO with the refractory period in IEEE 802.15.4-based WSNs have been carried out to demonstrate the application of the developed simulator, and study PCO's behaviour and evaluate its synchronization performance in realistic WSNs. It is shown that due to the existence of the transmission delay, the full synchronization cannot be achieved in the identical classical PCO clocks of wireless networks. Moreover, for the non-identical and time-varying PCO clocks in the WSNs, the achieved synchronization will lose gradually, and the time that PCO clocks maintain the synchronization is dependent on the refractory period.

The rest of this paper is as follows: Section 2 introduces the framework of the developed simulator. Section 3 presents the development of PCO clock model, and the classical PCO correction mechanism with the refractory period is detailed in Section 4. The proposed IEEE 802.15.4 compatible superframe based on the *desynchronization* mechanism, and timestamping are also introduced in Section 5. Section 6 details the implementation of PCO clock, *desynchronization* and FFD in OMNeT++. Finally, the time synchronization performance of identical classical PCO with the refractory period in both single-hop and multi-hop WSNs, and non-identical classical PCOs with the refractory period in single-hop WSNs are

evaluated in Section 7, followed by Section 8 presents the conclusion.

2. Simulation framework

The objective of the developed simulator is a simulation framework capable of realistically modelling and simulating drifting PCO clocks at higher resolution and in large-scale multi-hop realistic distributed sensor networks. This section presents the requirement analysis and the architecture of developed simulator.

Compared to the wired channel, the wireless channel in WSNs has complicated impacts, such as the fading, shadowing, interference and collision resulting from media sharing, hidden terminals and so on, on the time synchronization performance. Particularly, the classical PCO assumes the delay that a complete *Pulse* packet is received successfully is zero. However, due to the transmission delay, possible collision, backoff and re-transmission, the assumption of zero delay is unable to be met in any real-world WSNs. Other factors, such as the technological limitation of IEEE 802.15.4 wireless networks (e.g. the non-concurrent transmission constraint), also have negative impacts on the time synchronization. Since it is difficult to study these impacts analytically resulting from their complexity, it is recommended to adopt a realistic simulator to make a sound performance analysis. In order to build a realistic simulator of time synchronization, an accurate and detailed clock model, wireless sensor nodes, wireless channel and networks and in-depth understanding of the inter-effects among related factors are required. It is possible to develop such a detailed model from scratch; however, it will be a tedious and error-prone task (Huang et al., 2015). Thanks to many researchers' contributions to OMNeT++ open-source community, some existing modules and source codes can be re-used to develop the simulator.

The structure of proposed simulation framework is shown in Figure 1, and four main components of proposed simulation framework are as follows:

- *World Manager* module defines the environment model representing the geographical environment of realistic WSNs, namely, the size of the WSNs' deployment area, obstacles blocking/attenuating wireless signals and so on.
- *Connection Manager* module manages the mobility and connectivity, such as the varies of the wireless channel and the change of network topology resulting from the moving nodes.
- The wireless channel, *Channel*, denotes the features of the wireless channels and its impacts on packet exchange among sensor nodes, such as propagation delay, collision, attenuation and interference.
- WSNs nodes: there are three types of nodes, namely, master, slave and relay nodes. The relay node used in the multi-hop network is to realize the FFD functions, and the master and slave nodes in the simulator are to mimic the RFDs.

In Figure 1, four layers of ZigBee protocol stack are implemented to the simulator by C++ classes (referred to as modules), namely, the application layer (*app*), the network layer (*netw*), the MAC layer (*mac*) and the physical layer (*phy*). It is notable that, as a realistic simulator, the *phy* module mimics the TI CC2420 IEEE 802.15.4 chip, including the wireless channel sensing procedure, SNIR of each packet and so on, which ensures a better and much accurate simulation of the packet transmission delays, jitters and the PCO in IEEE 802.15.4 networks (Huang et al., 2015). Since the PCO indeed is a cross-layer protocol requiring two main procedures, resetting-firing and transmitting/receiving, the PCO is implemented by two modules in the simulator. The TDMA *mac* module is to transmit or receive the packet to/from wireless channel,

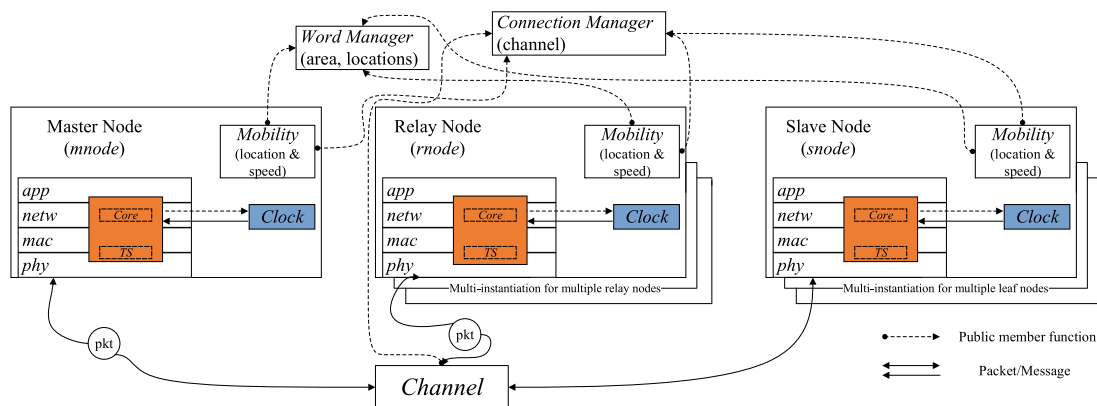


Figure 1. Simulation framework.

and the *Core* module is for firing a *Pulse* of PCO. Moreover, four additional modules used in the simulator are as follows: (i) *Clock* module, providing local time to other modules of the sensor node, simulates the node's clock. (ii) *Mobility* module stores and updates the location and speeds of nodes, reporting them to the *World Manager* and *Connection Manager*. (iii) *World Manager* and *Connection Manager* update the location of nodes and configure the wireless links in the network topology. (iv) *TS* module is for timestamping, which is able to realize the different accuracy timestamp.

Compared with most existing simulators, the proposed simulation framework provides the following unique features: (i) Reality and fidelity: the WSNs node's various layers, drifting PCO clock can be simulated accurately and realistically. (ii) Scalability and multi-hop simulation: simulation of several relay and slave nodes are required, as well as the multi-hop time synchronization in the large-scale WSNs. (iii) Extendibility and compatibility: the simulator is compatible with most existing simulators, e.g. MiXiM, so that the simulator can be easily extended to existing or emerging networks, such as WiFi, WiMAX, LTE and so on. These features make the proposed simulator more realistic for PCO synchronization of distributed sensor networks.

3. Modelling dynamics of pulse-coupled oscillator clocks

The clock of a sensor node usually is constructed from hardware and software components. To be specified, the clock module of a WSNs node consists of (i) a crystal oscillator ticks at a specified frequency; (ii) a *counter* (or a chain of *counters*), counts the number of ticks generated by the crystal oscillator. Physically, the output of a crystal oscillator is periodic sinusoidal or square waves, while, this output can be converted to a count value plus one per clock cycle through the process the *counter* which is digital timing circuit (Zong, Dai, & Gao, 2017), and this count value can be used to indicate the time by comparing and converting to the count value. The output of a crystal oscillator is generally called the *physical clock*, while, the cumulative count value of the *counter* register is called the *software clock*. To avoid the physical clock discontinuity, it is recommended to apply the time synchronization algorithms to *software clock*, rather than the *physical clock* (Bojic & Nymoen, 2015), and the time synchronization algorithms in this paper are therefore to adjust the *software clock*.

3.1. Drifting classic clock

In reality, owing to manufacturing tolerance and crystal capacitive loading mismatch, the phase of a crystal

oscillator suffers from random changes, and the oscillator frequency is also time-varying and drifting due to the crystal and oscillator temperature drift (Zong, Dai, & Gao, 2017). In order to model the non-identical and time-varying clock in the distributed system, a random process $\phi(t)$, representing all the instant phase deviation, is used to model the phase noise of an crystal oscillator. $\alpha(t)$ is to represent the oscillator frequency change at time t . The output of a drifting clock is modelled as

$$V(t) = V \sin \left(2\pi \left(f_0 t + \int_0^t \alpha(\tau) d\tau \right) + \phi(t) \right) \quad (1)$$

where t represents the *reference time*, V is a constant amplitude, f_0 means the nominal frequency and the period of the sinusoidal signal is $\tau_0 = (1/f_0)$.

As the time instant is represented by the phase of a sinusoidal wave, the sinusoidal wave is converted into a tick and a *counter* register counts the ticks digitally to generate the traditional time unit of microseconds, milliseconds, seconds, minutes and so on (e.g. the phase of 2π denotes a time period of τ_0 , which also indicates a *counter* increment of 1). This process is modelled by comparing the instant phase against 2π

$$\begin{aligned} n &= \frac{2\pi(f_0 t + \int_0^t \alpha(\tau) d\tau) + \phi(t)}{2\pi} \\ &= f_0 t + \int_0^t \alpha(\tau) d\tau + \frac{\phi(t)}{2\pi} \end{aligned} \quad (2)$$

where n is an integer indicating how many cycles have occurred, the n -th counting event can be referred to as the event of the *counter* reaching the value n . The floor operator $\lfloor x \rfloor$ means the largest integer not greater than x . The time instant $t[n]$ can be used to represent the *reference time* at the n -th event, in addition, it is common to use $C[n]$ to represent the *classic clock* time of the n -th event. The clock time of a drifting *classic clock* is given by

$$C[n] = n \cdot \tau_0 = t[n] + \frac{\int_0^{t[n]} \alpha(\tau) d\tau}{f_0} + \frac{\phi(t[n])}{2\pi f_0} \quad (3)$$

It is obvious that the *classic clock* $C[n]$ is inaccurate and different from the *reference time* $t[n]$ due to the accumulated frequency variations ($\int_0^{t[n]} \alpha(\tau) d\tau / f_0$) and the phase noise ($\phi(t[n]) / 2\pi f_0$).

3.2. Discrete model of pulse-coupled clocks

The difference between the *classic clock* time $C[n]$ and *reference time* $t[n]$ is regarded as the offset, which depends on the accuracy of the clock. Let $\theta[n]$ represent the offset

of *classic clock* $C[n]$ at time $t[n]$

$$\theta[n] = \frac{\int_0^{t[n]} \alpha(\tau) d\tau}{f_0} + \frac{\phi(t[n])}{2\pi f_0} \quad (4)$$

Even though the clock frequency suffers from the oscillator temperature drift, the frequency of drifting clock can also be considered as a constant value due to the slow frequency change in a sufficiently short clock update period. Therefore, the term $\int_0^{t[n]} \alpha(\tau) d\tau$ of (4) can be piecewise discretized as $\sum_{m=0}^{n-1} \alpha[m] \tau_0$. In addition, the term $(1/2\pi f_0)$ can be viewed as a scaling factor, and the possibility destiny function of $\phi[n]$ is similar to that of $((\phi(t[n]))/(2\pi f_0))$, therefore, $((\phi(t[n]))/(2\pi f_0))$ is able to be re-written as a discrete form of $\phi[n]$.

In the discretized *classic clock* model, the clock offset $\theta[n]$ of the n -th clock update event is

$$\theta[n] = \frac{\sum_{m=0}^{n-1} \alpha[m] \tau_0}{f_0} + \phi[n] \quad (5)$$

The skew $\gamma[n] = ((f(t[n]) - f_0)/f_0)$ represents the deviation from the nominal frequency f_0 . For discretized clock model, the skew $\gamma[n]$ is equal to $\gamma[n] = (\alpha[n]/\tau_0)$ for one clock update period of $[t[n], t[n+1]]$. Thus, by introducing $\omega_\theta[n] = \phi[n+1] - \phi[n]$, the offset of the $(n+1)$ -th clock update event is re-written as

$$\theta[n+1] = \theta[n] + \gamma[n] \tau_0 + \omega_\theta[n] \quad (6)$$

The skew $\gamma[n]$ is also assumed as a constant value within diminutive clock update period due to the slow change of skew, and the auto-regressive (AR) model is adopted to model the skew as a time-varying process in an auto-regressive manner with a small perturbation (Zong, Dai, & Gao, 2017). The skew of the $(n+1)$ -th clock update event is

$$\gamma[n+1] = p \gamma[n] + \omega_\gamma[n] \quad (7)$$

where $\omega_\gamma[n]$ is the noise with zero mean, and p , the parameter of the first-order AR model, is less than but close to 1. In addition, both offset noise $\omega_\theta[n]$ and skew noise $\omega_\gamma[n]$, two uncorrelated random process, are subjected to zero-mean gaussian distribution with standard deviations σ_θ and σ_γ respectively (Zong, Dai, & Gao, 2017).

In the PCO free-running mode, the clock state P increases from zero to threshold value defined by φ . When clock state P reaches the threshold, it is reset to zero. This progress is modelled by comparing the *classic clock* $C[n]$ against the multiple of threshold $(N[n] \varphi)$

$$P[n] = C[n] - N[n] \varphi = t[n] + \theta[n] - N[n] \varphi \quad (8)$$

where $N[n]$ denotes the total number of PCO fires at present n -th clock update event. If the PCO clock fires at the n -th clock update event, $N[n] = N[n-1] + 1$. Otherwise, $N[n] = N[n-1]$.

4. Classical clock correction mechanism

In the interactive mode of classical PCO with the refractory period indicated by δ , the clock state evolves as mentioned in the free-running mode, in addition, the clock state P is pulled up by one coupling strength ϵ only if P is greater than δ , upon the reception of a *Pulse* packet from another node due to the nodes are coupled with each other (Mathar & Mattfeldt, 1996). A *Pulse* packet is broadcasted immediately if the adjusted clock state P exceeds the threshold φ , and the interactive behaviour of the identical classical pulse-coupled oscillators is described as

$$P[k]^+ = \begin{cases} P[k]^-, & \text{if } P[k]^- \leq \delta \\ P[k]^- + \epsilon, & \text{if } P[k]^- > \delta \text{ and } (P[k]^- + \epsilon) < 1 \\ 0, & \text{if } P[k]^- > \delta \text{ and } (P[k]^- + \epsilon) \geq 1 \end{cases} \quad (9)$$

where $P[k]^+$ represents the PCO clock state of an infinitesimal time instant after local clock is corrected at the k -th time synchronization cycle, similarly, the $P[k]^-$ is the clock state of an infinitesimal time instant before drifting clock is corrected at the k -th event.

The network is declared full synchronization if the deviation of reference instants of any two coupled nodes' fire time is zero, this means that any coupled nodes in the network fire and broadcast the *Pulse* packets simultaneously. Even though the sensor nodes' clocks with different frequencies can achieve full synchronization, they will lose synchronization gradually due to the frequency differences (An et al., 2011). In the realistic WSNs, the non-identical and time-varying clocks are unable to achieve the full synchronization. Thus, the certain conditions, which are used to declare the synchronization is achieved in the network, need to be defined.

The synchronization error (indicated by Δ), is defined as the deviation of reference instants of master and the i -th node's fire time in a network

$$\Delta = \begin{cases} t_{P_M} - t_{P_i}, & \text{if } (t_{P_M} - t_{P_i}) \leq 0.5 \\ t_{P_M} - t_{P_i} - 1, & \text{if } (t_{P_M} - t_{P_i}) > 0.5 \end{cases} \quad (10)$$

where t_{P_M} is the reference instant of master's fire time, and the t_{P_i} denotes the reference instant of the i -th node's fire time.

If the synchronization errors of all sensor nodes are not greater than the delay, presented by ι (i.e. $-\iota \leq \Delta \leq \iota$), the network is declared synchronization. It is worth noting that delay between two coupled sensor nodes consists of transmission delay and propagation delay. As the propagation delay is up to 0.33 us in a maximum operation range of around 100 m, the propagation delay is neglected and only transmission delay is considered in this paper. In addition, the convergence time is the time that from the network is powered on to WSNs synchronization is achieved.

5. Desynchronization-based pulse-coupled clocks and timestamping

To solve the challenge that packets from concurrent transmitters interfere each other and no packet may be able to be successfully received in the synchronized PCO-based wireless network, the *desynchronization* is adopted to enable all the *Pulse* packets to be transmitted to the wireless channel in a uniformly distributed way. Next, since the TDMA mechanism is used in the Contention-Free Period (CFP) of IEEE 8.2.15.4 superframe to guarantee the specified slot to be assigned to the specified node, the *desynchronization* mechanism is able to be implemented in the CFP (Farahani, 2011).

5.1. Desynchronization-based pulse-coupled clocks

In embedded systems, since the analogue and physical *Pulse* of classical PCO cannot be transmitted in the wireless channel, the digital *SYNC* packet is adopted to model the PCO *Pulse* signal. Similar to IEEE 802.15.4 superframe, the proposed time synchronization cycle in Figure 2, bounded by neighbouring *SYNC*s of master node, consists of three types of periods, namely, Scheduled Offset (SO), *DESYNC* and Inactive Period. The SO is used to transmit the data by using CSMA or TDMA mechanisms. The *DESYNC* is to ensure sensor nodes to broadcast the *SYNC* packet to achieve the *desynchronization* by TDMA mechanism without queue function. The Inactive Period is for letting the sensor nodes sleep to reduce the power consumption. Thus, at the k -th time synchronization cycle, the i -th node fire time t_{p_i} is

$$t_{p_i} = kT + (t_{SO} + i\tau) \quad (11)$$

where t_{SO} is duration of SO, τ means the *pulse duration* representing the time between neighbouring *SYNC* packet by relay nodes, and T denotes the time synchronization cycle. Moreover, the *pulse duration* is needed to enable the relay nodes transmit or receipt the *SYNC* to avoid the transmission collision.

By taking the *desynchronization* mechanism into consideration, the i -th node clock state of (8) is re-written as

$$P'[n] = t[n] + \theta[n] - N[n]\varphi - (t_{SO} + i\tau) \quad (12)$$

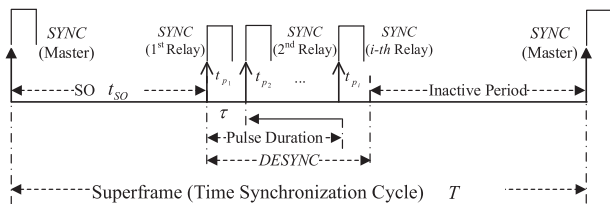


Figure 2. Proposed superframe.

It is notable that the PCO clock state of (12) can represent both perfect and drifting PCO clocks. The clock of (12) is the perfect *desynchronization*-based clock, due to the zero skew and offset (i.e. $\theta[n] = 0, \gamma[n] = 0$). Otherwise, the offset $\theta[n]$ and skew $\gamma[n]$ is updated based on (6) and (7) to produce the drifting *desynchronization*-based PCO clock.

5.2. Timestamping

The timestamp is generated on the local PCO clock based on the reception of *SYNC* packet. As shown in (Correll, Barendt, & Branicky, 2005), (Degesys et al., 2007), due to the interrupt and data processing, the higher layer timestamp, the higher measurement noise. The timestamp between the physical layer and MAC layer can realize the relatively balanced trade-off between the time synchronization accuracy and the expandability and cost of the system. Let $\omega_\eta[k]$ denote the measurement noise which is subjected to zero-mean gaussian distribution with standard deviation σ_η at the k -th time synchronization cycle, the local timestamp is

$$P^*[k] = P'[k] + \omega_\eta[k] \quad (13)$$

where $P^*[k]$ is the local timestamp containing the PCO clock state of *SYNC* packet received at the k -th time synchronization cycle.

At the k -th time synchronization cycle, the measurement clock offset of the i -th node PCO clock based on the reception of *SYNC* from master, $\theta_i[k]$, is given by

$$\theta_i[k] = ((P_i^*[k] - \kappa) + (t_{SO} + i\tau)) - \varphi \quad (14)$$

where κ is the transmission delay.

Similarly, if node i fires before node j at the k -th time synchronization cycle, and measurement clock offset of the i -th (or j -th) relay about the j -th (or i -th) relay, θ_{ij} (or θ_{ji}), is defined respectively

$$\begin{cases} \theta_{ij}[k] = ((P_j^*[k] - \kappa) - |i - j|\tau) - 0 \\ \theta_{ji}[k] = ((P_i^*[k] - \kappa) + |i - j|\tau) - \varphi_j \end{cases} \quad (15)$$

6. Simulator implementation: node structure

The focus of this paper is the simulation of node's PCO clock and WSNs time synchronization, rather than the simulation of MAC and physical layer of sensor nodes. As the MiXiM simulator, TS2 and simulator by (Franco, 2015) have implemented these general modules, such as wireless channel *Channel*, *World Manager*, *Connection Manager*, *mobility*, *TDMA mac* and physical layers *phy* of wireless sensor nodes, they are selected and reused as a starting point for developing the simulator. In this section,

the main attention is paid to the implementation of PCO clock module and the relay node. To be specified, two modules (*Clock* and *Core*) are discussed with details, the reader is referred to (Huang et al., 2015), (Franco, 2015) for more information of other modules. Furthermore, the PCO *Pulse* signal is modelled by *SYNC* packet based on the ZigBee standard. As the choice of a proper module structure is the key to achieve the aims of reusability and extensibility, the node's structure design is first presented and followed by the implementation of the relay node and clock implementation.

6.1. Node structure

Recalling Figure 1, there are three kinds of nodes, namely, the master node named with *mnode*, slave node named with *snode*, and relay node named with *rnode*. In the simulator, all of the sensor nodes are located in the deployment area defined by the *World Manager* module, and these nodes are connected via a wireless channel, *Channel*, managed by the module *Connection Manager*. Several nodes, including slave and relay nodes can be simulated simultaneously in a single network (e.g. *rnode*[0], *rnode*[1], *rnode*[2], ...).

According to the Zigbee protocol stack, a wireless sensor node of Figure 3 is constructed with three additional modules, namely, PCO *Clock*, *Core* and *Timestamp*. And the *mac* of Network Interface Card (*nic*) is modified to TDMA mechanism to realize the functions of PCO time synchronization mechanism. The PCO clock and the clock correction mechanism are simulated realistically by *Clock* module, and *Clock* also provides two interfaces to other modules. One interface is an OMNeT++ gate, which is used to simulate the *SYNC* packet. Each time the *counter* fires (i.e. clock state *P* reaches the threshold), a message named with *SYNC* will be sent to the *Core* module so that the *Core* is able to transmit a *SYNC* packet based on the received message of the *Clock* module.

Another interface of *Clock* module is public member functions. By calling these interface functions, e.g.

getTimestamp() and *adjustClock()*, the time of *Clock* module can be accessed, and the clock can be adjusted more easily by other modules.

Due to the character that the PCO indeed is a cross-layer protocol with data transmission mechanism at MAC layer for high synchronization accuracy, the functionalities of PCO are divided into two modules, namely, *Core*, and *TDMA mac*. The *Core* is responsible for all other operations (such as processing *SYNC* packet and adjusting the clock state) and *mac* is for the PCO's *SYNC* packet immediate transmission/reception. In the simulator, when the *SYNC* packet is sent from *Core* module to *mac* layer, the packet will be passed to the *phy* layer immediately to broadcast the *SYNC* packet, since the queue function of TDMA is cancelled. When the *phy* layer finishes receiving the *SYNC* packet from a wireless channel, it sends the packet up to the *mac* layer. Due to the TDMA mechanism without queue function and no *SYNC* packet is put in the queue to wait for transmission, the *mac* layer passes the *SYNC* to the *Core* module directly without delay.

In the *Core* module, only the packets for PCO protocol will be processed, and those packets from other application protocols are forwarded to the corresponding module. In addition, the *Timestamp* module of (Huang et al., 2015) has been implemented between the MAC layer and physical layer to enable the high accuracy timestamp. The timestamp is attached to the sent/received packet to indicate the sending/receiving time, when *phy* layer receives a packet from *Timestamp* module/wireless channel. The *Timestamp* module is the same for all master, slave and relay nodes, and the *Core* module in master and slave nodes are implemented by different classes. It is notable that all these timestamps are generated from the *Clock* module by calling interface function *getTimestamp()* of *Clock* module.

Remark 6.1: when the sensor nodes work in the *DESYNC* of time synchronization cycle, the MAC layer of RF module works in the TDMA mechanism (without queue functions) to ensure the *SYNC* packet be transmitted to the wireless channel or upper layer efficiently. However, during the Scheduled Offset, the RF module of wireless sensor node transmitter works either in the CSMA or normal TDMA mechanisms.

6.2. Implementation of relay node

The implementation of the relay node, *rnode*, is indicated in Figure 4, the *rnode* module is composed of several modules: a compound module *nic* consisting of *phy* layer, *mac* layer and *Timestamp* module, a basic network layer *netw*, general application layer with burst data transmission *appl* and a *Core* module. Furthermore, the *phy*

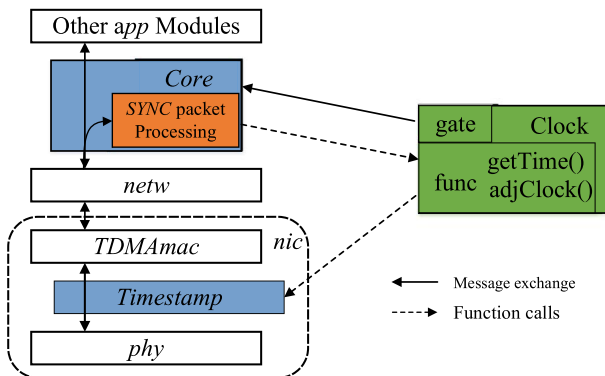


Figure 3. General structure of a wireless sensor node.

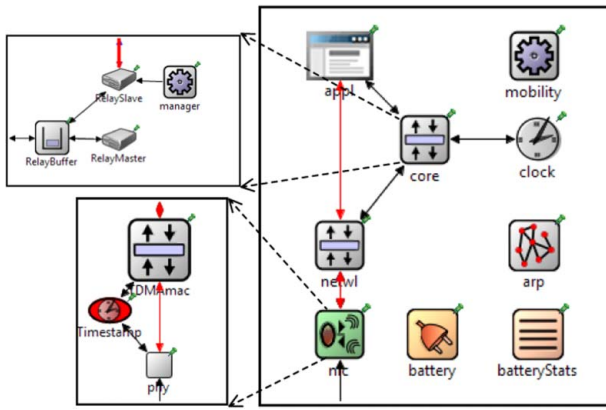


Figure 4. Implementation of the relay node with a realistic *Clock*, and *Core* modules.

and *mac* layers are implemented by *PhyLayerBattery* and *TDMAMac* modules respectively.

- (1) **Relay Core Module:** This module is implemented by a compound module consisting of *RelayBuffer*, *RelaySlave* and *RelayMaster* simple modules. The task of each module is realized in its member function named with *handleMessage()*. In the *RelaySlave* module, when a packet arrives at the module from the higher application layer, *Core* module or *Clock* module, the *handleMessage()* function will be invoked to process the packet and determine the source where is the packet from. If the packet is from the *Clock* module, the *RelayBuffer* will delete the *SYNC* packet firstly, and the re-generated full *SYNC* packet will be sent to the *RelayBuffer* to pass it to the lower *netw* layer. If the received packet is from the higher application layer, the *RelaySlave* will send the packet out to the *RelayBuffer* directly. Finally, if the received packet is from the *RelayBuffer*, *RelaySlave* and *RelayMaster* will check the received packet is for themselves. If yes, the *RelaySlave* and *RelayMaster* will process it, otherwise, the received packet will be discarded. The main tasks of *RelayMaster* module are: (i) It schedules the time synchronization cycle by using *scheduleAt()* of OMNeT++ API in the initialization function. (ii) Once the time synchronization of the previous hop is completed, the *handleMessage()* function of *RelayMaster* activates the synchronization of next hop. The responsibility of *RelayBuffer* module is to transmit the packet to corresponding modules (i.e. *RelaySlave*, *RelayMaster* modules and network layer *netw*). More specifically, when a packet arrives at the module either from the *RelayMaster* or the *RelaySlave*, the *RelayBuffer* will send the packet to the lower network layer. If the received packet is from *netw* module, the *RelayBuffer* will send the packet to both *RelaySlave* and *RelayMaster* modules.

- (2) **Relay Clock Module:** The *Clock* module in the relay node is to simulate the drifting PCO clock realistically. The task of *Clock* is for maintaining the clock state regularly according to the *desynchronization*-based PCO clock model of (12), providing the local time and timestamps to other modules within the sensor node. Moreover, the clock offset and skew are regularly updated according to both (6) and (7). It is worth noting that the *Clock* module can generate the perfect PCO clock time in the master node, by configuring clock offset and skew to zero.

Remark 6.2: The *SYNC* packet is generated by *Clock* module when the clock state P reaches the threshold value, meanwhile, the synchronization mechanism (i.e. interactive mode) is activated by themselves after the *SYNC* is received. Thus, it is unnecessary to schedule the time synchronization cycle in the sensor nodes, and the functions of *RelayMaster* module in relay node haven't been used in this paper. However, the *RelayMaster* module of the relay can be used in the PTP to realize the simulation of synchronization in multi-hop wireless networks.

7. Simulations and results

To verify the classical PCO clock model with the refractory period and evaluate the impacts of parameter variations (e.g. clock stability and coupling strength) on the time synchronization performance, extensive simulations have been done and this section presents these results. In the simulation, two typical drifting clocks presented in (Giorgi & Narduzzi, 2011) are simulated. The skews of the Clock A and B are subject to a random perturbation with standard variances $\sigma_\gamma = 10^{-8}$ and 10^{-9} respectively, and their offset noises are of standard variances $\sigma_\theta = 10^{-6}$ and 10^{-7} respectively. The clock update frequency f_0 is 32.768 kHz to mimic RTC clock of WSNs, and the threshold φ of PCO clock is set to 1s; thus, the time synchronization cycle is 1s. The *SYNC* packet is configured to 21 bytes based on the ZigBee standards (Eady, 2010), (Farahani, 2011). Finally, two kinds of network topologies which are indicated in Figure 5, namely, a single-hop wireless sensor network and a three-hop network topology, are simulated, and the configurations of simulations are summarized in Table 1.

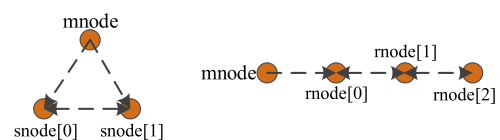


Figure 5. Network topologies (Left: single-hop WSN, Right: multi-hop WSN).

Table 1. Simulation configurations.

Symbol	Value	Unit
f_0	32.768	Kilohertz (kHz)
θ_0	1, 10, 400	Millisecond (ms)
γ_0	[10, 20, 30, 40, 50, ... 100]	Part Per Million (PPM)
σ_θ	$10^{-6}, 10^{-7}$	
σ_γ	$10^{-8}, 10^{-9}$	
δ	0.1, 1, 10	Millisecond (ms)
ε	9, 10, 20, 30, 40, ... 900	Millisecond (ms)
φ	1	Second (s)

7.1. Free-running mode of drifting PCO clock

The impacts of clock's parameter variations (i.e. initial offset and skew and offset and skew noise) in non-identical classical PCO free-running mode are first studied. Figure 6 shows how the offset $\theta[n]$ in the free-running mode varies against different clock's parameter variations. Three different clocks are considered in this simulation: the lower performance one (Clock A) corresponds to a *personal computer clock* (as Clock A in (Giorgi & Narduzzi, 2011)), while the higher performance one (Clock B) presents the same values of σ_θ and σ_γ as the Clock B characterized in (Giorgi & Narduzzi, 2011). The characteristics of Clock C is between those of the former two clocks. Moreover, two kinds of clock frequencies with initial skews γ_0 of 10 PPM and 100 PPM are also simulated.

In the free-running mode, as the time increases, the offset $\theta[n]$ increases from the initial offset $\theta_0 = 1$ ms to a value which depends on the clock operation time and the intrinsic stability of the unregulated clock. This means that, if a drifting PCO clock with 10 PPM of frequency variation is higher performance (like Clock B), the increment of clock offset is less than 2 ms during 90 seconds simulation. Even though the offset's increment of a Clock B

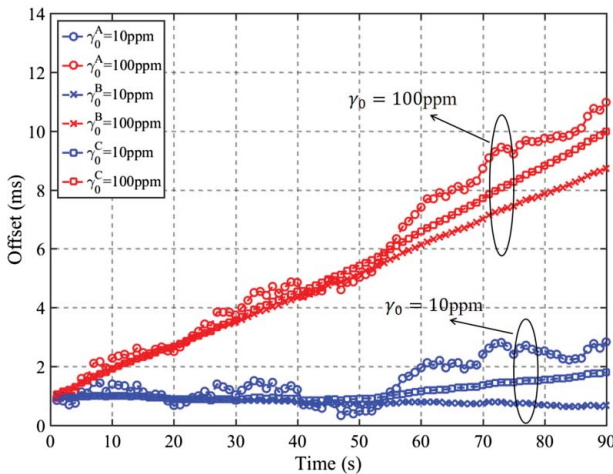


Figure 6. The offset of non-identical and time-varying PCO clock in free-running mode ($\theta_0 = 1$ ms, $\sigma_\theta^A = 10^{-6}$, $\sigma_\gamma^A = 10^{-8}$, $\sigma_\theta^B = 10^{-7}$, $\sigma_\gamma^B = 10^{-9}$, $\sigma_\theta^C = 10^{-7}$, $\sigma_\gamma^C = 10^{-8}$).

with 100 PPM is not greater than 9 ms during the simulation, the offset would be larger when the clock runs in the longer time.

On the other hand, if a drifting and lower performance PCO clock, as Clock A with frequency variation of 100 PPM, is adopted in the WSNs without clock correction method, the clock offset (e.g. 12 ms in 90-second simulation) has a significant negative effect on the performance of WSNs during the complete life cycle of sensor nodes. Therefore, a higher performance drifting clock with less frequency variation (e.g. the Clock B with 10 PPM in the simulator) has a relatively less negative effect on the WSNs performance.

7.2. Synchronization of identical classical PCO clocks in single-hop WSNs

The synchronization performances of identical classical PCO clock with the refractory period in single-hop wireless sensor networks are presented in Figures 7 and 8. Figure 7 demonstrates synchronization of PCO clock with two different θ_0 of -400 ms and 400 ms. For the identical clock with the initial offset of 400 ms, simulation results show the synchronization error Δ increases from initial offset to the maximum synchronization error of $+500$ ms, it is shifted to -500 ms approximately after reaching the maximum synchronization error. This interesting behaviour of PCO results from positive initial offset θ_0 , the excitatory coupling (i.e. $\epsilon > 0$) and the synchronization error Δ (which is defined between -500 ms and $+500$ ms). Since the excitatory coupling is used in the PCO, the clock state P can only be pulled up, rather than pulled down when the SYNC packet is received by the sensor node; moreover, the synchronization error is defined between the -500 ms and $+500$ ms, Δ is shifted

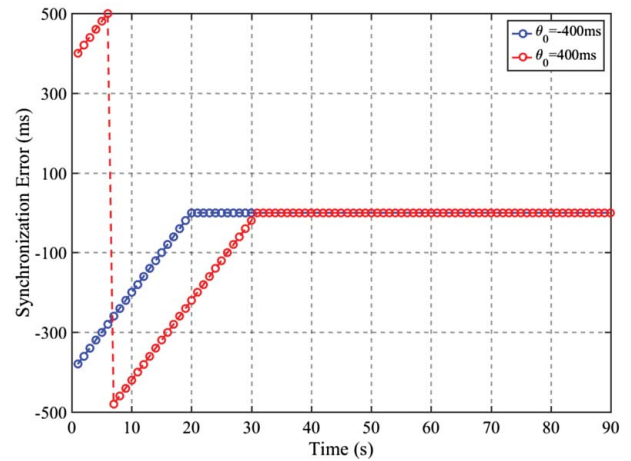


Figure 7. Synchronization of identical classical PCO with the refractory period and different initial offsets in single-hop WSNs ($\delta = 0.1$ ms, $\varepsilon = 20$ ms).

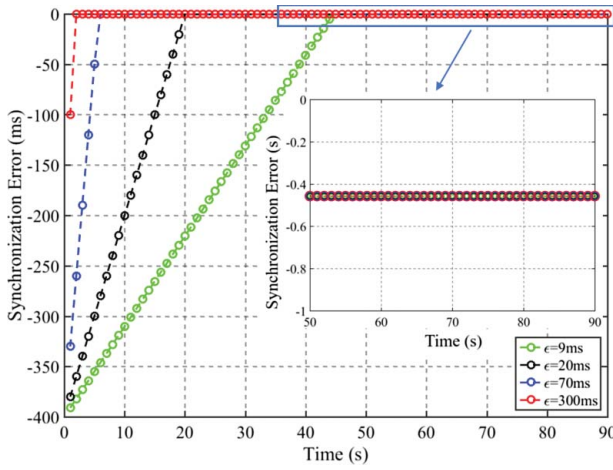


Figure 8. Synchronization of identical classical PCO clock with the refractory period in single-hop WSNs. ($\theta_0 = -400$ ms, $\delta = 0.1$ ms, $\varepsilon = 20$ ms).

to -500 ms once synchronization error of $+500$ ms is reached.

In addition, the synchronization errors of both clocks with different initial offsets convergence to zero approximately after finite time synchronization cycles, and it is obvious that the convergence speed of PCO clock with the negative initial offset (e.g. $\theta_0 = -400$ ms in the simulation) is faster.

Simulation results in Figure 8 show the PCO synchronization precision with different coupling strengths ε . The analysis of synchronization error between the drifting clock (i.e. slave clock) and the *reference clock* (i.e. master clock) indicates that, the synchronization error of identical classical PCO clock convergences to -0.458 ms during the simulation. Since the propagation delay for 50 m (the distance between two coupled sensor nodes) is 0.016 ms, it is able to be neglected compared with the clock update period $\tau_0 = 0.0305$ ms, and the transmission delay for the reception of a complete SYNC packet is 0.48 ms. Thus, the synchronization error is not greater than the delay (i.e. -0.48 ms ≤ -0.458 ms ≤ 0.48 ms), and slave clocks in the single-hop WSN eventually are synchronized to the master clock and time synchronization is achieved in the network. Both (Mathar & Mattfeldt, 1996) and (Degesys, Basu, & Redi, 2008) proved that higher convergence speed is achieved in the PCO with higher coupling strength, when the coupling strength is less than the specified critical coupling strength. Furthermore, simulation results also show that the synchronization precision is independent of the coupling strength ε , and same synchronization precisions of PCO clocks with different coupling strengths are attained in the network.

It is worth noting that the transmission delay of 0.48 ms exists in the network. This means that, if the master transmits a SYNC packet to a slave node, the slave receives

the SYNC packet and adjusts its drifting clock after 0.48 ms. Therefore, drifting slave clock is always lagged by 0.48 ms, and the synchronization error Δ between the slave and master clocks is -0.48 ms theoretically. By compensating the SYNC packet transmission delay in the clock correction method, the synchronization error may reduce to -0.022 ms which is less than one clock updated period.

7.3. Synchronization of identical classical PCO in multi-hop WSNs

Figure 9 shows the achieved synchronization of identical classical PCO clock in a multi-hop wireless sensor network. Due to the existing of transmission delay in the network, the synchronization error is accumulated with the increase of hop distance (i.e. $\text{rnode}[0].\Delta = -0.458$ ms, $\text{rnode}[1].\Delta = -0.916$ ms, $\text{rnode}[2].\Delta = -1.373$ ms). In addition, even though the initial offset of sensor nodes varies (i.e. blue and red lines in Figure 9), the achieved synchronization accuracy is same. This means the initial offset has no effect on the accuracy of PCO time synchronization in multi-hop wireless sensor networks.

7.4. Evaluation of non-identical classical PCO clocks synchronization in single-hop WSNs

The synchronization performance is analyzed for two non-identical classical PCO clocks (Clock A and B) and for different values of frequency variation, 10 PPM and 100 PPM, as reported in the legend of Figure 10. Since the initial offset θ_0 is less than the coupling strength ε , all the local clocks are adjusted and synchronized to the master clock based on the reception of SYNC packet at the 1st synchronization cycle. It coincides with (An et al.,

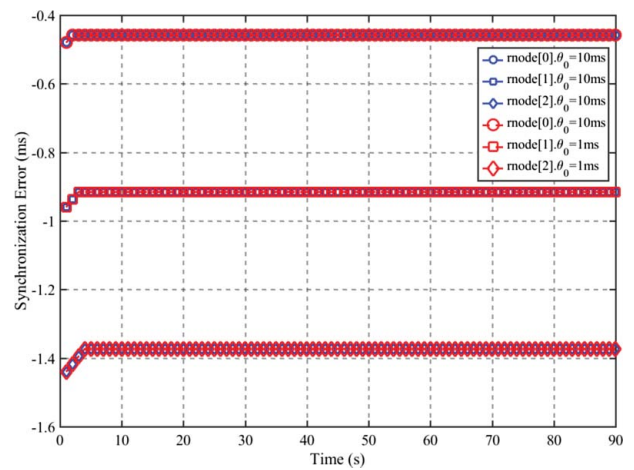


Figure 9. Synchronization error of identical PCO clock with the refractory period in multi-hop WSNs ($\theta_0 = 1$ ms, 10 ms, $\delta = 1$ ms, $\varepsilon = 20$ ms).

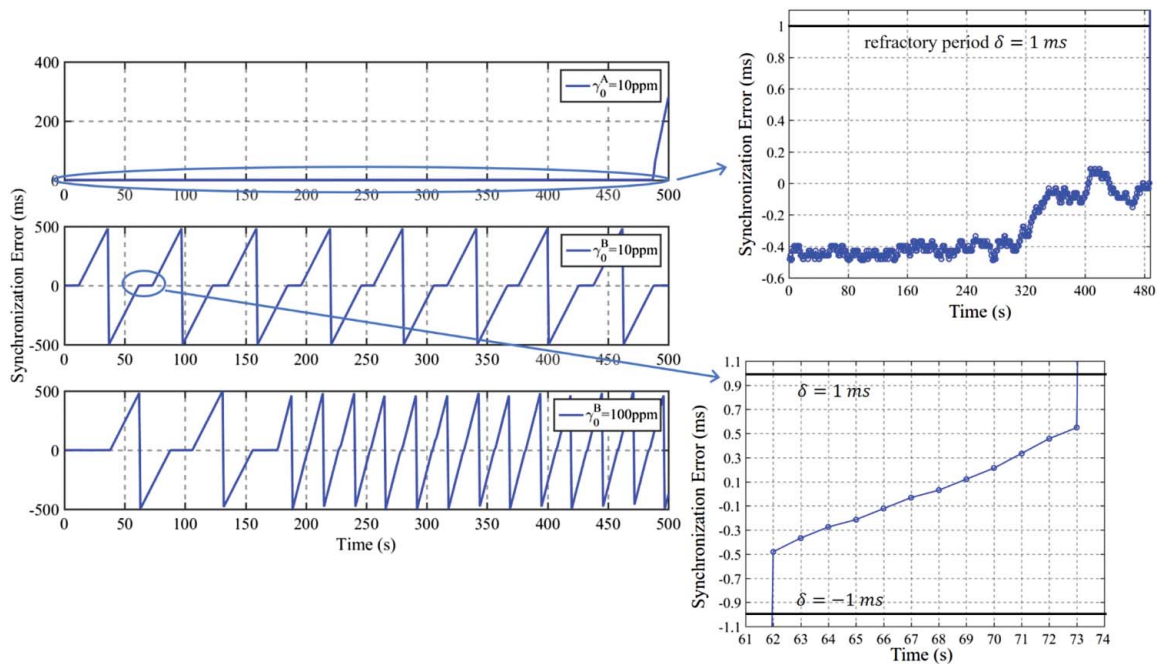


Figure 10. Synchronization error of non-identical and time-varying classical PCO clock with the refractory period in single-hop WSNs. ($\theta_0 = 10$ ms, $\sigma_\theta^A = 10^{-6}$, $\sigma_\gamma^A = 10^{-8}$, $\sigma_\theta^B = 10^{-7}$, $\sigma_\gamma^B = 10^{-9}$, $\delta = 1$ ms, $\varepsilon = 20$ ms).

2011) that due to the non-identical and time-varying PCO clock, the achieved synchronization will lose gradually, and sensor nodes will move into interactive mode again (see Figure 10).

To be specified, when the synchronization error is less than the refractory period (i.e. $-\delta < \Delta < \delta$), the drifting clock will stay in the free-running mode; otherwise, the PCO clock will move to the interactive mode to correct its drifting clock based on the reception of SYNC packet from master node. This means that, as shown in Figure 10, for a Clock B with 10 PPM, it can maintain synchronization for a longer time (e.g. 480 s in the simulation). On the other hand, if a lower performance and drifting clock with 100 PPM is adopted, the clock will maintain synchronization for a short time (e.g. 10 s in 500-second simulation).

Therefore, the classical PCO with the refractory period cannot be implemented to the realistic WSNs directly, and the classical PCO need be improved to ensure it can be implemented in the real-world environments.

8. Conclusion

In this paper, an open source simulator is developed for PCO in IEEE 802.15.4-based WSNs. The concept and structure of the developed simulator are detailed, followed by the development of a realistic model of drifting classical PCO clock (implemented with the refractory period) with configurable phase, frequency noises, adjustable and higher resolution, and TS superframe which is compatible with IEEE 802.15.4 superframe. The

detailed implementation of the drifting PCO clock, FFD node that benefit to simulate the multi-node and multi-hop WSNs are also presented to demonstrate the features of the simulator.

In particular, the impacts on the synchronization performance of classical PCO are examined and analyzed by simulation. Even though the identical classical PCO without drifting frequency is applied to the realistic single-hop wireless sensor networks, the full synchronization cannot be achieved. Due to the delay (i.e. transmission delay in this paper) in the realistic WSNs, synchronization error is accumulated with the increase of hop distance, when the classical identical PCO is applied to the multi-hop WSNs. Finally, resulting from the non-identical and time-varying PCO clock in practice, the achieved PCO synchronization will lose gradually. The classical PCO with refractory period, therefore, cannot be implemented to the real-world WSNs directly, and it need be improved to ensure the implementation of PCO in sensor networks.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by Northumbria University at Newcastle via a postgraduate research studentship.

ORCID

Yan Zong  <http://orcid.org/0000-0002-5196-8025>

References

- An, Z., Zhu, H., Li, X. [Xinrong], Xu, C., Xu, Y., & Li, X. [Xiaowei]. (2011). Nonidentical linear pulse-coupled oscillators model with application to time synchronization in wireless sensor networks. *IEEE Transactions on Industrial Electronics*, 58(6), 2205–2215.
- Bojic, I., & Nymoen, K. (2015, October). Survey on synchronization mechanisms in machine-to-machine systems. *Engineering Applications of Artificial Intelligence*, 45, 361–375.
- Correll, K., Barendt, N., & Branicky, M. (2005). Design considerations for software only implementations of the IEEE 1588 precision time protocol. *Proceedings of conference on IEEE 1588 standard for a precision clock synchronization protocol for networked measurement and control systems*.
- Degeys, J., Basu, P., & Redi, J. (2008, March). Synchronization of strongly pulse-coupled oscillators with refractory periods and random medium access. *Proceedings of 2008 ACM symposium on applied computing*.
- Degeys, J., Rose, I., Patel, A., & Nagpal, R. (2007, April). DESYNC: Self-organizing desynchronization and TDMA on wireless sensor networks. *Proceedings of 6th international symposium on information processing in sensor networks* (pp. 11–20).
- Eady, F. (2010). *Hands-On ZigBee: Implementing 802.15.4 with microcontrollers*. Oxford: Newnes.
- Elson, J., Girod, L., & Estrin, D. (2002, Winter). Fine-grained network time synchronization using reference broadcasts. *Proceedings of the 5th symposium on operating systems design and implementation* (pp. 147–163).
- Ernst, U., Pawelzik, K., & Geisel, T. (1995). Synchronization induced by temporal delays in pulse-coupled oscillators. *Physical Review Letters*, 74(9), 1570–1573.
- Farahani, S. (2011). *Zigbee wireless networks and transceivers*. Oxford: Newnes.
- Ferrari, F., Meier, A., & Thiele, L. (2010, March). Accurate clock models for simulating wireless sensor networks. *Proceedings of the 3rd international ICST conference on simulation tools and techniques*.
- Franco, A. (2015). Lab 1: TDMA. Retrieved from <http://omikron.eit.lth.se/ETSN01/ETSN01/labs/>
- Giorgi, G., & Narduzzi, C. (2011). Performance analysis of Kalman-Filter-based clock synchronization in IEEE 1588 networks. *IEEE Transactions on Instrumentation and Measurement*, 60(8), 2902–2909.
- Huang, Y., Li, T., Dai, X., Wang, H., & Yang, Y. (2015). TS2: A realistic IEEE1588 time-synchronization simulator for mobile wireless sensor networks. *SIMULATION*, 91(2), 164–180.
- IEEE Instrumentation and Measurement Society. (2008, July). IEEE standard for a precision clock synchronization protocol for networked measurement and control systems. *IEEE Standard 1588-2008 (Revision of IEEE Standard 1588-2002)*, 1–300.
- Khan, M. A., Hasbullah, H., & Nazir, B. (2014, September). Recent open source wireless sensor network supporting simulators: A performance comparison. *Proceedings of 2014 international conference on computer, communications, and control technology* (pp. 324–328).
- Kopke, A., Swigulski, M., Wessel, K., Willkomm, D., Klein Han-evel, P. T., Parker, T. E. V., . . . Valentin, S. (2008). *Simulating wireless and mobile networks in OMNeT++ the MiXiM vision. Proceedings of the 1st international conference on simulation tools and techniques for communications, networks and systems & workshops*.
- Kucuk, K., Bandirmali, N., Kavak, A., & Atmaca, S. (2013). A modified sectoral sweeper-based localization estimation and its implementation in a multi-hop wireless sensor networking environment by using OPNET. *SIMULATION*, 89(6), 746–761.
- Maroti, M., Kusy, B., Simon, G., & Ledeczi, A. (2004, November). The flooding time synchronization protocol. *Proceedings of the 2nd international conference on embedded networked sensor systems* (pp. 39–49).
- Mathar, R., & Mattfeldt, J. (1996). Pulse-Coupled decentral synchronization. *SIAM Journal on Applied Mathematics*, 56(4), 1094–1106.
- Mirollo, R., & Strogatz, S. (1990). Synchronization of pulse-coupled biological oscillators. *SIAM Journal on Applied Mathematics*, 50(6), 1645–1662.
- Mohammadpour, S., & Binazadeh, T. (2018, January). Robust finite-time synchronization of uncertain chaotic systems: Application on Duffing-Holmes system and chaos gyros. *Systems Science and Control Engineering*, 6(1), 28–36.
- NS3 official website. (2018). Retrieved from <https://www.nsnam.org>.
- OMNeT++ release 4.6. (2018). Retrieved from <https://www.omnetpp.org>.
- Petrioli, C., Petrocchia, R., Potter, J., & Spaccini, D. (2015, November). The SUNSET framework for simulation, emulation and at-sea testing of underwater wireless sensor networks. *Ad Hoc Networks*, 34, 224–238.
- Sharif, M., & Sadeghi-Niaraki, A. (2017, September). Ubiquitous sensor network simulation and emulation environments: A survey. *Journal of Network and Computer Applications*, 93, 150–181.
- Shed, S. (2017, January 03). Amazon now has 45,000 robots in its warehouses. *Business Insider*. Retrieved from <http://uk.businessinsider.com>
- Tyrrell, A., Auer, G., & Bettstetter, C. (2008). On the accuracy of firefly synchronization with delays. *Proceedings of first international symposium on applied sciences on biomedical and communication technologies*.
- Tyrrell, A., Auer, G., & Bettstetter, C. (2010). Emergent slot synchronization in wireless networks. *IEEE Transactions on Mobile Computing*, 9(5), 719–732.
- Xian, X., Shi, W., & Huang, H. (2008). Comparison of OMNeT++ and other simulator for WSN simulation. *Proceedings of 3rd IEEE conference on industrial electronics and applications* (pp. 1439–1443).
- Zong, Y., Dai, X., & Gao, Z. (2017, September). A software simulator of discrete pulse-coupled oscillators (PCO) time synchronization in wireless sensor networks. *Proceedings of 23rd international conference on automation and computing (ICAC)*.