Northumbria
University
NEWCASTLE

Proportional-Integral Synchronisation for Non-identical Wireless Packet-Coupled Oscillators with Delays

Yan Zong, *Member*, Xuewu Dai, *Senior Member*, and Zhiwei Gao, *Senior Member*

***Abstract*—Precise timing among wireless sensor nodes is a key enabling technology for time-sensitive industrial Wireless Sensor Networks (WSNs). However, the accuracy of timing is degraded by manufacturing tolerance, ageing of crystal oscillators, and communication delays. This paper develops a framework of Packet-Coupled Oscillators (PkCOs) to characterise the dynamics of communication and time synchronisation of clocks in WSNs. The non-identical clock is derived to describe the embedded clock's behaviour accurately. A Proportional-Integral (PI) packet coupling scheme is proposed for synchronising networked embedded clocks, while, scheduling wireless *Sync* packets to different slots for transmission. It also possesses the feature of automatically eliminating the effects of unknown processing delay, which further improves the synchronisation performance. The rigorous theoretical analysis of PI-based PkCOs is presented via studying a closed-loop time synchronisation system. The performance of PI-based PkCOs is evaluated on a hardware testbed of IEEE 802.15.4 WSN. The experimental results show that the precision of the proportional-integral PkCOs protocol is as high as 60μs (i.e. 2 ticks) for 32.768kHz crystal oscillator-based clocks.**

***Index Terms*—Time synchronisation, packet-coupled oscillators, wireless sensor networks.**

I. Introduction

NOwadays, with the ever-growing developments in micro-electro-mechanical systems (MEMS) technology, wireless communication and digital electronics, the manufacturing of low-cost and low-power tiny sensor nodes becomes feasible. These sensor nodes are usually deployed in an area to form a network, which is called the Wireless Sensor Network (WSN), in order to monitor and collect environmental information, and realise different applications. In WSNs, thanks to its trade-off between high-quality signal accuracy and cost, the crystal oscillator is widely chosen as the clock source [1]. However, it fails to produce the same frequency, owing to internal factors (e.g. manufacturing tolerance [2]) and external environmental conditions, such as temperature [3] and supply voltage, and a common sense of timing does not exist in the network without a management protocol. As a result, a Time Synchronisation (TS) protocol on the application layer is required to provide a common notion of time in WSNs.

In nature, synchronisation is one of the most common and captivating phenomena (e.g. [4], [5], [6]). In 1975, [7] mathematically described the self-synchronisation behaviour of cardiac pacemaker through Peskin's model. Fifteen years later, inspired by Peskin's model, Mirollo and Strogatz proposed the Pulse-Coupled Oscillators (PCO) model [8] to study a more general version of Psekin's model. In [8], it is proved that under almost all initial conditions, N identical oscillators in a fully-connected network are synchronised by using the excitatory coupling scheme. Since then, the PCO model has attracted a lot of attention in mathematics and physics communities, and has also offered insights on a number of phenomena (e.g. synchronous flashing of fireflies [9]).

In the PCO model, the oscillator works in either free-running mode or interactive mode. In the free-running mode, each oscillator behaves as an isolated oscillator, whose state P rises toward the threshold monotonically and concavely. Once P reaches the threshold, the oscillator fires (which means a *Pulse* being broadcasted for synchronisation purposes), and P is instantly reset to zero, after which the cycle repeats. When an oscillator receives a *Pulse* from another oscillator, it moves into the interactive mode, where the state P is increased by a constant coupling strength $\epsilon > 0$ (i.e. the excitatory coupling scheme), after which it returns to the free-running mode. Finally, all the oscillators simultaneously broadcast *Pulse*s, and the network achieves synchronisation.

Due to remarkably inherent simplicity of the coupling mechanism, and achieving synchronisation and communication in an intertwined and inseparable fashion [10], the PCO model is particularly suitable for low-cost and resource-constrained WSNs. In practice, the PCO's firing-resetting behaviour is similar to the periodic resetting feature of the clock module in embedded systems, where the counter register of each clock driven by a crystal oscillator, increases periodically from zero to the threshold. Once the value in the counter register matches the threshold, it is reset to zero; meanwhile, a pulse (i.e. an interrupt signal) is emitted and sent to the processor for triggering an interrupt.

Inspired by the equivalent relationship between the pulse-coupled oscillator and embedded clock, this paper studies a more realistic version of the PCO model, namely, Packet-Coupled Oscillators (PkCOs), in order to characterise the dynamics of communication and synchronisation of clocks

Manuscript received October 16, 2019; revised May 13, 2020 and August 11, 2020; accepted October 20, 2020. This work was supported by the University of Northumbria at Newcastle *via* a postgraduate research studentship. *(Corresponding author: Xuewu Dai.)*

The authors are with the Department of Mathematics, Physics and Electrical Engineering, Northumbria University, Newcastle upon Tyne, United Kingdom (e-mail: {yan.zong, xuewu.dai, zhiwei.gao}@northumbria.ac.uk).

in WSNs. Instead of adopting the identical oscillator, the non-identical oscillator is used to describe the real clock's behaviour. The wireless *Sync* packet is utilised for synchronisation purposes, rather than the physical *Pulse* signal; all the *Sync*s are allocated to different time slots for packet transmission. Moreover, this work adjusts the local clock through the clock offset estimate obtained from the timestamp.

A. Related Work

1) Non-identical Clock: As a consequence of internal factors and external environmental conditions, the crystal oscillator is subject to variations in phase and frequency. This means that the real clock's time information differs from that of a reference clock. However, until now, the identical clock with a same and non-drifting frequency is still used for studying the PCO synchronisation in WSNs [11], [12], [13], [14].

Typically, the accuracy of the quartz crystal oscillator is less than 100 Parts Per Million (ppm) [15], or even better, around 5 ppm [1]. Thus, it is reasonable to model the crystal oscillator-based clock as a non-identical clock (i.e. the frequency is different, but constant). Even though [16] and [17] propose a non-identical clock, both of which lack taking the clock's periodic resetting behaviour into account. In this paper, we derive a non-identical state-space model for describing the embedded clock with a periodic resetting feature, which is also a state equation in the state-space representation of a closed-loop time synchronisation system.

2) Proportional-Integral Packet-Coupling Scheme: In PCO synchronisation, the pulse coupling mechanism can be classified into two types, namely, an excitatory (or inhibitory) coupling algorithm (e.g. [8], [16], [18]) and a phase response function-based coupling scheme (e.g. [12], [14], [19]). Practically, due to the limitation of Radio Frequency (RF) communication, the packet exchange delay appears in transmitting and receiving packets, and wireless nodes need time to execute several operations (e.g. assembling packets, media access, and so on). However, once the packet exchange delay is introduced to the PCO model of the works cited above, infinite *Pulse*s are transmitted in the network, which leads to the failure of synchronisation in WSNs [20]. Thus, several works (e.g. [16], [21]) adopt the refractory period, where the reception of *Pulse*s has no effect on the local oscillator's state. Even though the implementation of a refractory period can let the network realise clock synchronisation, the packet exchange delay cannot be compensated. Furthermore, the performance of these synchronisation algorithms is challenging to analyse rigorously. Thus, instead of using existing coupling strategies, this work adopts the measured offset, obtained from the timestamp, to correct the local clock. We also model the procedure of packet exchange as the measurement equation of a closed-loop synchronisation system.

Owing to the difficulty of real-time computing on resource-constrained WSN nodes, and the impossibility of real-time counter register access resulting from the complex microprocessor architecture [22], the processing delay occurs when the processor performs the calculation and accesses the register (i.e. the employment of correction value to the counter register). In [1], the experimental results indicate that the utilisation of a Proportional (P) controller with a delay compensation strategy can eliminate the effects of processing delay. However, this delay is difficult to measure in industrial systems, and also varies in different hardware environments. In this work, the Proportional-Integral (PI) controller is adopted to cancel the impacts of processing delay automatically. Thus, an alternative solution, which is a closed-loop network system consisting of state and measurement equations and a PI controller, is proposed to study the theoretical performance of PkCOs in WSNs.

3) Desynchronisation: If the PCO model is directly applied to WSNs, all the synchronised nodes simultaneously send *Sync* packets to the same channel, when all the clocks achieve time synchronisation. Obviously, transmitted *Sync*s interfere with each other, and no packets can be received successfully. To avoid the occurrence of packet collision caused by concurrent transmission, [23] proposes desynchronisation (DESYNC) to let *Sync*s be sent in a uniformly distributed fashion, which means that *Sync* packets are scheduled and allocated to different time slots for transmission, thereby minimising the possibility of collision.

In the literature (e.g. [11] – [14], [23]), once the scheduling of *Sync*s is achieved, N WSN nodes send the control traffic (i.e. *Sync* packets) with interval space T/N during each time synchronisation cycle T. This means that they provide no solution for separating the control traffic from the data stream transmitted on the same wireless channel. Although [18] proposes a protocol to allow natural separation of the control traffic from the data stream, the dithered quantisation function, used for realising the scheduling of *Sync*s, is not integrated to the sensor nodes. The processor of resource-constrained nodes has to realise the dithered quantisation function through a software floating-point calculation. However, the software floating-point unit may reduce the calculation precision. This paper proposes an easy-to-implement scheme for realising the scheduling of *Sync*s, which can naturally separate the data stream from the control traffic.

B. Contribution and Paper Organisation

In this work, we propose a framework of PI-based PkCOs to characterise the dynamics of communication and synchronisation of embedded clocks in WSNs. A non-identical state-space clock is derived to describe the clock's behaviour accurately, which is the state equation of a closed-loop system. The procedure of packet exchange in PkCOs is modelled as the measurement equation, and the effects of packet exchange and processing delays on synchronisation precision are also analysed. The state and measurement equations and PI controller constitute a closed-loop time synchronisation system for theoretical analysis. The stability condition of the PkCOs algorithm is proved. The convergence analysis indicates that the impacts of unknown processing delay are fully eliminated, leading to better precision. In addition to scheduling *Sync* packets to different time slots for transmission, the proportional-integral packet-coupling mechanism also provides a solution to automatically separate the control traffic from the data stream transmitted on the same wireless channel. The experimental

results on an IEEE 802.15.4 hardware testbed demonstrate that the PI-based PkCOs protocol retains synchronisation with the precision of around $60\mu s$ (i.e. 2 ticks) in the one-hour test.

The rest of this paper is organised as follows: the non-identical clock is derived in Section 2. Then, Section 3 presents the PI-based packet coupling scheme, and the effects of packet exchange and processing delay on synchronisation performance are studied. In addition, the rigorous theoretical analysis of the algorithm's stability and convergence are also provided. The simulation and experimental results are given in Section 4 and Section 5, respectively. Finally, Section 6 shows the conclusions.

II. Clock Model

As a result of the equivalent relationship between the embedded clock and packet-coupled oscillator, the crystal oscillator-based clock is described as the free-running oscillator of PkCOs in the following section. A non-identical clock model is considered in this work since each crystal oscillator provides a signal with a different frequency for the clock module. To understand how the embedded clock is modelled, the process by which their equations are derived is briefly presented.

A. A Non-identical Clock

Referring first to the case of a perfect clock, in embedded systems, the clock module is constructed from two parts: (i) a crystal oscillator, ticking at the nominal frequency $f_0 = 1/\tau_0$ where τ_0 is the crystal oscillator period, and (ii) a counter register counts the number of ticks generated by a crystal oscillator. Through the process of counting, the periodic signal produced by a crystal oscillator is converted into an integer that is increased by one per crystal oscillator period. Once the cumulative value of the counter register matches the predefined threshold, it is reset and starts counting from zero again; meanwhile, a pulse interrupt signal is sent to the processor for triggering an event, for example, in this paper, a *Sync* packet being transmitted for the clock synchronisation purpose. Let $t[n]$ denote the time reported by such a crystal oscillator clock at the n-th clock event, $t[n]$ is calculated as

$$t[n] = n\tau_0 = \frac{n}{f_0}. \tag{1}$$

For an ideal crystal oscillator-based clock whose frequency is the same as the nominal frequency f_0, $t[n]$ is accurate and referred to as the *reference time*. Such a perfect embedded clock is also called the *reference clock* or *master clock* thereafter, and the corresponding node is referred to as the *master node*.

In the PkCOs synchronisation method, the clock threshold φ_0 (which is equal to the time synchronisation cycle T) is much greater than the clock period τ_0 (i.e. $\varphi_0 \gg \tau_0$), it is reasonably assumed that the clock is updated m_0 times during a single synchronisation cycle, following $T = m_0\tau_0$. Taking the clock's periodic resetting behaviour into account, the dynamics of $P_0[n]$ are

$$P_0[n] = t[n] - \sum_{h=0}^{k} \varphi_0[h], \tag{2}$$

where φ_0 is the master clock threshold, $k = \lfloor n/m_0 \rfloor$ represents how many clock resettings have occurred from $n = 0$ to the n-th clock event, where floor function $\lfloor n/m_0 \rfloor$ denotes the greatest integer less than or equal to n/m_0. Recalling that the synchronising of clocks occurs when a clock fires, k also denotes the number of synchronisation cycles so far. In other words, the clock is at the k-th synchronisation cycle. In addition, from the viewpoint of embedded systems, n is an integer value in the counter register, and the event of the counter reaching n is referred to as the n-th clock event.

Practically, due to internal factors and environmental conditions, the crystal oscillator-based clock on each *sensor node* cannot keep its clock state $P_i[n]$ the same as $P_0[n]$, and $P_i[n]$ at the n-th event is expressed as

$$P_i[n] = t[n] + \frac{\sum_{h=0}^{n} \chi_i[h]\tau_0}{f_0} + \frac{\phi_i[n]}{2\pi f_0} - \sum_{h=0}^{k} \varphi_i[h], \tag{3}$$

where φ_i is the i-th sensor node clock threshold, $\phi_i[n]$ is the random process representing all instant phase fluctuations from $t[0]$ to $t[n]$. χ_i is the deviation of the clock frequency from its nominal value f_0 (i.e. $\chi_i = f_i - f_0$), whose accumulated effects over time are phase fluctuations $\sum_{h=0}^{n} \chi_i[h]\tau_0$.

B. State-space Model of A Non-identical Clock

Let clock offset $\theta_i[n]$ denote the difference between clock state $P_i[n]$ reported by the i-th unperfect clock and the ideal clock's state $P_0[n]$, yields

$$\theta_i[n] = P_i[n] - P_0[n]. \tag{4}$$

The offset $\theta_i[n]$ of the i-th clock is obtained by substituting (2) and (3) into (4)

$$\theta_i[n] = \frac{\sum_{h=0}^{n} \chi_i[h]\tau_0}{f_0} + \frac{\phi_i[n]}{2\pi f_0} - \sum_{h=0}^{k} \Delta\varphi_i[h], \tag{5}$$

where $\Delta\varphi_i = \varphi_i - \varphi_0$. It can be seen that the clock offset is the result of three contributing sources, namely, an accumulated phase fluctuation $(\sum_{h=0}^{n} \chi_i[h]\tau_0)/f_0$ owing to the frequency deviation χ_i, random phase noise $\phi_i[n]/2\pi f_0$, and joint effects of the clock's resetting behaviour and non-identical frequency $\sum_{h=0}^{k} \Delta\varphi_i[h]$.

By introducing skew $\gamma_i = (f_i - f_0)/f_0$, which is the normalised difference between the local clock frequency f_i and f_0, the recursive form of clock offset can be calculated from (5):

$$\theta_i[n+1] = \begin{cases} \theta_i[n] + \gamma_i\tau_0 + \overline{\omega}_{\theta_i}[n] & \text{if } \frac{n+1}{m_i} \notin \mathbb{N}^+ \\ \theta_i[n] + \gamma_i\tau_0 + \overline{\omega}_{\theta_i}[n] - \Delta\varphi_i & \text{if } \frac{n+1}{m_i} \in \mathbb{N}^+ \end{cases}, \tag{6}$$

where $\overline{\omega}_{\theta_i}[n] = (\phi_i[n+1] - \phi_i[n])/2\pi f_0$ means the Gaussian random noise process [24], [25]. $\mathbb{N}^+$ is the set of positive integers $\{1, 2, 3, ...\}$. In this work, m_i of the i-th non-identical clock is equal to m_0, and $m = m_i = m_0$.

From the perspective of control theory, the non-identical clock model (6) is a switching system. In order to avoid the difficulty of stability analysis resulting from the model's switching characteristic, a concise state-space clock model is considered, since it provides a simpler approach when

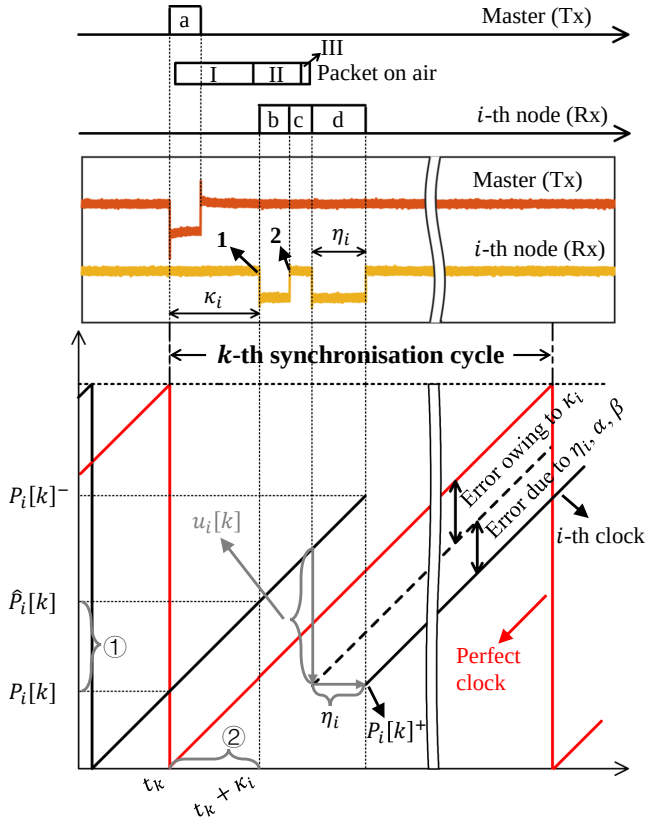


Fig. 1. Example of the packet exchange delay and processing delay ((a): *Sync* packet transmission, (b): COUNT register read access, (c): clock offset calculation, (d): COUNT register read access, clock state calculation and COUNT register write access; I: physical layer header and MAC layer header including PAN ID, source address and destination address, II: MAC payload (i.e. *Sync*), III: frame check sequence; ①: value of clock state increment within packet exchange delay duration, ②: value of packet exchange delay.)[1].

controlling techniques are employed for theoretical analysis. During each synchronisation cycle (i.e. the clock threshold), there exist m clock update events. Hence, at the (km)-th clock update event, the clock equation is

$$\theta_i[(k+1)m] = \theta_i[km] + \gamma_i T - \Delta\varphi_i + L\Omega_i[km], \quad (7)$$

where $L = [1, 1, ..., 1] \in \mathbb{R}^{1\times m}$ is the process noise transition row matrix, $\Omega_i[km] = [\overline{\omega}_i[km], \overline{\omega}_i[km+1], ..., \overline{\omega}_i[(k+1)m-1]]^T \in \mathbb{R}^{m\times 1}$ is the process noise vector.

To simplify the notation, let $\theta_i[k] = \theta_i[km]$ and $\omega_i[k] = L\Omega_i[km]$. Finally, the concise non-identical state-space clock, which is a state equation in the state-space representation of a closed-loop time synchronisation system (28), is given by

$$\theta_i[k+1] = \theta_i[k] + \gamma_i T - \Delta\varphi_i + \omega_i[k]. \quad (8)$$

[1]In the experiments, the i-th node receives the *Sync* at point 1 and finishes reading the clock register at point 2. The experimental results show that the procedure c (i.e. the clock offset calculation) has no effects on synchronisation performance. In addition, due to reading register twice in (b) and (d) (see (9) and (14)), the register read access has no significant impacts on synchronisation accuracy. For the purpose of modelling and analysis, in this paper, we ignore the procedure of clock offset calculation, and treat the COUNT register read access duration as a part of the processing delay. This means that the timestamp is generated at point 1.

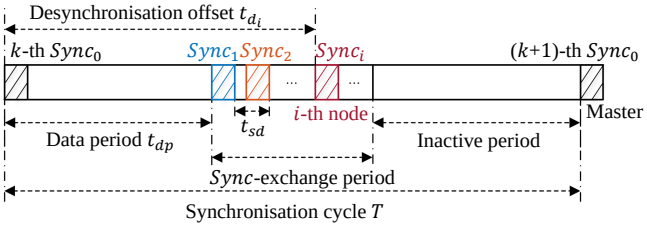


Fig. 2. Proposed desynchronisation superframe ($Sync_0$ is from the master, and $Sync_i$ is from the i-th sensor node).

III. Packet Coupling Scheme

In the PkCOs protocol, the packet coupling scheme provides solutions for estimating the offset of the i-th sensor node clock and for correcting its local clock. In Fig. 1, at the k-th synchronisation cycle, the master transmits the k-th *Sync* packet to the sensor nodes within its transmission range. On the reception of master's *Sync* after the packet exchange delay κ_i, the i-th node associates to its timestamp $\hat{P}_i[k]$ via reading the clock's counter register. Due to the difficulty of real-time computing and the complexity of processor architecture, there always exists a processing delay η_i for the sensor node to determine its offset estimate $\hat{\theta}_i[k]$ from $\hat{P}_i[k]$, and to correct the local clock.

A. Proportional-Integral Packet Coupling Algorithm

To be specific, in Fig. 1, the perfect clock resets, and a *Sync* packet is sent at the time t_k, which also indicates the start of the k-th synchronisation cycle. The i-th sensor node receives the *Sync* at $t_k + \kappa_i$, due to the existence of the packet exchange delay. Once the i-th wireless node receives the *Sync*, it generates a timestamp $\hat{P}_i[k]$, following

$$\hat{P}_i[k] = P_i^{t_k+\kappa_i}, \quad (9)$$

where the packet exchange delay κ_i is the Gaussian random process with the non-zero mean of $\bar{\kappa}_i$ and the finite variance of $\sigma^2_{\kappa_i}$; that is $\kappa_i \sim (\bar{\kappa}, \sigma^2_{\kappa_i})$.

To avoid packet collision resulting from concurrent packet transmission in a synchronised network, this paper proposes a superframe, as shown in Fig. 2, for realising the scheduling of *Sync* packets. The DESYNC superframe consists of three types of periods, namely - data period (DP), *Sync*-exchange period (SP) and inactive period. The DP transmits the data stream via either the Carrier-Sense Multiple Access (CSMA) or Time-Division Multiple Access (TDMA) mechanisms. Through the TDMA scheme, SP guarantees a time slot t_{d_i} for the i-th node to access the wireless channel, in order to send a *Sync* packet with low-latency. The proposed superframe has an inactive period, which allows a WSN node to enter sleep mode for conserving energy. Thus, the desynchronisation offset t_{d_i} is

$$t_{d_i} = \begin{cases} 0, & i = 0 \text{ (master node)} \\ t_{dp} + (i-1)t_{sd}, & i \geq 1 \text{ (sensor node)} \end{cases}, \quad (10)$$

where t_{dp} is the DP's duration, and t_{sd} represents the slot duration, which is the unit time allocated for a wireless sensor node to transmit its *Sync* packets.

At the k-th synchronisation instant, using the local timestamp, the i-th node can estimate its offset $\hat{\theta}_i[k]$, namely,

$$\hat{\theta}_i[k] = \theta_i^{t_k+\kappa_i}, \tag{11}$$

from the master.

In [26], the utilisation of the proportional controller (i.e. $u_i[k] = \alpha(t_{d_i} - \hat{\theta}_i[k])$) can retain the stability of synchronisation; while it fails to remove the effects of processing delay. [1] introduces a delay compensation strategy to P-based PkCOs for effectively cancelling the impacts of processing delay; however, this delay is difficult to measure in industrial applications. In order to improve the PkCOs performance, this paper uses the PI controller to automatically compensate for the unknown processing delay, yielding

$$\begin{cases} w_i[k+1] = w_i[k] + \beta\left((-t_{d_i}) - \hat{\theta}_i[k] + \bar{\kappa}_i\right) \\ u_i[k] = w_i[k] + \alpha\left((-t_{d_i}) - \hat{\theta}_i[k] + \bar{\kappa}_i\right) \end{cases}, \tag{12}$$

where $w_i[k]$ is the integral controller. α and β are gains of proportional and integral controllers, respectively. The control input $u_i[k]$ is equivalent to the coupling strength in the PCO model, which is a constant amount ϵ. However, in this work, thanks to the utilisation of a PI controller, the coupling strength is a dynamic adaptive correction input $u_i[k]$, from the perspective of the PCO model. In order to realise the scheduling of *Sync* packets, the comparison between t_{d_i} and offset estimate $\hat{\theta}_i[k]$ (i.e. $((-t_{d_i})-\hat{\theta}_i[k])$) is fed to the PI controller, rather than $(0 - \hat{\theta}_i[k])$. Once the network achieves clock synchronisation, the i-th wireless node fires and transmits *Sync* packets at the allocated time slot t_{d_i}. This means that when the clock synchronisation is achieved, the i-th clock offset $\theta_i[n]$ converges to $-t_{d_i}$ to realise the scheduling of *Sync* packets, rather than zero.

Ideally, the clock correction input $u_i[k]$ is applied to a drifting clock at $t_k + \kappa_i$, and the clock state $P_i[k]^+$ after it is corrected is given by

$$P_i[k]^+ = P_i^{t_k+\kappa_i} - u_i[k]. \tag{13}$$

However, owing to the impossibility of real-time computing and real-time counter register access, the processing delay η_i is required for the timestamping, clock state calculation and employment of clock correction, and the local clock is actually corrected at the time $t_k + \kappa_i + \eta_i$:

$$P_i[k]^+ = P_i^{t_k+\kappa_i+\eta_i} - u_i[k], \tag{14}$$

where the processing delay η_i is also considered as the Gaussian random process with the non-zero mean $\bar{\eta}_i$ and the finite variance of $\sigma_{\eta_i}^2$, which is $\eta_i \sim (\bar{\eta}_i, \sigma_{\eta_i}^2)$.

From (4), it can be seen that the existence of clock offset leads to the inaccuracy of the clock state. Hence, the clock correction action to clock state is essentially equivalent to the application of correction input $u_i[k]$ to offset. And the clock correction algorithm (14) is rewritten as

$$\theta_i[k]^+ = \theta_i^{t_k+\kappa_i+\eta_i} - u_i[k], \tag{15}$$

where $\theta_i[k]^+$ is the clock offset after it is corrected.

Until now, we have analysed and modelled packet exchange and processing delays in the temporal dimension. However, owing to the difficulty of theoretically analysing delays, and the failure of cancelling the effects of delays in the temporal dimension, we have to study the impacts of delays, and eliminate them from the state dimension. In the succeeding section, the mathematical modelling of delays in the state dimension and the analysis of the proportional-integral closed-loop synchronisation system are demonstrated.

Remark 1. In the experiments, the packet exchange delay κ_i is almost deterministic (see Table 1), and can be measured via some instruments (e.g. a logic analyser in this paper). Thus, we can remove its impacts by subtracting $\bar{\kappa}_i$ from the timestamp $\hat{P}_i[k]$ (i.e. the feedforward control strategy). The value of $\left((-t_{d_i}) - \hat{\theta}_i[k] + \bar{\kappa}_i\right)$ in (12) is calculated from the following expression

$$(-t_{d_i}) - \hat{\theta}_i[k] + \bar{\kappa}_i = \begin{cases} \hat{P}_i[k] - \bar{\kappa}_i + t_{d_i} & if \hat{P}_i[k] - \bar{\kappa}_i + t_{d_i} < \frac{\varphi_i}{2} \\ \hat{P}_i[k] - \bar{\kappa}_i + t_{d_i} - \varphi_i & if \hat{P}_i[k] - \bar{\kappa}_i + t_{d_i} \geq \frac{\varphi_i}{2} \end{cases}. \tag{16}$$

B. Stability Analysis

In this paper, as the clock update period is assumed to be sufficiently small, it is reasonable to consider that the value of the clock state increment within (packet exchange or processing) delay durations (which is in the state dimension) is equal to the value of (packet exchange or processing) delays (which is in the temporal dimension)[2]. In addition, the non-identical clock possesses a different clock frequency; thus, the extra offset value δ, which is dependent on the length of packet exchange and processing delays, also contributes to the clock offset of the state dimension. The packet exchange delay in the state dimension satisfies

$$P_i^{t_k+\kappa_i} - P_i^{t_k} = \kappa_i + \delta_{\kappa_i}, \tag{17}$$

where δ_{κ_i} is the extra offset within a duration of κ_i; and

$$P_i^{t_k+\kappa_i+\eta_i} - P_i^{t_k+\kappa_i} = \eta_i + \delta_{\eta_i}, \tag{18}$$

is for the processing delay, similarly, δ_{η_i} is the extra clock offset during η_i.

According to the intrinsic relationship between the clock state $P_i[k]$ and clock offset $\theta_i[k]$, the following expressions are obtained:

$$\theta_i^{t_k+\kappa_i} - \theta_i^{t_k} = \kappa_i + \delta_{\kappa_i} \tag{19}$$

for the packet exchange delay, and

$$\theta_i^{t_k+\kappa_i+\eta_i} - \theta_i^{t_k+\kappa_i} = \eta_i + \delta_{\eta_i} \tag{20}$$

for the processing delay. Hence, the measurement equation (11) is modified to

$$\hat{\theta}_i[k] = \theta_i[k] + (\kappa_i + \delta_{\kappa_i}). \tag{21}$$

Lemma 1. From Fig. 1, it can be seen that the correction input should be applied to the clock offset at $t_k + \kappa_i$, yielding,

$$\theta_i^{(t_k+\kappa_i)^+} = \theta_i^{(t_k+\kappa_i)^-} - u_i^{t_k+\kappa_i}, \tag{22}$$

where $\theta_i^{(t_k+\kappa_i)^+}/\theta_i^{(t_k+\kappa_i)^-}$ is the clock offset after/before it is corrected at the time $t_k + \kappa_i$. $u_i^{t_k+\kappa_i}$ is the correction input,

[2]An example of the packet exchange delay is shown in Fig. 1.

which is equal to $u_i[k]$ in (12). Based on (20), the clock offset after η_i ought to be

$$\theta_i^{t_k+\kappa_i+\eta_i} = \theta_i^{(t_k+\kappa_i)^+} + (\eta_i + \delta_{\eta_i}). \tag{23}$$

In practice, the processing delay η_i occurs when the sensor node performs the clock state calculation and accesses the counter register. The actual offset $\theta_i^{t_k+\kappa_i+\eta_i}$, after clock correction via employing $u_i^{t_k+\kappa_i}$, is

$$\theta_i^{t_k+\kappa_i+\eta_i} = \theta_i^{(t_k+\kappa_i)^-} - u_i^{t_k+\kappa_i}. \tag{24}$$

Comparison between (23) and (24) indicates that, as a result of the processing delay, there exists a difference of $-(\eta_i+\delta_\eta)$ on the clock offset at $t_k+\kappa_i+\eta_i$, which means the extra value of $(\eta_i + \delta_\eta)$ is unintentionally employed to correct the local clock. This procedure is modelled as

$$\theta_i^{t_k+\kappa_i+\eta_i} = \theta_i^{(t_k+\kappa_i)^-} - \left(u_i^{t_k+\kappa_i} + (\eta_i + \delta_{\eta_i})\right). \tag{25}$$

The clock correction action (25) is in the k-th synchronisation cycle, and the packet exchange delay and processing delay is much less than the synchronisation cycle of this work. For theoretical analysis, (25) is rewritten as

$$\theta_i[k]^+ = \theta_i[k]^- - \left(u_i[k] + (\eta_i + \delta_{\eta_i})\right), \tag{26}$$

Remark 2. The effects of packet exchange delay are in the temporal dimension. As a result of the collision-free *Sync* transmission in the PkCOs protocol, the packet exchange delay is almost deterministic with little variance, and the feedforward control can be utilised to compensate for this delay. The impacts of processing delay are in the state dimension. In PkCOs, during the same synchronisation cycle, the earlier or later employment of correction input to a local clock has no effects on synchronisation performance. However, the impacts of processing delay are still shown on the achieved precision (i.e. the asymptotic convergence error). Thus, the PI controller can be adopted to fully eliminate η_i.

By employing correction input $u_i[k]$ to the non-identical clock (8), the following expression is obtained:

$$\theta_i[k+1] = \theta_i[k] + u_i[k] + \gamma_i T - \Delta\varphi_i + \omega_i[k]. \tag{27}$$

Theorem 1. Given a single cluster network, consisting of a master node with perfect clock and N sensor nodes with drifting clocks of skews γ_i, all sensor node clocks asymptotically synchronise with the master clock and achieve a steady synchronisation state, if and only if gains α and β of the PI controller satisfy the following conditions

$$\begin{cases} 0<\alpha<2 \\ 0<\beta<\frac{\alpha^2}{4} \end{cases} \text{or} \begin{cases} 2\le\alpha<4 \\ (2\alpha-4)<\beta<\frac{\alpha^2}{4} \end{cases} \text{or} \begin{cases} 0<\alpha<4 \\ \frac{\alpha^2}{4}<\beta<\alpha \end{cases}.$$

This means that the cluster network is stable and the i-th clock offset tracks t_{d_i} at the synchronised state; that is

$$\lim_{k\to\infty} \theta_i[k] = -t_{d_i}.$$

In other words, the i-th node is scheduled and transmits its *Sync* packet at the allocated time slot t_{d_i}.

Proof. As indicated in the PkCOs protocol, the process of synchronising a drifting clock with the master clock can be modelled as a state-space feedback control system. The aim of time synchronisation is to keep the clock offset as small as possible, while, the concurrent packet transmission leads to the occurrence of packet collision in the network. Therefore, the reference input, denoted by $-t_{d_i}$, for each node, is configured to a different value, which means that the i-th clock offset follows $-t_{d_i}$ in the steady state, and the i-th sensor node is desynchronised to the slot t_{d_i} for *Sync* packet transmission. Eventually, according to Lemma 1, the proportional-integral closed-loop clock synchronisation is given by

$$\begin{cases} \theta_i[k+1] = \theta_i[k] + u_i[k] + (\gamma_i T - \Delta\varphi_i + \omega_i[k]) \\ \hat{\theta}_i[k] = \theta_i[k] + (\kappa_i + \delta_{\kappa_i}) \\ w_i[k+1] = w_i[k] + \beta\left((-t_{d_i}) - \hat{\theta}_i[k] + \bar{\kappa}_i\right) \\ u_i[k] = w_i[k] + \alpha\left((-t_{d_i}) - \hat{\theta}_i[k] + \bar{\kappa}_i\right) - (\eta_i + \delta_{\eta_i}) \end{cases}. \tag{28}$$

For the purpose of stability analysis, the system (28) is rewritten in the following matrix form:

$$\begin{aligned} \begin{bmatrix} \theta_i[k+1] \\ w_i[k+1] \end{bmatrix} &= \begin{bmatrix} 1-\alpha & 1 \\ -\beta & 1 \end{bmatrix} \begin{bmatrix} \theta_i[k] \\ w_i[k] \end{bmatrix} + \begin{bmatrix} -\alpha \\ -\beta \end{bmatrix} t_{d_i} \\ &+ \begin{bmatrix} 1 & -\alpha \\ 0 & -\beta \end{bmatrix} \begin{bmatrix} -(\eta_i+\delta_{\eta_i}) + (\gamma_i T - \Delta\varphi_i + \omega_i[k]) \\ (\kappa_i + \delta_{\kappa_i}) - \bar{\kappa}_i \end{bmatrix}. \end{aligned} \tag{29}$$

Let $x_i[k] = \left[\theta_i[k], \quad w_i[k]\right]^T$, $o_i[k] = \left[-(\eta_i+\delta_{\eta_i}) + (\gamma_i T - \Delta\varphi_i + \omega_i[k]), \quad (\kappa_i+\delta_{\kappa_i}) - \bar{\kappa}_i\right]^T$, (29) is simplified to

$$x_i[k+1] = Ax_i[k] + Bt_{d_i} + Co_i[k], \tag{30}$$

where three matrices A, B and C are equal to

$$A = \begin{bmatrix} 1-\alpha & 1 \\ -\beta & 1 \end{bmatrix},\ B = \begin{bmatrix} -\alpha \\ -\beta \end{bmatrix},\ C = \begin{bmatrix} 1 & -\alpha \\ 0 & -\beta \end{bmatrix},$$

respectively.

Using the z-transformation, we can find that the characteristic polynomial of the system (30) is

$$(z-1)^2 + (\alpha(z-1) + \beta),$$

and the polynomial has roots inside the unit circle if and only if

$$\begin{cases} 0<\alpha<2 \\ 0<\beta<\frac{\alpha^2}{4} \end{cases} \text{or} \begin{cases} 2\le\alpha<4 \\ (2\alpha-4)<\beta<\frac{\alpha^2}{4} \end{cases} \text{or} \begin{cases} 0<\alpha<4 \\ \frac{\alpha^2}{4}<\beta<\alpha \end{cases}.$$

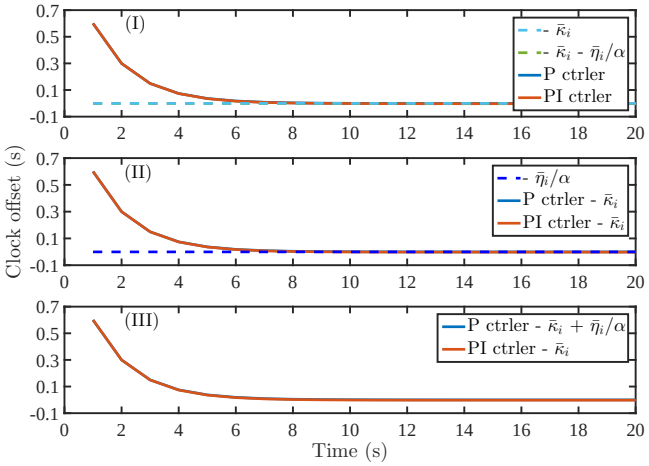
Therefore, if the PI controller's gains satisfy the conditions above, all sensor node clocks synchronise with the master clock, and the cluster network is stable and in the steady synchronisation state.

Next, the z-transformation of (30) is

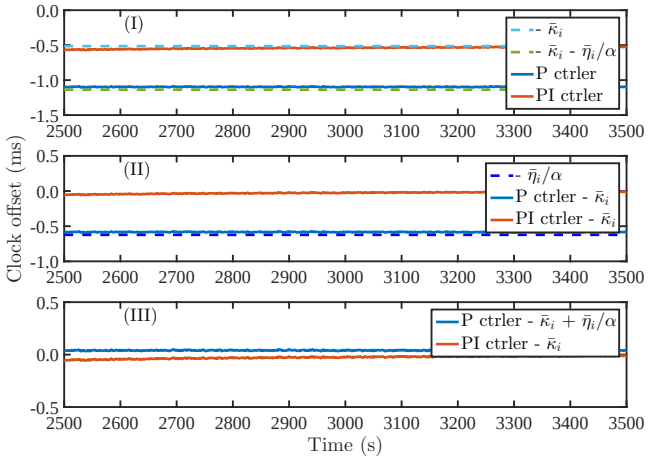
$$x[z] = (zI-A)^{-1}\frac{Bt_{d_i}}{1-z^{-1}} - (zI-A)^{-1}Co_i[z].$$

From the final value theorem, it is easy to find that $x_i[k]$ converges to a certain value following

$$\begin{aligned} \lim_{k\to\infty} x_i[k] &= \lim_{z\to1}(z-1)x_i[z] \\ &= \lim_{z\to1}(z-1)\left(\frac{z\left(zI-A\right)^{-1}Bt_{d_i}}{(z-1)} - (zI-A)^{-1}Co_i[z]\right) \\ &= \begin{bmatrix} \alpha & -1 \\ \beta & 0 \end{bmatrix}^{-1} \begin{bmatrix} -\alpha \\ -\beta \end{bmatrix} t_{d_i} = \begin{bmatrix} -t_{d_i} \\ 0 \end{bmatrix}. \end{aligned}$$



(a) Offset evolution from 0 seconds to 20 seconds.



(b) Clock offset in the steady synchronisation state.

Fig. 3. Clock offset evolution under the proportional and proportional-integral PkCOs synchronisation protocols ($t_{d_i} = 0$) in the point-to-point network.

Obviously, in the steady synchronised state, the i-th clock offset $\theta_i[k]$ is close to $-t_{d_i}$, which means that the i-th sensor node is scheduled and transmits its *Sync* packet at the allocated slot t_{d_i}. □

IV. Simulation Results

To validate the theoretical results presented in the preceding sections, the simulations are conducted in Simulink by employing parameters obtained from the hardware testbed (see Section 5). In the simulations, a non-identical clock with an initial clock offset $\theta_i[0] = 0.6s$ and a skew of $\gamma_i = 10$ ppm is adopted. The clock offset is subject to a random perturbation with standard deviation $\sigma_{\theta_i} = 10^{-6}$ [24]. The synchronisation cycle is configured to $1s$, which also means the clock threshold is 1 second. In addition, the average values of packet exchange delay and processing delay are respectively set to $513.873\mu s$ and $311.475\mu s$; the standard deviations of their noises are $\sigma_{\kappa_i} = 0.296 \times 10^{-6}$ and $\sigma_{\eta_i} = 3.899 \times 10^{-6}$, respectively. The closed-loop (point-to-point) network system consisting of one master and a sensor node, is simulated.

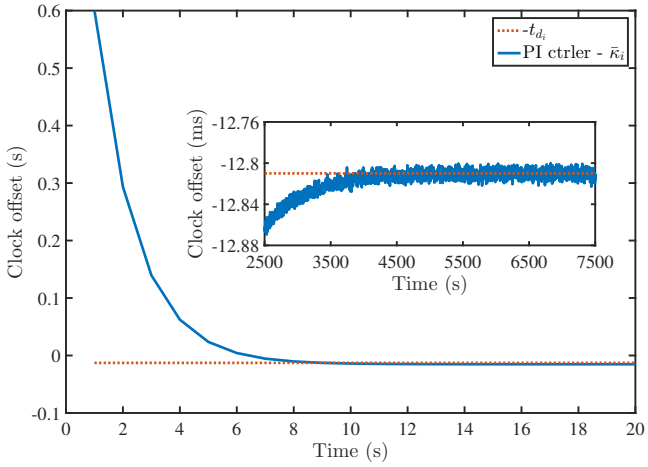


Fig. 4. Evolution of the clock offset by using PI-based PkCOs in a point-to-point network.

Fig. 3.a shows the evolution of the clock offset over time, and the performance of different control strategies is detailed in Fig. 3.b. Clearly, it can be seen that both the P and PI controllers guarantee that the network system is stable. The reference input $-t_{d_i}$ is zero, which means the clock offset should be zero in the steady synchronisation state. However, in Fig. 3.b, the proportional controller lets the clock offset converge to the value of $-\bar{\kappa}_i - \bar{\eta}_i/\alpha$, which is dependent on the packet exchange and processing delays. The addition of an integral controller automatically cancels the effects of processing delay, and the offset θ_i converges to $-\bar{\kappa}_i$ under the PI controller, which is impossible in the P controller (see Fig. 3.b).

From the experimental results, the packet exchange delay κ_i is almost deterministic with little variance. Thus, we can subtract the mean value of packet exchange delay from the timestamp $\hat{P}_i[k]$ (i.e. the feedforward control). Fig. 3.b.II indicates that, by compensating for the packet exchange delay, the clock offset tracks the reference input $-t_{d_i} = 0ms$ under the PI controller; while a P controller lets the clock offset approach $-\bar{\eta}_i/\alpha$.

Furthermore, if the compensation of processing delay is applied to the P controller, in the synchronised state, the clock offset is close to zero (see Fig. 3.b.III). However, the asymptotic error in the proportional controller is slightly larger than that of the PI controller, since the integral controller naturally eliminates the effects of processing delay, while, the delay compensation strategy can only cancel the deterministic part of processing delay.

Fig. 4 presents the clock offset θ_i is closed to $-12.81ms$ by compensating the packet exchange delay and setting the reference input $-t_{d_i}$ to $-12.81ms$. In other words, once the network achieves clock synchronisation, the i-th node fires and transmits its *Sync* at allocated time slot $t_{d_i} = 12.81ms$ within each synchronisation cycle.

V. Experimental Results

To evaluate the performance of PI-based PkCOs, it is implemented on an Atmel SMART SAM R21 hardware testbed. In

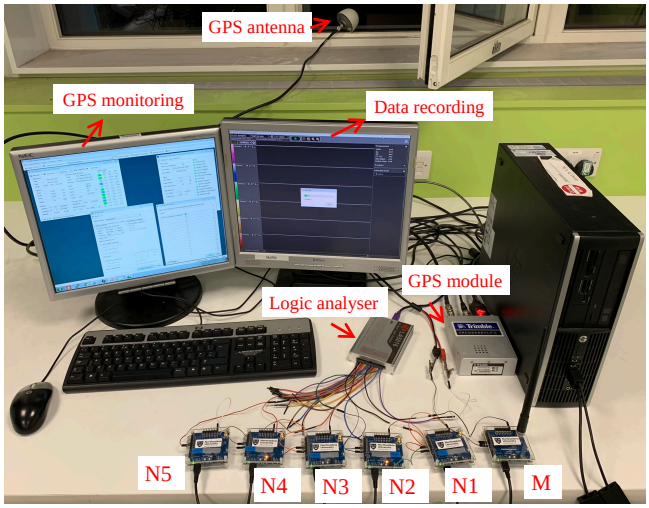


Fig. 5. Hardware testbed (M denotes the master node, and N1, N2, N3, N4 and N5 represent five sensor nodes).

Fig. 5, the Trimble Thunderbolt E GPS Disciplined Clock [27], providing a Pulse Per Second (PPS) signal, is connected to the PA06 pin (which is used for triggering a processor's external interrupt) of the master node. Once the master receives a PPS signal from the GPS module, it issues an external interrupt and then sends a *Sync* packet to the wireless channel directly. The PA19 pin of each node is attached to a Kingst Logic Analyser LA5016 [28], which records the time of the packet transmission and reception, mac-level timestamping, clock offset calculation, and clock firing and resetting events.

In the implementation, the clock state is represented by a 32-bit COUNT register of Real-Time Clock (RTC) module using a 32.768kHz crystal oscillator as a clock source. The compare register COMP0 is set to 32767, which means that the clock threshold is 1 second. Once the COUNT register reaches COMP0, it is reset to zero, and a compare match interrupt is triggered, in order to send a 21-byte *Sync* packet for synchronisation purposes. α and β are $1/2$ and $1/1300$, respectively.

Due to the use of the desynchronisation scheme, it is unnecessary to check the status of the wireless channel before packet transmission. Therefore, in the RTC interrupt handler, the *Sync* packet is directly transmitted without the CSMA scheme, and the backoff period is configured to zero. In the sensor node's reception procedure, once the MAC addressing fields (i.e. PAN ID and destination address) of the received *Sync* packet matches local addresses, an AMI (Address Match Interrupt) is issued to generate a timestamp and correct the local clock. Note that all the aforementioned interrupts (i.e. external interrupt, compare match interrupt, and AMI interrupt) are hardware interrupts issued by the processor.

In order to thoroughly study the effectiveness of the PI-based PkCOs protocol, the PCO synchronisation algorithm [8] is selected as a comparison. In the PCO protocol, the i-th node advances its state P_i by one coupling strength ϵ, when receiving a *Sync* packet from the master node. Once P_i exceeds the threshold φ_i, P_i is reset to 0 whilst a *Sync* is sent to the channel. The behaviour of the PCO model is described as

$$P_i[k]^+ = \begin{cases} P_i[k]^- + \epsilon, & \text{if } P_i[k]^- + \epsilon < \varphi_i \\ 0, & \text{if } P_i[k]^- + \epsilon \geq \varphi_i \end{cases}, \tag{31}$$

where ϵ is set to $30.5ms$ in the experiments.

In addition, for the purpose of guaranteeing a fair comparison between the PkCOs and PCO protocols, this paper adopts a synchronisation precision, which is denoted by Δ, as the evaluation metric to study their performance. In the steady synchronised state, the accuracy $\Delta_i[k]$ is defined as the difference of reference time between the i-th sensor node clock fires and master clock fires, following

$$\Delta_i[k] := t_i[k] - t_0[k] - t_{d_i}, \tag{32}$$

where $t_0[k]$ is the reference time of the master clock's fire event. Similarly, $t_i[k]$ represents the i-th clock's fire time t. The average value and standard deviation of precision Δ are also used to show their synchronisation performance.

A. Packet Exchange Delay and Processing Delay

For the purpose of investigating the effects of packet exchange delay and processing delay on synchronisation performance, both of these two delays require to be analysed on the wireless node. Table 1 shows the mean values and standard deviations of the packet exchange delay κ_i between the master's packet sending and the sensor node's packet reception, and the processing delay η_i of the i-th sensor node. It can be seen that the average values of the packet exchange delay and processing delay are about $513\mu s$ and $311\mu s$, respectively. Thanks to the proposed desynchronisation approach, the packet exchange delay κ_i is almost deterministic with little variance, around $0.3\mu s$. Thus, it is feasible to eliminate the effects of packet exchange delay, by subtracting its mean value $\bar{\kappa}_i$ from the timestamp $\hat{P}_i$. Even though the processing delay is estimated in the experimental environments, it is difficult to measure in industrial systems, and it also varies on different platforms.

In the experiments, the clock frequency is 32.768kHz, which means the clock resolution (i.e. the clock update period) is $30.5\mu s$, and the assumption of sufficiently high clock resolution cannot be met. Thus, the values of the delays (i.e. $\kappa_i + \delta_{\kappa_i}$ and $\eta_i + \delta_{\eta_i}$) fail to be integer multiples of the clock update period. In other words, the employment of the delay compensation strategy cannot accurately compensate for delays in the state dimension, and there always exists a bias between the achieved synchronisation precision and the desired precision (i.e. zero).

B. Time Synchronisation in A Point-to-Point Network

Fig. 6 and Fig. 7 show the evolution of precision over time under the PkCOs and PCO protocols. Time is measured in terms of the number of synchronisation cycles. Clearly, both P-based and PI-based PkCOs ensure the network realises clock synchronisation. However, due to the existence of packet exchange delay and processing delay, the perfect synchronisation fails to realise, and the achieved precision cannot be zero in the synchronised state. To be specific, the proportional

TABLE I
CHARACTERISTICS OF THE PACKET EXCHANGE AND PROCESSING DELAYS

	Packet exchange delay $\kappa_i(\mu s)$		Processing delay $\eta_i(\mu s)$	
	Mean	Std dev	Mean	Std dev
Node 1	513.863	0.293	311.529	4.062
Node 2	513.885	0.295	311.421	3.772
Node 3	513.874	0.293	311.545	4.091
Node 4	513.873	0.302	311.311	3.398
Node 5	513.871	0.297	311.568	4.173

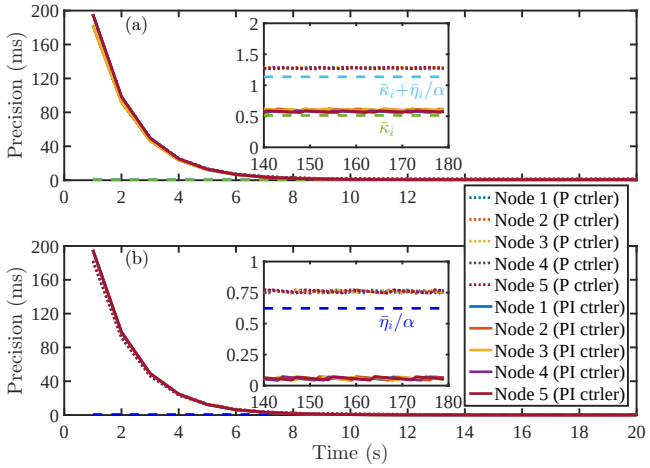


Fig. 6. Evolution of the synchronisation precision under the P-based and PI-based PkCOs algorithms.

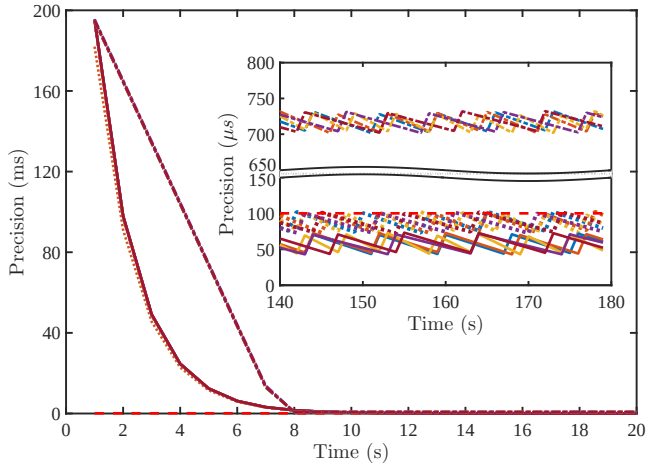


Fig. 7. Precision evolution under the PkCOs and PCO protocols (solid line: PI-based PkCOs, dotted line: P-based PkCOs, dash-dotted line: PCO, red dashed line: accuracy in [16]).

PkCOs protocol lets precision approach $\bar{\kappa}_i + \bar{\eta}_i/\alpha$. Thanks to the addition of an integral controller, the PI-based PkCOs algorithm is capable of automatically removing the effects of unknown processing delay, and it can obtain a synchronisation precision of about $\bar{\kappa}_i$ (see Fig. 6.a).

Since the packet exchange delay is almost deterministic and can be measured via the logic analyser, we can use the feedforward control strategy to cancel its impacts. Once the packet exchange delay is compensated, the PI-based PkCOs protocol achieves synchronisation with a precision of around $60\mu s$; while, under the proportional PkCOs algorithm, the precision is close to $\bar{\eta}_i/\alpha$ (see also Fig. 6.b). In addition to the impossibility of the sufficiently high clock update period, in the PI-based PkCOs protocol, a synchronisation precision of $60\mu s$ also results from the inaccuracy of the software floating-point calculation.

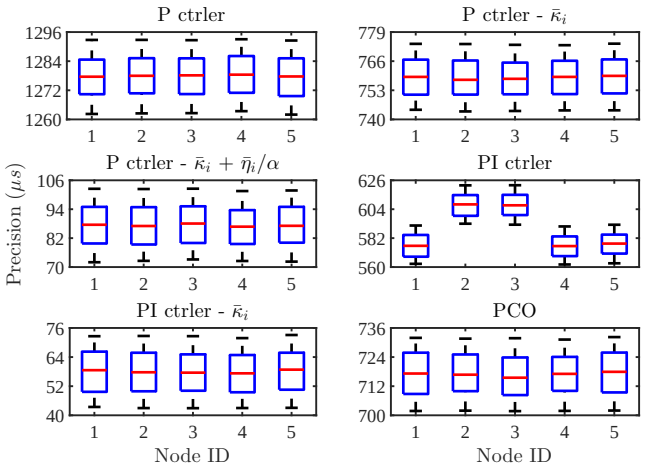


Fig. 8. Synchronisation precision under the PkCOs and PCO algorithms.

Similarly, in Fig. 7, if the mean value $\bar{\eta}_i$ is employed to P-based PkCOs to compensate for the processing delay, the achieved precision is about $90\mu s$. However, it is worse than that of proportional-integral PkCOs, as the integral controller completely eliminates the effects of processing delay, which is impossible in the P controller. Furthermore, the PCO model only realises synchronisation with a precision of around $700\mu s$, owing to the impossibility of compensating for delays.

In the experiments, since the sensor node clock's frequency differs slightly from that of the master clock, and only the clock offset is corrected, the achieved synchronisation precision decreases gradually due to the frequency differences between the master and WSN node clocks. Meanwhile, owing to the limitation of the clock resolution, once the realised synchronisation precision reduces by one clock period (i.e. $30.5\mu s$ in this work), the synchronisation scheme is activated again to correct the local clock. Thus, the behaviour of the achieved precision acts as the sawtooth pattern, as shown in Fig. 7. Moreover, the non-identical clock frequency leads to different sawtooth patterns.

Fig. 8 indicates the achieved precision under different synchronisation protocols. The central red mark is the median, and the bottom and top edges of the box, respectively, denote the 25-th and 75-th percentiles. The whiskers extend to the most extreme data points. In addition, the average values and standard deviations of the realised synchronisation precision on five WSN nodes are summarised in Table 2.

TABLE II
AVERAGE VALUES AND STANDARD DEVIATIONS OF PRECISION ON THE PKCOS AND PCO PROTOCOLS (UNIT: MICROSECOND (μs))

	P controller		P controller $-\ \bar{\kappa}_i$		P controller $-\ \bar{\kappa}_i + \bar{\eta}_i/\alpha$		PI controller		PI controller $-\ \bar{\kappa}_i$		PCO	
	Mean	Std dev	Mean	Std dev	Mean	Std dev	Mean	Std dev	Mean	Std dev	Mean	Std dev
Node 1	1277.416	9.035	758.773	8.876	87.640	8.895	576.391	8.933	57.729	8.982	717.073	9.053
Node 2	1277.901	8.829	758.553	8.861	87.142	9.056	607.267	8.972	58.140	9.080	717.046	8.919
Node 3	1277.961	8.844	758.328	8.706	87.751	8.859	607.306	9.001	57.622	8.768	716.023	8.959
Node 4	1278.694	8.817	758.655	8.831	86.576	8.679	575.865	8.855	57.282	8.841	716.915	8.592
Node 5	1277.817	8.971	759.291	9.060	87.472	8.703	577.204	8.644	58.295	8.877	717.509	9.154

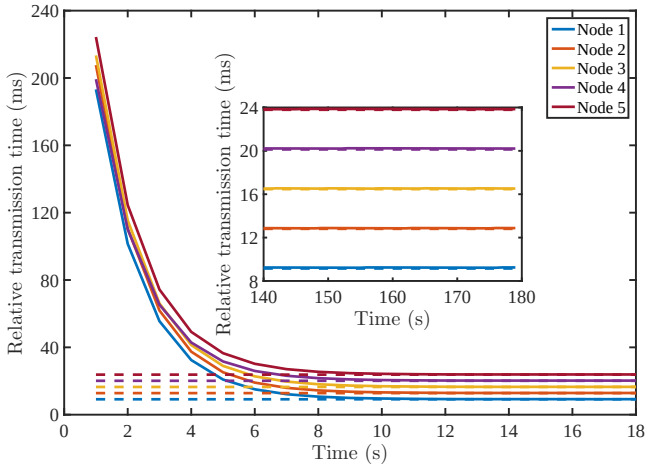


Fig. 9. Transmission time relative to the master during each synchronisation cycle (dashed line: desynchronisation duration t_{d_i}, solid line: relative transmission time).

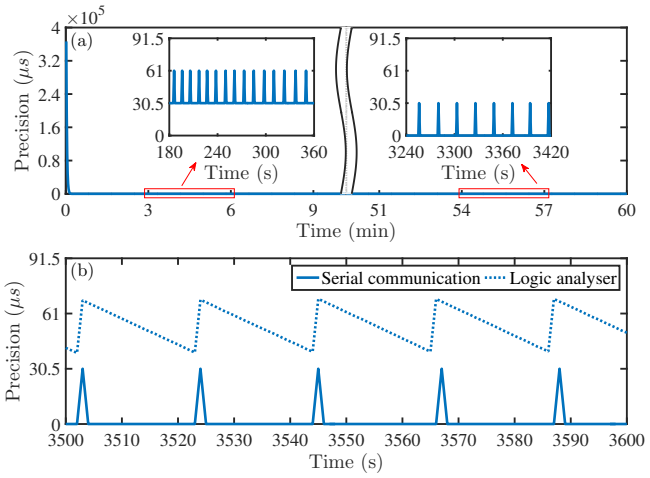


Fig. 10. Long-term stability of the PI-based PkCOs scheme.

C. *Synchronisation in A Single-hop Cluster WSN*

To evaluate the performance of the PI-based PkCOs protocol, it is examined in a single-hop WSN consisting of a master node and five sensor nodes. Although both master and sensor nodes are capable of sending *Sync* packets, sensor nodes can only receive *Sync* packets from the master, which is connected to GPS. In order to realise the scheduling of *Sync*s, data period t_{dp} and slot duration t_{sd} are set to $9.15ms$ and $3.66ms$, respectively. This means that five sensor nodes transmit *Sync*s at corresponding allocated time slots, namely, $9.15ms$, $12.81ms$, $16.47ms$, $20.13ms$ and $23.79ms$, respectively.

Fig. 9 demonstrates the time of the node's packet transmission event relative to the master. Clearly, once the i-th sensor node's clock synchronises with the master clock, the i-th node is allocated to the designated time slot t_{d_i} to send its *Sync* packet. For example, when the 2-nd clock synchronises with the master clock, Node 2 is allocated to slot $12.81ms$, which is relative to the master's packet transmission, to send *Sync*s.

In the end, we also examine the long-term stability of the PI-based PkCOs protocol. Due to the limitation of the logic analyser, only 200-second data can be sampled, and the serial communication method is adopted to record data. In the experiments, through serial communication, the data read from COUNT is sent to the PC for offline performance analysis.

The time synchronisation precision obtained from the serial communication method is shown in Fig. 10.a. It is clear that the achieved precision approaches zero. To distinctly show the performance of the PkCOs algorithm, the realised accuracy in the last 100 seconds is indicated in Fig. 10.b. Due to the non-identical clock frequency and limitation of clock resolution, each time the achieved precision loses by one tick (i.e. $30.5\mu s$), the processor activates the PkCOs scheme again to correct the local clock. The period of the data's pattern from serial communication is the same as that of the sawtooth behaviour from the logic analyser. Thus, during the one-hour experiments, the achieved precision maintains around $60\mu s$.

In [16], Even though a precision of around $100\mu s$ is obtained under the PCO-like synchronisation protocol, the clock frequency is $115,200$Hz (i.e. 1 tick is about $8.68\mu s$). This means that [16] achieves synchronisation with the accuracy of 11.5 ticks. In this work, the PI-based PkCOs protocol realises a precision of around 2 ticks. Therefore, by using similar hardware configurations, the proposed synchronisation method has a clear potential to overcome the results in [16].

VI. CONCLUSION

In this paper, we propose a framework of PkCOs to characterise the dynamics of communication and time synchronisation of clocks in WSNs. The proportional-integral controller is utilised to compensate for the unknown processing delay automatically. The experimental results show that, in the

single-cluster wireless network, the PI-based PkCOs protocol achieves synchronisation with a precision of $60\mu s$ (i.e. 2 ticks) on 32.768kHz crystal oscillator-based clocks.

ACKNOWLEDGMENT

Y. Zong thanks Dr Tim Phillips for providing advice to revise the manuscript.

REFERENCES

[1] Y. Zong, X. Dai, P. Canyelles-Pericas, K. Busawon, R. Binns, and Z. Gao, "Modelling and Synchronisation of Delayed Packet-Coupled Oscillators in Industrial Wireless Sensor Networks," in *IFAC Proc. Volumes*, Jul. 2020.
[2] F. Tirado-Andres, A. Rozas, and A. Araujo, "A Methodology for Choosing Time Synchronization Strategies for Wireless IoT Networks," *Sensors*, vol. 19, no. 16, pp. 3476–3494, Aug. 2019.
[3] A. Elsts, X. Fafoutis, S. Duquennoy, G. Oikonomou, R. Piechocki, and I. Craddock, "Temperature-Resilient Time Synchronization for the Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 5, pp. 2241–2250, May. 2018.
[4] J. P. Ramirez, L. A. Olvera, H. Nijmeijer, and J. Alvarez, "The Sympathy of Two Pendulum Clocks: Beyond Huygens' Observations," *Scientific Reports*, vol. 6, no. 23580, Mar. 2016.
[5] H. Hennig, "Synchronization in Human Musical Rhythms and Mutually Interacting Complex Systems," *Proc. National Academy Sci. United States America (PNAS)*, vol. 111, no. 36, pp. 12 974–12 979, Sep. 2014.
[6] M. K. Mcclintock, "Menstrual Synchrony and Suppression," *Nature*, pp. 244–245, Jan. 1971.
[7] C. S. Peskin, *Mathematical Aspects of Heart Physiology*, 1975.
[8] R. E. Mirollo and S. H. Strogatz, "Synchronization of Pulse-Coupled Biological Oscillators," *SIAM J. Appl. Math.*, vol. 50, no. 6, pp. 1645–1662, 1990.
[9] H. M. Smith, "Synchronous Flashing of Fireflies," *Science*, vol. 82, no. 2120, pp. 151–152, Aug. 1935.
[10] R. Pagliari, Y.-W. P. Hong, and A. Scaglione, "Bio-Inspired Algorithms for Decentralized Round-Robin and Proportional Fair Scheduling," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 4, pp. 564–575, May. 2010.
[11] T. Anglea and Y. Wang, "Phase Desynchronization: A New Approach and Theory Using Pulse-Based Interaction," *IEEE Trans. Signal Process.*, vol. 65, no. 5, pp. 1160–1171, Mar. 2017.
[12] F. Ferrante and Y. Wang, "Robust Almost Global Splay State Stabilization of Pulse Coupled Oscillators," *IEEE Trans. Autom. Control*, vol. 62, no. 6, pp. 3083–3090, Jun. 2017.
[13] L. Ferrari, A. Scaglione, R. Gentz, and Y.-W. P. Hong, "Convergence Results on Pulse Coupled Oscillator Protocols in Locally Connected Networks," *IEEE/ACM Trans. Netw.*, vol. 25, no. 2, pp. 1004–1019, Apr. 2017.
[14] H. Gao and Y. Wang, "On Phase Response Function Based Decentralized Phase Desynchronization," *IEEE Trans. Signal Process.*, vol. 65, no. 21, pp. 5564–5577, Nov. 2017.
[15] J. Zhang, M. Wang, M. Hua, W. Yang, and X. You, "Robust Synchronization Waveform Design for Massive IoT," *IEEE Trans. Wireless Commun.*, vol. 16, no. 11, pp. 7551–7559, Nov. 2017.
[16] Z. An, H. Zhu, X. Li, C. Xu, Y. Xu, and X. Li, "Nonidentical Linear Pulse-Coupled Oscillators Model With Application to Time Synchronization in Wireless Sensor Networks," *IEEE Trans. Ind. Electron.*, vol. 58, no. 6, pp. 2205–2215, Jun. 2011.
[17] J. Klinglmayr and C. Bettstetter, "Self-organizing Synchronization with Inhibitory-Coupled Oscillators: Convergence and robustness," *ACM Trans. Auton. Adaptive Syst.*, vol. 7, no. 3, Sep. 2012.
[18] R. Gentz, A. Scaglione, L. Ferrari, and Y.-W. P. Hong, "PulseSS: A Pulse-Coupled Synchronization and Scheduling Protocol for Clustered Wireless Sensor Networks," *IEEE Internet of Things J.*, vol. 3, no. 6, pp. 1222–1234, Dec. 2016.
[19] A. Tyrrell, G. Auer, and C. Bettstetter, "On the Accuracy of Firefly Synchronization with Delays," in *Proc. 1st Int. Symp. Applied Sciences Biomedical Commun. Technologies*, Oct. 2008.
[21] R. Pagliari and A. Scaglione, "Scalable Network Synchronization with Pulse-Coupled Oscillators," *IEEE Trans. Mobile Comput.*, vol. 10, no. 3, pp. 392–405, Mar. 2011.
[20] Y.-W. Hong and A. Scaglione, "A Scalable Synchronization Protocol for Large Scale Sensor Networks and Its applications," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 5, pp. 1085–1099, May. 2005.
[22] *SAM R21 Xplained Pro Evaluation Kit*.
[23] J. Degesys, I. Rose, A. Patel, and R. Nagpal, "DESYNC: Self-Organizing Desynchronization and TDMA on Wireless Sensor Networks," in *Proc. 6th Int. Symp. Inf. Process. Sensor Netw.*, Apr. 2007.
[24] G. Giorgi and C. Narduzzi, "Performance Analysis of Kalman-Filter-Based Clock Synchronization in IEEE 1588 Networks," *IEEE Trans. Instrum. Meas.*, vol. 60, no. 8, pp. 2902–2909, Aug. 2011.
[25] Y. Huang, T. Li, X. Dai, H. Wang, and Y. Yang, "TS2: A Realistic IEEE1588 Time-Synchronization Simulator for Mobile Wireless Sensor Networks," *SIMULATION*, vol. 91, no. 2, pp. 164–180, Jan. 2015.
[26] Y. Zong, X. Dai, Z. Gao, K. Busawon, R. Binns, and I. Elliott, "Synchronization of Pulse-Coupled Oscillators for IEEE 802.15.4 Multi-Hop Wireless Sensor Networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM'18)*, Dec. 2018.
[27] *The Trimble Thunderbolt E GPS Disciplined Clock*.
[28] *Kingst Logic Analyzer LA5016*.

Yan Zong received the B.Eng. degrees in Electrical and Electronic Engineering from Northumbria University, UK (First-Class Honours), and Nanjing Normal University, PRC, in 2016, and the Ph.D. degree in Electrical Engineering from Northumbria University, UK, in 2020. His research interests cut across several disciplines, which include time synchronisation in industrial Wireless Sensor Networks (WSNs) and industrial Internet of Things (IoT), consensus problem in Multi-Agent Systems (MAS), synchronisation in complex networks, and (robust) control.

He is currently an Embedded Software Engineer at Keiky Limited, UK.

Xuewu Dai received the B.Eng. in electronic engineering and the M.Sc. in computer science from Southwest University, China, in 1999 and 2003, respectively, and the Ph.D. from the University of Manchester, U.K., in 2008. His research interests include robust state estimation, networked control systems and synchronisation, and their applications to the time-sensitive Industrial Internet of Things.

He is a Senior Lecturer with the Department of Mathematics, Physics and Electrical Engineering, Northumbria University. Prior that, he was a postdoc at the Department of Engineering Science, University of Oxford (2011-2013) and at the Department of Electronic and Electrical Engineering, UCL (2009-2011).

Dr Dai was awarded the Early Career Research Prize by SWIG UK.

Zhiwei Gao received the B.Eng. degree in electrical engineering and automation and the M.Eng. and Ph.D. degrees in systems engineering from Tianjin University, Tianjin, China, in 1987, 1993, and 1996, respectively. His research interests include stochastic control systems, data-driven modelling, estimation and filtering, fault diagnosis, resilient control, intelligent optimisation, power electronics, wind energy systems, electric vehicle batteries, and bioinformatics.

He is currently with the Faculty of Engineering and Environment, University of Northumbria, Newcastle upon Tyne, U.K., as a Reader.

Dr Gao serves several leading international journals as academic editors, and organised more than ten special issues in the premier international journals.