Synchronisation of Packet Coupled Low-accuracy RC Oscillator Clocks for Wireless Networks

Yan Zong, *Member, IEEE,* Xuewu Dai, *Member, IEEE,* Pep Canyelles-Pericas, *Member, IEEE,* Zhiwei Gao, *Fellow, IEEE,* Wai Pang Ng, *Senior Member, IEEE,* Krishna Busawon, *Senior Member, IEEE,* and Richard Binns

***Abstract*—Time-sensitive wireless applications have strict requirements on real-time data transmission and control operation. Even though time synchronisation has been extensively studied for providing a common timing among distributed wireless nodes, there still exists a lack of research for low-accuracy and large-drifting clocks, such as internal Resistor-Capacitor (RC) oscillator clocks with around 4×10^5 parts per million (ppm) frequency drift, which are widely used in wearable sensor systems. This paper proposes a Proportional Packet-Coupled Oscillators (P-PkCOs) protocol for synchronising poor-performing internal RC oscillator clocks with high disturbances in the single-cluster wireless network. The behaviour of such a drifting clock is described by a non-identical and time-varying model. To achieve time synchronisation on low-accuracy internal RC oscillator clocks, a packet-coupled synchronisation scheme is proposed for adjusting drifting clocks via the proportional control-based correction scheme. The RC oscillator frequency in an embedded system cannot be corrected, and this work utilises the clock threshold adjustment as a substitute for frequency correction. The stability region of controller parameters is given to guarantee that the clock threshold approaches a value, which is jointly determined by the nominal threshold and the corresponding clock frequency. We also propose a linear matrix inequality condition to prove that the P-PkCOs performance is robust against the large clock disturbances. We demonstrate the implementation of P-PkCOs. The experimental results show that P-PkCOs can achieve and maintain robust time synchronisation on the internal RC oscillator clocks.**

***Index Terms*—Time synchronisation, packet-coupled oscillators, internal RC oscillator clocks, wireless networks, pulse-coupled oscillators.**

I. Introduction

OVER the last two decades, for the purposes of increasing productivity and safety, optimising costs and improving product quality, the introduction of advanced technologies of Cyber-Physical System (CPS), Information and Communication Technologies (ICT) and Internet of Things (IoT) into the manufacturing sector has transformed the traditional industry, pushing it towards the fourth industrial revolution (Industry 4.0). The Industrial IoT (IIoT) provides various services (e.g. distributed data acquisition [1]) in numerous industry sectors, such as manufacturing and biometric authentication. These mission-critical IIoT applications have stringent requirements on real-time data transmission and control operation command. For example, the process automation of oil and gas industries can only tolerate delays at the millisecond level [2].

As a fundamental technology of IIoT, wireless networks play an important role in the fourth industrial revolution. The core components of networks are wireless nodes. These distributed nodes collaborate to accomplish different tasks (e.g. performing a periodic operation). To ensure that a wireless network works appropriately, all the nodes in the network need to share precise timing.

Typically, the clock module in an embedded system is based on a *hardware oscillator* [e.g. crystal oscillator, internal Resistor-Capacitor (RC) oscillator]. Each hardware oscillator drifts from an ideal oscillator at a rate. In other words, the hardware oscillator frequency deviates from the nominal value. The drifting rate (i.e. frequency deviation) is denoted in the unit of parts per million (ppm), which also represents the number of microseconds an embedded clock drifts from the perfect one every second.

The IIoT applications often adopt a high-performance crystal oscillator as the clock source. The performance of such a crystal oscillator clock is usually less than 100ppm [3], and several external units are required to form a crystal oscillator clock. However, for wireless Body Sensor Networks (BSNs) in Industry 4.0, there still exist demands of saving space and reducing hardware size, for example, with wearable biometric authentication devices [4]. The internal RC oscillator is composed of resistors and capacitors inside the integrated circuit, and no additional parts are needed, compared to the crystal oscillator clock. Thus, the utilisation of internal RC oscillator clocks can meet the requirements in BSNs.

The internal RC oscillator clock is a low-accuracy clock with high noises and disturbances. This work selects Atmel SAM R21 boards [5] as an experimental testbed, where there is an internal RC oscillator clock with a nominal frequency

Manuscript received May 21, 2022; revised October 11, 2022; accepted December 5, 2022. This work was supported in part by the University of Northumbria at Newcastle via a postgraduate research studentship, in part by the Engineering and Physical Sciences Research Council (EPSRC) [grant number EP/W028727/1]. *(Corresponding author: Xuewu Dai.)*

Y. Zong is with the College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, China, and also with the Department of Mathematics, Physics and Electrical Engineering, Northumbria University, Newcastle upon Tyne, U.K. (e-mail: y.zong@nuaa.edu.cn).

X. Dai, Z. Gao, W. Ng, K. Busawon, and R. Binns are with the Department of Mathematics, Physics and Electrical Engineering, Northumbria University, Newcastle upon Tyne, U.K. (e-mail: {xuewu.dai, zhiwei.gao, wai-pang.ng, krishna.busawon, richard.binns}@northumbria.ac.uk).

P. Canyelles-Pericas is with the Department of Integrated Devices and Systems, MESA+ Institute for Nanotechnology, University of Twente, the Netherlands (e-mail: j.canyellespericas@utwente.nl).

Digital Object Identifier 10.1109/TWC.2022.3229214

TABLE I
PACKET-EXCHANGE SYNCHRONISATION PROTOCOL COMPARISON

TS protocols	Hardware platform	Clock skew	Number of packet-exchange each TS cycle	Key technologies	TS precision
PkCOs [8]	SAM R21	10 ppm	1	Proportional-integral controller	30.5 μs
R-PkCOs [10]	SAM R21	20 ppm	1	Static output controller	6 μs
BATS [11]	TelosB	40 ppm	1	Linear regression	2.5 μs
PISync [12]	Mica2	40 ppm	1	Proportional-integral controller	2 μs
RDC-RMTS [13]	SSWS	50 ppm	5	Maximum-likelihood estimation	8 μs
FTSP [14]	Mica2	40 ppm	1	Linear regression	2.3 μs
RBS [15]	Berkeley mote	3 ppm	3	Linear regression	11.2 μs

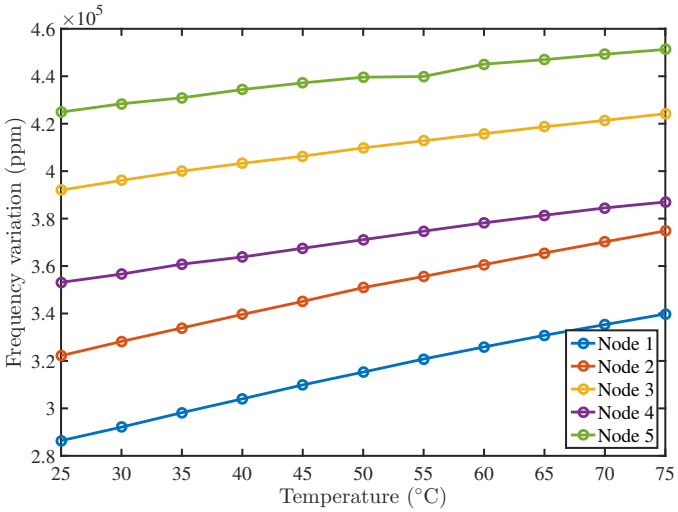


Fig. 1. Impacts of the environmental temperature on the uncalibrated internal RC oscillator clock frequency of the Atmel SAM R21 board (10-minute data at each temperature point is collected using a laboratory oven with temperatures varying from 25°C to 75°C by the step of 5°C).

32.768kHz on each board. Fig. 1 shows how the environmental temperature affects the uncalibrated internal RC oscillator clock's 32.768kHz nominal frequency. It can be seen that the frequency difference between the internal RC oscillator clock and a perfect 32.768kHz clock is about 4×10^5ppm[1]. From [6, p. 1087], the performance of the well-tuned internal RC oscillator clock is still between -1.3×10^5ppm and 6×10^4ppm. This means that the offset[2] increment every second is between -130ms (for the above well-calibrated embedded clock) and 400ms (for an untuned internal RC oscillator clock). Hence, the direct deployment of wireless nodes with internal RC oscillator clocks cannot provide accurate timing information to time-sensitive wireless applications, and an important protocol, namely, Time Synchronisation (TS), is required to let the low-accuracy embedded clocks of all the distributed wireless nodes in a network possess the same notion of time.

Inspired by the Pulse-Coupled Oscillators (PCO) model [7], we proposed the Packet-Coupled Oscillators (PkCOs) protocol for synchronising high-performance crystal oscillator clocks with around 10ppm [8]. In the PkCOs model, the firing-resetting behaviour is equivalent to the clock's periodic resetting feature. The free-running clock is modelled as an uncoupled oscillator, and the packet transmission in a wireless network is represented by the *Sync* firing of PkCOs. Our earlier works (i.e. [8], [9], [10]) only studied the synchronisation performance of PkCOs on high-performance crystal oscillator clocks. However, synchronising the large-drifting RC oscillator clocks raises more significant challenges than synchronising the crystal oscillator clocks. This paper extends PkCOs in [8] by considering the low-accuracy and large-drifting clocks with high noises (e.g. internal RC oscillator clocks in Fig. 1). Thus, we propose a Proportional Packet-Coupled Oscillators (P-PkCOs) protocol for synchronising the poor-performing clocks, and a Proportional (P) control-based correction method is also proposed. The hardware internal RC oscillator frequency in an embedded system cannot be corrected, and this work utilises the clock threshold adjustment as a substitute for frequency correction.

A. Related Work

In the literature, most existing time synchronisation works focus on highly accurate crystal oscillator clocks (less than 100ppm), for example, the clock performance of TelosB [11] and Mica2 [12] is around 40ppm, as shown in Table 1. To the best of our knowledge, there is no report on the synchronisation of extremely low-accurate clocks (e.g. the internal RC oscillator clock of about 4×10^5ppm on Atmel SAM R21). In time synchronisation, many theoretical studies have been proposed [e.g. the Pulse-Coupled Oscillators (PCO) model and its variants [16], [17]], but with assumptions, such as identical and non-drifting frequency. However, these assumptions cannot be met in the practice of wireless systems. Each clock possesses the non-identical and drifting frequency. In the communication engineering community, time synchronisation is based on the packet-exchange strategy. Typical works include [13], [18]. However, these protocols work well for crystal oscillator clocks, but can easily fail to realise synchronisation on low-accurate internal RC oscillator clocks. Note that the single-cluster topology is a popular solution for wireless BSN [1], [4]. This work possesses no need to study the time synchronisation of low-accuracy internal RC oscillator clocks in multi-hop wireless networks.

[1]See Section 6b for more details on measuring clock performance.
[2]The clock offset is defined as the time difference between two clocks.

In the PCO-like protocols, two clock models, namely, the ideal clock model with the identical and non-drifting frequency (e.g. [16], [17]), and the non-identical clock (e.g. [8], [9]) are still used for theoretical study. For instance, [8], [9] assume the clock frequency is distinct but constant. This is not true in real-world environments, since frequency variations are subjected to manufacturing tolerance and environmental factors such as operating temperature. Thus, there still exists a lack of the non-identical and time-varying clock model. We fill this gap by modelling the behaviour of an internal RC oscillator clock, which offers a solution for our protocol's theoretical analysis.

In addition, the synchronisation solution in the recent PCO-like protocols is a response function-based strategy (e.g. [16], [17]), where only the offset is adjusted according to the response function. The synchronisation precision is about $650\mu s$ on crystal oscillator clocks [16], which is worse than the achieved synchronisation accuracy via the packet-exchange synchronisation protocols (see Table 1).

The packet-exchange synchronisation protocols need frequent packet exchange and clock correction to maintain synchronisation on low-noise 100ppm clocks, leading to more power consumption. To avoid the above issue, several processing techniques [e.g. linear regression, and Maximum-Likelihood Estimation (MLE)] are employed to calculate and adjust the frequency difference (i.e. skew) among clocks. However, during the experiments, the skew correction is made on the logical (or virtual) clock, which is an affine function of the clock module of an embedded system [11] (also see Fig. 5b). The clock's hardware oscillator frequency cannot be corrected in an embedded system. Moreover, from [19], once large disturbances occur on clock frequencies, the above algorithms fail to realise synchronization. This is due to the lack of considering the high-drifting clock frequency. Hence, to achieve and maintain time synchronisation of low-accuracy clocks in the single-cluster wireless network, we propose a packet-coupled synchronisation scheme to simultaneously adjust clock offsets and skews. Furthermore, this paper utilises the clock threshold adjustment as a substitute for frequency (i.e. skew) correction in the hardware experiments.

In IIoT, wireless networks usually adopt the software timestamping [e.g. at the Medium Access Control (MAC) layer, or the upper layer], due to no additional hardware requirement. However, uncertainties and delays are introduced to the timestamp, while the processor executes the code of reading the counter register (see Section 2). Thus, the employment of complete offset and skew estimates may degrade performance. This work proposes a proportional control-based correction scheme to improve synchronisation performance.

Even though the performance of PkCOs is studied in both single-hop [8] and multi-hop [9] wireless networks, the PkCOs protocol only adjusts the clock offset for realising synchronisation. In [10], we present the robustness design of synchronisation protocol parameters under the disturbances (from the clock, packet-exchange delay and processing delay [8]), while only the low-noise clock (less than 10ppm) is considered. Due to the use of crystal oscillator clocks, the dominant effects on the synchronisation precision are from the delays [10]. This work first shows the stability region of controller parameters to guarantee that the clock threshold approaches a value, which is only determined by the nominal threshold and the average value of the corresponding clock skew. We also propose a distinct Linear Matrix Inequality (LMI) condition to prove that the P-PkCOs performance is robust against the clock noises (which are the dominant disturbances) and the timestamp uncertainties.

TABLE II
MAIN SYMBOLS AND DESCRIPTION

Symbols	Description
$P_i[n]$, $\hat{P}_i[k]$	i-th node's time variable, estimated time variable at the k-th cycle
$\theta_i[k]$, $\hat{\theta}_i[k]$	i-th node's clock offset, estimated offset at the k-th cycle
$\gamma_i[n]$, $\hat{\gamma}_i[k]$	i-th node's clock skew, estimated skew at the k-th cycle
φ_i	i-th node's clock threshold
$u_{\theta_i}[k]$, $u_{\gamma_i}[k]$, $u_{\varphi_i}[k]$	i-th node's offset correction input, skew correction input, threshold correction input at the k-th cycle
$x_i[k]$, $y_i[k]$, $u_i[k]$	i-th node's clock state vector, observation vector, control input vector at the k-th cycle
$\omega_i[k]$, $\nu_i[k]$	i-th node's clock noise vector, observation noise vector at the k-th cycle
$\eta_i[k]$	i-th node's disturbances including clock and observation noises at the k-th cycle
$o_i[k]$	i-th node's performance output vector at the k-th cycle
ρ	Robustness performance upper bound
α, β	Proportional controller parameters

B. Contributions and Paper Organisation

In this work, we propose the P-PkCOs protocol to synchronise low-accuracy internal RC oscillator clocks in a single-cluster wireless network. The behaviour of such a drifting clock is described as the non-identical and time-varying model. To achieve time synchronisation on poor-performing internal RC oscillator clocks, a packet-coupled synchronisation scheme is proposed for simultaneously adjusting both clock offsets and skews by using the proportional control-based correction scheme. The hardware RC oscillator frequency in an embedded system cannot be corrected, and this work utilises the clock threshold adjustment as a substitute for frequency correction during the hardware experiments. The stability region of controller parameters is given to guarantee that the clock threshold approaches a value, which is determined by the nominal threshold and the mean value of the corresponding clock skew. We propose a LMI condition to prove that the P-PkCOs performance is robust against the dominant clock disturbances. We also demonstrate the implementation of the proposed P-PkCOs protocol. The source code is provided as open access to the researchers and engineers. The experimental results show that P-PkCOs can maintain robust time synchronisation on the low-accuracy internal RC oscillator clocks.

The rest of this paper is organised as follows: the mathematical formulation of the low-accuracy internal RC oscillator clock and the packet-exchange mechanism is given in Section 2. Next, Section 3 presents the proportional clock correction

scheme. Section 4 shows the theoretical analysis, which is verified by using numerical simulations in Section 5. Section 6 demonstrates the implementation of P-PkCOs, and the experimental evaluation of the proposed protocol. Finally, Section 7 concludes this work.

II. Problem Formulation

In this section, by mathematically describing the internal RC oscillator clock's specific features (i.e. drifting frequency, and periodic resetting characteristic), we propose a non-identical and time-varying clock model, which is used to form a closed-loop synchronisation system for theoretical analysis (in Section 4). Meanwhile, a timestamped packet-exchange mechanism is utilised to estimate the offset and skew of each drifting clock. The proposed proportional correction method (in Section 3) uses these offset and skew estimates for clock adjustment, thereby realising robust time synchronisation of low-accuracy clocks in a network. Table 2 presents the main symbols adopted in this work.

A. Clock Model

Let a single-cluster wireless network be described by the directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where $\mathcal{V} = \{0, 1, ..., N\}$ denotes a set of nodes, and a set of edges $\mathcal{E}$ induced by the adjacency matrix $\mathcal{A}$. The network consists of a single root node and N regular nodes represented by the set $\mathcal{R} = \{i : i \in \mathcal{V}, \text{ and } i \neq 0\}$. The root node is unique, and is equipped with a Global Positioning System (GPS) clock to provide the reference time to all the regular nodes. Once a regular node receives a *Sync* packet from the root node, it adjusts its drifting clock to synchronise with the GPS clock. Throughout this work, the reference clock is called the master clock, and the root node and regular node are, respectively, referred to as the master node and sensor node.

In an embedded system, the clock module of a wireless node usually is constructed from (i) an internal RC oscillator ticking at the frequency f, (ii) and a counter register (or a chain of counters), counting the number of ticks generated by the internal oscillator. Each time the counter register matches the threshold value φ, it is reset to 0, and starts counting from zero again (see Fig. 5b). Through the processing of the counter register, the periodic signal produced by the internal RC oscillator is converted to an integer value. By converting this value, the traditional time with the unit of, for example, seconds, minutes, and hours, is obtained. In the following, an iterative form two-state model is derived to characterise the behaviour of the low-accuracy internal RC oscillator clock, since it provides a simple approach when control theory is applied for robustness analysis.

1) An Ideal Clock: Here, we first present the case of an ideal sinusoidal internal RC oscillator, the output voltage of such a hardware oscillator is defined as:

$$V(t) = V_0 \sin(2\pi f_0 t), \tag{1}$$

where t represents the reference time. V_0 and f_0 are the nominal amplitude and frequency, respectively. The nominal (clock update) period is $\tau_0 = 1/f_0$.

The periodicity of the sinusoidal signal allows the generation of a uniform sequence of events, and each event corresponds to a unique instantaneous phase value [20]. The occurrence of clock events satisfies the following expression

$$n = \left\lfloor \frac{2\pi f_0 t}{2\pi} \right\rfloor = \lfloor f_0 t \rfloor, \tag{2}$$

where the floor function $\lfloor f_0 t \rfloor$ means the greatest integer less than or equal to $f_0 t$. The event of counter reaching n is referred to as the n-th clock update event. In the literature, the time instant $t[n]$ is typically utilised to represent the reference time associated with the n-th clock event. The nominal period τ_0 is known and constant [21], the reference time $t[n]$ of the corresponding n-th event equals $t[n] = n\tau_0 = n/f_0$.

However, the monotonically increasing variable $t[n]$ cannot model the clock's counter periodic resetting feature. Hence, P-PkCOs introduces the time variable $P_0[n]$ to describe this characteristic, yielding

$$P_0[n] = n\tau_0 - \sum_{h=1}^{\lfloor \frac{n}{m_0} \rfloor} \varphi_0, \tag{3}$$

where φ_0 is the ideal clock's threshold, which is equal to the synchronisation cycle T in this work. The threshold φ_0 is much larger than τ_0, then it is reasonable to assume that the clock updates m_0 times during a single cycle (i.e. $T = m_0\tau_0$). Since the ideal internal RC oscillator clock resets every m_0 update cycles, let n represent the clock is reset at n-th cycle. That is $n = \{m_0, 2m_0, 3m_0, ..., km_0\}$. The variable k means that the clock's counter resetting totally occurs k times.

2) A Non-identical Clock with Drifting Frequency: In practice, the sensor node's clock is subjected to variations in phase and frequency due to complex factors, such as manufacturing tolerance, capacitive loading mismatch, environmental temperature and power supply voltage [22]. Mathematically speaking, these variations are owing to two main sources, which are considered as separate components of the random process: (i) instantaneous phase variations $\phi_i(t)$ denoting all instant phase fluctuations; (ii) deviations $\chi_i(t) = f_i(t) - f_0$ of the internal oscillator frequency $f_i(t)$ from its nominal value f_0, whose accumulated effects over time are phase fluctuations $2\pi \int_0^t \chi_i(\tau) d\tau$ [20]. Based on the ideal case (1), the instantaneous output voltage value of a drifting embedded clock is modelled as

$$V(t) = (V_0 + v_i(t)) \sin\left(2\pi \left(f_0 t + \int_0^t \chi_i(\tau)\, d\tau\right) + \phi_i(t)\right) \tag{4}$$

where $v_i(t)$ is the random amplitude fluctuation. $P_i[n]$ of the i-th drifting clock at the n-th event is given by

$$P_i[n] = n\tau_0 + \frac{\int_0^{t[n]} \chi_i(\tau) d\tau}{f_0} + \frac{\phi_i(t[n])}{2\pi f_0} - \sum_{h=1}^{\lfloor \frac{n}{m_i} \rfloor} \varphi_0, \tag{5}$$

where m_i represents the number of times that the drifting clock updates during φ_i. From the viewpoint of an embedded system, m_0 and m_i are the value stored in the threshold register.

Even though the clock frequency suffers from environmental conditions, the deviations of $\chi_i(t)$ can also be seen as a constant value during a sufficiently short clock update period. The term $(\int_0^{t[n]} \chi_i(\tau)d\tau)/f_0$ is discretised as $\sum_{h=0}^{n-1} \chi_i[h]\tau_0$. By considering $1/2\pi f_0$ as a scaling factor, $\phi_i(t[n])/(2\pi f_0)$ is modified to a discrete form $\phi_i[n]$ [21]. Thus, we have the following form of $P_i[n]$

$$P_i[n] = n\tau_0 + \frac{\sum_{h=0}^{n-1} \chi_i[h]\tau_0}{f_0} + \phi_i[n] - \sum_{h=1}^{\lfloor \frac{n}{m_i} \rfloor} \varphi_0, \quad (6)$$

Let the clock offset $\theta_i[n]$ denote the difference between $P_i[n]$ and $t[n]$ (i.e. $\theta_i[n] \triangleq P_i[n] - t[n]$). $\theta_i[n]$ is given by

$$\theta_i[n] = \frac{\sum_{h=0}^{n-1} \chi_i[h]\tau_0}{f_0} + \phi_i[n] - \sum_{h=1}^{\lfloor \frac{n}{m_i} \rfloor} \varphi_0. \quad (7)$$

Note that the variable $\chi_i[n]$ in [8] and [9] is a constant value, which varies in this paper. The work in [10] implicitly assumes that the auxiliary variable m_i is equal to m_0; we remove this assumption to let the clock offset $\theta_i[n]$ in (7) be a general case.

The clock skew $\gamma_i[n] \triangleq \chi_i[n]/f_0$ is utilised to represent the normalised difference between $f_i[n]$ and f_0. Moreover, as a result of the clock's counter resetting behaviour, if a drifting clock resets between two consecutive update events, the value of $(\sum_{h=1}^{\lfloor (n+1)/m_i \rfloor} \varphi_0 - \sum_{h=1}^{\lfloor n/m_i \rfloor} \varphi_0)$ is φ_0; otherwise, it always equals zero. Moreover, we also consider the offset fluctuation as a random process, with the stationary increment represented by a zero-mean random variable [20], yielding $\phi_i[n+1] = \phi_i[n] + \overline{\omega}_{\theta_i}[n]$. The recursive form clock offset is obtained:

$$\theta_i[n+1] = \begin{cases} \theta_i[n] + \gamma_i[n]\tau_0 + \overline{\omega}_{\theta_i}[n], & \text{if } \frac{n+1}{m_i} \notin \mathbb{N}^+ \\ \theta_i[n] + \gamma_i[n]\tau_0 + \overline{\omega}_{\theta_i}[n] - \varphi_0, & \text{if } \frac{n+1}{m_i} \in \mathbb{N}^+ \end{cases}, \quad (8)$$

where $(n+1)/m_i \in \mathbb{N}^+$ implies the occurrence of clock counter resetting. $\overline{\omega}_{\theta_i}[n]$ is the white Gaussian random noise process with the standard deviation of σ_{θ_i} [20]. $\phi_i[n]$ indicates the discrete-time white frequency noise.

By modelling the internal RC oscillator clock frequency as a time-varying process in an Auto-Regressive (AR) manner with perturbations, the AR solution in the second-order model can precisely describe how the skew fluctuation evolves in time. Similar to [20], the skew fluctuation is assumed as a random process. The following expression can be used to model the drifting clock skew

$$\gamma_i[n+1] = p\gamma_i[n] + \overline{\omega}_{\gamma_i}[n], \quad (9)$$

where the parameter p is a positive number, which is less than but close to 1 [23]. $\overline{\omega}_{\gamma_i}[n]$ indicates the zero-mean white Gaussian random noise process with the standard deviation of σ_{γ_i}. $\gamma_i[n]$ denotes the random-walk frequency noise.

From (8) and (9), the recursive equations that account for the behaviour of a drifting internal RC oscillator clock at successive clock events are written as

$$\begin{cases} \theta_i[n+1] = \begin{cases} \theta_i[n] + \gamma_i[n]\tau_0 + \overline{\omega}_{\theta_i}[n], & \text{if } \frac{n+1}{m_i} \notin \mathbb{N}^+ \\ \theta_i[n] + \gamma_i[n]\tau_0 + \overline{\omega}_{\theta_i}[n] - \varphi_0, & \text{if } \frac{n+1}{m_i} \in \mathbb{N}^+ \end{cases} \\ \gamma_i[n+1] = p\gamma_i[n] + \overline{\omega}_{\gamma_i}[n]. \end{cases} \quad (10)$$

The above (10) is a one-step clock update model, which means that the drifting clock is updated at each event n (with the interval of one clock period τ_0). In the meanwhile, the occurrence of internal RC oscillator clock's counter resetting takes place at the clock event of $n = \{m_i, 2m_i, 3m_i, ..., km_i\}$. For theoretical analysis, we extend our one-step update model (10) to a m_i-step update model. Specifically, the i-th drifting clock reaches the threshold value φ_0 at the n-th event, where $n = \{m_i, 2m_i, 3m_i, ..., km_i\}$, $\theta_i[n+m_i]$ and $\gamma_i[n+m_i]$ are expressed as

$$\begin{cases} \theta_i[n+m_i] = \theta_i[n] + H\gamma_i[n] + F_\gamma \widetilde{\omega}_{\gamma_i}[n] + L_\theta \widetilde{\omega}_{\theta_i}[n] - \varphi_0 \\ \gamma_i[n+m_i] = p^n \gamma_i[n] + L_\gamma \widetilde{\omega}_{\gamma_i}[n] \end{cases} \quad (11)$$

where $\widetilde{\omega}_{\theta_i}[n] = [\overline{\omega}_{\theta_i}[n], \overline{\omega}_{\theta_i}[n+1], ..., \overline{\omega}_{\theta_i}[n+m_i-1]]^T \in \mathbb{R}^{m_i \times 1}$, and $\widetilde{\omega}_{\gamma_i}[n] = [\overline{\omega}_{\gamma_i}[n], \overline{\omega}_{\gamma_i}[n+1], ..., \overline{\omega}_{\gamma_i}[n+m_i-1]]^T \in \mathbb{R}^{m_i \times 1}$. H and three matrices L_θ, L_γ and F_γ are, respectively, equal to $H = \sum_{h=0}^{m_i-1} p^h \tau_0$, $L_\theta = [1, 1, ..., 1] \in \mathbb{R}^{1 \times m_i}$, $L_\gamma = [p^{m_i-1}, p^{m_i-2}, ..., 1] \in \mathbb{R}^{1 \times m_i}$, and $F_\gamma = [\sum_{h=0}^{m_i-2} p^h \tau_0, \sum_{h=0}^{m_i-3} p^h \tau_0, ..., \sum_{h=0}^{m_i-m_i} p^h \tau_0, 0] \in \mathbb{R}^{1 \times m_i}$.

Through defining $n = km_i$, the clock model (11) is modified to

$$\begin{cases} \theta_i[(k+1)m_i] = \theta_i[km_i] + H\gamma_i[km_i] + F_\gamma \widetilde{\omega}_{\gamma_i}[km_i] \\ \qquad\qquad + L_\theta \widetilde{\omega}_{\theta_i}[km_i] - \varphi_0 \\ \gamma_i[(k+1)m_i] = p^{m_i} \gamma_i[km_i] + L_\gamma \widetilde{\omega}_{\gamma_i}[km_i] \end{cases}, \quad (12)$$

where $\widetilde{\omega}_{\theta_i}[km_i] = [\overline{\omega}_{\theta_i}[km_i], \overline{\omega}_{\theta_i}[km_i+1], ..., \overline{\omega}_{\theta_i}[(k+1)m_i-1]]^T \in \mathbb{R}^{m_i \times 1}$. $\widetilde{\omega}_{\gamma_i}[km_i] = [\overline{\omega}_{\gamma_i}[km_i], \overline{\omega}_{\gamma_i}[km_i+1], ..., \overline{\omega}_{\gamma_i}[(k+1)m_i-1]]^T \in \mathbb{R}^{m_i \times 1}$.

Let $\theta_i[km_i] = \theta_i[k]$, $\gamma_i[km_i] = \gamma_i[k]$, $\widetilde{\omega}_{\theta_i}[km_i] = \widetilde{\omega}_{\theta_i}[k]$, and $\widetilde{\omega}_{\gamma_i}[km_i] = \widetilde{\omega}_{\gamma_i}[k]$. Eventually, from (12), the non-identical and time-varying clock model is obtained

$$\begin{cases} \theta_i[k+1] = \theta_i[k] + H\gamma_i[k] + F_\gamma \widetilde{\omega}_{\gamma_i}[k] + L_\theta \widetilde{\omega}_{\theta_i}[k] - \varphi_0 \\ \gamma_i[k+1] = p^{m_i} \gamma_i[k] + L_\gamma \widetilde{\omega}_{\gamma_i}[k] \end{cases}. \quad (13)$$

Until this point, we have provided the analytical expression of a poor-performing internal RC oscillator clock. In the following, we use this recursive form model to construct a closed-loop synchronisation system for P-PkCOs robustness analysis, where the stability region of controller parameters also is given (see Section 4). Note that the uncoupled oscillator (possessing the periodic firing and resetting behaviour) in P-PkCOs is equivalent to the drifting clock model (13), consisting of an internal RC oscillator and a counter register.

The proposed two-state clock model can characterise the behaviour of the white Frequency Modulation (FM) noise and random-walk FM noise [see (8) and (9)]. Their impacts in an Allan variance plot are, respectively, represented by regions with slopes of -1 and $+1$. Both [20] and [24] point out that the finite-order state model fails to model the flicker FM noise (the slope of 0 in the Allan variance plot); the

proposed second-order non-identical and time-varying clock model cannot include the flicker FM noise. In addition, for the Phase Modulation (PM) noise (the slope of the Allan variance plot is -2), its impacts are mainly related to the short-term (e.g. less than 10ms in [20]) clock frequency stability. The clock update period τ_0 in the simulations is not small than 10ms (i.e. $\tau_0 \geq 10$ms), and the time synchronisation cycle T is 1 second. As a result, the P-PkCOs protocol possesses a tiny effect on the PM noise, and we also do not consider this noise.

B. Timestamped Packet-exchange Mechanism

Next, in order to measure the internal RC oscillator clock offset and skew of a sensor node, the P-PkCOs protocol utilises a timestamped packet-exchange scheme. This method possesses the features of the reduced bandwidth demand in the Radio frequency (RF) communication, as only one *Sync* packet is sent and received during each synchronisation cycle. Specifically, at the k-th cycle, when the master's clock reaches φ_0, a *Sync* packet is transmitted to the wireless channel. Upon receiving *Sync* from the master node, node i generates a local timestamp $\hat{P}_i[k]$ via reading its internal RC oscillator clock module, and only this local timestamp is needed for offset and skew calculation, since the transmission of a *Sync* packet contains the timing information (i.e. the master clock arriving φ_0). The variable $\hat{P}_i[k]$ satisfies the following form

$$\hat{P}_i[k] = P_i[k] + \Delta P_i[k], \tag{14}$$

where the random variable $\Delta P_i[k]$ is the timestamp uncertainty resulting from different timestamping mechanisms (e.g. hardware and software timestamping) [20].

Once the timestamp $\hat{P}_i[k]$ is obtained, the offset estimate $\hat{\theta}_i[k]$ between node i and the master node is calculated from

$$\hat{\theta}_i[k] = \begin{cases} \hat{P}_i[k], & \text{if } \hat{P}_i[k] < \frac{\varphi_i}{2} \\ \hat{P}_i[k] - \varphi_i, & \text{if } \hat{P}_i[k] \geq \frac{\varphi_i}{2} \end{cases}. \tag{15}$$

Based on (7), it can be seen that the existence of clock offset leads to the inaccuracy of the time variable. Hence, according to (14), the offset estimate also is re-written as the following expression:

$$\hat{\theta}_i[k] = \theta_i[k] + \nu_{\theta_i}[k], \tag{16}$$

where $\nu_{\theta_i}[k] = \Delta P_i[k]$ is the offset estimate noise.

The only employment of $\hat{\theta}_i[k]$ to each internal RC oscillator clock fails to maintain time synchronisation, or even to achieve synchronisation, owing to the large-drifting clock frequency [19]. Thus, this work estimates the skew for clock adjustment purposes. Let $\hat{\gamma}_i[k]$ represent the skew estimate between node i and the master node, and it can be calculated from $\hat{\gamma}_i[k] = (\hat{\theta}_i[k] - \hat{\theta}_i[k-1]^+)/T$. However, the actual offset $\hat{\theta}_i[k-1]^+$ after clock correction is unknown. The clock offset converges to zero at synchronised state, and we assume that the clock offset is perfectly corrected (i.e. $\hat{\theta}_i[k-1]^+ = 0$). The skew estimate $\hat{\gamma}_i[k]$ is obtained from

$$\hat{\gamma}_i[k] = \frac{\hat{\theta}_i[k]}{T}. \tag{17}$$

Likewise, for theoretical study, $\hat{\gamma}_i[k]$ is given by

$$\hat{\gamma}_i[k] = \gamma_i[k] + \nu_{\gamma_i}[k], \tag{18}$$

where $\nu_{\gamma_i}[k] = \Delta P_i[k]/T$ is the skew measurement noise.

The aim of synchronisation is to keep the offset and skew of each drifting clock as close to zero as possible. That is

$$\lim_{k\to\infty} \theta_i[k] = 0, \text{and } \lim_{k\to\infty} \gamma_i[k] = 0. \tag{19}$$

In the next section, a proportional clock correction solution is proposed to achieve time synchronisation on low-accuracy internal RC oscillator clocks.

III. Proportional Clock Correction Scheme

In the wireless network, the over-correction may occur, and the time synchronisation performance degrades, owing to timestamp inaccuracy. To solve the issue above, we adopt a proportional controller to adjust the local clock:

$$\begin{cases} \theta_i[k]^+ = \theta_i[k]^- + u_{\theta_i}[k] \\ \gamma_i[k]^+ = \gamma_i[k]^- + u_{\gamma_i}[k] \end{cases}, \tag{20}$$

where $\theta_i[k]^+/\theta_i[k]^-$ is the clock offset after/before the clock is corrected at the k-th synchronisation cycle. $\gamma_i[k]^+/\gamma_i[k]^-$ is the clock skew after/before the drifting clock is adjusted. The correction inputs $u_{\theta_i}[k]$ and $u_{\gamma_i}[k]$ are, respectively, equal to

$$\begin{cases} u_{\theta_i}[k] = -\alpha\hat{\theta}_i[k] \\ u_{\gamma_i}[k] = -\beta\hat{\gamma}_i[k] \end{cases}, \tag{21}$$

where α and β are parameters of the proportional controller. A closed-loop synchronisation system is considered for robustness performance analysis. The progress of deriving such a closed-loop system is presented in the following. Let's firstly re-write (13) in the matrix-vector form

$$x_i[k+1] = A^{m_i}x_i[k] + \omega_i[k] + R\varphi_0, \tag{22}$$

where $x_i[k] = [\theta_i[k], \gamma_i[k]]^T$ is the clock state vector. $\omega_i[k] = [\widetilde{\omega}_{\theta_i}^T[k]L_\theta^T + \widetilde{\omega}_{\gamma_i}^T[k]F_\gamma^T, \widetilde{\omega}_{\gamma_i}^T[k]L_\gamma^T]^T$ represents the clock noise vector. Two matrices A and R are, respectively, equal to

$$A = \begin{bmatrix} 1 & \tau_0 \\ 0 & p \end{bmatrix}, \text{and } R = \begin{bmatrix} -1 \\ 0 \end{bmatrix}.$$

By introducing the correction input vector $u_i[k] = [u_{\theta_i}[k], u_{\gamma_i}[k]]^T$ into (22), it is modified to

$$x_i[k+1] = A^{m_i}x_i[k] + Bu_i[k] + \omega_i[k] + R\varphi_0, \tag{23}$$

where B is the 2×2 identity matrix. The input vector $u_i[k]$ satisfies the following expression

$$u_i[k] = K(\boldsymbol{r} - y_i[k]), \tag{24}$$

where $\boldsymbol{r}$ is the 2×1 reference input vector, and the control matrix K equals $K = \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix}$. From (16) and (18), the clock observation vector $y_i[k] = [\hat{\theta}_i[k], \hat{\gamma}_i[k]]^T$ is obtained

$$y_i[k] = Cx_i[k] + \nu_i[k], \tag{25}$$

where C is the 2×2 identity matrix. $\nu_i[k] = [\nu_{\theta_i}[k], \nu_{\gamma_i}[k]]^T$ is the vector of two zero-mean Gaussian random variables $\nu_{\theta_i}[k]$

and $\nu_{\gamma_i}[k]$ representing the estimate uncertainties [20], whose standard deviations correspond to $\sigma_{\hat{\theta}_i}$ and $\sigma_{\hat{\gamma}_i}$, respectively.

As shown in the clock modelling, synchronising an internal RC oscillator clock with the master clock can be modelled as the feedback control procedure consisting of (23), (24) and (25). Since the synchronisation is to keep the deviation between two clocks as close to zero as possible, it implies that the reference input, denoted by a vector $\boldsymbol{r}$, is $\mathbf{0}$ (i.e. $\boldsymbol{r} = [0, 0]^T$), which means that the difference between two clocks is zero at steady state. Then the closed-loop synchronisation system is given by

$$\begin{cases} x_i[k+1] = A^{m_i} x_i[k] + B u_i[k] + \omega_i[k] + R\varphi_0 \\ y_i[k] = C x_i[k] + \nu_i[k] \\ u_i[k] = -K y_i[k] \end{cases}. \quad (26)$$

Note that the above equation also mathematically shows the framework of P-PkCOs, consisting of an oscillator (23), and the timestamped packet-exchange scheme (25) with the proportional controller (24).

To analyse the robustness of the P-PkCOs protocol, the system (26) is re-written as

$$\begin{cases} x_i[k+1] = \bar{A} x_i[k] + \eta_i[k] \\ o_i[k] = \bar{C} x_i[k] \end{cases}, \quad (27)$$

where $o_i[k]$ is the performance output vector. $\eta_i[k] = -BK\nu_i[k] + \omega_i[k] + R\varphi_i$ is the disturbance vector. The matrices $\bar{A}$ and $\bar{C}$ respectively equal

$$\bar{A} = \begin{bmatrix} 1-\alpha & T \\ 0 & p^{m_i} - \beta \end{bmatrix}, \ \bar{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

So far, we have introduced the proportional clock correction method to adjust the offset and skew for achieving and maintaining time synchronisation on low-accuracy internal RC oscillator clocks. The closed-loop synchronisation system also has been established. Based on this closed-loop system, the following section shows the stability and robustness analysis of the P-PkCOs protocol in a single-cluster wireless network, which also guides the parameter selection of the P controller.

IV. Stability and Robustness Analysis

This section presents the synchronisation condition for a single-cluster wireless network. If the proposed condition is satisfied, the analysis indicates that the wireless network system is robust in the presence of disturbances $\eta_i[k]$ caused by large-drifting clock frequency and timestamp uncertainty. This also means that, by using the proposed P-PkCOs method, the internal RC oscillator clocks of sensor nodes can maintain robust synchronisation with the master clock in the network.

Definition 1: The closed-loop synchronisation system is asymptotically stable, if both offset and skew approach zero, namely, $\theta_i[\infty] = 0$, and $\gamma_i[\infty] = 0$.

Definition 2: The transfer function of (27) is $G[z] = \bar{C}(zI - \bar{A})^{-1}$. The performance H_∞ of (27) is guaranteed, if the infinity norm $\|G[z]\|_\infty$ of the transfer function $G[z]$, which is the minimal upper bound (i.e. supremum) of the largest singular number of $G[z]$ over the unit circle, and also equals the two-norm ratio between $o_i[k]$ and $\eta_i[k]$, is less than ρ. That is

$$\begin{aligned} \|G[z]\|_\infty &= \max_{0 \le \vartheta \le 2\pi} \left\| G(e^{j\vartheta}) \right\| \\ &= \frac{\|o_i\|_2}{\|\eta_i\|_2} = \sup_{\|\eta_i\|_2 \le 1} \|o_i\|_2 < \rho. \end{aligned} \quad (28)$$

where the two-norms of $o_i[k]$ and $\eta_i[k]$ are $\|o_i\|_2 = (\sum_{h=0}^{\infty} \|o_i[k]\|^2)^{\frac{1}{2}}$ and $\|\eta_i\|_2 = (\sum_{h=0}^{\infty} \|\eta_i[k]\|^2)^{\frac{1}{2}}$, respectively.

Theorem 1: Given a single-cluster network system $\mathcal{G}$, consisting of an ideal master clock and N sensor nodes' internal RC oscillator clocks with the drifting frequencies $\gamma_i[k]$, a scalar $\rho > 0$ exists, and the prescribed performance H_∞ is guaranteed, if α and β of the proportional controller satisfy the following condition:

$$\alpha \in (0, 2), \ \beta \in (0, 2); \quad (29)$$

and there also exists a symmetric matrix U such that

$$\left[\begin{array}{cc|c} \begin{bmatrix} -U & 0 \\ 0 & -\rho^2 I \end{bmatrix} & & * \\ \hline \multicolumn{2}{c|}{\begin{bmatrix} U\bar{A} & U \\ \bar{C} & 0 \end{bmatrix}} & \begin{bmatrix} -U & 0 \\ 0 & -I \end{bmatrix} \end{array}\right] < 0. \quad (30)$$

Proof: The single-cluster network $\mathcal{G}$ can be decomposed into N two-dimensional systems (26). For an arbitrary closed-loop system (26), let p and T be equal to 1, the characteristic equation of (26) is

$$(\lambda_1 - 1 + \alpha)(\lambda_2 - 1 + \beta). \quad (31)$$

The eigenvalues of the characteristic equation above are

$$\lambda_1 = 1 - \alpha, \ \lambda_2 = 1 - \beta \quad (32)$$

respectively. As specified in [25], when these two eigenvalues $\{\lambda_1, \lambda_2\}$ are within the unit circle, the system (26) is stable.

Through applying the Schur complement to (30), the following linear matrix inequality is obtained:

$$\begin{bmatrix} U & 0 \\ 0 & \rho^2 I \end{bmatrix} > \begin{bmatrix} \bar{A} & I \\ \bar{C} & 0 \end{bmatrix}^T \begin{bmatrix} U & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \bar{A} & I \\ \bar{C} & 0 \end{bmatrix}. \quad (33)$$

We can easily verify that (33) is feasible under the proportional controlling strategy with $\alpha \in (0, 2)$ and $\beta \in (0, 2)$ [26]. Since the matrix inequality (33) holds strictly, there always exists a scalar value ι, where $0 < \iota < 1$, such that

$$\begin{bmatrix} U & 0 \\ 0 & (1-\iota)\rho^2 I \end{bmatrix} > \begin{bmatrix} \bar{A} & I \\ \bar{C} & 0 \end{bmatrix}^T \begin{bmatrix} U & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \bar{A} & I \\ \bar{C} & 0 \end{bmatrix} \quad (34)$$

holds.

From (34), we find that $U > \bar{A}^T U \bar{A}$; hence, the closed-loop system is asymptotically stable [which is consistent with the analysis in (32)], and the expression

$$V_i[k] = x_i^T[k] U x_i[k] \quad (35)$$

is a Lyapunov function of the system (27). In order to prove that the supremum norm of $\|o_i\|_2^2$ is $\rho^2 \|\eta_i\|_2^2$, by defining

$\Delta V_i[k] = V_i[k+1] - V_i[k]$, $J_k = \|o_i[k]\|^2 - (1-\iota)\rho^2\|\eta_i[k]\|^2$ is re-written as

$$\begin{aligned} J_k &= \left(\|o_i[k]\|^2 - (1-\iota)\rho^2\|\eta_i[k]\|^2 + \Delta V_i[k]\right) - \Delta V_i[k] \\ &= \begin{bmatrix} x_i[k] \\ \eta_i[k] \end{bmatrix}^T \left(\begin{bmatrix} \bar{C} & 0 \end{bmatrix}^T \begin{bmatrix} \bar{C} & 0 \end{bmatrix} - \begin{bmatrix} U & 0 \\ 0 & (1-\iota)\rho^2 \end{bmatrix} \right. \\ &\quad \left. + \begin{bmatrix} \bar{A} & I \end{bmatrix}^T \begin{bmatrix} \bar{A} & I \end{bmatrix} \right) \begin{bmatrix} x_i[k] \\ \eta_i[k] \end{bmatrix} - \Delta V_i[k] \\ &= \begin{bmatrix} x_i[k] \\ \eta_i[k] \end{bmatrix}^T \left(\begin{bmatrix} \bar{A} & I \\ \bar{C} & 0 \end{bmatrix}^T \begin{bmatrix} U & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \bar{A} & I \\ \bar{C} & 0 \end{bmatrix} \right. \\ &\quad \left. - \begin{bmatrix} P & 0 \\ 0 & (1-\iota)\rho^2 I \end{bmatrix} \right) \begin{bmatrix} x_i[k] \\ \eta_i[k] \end{bmatrix} - \Delta V_i[k]. \end{aligned} \tag{36}$$

Thus, $J_k < -\Delta V_i[k]$, and the following inequality is also given

$$\|o_i[k]\|^2 < (1-\iota)\rho^2\|\eta_i[k]\|^2 - \Delta V_i[k]. \tag{37}$$

Next, we sum from $j = 0$ to $j = k$ on both sides of the inequality above. Based on the zero initial condition and energy-bounded disturbances [26], the following inequality is obtained:

$$\begin{aligned} \sum_{j=0}^{k} \|o_i[j]\|^2 &< (1-\iota)\rho^2 \sum_{j=0}^{k} \|\eta_i[j]\|^2 - V_i[k+1] \\ &< (1-\iota)\rho^2\|\eta_i\|_2^2 - V_i[k+1]. \end{aligned} \tag{38}$$

The system (27) is asymptotically stable, which means that $x_i[k]$ approaches zero at steady synchronisation state (i.e. $\lim_{j\to\infty} x_i[j] = 0$). From (38), we have

$$\|o_i\|_2^2 < (1-\iota)\rho^2\|\eta_i[k]\|_2^2 < \rho^2\|\eta_i\|_2^2. \tag{39}$$

For an arbitrary two-dimensional system (26), according to (39), the prescribed performance H_∞ is guaranteed (i.e. $\sup_{\|\eta_i\|_2 \le 1} \|o_i\|_2 < \rho$). In this paper, $\mathcal{G}$ is a directed single-cluster topology. The networked system can also achieve robustness in the presence of disturbances from the drifting clock frequency and timestamp uncertainty. That is to say, in the single-cluster wireless network, by using the P-PkCOs protocol with the proportional controller parameters obtained from Theorem 1, the sensor node's low-accuracy internal RC oscillator clock can maintain robust synchronisation with the master clock. ■

The internal RC oscillator frequency (or skew) in an embedded system cannot be corrected. Fortunately, as the clock frequency and threshold meet the inverse proportionality, the clock threshold adjustment is considered as a substitute for skew correction, following

$$\varphi_i[k]^+ = \varphi_i[k]^- + \beta\hat{\theta}_i[k] \tag{40}$$

where $\varphi_i[k]^+/\varphi_i[k]^-$ is the threshold after/before the i-th clock is corrected at the k-th synchronisation cycle. The following Theorem 2 shows the relationship between the skew and threshold at steady synchronised state.

Theorem 2: Given a single-cluster network $\mathcal{G}$, consisting of a master clock and N sensor node internal RC oscillator clocks with the average skews $\bar{\gamma}_i$, if and only if the parameters α and β satisfy the following condition:

$$\alpha \in (0,2),\ \beta \in (0,2), \tag{41}$$

the i-th sensor node's internal RC oscillator clock synchronises with the master clock. At steady synchronised state, the clock threshold follows

$$\lim_{k\to\infty} \varphi_i[k] = \varphi_0(1+\bar{\gamma}_i). \tag{42}$$

Proof: By substituting $f_i[k] = \bar{f}_i + \Delta f_i[k]$ and $\gamma_i[k] = \bar{\gamma}_i + \Delta\gamma_i[k]$ into $f_i[k] = f_0 + f_0\gamma_i[k]$, we can obtain

$$\bar{f}_i + \Delta f_i[k] = f_0 + f_0(\bar{\gamma}_i + \Delta\gamma_i[k]), \tag{43}$$

where $\Delta f_i[k]$ is the deviation of $f_i[k]$ from its mean value $\bar{f}_i$. $\Delta\gamma_i[k]$ represents the deviation of $\gamma_i[k]$ from the mean value $\bar{\gamma}_i$.

Since the reciprocal of the clock frequency is period, the following expression is also given

$$\tau_i[k] = \frac{f_0\tau_0}{f_i[k]}. \tag{44}$$

The clock threshold adjustment is utilised as a substitute for skew correction, at steady state, $m_i[k]$ is adjusted to

$$\begin{aligned} \lim_{k\to\infty} m_i[k] &= \lim_{k\to\infty} \frac{\varphi_0}{\tau_i[k]} \\ &= \lim_{k\to\infty} \left(\frac{m_0\varphi_0\left(f_0 + f_0\bar{\gamma}_i\right)}{f_0\varphi_0} + \frac{m_0\varphi_0\Delta\gamma_i[k]}{f_0\varphi_0} \right) \\ &= \lim_{k\to\infty} \left(m_0\left(1+\bar{\gamma}_i\right) + \frac{m_0\Delta\gamma_i[k]}{f_0} \right). \end{aligned} \tag{45}$$

$\Delta\gamma_i[k]$ is assumed to be negligible. Once the i-th internal RC oscillator clock synchronises with the master clock, $m_i[k]$ approaches to

$$\lim_{k\to\infty} m_i[k] = m_0(1+\bar{\gamma}_i). \tag{46}$$

Thus, the threshold $\varphi_i[k] = m_i[k]\tau_0$ is close to

$$\lim_{k\to\infty} \varphi_i[k] = \varphi_0(1+\bar{\gamma}_i). \tag{47}$$

■

Theorem 2 implies that once α and β of the proportional correction solution satisfy the conditions of (41), both offset and skew approach zero at steady state, and all the sensor nodes in a single-cluster wireless network can maintain time synchronisation on internal RC oscillator clocks. However, due to the difficulty of correcting internal RC oscillator frequency in an embedded system, the clock threshold adjustment (40) is considered as a substitute. Thus, at steady synchronised state, the threshold $\varphi_i[k]$ converges to the value of $\varphi_0(1+\bar{\gamma}_i)$, which is only determined by the average value of the corresponding clock skew $\bar{\gamma}_i$ and the nominal threshold φ_0. Furthermore, Theorem 1 indicates that, for any pair of α and β falling in the range of (29) [or (41)], which also satisfy the condition of (30), the networked system is robust in the presence of the dominant clock noises and the timestamp uncertainties.

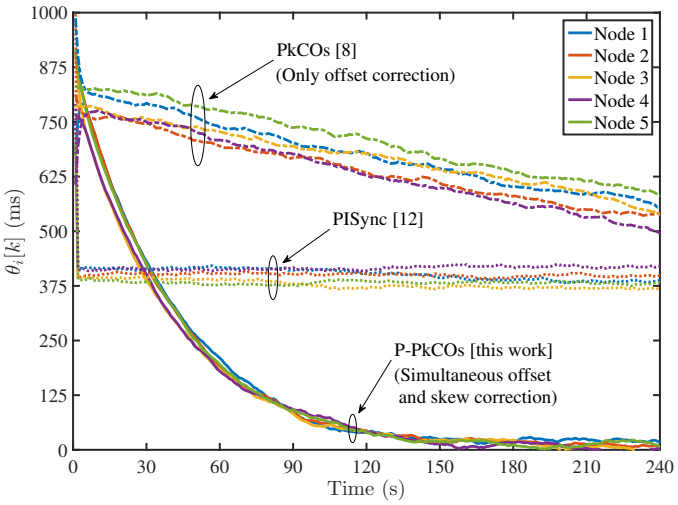


Fig. 2. Evolution of the clock offset via PkCOs, PISync and P-PkCOs.

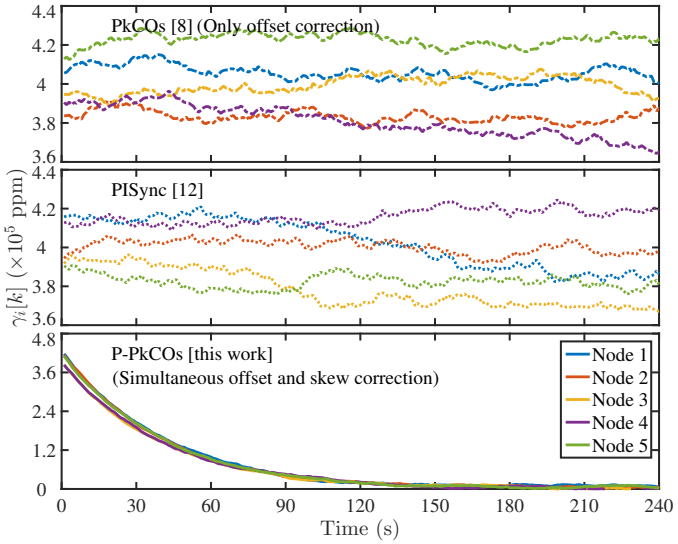


Fig. 3. Evolution of the clock skew via PkCOs, PISync and P-PkCOs.

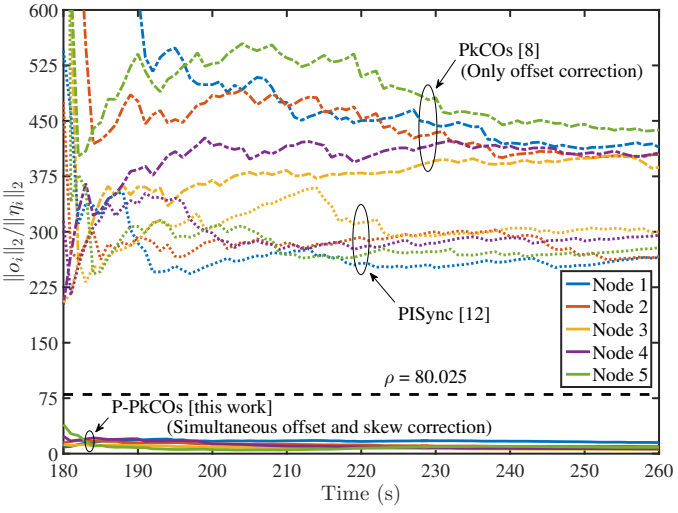


Fig. 4. Evolution of $\frac{\|o_i\|_2}{\|\eta_i\|_2}$ through PkCOs, PISync and P-PkCOs.

V. Simulation Results

This section uses numerical simulations to validate the theoretical results presented in Section 4. In the simulations, the non-identical and time-varying clock with the initial clock offset $\theta_i[0] = 0.8$ seconds and the skew of $\gamma_i[0] = 4 \times 10^5$ppm is adopted (see Table 3). The clock offset and skew are subjected to random perturbations with standard deviations of 1ms and 1000ppm. The standard deviation of the timestamp uncertainty is $4\mu s$ [8]. The synchronisation cycle is 1 second. α and β are equal to 0.5 and 0.025, respectively. The single-cluster network, consisting of one master node and five sensor nodes, is simulated. Note that the term φ_0 in (8) is not included in the simulations.

Figs. 2 and 3 show the evolution of clock offset and skew during the simulations. In Fig. 2, two methods (i.e. P-PkCOs and PISync) guarantee that the clock offset converges to a constant value at steady synchronised state. The large parameter $\alpha = 1$ is utilised in PISync, and its convergence speed is faster than that of P-PkCOs. Even though PISync adopts the adaptive β tuning solution, the order-of-magnitude of β is still minuscule (around 10^{-4}) in the simulations. Thus, the PISync protocol cannot overcome the effects of drifting clock skew (with the standard deviation of 1000ppm), as shown in Fig. 3. The clock skew in PISync fails to approach zero, and the clock offset is only about 400ms at steady state. We adopt the clock offset between 180 and 240 seconds to calculate the average value and standard deviation. The offset mean values in P-PkCOs and PISync are equal to 11.61ms and 391.12ms, respectively, and the corresponding standard deviations are 5.48ms and 2.36ms.

In our prior work PkCOs, although an integral controller is used for clock offset correction, $\alpha = 0.5$ in the proportional controller (for clock offset adjustment), and no clock skew correction cannot let the clock offset approach zero during the 240-second simulations (see Fig. 2). However, by using P-PkCOs with simultaneous offset and skew correction, and the proportional controller's parameters obtained from Theorem 1, the clock offset and skew are close to zero at synchronised state.

Fig. 4 presents the evolution of $\frac{\|o_i\|_2}{\|\eta_i\|_2}$ over time. $\frac{\|o_i\|_2}{\|\eta_i\|_2}$ is calculated according to (28). It can be seen that, for the P-PkCOs protocol, $\frac{\|o_i\|_2}{\|\eta_i\|_2}$ changes from the initial value to around 16, which is smaller than $\rho = 80.025$ calculated from Theorem 1 (the dashed line in Fig. 4). $\frac{\|o_i\|_2}{\|\eta_i\|_2}$ in PISync and PkCOs are about 300 and 450, respectively, since these two algorithms cannot let the offset and skew approach zero. By using P-PkCOs and the proposed LMI condition (i.e. Theorem 1), the single-cluster network can realise time synchronisation, and also is robust in the presence of disturbances from drifting clock frequency and timestamp uncertainty. Note that ρ is for the worst case, and is a conservative value due to the LMI method. Here, we only study the robustness of Theorem 1 using simulations. The clock threshold correction (i.e. Theorem 2) is verified in the experimental results.

In addition, the simulation results of a multi-hop wireless network show that, the P-PkCOs synchronisation precision is poor, due to the use of low-accuracy clocks. The precision

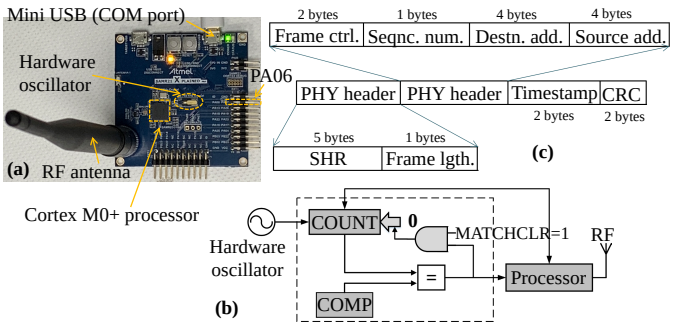


Fig. 5. (a). Prototype of the IEEE 802.15.4 radio board. (b). Structure of the real-time clock module and (c). *Sync* architecture.

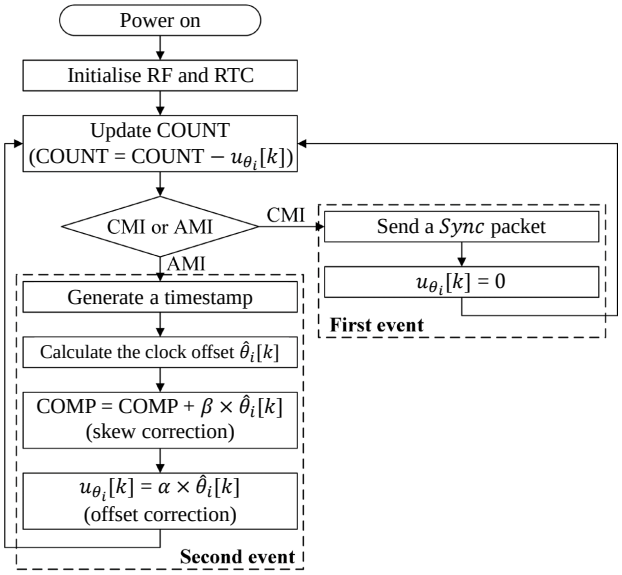


Fig. 6. Flow diagram of a wireless node operation for P-PkCOs synchronisation.

mean value of P-PkCOs reduces from 6.28ms (on the first-hop nodes) to 111.82ms (on the third-hop nodes). The precision standard deviations on the first-hop, second-hop and third-hop nodes, respectively, are 5.10ms, 8.86ms, and 32.24ms. The single-cluster topology is a popular solution for body sensor networks. Thus, according to the above multi-hop simulation results, it is not necessary to study the time synchronisation of low-accuracy internal RC oscillator clocks in multi-hop wireless networks, particularly for the BSNs [1], [4].

VI. Implementation and Experimental Evaluation

This section commences with the implementation of the proposed P-PkCOs synchronisation protocol. Then, we analyse the drift characteristics of low-accuracy internal RC oscillator clocks with a nominal frequency 32.768kHz. The performance of P-PkCOs is studied in a single-cluster wireless network. Moreover, four algorithms (i.e. PISync, PkCOs, RBS and PCO) are also selected and implemented in the same hardware experimental settings for comparison.

A. Implementation on A Hardware Testbed

In order to evaluate the performance of P-PkCOs, the proposed protocol is implemented on the Atmel SMART SAM R21 board consisting of a low-power 32-bit ARM Cortex-M0+ processor and an integrated ultra-low power 2.4GHz ISM band transceiver RF233 (see Fig. 5). In the implementation, the time variable $P_i[n]$ is represented by a 32-bit COUNT register of the Real-Time Clock (RTC) module using the internal RC oscillator (with a nominal frequency 32.768kHz) as the clock source[3]. The RTC's threshold register COMP is adopted to denote φ_i, and COMP is configured to 32767. This means that the RTC threshold is 1 second. The backoff period is set to zero to guarantee the low-latency *Sync* packet transmission. Once the COUNT value matches the threshold register, a 21-byte *Sync* packet (see Fig. 5c) is immediately sent for synchronisation.

Fig. 6 illustrates the node operation flow diagram for P-PkCOs synchronisation. When the supply voltage is applied to a wireless node, the sensor node goes into the initialisation state for initialling the RF and RTC modules. Once the initialisation procedure is completed, the node enters the free-running mode, where the register COUNT increases from zero toward the pre-configured value in the threshold register. During its life cycle, the wireless node almost stays in the free-running mode to count the periodic signal generated by the internal RC oscillator. The wireless node moves into the interactive mode to execute the corresponding missions, while the hardware interrupts [i.e. Compare Match Interrupt (CMI) or Address Match Interrupt (AMI)] are issued. Specifically, the first event is signalled by a compare match interrupt of the RTC module. COUNT is continuously compared with a programmed value stored in the COMP register. Each time a compare match occurs, the hardware Interrupt Service Routine (ISR) corresponding to the RTC clock matching takes place. In this case, the COUNT register is reset to zero, and a *Sync* packet is transmitted (i.e. node fires).

The reception of a *Sync* packet triggers the second event, occurring whenever the destination addressing fields of the received packet match local addresses. As soon as an address match interrupt is issued, the local timestamp is generated by reading the COUNT register. The clock offset estimate $\hat{P}_i[k]$ is utilised for adjusting the drifting clock. The microprocessor can manage only one of two events described above at once, due to the same ISR priority configuration of CMI and AMI. Moreover, it is necessary to check whether the compare match interrupt occurs by using the Nested Vectored Interrupt Controller (NVIC) command during the second event, since a slightly longer time is required for the processor to execute the second event, compared with the time required for the first event. Algorithm 1 shows the pseudocode in an event-driven fashion, which is a straightforward implementation of the flowchart shown in Fig. 6. In addition, **our implementation is publicly available at https://github.com/zongyan/Packet-Coupled-Oscillators**.

[3]Here, we adopt the internal RC oscillator (i.e. OSC32K) as the clock source; the external hardware crystal oscillator (i.e. XOSC32K) in Fig. 5 is only for demonstration purposes, and is not used in this work.

Algorithm 1 Proportional packet-coupled oscillators

Initialisation
  initialise RF and RTC;
Event: clock compare match occurs
  reset COUNT ($P_i[k] = 0$);
  $send\_pkt(Sync)$;
Event: *Sync* address match occurs
  read COUNT ($\hat{P}_i[k] = \text{COUNT}$);
  estimate the clock offset, according to (15):
    if $\hat{P}_i[k] < (\text{COMP}/2)$ **then**
      $\hat{\theta}_i[k] = \hat{P}_i[k]$;
    elseif $\hat{P}_i[k] \geq (\text{COMP}/2)$ **then**
      $\hat{\theta}_i[k] = \hat{P}_i[k] - \text{COMP}$;
    end if
  correct the clock, following
    if $(\hat{P}_i[k] - \alpha \times \hat{\theta}_i[k]) < \text{COMP}$ **then**
      $\text{COMP} = \text{COMP} + \beta \times \hat{\theta}_i[k]$;
      $\text{COUNT} = \hat{P}_i[k] - \alpha \times \hat{\theta}_i[k]$;
    elseif $(\hat{P}_i[k] - \alpha \times \hat{\theta}_i[k]) \geq \text{COMP}$ **then**
      reset COUNT ($P_i[k] = 0$);
      $send\_pkt(Sync)$;
    end if

Note that we only need to implement (21) and (40) on the hardware testbed, and Theorem 1 is used offline to obtain α and β of the proportional controller. The controllers of all the sensor nodes are with the same structure and the same α and β. This means that our method is scalable in a network. FTSP uses the flooding packet transmission strategy and the linear regression method to realise clock synchronisation. The computational complexity of FTSP in a network is $\mathcal{O}(N \times M)$, where N is the total number of sensor nodes in a wireless network, and we assume that the average number of neighbouring nodes for each sensor node is M. The computational complexity in RBS is $\mathcal{O}(N^2)$ for a whole network, as the receiver-to-receiver packet-exchange strategy and linear regression calculation are adopted. In the proposed P-PkCOs protocol, each sensor node only needs to exchange a *Sync* packet with one neighbouring node (i.e. its parent node); according to (21) and (40), only two 32-bit floating multiplication operations are needed on each sensor node. The computational complexity of P-PkCOs in a wireless network is $\mathcal{O}(N)$. Even though the PISync protocol's computational complexity for a network also equals $\mathcal{O}(N)$, every sensor node requires at least two 32-bit floating multiplications, one division and two addition operations.

Next, we evaluate the synchronisation precision of P-PkCOs in a single-cluster wireless network (see Fig. 7). The Trimble Thunderbolt E GPS Disciplined Clock [28], providing the Pulse Per Second (PPS) signal, is connected to the GPIO PA06 pin of the wireless board, which is associated with an external interrupt of the processor. We use the serial port on the radio board for data collection and performance evaluation, as it provides a reliable way to send the register data of wireless boards to the PC.

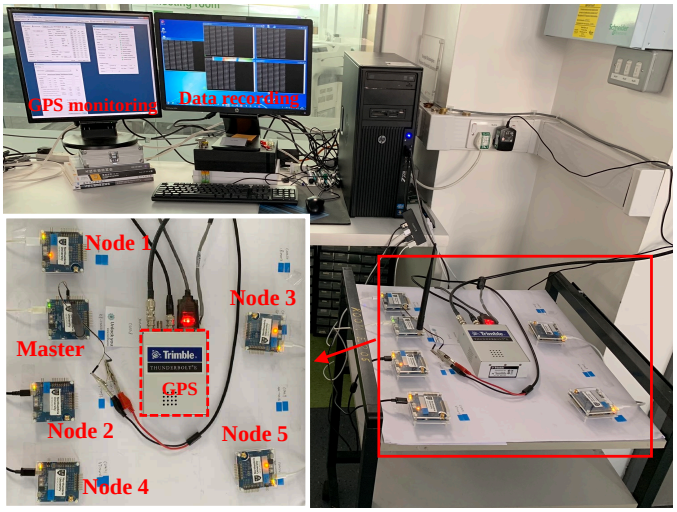


Fig. 7. Hardware testbed.

B. Free-running Internal RC Oscillator Clocks

To gain perspective on the synchronisation performance, the drifting characteristics of poor-performing uncalibrated internal RC oscillator clocks with nominal 32.768kHz frequencies are first examined. During the experiments, the GPS clock is considered as the reference clock. Once the wireless node receives the PPS signal from GPS, the processor issues an external interrupt for reading the COUNT register. When the number of received PPS signals reaches a specific value, saying 30 in the experiments, the clock of each node restarts and all variables used for offline data analysis are reset to zero. After that, the data collected from each wireless sensor node is sent to the PC, and the clock offset is calculated. In addition, the data collected from the wireless node consists of a series of data groups. Each group contains a 32-bit hexadecimal number corresponding to the value of COUNT on each node, the number of received PPS signals, the clock COUNT resetting times, and a 32-bit hexadecimal value stored in the COMP register.

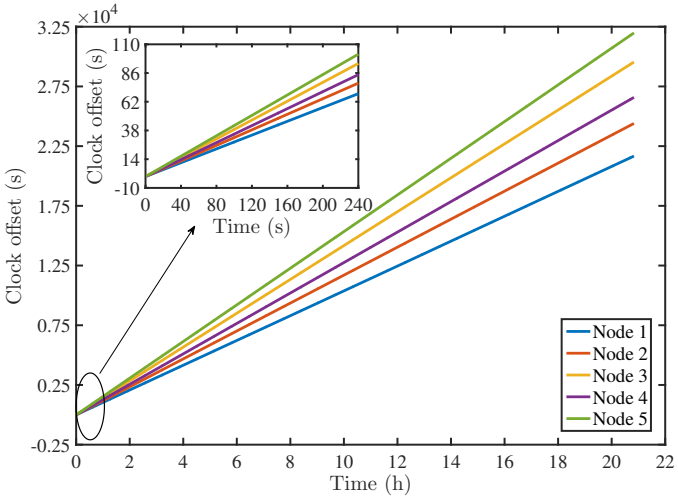


Fig. 8. Offsets of internal RC oscillator clocks on 5 sensor nodes within the 24-hour experiments.

TABLE III
CLOCK SKEWS AND INITIAL OFFSETS OF UNCALIBRATED INTERNAL RC OSCILLATOR CLOCKS ON 5 WIRELESS NODES DURING THE 24-HOUR EXPERIMENTS

	Clock skew $\bar{\gamma}_i$ (ppm)	Initial offset $\theta_i[0]$ (ms)
Node 1	2.889×10^5	-1.16×10^4
Node 2	3.253×10^5	-5.082×10^3
Node 3	3.939×10^5	-2.895×10^3
Node 4	3.544×10^5	-1.781×10^3
Node 5	4.264×10^5	-2.409×10^3

Fig. 8 shows the evolution of the COUNT value difference (i.e. clock offset) between the master and sensor nodes throughout the 24-hour experiments. Each node's clock demonstrates different drifting characteristics (e.g. non-identical and time-varying skew). By using the linear polynomial fitting, the clock skew's average values are summarised in Table 3. The internal RC oscillator clock skew is around 4×10^5ppm. This means that such an embedded clock drifts about 400ms per second. In other words, without clock synchronisation methods, the offset increment of internal RC oscillator clocks is about 110 seconds during the 240 seconds, and is around 3.25×10^4 seconds in the 24-hour experiments.

C. Synchronisation of Internal RC Oscillator Clocks

Next, we study the synchronisation precision of P-PkCOs, PISync, PkCOs, PCO and RBS on internal RC oscillator clocks. The wireless network, which consists of one master node connected to the GPS clock and five sensor nodes, is used in the experiments. Since the master node is connected to GPS, the master issues an external interrupt with broadcasting a *Sync* packet directly, once it receives the PPS signal. Upon receiving a *Sync* packet from the wireless channel, the WSN node issues the AMI interrupt for reading the COUNT register and adjusting the local clock. Similarly, when receiving the 30-th *Sync* packet from the master node, the sensor node restarts the clock module and resets all variables used for performance evaluation. After that, a series of data groups, consisting of the COUNT register value, the received *Sync* times, the number of clock COUNT resetting times and the COMP value, from each wireless node are sent to the PC. Moreover, we use synchronisation precision, which is the COUNT difference between the sensor node clock and the master clock, as an evaluation metric to analyse the achieved performance. The same $\alpha = 0.5$ and $\beta = 0.025$ configurations are adopted in the hardware experiments.

Fig. 9 indicates the evolution of precision over time by using three protocols. The time is measured in terms of the number of synchronisation cycles. Even though the PkCOs protocol adopts a Proportional-Integral (PI) controller for clock offset correction (i.e. COUNT), no correction action on the clock skew (i.e. the threshold COMP register) cannot guarantee that this algorithm achieves synchronisation on internal RC oscillator clocks. Moreover, as shown in Fig. 9, the integral controller's utilisation may worsen performance. Instead, PISync utilises the adaptive tuning solution for clock

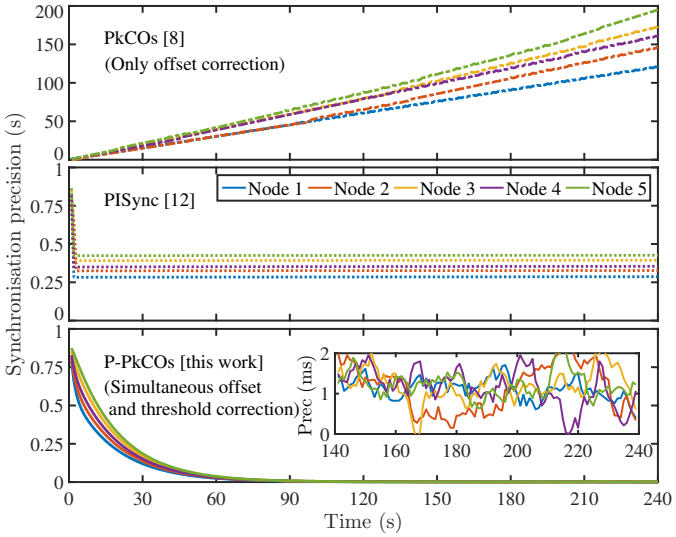


Fig. 9. Evolution of the precision through PkCO, PISync and P-PkCOs.

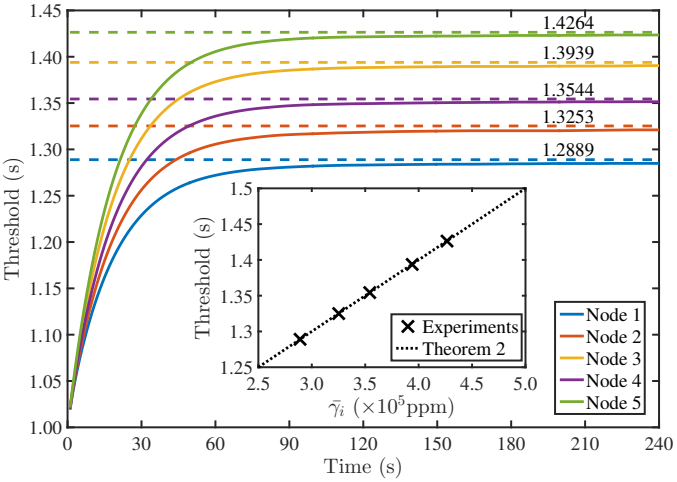


Fig. 10. Evolution of the clock threshold under the P-PkCOs protocol.

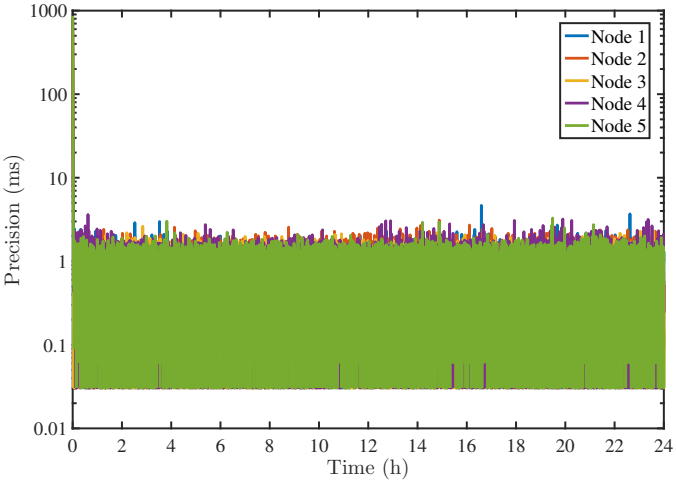


Fig. 11. Evolution of the precision through the P-PkCOs protocol within the 24-hour experiments.

TABLE IV
AVERAGE VALUES AND STANDARD DEVIATIONS OF PRECISION UNDER FIVE SYNCHRONISATION PROTOCOLS (UNIT: MILLISECOND (ms))

	240-second experiments										24-hour experiments	
	P-PkCOs		PISync		PkCOs		PCO		RBS		P-PkCOs	
	Mean	Std dev	Mean	Std dev	Mean	Std dev	Mean	Std dev	Mean	Std dev	Mean	Std dev
Node 1	1.122	0.263	285.987	0.552	n/a	n/a	n/a	n/a	n/a	n/a	0.593	0.365
Node 2	1.136	0.594	326.528	0.361	n/a	n/a	n/a	n/a	n/a	n/a	0.619	0.528
Node 3	1.216	0.408	393.839	0.301	n/a	n/a	n/a	n/a	n/a	n/a	0.597	0.431
Node 4	1.157	0.456	353.601	0.453	n/a	n/a	n/a	n/a	n/a	n/a	0.592	0.428
Node 5	1.229	0.305	426.061	0.220	n/a	n/a	n/a	n/a	n/a	n/a	0.604	0.426

skew correction; however, the order-of-magnitude of β is too small to overcome the effects of drifting frequency (of about 4×10^5ppm). Thus, PISync only achieves a precision of around 400ms. With the aid of Theorem 1, P-PkCOs (with the clock offset and skew correction solution) can realise robust time synchronisation with the precision of up to 2ms on poor-performing internal RC oscillator clocks. Table 4 summarises the average values and standard deviations of the realised synchronisation precision (between 140 seconds and the end of experiments) under five protocols. Owing to no clock frequency (i.e. threshold) adjustment, PkCOs, PCO and RBS fail to realise time synchronisation during the experiments. There are no precision average values and standard deviations in these three algorithms.

Fig. 10 demonstrates how the threshold varies against time under the P-PkCOs protocol. The experimental results show that the threshold converges to a particular value, which is only dependent on the nominal threshold and the mean value of the corresponding clock skew (see Theorem 2). For instance, the skew's mean value $\bar{\gamma}_1$ on Node 1 is 2.889×10^5ppm (see Table 3). Thus, the clock drift requires to be compensated by 0.2889 seconds to guarantee that the same time synchronisation cycle is achieved. The clock threshold is adjusted to 1.2889 seconds through the threshold correction solution to obtain cycle $T = 1$ second. In other words, if we configure the initial value in the COMP register to $\varphi_0(1 + \bar{\gamma}_i)$, the network may realise robust time synchronisation on the internal RC oscillator clocks by correcting clock offsets.

Finally, we also study the long-term synchronisation performance of the P-PkCOs protocol in the same single-cluster network. From Fig. 11, it can be seen that the precision of around 1.2ms is achieved during the 24-hour experiments. Therefore, on uncalibrated internal RC oscillator clocks (with 4×10^5ppm), the clock offset increases by 400ms per second. Our P-PkCOs protocol can keep the synchronisation accuracy be about 1.2ms, while PISync guarantees that the precision is around 400ms and PkCOs fails to realise synchronisation on such poor-performing clocks.

VII. CONCLUSION

In this paper, the behaviour of a low-accuracy internal RC oscillator clock is mathematically described as the non-identical and time-varying model. To achieve time synchronisation on poor-performing clocks, a packet-coupled synchronisation scheme is proposed to correct internal RC oscillator clocks through local timestamps. Also, a proportional controller is used to further improve TS performance. The stability region of controller parameters is given to guarantee that the clock threshold approaches a value, which is only determined by the nominal threshold and the average value of the corresponding clock skew. We propose a LMI condition to prove that the P-PkCOs performance is robust against the disturbances (from the timestamp uncertainties and the dominant drifting clock frequency noises). The experimental results show that the P-PkCOs protocol can guarantee robust synchronisation with the precision of 1.2ms on poor-performing uncalibrated internal RC oscillator clocks (with 4×10^5ppm). In contrast, PISync only achieves synchronisation with the accuracy of 400ms, and PkCOs, PCO and RBS cannot realise time synchronisation on such clocks.

REFERENCES

[1] Y. Zong, S. Liu, X. Liu, S. Gao, X. Dai, Z. Gao., "Robust Synchronised Data Acquisition for Biometric Authentication," *IEEE Trans. Ind. Informat.*, (early access).

[2] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund., "Industrial Internet of Things: Challenges, Opportunities, and Directions," *IEEE Trans. Ind. Informat.*, vol. 14, no. 11, pp. 4724-4734, Nov. 2018.

[3] J. Boyle, J. Reeve, and A. Weddell., "DiStiNCT: Synchronizing Nodes With Imprecise Timers in Distributed Wireless Sensor Networks," *IEEE Trans. Ind. Informat.*, vol. 13, no. 3, pp. 938-946, Jun. 2017.

[4] G. Cerone, A. Botter, and M. Gazzoni., "A Modular, Smart, and Wearable System for High Density sEMG Detection," *IEEE Trans. Biomed. Eng.*, vol. 66, no. 12, pp. 3371-3380, Dec. 2019.

[5] "Atmel SAM R21 Xplained Pro Evaluation Kit." [Online]. Available: https://www.microchip.com/en-us/development-tool/ATSAMR21-XPRO (Accessed: September 12, 2022).

[6] "Atmel SAM R21 ARM-Based Microcontroller Datasheet." [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/SAM-R21_Datasheet.pdf (Accessed: September 12, 2022).

[7] 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.

[8] Y. Zong, X. Dai, and Z. Gao., "Proportional-Integral Synchronisation for Non-identical Wireless Packet-Coupled Oscillators with Delays," *IEEE Trans. Ind. Electron.*, vol. 68, no. 11, pp. 11598-11608, Nov. 2021.

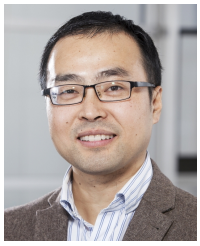
[9] Y. Zong, X. Dai, S. Gao, P. Canyelles-Pericas, and S. Liu., "PkCOs: Synchronisation of Packet-Coupled Oscillators in Blast Wave Monitoring Networks," *IEEE Internet Things J.*, (early access).

[10] Y. Zong, X. Dai, Z. Wei, M. Zou, W. Guo, and Z. Gao., "Robust Time Synchronisation for Industrial Internet of Things by H_∞ Output Feedback Control," *IEEE Internet Things J.*, (early access).

[11] X. Huan, K. Kim, S. Lee, E. Lim, and A. Marshall., "A Beaconless Asymmetric Energy-efficient Time Synchronization Scheme for Resource-constrained Multi-Hop Wireless Sensor Networks," *IEEE Trans. Commun.*, vol. 68, no. 3, pp. 1716-1730, Mar. 2020.
[12] K. S. Yildirim, R. Carli, and L. Schenato., "Adaptive Proportional–integral Clock Synchronization in Wireless Sensor Networks," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 2, pp. 610-623, Mar. 2018.
[13] F. Shi, S. Yang, X. Tuo, L. Ran, and Y. Huang., "A Novel Rapid-flooding Approach with Real-time Delay Compensation for Wireless Sensor Network Time Synchronization," *IEEE Trans. Cybern.*, early access.
[14] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi., "The Flooding Time Synchronization Protocol," in *Proc. 2nd Int. Conf. Embedded Netw. Sensor Syst. (SenSys)*, Nov. 2004.
[15] J. Elson, L. Girod, and D. Estrin., "Fine-grained Network Time Synchronization Using Reference Broadcasts," in *Proc. 5th Symp. Operating Syst. Design Implementation*, Dec. 2002.
[16] R. Gentz, A. Scaglione, L. Ferrari, and Y. Hong., "PulseSS: A Pulse-coupled Synchronization and Scheduling Protocol for Clustered Wireless Sensor Networks," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1222-1234, Dec. 2016.
[17] Z. Wang, and Y. Wang., "An Attack-Resilient Pulse-Based Synchronization Strategy for General Connected Topologies," *IEEE Trans. Autom. Control*, vol. 65, no. 9, pp. 3784-3799, Mar. 2020.
[18] H. Wang, R. Lu, Z. Peng, and M. Li., "Timestamp-Free Clock Parameters Tracking Using Extended Kalman Filtering in Wireless Sensor Networks," *IEEE Trans. Commun.*, vol. 69, no. 10, pp. 6926-6938, Oct. 2021.
[19] Y. Zong, X. Dai, Z. Gao, R. Binns, and K. Busawon., "Simulation and Evaluation of Pulse-coupled Oscillators in Wireless Sensor Networks," *Syst. Sci. Control Eng.*, vol. 6, no. 1, pp. 337-349, Jul. 2018.
[20] G. Giorgi, and C. Narduzzi., "Performance Analysis of Kalman-filter-based Clock Synchronisation in IEEE 1588 networks," *IEEE Trans. Instrum. Meas.*, vol. 60, no. 8, pp. 2902-2909, Aug. 2011.
[21] 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.
[22] J. A. Barnes, A. R. Chi, L. S. Cutler, D. J. Healey, D. B. Leeson, T. E. McGunigal, J. A. Mullen, W. L. Smith, R. L. Sydnor, R. F. C. Vessot, and G. M. R. Winkler., "Characterization of Frequency Stability," *IEEE Trans. Instrum. Meas.*, vol. IM-20, no. 2, pp. 105-120, May. 1971.
[23] Z. Yang, J. Pan, and L. Cai., "Adaptive Clock Skew Estimation with Interactive Multi-model Kalman Filters for Sensor Networks," in *Proc. 29th Int. Conf. Commun. (INFOCOM)*, May, 2010.
[24] C. Zucca, and P. Tavella., "The Clock Model and Its Relationship with the Allan and Related Variances," *IEEE Trans. Ultrason., Ferroelectr., Freq. Control*, vol. 52, no. 2, pp. 289-296, Feb. 2005.
[25] K. Ogata., Modern Control Engineering, 5th ed. Pearson, 2009.
[26] L. Yu., An LMI Approach to Robust Control (in Chinese), Tsinghua University Press, 2002.
[27] M. V. Barel, G. Heinig, and P. Kravanja, "A superfast method for solving toeplitz linear least squares problems," *Linear Algebra Appl.*, vol. 366, pp. 441–457, Jun. 2003.
[28] "ThunderboltA E GPS Disciplined Clock." [Online]. Available: https://www.trimble.com/Timing/thunderbolt-e.aspx (Accessed: September 12, 2022).

Yan Zong received the B.Eng. degree in electrical and electronic engineering from Nanjing Normal University, China, in 2016, and the Ph.D. degree in electrical engineering from Northumbria University, U.K., in 2020. His research interests cut across several disciplines, which include wireless networked control, machine learning, and their applications to the Internet of Things and aerial swarms.

He is currently an assistant professor at Nanjing University of Aeronautics and Astronautics, China. Prior to that, he was a postdoc at Cranfield University, U.K., from 2021 to 2022.



Xuewu Dai received the B.Eng. degree in electronic engineering and the M.Sc. degree in computer science from Southwest University, China, in 1999 and 2003, respectively, and the Ph.D. degree 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 to 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.

Pep Canyelles-Pericas received the double degrees in B.Eng. electrical and electronic engineering from the Technical University of Catalonia, Barcelona, Spain, and the University of Northumbria, Newcastle upon Tyne, U.K., in 2006. In 2008 he obtained a M.Sc. degree in engineering management from the University of Sunderland, U.K. He graduated for his Ph.D. degree in control systems from the University of Northumbria, in 2016. His research interests are acoustofluidics, optical sensing, instrumentation, synchronization, and control.

He is currently a Researcher with the MESA+ Institute of Nanotechnology, University of Twente, Enschede, the Netherlands. Between 2009 and 2012, he worked as Optoelectronics Research Engineer with the University of the Balearic Islands, Palma, Spain. After Ph.D. graduation, he engaged with Innovate U.K. projects in the Knowledge Transfer Partnership (KTP) and the Innovation to Commercialization of University Research (ICURe) schemes.

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, Northumbria University, 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.

Wai Pang Ng received his B.Eng. (Hons) in Communications and Electronic Engineering, from the University of Northumbria, U.K. and his Ph.D. in Electronic Engineering from the University of Wales, Swansea. His research interests include radio-over-fiber, cognitive radio, high speed optical communications, adaptive digital signal processing and distributed fibre sensing.

He is currently Professor of Electrical Engineering and Transnational Education Lead in the Department of Mathematics, Physics and Electrical Engineering, Northumbria University, U.K. and before that he was Associate Professor, Reader and Senior Lecturer. He has worked as Senior Networking Software Engineer at Intel Corporation.

Prof. Ng was the Chair of IEEE UK&RI Communications Chapter (2011-2015) and the Publicity Chair of IEEE ICC 2015 and ICC 2016. He has also served as Guest Editor for IET Communications. He was Symposium Co-Chair of Signal Processing for Communications Symposium at ICC 2009, Technical Chair of CSNDSP 2010, Technical Chair of NOC/OC&I 2011, and TPC members for numerous international conferences including GLOBECOM and ICC since 2006. He is also a member of SPCE TC, TCGCC, ONTC and FiWi Sub-TC.

Krishna Busawon received his first degree in Mathematics and Fundamental Sciences from the University of St-Etienne in 1989. He went to the University of Lyon where he obtained his B.Eng. and M.Sc. Degree in Electrical Engineering in 1990 and 1991 respectively. He then went on to continue his post-graduate studies in the same university and consequently, and obtained his M.Phil. and the Ph.D. degree in Control Systems Engineering in 1992 and 1996 respectively. His research interest lies mainly in the area of mathematical modelling, nonlinear control and observer design, fault detection and isolation with application to various engineering disciplines such as mechanical and power systems.

He is a Professor in Control Systems Engineering and currently is the head of the Power and Control research group in the Faculty of Engineering and Environment at Northumbria University. He was appointed as a Research Fellow at Simon Fraser University in 1997. Then, he joined the University of Nuevo León in Mexico where he worked as a Lecturer in the Department of Mechanical and Electrical Engineering (FIME). In the year 2000, he joined Northumbria University where he was appointed as a Senior Lecturer in the School of Computing, Engineering and Sciences. Later in 2006, he became a Reader in Control Systems Engineering and a Professor since 2013 at the same university.

Prof. Busawon is the Fellow of the IET and the Chair of the IEEE Control Systems Society for the UK and Ireland.

Richard Binns graduated from Huddersfield University with a degree in Electronic and Information Engineering in 1993 and also a Ph.D. in Analogue Test strategies in 1997. In 2001, He moved to Northumbria University on an EPSRC post-doctoral contract looking into Analogue Synthesis tool development in collaboration with Ericson Components and Cadence Design Systems. His current research works are varied from the design of electronics for visible light communications, energy management in electric vehicles, research into radiation detection mechanisms for personal dosimetry and power control systems development.

He is the Head of Department of Mathematics Physics and Electrical Engineering at Northumbria University. He is currently responsible for a department of 50 staff, a member of the faculty executive routinely headed up professional body accreditations and establishing collaborative links to institutions in Malaysia, Singapore and China.