

Establishing Wireless Time-Triggered Communication using a Firefly Clock Synchronization Approach

Robert Leidenfrost
Vienna University of Technology
Austria

robert.leidenfrost@gmail.com

Wilfried Elmenreich
Mobile Systems Group
Institute of Networked and Embedded Systems
University of Klagenfurt
Austria

wilfried.elmenreich@uni-klu.ac.at

Abstract

In South-East Asia, huge swarms of fireflies synchronously emit light flashes to attract mating partners. The underlying principle can be used to implement a robust and scalable distributed synchronization approach in wireless sensor networks. This paper describes the adaption of this principle for wireless sensor networks using battery-powered low-cost nodes based on an off the shelf IEEE 802.15.4 MAC layer in order to establish a global notion of time. This global notion of time is the basis for a wireless time-triggered network protocol, but also provides a service that can be used by real-time applications. The time-triggered behavior enables the implementation of an energy-efficient protocol, where sender and receiver units can be turned off during silent phases. The approach is evaluated by simulation and a real world case study using AVR®Z-LinkTM 802.15.4/ZigBee nodes. Results exploring various network topologies, parameter choices, and realistic clock source deviations show that this approach works in multi-hop topologies and there exists the potentiality to save more than two thirds of a node's energy consumption.

1. Introduction

The time-triggered paradigm [3] is well established for high dependable systems such as X-by-wire applications in the automotive and avionics domain. The basic element is a global timebase that is distributed among the nodes through clock synchronization. All communication activities are scheduled according to a predefined, periodic scheme. This simple but robust scheme enables the design of dependable distributed systems and simplifies system verification and diagnosis.

However, the time-triggered approach is usually thought of not being very resource-efficient regarding its average performance and, therefore, was considered of minor interest to sensor networks with a high demand for energy efficiency so far.

In this paper we will show a different application of a time-triggered architecture, where the global synchronicity is used to enable synchronized sleep schedules in a wireless network cluster which can save a considerable amount of energy at each node. This is especially useful in situations with low duty-cycles, e.g., a sensor network that is utilizing only a

fraction of its available bandwidth. In order to provide a common timebase we propose the application of a Reachback Firefly Algorithm (RFA), which is an algorithm inspired by the synchronous blinking behavior of natural fireflies in South-East Asia [2]. Thus we gain a robust self-organizing approach for synchronization without the need for dedicated time server nodes.

Due to the *a priori* known message schedule, the synchronized nodes are then able to predict the timing of incoming messages and can turn off their receivers when no transmissions of interest are scheduled. Since listening on the channel is a significant energy consumer of a typical wireless sensor node, the overall power consumption can thus be reduced in favor of battery lifetime. The global time can also support the application in tasks like timestamping, synchronous measurements, and timely coordinated distributed actions.

The objective of this paper is to describe a solution for a time-triggered wireless communication approach supporting scalability, graceful degradation, and efficient power management. The evaluation in this paper gives realistic figures for the precision of the clock synchronization and the achievable savings in power consumption.

The rest of the paper is structured as follows: Section 2 describes the basic features and operation of the RFA. Section 3 presents the design of our approach consisting of clock synchronization, a modified RFA and an energy saving scheme. Section 4 and 5 describe the evaluation of a case study implementation by simulation and on real hardware. Results are discussed in Section 6. Related work is treated in Section 7. The paper is concluded in Section 8.

2. Reachback Firefly Algorithm

The RFA was introduced in [11] and supports complete scalability, graceful degradation, and a simple calculation. It is based on the Pulse-coupled Biological Oscillators (PCO) model [8], but with the difference that it is more appropriate for the practical implementation in wireless networks.

The synchronization concept is based on a *state variable* $x = f(\phi)$, which characterizes an oscillator and corresponds to the charge of a firefly. The authors of the PCO model state that the *state function* $f : [0, 1] \rightarrow [0, 1]$ must be a smooth, monotonically increasing, and concave down function in order to achieve synchronicity. The *phase variable* ϕ therefore linearly increases with time. Note that we assume $x, \phi \in [0, 1]$ for normalization. Thus, ϕ is characterized by (i) $d\phi/dt = 1/T$, where T denotes the cycle period, (ii) $\phi = 0$ at the beginning of a cycle, and (iii) $\phi = 1$ when the oscillator reaches the threshold $x = 1$.

The coupling between the oscillators is defined by the *firing function* $\phi'(\phi)$:

$$\phi'(\phi) = \min(f^{-1}(f(\phi) + \varepsilon), 1)$$

This function depends on the *pulse strength* $\varepsilon > 0$ and is always calculated immediately after an oscillator receives a firing event (or flash in case of a firefly). We further use the term of *phase advance* or *phase jump* to define the increase in the phase-domain, denoted by $\Delta(\phi) = \phi'(\phi) - \phi$. Thus, due to the concave down state function, a constant addition in the state-domain results in a variable increase in the phase-domain and the phase advance in the beginning of a cycle is smaller than later in the cycle.

To combat the assumption problems of the PCO model resulting from the limits in wireless networks, the RFA additionally uses the notion of reachback response and pre-emptive message staggering.

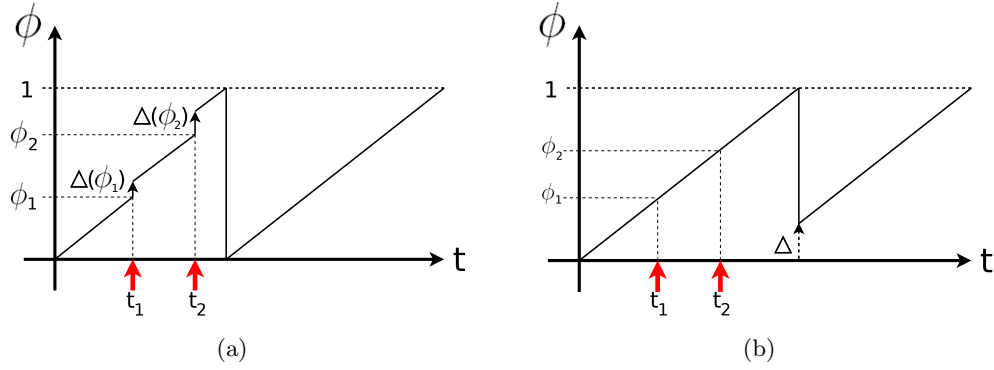


Figure 1. Comparison of the original PCO model (a) and the RFA (b). In the PCO-model, an oscillator immediately reacts to a firing event. In contrast, The RFA applies the overall phase jump at the beginning of the next cycle: $\Delta = \Delta(\Delta(\phi_1) + \phi_2)$.

In the original PCO model, an oscillator immediately reacts to each firing event. In contrast, the *reachback response* records the timestamps of all received firing events and calculates an overall phase jump once at the end of each period which is then applied at the beginning of the next cycle. Thus, if a node reaches the period end, it “reaches back in time” and reacts to the firing events of the past period. This principle is visualized in Figure 1.

A further problem in the PCO model occurs in the case of an already synchronized network comprising several nodes. If so, all nodes will trigger the transmission event for the synchronization message at the same time. As a result, the messages will collide and the collision avoidance mechanism of the CSMA/CA scheme takes effect. The resulting delay jitter can then be avoided by using MAC timestamping. However, the backoff scheme of the IEEE 802.15.4 standard ([4]) allows to backoff a message at most 5 times which results in a maximum backoff time of at most 36.48ms (at 2.4Ghz). Since the serialization delay of a full message is at most 4.256ms (133 bytes at 250 kbps), there can be at most 8 active wireless nodes without losing messages in the best case. Therefore, a bigger network comprising more than 8 nodes in the same broadcast domain requires an additional message staggering delay at an upper layer. A second reason for the additional message staggering is that the original IEEE 802.15.4 standard does not provide a MAC timestamping mechanism and thus does not allow to reduce the delay jitter due to the backoff scheme. The only way to reduce the delay jitter then was to modify the default values of some MAC specific attributes (i.e., $macMinBE$ and $macMaxCSMABackoffs$ were set to 0) in order to switch off the backoff mechanism. To avoid the resulting higher probability of transmission failures, the *pre-emptive message staggering* explicitly adds a timestamped random transmission delay to the firing messages at the application layer.

3. Time-Triggered Approach

The principal purpose of many protocols used in sensor networks is aimed at reducing power consumption through synchronized sleep schedules. Such an approach is also referred to as a low duty-cycle concept where the transceiver module of all nodes is periodically activated only for a short time with a period length from seconds up to hours. In contrast to such protocols, our concept is based on a time-triggered approach where a

node takes advantage of the *a priori* known transmission events. These events must be globally coordinated by the use of rounds and are stored in a file called Round Description List (RODL) file. In the current implementation such a round corresponds to a cycle of our synchronization algorithm. A round is further divided into a number of slots. Every node in a network must have its own RODL file and statically assigns a communication activity (e.g, idle slot, transmission, reception, task execution) to each slot in each round. This allows the setup of a collision-free communication and further improves the energy consumption by switching off the transceiver if it is not required. Figure 2 shows the time

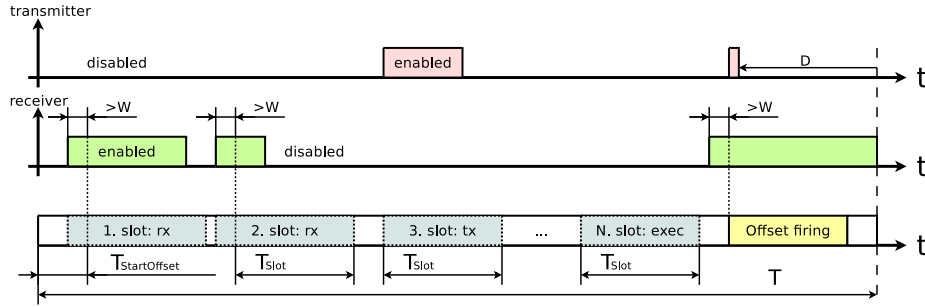


Figure 2. The principle of the time-triggered approach.

diagram of a time-triggered approach for a single node. Therein, a period is subdivided into several slots whereas each slot corresponds to either a *receiving slot*, a *sending slot*, an *execution slot*, or an *idle slot*. Concerning the energy awareness, the most important slots are the receiving slots since they determine how much energy is spent on listening and receiving. In the diagram, the first and the second slot are assigned to be receiving slots. The active time for the receiver unit differs between these slots. This comes from the automatic deactivation after the receiver has recognized the end of a transmission. The parameter w denotes the synchronization window described in Section 3.3.

The time-triggered approach also depends on the notion of a *global time*. Note that this is an abstract notion, because a distributed clock synchronization algorithm can only approximate the global time. Thus, the best achievable precision Π in an ensemble of clocks is lower bounded to the convergence function of the synchronization algorithm and the maximum drift rate ρ of all clocks in that ensemble. This is also known as the synchronization condition $\Pi \geq \Phi + \Gamma$. Therein $\Gamma = 2 \cdot \rho \cdot T$ denotes the drift offset and Φ the convergence function. In our approach, the convergence function is defined by the RFA, which periodically performs a state correction due to the reachback response. In order to get promising results, the global time must be approximated with a very high precision. Thus we have to minimize the drift offset. This can either be done by using high quality crystal oscillators or a more frequent resynchronization. Both approaches have their drawbacks, because in mass production, crystal oscillators would be expensive compared to the cheap internal RC-oscillators in low-cost nodes. Furthermore, a shorter period time results in the exchange of more synchronization messages in the same time and thus would affect the energy performance. However, the reduction of the maximum drift rate ρ can also be achieved by a *rate correction algorithm*. In our approach, this algorithm is performed in the digital domain and makes use of the concept of virtual clocks. A *virtual clock* abstracts the physical clock by the use of macroticks. A macrotick comprises several microticks which are generated by a physical clock. The principle of this concept is to change the number

of microticks a macrotick contains in order to adjust the granularity respectively frequency of the virtual clock. In the current implementation a macrotick corresponds to a complete cycle length. Thus, the duration of the periods can easily be changed by adjusting the threshold value of the physical timer/counter.

3.1. Clock State Correction

The clock state synchronization is established by the RFA model and uses the definition of the smooth, monotonically increasing, and concave down state function from [8] to calculate the overall phase jump Δ . The original state function in the PCO model is defined to be

$$f(\phi) = \frac{1}{b} \cdot \ln(1 + [e^b - 1] \cdot \phi) \quad (1)$$

where the parameter b is called *dissipation factor* and measures the extent to which $f(\phi)$ is concave down. Consider the dissipation factor is bigger than 1 and the pulse strength within $0 < \varepsilon < 1$, then the phase jump equals

$$\Delta(\phi) = \min(1, f^{-1}(f(\phi) + \varepsilon)) - \phi. \quad (2)$$

The direct implementation of all these functions would result in a time-consuming calculation process. Therefore, we simplified the equation by inserting the inverse function f^{-1} in Equation 2. Let $f^{-1}(x) = \frac{e^{bx}-1}{e^b-1}$, then the simplified phase jump equals

$$\Delta(\phi) = \min(1, \alpha \cdot \phi + \beta) - \phi \quad (3)$$

where $\alpha = e^{\varepsilon b}$ and $\beta = \frac{\alpha-1}{e^b-1}$. Assuming a strong dissipation factor $b \gg 1$, then β is negligible and the approximated phase jump can be reduced to

$$\Delta(\phi) = \min(1, \alpha \cdot \phi) - \phi. \quad (4)$$

As a result, we have a linear Phase Response Curve (PRC), where the *coupling factor* α specifies the strength of coupling between the oscillators and depends on the product of the dissipation factor b and the pulse strength ε . This result is similar to the simplified firing function described in [11].

Since the purpose of this work should demonstrate that such a synchronization approach works with an off the shelf communication stack without MAC-timestamping, we have to expect a delay jitter due to the MAC layer. Note that the *delay jitter* ε is defined to be the maximum absolute deviation of the delay a message encounters during the communication. However, an impossibility result of Lundelius and Lynch [7] state that this delay jitter lower bounds the best achievable synchronization precision to $\varepsilon \cdot (1 - \frac{1}{N})$, where N denotes the number of perfect clocks in the network.

Lower bound for the coupling factor α . To obtain good synchronization results with respect to different parameter choices, it is important to find bounds for these parameters. Further we want to calculate the optimal coupling factor with respect to the clock drift, so that the Firefly algorithm can synchronize the nodes. For this reason, we first denote a virtual clock of node k with VT^k . Next we want to get the duration of the synchronization interval of a specific clock k . For this, we introduce the term I^k , which determines the

interval of clock k with respect to a real-time reference clock z : $I^k = z(VC_{i+T}^k) - z(VC_i^k)$. Now we need to define the maximum deviation between the intervals of any clocks in an ensemble and is herein after referred to as the *virtual clock skew*, denoted by T_{skew} : $T_{skew} = \max_{j,k} (|I^j - I^k|)$.

W.l.o.g, we assume $I^j > I^k$. If the Firefly algorithm is based on Equation 4, then the coupling factor α must be greater than $\frac{I^j}{I^k}$. Otherwise, the system will never get synchronized. Regarding an ensemble of clocks with a maximum drift rate ρ , then the lower bound can be also defined by

$$\alpha > \frac{1 + \rho}{1 - \rho}.$$

The proof is trivial. Consider the clocks are already perfectly synchronized, but because of the virtual clock skew the node with the shorter interval I^k will reach the phase threshold ϕ_{th} earlier than the other one with the interval I^j . Consequently, node k fires and node j receives the fire event at ϕ_{fire}^j , where $\phi_{fire}^j = \phi_{th} \cdot \frac{I^k}{I^j}$. The phase jump performed by clock j at the phase ϕ_{fire}^j is denoted with $\Delta(\phi_{fire}^j)$. In order to keep the system synchronized, the following equation must hold: $\Delta(\phi_{fire}^j) + \phi_{fire}^j \geq \phi_{th}$. Since ϕ_{th} is normalized to 1, the result after substituting $\Delta(\phi_{fire}^j)$ with Equation 4 is $\alpha \geq \frac{I^j}{I^k}$. Hence, if the equation above is valid, the precision of an ensemble of clocks based on the Firefly algorithm is lower bounded to T_{skew} and the overall synchronization interval is determined by the clock with the shortest period.

Note that in the absence of a rate correction mechanism, a good choice for the coupling factor is $\alpha = \frac{1+3\rho}{1-\rho}$.

Upper bound for the coupling factor α . It is obvious that the coupling factor α of the phase jump function must have an upper bound. Otherwise, the clocks may not get synchronized due to a mutual excitation. Note that if a network is already perfectly synchronized and we neglect the communication jitter and any inaccuracies, then the system will keep synchronized independent of the choice of the coupling factor α .

The upper bound comes from the original RFA model, because there may exist initial configurations of an all-to-all topology such that the nodes achieve a fixpoint. For instance, given two nodes A and B with the corresponding phases ϕ_A and ϕ_B immediately after A has reached the period end, where $\phi_A < \phi_B$, then the following condition must hold: $\alpha < \phi_{th} / \min(\phi_B, \phi_{th} - \phi_A, \phi_{th} + \phi_A - \phi_B)$. Note that ϕ_{th} defines the specified threshold of the virtual clock and corresponds to the period length. This variable is normalized to 1. If the condition does not hold, then the phase jump function can be reduced to $\Delta(\phi) = 1 - \phi$. This results in a fixpoint where the nodes periodically change their phase state without achieving synchronicity. Therefore, it can be shown that the initial configuration $\phi_A = \frac{\phi_{th}}{3}$ and $\phi_B = \frac{2\phi_{th}}{3}$ requires that $\alpha < \frac{3}{2}$. Note that such fixpoints also exist in larger topologies, but are more complex and thus occur more infrequent. However, in reality, natural inaccuracies and jitter will let such a network eventually become synchronized with a higher time-to-sync.

Another upper bound results from our implementation, because we decided that if the phase jump is greater than half the threshold phase ϕ_{th} , then a node should not transmit a synchronization message in the same period. This is necessary, because otherwise the new start phase could be within the message staggering delay. Therefore, the coupling factor

α is bounded to $\alpha < \frac{2 \cdot N}{2 \cdot N - 1}$, where N denotes the maximum number of firing events a node may receive in a period.

Rate of synchronization. The authors from [8] and [11] have analyzed the synchronization rate for a network with two oscillators. In this case, they have proven that the time to synchrony is inversely proportional to the coupling factor α and further depends on the initial phase difference of the nodes at $t = 0$, denoted by $\delta = \phi_A^{(0)} - \phi_B^{(0)}$. Therefore, the number of iterations k until synchronicity equals

$$k \approx \frac{1}{\alpha} \cdot \ln \frac{1}{2 \cdot \delta - 1}.$$

The authors have also analyzed the case of n oscillators. However, considering a multi-hop topology requires a more sophisticated solution and is stated in [6].

3.2. Clock Rate Calibration

The concept of clock rate calibration combats the problem of frequency deviations due to the high clock drift of the RC-oscillators usually used in low-cost devices. This approach should allow a longer resynchronization interval with the same synchronization precision.

The core concept of our rate calibration algorithm is that a node measures and stores the durations of one or more synchronization intervals of all neighboring nodes by the use of timestamped messages. The node then smoothly adapts the own synchronization interval to the average of all the measured intervals by the use of a smoothing factor *sigma*. This factor ensures that the virtual clocks smoothly converge to the overall average interval. Furthermore, the smoothing factor is also used as a regularization parameter in order to avoid common-mode drift. Otherwise, if we would directly adjust the clock, the average interval over all clocks would become continuously longer or shorter. Note that a positive side effect of this approach is that the partial averaging of several intervals from the same node also reduces the influence of the delay jitter. The rate adaption is performed by changing the threshold value of the physical timer/counter which also represents the duration of a macrotick of the virtual clock. This is done by adding an adjustment value H to the threshold. If the adjustment value is positive, then the synchronization interval is getting longer. A negative adjustment value results in a shorter synchronization interval.

3.3. Energy Awareness

The energy consumption is an important quality characteristic of each communication protocol used in sensor networks. Often more than 50 percent of energy is used for idle listening [12]. Therefore, it is necessary to reduce the major energy sources. Some Media Access Control (MAC) protocols have already incorporated such a concept (e.g. S-MAC, T-MAC, etc.). However, we assume that the underlying MAC layer is only responsible for the medium access control and not for energy improvements. For this reason, we assign the tasks for energy reduction to the upper layers.

In order to reduce power consumption caused by idle listening, it is necessary to turn off the transceiver module as much as possible. In literature a protocol based on such a scheme is called to be a duty-cycle protocol. In such protocols a node becomes dormant most of the time and only wakes up for a short time if it is necessary. The *duty-cycle* is determined to be the ratio between the duration used for listening to the medium and the duration

of the complete period. Note that the bounded synchronization precision necessitates that the receiver module must be enabled some time prior, before any transmission is already started. To guarantee this behavior, the difference of the point in time the receiver is enabled and the first transmission may start should be greater than the synchronization window w , where w defines the upper bound of the synchronization precision. A good way is to choose $2 \cdot w$, because in the worst case, two nodes may deviate $\pm w$ with respect to a third node. However, the two nodes then deviate $2w$ to each other. Therefore, we say that a node is synchronized, if the maximum absolute deviation to all other nodes is smaller than the specified synchronization window. Last but not least, it is important that after a number of periods, each node enables the transceiver for a complete period. This idle listening avoids clique building in the case a synchronized cluster is not able to receive messages from another cluster which may has a complete different synchronization schedule.

4. Evaluation by Simulation

We evaluated our approach with a probabilistic wireless sensor network simulator called JProwler¹. JProwler has been developed by the Institute of Software Integrated Systems at the University of Vanderbilt and is basically configured to simulate the behavior of the Berkeley Mica Motes running TinyOS with the B-MAC protocol. It is a Java version of the Prowler[9] network simulator which is used for verifying and analyzing communication protocols of ad-hoc wireless sensor networks. Note that B-MAC is very similar to the IEEE 802.15.4 standard, because both implement the same CSMA/CA mechanism. Therefore, by modifying the MAC layer specific constants the simulation is used for the ZigBee nodes. In detail, the transmission time of a sync-message was modified to $0.375ms$, the minimum waiting time to $1ms$, and the random waiting time to $1.25ms$. Similarly to the implementation in the real hardware, the backoff scheme was deactivated in order to reduce the resulting delay jitter. The graphical user interface was enhanced by several new dialogs which enables the user to modify various parameters during the simulation. We further extended the simulator by an oscillator model. Thus, every virtual node is based on an oscillator, e.g., RC-oscillator or several crystal cuts. This allows the simulation of clock drift and its influence on the clock synchronization. Due to the fact that the frequency of an oscillator heavily depends on the supply voltage and the ambient temperature, the enhanced JProwler also contains the simulation of the ambient temperature. Other new features are the adjustment of the simulation speed and enabling/disabling nodes during the simulation.

4.1. Experiments and Results

The simulation results discussed in this chapter should give an overview of the achievable quality of our synchronization approach. For this reason, several network topologies have been developed and simulated. The results are compared due to different parameter choices, i.e., the coupling factor α and the number of nodes in the network.

In order to compare the simulation results with the outcomes from [11], the evaluation metrics are similar. Therefore, the two important parameters are the amount of time until

¹ISIS, Institute For Software Integrated Systems: <http://www.isis.vanderbilt.edu/Projects/nest/jprowler>

the system achieves synchronicity and the quality of precision. The *time to sync* defines the time until all nodes have entered the synchronization state and is determined by two parameters. These are the synchronization window w and the number of required periods a node has to keep within this window. In the following we call the amount of required periods *synchronization periods* and is usually set to 10. A node only enters the sync-state, if the maximum absolute deviation with respect to the other nodes is within the synchronization window for 10 out of the last 11 firing iterations. The definition of the *50th and 90th Percentile Group Spread* differs from that one defined in [11], because we only refer to one firing group. Therefore, the group spread in the simulation is defined to be the maximum absolute time difference between any two nodes in the network and thus cannot be greater than half the synchronization interval. We characterize the group spread distribution with the 50th and 90th percentile.

In order to avoid incorrect results due to settling effects during the startup phase, the start of the group spread measurement is postponed against the time to sync (t_s) and the time the experiment ends (t_e). On this account, the group spread measurement is performed during the interval $[t_s + \frac{t_e - t_s}{2}, t_e]$.

4.1.1. Parameter Settings

Several parameter settings are the same for all experiments and are adapted to simulate the behavior of our testbed environment. For instance, every virtual node is based on a virtual RC-oscillator. According to the datasheet, the real nodes have a nominal frequency of $8MHz \pm 10\%$. For this reason, every virtual node encounters a random initial clock drift between $-100ms$ and $+100ms$ per second. The general values of the other parameters are denoted in the table from Table 1.

Table 1. The general parameter choice used in all simulator experiments.

Parameter	Value
Oscillator technology	RC-oscillator
Initial clock drift [ppm]	± 100000
Interval time [ms]	1000
Granularity	10000 ticks/period
Coupling factor	1.01
Transmission delay [μs]	375
Delay jitter [μs]	1250
Synchronization window [ms]	10
Evaluation end [periods]	3600

4.1.2. Simulation Results

The next paragraphs discuss the simulation results in dependence of several network topologies and parameter choices. Every configuration was simulated over 3600 periods.

The all-to-all topology. The all-to-all communication topology is mainly used to measure the quality of the synchronization in dependence of the number of nodes and the coupling factor α . Therein, every node is in the transmission range of each other.

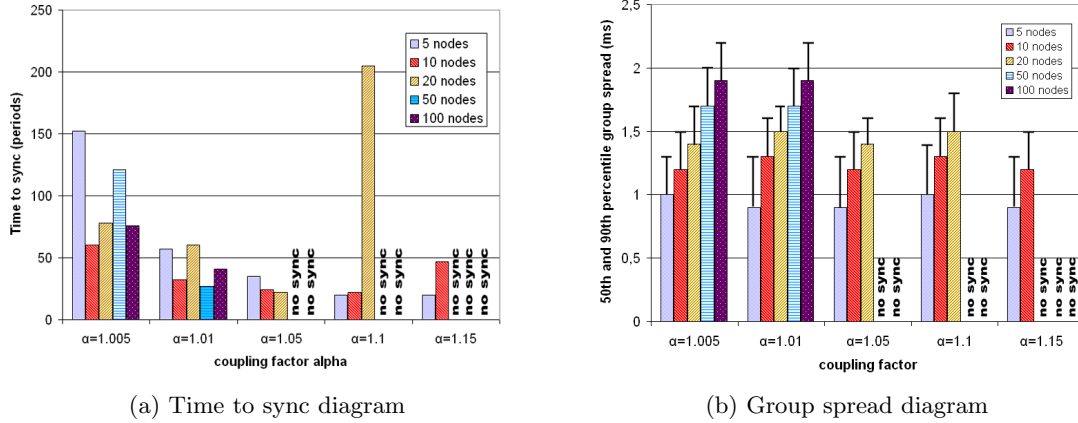


Figure 3. The time to sync and the group spread for an all-to-all topology experiment in dependence of the network diameter and different coupling factors. The solid bars in (b) represent the 50th percentile group spread, while the error bars correspond to the 90th percentile.

The simulation results based on this topology give a good overview on the impact of different coupling factors. According to the diagrams in Figure 3, the time to sync decreases with an increasing coupling factor α . If the factor is too big, then synchronicity will not be achieved. This effect is due to the upper bound of the coupling factor. It seems that the group spread is the same and thus independent of the coupling factor. The high time to sync bar in Figure 3a with $\alpha = 1.1$ and a number of 20 nodes comes from the fact that the coupling factor was too high.

The multi-hop topology. This communication topology is the most important one, because in reality many sensor network are based on a source-to-sink communication topology with a communication path consisting several hops. The simplest multi-hop scenario is a network comprising n nodes, which are ordered in a chain and can only communicate with the immediate neighbors. We further call the chain size *network diameter*. Such a network with a big network diameter is often very problematic to synchronize, because every hop involves a communication delay, which degrades the overall synchronization precision. Our solution is based on grouped multi-hop networks. Therein, the nodes are replaced with groups comprising several nodes in all-to-all topology which all have a bidirectional communication link to the immediate neighboring groups. Note that all grouped multi-hop topologies treated in our experiments have the same network diameter of 10 hops but vary in the group size.

The diagrams in Figure 4 shows the time to sync and the group spread in dependence of a different group size and coupling factor. The network diameter is always 10 hops. These diagrams lead to the result that the precision increases with a bigger group size. This effect is caused by the better information about the interval drift due to the increased number of neighboring nodes. If a node has more neighboring nodes, then the node receives more information about the clock drift and can more precisely calibrate the interval duration, which also improves the synchronization precision. However, it is also important to have a preferably small coupling factor. On the one hand this increases the time to sync, but on the other hand this also increases the possibility that the network achieves synchronicity.

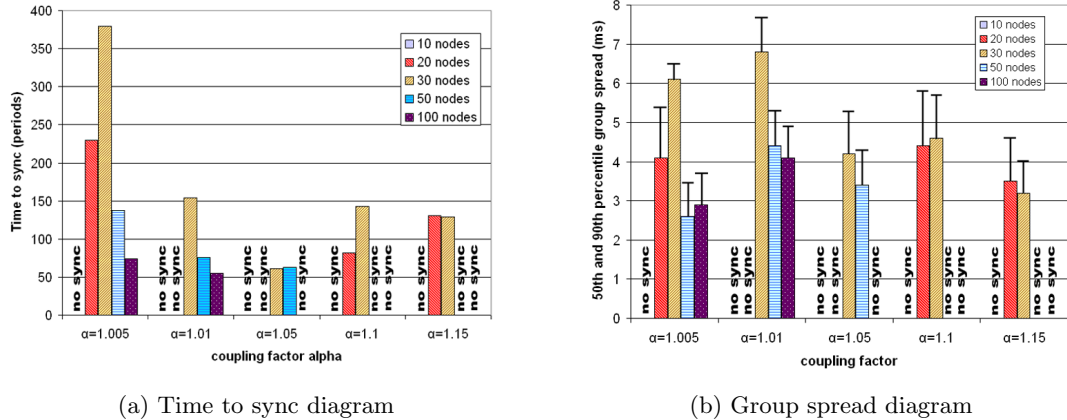


Figure 4. The time to sync and the group spread for a multi-hop topology with a network diameter of 10 in dependence of the cluster size and different coupling factors. Note that the number of nodes must be divided by 10 to get the group size. The solid bars in (b) represent the 50th percentile group spread, while the error bars correspond to the 90th percentile.

As a result, it is difficult to find the best parameter settings for a given multi-hop network, but it is definitely a good choice to have a group size of more than one node. This also increases the dependability and availability of the network.

5. Evaluation on Real Hardware

The simulation results provide a good basis for several parameter estimations in order to optimize the synchronization precision for different network topologies. However, the simulator does not support information about power consumption and further never fully reflects the real world scenario. For this reason, we implemented and evaluated our distributed algorithm in combination with the time-triggered approach on real hardware.

5.1. Testbed Description

The testbed is based on Atmel's demonstration kit ATAVRRZ200 [1]. The kit features two component boards: The Display Board and the Remote Controller Board (RCB)s. The Display Board is based on an Atmega128 controller and features an LCD-module. This board also works as a docking station for programming the RCBs. The RCBs therefore are based on an Atmega1281 controller and contain an AT86RF230 (2450 MHz band) radio transceiver. The implementation of our approach is done with the AVR®Z-Link™802.15.4/ZigBee nodes. The information about the synchronization precision is gathered via the established TDMA scheme. Therefore, beside energy savings, the time-triggered approach also serves as an evaluation protocol. For this, we used a modified version of the TTP/A protocol [5].

The synchronization algorithm was implemented analogously to the implementation in JProver. A simple RC-oscillator based 16-bit timer was used to generate the synchronization interval with a duration of one second. The only differences with respect to the parameter choices in Table 1 are a higher granularity of the virtual clock (31250 ticks/period)

and a higher transmission delay of $896\mu s$.

We modified the initial settings of the MAC sublayer, i.e. the minimum backoff exponent, to reduce the transmission delay. We further assumed that it is better to omit a message than to transmit postponed synchronization data, where the time information is out of date. In our implementation, every node is configured as a Full-Function Device (FFD) and no association process is required. This necessitates the use of individual predefined 16-bit short addresses for each node.

5.2. Experiments and Results

The evaluation metrics for the testbed experiments are similar to the one used for the simulation experiments. In order to observe the relative deviations over all nodes in a network, we decided that every node transmits its own maximum absolute deviation of the last period to a central evaluation node, which then calculates the maximum over all received deviations. These values over several minutes are then taken to compute the 50th and the 90th percentile group spread.

To be able to compare the testbed results with the simulator results, the parameter configuration has to be the same as used in the simulator experiments. Since in reality it is not possible to speedup the time, we reduced the experiment time to 720 periods.

5.2.1. Testbed Results

The all-to-all topology. For the all-to-all topology experiment, we measured the group spread in dependence of several coupling factors. The results of experiments in a network of 5 ZigBee nodes are visualized as histograms. All histograms in dependence of different coupling factors look similar and have a right-skewed distribution.

Table 2. Comparison of several parameters in dependence of different coupling factors. The values between the brackets correspond to the simulation results with the same all-to-all network configuration comprising 5 nodes.

Parameter choice	Time to sync (periods)	50th-percentile (μs)	90th-percentile (μs)	Maximum deviation (μs)	Standard deviation (μs)
$\alpha = 1.005$	105 (152)	672 (1000)	2005 (1300)	3456 (2200)	538 (257)
$\alpha = 1.010$	79 (57)	704 (900)	1632 (1300)	2592 (2000)	410 (250)
$\alpha = 1.050$	24 (35)	704 (900)	1973 (1300)	3040 (1900)	501 (262)
$\alpha = 1.100$	33 (20)	672 (1000)	1723 (1400)	3104 (2000)	451 (267)
$\alpha = 1.150$	14 (20)	732 (900)	1965 (1300)	3776 (1800)	565 (250)

The table in Table 2 contains the simulation results and the testbed results with the same network configuration comprising 5 nodes in all-to-all topology. This demonstrates that the results are similar. For instance, the time to sync and the 50th percentile group spread of the testbed system are mostly better than the simulation results. Only the 90th percentile group spread and the maximum group spread are worse compared to the simulation results. The results from an all-to-all topology comprising 9 nodes are comparable with those denoted in Table 2. Note that only a maximum number of 9 nodes were available for our experiments.

We investigated the behavior of precision degradation and found out that the testbed system suffered from a higher delay jitter as was expected. Furthermore, the higher transmission delay also degrades the precision. Note that we have already compensated the constant delays in the implementation. However, there may exist other delays which we have not considered. Simulation experiments have shown that a higher transmission delay or a higher delay jitter are the major reasons for the precision degradation due to the state correction algorithm and hardly affect the rate calibration scheme.

The multi-hop topology. The results from the multi-hop experiments are important in order to get an overview about the limits of our synchronization approach. The first scenario was made up of 5 nodes ordered in a chain, where a node can only communicate with the immediate neighbors. The only difference between the simulation and the testbed environment is that the testbed environment does not have an omniscient observer, which is able to continuously measure the synchronization deviation among all nodes. For this reason, we decided to measure the time difference between the edge nodes with the aid of an oscilloscope, whereas each node periodically sets an output pin at the same phase state for a short time. Unfortunately, these measurements can not be gathered automatically over several periods. Therefore, we manually made snapshots over several minutes and took those diagrams, which display the biggest time deviation. The results show that the precision of a realistic multi-hop network with 4 hops is about $3ms$. It is interesting that the simulation of the same network with a configured delay jitter of $1250\mu s$ lead to the same result.

To get an overview of the precision degradation with respect to the network diameter, another multi-hop experiment with 9 nodes was performed. The measurement setup is similar to the previous multi-hop network. The measurement results show a maximum deviation between the edge nodes of up to $14ms$. It is obvious that such a precision is unacceptable. In summary, our synchronization algorithm dramatically degrades with each hop and is therefore not applicable for the use in a highly multi-hop sensor network. These bad results highly likely come from the delay jitter. The simulator presented similar results, if we adjust a delay jitter of $2500\mu s$. The higher delay jitter in the testbed system may result from the fact that each node in the testbed environment is in the communication range of each other. To simulate the multi-hop network, we have simply implemented a message filter. Thus the probability of message collisions in the testbed environment is higher than in the simulation environment and therefore may cause the unexpected higher delay jitter.

We further measured the behavior of a grouped multi-hop network comprising 3 clusters with a cluster size of 2 and additional two edge nodes which have a communication link to the corresponding two edge groups. This was the only acceptable configuration with a number of 9 nodes. The maximum deviation between the edge nodes which was measured over about 10 minutes is $7ms$. In the simulator we had to configure a delay jitter of $7ms$ or in the case of no delay jitter an uncompensated additional transmission delay of $1.5ms$ to get the same result. Therefore, it is highly likely that beside the delay jitter, the testbed environment additionally suffers from a longer communication delay, which was not regarded in the current implementation. Unfortunately, due to the limited number of nodes, we were not able to make further experiments with a bigger group size. Thus we must rely on the simulation experiments which show us that a bigger group size usually results in a better synchronization precision.

5.2.2. Energy Measurements

The energy consumption plays an important role for the device lifetime in battery-powered wireless networks, especially if no infrastructure is available. All RCBs are battery-powered with two 1.5V AAA-batteries and thus have a voltage supply of 3 Volt. In order to get the device lifetime, the average power consumption P_{avg} is compared with the electrical energy W_{bat} of the batteries, which we assume to be about $3V \cdot 1200mAh = 3600mWh$. The lifetime in hours is the ratio W_{bat}/P_{avg} and can be reduced to the equivalent formula $t_{life} = E_{bat}/I_{avg}$, where E_{bat} corresponds to the battery charge, denoted in mAh . If so, then the formula determines the device lifetime in hours and I_{avg} defines the average current consumption.

For further energy calculations, the current consumption during a complete period can be classified into four parts. These are the firing time, idle time, execution time, and transmission time.

The *firing time* results from the message staggering delay and corresponds to the interval where the transceiver is enabled and the nodes are allowed to transmit their synchronization messages. In our test application, this interval is also called part 1 of the firing time and equals the duration of $50ms$. The consumed current during this time is about $20mA$. Note that there exists an interval between the end of the firing time and the period end which acts as a safety margin in the case a node starts a transmission exactly at the end of the firing time. If so, the transceiver must be enabled as long as the transmission continuous. This safety margin is further named part 2 of the firing time and consumes a current of $24mA$.

The *idle time* is the part, where the current drops to a minimum. The reason for the small current lies in the fact that the device is dormant, i.e., the transceiver is disabled. With our ZigBee nodes, we measured a current of about $6.2mA$.

The *execution time* is the time where the RCB device executes some code. This is always the case at the end of each period, where the device has to execute the RFA. Other execution tasks must be configured in the RODL file. In the test application used for the energy measurement, the RODL file only contains one execution slot in each period. This task is responsible for data preparation. We measured a current of about $11mA$ for a duration of $1ms$. This energy part mainly depends on the amount of code of the executed

Table 3. Listing of the major energy consumers and their corresponding battery discharge. The energy calculation assumes a working voltage of 3 Volt.

Energy consumer	I [mA]	t [s]	Battery discharge per cycle	Energy consumption per cycle
Firing Time, Part1	20.0	0.060	1.200 mAs	3.600 mJ
Firing Time, Part2	24.0	0.013	0.312 mAs	0.936 mJ
Execution Time	11.0	0.001	0.011 mAs	0.033 mJ
Transmission Time	25.0	0.005	0.125 mAs	0.375 mJ
Idle Time	6.2	t_{idle}	$6.2mA \cdot t_{idle}$	$18.6mW \cdot t_{idle}$
Σ		T	$1.648 mAs$ $+ 6.2mA \cdot t_{idle}$	$4.944 mJ$ $+ 18.6mW \cdot t_{idle}$

tasks.

The *transmission time* corresponds to the time, where the device is transmitting data. Normally, this is always the case when the RCB wants to broadcast its synchronization message during the firing time. Other transmissions during the period must be registered in the RODL file. We further measure the energy consumption of a registered transmission slot, which is used to broadcast test data. We measured a current consumption of about $25mA$ over a time of $4.8ms$. Note that this duration does not equal the real transmission time of about $1ms$. This comes from the fact, that the transceiver requires some time for the startup phase and that the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) scheme at the MAC layer may also cause some delay although the backoff scheme was disabled.

Table 3 sums up all different energy consumers with the corresponding battery discharge in mA s. In the case the period duration T is exactly one second, the values defined in this table results in a battery discharge per cycle of about $E_{device} = 7.358mA$ s. Thus, the average current consumption I_{avg} also equals $7.358mA$. Assuming that our batteries deliver a charge of about $E_{bat} = 1200mAh$, then the resulting lifetime (t_{life}) in hours can be calculated as follows:

$$t_{life} = \frac{E_{bat}}{I_{avg}} = \frac{1200mAh}{7.358mA} = 163h \simeq 1week$$

The configured duty-cycle for this result is about 7 percent, but could be reduced by increasing the period time. If we assume that the time slices of the other consumers are for the most part constant, then a larger period time induces also a larger idle time. Note that a larger period time usually also entails a degradation in precision. The duty-cycle is defined to be the ratio between the sum of the two firing times ($t_{f,1}, t_{f,2}$), the execution time (t_e), and the transmission time (t_t) and the complete period time (T). Thus, the duty-cycle is hereinafter denoted by DC and corresponds to the equation $DC = \frac{t_{f,1} + t_{f,2} + t_e + t_t}{T}$.

To follow up on our special energy example, we further want to calculate the improvement of the lifetime with respect to the lifetime as if no synchronization approach would be established, i.e., the duty-cycle equals 100%. In that case, the average current consumption equals $23.752mA$. Consequently, a duty-cycle of 100% corresponds to a lifetime of about $50\frac{1}{2}$ hours. A comparison among the lifetime with a duty-cycle of 100% and the achieved lifetime with our configured duty-cycle of about 7% shows that the synchronization approach improves the lifetime by a at least a factor of three.

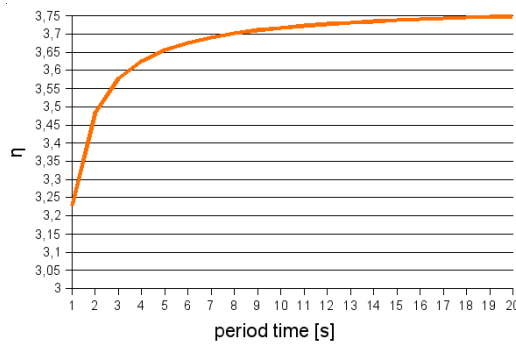


Figure 5. The lifetime improvement η as a function of the period time.

To illustrate the dependence between the lifetime improvement and the period time, we introduce the improvement factor, denoted by η . This factor is the ratio between the improved lifetime and the reference lifetime corresponding to a duty-cycle of 100% at the same period time. The improvement factor equals $\eta = \frac{I_{avg}(100\%)}{I_{avg}(DC(T))}$ and is visualized in Figure 5.

6. Discussion

The simulator and testbed results have shown that the simulator provides promising results which are mostly similar to the testbed results. Several experiments have shown that the rate calibration works well in the presence of high delay jitter and transmission delay. However, the results from the multi-hop topology experiments in the testbed system are worse compared to the simulation results. This comes from the fact that our testbed environment suffers from an unexpected delay jitter and further an additional communication delay which was not regarded in the state correction algorithm.

These conditions and the presence of asynchronous communication patterns in realistic sensor networks regarding highly multi-hop scenarios with high requirements on availability and dependability makes the employment with low-cost nodes in such an environment usually unacceptable. Otherwise, the proposed grouped multi-hop topology with a small network diameter allows the forwarding more reliable than it would be in a standard ad-hoc network. For instance, the nodes in such a single group could be statically configured via the RODL file such that they all forward the same message to the neighboring groups in different slots, but within the same period. In other words, such a group comprises several nodes which have the same functionality (e.g., sensor measurement, relaying, sensor fusion, computation, actuator control in combination with Triple Modular Redundancy (TMR)). If so, then a single node failure has no impact on the network behavior, since there exist no dedicated nodes. Thus, graceful degradation is the main advantage of this protocol.

The time-triggered approach *a priori* requires a static communication plan (the RODL file) for each node in the network. Therefore, the network and possible multi-hop scenarios must be analyzed before the RODL file can be created. This requires additional work and expenses compared to other protocols. However, the inherent advantage is that the deployed communication schedule can be perfectly adapted to the current known static network topology. Therefore, multihop routing algorithms are no longer necessary, since the routing is done implicitly during the configuration process of the RODL files. Though, a change in the network topology may also involve a reconfiguration of all RODL files.

We have shown that our approach provides a duty-cycle of about 7% and thus reduces the energy consumption in a simple network by at least a factor of three. Note that the duty-cycle heavily depends on the synchronization precision. Since we consider the implementation on low-cost nodes by the use of an off the shelf MAC layer which does not support MAC timestamping, our synchronization precision is limited to the MAC specific delay jitter. Other established protocols such as S-MAC or the slotted variant of IEEE 802.15.4 use an alternative or more sophisticated clock synchronization approach and thus usually achieve a duty-cycle of lower than 1%. However, such protocols usually demand a dedicated master node (and maybe additional backup nodes) which represents a single point of failure. In contrast, our approach has the inherent advantage that it does not require any dedicated nodes. Furthermore, a higher synchronization precision and thus a

lower duty-cycle can indeed be achieved by more accurate but also more expensive external crystal oscillators. As a result, the tradeoff lies between the costs of a node and the power consumption.

7. Related Work

There exists many work which treats the biologically inspired Firefly synchronization model for realizing the communication in sensor networks. However, we have not found any reports covering such a synchronization approach for establishing a TDMA communication scheme. This may be due to the bounded precision caused by the clock drift of cheap oscillators used in low-cost nodes and that the time to sync is relatively big in contrast to other synchronization schemes.

The most important related work on Firefly synchronicity refers to [11]. Therein, the authors present the Reachback Firefly Algorithm, which is well-suited for the implementation in sensor networks. The algorithm was simulated with TOSSIM in contrast to several parameter choices (e.g., different node topologies, Firing Function Constant (FFC)² values, and network diameter).

In [10], the authors introduce a time advance strategy based on the PCO model, which takes the delays in wireless systems into account. Similarly to [11], they incorporate the fact that a node cannot transmit and receive at the same time. The time advance strategy presented in this paper compensates the delay, which is responsible for the lower bound of the accuracy. This delay depends on the dominant transmission and decoding delay. The compensation is done by delaying the transmission of the synchronization messages.

8. Conclusion and Outlook

An alternative synchronization algorithm based on the synchronous flashing of fireflies was introduced in order to establish a global timebase that supports the implementation of a time-triggered approach based on the off the shelf MAC layer IEEE 802.15.4. This allows a collision-free communication and a reduction of power consumption by at least a factor of 3. The synchronization is based on a self-organized principle with a simple calculation and provides complete scalability and graceful degradation. This is beneficial for the use in sensor networks. This has the inherent advantage, that no dedicated synchronization node is required and thus there exist no single point of failure. Furthermore, the additional rate calibration scheme allows a longer resynchronization interval and the use of cheap oscillators with high drift rates, which are usually featured in low-cost nodes.

The approach has been evaluated by simulation and an implementation in a real testbed environment. Several experiments based on an all-to-all topology have shown that it is possible to achieve a synchronization precision which is lower than 1ms. Unfortunately, the testbed system suffered from an unexpected delay jitter and an additional communication delay. For this reason, the testbed results considering highly multi-hop topologies were worse compared to the simulation results with a low delay jitter.

Future work will rely on the improvement of the synchronization precision by the use of an alternative MAC-Stack supporting MAC timestamping. Furthermore, we want to

²The FFC is defined to be the inverse of the coupling strength ϵ .

compare the results of our approach with the use of different testbed environment and also a different synchronization approach.

Acknowledgment

This work was supported by the Austrian FWF project TTCAR under contract No. P18060-N04. We would like to thank the anonymous reviewers for their constructive comments on an earlier version of this paper.

References

- [1] Atmel Corporation. *ATAVRRZ200 Demonstration Kit AT86RF230 (2450 MHz band) Radio Transceiver User Guide*, July 2006. Document No. 5183AZIGB12/07/06.
- [2] J. Buck. Synchronous rhythmic flashing of fireflies. *The Quarterly Review of Biology*, 63(3):265–289, September 1988.
- [3] W. Elmenreich, G. Bauer, and H. Kopetz. The time-triggered paradigm. In *Proceedings of the Workshop on Time-Triggered and Real-Time Communication*, Manno, Switzerland, December 2003.
- [4] IEEE Computer Society. *IEEE Standard for Information technology – Telecommunication and information exchange between systems – Local and metropolitan area networks – Specific requirements. Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*. Institute of Electrical and Electronics Engineers, September 2003.
- [5] H. Kopetz et al. Specification of the TTP/A protocol. Research Report 61/2002, Technische Universität Wien, Institut für Technische Informatik, Vienna, Austria, September 2002. Version 2.00.
- [6] D. Lucarelli and I.-J. Wang. Decentralized synchronization protocols with nearest neighbor communication. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys'04)*, pages 62–68, November 2004.
- [7] J. Lundelius and N. Lynch. An upper and lower bound for clock synchronization. *Information and Control*, 62(1):190–204, 1984.
- [8] R. E. Mirollo and S. H. Strogatz. Synchronization of pulse-coupled biological oscillators. *SIAM Journal on Applied Mathematics*, 50(6):1645–1662, December 1990.
- [9] G. Simon, P. Volgyesi, M. Maroti, and A. Ledeczi. Simulation-based optimization of communication protocols for large-scale wireless sensor networks. In *Proceedings of the IEEE Aerospace Conference*, volume 3, pages 1339 – 1346, March 2003.
- [10] A. Tyrrell, G. Auer, and C. Bettstetter. Firefly synchronization in ad hoc networks. In *Proceedings of the MiNEMA Workshop*, Leuven, Belgium, February 2006.
- [11] G. Werner-Allen, G. Tewari, A. Patel, M. Welsh, and R. Nagpal. Firefly-inspired sensor network synchronicity with realistic radio effects. In *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems (SenSys'05)*, pages 142–153, November 2005.
- [12] W. Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. In *IEEE/ACM Transactions on Networking*, 12(3):493, June 2004.