Decentralized Brains: A Reference Implementation with Performance Evaluation

Aswin Karthik Ramachandran Venkatapathy[1,2(✉)], Anas Gouda[1], Michael ten Hompel[1,2], and Joseph Paradiso[3]

[1] TU Dortmund University, 44227 Dortmund, Germany
{aswinkarthik.ramachandran,anas.gouda}@tu-dortmund.de
[2] Fraunhofer Institute for Material-Handling and Logistics, 44227 Dortmund, Germany
michael.ten.hompel@iml.fraunhofer.de
[3] Medialab, Massachusetts Institute of Technology, Cambridge 02139, USA
joep@media.mit.edu
https://flw.mb.tu-dortmund.de

Abstract. *Decentralized Brains* is a concept developed for multiple parallel control of decentralized collaborative swarms and systems. This systems communication paradigm comprises of local peer-to-peer control as well as the global state management which is required for large-scale collaborative systems. The scenarios vary from self-assembly protocols for aerospace structures to organizing a warehouse in a material-handling context where heterogeneous systems collaboratively accomplish a task. A reference implementation of the conceptualized protocol is developed and deployed in a 345 node test bed. A reliable broadcast communication primitive using synchronous broadcast is deployed in a dual-band System on Chip (SoC) micro-controller. The performance of the adopted synchronous broadcast for network-wide flooding and consensus is presented in this article. The firmware is based on the latest branch of Contiki-Open Source Operating System - Next Generation (Contiki-NG) to keep further open source implementations easier and modularized as per the ISO OSI networking model. Using the concepts of multi-hop mesh networking, network flooding using synchronous broadcasts from wireless sensor networks and multi-band radio controllers for cognitive radios, a hardware-software architecture is developed, deployed and evaluated. The synchronous broadcast has a success rate of more than 95% in network wide floods and the implicit network wide time synchronisation of less than 1 µs which is evaluated using experiments using a 345 node test bed is presented in this paper. The developed communication primitives for the target hardware CC1350 STK and the developed experiments are available at https://github.com/akrv/deBr.

Keywords: Wireless sensor networks · Decentralized communication · Network flooding · Constructive interference · Synchronous broadcast · Concurrent transmissions · Distributed consensus · Time synchronization

L. Peñalver and L. Parra (Eds.): Industrial IoT 2020, LNICST 365, pp. 80–99, 2021.
https://doi.org/10.1007/978-3-030-71061-3_6

1 Introduction

The terminology of *Decentralized Brains* is inspired by evolutionary biology. Cephalopods are a family of species from the molluscan class Cephalopoda such as a squid, octopus, or nautilus. From the perspective of evolution, the class of Cephalopoda has multiple brains and they work towards a common goal in a decentralized manner. The reason for such organisms to have separate brains or a *decentralized brains* is to delegate motor processes without detracting from other important functions. Neurons in an octopus are found in the arms, which can independently taste and touch and also control basic motions without the supervision of a central brain [1,2]. In other words, the decentralized brains of an octopus would order to grapple its prey after sensing it and the nerve cells in its tentacles would make the individually detailed control decisions regarding that action. This decentralized intelligence allows for autonomous task completion—an intent is broadcasted across the *brains* and respective nervous systems take action locally. Socially networked industry was a terminology coined to explicitly refer to the collaborative nature of industrial entities. Here a social network between the sensors, robots, drones and various workstations is established to exchange, negotiate and act on the information that is available in the network. Various scenarios of socially networked industries are explored in previous literature [3–6]. Collective behavior from a biological perspective often involves large numbers of autonomous organs or organisms interacting to produce complex assemblies [3]. *Kilobots*, an effort to mimic biology inspired self-assembly demonstrated the ability of self-assembly in a large-scale autonomous miniaturized robotic system [3]. It was achieved by creating and programming swarm behaviour to achieve a global behavior with the nodes i.e., the robots interacting between each other [3]. *Decentralized Brains* focuses on the communication

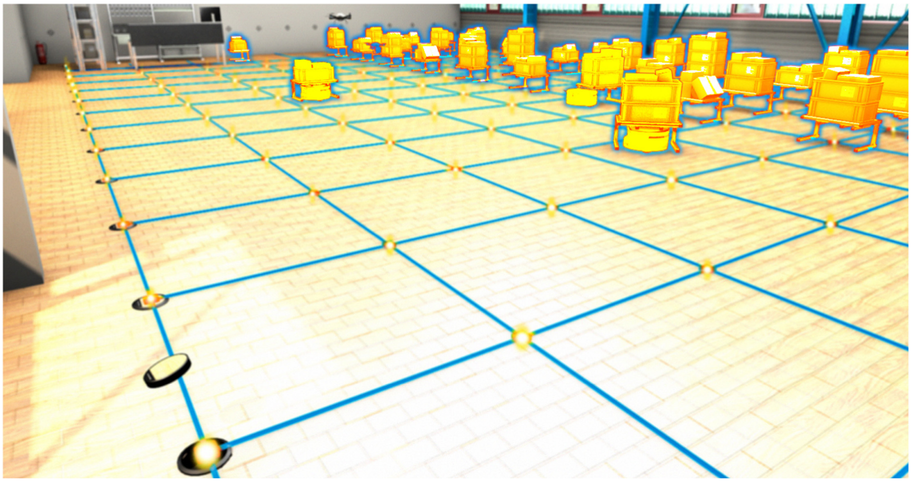

Fig. 1. Sensor floor 3d render showing the topology of the nodes.

between the nodes in scenarios of self-assembly and heterogeneous collaboration of nodes taking care of the communication payload dissemination. This allows the implementation of hardware-based extensions to existing centralized industrial scenarios to retrofit decentralized collaboration on a network level [2]. Using such a communication paradigm it becomes easier to implement and deploy multiple parallel control of decentralized systems. An abstract example from the perspective of networking and communication is when a node wants to discover and join a network and wants to start operating within the network to perform collaborative tasks. The collaborative task can be for self-assembly where the control systems have to exchange precise information between interacting nodes and disseminate the change in global states across all nodes in the network. We present this paper as a follow up to the *Decentralized Brains* concept [2], in advance of future large-scale integration of the communications architecture in diverse production systems. **Contributions** to this paper are (*i*) development of a networking architecture using well-proven, low-power, low data-rate industrial Cyber Physical Systems (CPS) and wireless communication standards applicable for self-assembly protocols, (*ii*) evaluation of a novel method for broadcasting and network flooding in a dual-band SoC and (*iii*) an implementation and performance evaluation on a 345 node test bed called the sensor floor deployed using commercially available off-the-shelf hardware (Fig. 1).

The background section gives insights on the concept of *Decentralized Brains* and low data-rate standards with some details on the software stacks used. Section 3 discusses the concept of synchronous broadcast which is the main concept *Decentralized Brains* is based on, followed by the hardware and software implementation where the core contribution of this paper exists. Section 4 presents our experimental test-bed and two experiments, first one shows that synchronous broadcast is possible for Sub-1-GHz band the second presents running our protocol for 50000 packets.

2 Background

In this section the elements that are required to form a low data-rate *Decentralized Brains* network are described with their respective architectures. The underlying architectures are well-known in the field of Wireless Sensor Networkss (WSNs). These architectures are implemented in a special dual SoC hardware to demonstrate the effects of *Decentralized Brains*. Leveraging the concept of time synchronization, a reliable synchronous broadcast approach is implemented by creating constructive interference that becomes the fundamental element for *Decentralized Brains*.

Few worth notable related-works that used the phenomenon of synchronous transmissions are Glossy [7], Low-power Wireless Bus (LWB) [8], Chaos [9], and Baloo [10]. The basic synchronous broadcast primitive was developed by Glossy [7] which is used in many other communication protocols such as LWB and Chaos. *Decentralized Brains* also proposes to use an adaptation of Glossy to the target hardware with dual-band SoC from Texas Instruments CC1350.

The method used for developing a reliable broadcast using synchronous transmissions is called the capture effect. Capture effect happens due to the intrinsic redundancy available in digital modulation schemes. Glossy [7] explores the nature of synchronous transmissions using constructive interference in low-power WSNs. The phenomenon of constructive interference based synchronous broadcast has been exploited in other works to develop routing-less networking for low-power WSNs. LWB turns a multi-hop low-power wireless network into an infrastructure similar to a shared bus, where all nodes are potential receivers of all data [8]. It achieves this by mapping all traffic demands on fast network floods, and by globally scheduling every flood using Glossy [7,8]. As a result, LWB inherently supports one-to-many, many-to-one, and many-to-many traffic [8]. Baloo is a flexible network stack design framework to facilitate the development of protocols based on Synchronous Transmissions to implement a wide variety of network layer protocols, while introducing only limited memory and energy overhead [10].

2.1 Decentralized Brains

The distributed cognition framework in [11], for collaborative efforts between humans, defines distributed cognition as cognitive activities viewed as computations which take place via the propagation of various information and knowledge transformation through media. The media here refers to both internal (e.g. individual memories) and external representations (wireless communication and other sensing capabilities). The states of the representations refer to various information and knowledge resources transformation during collaborative manoeuvres. According to the study [11], the way knowledge is propagated across different representational states is characterized by communicative pathways that are continuously interrupted and coordinated sequences of action by the demands of an ever-changing environment. Here, the evolution of the nervous system in an octopus is used as an inspiration and the notion of distributed cognition is used to conceptualize the idea of *Decentralized Brains*. The representational states are reliably replicated between the collaborating systems in a distributed wireless communication architecture [2]. This allows for local actuation without global coordination and facilitates multiple parallel control for collaborative maneuvers [2].

In distributed systems, the most common replication topology is to have a single leader that then replicates the changes to all the other nodes in the network with the benefit of avoiding conflicts caused by concurrent writes [2]. All the clients are writing to the same server, so the coherence in the data is maintained by the leader [2]. As shown in Fig. 2, our data replication spans two radio physical layers with multiple state transitions starting from network discovery. This helps in reducing channel contention while keeping lower bounds in latency [2].

The consensus protocol is inherently developed as a centralized protocol with network discovery and leader election methods. The main motivation of this architecture is to understand and identify the workhorse of a consensus protocol and to reliably develop those networking primitives in a modular manner.

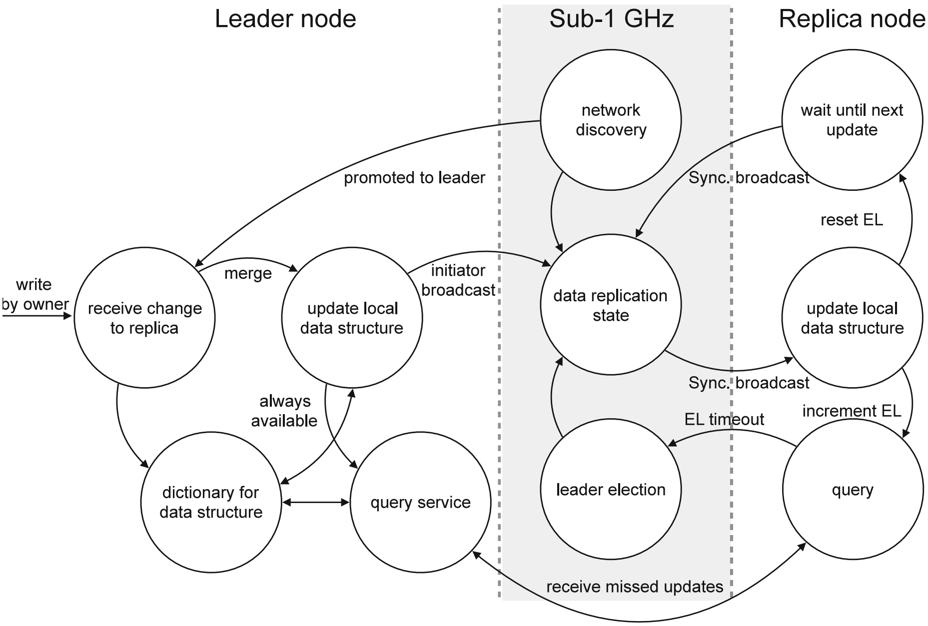

Fig. 2. *Decentralized Brains* networking architecture [2]

Modularity and reproducibility of the results in wireless sensor networks will facilitate further complex consensus algorithms to be implemented. From understanding and implementing a decentralized context broker [12], it can be drawn out that the fundamental requirement for a consensus protocol to reliably function is to have a robust atomic broadcast communication primitive within the network. Therefore, a method of reliable network flooding is identified in [2] and it is implemented taking into consideration the target hardware. An experiment is setup to prove that the broadcast communication primitive can be reliably deployed in Sect. 3.

The communication flow of the *Decentralized Brains* is illustrated in Fig. 2. There are two types of nodes in the network, the leader node and the replica node. Leveraging the fact that each node is equipped with a dual band networking, it is possible to operate two networking paradigms in parallel. One of the networks is deployed in 2.4 GHz networking which is a IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) network and the other network is deployed in the Sub-1-GHz network which performs the data replication operation among the nodes [2]. A cohesive network spanning over two radio networks is developed in *Decentralized Brains* to increase the communication throughput and decrease the latency for replication [2]. The data replication state is met when networking flooding is performed using the synchronous broadcasts to propagate the same message across a large number of nodes that are sparsely deployed in a large spatial area or in a dense environment [2].

In this article we take into consideration the dense environment as usually seen in industry 4.0 scenarios [2]. The data that needs to be replicated is sent from the originator of the data to the leader node [2]. The leader node initiates the synchronous broadcast which results in a network flooding in the sub-1-GHz band [2]. The leader node also receives the changes to replica and it manages the most recent replica of the data structure which will be propagated throughout the network using the synchronous broadcasts [2]. For synchronous broadcasts, a strict clock synchronisation is required between the nodes which is implemented and tested in Sect. 4. 6LoWPAN is the networking layer provided by Contiki-NG which is used to create the 2.4 GHz network for one-to-one communication between the originator data and the leader node [13]. This networking layer provides a multi-hop mesh network, therefore the originator and leader nodes can reliably communicate in harsh environments [13]. Since there are extensive amount of literature available discussing the reliability and the performance of a large-scale 6LoWPAN based networking, the focus of this work is to present the performance evaluation of the synchronous broadcasts which is a newly developed networking primitive for this hardware. In the following Sect. 2.2, the idea for using Contiki-NG as the background for the networking layer of *Decentralized Brains*.

2.2 Why Use Contiki-NG?

The Contiki-NG official documentation states the following and also delivers on those points which is the main motivation for using Contiki-NG.

> Contiki-NG is an open-source, cross-platform operating system for Next-Generation IoT devices. It focuses on dependable (secure and reliable) low-power communication and standard protocols, such as IPv6/6LoWPAN, 6TiSCH, RPL, and CoAP. Contiki-NG comes with extensive documentation, tutorials, a road-map, release cycle, and well-defined development flow for smooth integration of community contributions.

Above all of this, it also liberates the developer and provides an operating system that is hardware agnostic. We strive to develop the *Decentralized Brains* also with the same philosophy as Contiki-NG to provide with the developers to apply synchronous broadcast communication primitive however possible in applications and not only for a distributed consensus in low-power WSNs. Moreover, the software support with well mature features that are required for developing the *Decentralized Brains* networking layer is provided by Contiki-NG. This allows for the extending the *Decentralized Brains* networking paradigm further into other hardware systems and applications. The network stack of Contiki-NG is shown in Table 1 which shows the necessary components that are developed for the IEEE 802.15.4 standard. It provides the Medium Access Control (MAC) layer for the 6LoWPAN networking using Carrier Sense Multiple Access (CSMA)/CA. The operating system also provides the flexibility to turn-off and on the features during run-time which is a requirement for developing the synchronous broadcasts communication primitive. Here the MAC layer is extended to provide the

Table 1. Contiki netstack along with *Decentralized Brains* netstack

OSI layers	Contiki-NG	Decentralized Brain implementation
Application	Web-socket, http-socket, coap.c	DeBr
Transport	udp-socket, tcp-socket	
Network, routing	uip6, rpl	Not required
Adaption	sicslowpan.c	
MAC	csma.c	Time synchronised scheduling
Duty cycling	nullrdc, contikimac	Synchronous broadcast
Radio	MSP430, CC1350, ...	CC1350 (Sub-1-Ghz)

functionality for synchronous broadcasts as the opposite of listen-before talk is required during this type of network flooding. In this network flooding, all nodes that are going to participate in the synchronous broadcast round will not listen to the channel rather transmit the payload as soon as the time constraints are met.

With a strict time synchronisation protocol, an effect called constructive interference is achieved between the nodes that are simultaneously transmitting in the medium [2,7]. The Radio Duty Cycling (RDC) layer and the MAC layer are the two layers that require flexible and scheduled operation using the synchronised time and the modularity provided by Contiki-NG in such granularity makes it easier to implement features that are necessary for *Decentralized Brains* communication layer. A reproducible, modular code base that can be used to develop applications is the goal of *Decentralized Brains* and Contiki-NG accelerates this process.

3 Communication Overview

In this section, the workhorse of the *Decentralized Brains* is presented. For any consensus algorithm to function reliably, a robust broadcast mechanism is required. Such a broadcast mechanism is identified and developed for the large-scale low power consensus [2]. In WSNs, network flooding a concept to distribute information updates quickly to every node in a large network. In the case of *Decentralized Brains*, we developed a method that was presented in [7] called Glossy, where the effect of constructive interference is achieved to reduce the time and reliably flood the network with the data that needs to be distributed across the network. The concept of synchronous broadcast is presented in the following Sect. 3.1. There are certain requirements for the design of the presented concepts which arise due to the choice of hardware. The features for low power operation which are considered as design choices for implementing the necessary components of *Decentralized Brains* are discussed in Sect. 3.2 followed by the implementation details in the target hardware which is discussed in the same Sect. 3.2.

3.1 Synchronous Broadcast

When a node is transmitting in a wireless medium, the node is broadcasting the packet information in the channel for all the nodes in the network. Due to this broadcast nature of wireless communications, interference occurs whenever stations are close enough to listen to each other and whenever they are transmitting concurrently [7]. Collision or interference is an effect due to the overlap of signals in both the time and space when the transmitting nodes share the same physical layer characteristics [7]. Digital wireless communication implements multiple factors for redundancy in radio communication to mitigate for interference. Therefore, the communication link reliability becomes a probability in success of transmission from a Boolean if the communication will be successful or not for a given scenario. Interference is one such factor that reduces the probability that a receiver will correctly detect the information embedded in the wireless signals [7]. Interference has been always considered as destructive in nature owing it to the fact that the communication reliability reduces due to interference. There are two types of effects due to interference which can be called *Constructive* and *destructive* interference [7].

We explore the nature of constructive interference in this case to develop the most important communication primitive of *Decentralized Brains*. If the overlap of the transmitted signals does not superpose with each other, then the interference is destructive and the probability of the transmitted data reaching the destination reduces. Whereas, when the base band signals from multiple transmitters superpose, the receiver detects the superposition of the transmitted signals that are generated by multiple transmitters [7]. For achieving the effect of the constructive interference it is necessary for the transmitted base band signals to be within a time window with respect to the carrier frequency used for communication to allow for the detection of superposition. The time window is strict as the mismatch in the temporal synchronisation will effect the transmission reliability due to destructive interference. To achieve the constructive effect, it is necessary for the transmitting stations to be time synchronised. Due to the complexity, cost for energy, and the hardware support for reliable software execution required for time synchronisation, the effect of constructive interference has not been extensively exploited in WSNs [7,14]. In [7], a low power time synchronisation algorithm is implemented to achieve the effect of constructive interference whereas in *Decentralized Brains* a different approach is used which is discussed in Sect. 3.2. Using the time synchronization, we enable nodes to synchronize their clocks which is required for waking up and listening to a broadcast, calculate any skew, correct the clock and wait for the predetermined delay to retransmit. By synchronizing to another transmitters clock, we exploit the nature of constructive interference which allows for concurrent multiple transmissions to take place in the same channel. Using CSMA/CA MAC strategy, the broadcast medium is used efficiently for contention based wireless transmission by avoiding any collisions of two transmitting stations in the medium. But to allow for concurrent multiple transmissions using constructive interference, we enable the devices to transmit into the wireless channel all at the same time. Even though

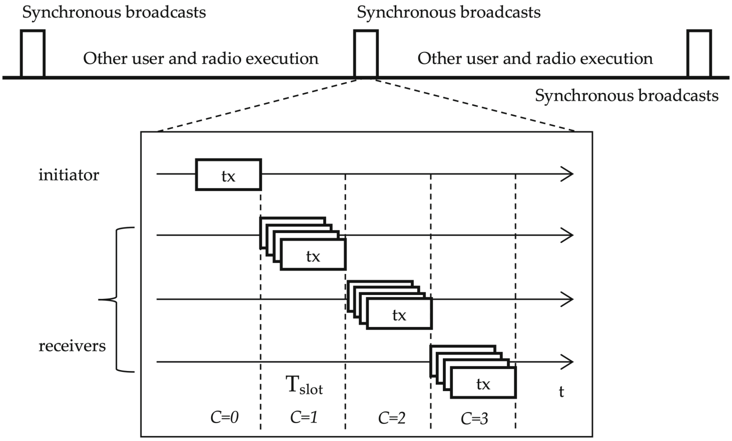

Fig. 3. Synchronous broadcast rounds with T_{slot} and counter C for number keeping track of number of transmissions [2,7]

it is contrary to the operation of the well proven CSMA/CA MAC strategy, with the help of strict time synchronisation we make it possible for the devices to transmit simultaneously. By creating the opportunity for nodes to overhear packets from neighboring nodes using [7], nodes turn on their radios, listen for the transmitted packets over the wireless medium, and relay overheard packets immediately after receiving them with an allowed software delay where time synchronisation is performed amongst the nodes. Since the neighbors of a sender receive a packet at the same time, they also start to relay the packet at the same time. Here the time at which each node transmits after reception is governed by the time synchronisation and which nodes are allowed to transmit from the set of received messages are defined in the communication protocol. This again triggers other nodes to receive and relay the packet. In this way, glossy benefits from concurrent transmissions by quickly propagating a packet from a source node (initiator) to all other nodes (receivers) in the network [7]. Based on [7], the temporal offset among concurrent IEEE 802.15.4 transmitters must not exceed 0.5 µs to generate constructive interference with high probability.

As shown in Fig. 3, the radio transitions between three states looping between two states until the counter is exhausted. When a new node joins the network without using the network discovery, the node synchronizes its clock with the root node by listening to the broadcast. The broadcast frame carries a counter which is used for determining the number of transmissions the node has to make with the same packet. Once the counter is 0, the node goes to sleep until the next broadcast round.

3.2 Design and Implementation

There are two main differences between the implementation of Glossy [7] and synchronous broadcast in *Decentralized Brains* [2]. The target hardware where synchronous broadcast is developed is a dual-band SoC (2.4 GHz & Sub-1-GHz) which can run two radio networks in two different physical layers. We choose the 2.4 GHz for data communication in a multi-hop mesh network topology where the nodes communicate with addresses in a 6LoWPAN IPv6 networking paradigm. The synchronous broadcast is developed in the Sub-1-GHz physical layer. Two main reasons for such a choice in developing the synchronous broadcasts is to increase the range of each of the nodes and to reduce the strict temporal distance window to improve the reliability of synchronous broadcasts. Moreover, it leverages the flexible, hardware specific simplelink SDK for deploying a low power multi-radio network. We precisely developed the synchronous broadcast in the 868 MHz band because the temporal distance is higher to achieve the effect of constructive interference and it allows the nodes to be synchronized less frequently. Therefore, the precision of synchronisation required can be less compared to glossy [7].

Why CC1350? is because of the ability to run two different radio networks in two different physical layers and also that the simplelink SDK allows it to be ported to other micro-controllers from TI to allow developers to freely choose hardware depending on the application scenario.

Due to the choice of the hardware and the development of synchronous broadcast in 868 MHz, it is not necessary to implement the time synchronisation explicitly as required in Glossy [7] as the temporal distance required for performing reliable constructive interference increases. Here we choose to implement the time synchronisation using the two hardware clock sources and not using an extra hardware capability to track the clock changes using capture units as the requirements are performed inherently by the hardware.

Implementation: Time synchronisation is required for synchronous broadcasts to achieve constructive interference. It is implemented in [7] where time synchronisation depends on 2 clock sources which are the Real-Time clock (RTC) and another higher frequency clock. The high frequency clock is sourced from the micro-controller's internal clock. It is widely known that this clock is unreliable due to skew and micro-controller operations. RTC provides better accuracy against clock drifts on the long term and the higher frequency clock provides better time resolution to achieve accurate synchronous broadcasts. For the implementation in [7] the Virtual High-resolution Time (VHT) approach [15] was used, which uses both the RTC and an internal high-frequency Digital Controlled Oscillator (DCO) from MSP430 MCU. VHT ensures high precision in time synchronisation and low power consumption which has been thought to be an oxymoron in designing distributed low power electronics [15].

For our implementation Dual-band CC1350 MCU is used with TI Simplelink SDK. CC1350 is a dual band SoC that contains 2 ARM Cortex cores, one is the main MCU core that runs the user logic and the operating system and the second core is dedicated for the radio functions and dual operation. The dedicated radio core is implemented to power up and down depending on the radio usage for power conservation purposes.

CC1350 radio operations are scheduled and time stamped with a separate 4 MHz timer, called RAT timer. Therefore, radio operations scheduling resolution is limited to that frequency. This dedicated core/timer of CC1350 gives it an advantage of running flooding protocols as it guarantees exact timing of the execution of radio commands, hence it gives more deterministic control over temporal distance and required delays between Glossy floods.

VHT is used in Glossy but in the *Decentralized Brains* we use the hardware supported implicit clock synchronisation. This synchronization happens between the 4 MHz RAT timer and the 32 KHz RTC clock. As the RAT-RTC synchronization command is issued for the first time, the radio core waits for the next RTC tick to start the RAT timer. To handle the power downs of the radio core, every time before issuing the power down command a RAT stop command is issued. This command returns a synchronization parameter that is passed to the RAT start command at the next radio core power up. This method keeps consistency between RTC and RAT and preserves the required clock resolution and power limits.

Also another feature of CC1350 is that time stamping of received packets is done automatically by the RF core using RAT time. These two features simplify the calculations of time for the next flood and provide a deterministic behaviour of the temporal distance between the nodes. Instead of a software time stamping, calculating and waiting for a software-delayed time as in [7], *Decentralized Brains* implements a full hardware based approach on CC1350. To adhere to the requirements for low-power operational constraints, the *Decentralized Brains* software uses only event-based callbacks for scheduling of floods as well as other auxiliary run-time software components.

As soon as a non-initiator node receives its first flood packet, it stores its base time with the received packet time stamp, increments the packet relay counter and then transfers to transmission mode. The time between packets within a flood is fixed, so the node immediately issues a transmission command to the RF core to send after this fixed time. The RF core handles the power up and down and automatically wakes up before the transmission time. Depending on the specified number of transmissions, the node may switch to the receiving mode until the number of re-transmissions is done. Also as the time between floods is fixed, the node schedules the beginning time of the next flood with the last callback of each flood.

4 Experimental Setup and Results

This section presents the following three parts: *(i)* developing and performing experiments to prove the effect of concurrent transmissions, *(ii)* a large-scale experiment developed to analyse the performance of synchronous broadcasts and finally *(iii)* the performance analysis to understand the reliability of synchronous

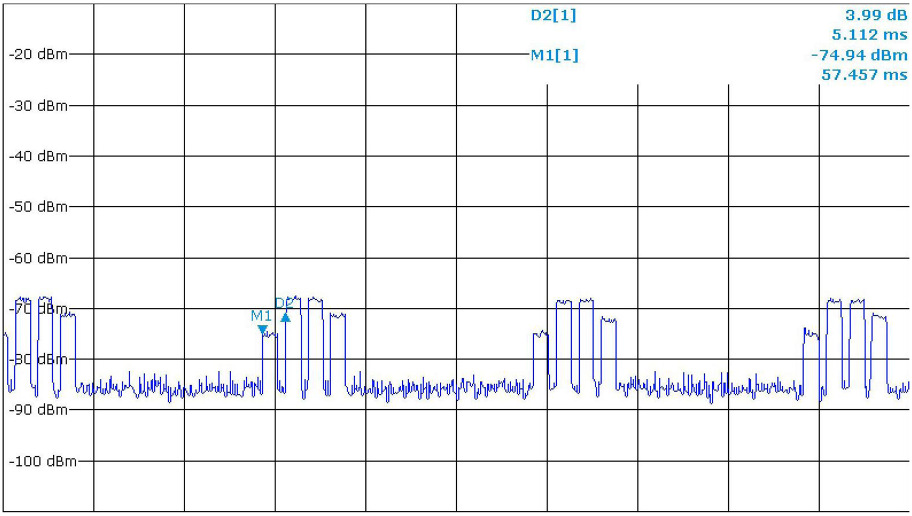

Fig. 4. Received signal power measured by spectrum analyzer. The X-axis is a time sweep of 2 s with distance between two vertical lines measuring 20 ms. The experiment is done using one initiator node and 3 non-initiator node.

broadcasts used in *Decentralized Brains* for developing decentralized consensus algorithms. Each experiment is performed with a different setup, hence the setup for each experiment will be explained individually. Initially the effect of concurrent transmission on received signal power and distortion are presented with the help of measurements using a spectrum analyzer. As discussed in Sect. 3.1, multiple nodes are transmitting simultaneously within an allowed window of time. This experiment visualizes the received signal power measured with the help of a spectrum analyzer listening at 868 MHz where nodes are placed away from the measuring antenna equidistantly.

4.1 Observing Effect of Concurrent Transmissions

Figure 4 shows the effect of concurrent transmission when there are two nodes which are the initiator node and only one non-initiator node. In Fig. 5, the same experiment is done with one initiator and 3 non-initiator nodes transmitting simultaneously with the allowed window of temporal displacement is shown. In both of the experiments including two nodes and four nodes, the nodes receiving the packets from the designated imitator node re-transmit the packet three times as soon as they receive it. The number of re-transmission is arbitrarily chosen as three to demonstrate the effect of constructive interference during synchronous broadcasts.

It can be observed from Figs. 4 and 5 that every time a transmission occurs, there are peaks in the signal acquisition which are marked by the markers *M1* and *D2*. During these transmissions, since there is more than one transmitting node, the received power of the signal and distortion increases with the number

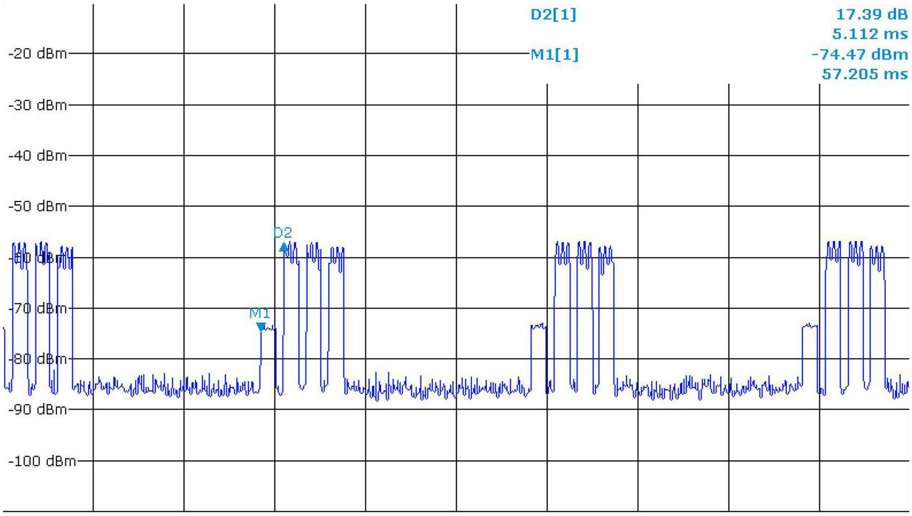

Fig. 5. Two nodes transmitting with different delays starting at 0 where absolute concurrent transmission occurs. The x-axis is the delay with a resolution of 0.25 µs. The experiment is performed for 20 times starting from 0.25 µs delay to 5 µs delay.

of concurrent transmissions. The first peak with the marker M1 is a packet transmitted by the initiator node hence, the measurement is not distorted and the received power is lower compared to the following peaks. The second and third peaks in the figures come from all of the nodes, hence they have the highest power and highest distortion. The last packet in each flood comes from non-initiator nodes, hence they have received a measured power slightly smaller than the previously received packets' measured power. This experiment is repeated for multiple rounds to ensure the effect of constructive interference.

In the *Experimental setup*, we present two experiments where both of the experiments were performed with a physical layer radio configuration of 50 Kpbs in a 2-GFSK modulation. The first experiment empirically quantifies the allowed temporal distance. This effect was already demonstrated for 2.4 GHz using the QPSK modulation in Glossy [7] but we perform the experiment in Sub-1-GHz band to prove the feasibility of constructive interference in this frequency band. Temporal distance is the allowed time window within which the effect of interference is constructive. The second experiment provides an insight into the effect of constructive interference due to the number of concurrent transmissions. It also helps in studying the effect of constructive interference in relation to the number of neighbors transmitting simultaneously.

The first experiment is performed in order to prove that constructive interference works for Sub 1-GHz and also calculates the minimum temporal distance. The experiment is performed by sending a signal from two nodes, one is delayed and the other one is non-delayed. To prove constructive interference and prevent the capture effect from making the receiving (RX) nodes capturing the

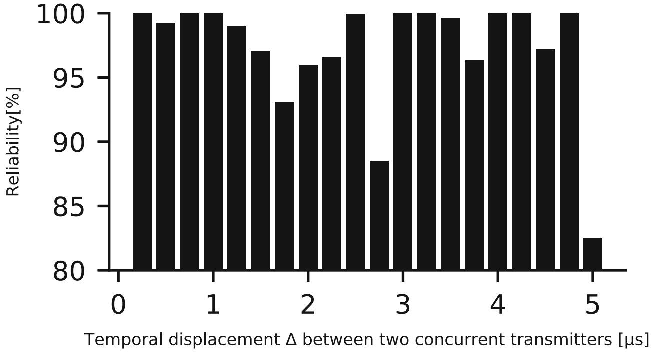

Fig. 6. The experiment studies the effect of temporal distance in case of two concurrent transmissions. The first transmitter sends the packet on the expected time and the other is delayed, The x-axis represents the amount of delay for each experiment with a resolution of 0.25 µs. The experiment is done for 20 times starting from 0.25 µs delay to 5 µs delay.

first signal, the non-delayed node sends at a power of −10 dBm and the delayed signal sends at a power of 0 dBm. The experiment is repeated multiple times increasing the delay by a step of 0.25 µs each time. As observed in Fig. 6, the Sub-1-GHz behaves in a similar way to 2.4 GHz [7] where the shift in temporal distance results in several valleys and peaks. The signals get constructively and destructively interfered, but Sub-1-GHz (2-GFSK) seems to be prone to bigger temporal distances than 2.4 MHz as demonstrated in Glossy [7]. Figure 6 shows that the allowed temporal distance for Sub 1-GHz to reliably perform synchronous broadcasts due to constructive interference can be 1.25 µs. The signals can constructively interfere with a reliability of 97.06%.

The second experiment studies the effect of a number of neighboring nodes during synchronous broadcast rounds. The experiment starts with two nodes transmitting simultaneously. With each round of synchronous broadcast, the number of participating nodes is increased with an extra node. As observed in Fig. 7 flooding works at a 99.99% when there are 2 to 3 neighboring nodes. The behaviour starts to deteriorate in the experiment as more nodes join the synchronous broadcast round. After three nodes with a high reliability the nature of interference slowly shifts reaching a reliability of 72% when there are 8 neighboring nodes. Even though the reliability is above 70% for 8 nodes, this means that for every 10 packets sent, the constructive interference effect cannot be seen and three packets are lost. With a spatial long stretching, densely populated network for distributed sensing or collaborating control systems, this reliability can still offer certain capabilities that were not possible previously.

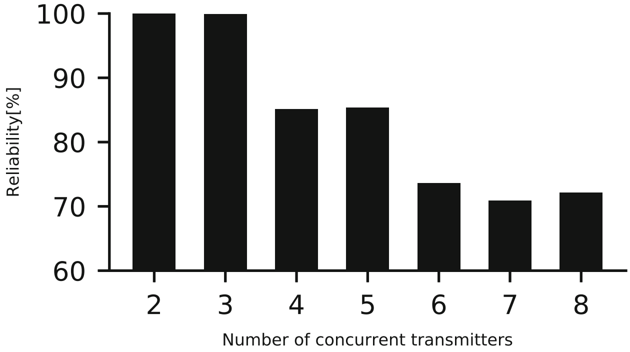

Fig. 7. This experiment studies the effect of the number of neighbours on the reliability of the packet received. The number of concurrent transmitters starts at 2 and increased by 1 for each next experiment.

4.2 Performance Evaluation

In this section we test our implementation on a local test bed (sensor-floor). We use a 345 node deployment to test the implementation. The 345 node deployment is called the sensor-floor which is a sensor network array embedded under the floor with CC1350 sensor tags. The floor contains 345 nodes, out of which, we choose a subset of 101 nodes randomly for every experiment run to count for the effect of spatial orientation and environment of the nodes for multi-path and scattering effects. 1 Initiator and 10 nodes for 10 concurrent transmissions are chosen randomly from the network of 345 nodes. Additionally these random patterns will allow for testing against different pattern arrangements of nodes along with the distance between nodes in the range of 1 to 30 m. The pattern choice and the sequence of transmissions performed for analysing the characteristics of constructive interference is listed in the Table 2.

As the Sub-1-GHz band can operate over greater distances which is greater than our test bed area. Therefore, we simulate the effect of hops as in the case of spatially distributed nodes in our experiments. Flood packets have their first byte used as a counter, this counter is set to zero by the initiator and incremented by 1 by all nodes with each re-transmission within a flood. Nodes are split into sequence groups, each group discards packets with a counter number less than a specific programmed number. As shown in Table 2 where the tests are presented, when the nodes are programmed to do 2 re-transmission (N = 2), with discarded packets marked as (x). Each sequence group keeps discarding flood packets until their programmed sequence number. When the sequence group identifier matches the counter byte then the sequence group joins the flood. In this manner a multi-hop test of 10 hops is simulated in our test bed for evaluating performance of spatially distributed nodes.

A flood is considered successful when a node receives the flood packet with the correct CRC (cyclic redundancy check) at least once, but we do not count floods in case no packets are received. The success rate is the rate of successful

Table 2. Testing on the sensor floor for re-transmissions (N = 2). For each of the 10 sequences, 10 nodes transmit concurrently. Each sequence is programmed to discard packets less than their sequence number based on the counter byte in the packet (First byte)

Node/re-transmission	0	1	2	3	4	5	6	7	8	9	10	11	12
Initiator	TX	RX	TX										
Sequence 0	RX	TX	RX	TX									
Sequence 1	x	RX	TX	RX	TX								
Sequence 2	x	x	RX	TX	RX	TX							
Sequence 3	x	x	x	RX	TX	RX	TX						
Sequence 4	x	x	x	x	RX	TX	RX	TX					
Sequence 5	x	x	x	x	x	RX	TX	RX	TX				
Sequence 6	x	x	x	x	x	x	RX	TX	RX	TX			
Sequence 7	x	x	x	x	x	x	x	RX	TX	RX	TX		
Sequence 8	x	x	x	x	x	x	x	x	RX	TX	RX	TX	
Sequence 9	x	x	x	x	x	x	x	x	x	RX	TX	RX	TX

Table 3. Overall success rate of testing on sensor floor

Concurrent transmissions (N)	Average success rate	Max no. of concurrent transmissions
N = 1	98.72%	10
N = 2	97.51%	20
N = 3	97.94%	30
N = 6	99.50%	50

floods over the overall number of floods averaged among all nodes on the floor. Each of the tests runs for 50000 packets.

Table 3 shows the result for the different number of re-transmissions. As it can be observed the increasing of re-transmissions helps in increasing the success rate. The success rate can reach 99.50% when there are 6 re-transmissions per synchronous broadcast flood round.

4.3 Energy Profile of *Decentralized Brains* Broadcast

In this section, the energy profile of the synchronous broadcast experiment is presented with the help of energy measurements using a oscilloscope. 1. as labelled in Fig. 8, startup phase or boot phase from the low energy mode to receive mode. Here initialization of all timers, clock synchronization, and the timers to extrapolate the high-speed clock from the slow clock are performed.

The second peak i.e., 2. as labelled in Fig. 8, is the reception of the first message from the initiator during the synchronous broadcast round.

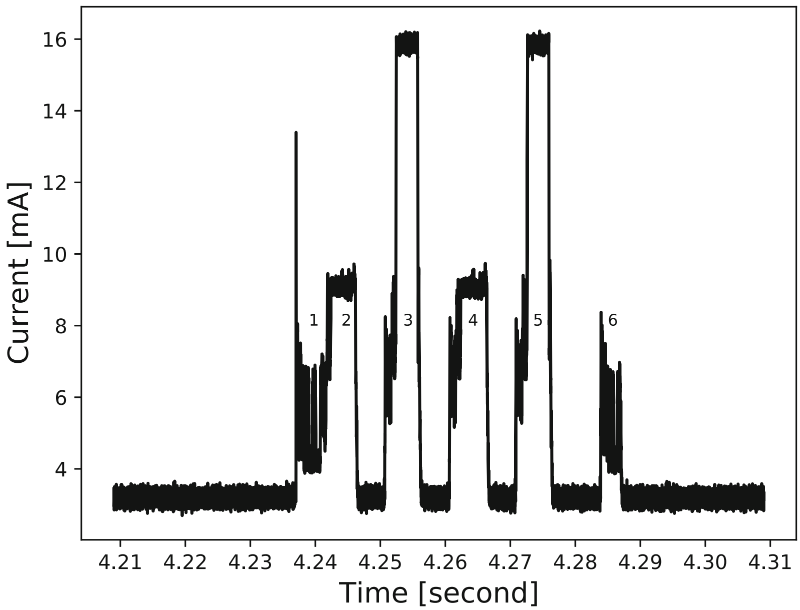

Fig. 8. Energy profile of synchronous broadcast

Until the third transient, label 3. in Fig. 8, the observed current draw, is the pre-defined software delay to synchronize all transmitters after message timestamping. The third peak is a decentralized synchronous transmission, where all other nodes are expected to be within the guaranteed temporal distance for effective constructive interference during transmission. The third peak observed in the figure is higher than the previous peak as transmission requires more energy than reception.

After taking part in the first transmission round, the node stays awake, listening to the next message to be transmitted again. There are four synchronous transmissions. The device under investigation will send only half the time as the alternating times; the node will listen for the same payload and clock synchronization.

The fifth peak (label 5 in Fig. 8) is the alternating second synchronous transmission attempt by this node. It is crucial to create an optimization plan across all nodes on the number of times a node can participate in synchronous broadcast transmission. In this scenario, a round is with four attempts, and every node can only participate two times. The larger and denser the network, the number of transmissions within a flood round should be increased. Reducing the participating nodes per round will optimize the network's overall energy cost.

The sixth peak (label 6 in Fig. 8) is the final round, where the node schedules its next network flood round to wake up if the application wants to conserve energy or allows other application-based processing and communication. The energy required is relatively lower than the reception and transmission since the

CPU schedules a timer enabling all the interrupts and returns to idle in this experiment scenario.

Cumulatively calculating the energy requirements from the measured energy profile, there are four synchronous broadcast rounds. We can calculate the overhead for decentralized synchronization as two transmissions and two receptions, each of which requires 13.4 mA at 10 dBm and 5.4 mA, respectively, as per the data sheet [16]. Use case dictates the frequency of decentralized synchronization that limits the recurrence of the synchronous broadcast. We can plan broadcast rounds with up to ten minutes of time delay between each round. This is the allowable limitation for time synchronization using message time stamping, after which the nodes cannot guarantee the temporal distance for synchronous broadcasts. Therefore, we can achieve decentralized synchronization at its highest level in the low-power wireless sensor networks with its calculated energy overhead required to perform the necessary communication primitives.

5 Conclusion

Decentralized Brains is a framework for developing distributed sensing and collaborative control systems in low-power, low data-rate WSNs networks [2]. In this article, the most important feature of *Decentralized Brains* called the synchronous broadcasts using constructive interference is developed, deployed in 345 low-power nodes and tested for performance. Various design choices and differences to existing implementations of constructive interference are discussed. The synchronous broadcasts module for *Decentralized Brains* is developed as part of the Contiki-NG to improve the transferable nature of the concepts and to make the features accessible for other industrial application developers. Leader election and network discovery are two further modules that leverage the synchronous broadcasts to create a decentralized network. When the network loses the initiator node which acts as the leader node due to energy constraints or because the network is highly mobile as discussed for self-assembly of space architectures [17], the network enters into a leader election phase where all nodes decide on another leader node using the same synchronous broadcast communication primitive [2].

5.1 Future Work

Porting Implementation to 2.4 MHz. As Simplelink SDK provides a unified API across several TI MCU, our implementation has the possibility to also be ported to 2.4 MHz devices that use the same SDK, for example TI CC2650. As per [7], the temporal distance for 2.4 MHz is 0.5 µs which is possible to be achieved with CC2650. As the radio clock of CC2650 is also 4 MHz the minimum allowed temporal distance is 0.25 µs. The SDK API for both CC1350 and CC2650 is identical, therefore the effort to port the code would only involve changes to the RF settings and time between the synchronous broadcast network floods.

This will allow for running the currently developed *Decentralized Brains* synchronous broadcast mechanism in CC2560.

Development of a Full Stack. As TI simplelink SDK has the option to run multiple radio instances as part of a single micro-controller core, we leverage it to implement the synchronous broadcasts. It also allows us to develop a network abstraction to run multiple radio protocols for communication. *Decentralized Brains* is communication protocol which leverages Glossy for synchronous broadcast, but the whole communication paradigm spans over two physical layers using the dual-band SoC for IPv6 based addressable communication within the network and a routing-less data replication scheme. The reliable synchronous broadcast is a crucial feature for the data replication scheme which was developed, implemented, and evaluated in this paper. To make this possible, to improve reproducibility of the results and to increase the use of constructive interference for network flooding in industrial applications it is proposed to write a dedicated MAC and NET layers in Contiki-NG for the target hardware. Since the target hardware SDK already supports multiple low-power micro-controllers the same code base can be compiled and reused out-of-the-box. This would allow to make the communication stack more code friendly and application realistic.

Acknowledgment. Part of the work on this paper has been supported by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 "Providing Information by Resource-Constrained Analysis", project A4.

References

1. Godfrey-Smith, P.: Other Minds: The Octopus, the Sea, and the Deep Origins of Consciousness. Farrar, Straus and Giroux, New York (2016)
2. Ramachandran Venkatapathy, A.K., Ekblaw, A., ten Hompel, M., Paradiso, J.: Decentralized brain in low data-rate, low power networks for collaborative manoeuvres in space. In: Proceedings of the 6th IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE 2018), pp. 83–88. IEEE (2018)
3. Rubenstein, M., Cornejo, A., Nagpal, R.: Programmable self-assembly in a thousand-robot swarm. Science **345**(6198), 795–799 (2014)
4. Ramachandran Venkatapathy, A.K., Bayhan, H., Zeidler, F., ten Hompel, M.: Human machine synergies in intra-logistics: creating a hybrid network for research and technologies. In: Federated Conference on Computer Science and Information Systems (FedCSIS 2017), pp. 1065–1068. IEEE (2017)
5. Ramachandran Venkatapathy, A.K., Riesner, A., Roidl, M., Emmerich, J., ten Hompel, M.: Phynode: an intelligent, cyber-physical system with energy neutral operation for phynetlab. In: Smart SysTech 2015; European Conference on Smart Objects, Systems and Technologies, pp. 1–8 (2015)
6. Ramachandran Venkatapathy, A.K., Riesner, A., Roidl, M., Emmerich, J., ten Hompel, M.: Phynetlab: architecture design of ultra-low power wireless sensor network testbed. In: Proceedings of the IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM 2015), pp. 1–6 (2015)

7. Ferrari, F., Zimmerling, M., Thiele, L., Saukh, O.: Efficient network flooding and time synchronization with glossy. In: Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks, pp. 73–84 (2011)
8. Ferrari, F., Zimmerling, M., Mottola, L., Thiele, L.: Low-power wireless bus. In: Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems, pp. 1–14 (2012)
9. Al Nahas, B., Duquennoy, S., Landsiedel, O.: Network-wide consensus in low-power wireless networks. In: SenSys: Proceedings of the 15th ACM Conference on Embedded Networked Sensor Systems (2017)
10. Jacob, R., Bächli, J., Da Forno, R., Thiele, L.: Synchronous transmissions made easy: design your network stack with baloo. In: Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks (EWSN 2019), pp. 106–117. Junction Publishing (2019)
11. Rogers, Y., Ellis, J.: Distributed cognition: an alternative framework for analysing and explaining collaborative working. J. Inf. Technol. **9**(2), 119–128 (1994)
12. Ramachandran Venkatapathy, A.K., ten Hompel, M.: A decentralized context broker using byzantine fault tolerant consensus. Ledger, vol. 4 (2019)
13. Dunkels, A., Gronvall, B., Voigt, T.: Contiki - a lightweight and flexible operating system for tiny networked sensors. In: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks, pp. 455–462 (2004)
14. Swami, A., Zhao, Q., Hong, Y.-W., Tong, L.: Wireless Sensor Networks: Signal Processing and Communications Perspectives. John Wiley & Sons, Chichester (2007)
15. Schmid, T., Dutta, P., Srivastava, M.B.: High-resolution, low-power time synchronization an oxymoron no more. In: Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks, IPSN 2010, pp. 151–161. ACM, New York (2010)
16. Texas Instruments: CC1350 datasheet: CC1350 SoC with RF Core (2016). http://www.ti.com/lit/ds/symlink/cc1350.pdf
17. Ekblaw, A., Paradiso, J.: Self-assembling space structures: buckminsterfullerene sensor nodes. In: AIAA/AHS Adaptive Structures Conference, p. 0565 (2018)