Competition: RedNodeBus, Stretching out the Preamble

Antonio Escobar-Molero[2], Javier Garcia-Jimenez[1], Jirka Klaue[1], Fernando Moreno-Cruz[2], Borja Saez[2], Francisco J. Cruz[3], Unai Ruiz[3], Angel Corona[4]
[1]RedNodeLabs, [2]Infineon Technologies AG, [3]eesy-innovations GmbH, [4]Bernitz Electronics GmbH
antonio.escobar@infineon.com, javi@rednodelabs.com, jirka@rednodelabs.com

Abstract

A real-time wireless bus based on flooding and the capture effect is proposed to achieve highly reliable broadcast communication in a Wireless Sensor Network (WSN) working in harsh environments, in a multi-source-to-multi-sink topology, where multiple hops are required (Fig. 1). Different sources access the medium without colliding using network-wide predefined time slots and frequency channels. Frequency-, spatial- and time-diversities are exploited using redundant retransmissions. Packet deliveries are latency-bounded, and messages are discarded after a predetermined time-to-live; in order to achieve an optimal trade-off between reliability, energy consumption and latency. Furthermore, long packet preambles are used to ease the synchronization requirements and favor the capture effect.

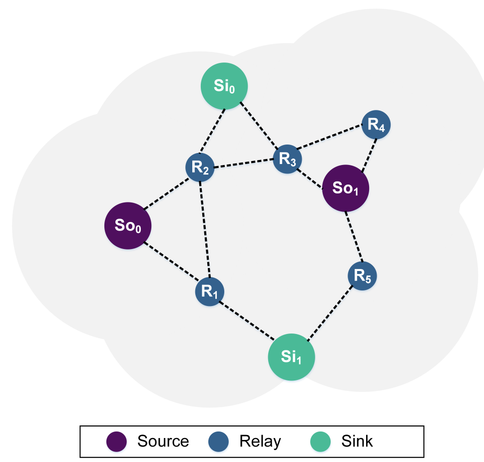

Figure 1. Multi-source-to-multi-sink topology

1 Introduction

RedNodeBus is based on RedFixHop (winner of the EWSN Dependability Competition in 2016 [8]) and BigBangBus (winner in 2018 [1]) protocols. These protocols use flooding, as opposed to routing, and Concurrent Transmissions (CT) to optimize reliability and latency, while keeping network operation simple, as in Glossy [7]. CT requires a tight synchronization accuracy between the simultaneous transmitters and suitable modulation techniques to decrease internal interference to tolerable levels [2]. Noncoherent FSK demodulators, and soft-decision receivers with Direct-Sequence Spread Spectrum (DSSS), like the CC2420, are particularly suited to CT-based protocols. While phase and amplitude properties are heavily distorted when several transmitters send the same packet simultaneously, frequency properties are mostly preserved (assuming temporal misalignments smaller than half the symbol period) [6].

The key feature of RedFixHop, BigBangBus and RedNodeBus is the usage of out-of-the-box thinking to overcome hardware limitations, while keeping complexity low.

In RedFixHop [8][3][2][4][5], the required synchronization precision is achieved using hardware-triggered retransmissions (automatic ACKs of the radio transceiver). However, many commercial transceivers—like the CC2420 used in the competition—limit the efficacy of this mechanism, restricting the payload length to only 1 Byte [2].

In BigBangbus [1], to overcome these limitations, payload-efficient software-triggered retransmissions exploiting the capture effect, which requires less demanding synchronization accuracy [9], are used. BigBangBus defines a decentralized wireless bus, with ultra-robust time synchronization—surviving even in the noisiest conditions—, together with a hard-to-jam channel hopping scheme.

RedNodeBus builds upon BigBangBus, extending the applicability of the protocol to longer payloads (successfully tested up to 64 Bytes), by increasing the length of the packet preambles in order to boost the chances of triggering the capture effect and decrease the tight inter-node synchronization requirements. RedNodeBus is a production-grade dependable protocol, including state-of-the-art security mechanisms, designed by RedNodeLabs [10].

International Conference on Embedded Wireless Systems and Networks (EWSN) 2019
25–27 February, Beijing, China

ISBN: 978-0-9949886-3-8

2 RedNodeBus

RedNodeBus is designed to work reliably in both multipoint-to-point and point-to-multipoint traffic patterns, due to its flooding and broadcast nature. It takes part in both Categories 1 and 2 of the Dependability Competition. The most relevant changes of the protocol with respect to BigBangBus [1] are:

- *Longer Preambles*. The internal DCO of the MSP430 is inaccurate, requiring complex calibration mechanisms if a time resolution higher than the one provided by the precise 32 kHz external crystal oscillator is needed. The time step of the 32 kHz oscillator (30.5 μs) is similar to the time required to send a byte (32 μs). It is challenging to implement a flooding scheme with guard times spanning a few byte periods, while keeping the tight synchronization required for concurrent transmissions to work (around 0.5 μs [7]). RedFixHop and BigBangBus were designed for extremely short packets (1 Byte payload). Therefore, packet losses and collisions due to synchronization errors were not so costly. In RedNodeBus, longer packets (up to 64 Bytes payload) force the implementation of better synchronization techniques, since time- and energy-costs of packet retransmissions are high. In order to keep the complexity low, flooding and channel-hopping synchronization mechanisms only use the slow 32 kHz oscillator. As a novelty, guard times between packet transmissions are generated using the accurate radio transceiver clock, by increasing the length of the preamble. Since receivers do not need to detect the whole preamble to synchronize, repetitions can be continuously triggered without calculating inaccurate delays with the microcontroller clock. An additional positive effect of using longer preambles is the enhancement of the capture effect. Since radio transceivers do not effectively start the packet reception until the SFD is detected, longer preambles favor the synchronization with the most energetic reception, as long as concurrent transmissions arrive with a relative delay lower than the total duration of the preamble. In RedNodeBus, the preamble is extended from the IEEE 802.15.4 standard specification, 4 Bytes (128 μs), to 10 Bytes (320 μs); adding an accurate guard time of 192 μs (more than 6 ticks of the 32 kHz oscillator). The cost of improving the synchronization robustness is the additional time and energy spent in every transmission, but it greatly pays off due to the decreased number of packet losses and collisions.

- *Message Expiration*. As a design choice, and to make the protocol suitable to both Categories of the Competition and an arbitrary number of sink nodes, it features no packet ACKs. Messages are repeated during a predefined time after their generation. After this time, which is dynamic and increases if the jamming level is perceived to be high, messages are either assumed to have reached their destination or considered too old to be delivered. Messages can only reach their destination with a latency lower than a predetermined threshold. As a result, average latency and energy consumption improve, while reliability slightly decreases. In low-noise conditions, the protocol behaves as a dependable real-time wireless bus, and packets are delivered within one bus period. In high-noise conditions, the protocol can be considered a best-effort latency-bounded wireless bus.

- *Synchronization Beacons*. As part of the process to adapt the protocol to longer packets, synchronization beacons are introduced. Highest energy consumptions and packet losses occur when relays are not properly synchronized with the protocol channel-hopping scheme. Beacons are very short packets (empty payload), which only contain synchronization information. Being so short, they are hard to jam and require less transmission power. Beacons are always sent at the beginning, during the set-up phase, and when there are no new messages to send (all the messages in the local buffer are considered too old to be retransmitted). The beacon strategy, by design, decreases overall energy consumption when messages are generated less frequently, effectively setting a network-wide low-power mode when no new messages are detected, while keeping the network tightly synchronized.

3 References

[1] A. Escobar, F. Moreno, A. J. Cabrera, J. Garcia-Jimenez, F. J. Cruz, U. Ruiz, J. Klaue, A. Corona, D. Tati, and T. Meyerhoff, "Competition: Bigbangbus," in *Proceedings of the 2018 International Conference on Embedded Wireless Systems and Networks*, EWSN '18, (USA), pp. 213–214, Junction Publishing, 2018.

[2] A. Escobar, F. J. Cruz, J. Garcia-Jimenez, J. Klaue, and A. Corona, "Redfixhop with channel hopping: Reliable ultra-low-latency network flooding," in *2016 Conference on Design of Circuits and Integrated Systems (DCIS)*, pp. 1–4, Nov 2016.

[3] A. Escobar, C. Gonzalez, F. J. Cruz, J. Garcia-Jimenez, J. Klaue, and A. Corona, "Redfixhop: Efficient ultra-low-latency network flooding," in *2016 13th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pp. 1–2, June 2016.

[4] A. Escobar, J. Garcia-Jimenez, F. J. Cruz, J. Klaue, A. Corona, and D. Tati, "Competition: Redfixhop with channel hopping," in *Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks*, EWSN '17, pp. 264–265, Junction Publishing, 2017.

[5] A. Escobar, F. Moreno, A. J. Cabrera, J. Garcia-Jimenez, F. J. Cruz, J. Klaue, A. Corona, D. Tati, and D. P. Morales, "Demo: Redfixhop - reliable real-time network flooding," in *Proceedings of the 2018 International Conference on Embedded Wireless Systems and Networks*, EWSN '18, (USA), pp. 197–198, Junction Publishing, 2018.

[6] A. Escobar-Molero, "Improving reliability and latency of wireless sensor networks using concurrent transmissions," *at - Automatisierungstechnik*, vol. 67, pp. 42–50, Jan. 2019.

[7] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "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, April 2011.

[8] J. Klaue, A. Corona, M. Kubisch, J. Garcia-Jimenez, and A. Escobar, "Competition: Redfixhop," in *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*, EWSN '16, (USA), pp. 289–290, Junction Publishing, 2016.

[9] K. Whitehouse, A. Woo, F. Jiang, J. Polastre, and D. Culler, "Exploiting the capture effect for collision detection and recovery," in *The Second IEEE Workshop on Embedded Networked Sensors, 2005. EmNetS-II.*, pp. 45–52, May 2005.

[10] "Rednodelabs: developing the next generation of wireless sensor networks," *https://www.rednodelabs.com*.