How Different Transceiver Hardware Effects Concurrent Transmissions in WSNs

Georg von Zengen, Alexander Baumstark, Alexander Willecke, Ulf Kulau and Lars C. Wolf
Institute of Operating Systems and Computer Networks
Technische Universität Braunschweig
Email: {vonzengen,abaumsta,willecke,kulau,wolf}@ibr.cs.tu-bs.de

***Abstract*—The efficiency and effectiveness of network flooding protocols has recently been shown by many publications as well as real world deployments. One notable protocol is Glossy which combines Concurrent Transmission (CT) with Constructive Interference (CI). A drawback of Glossy is that up to now it has only been implemented and evaluated for one type of radio transceiver. In this paper we first present the Glossy implementation for an AT86RF233 radio transceiver which simplifies the packet forwarding due to a shared receive and transmit buffer. We evaluate our implementation against the original Glossy implementation in a minimalist setup as well as in a real world testbed. However, we observed a noticeable difference in the timing accuracy of the transceiver chips but without adverse effects on the network's performance. For this reason, in the second part of this paper we provide a deeper investigation by using Software Defined Radios to emulate concurrent transmission for different types of transceivers. This work confirms the general benefit of constructive interference but also shows its limitations. Moreover, we give novel insight into hardware dependability of constructive interference in concurrent transmissions.**

I. Introduction

Dependability is a key factor for the success of Wireless Sensor Networks (WSNs) and ever-growing Internet of Things (IoT) applications. For this reason, protocols are required that can overcome challenges like RF-interference or individual tolerances, e.g., clock drifts. To underline the importance of this need, the International Conference on Embedded Wireless Systems and Networks (EWSN) started an annual dependability competition in 2016. Working groups from both academia and industry can attend to compare their communication protocols in a competitive environment on a real-world WSN testbed. Indeed, it is conspicuous that the winning teams of 2016 [11] and 2017 [15] both rely on flooding mechanisms. Moreover, both approaches, as well as the runner-ups [19], [6], are either directly or indirectly based on Glossy [7] which implements an efficient flooding mechanism together with Concurrent Transmission (CT) and Constructive Interference (CI).

However, existing research on Glossy-like protocols in larger testbeds as well as the Dependability Competition is based on one radio transceiver only, the Chipcon CC2420 [21]. As this transceiver is not recommended for new designs [21], other transceivers must be evaluated whether the CT is working as reliable as for the CC2420. Brachmann et al. [4] did some first work on using its successor, the CC2520, which is merely an improved version of the CC2420. There is also a Glossy implementation for the CC430[1]. Unfortunately, there is no evaluation available for this implementation and Liao et al. [14] showed that CT is not working on non-Direct Sequence Spread Spectrum (DSSS)-transceivers like the CC430. Thus, we do not consider this implementation in this paper but we present results of an implementation and evaluation of CT on an applicable transceiver from a different manufacturer, in particular the Atmel AT86RF233 [3]. Moreover, this transceiver has the advantage of a shared receive and transmit buffer, that potentially enables faster retransmissions, as copying of the content is not required. This advantage was already mentioned by the authors of the original Glossy paper [7] but not investigated by now.

The remainder of this paper is structured as follows: After discussing some related work in Section II, we describe the challenges and benefits of using the AT86RF233 for CT in Section III. In Section IV we evaluate the use of the AT86RF233 in comparison to the original Glossy implementation on the CC2420. This evaluation was performed in a lab setup to control as many parameters as possible. We compare both implementations on different metrics, mainly timing differences, Received Signal Strength (RSS) and Link Quality Indicator (LQI) that all lead to Packet Error Rate (PER). We also provide a novel investigation of the effects of CT to the LQI of a transmission. Considering that the LQI is the only metric that is able to indicate the amount of interference a transmission suffers from [3], the significance of LQI is underestimated in prior evaluations of CT protocols. To be able to evaluate our implementation in a more realistic environment we deployed a testbed inside our office building. The description of this testbed as well as the discussion of the results is part of Section V. As some of the findings in our evaluations and Section VI were contradictory to the existing literature, we used an Software Defined Radio (SDR) based channel emulator [23] to take a deeper look into the mechanisms that enable both, CI and CT. A brief description of the emulator and results of the evaluation are discussed in Section VII. To the best of our knowledge we are the first to provide real evaluations on the the impact of the delay between concurrent transmissions, the impact of CI in noisy environments and the significant hardware dependencies of transceivers. Section VIII concludes the paper.

[1]https://github.com/ETHZ-TEC/LWB

II. Related Work

Long before the presentation of Glossy [7], flooding-based protocols like Spin [10], Trickle [13] or Flash [16] have been a vital part of WSN research to allow a reliable broadcast of data messages with low latency. Compared to classical routing protocols like CTP [9] or RPL [25], flooding mechanisms do not need to maintain the context graph of a WSN to distribute data throughout the network. Moreover, flooding protocols are well suited to be used with underlying LPL MAC protocols, e.g., SpeckMAC [26] or X-MAC [5] which aim to duty-cycle the radio transceiver to save energy.

The general principle of flooding protocols is that received messages are retransmitted by each node via broadcast. However, the massive retransmissions lead to a broadcast storm that prunes the network performance resulting in an increased PER [28] due to collisions [22]. To overcome this issue Glossy exploits CI while flooding data through the WSN. In particular Glossy targets CI on *chip*-layer, where groups of 4 bits (nibble) that are mapped to a pseudo-random noise (PN) sequence of 32 physical-layer bits (chips) before Offset Quadrature Phase-Shift Keying (O-QPSK) modulation. Thus, according to the IEEE 802.15.4 standard for 2.4 GHz [1], a chip is transmitted every $T_C = 0.5\,\mu s$ at a data rate of 250 kbps. In further consequence, to enable a CI or non-destructive interference [24], respectively, the time-shift between concurrent senders has to be $< T_C$. This tough timing requirements lead to the fact that Glossy is both, a fast network flooding as well as an implicit time synchronization protocol [7]. With an average synchronization error of $\overline{E} = 0.4\,\mu s$, Glossy outperforms several other WSN time synchronization protocols, e.g. TPSN ($\overline{E} = 22.66\,\mu s$) [8], FTSP ($\overline{E} = 3.0\,\mu s$) [17] or RATS ($\overline{E} = 2.7\,\mu s$) [12].

The performance as well as the CT capabilities of Glossy have been evaluated in several publications [20], [24]. Furthermore, Glossy's usability for real world WSN has been proven, e.g., by the Dependability Competition of the EWSN (c.f. Section I). Nevertheless, Glossy has never been used on other radio transceivers than the Chipcon CC2420 or its almost identical successors.

The general feasibility and the preconditions of CT have been investigated in [14], [18]. Liao et al. [14] investigated the effect of the carrier frequency offset on CT. They also showed that DSSS is one of the main reasons why CT works. Rao et al. [18] analyzed how the number and position of concurrent transmitters affect the CT. As a result, more than three concurrent transmitters lead to a higher bit error rate than less transmitters.

III. Concurrent Transmission on AT86RF233

In this section we will elaborate the benefits and challenges that the AT86RF233 brings into CT. The most obvious advantage of the AT86RF233 is the shared receive and transmit buffer. In theory the correct utilization of this buffer should lead to a shorter and more constant inter-transmission time. When a CC2420 has to forward a received data frame, the entire frame has to be copied from the receive buffer to the transmit buffer via the Serial Peripheral Interface (SPI). In case of the AT86RF233, only a few bytes of the frame need to be read and written from and to a single buffer. This shorter and less complex data exchange with the transceiver should result in a more constant processing time which further leads to a better synchronized transmission of different nodes during network flooding.

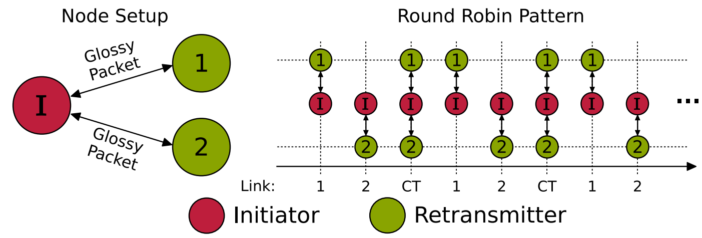


Fig. 1: Evaluation setup with one initiator (Node I) and two retransmitters, Node1 and Node2.

A. Implementation

In our implementation we focused on the basics to realize a CT system with minimum processing time variation. If a receive interrupt occurs, our implementation works as follows.

Algorithm 1 Glossy implementation on AT86RF233

```
read(protocol_identifier)
read(destination_address)
if protocol_identifier == CT && destination_address == broadcast then
    read(seq_num)
    read(hop_count)
    if seq_num not transmitted N times then
        write(hop_count+1)
        transmit packet
    else
        read(payload)
```

Our minimalistic approach – shown in Algorithm 1 – reads and writes a constant amount of bytes for every retransmitted packet independent of the payload size. With just five bytes to read and one byte to be written, the inter-transmission time is significantly shorter than for the original Glossy implementation that has to copy the entire frame whenever it is retransmitted. In contrast we read the entire frame only once when the `hop_count` has reached the configured value `N`. The decision whether a packet needs to be retransmitted or not is done based on a sequence number `seq_num` that identifies the packet and the `hop_count`. To prevent interference with other IEEE 802.15.4 networks our approach only retransmits broadcast packets with a predefined identifier `protocol_identifier`. However, at current stage the original Glossy as well as our implementation only perform in homogeneous networks. To achieve an interoperability in the future, our implementation would need to add additional and therefore curbing delays to slow down towards the payload-dependent Glossy implementation.

IV. Comparative Evaluation

To compare our implementation against the original Glossy implementation, we firstly choose a minimal evaluation setup depicted in Figure 1. The setup consists of one node that initially transmits a packet (Initiator) and two retransmitter nodes (Node1 and Node2) that retransmit the received packet back to the initiator node. The two retransmitters work in a round robin manner to minimize external influences on our evaluation results. In particular, the nodes are configured that only every third packet is retransmitted with CT and the others are transmitted either by Node1 or Node2. Thus, we get three different links: the CT link, the link from retransmitter Node1 and the link from retransmitter Node2 (cf. Figure 1). That way we can assume that all the three links were exposed to the same external interference during the evaluation. All nodes in the setup were either equipped with an AT86RF233 when we evaluate our implementation or with a CC2420 when the original Glossy was evaluated. We chose this minimalist setup, as Rao et al. [18] showed that a higher number of nodes is likely to decrease the performance of CT.

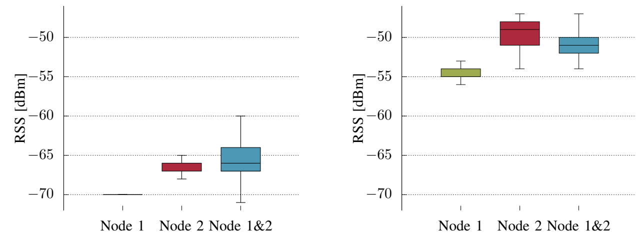
Initially, we measured the packet loss of both, our implementation and the original Glossy implementation. To do so we configured the retransmitters to transmit the packet just one time (`N=1`) and measured how many of these transmissions were received by the initiator. In total for each implementation the Initiator initiated about 5000 transmissions with a spacing of $\approx 50\,\mathrm{cm}$ between the nodes.

The results listed in Table I show that a packet loss of around five percent can be observed for both implementations. Admittedly, our implementation has a slightly higher packet loss for all configurations compared to original Glossy. As this increased packet loss appears for all configurations it is likely to be an effect of the AT86RF233, HF-paths or antennas used. However, the most important result of this evaluation is that neither our nor the original Glossy implementation was able to reduce the packet loss for CT significantly. The packet loss is relative high for such a small setup, this is likely due to the high WIFI usage in our office.

A value often used to classify CT and CI is the RSS [7], [24], [27]. This value gives the energy received by the antenna while the packet reception was in progress. The general idea is that the RSS increases with the number of concurrent transmitters. To evaluate this aspect we measured the RSS of every received packet at the initiator. In Figure 2a the results for the AT86RF233 are shown, where the box for Node1 is at $-70\,\mathrm{dBm}$ without variance. As expected the RSS is higher

TABLE I: Packet loss comparison between CT and no CT with only one retransmission for AT86RF233 and CC2420

Sending Nodes		Packet Loss [%]	
Node1	Node2	AT86RF233	CC2420
✓		5.764	5.096
	✓	6.715	2.166
✓	✓	6.603	4.841



(a) AT86RF233 (b) CC2420

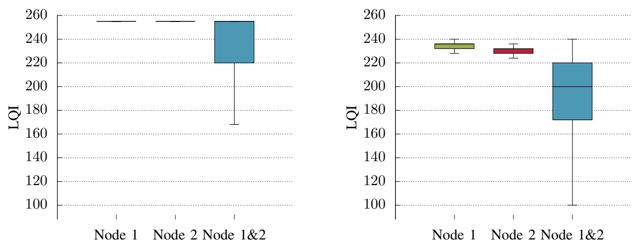
Fig. 2: Comparison of RSS values for the three different retransmission scenarios: only Node1 or Node2 are retransmitting or both (CT).

if both retransmitters are transmitting at the same time (Node 1&2 box in Figure 2a). On the one hand, the maximum value of $-60\,\mathrm{dBm}$ for concurrent retransmission is way higher than for both single retransmission links. But on the other hand, the minimum value is also lower than for non CT configurations.

The RSS values for the CC2420 – shown in Figure 2b – look slightly different. Besides the fact that the received energy was overall higher, which strengthens our assumption about the difference in packet loss between the transceiver chips, there is no clear sign that the RSS is higher for CT configurations.

As the RSS does not give us a clear answer how the AT86RF233 performs in CT, we also measured a much more complex and much less appreciated metric for wireless transmissions, the LQI. The LQI gives an estimation of the sameness between the received chip sequence and the most equal one from the IEEE 802.15.4 standard [1]. To give a better understanding why we choose the LQI and why it is a better classifier for CT than the RSS we explain it in the following paragraph.

As mentioned above, in IEEE 802.15.4 each four bit (nibble) are represented by a 32 bit long chip sequence on the physical-layer, therefore these bits are called chips. As nibbles have 16 different states, there are only 16 different chip sequences needed to represent every possible communication in the MAC-layer. These 16 chip sequences are defined in the IEEE 802.15.4 standard along with the represented nibble. If a transceiver receives a chip sequence it tries to match this sequence with the ones from the standard and then selects the one with the lowest difference. Afterwards this sequence is used to derive the MAC-Layer bits. The LQI is calculated from the difference between the received chip sequence and the selected chip sequence. In IEEE 802.15.4 it is only standardized that a lower LQI means a higher difference and that it ranges from 0 to 255. Hence, for each IEEE 802.15.4 compatible transceiver chip the LQI gives an estimation of the chip error rate and therefore an estimation of how much the signal was interfered during its transmission. As the RSS only gives the received energy, no matter whether the signal belongs to the received packet or is just noise, the RSS is not suitable to distinguish whether a CT is constructive or destructive. In sum, the LQI seems to be the more appropriate metric to analyze the potential effect of CI during CT which



(a) AT86RF233 (b) CC2420

Fig. 3: Comparison of LQI values for the three different retransmission scenarios: only Node1 or Node2 are retransmitting or both (CT).

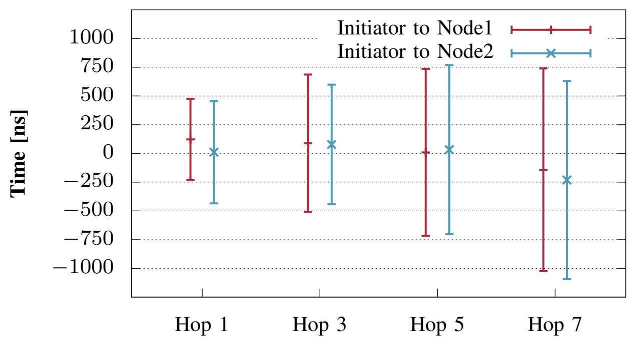


Fig. 4: Time synchronization accuracy over multiple hops.

is also investigated later on in Section VII.

In Figure 3a the results of the LQI evaluation for the AT86RF233 are shown. For the two single transmission configurations the boxes are very thin because the LQI was always 255. In contrast, for the CT configuration – Node 1&2 – a much larger variance of values can be observed. Thus, we can ascertain that not all of the CTs perform CI.

For the LQI evaluation of the CC2420 the results in Figure 3b show a similar result. Here the LQI was not 255 for all single transceiver packets, which might be due to a different conversion from the chip errors to the specific LQI value, as this is not standardized. But for the CT configuration the same effect was observed: the variance in LQI values rises significantly. An effect that could not be observed for the CC2420 is that the LQI in CT should lead to higher values than in single transceiver configurations. Even the maximum value of LQI is never overshot which would be expectable. This indicates that the original Glossy was not able to introduce the intended CI in our setup.

As Glossy is also known for its good time synchronization capabilities we evaluated this aspect of our implementation as well. Therefore, we configured our retransmitters to return the packet three times (`N=3`) and measured the difference in time between the initiator and the specific retransmitter after each hop.

In Figure 4 we show the mean value of the time synchronization (AT86RF233) with the standard deviation for both retransmitting nodes separately. As a result, both nodes stay within the range of the synchronization error of Glossy given by Ferrari et al., in particular, $\leq 0.4 \pm 4.8\,\mu s$ [7].

V. Testbed Evaluation

To evaluate the AT86RF233 and our implementation we deployed a testbed in our office building that is shown in Figure 5. The testbed consists of 13 nodes, where node 0 was configured as the initiator and the other twelve nodes as retransmitters. All nodes were equipped with a ublox NEO-M8Q GNSS-Module [2] to achieve a ground truth time synchronization.

As we have shown that the AT86RF233 is capable to perform CT reliably in a minimal lab setup, we evaluated the same in our larger testbed. For different configurations of `N` (`N=1,···,5`) the initiator transmits a packet into the network and each retransmitter retransmits the packet `N` times. For the evaluation we gathered the hop-counts of each received packet and from that we calculated the packet delivery rate over the number of retransmissions. We observed that nodes that are located far away from the initiator need more retransmissions to be reached reliably. For example, for `N=1` a delivery ratio of only 82 % is achieved for node 3 while the delivery ratio at node 4 is higher than 99 %. In our testbed `N≥3` retransmission were enough to reach an average reliability of 99 %.

VI. RX Jitter issue

During the implementation we noticed a variation of the timing of the reception start interrupt (*rx_start*) on different receiving AT86RF233s. To measure that jitter we used one transmitting AT86RF233 and two receiving radios. On both receiving nodes we measured the timings of the *rx_start* interrupts with a logic analyzer. The *rx_start* interrupt indicates the reception of a frame and is triggered directly after a valid PHY header was detected.

As illustrated in Figure 6 the *rx_start* jitter (J_{rx_start}) affects the *tx_start* jitter (J_{tx_start}) additively. The processing jitter ($J_{processing}$) is related to the phase of the processor's clock as well as potentially inefficient implementation. To verify that we do not add too much jitter in our implementation, we also measured the timing of the *tx_start* interrupt in the same setup. We performed this evaluation for both types of radio transceivers, the AT86RF233 with our software implementation and the CC2420 with the original Glossy implementation, respectively. In Table II the results of this measurements are summarized.

While both implementations show almost ideal mean jitters, the standard deviation of the AT86RF233's J_{rx_start}

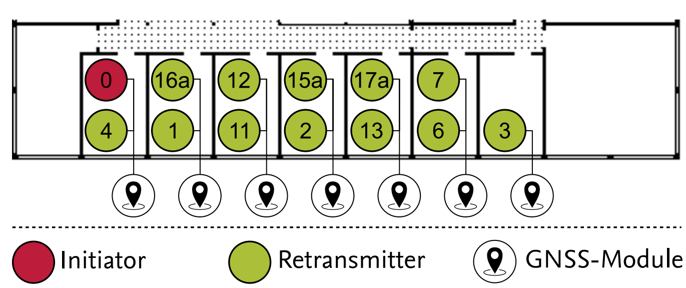


Fig. 5: Testbed setup in our offices with connected GNSS-Modules for ground truth time synchronization.

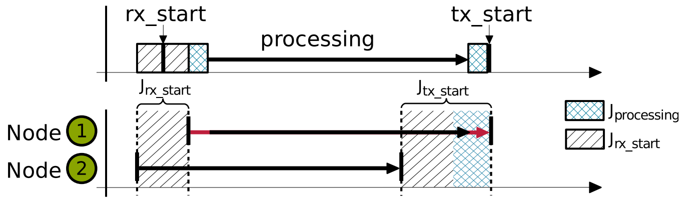


Fig. 6: Illustration of the *rx_start* and *tx_start* jitter.

TABLE II: Reception-Jitters for AT86RF233 and CC2420

Transceiver	**Signal**	**Jitter** [ns]	$_{min}$**Jitter**[ns]	$_{max}$**Jitter**[ns]
AT86RF233	J_{rx_start}	2±398.689	-1063	1063
	J_{tx_start}	40±429.083	-1313	1313
CC2420	J_{rx_start}	0±59.439	-188	188
	J_{tx_start}	61±134.685	-376	500

is conspicuous and, certainly, leads to a high standard deviation for J_{tx_start}. On the one hand, as $J_{processing} = J_{tx\_start} - J_{rx\_start}$, it can be seen that the processing jitter for both implementations $J_{processing}$ is negligible and show a comparable timing accuracy.

On the other hand the heavily fluctuating *rx_start* jitter J_{rx_start} leads to a serious issue but is solely dependent on the internal interrupt logic of the AT86RF233 [3]. Therefore, a mitigation of J_{rx_start} in software is impossible.

Nevertheless, although the jitter for the AT86RF233 is close to the 500 ns and therefore close to the threshold of CI [7], our evaluation in Section IV shows that the packet loss is not significantly higher as for one transmitter.

VII. Concurrent Transmission Emulator

The previous Section VI raised doubts if the synchronization of the AT86RF233 is accurate enough to perform CT in a beneficial manner, but the evaluations in Section V and Section IV showed a good network performance. Motivated by this discrepancy we decided to investigate the behavior of both transceivers in greater detail by using our CT-emulator [23]. This CT-emulator is based on two *HackRF One*[2] SDRs. The first one receives the signal from a CC2420 or AT86RF233 transceiver and forwards it to *GNURadio*[3] for further signal processing. In *GNURadio* signal-processing blocks can be utilized to manipulate the signal according to the situation we need, e.g. to emulate CT. Afterwards, the second *HackRF One* transmits the manipulated signal to a second CC2420 or AT86RF233 transceiver. The setup with CC2420 based sensor nodes is shown in Figure 7.

[2]https://greatscottgadgets.com/hackrf/

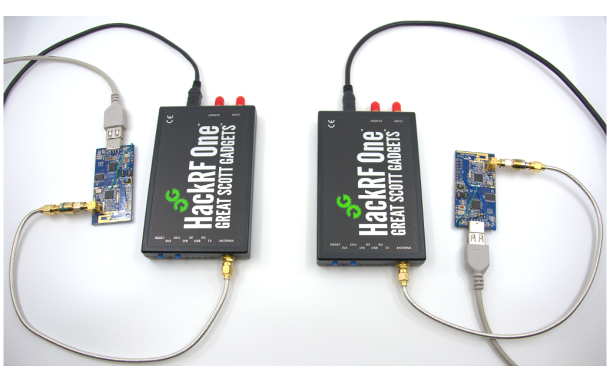


Fig. 7: Emulator setup used for our emulations with two CC2420 based sensor nodes connected to the HackRFs by SMA-cables.

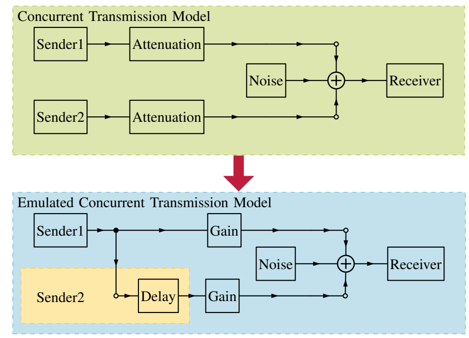
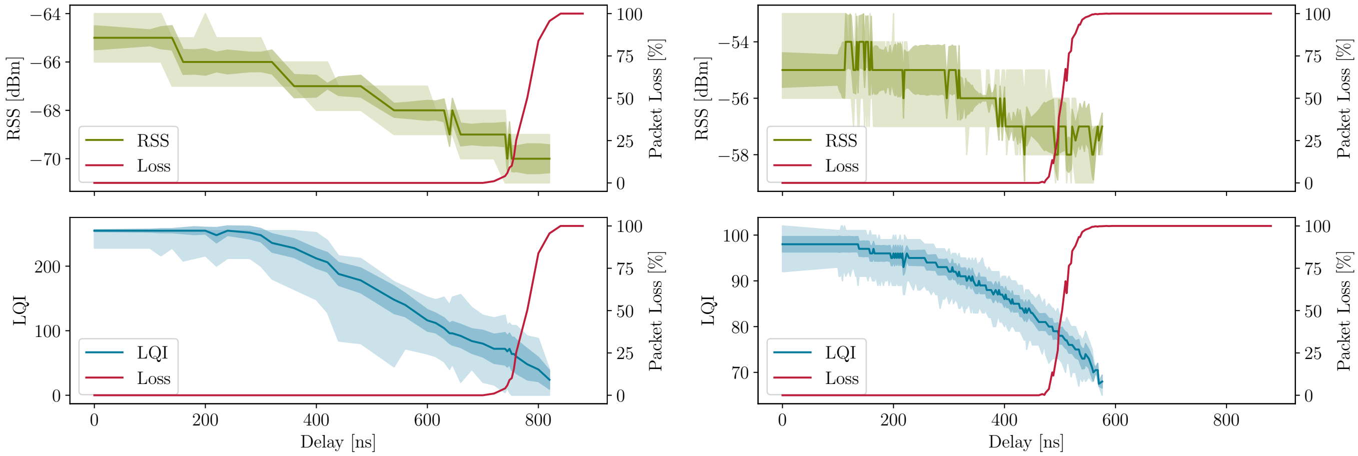


Fig. 8: Conversion from the real world CT model to our emulated CT model, using a *Delay*-block to emulate a second, delayed transmitter.

To emulate CT with only one transmitting CC2420/AT86RF233, we duplicate the received signal in *GNURadio*. With the *Delay*-block shown in Figure 8 we can model how much the two transmissions are delayed from each other. The two *Gain*-blocks are representing the *Attenuation*-blocks from the *Concurrent Transmission Model* and therefore emulate the different distances and transmission powers. Before the signals are transmitted to the second CC2420/AT86RF233, they are summed with a configurable level of noise. As the *Delay*-block delays the signal by a defined number of samples we are able to perform evaluations where all concurrent transmissions have the same constant delay. To the best of our knowledge this is the first setup that can evaluate large numbers of packets under the same CT conditions with signals transmitted and received by real WSN-hardware.

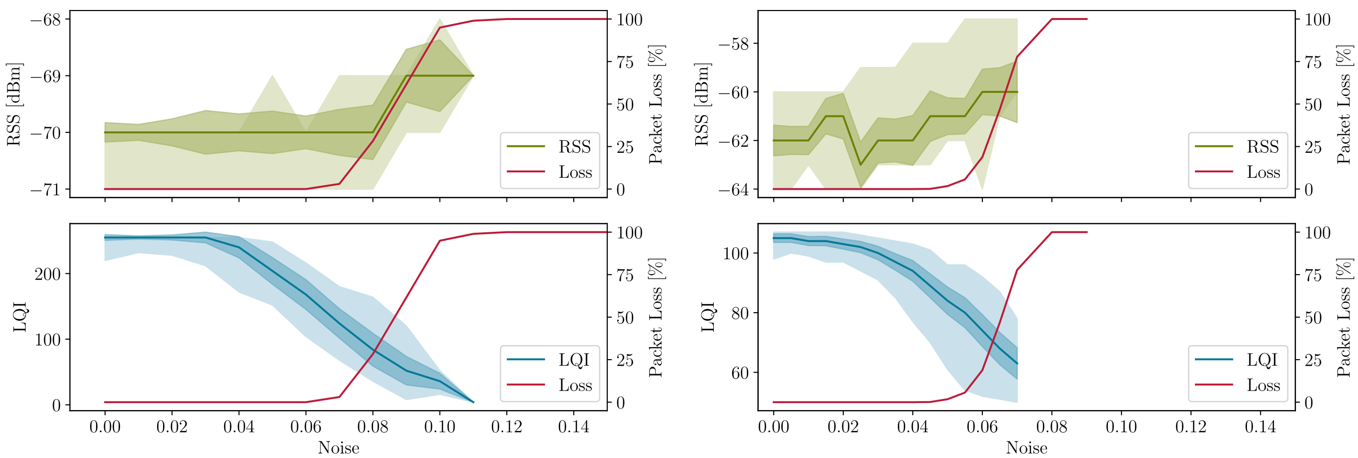
In the following, the described CT-emulator is utilized to get a deeper insight into the conditions under which CT works. Furthermore, we investigate how much CT's performance depends on the used hardware and transceiver, respectively. Therefore, we compare sender and receiver pair of the same type. For the performance analysis we measured the packet loss, the RSS, and the LQI under certain conditions. For each point we plotted in the following, there are at least 100 measurement. Most of the points are based on much more measurements, depending on the packet loss. In our first evaluation we considered an ideal environment. Thus, our emulator did not add any noise to the signal. In Figure 9 the measured packet loss, the RSS, and the LQI are plotted over the different transmission delays. The red line represents the packet loss, the green line is the median of RSS and the blue one the median of LQI. For both, RSS and LQI, the lightest

[3]https://www.gnuradio.org/



(a) AT86RF233 (b) CC2420

Fig. 9: AT86RF233 and CC2420 CT performance in an ideal evironment without noise.



(a) AT86RF233 (b) CC2420

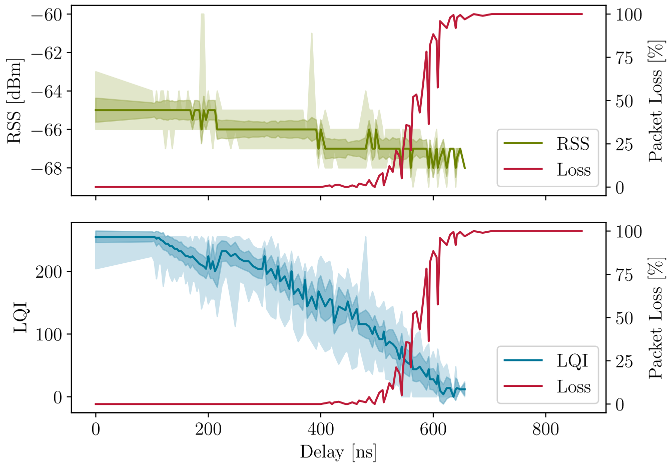
Fig. 10: Single transmission under noise influence. Noise is given without unit due to emulator limitations, signal amplitude is 0.5 in the same scale.

shade represents the area between the minimum and maximum values, while the darker area shows the standard deviation. In Figure 9a it can be seen, that the RSS and LQI are both degrading with rising transmission delay for the AT86RF233. While this result was somehow expected, the delay at which the packet loss is rising is notable. With 700 ns the practically measured limit for CI is significantly higher than the 500 ns postulated in literature, e.g. by Ferrari et al. [7]. Nevertheless, the LQI indicates that the CI starts to degrade significantly for delays > 250 ns. As there is no noise in the emulated channel, the destructive interference can be traced back to the delayed signal.

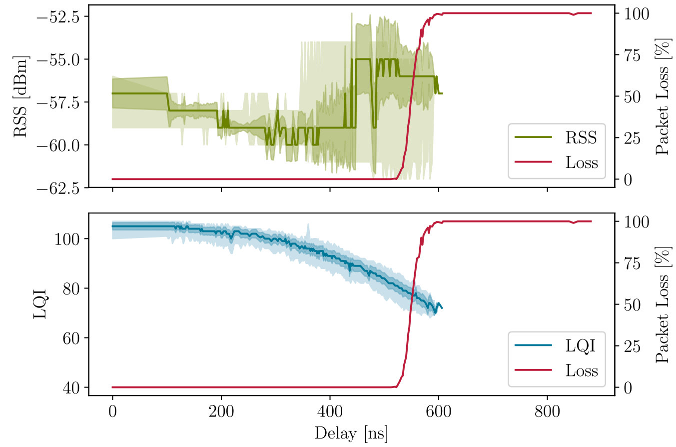
The results for the CC2420, shown in Figure 9b, confirm the assumption that CT works with delays up to 500 ns. But from the LQI measurements it can be seen that the quality of the received signal is decreasing much earlier, in particular at 150 ns.

The scenario of a noise free channel is rather unrealistic, so that we used our emulator to generate and add different noise levels. To get a ground truth we measured how the noise affects the signal from a single transmitter, thus without CT. Figure 10 shows the results for this evaluation, where the colors and corresponding shades have the same meaning as in Figure 9. The noise is without a unit, which is due to the fact that our emulator is not able to measure the incoming signal's energy in real physical units. In this evaluation the amplitude of the signal was 0.5 and the noise is given in the same scale.

The RSS measurements for the AT86RF233 and CC2420 had been expected, indeed, the RSS increases while the noise increases. Also the LQI measurements offer no surprises: An increased noise level leads to a decreased LQI and leads finally to a higher packet loss. In Figure 10b the RSS and LQI graphs end at a noise level of about 0.07 as the packet loss reached 100 % and no more measurements could be obtained. Hence, we stopped the measurements at a noise level of 0.09. One remarkable result from this evaluation is that the AT86RF233 seems to be able to decode a signal with a higher noise than the CC2420 (cf. Figure 10a). This is likely an explanation why

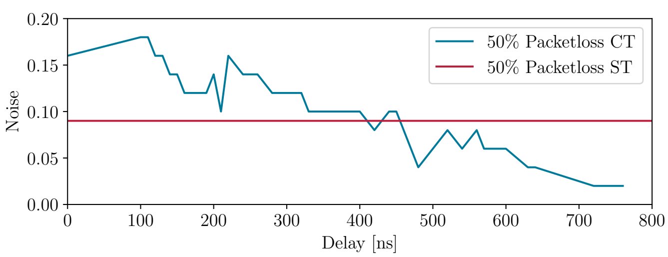


(a) AT86RF233

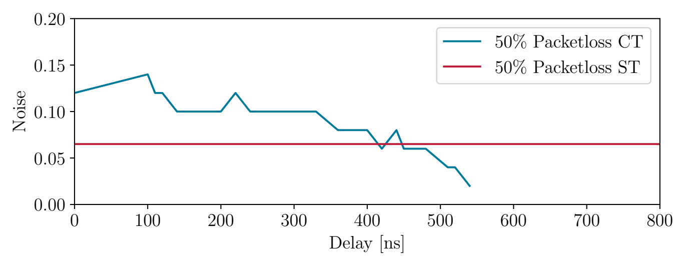


(b) CC2420

Fig. 11: Concurrent Transmission under noise influence. Noise level was chosen to the level where a single transmitter had 99 % packet loss (0.07). Noise is given without unit due to emulator limitations, both transmited signal had an amplitude of 0.5 in the same scale.



(a) AT86RF233



(b) CC2420

Fig. 12: 50 % packet loss threshold over diffrent noise levels and transmission delays. The red line shows the noise level at which a ST has 50 % packet loss. Noise is given without unit due to emulator limitations, both transmited signal had an amplitude of 0.5 in the same scale.

the AT86RF233 is able to decode CT signals with a higher transmission delay than the CC2420.

To evaluate the effectiveness of CI we first determine the noise that we can add to a signal of a single transmitter where the receiver is still able to decode 1 % of the packets (packet loss 99 %). Afterwards we used the lower noise level, of the two transceiver chips, to evaluate CT this noise level. This noise level is further used to disturb a CT signal, but now CI should improve the reception of packets as two transmitters with an amplitude of 0.5 each should add up to 1.0 in best case. To make the evaluation more comparable we used the same noise level for both the AT86RF233 and the CC2420, in particular 0.07.

Figure 11 shows the results of the CT evaluation under noise. For both transceivers the RSS is more fluctuating. While the RSS for the AT86RF233 is monotonically decreasing (cf. Figure 11a), the RSS for the CC2420 decreases between 0 ns and 200 ns but increases again from 500 ns (cf. Figure 11b). However, for both cases the LQI is the more meaningful metric and therefore a better indicator for the quality of a CT signal. Another important observation from this measurement are the thresholds for CI indicated by the increased packet loss compared to the noiseless channel (cf. Figure 9). For the CC2420 this threshold is nearly at the same transmission delay as in Figure 9b. But for the AT86RF233 the packet loss rises about 200 ns earlier (cf. Figure 9a). This observation leads to the assumption that, at least for the AT86RF233, the ability to decode CT signals depends on the noise level on the channel.

To prove this assumption we measured the packet loss over several transmission delay and noise combinations. In Figure 12 we plotted the highest noise level where at least 50 % of the packets were received as a function of the transmission delay. As a comparison we also plotted the noise level at which at least 50 % of packets for a Single Transmission (ST) (without CT) were received (red line). Note that the ST noise level is not depending on a delay, it is a horizontal line as a reference. The threshold value of 50 % packet loss was chosen because we see it as a border where links must be considered unreliable, but every other chosen percentage would show a similar graph, just slightly shifted, squeezed or stretched. As a result it can be seen that for both, the AT86RF233 and CC2420, the ability to handle transmission delay decreases with increasing noise. Figure 12 also shows that a CT signal with more than 400 ns transmission delay is less likely to be decoded correctly than a ST signal, even though the CT signal has twice as much energy. Therefore, to get a real gain from CT the transmission delay should to be far below $500\,\mathrm{ns}$.

VIII. Conclusion

In this paper we showed that the AT86RF233 is capable to achieve the same performance in CT as the CC2420 using the Glossy protocol. We proved this by performing evaluations in a minimalist scenario and also in a larger real world testbed located in an office environment. During these evaluations we revealed that the AT86RF233 has much more and wider variations in its timing accuracy. In Section VI we showed why the timing of the AT86RF233 is not in line with $500\,\mathrm{ns}$ border. To investigate why our implementation for the AT86RF233 shows no performance disadvantage compared to Glossy we used a CT channel emulator. With the aid if this emulator we gained novel insights into the hardware dependency of transceivers when performing CT. As the manageable transmission delay significantly differs between the CC2420 and the AT86RF233, we also investigated the impact of noise to CT. To conclude our findings: Both AT86RF233 and CC2420 are generally capable to perform CT, however, as, e.g. the AT86RF233 is capable to handle larger transmission delays, the CT performance obviously depends on the used hardware and the noise level. To enable CT based communication for heterogeneous WSNs deep knowledge of the transceivers' characteristics is inevitable.

References

[1] IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). *IEEE Std 802.15.4-2011 (Revision of IEEE Std 802.15.4-2006)*, pages 1–314, September 2011.

[2] ublox NEO-M8 Datasheet, Aug. 2016.

[3] Atmel Corporation, San Jose. *Low Power, 2.4GHz Transceiver for ZigBee, RF4CE, IEEE 802.15.4, 6LoWPAN, and ISM Applications*, 8315e–mcu wireless–07/14 edition, July 2014.

[4] M. Brachmann, D. Becker, and S. Santini. Towards enabling concurrent transmissions in heterogeneous networks. In *Proceedings of the 14th International Conference on Information Processing in Sensor Networks*, IPSN '15, pages 354–355, New York, NY, USA, 2015. ACM.

[5] M. Buettner, G. V. Yee, E. Anderson, and R. Han. X-mac: A short preamble mac protocol for duty-cycled wireless sensor networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, SenSys '06, pages 307–320, New York, NY, USA, 2006. ACM.

[6] 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, pages 264–265, USA, 2017. Junction Publishing.

[7] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. Efficient network flooding and time synchronization with glossy. In *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 73–84, April 2011.

[8] S. Ganeriwal, R. Kumar, and M. B. Srivastava. Timing-sync protocol for sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 138–149. ACM, 2003.

[9] O. Gnawali, R. Fonseca, K. Jamieson, M. Kazandjieva, D. Moss, and P. Levis. Ctp: An efficient, robust, and reliable collection tree protocol for wireless sensor networks. *ACM Trans. Sen. Netw.*, 10(1):16:1–16:49, Dec. 2013.

[10] W. R. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, MobiCom '99, pages 174–185, New York, NY, USA, 1999. ACM.

[11] 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, pages 289–290, USA, 2016. Junction Publishing.

[12] B. Kusy, P. Dutta, P. Levis, M. Maroti, A. Ledeczi, and D. Culler. Elapsed time on arrival: a simple and versatile primitive for canonical time synchronisation services. *International Journal of Ad Hoc and Ubiquitous Computing*, 1(4):239–251, 2006.

[13] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation - Volume 1*, NSDI'04, pages 2–2, Berkeley, CA, USA, 2004. USENIX Association.

[14] C. H. Liao, Y. Katsumata, M. Suzuki, and H. Morikawa. Revisiting the so-called constructive interference in concurrent transmission. In *2016 IEEE 41st Conference on Local Computer Networks (LCN)*, pages 280–288, Nov 2016.

[15] R. Lim, R. Da Forno, F. Sutton, and L. Thiele. Competition: Robust flooding using back-to-back synchronous transmissions with channel-hopping. In *Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks*, EWSN '17, pages 270–271, USA, 2017. Junction Publishing.

[16] J. Lu and K. Whitehouse. Flash flooding: Exploiting the capture effect for rapid flooding in wireless sensor networks. In *IEEE INFOCOM 2009*, pages 2491–2499, April 2009.

[17] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi. The flooding time synchronization protocol. In *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*, SenSys '04, pages 39–49, New York, NY, USA, 2004. ACM.

[18] V. S. Rao, M. Koppal, R. V. Prasad, T. V. Prabhakar, C. Sarkar, and I. Niemegeers. Murphy loves ci: Unfolding and improving constructive interference in wsns. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9, April 2016.

[19] P. Sommer and Y.-A. Pignolet. Competition: Dependable network flooding using glossy with channel-hopping. In *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*, EWSN '16, pages 303–303, USA, 2016. Junction Publishing.

[20] M. Suzuki, Y. Yamashita, and H. Morikawa. Low-power, end-to-end reliable collection using glossy for wireless sensor networks. In *2013 IEEE 77th Vehicular Technology Conference (VTC Spring)*, pages 1–5, June 2013.

[21] Texas Instruments. *Chipcon CC2420 Datasheet*, swrs041c edition, 2017.

[22] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu. The broadcast storm problem in a mobile ad hoc network. *Wirel. Netw.*, 8(2/3):153–167, Mar. 2002.

[23] G. von Zengen, A. Willecke, and L. C. Wolf. Demo: Investigating concurrent transmission using software defined radios. In *Proceedings of the 2018 International Conference on Embedded Wireless Systems and Networks*, EWSN '18, Madrid, Spain, Feb. 2018.

[24] Y. Wang, Y. Liu, Y. He, X. Y. Li, and D. Cheng. Disco: Improving packet delivery via deliberate synchronized constructive interference. *IEEE Transactions on Parallel and Distributed Systems*, 26(3):713–723, March 2015.

[25] T. Winter et al. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. RFC 6550 (Proposed Standard), Mar. 2012.

[26] K.-J. Wong and D. K. Arvind. Speckmac: Low-power decentralised mac protocols for low data rate transmissions in specknets. In *Proceedings of the 2Nd International Workshop on Multi-hop Ad Hoc Networks: From Theory to Reality*, REALMAN '06, pages 71–78, New York, NY, USA, 2006. ACM.

[27] D. Yuan and M. Hollick. Let's talk together: Understanding concurrent transmission in wireless sensor networks. In *38th Annual IEEE Conference on Local Computer Networks*, pages 219–227, Oct 2013.

[28] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 1–13. ACM, 2003.