Poster: Stalwart – a Predictable Reliable Adaptive and Low-latency Real-time Wireless Protocol

Romain Jacob†, Licong Zhang‡, Marco Zimmerling*,
Jan Beutel†, Samarjit Chakraborty‡, and Lothar Thiele†

†ETH Zurich, Switzerland
firstname.lastname@tik.ee.ethz.ch

‡TU Munich, Germany
firstname.lastname@rcs.ei.tum.de

*TU Dresden, Germany
firstname.lastname@tu-dresden.de

ABSTRACT

This paper introduces Stalwart, a novel system design for wireless Cyber-Physical Systems (CPS) including a scheduling framework that provides real-time guarantees, minimizes end-to-end latency between application tasks, minimizes communication energy, and ensures safety in terms of conflict-free communication.

CCS CONCEPTS

• **Computer systems organization → Sensors and actuators**; **Real-time system specification**;

KEYWORDS

Wireless, Real-time, CPS, Task- and Network-scheduling

1 INTRODUCTION

Over the past decades, industry has been widely relying on wired field buses, for good reasons. It is a simple architecture, predictable, fast, and fault-tolerant. This ushered the era of Cyber-Physical Systems (CPS), which combines sensing, online computation, and actuation, effectively achieving adaptability in distributed control systems. But there are limits to what wired systems can do! Now envision a wireless system that would provide similar properties than a wired bus – this would bring wireless CPS to the next level. Our work attempts to realize this vision.

Challenges. The concept of a wireless bus has been proposed in the literature, *e.g.*, [4], but thus far never instantiated in a way that satisfies the requirements of a large class of CPS applications: reliability, timing predictability, low end-to-end latency at the application level, energy efficiency, and (limited) runtime adaptability.

To understand the challenge of wireless CPS, one must apprehend the fundamental difference between a field bus and a wireless network. In a field bus, whenever a node is not transmitting, it can idly listen for incoming messages. Upon request from a central host, or controller, each node can *wake up and react quickly*. For a low-power wireless node, the major part of the energy consumed goes to its radio. Therefore, energy efficiency dictates to turn the radio off whenever possible, thus enabling long autonomous execution without an external power source. A node is then *completely unreachable* until it wakes up. Thus, two nodes require overlapping wake-up time intervals to communicate.

This observation often results in wireless system designs that minimize energy consumption by using *rounds*, *i.e.*, time intervals where all nodes wake up, exchange messages, then turn off their radio [4, 7, 9]. A scheduling scheme is required to define when the rounds take place – *i.e.*, when to wake up – and which nodes are allowed to send messages during the round. However, CPS do not only exchange messages but also execute tasks, *e.g.*, sensing or actuation. Oftentimes, the system requirements are specified end-to-end, *i.e.*, between tasks distributed over multiple nodes. Meeting such requirements calls for co-scheduling the execution of tasks and the communication of messages, as done for wired systems [3].

Contributions. Let us consider wireless CPS based on communication rounds, *i.e.*, trying to minimize energy consumption. For such systems, state-of-the-art scheduling methods, timing analysis, and other existing concepts for wired buses *are not directly applicable*, as they rely on the assumption that communication can be scheduled at any point in time, which does not conform with the use of communication rounds in a wireless setting.

The challenge is twofold. On the one hand, the complex joint optimization problem for co-scheduling distributed tasks and communication rounds cannot be solved online in a low-power setting, *i.e.*, with low computing power. On the other hand, CPS often require some runtime adaptability. Therefore, we propose to partition the scheduling problem into an offline and an online phase. First, multiple schedules – one for each operation mode – are synthesized offline based on an ILP formulation, such that timing constraints are satisfied. The corresponding scheduling tables are distributed to the network. Then online, before every round, we just select the current operation mode and the corresponding schedule phase.

This approach brings another benefit. It induces only very little communication overhead for the distribution of the schedule at runtime. Moreover, by sending the schedule phase at every round, we can guarantee the reliability and fault-recovery of the protocol. The schedule phase is sufficient for any node to uniquely determine the schedule for the current round and when to wake up for the next round, even under packet losses.

Together, those ideas form a wireless CPS design and scheduling framework that we call Stalwart. This paper introduces the main underlying concepts of our design and outline our current and future work in this direction.

2 OVERVIEW OF STALWART DESIGN

The general design of Stalwart relies on seven concepts, which we briefly present and motivate in this section. Most of those concepts

SenSys '17, Delft, Netherlands

DOI: 10.1145/3131672.3136969

are individually known and valued. We combine them together for the first time to realize an elegant wireless CPS solution.

(1) We rely on *Glossy as wireless communication primitive* [5]. Glossy is a flooding protocol based on constructive interference, which has been proven to be highly reliable and energy efficient. As flooding is independent of the network state, Glossy simplifies the scheduling problem and supports predictability, as previously leveraged by other protocols, *e.g.*, [10].
(2) We adopt a *contention-free, centrally-controlled, time-slotted design*, *i.e.*, no individual node initiates communication unless it is guaranteed that it does not disturb the rest of the network – a classic design choice for real-time protocols, *e.g.*, [9, 10]. The central node is called the *host*.
(3) To meet tight deadline requirements of CPS, often specified end-to-end between distributed application tasks over multiple nodes (*e.g.*, 10-500 ms delay for a distributed closed-loop control system), we *co-schedule all task executions and message exchanges*, similarly to the state-of-the-art for wired systems [3].
(4) We *statically synthesize the complete schedule* for all task executions and message communication such that real-time constraints are met, end-to-end latency is minimized, and the energy consumed for communication is minimized. Schedules are obtained by solving an ILP formulation. This enables the computation of optimized schedules even for our complex co-scheduling scenario and avoid runtime computation overheads.
(5) The resulting *scheduling tables are distributed*. Each node gets its own schedule – it does not need the other node schedules. This limits the communication overhead to distribute schedules at runtime and has a little memory cost.
(6) Stalwart caters for runtime adaptability by switching between multiple pre-configured operation modes, similar to [6]. The schedule synthesis enforces that *real-time applications have the same schedule* in all modes, which enables predictable and safe switches between modes.
(7) Finally, to minimize the energy consumed by wireless communication, we *group message transmissions into communication in rounds*, *i.e.*, time intervals where all nodes turn their radio on and communicate using Glossy floods. As in LWB [4], each round starts with a communication from the host, followed by a varying number of slots, each allocated to one unique node to start transmitting. In Stalwart, the host sends a *beacon* that contains the operation mode and schedule phase. As each node knows its own schedule, this is sufficient to know *(i)* which slots (if any) are allocated to it for the current round, and *(ii)* when is the next round, *i.e.*, when to wake up next after the current round is over.

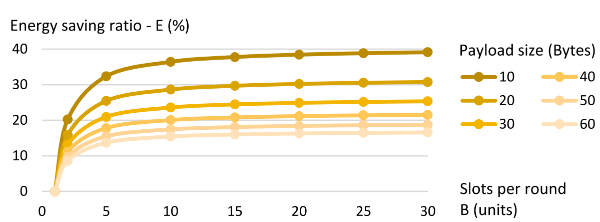

Figure 1: Relative radio-on time benefit of using rounds compared to single messages. *As each round requires only one beacon from the host, the benefit of using rounds grows with the number of number of slots per round (X-axis). Conversely, those savings become less significant as the payload size increases (lighter colors).*

The main technical challenge in the design of Stalwart is to merge together two state-of-the-art techniques: network calculus, which is a natural framework for round-based scheduling, and linear constraint programming methods, like ILP. The first is inherently non-linear while the second only supports linear constraints by default. We combine both to form the underlying scheduling framework of Stalwart.

3 ON-GOING AND FUTURE WORK

To evaluate Stalwart, our wireless CPS design, we consider two metrics as being of primal importance: *(i)* the accumulated radio-on time, commonly used as a mock-up for the energy consumed by a low-power wireless node, and *(ii)* the minimal achievable end-to-end latency, including in comparison to the state of the art [8]. We are developing a quantitative evaluation of the design parameters of Stalwart, *e.g.*, the number of slots per round, based on timing and energy models of a publicly available implementation of Glossy [1]. As an example, Fig. 1 shows the benefit of our design using rounds with respect to radio-on time, compared to a similar scheme that does not group transmissions into rounds.

To validate Stalwart design and demonstrate its practicality, we are currently working on a real-world implementation and testing of Stalwart, which requires to take implementation and platform specific delays into account in the schedule synthesis, similar to [8].

Finally, Stalwart currently supports periodic traffic only. Looking further ahead, it would be interesting to efficiently support sporadic traffic – similarly to recent work on TTEthernet [2] for example – thus enabling event-triggered fault-tolerance.

Acknowledgments. This work is supported by Nano-Tera.ch, with Swiss Confederation financing, and by the German Research Foundation (DFG) within CFAED and SPP 1914, project EcoCPS.

REFERENCES

[1] 2017. Low-Power Wireless Bus (LWB). Online. (2017). Retrieved August 14, 2017 from https://github.com/ETHZ-TEC/LWB
[2] Mohammad Ashjaei, Nima Khalilzad, Saad Mubeen, Moris Behnam, Ingo Sander, Luis Almeida, and Thomas Nolte. 2017. Designing end-to-end resource reservations in predictable distributed embedded systems. *Real-Time Systems* (2017), 1–41.
[3] Silviu S. Craciunas and Ramon Serna Oliver. 2016. Combined task- and network-level scheduling for distributed time-triggered systems. *Real-Time Systems* (2016).
[4] Federico Ferrari, Marco Zimmerling, Luca Mottola, and Lothar Thiele. 2012. Low-power Wireless Bus. In *Proc. of ACM SenSys*.
[5] Federico Ferrari, Marco Zimmerling, Lothar Thiele, and Olga Saukh. 2011. Efficient network flooding and time synchronization with Glossy. In *Information Processing in Sensor Networks (IPSN)*. 73–84.
[6] Gerhard Fohler. 1993. Changing operational modes in the context of pre run-time scheduling. *IEICE transactions on information and systems* 76, 11 (1993), 1333–1340.
[7] IEEE 802 Working Group and others. 2011. IEEE Standard for Local and Metropolitan Area NetworksŮPart 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). *IEEE Std* 802 (2011), 4–2011.
[8] Romain Jacob, Marco Zimmerling, Pengcheng Huang, Jan Beutel, and Lothar Thiele. 2016. End-to-end Real-time Guarantees in Wireless Cyber-physical Systems. In *37th IEEE Real-Time Systems Symposium (RTSS)*.
[9] WirelessHART. 2007. WirelessHART. (2007). Retrieved August 14, 2017 from https://fieldcommgroup.org/technologies/hart/
[10] Marco Zimmerling, Luca Mottola, Pratyush Kumar, Federico Ferrari, and Lothar Thiele. 2017. Adaptive Real-time Communication for Wireless Cyber-physical Systems. *ACM Transactions on Cyber-Physical Systems* (2017).