


The Time-Triggered Wireless Architecture (Artifact)

Romain Jacob 

ETH Zurich, Switzerland

Licong Zhang

TU Munich, Germany

Marco Zimmerling 


TU Dresden, Germany

Jan Beutel 

ETH Zurich, Switzerland

Samarjit Chakraborty 

University of North Carolina at Chapel Hill, NC, United States

Lothar Thiele 

ETH Zurich, Switzerland

Abstract

This artifact contains a stable version of all the data and source code required to reproduce or replicate the results presented in *The Time-Triggered Wireless Architecture*.

One GitHub repository serves as main hub for all information related to the artifact. The

README file contains detailed instructions for

- Running the TTnet model
- Compiling and running TTnet
- Running the TTW scheduler
- Reproducing the data processing
- Reproducing the plots

2012 ACM Subject Classification Computer systems organization → Real-time system architecture; Computer systems organization → Sensors and actuators; Networks → Sensor networks

Keywords and phrases Time-triggered architecture, wireless bus, synchronous transmissions

Digital Object Identifier 10.4230/DARTS.6.1.5

Funding *Marco Zimmerling*: German Research Foundation (DFG) within the Emmy Noether project NextIoT (grant ZI 1635/2-1)

Lothar Thiele: Swiss National Science Foundation program “NCCR Automation”

Acknowledgements We thank Andreas Biri for his feedback on earlier versions of the artifact.

Related Article Romain Jacob, Licong Zhang, Marco Zimmerling, Jan Beutel, Samarjit Chakraborty, and Lothar Thiele, “The Time-Triggered Wireless Architecture”, in 32nd Euromicro Conference on Real-Time Systems (ECRTS 2020), LIPICs, Vol. 165, pp. 19:1–19:25, 2020.

<https://doi.org/10.4230/LIPICs.ECRTS.2020.19>

Related Conference 32nd Euromicro Conference on Real-Time Systems (ECRTS 2020), July 7–10, 2020, Virtual Conference

1 Scope

The artifact described in this document allows to replicate all computational experiments and data analysis of the related article. This includes the derivation of TTW schedules, the data analysis of the TTnet model validation, and all the plots presented in the article.



© Romain Jacob, Licong Zhang, Marco Zimmerling, Jan Beutel, Samarjit Chakraborty, and Lothar Thiele; licensed under Creative Commons Attribution 3.0 Germany (CC BY 3.0 DE)

Dagstuhl Artifacts Series, Vol. 6, Issue 1, Artifact No. 5, pp. 5:1–5:3



DAGSTUHL ARTIFACTS SERIES
Dagstuhl Artifacts Series
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

5:2 The Time-Triggered Wireless Architecture (Artifact)

The article also contains networking experiments, which were run on the FlockLab tested¹ over the course of several months. As these experiments are difficult to replicate, the artifact contains the raw data we have collected, along with the FlockLab configuration files used to obtain them to allow an eventual reproduction.

Some parts of the artifact are hosted on dedicated GitHub repositories (with corresponding Zenodo archives). Relevant links are available in the “Getting the artifact” section.

2 Content

The artifact package includes:

TTnet

An implementation of TTnet, the wireless network stack, presented in Section 3.2.

We provide necessary instructions to compile the source code (C code). The compiled firmware can be run remotely using the FlockLab testbed (instructions in the README file).

TTnet model

The time and energy model of TTnet, presented in Section 3.3.

We provide an implementation of the model (Python code). You can run this code *without any installation required* directly in your web browser, using Binder.² If you want to run the code locally, the repository contains a *requirements.txt* file that lists all dependencies.

TTW scheduler

The TTW scheduler, presented in Section 4.

We provide an implementation of the scheduler (Matlab code). It relies on the Gurobi software to solve the MILP formulation which underlines the scheduling problem. As such, it is the hardest part of our artifact to successfully re-run (both Matlab and Gurobi being commercial software).

Data processing

All raw data and processing scripts for the TTnet model validation, presented in Section 5.1.

The scripts are written in Python. As for the TTnet model, you can run this code in your web browser using Binder, or locally based on the *requirements.txt* file.

Plotting scripts

The Python scripts used to generate the plots presented in the article.

As for the **TTnet model**, you can run this code in your web browser using Binder, or locally based on the *requirements.txt* file.

3 Getting the artifact

The artifact endorsed by the Artifact Evaluation Committee is available free of charge on the Dagstuhl Research Online Publication Server (DROPS).

The artifact is also available within several GitHub repositories, with corresponding releases archived on Zenodo. Refer to these for the latest version of the artifact.

- The main repository serves as a hub for all information related to TTW’s artifact.

GitHub <https://github.com/romain-jacob/TTW-Artifacts>

Zenodo <https://doi.org/10.5281/zenodo.3759221>

¹ <http://flocklab.ethz.ch/>

² <https://mybinder.org/>

- The source code of the TTnet network stack is available as part of the Baloo framework.
GitHub <https://github.com/ETHZ-TEC/Baloo>
Zenodo <https://doi.org/10.5281/zenodo.3510171>
- The code of the TTW Scheduler is available in a dedicated repository.
GitHub <https://github.com/romain-jacob/TTW-Scheduler>
Zenodo <https://doi.org/10.5281/zenodo.3530665>

4 Tested platforms

The computational artifact can be run on any personal computer compatible with Matlab, Gurobi, and Python 3. The networking experiments require specialized hardware which is accessible on public testbeds (see README for details).

5 License

The artifact is available under the GPL-3.0 license.

6 MD5 sum of the artifact

6bd1f42db74da4fe0ebfb8f977fa2eae

7 Size of the artifact

7.6 MB