TTTech

State-of-the-Art Fault-Tolerant Clock Synchronization for Ultra-high Reliable Systems

Wilfried Steiner

Agenda

- Byzantine Fault-Tolerance
- SAE AS6802 Byzantine Fault-Tolerant (Clock) Synchronization
- Protocol Verification for Ultrahigh Reliable Systems
- Conclusion

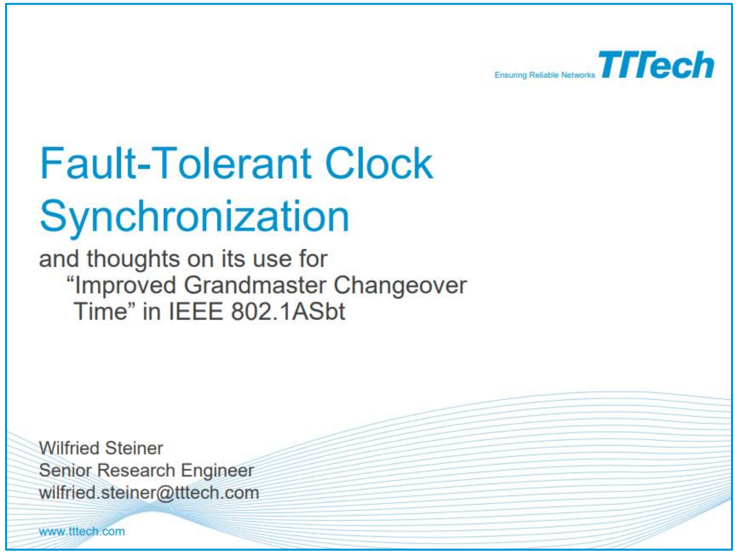
Agenda

- Byzantine Fault-Tolerance
- SAE AS6802 Byzantine Fault-Tolerant (Clock) Synchronization
- Protocol Verification for Ultrahigh Reliable Systems
- Conclusion

Byzantine Fault-Tolerance

(3k+1) clocks are necessary to tolerate the Byzantine Failure of k clocks (e.g., k=2 → 7 clocks)

- Ultrahigh reliable systems require a system failure rate to be in the order of 10^{-9} failures/h or lower.
- Only a distributed fault-tolerant computer system with well-defined fault-containment units (FCUs) achieves the ultrahigh reliability requirement.
- FCUs may fail in a Byzantine failure mode and the distributed computer system must be designed to mitigate this failure mode.
- In order to tolerate *k* Byzantine failures *n=3k+1* nodes are needed.
- The concept to Byzantine fault tolerance has been introduced in: Lamport, L., Shostak, R., Pease, M. (1982). *The Byzantine Generals Problem*. Comm. ACM TOPLAS. Vol. 4 (3). (pp.382-401).



https://www.ieee802.org/1/files/public/docs2012/new-avb-wsteiner-fault-tolerant-clock-synchronization-0112-v01.pdf

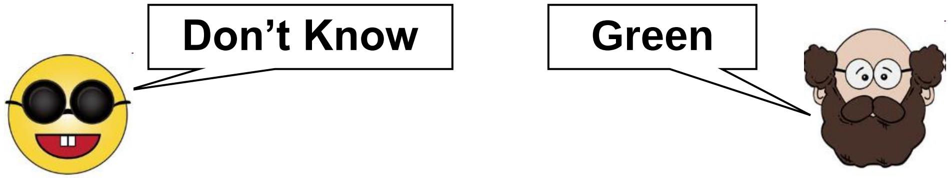
Byzantine Fault-Tolerance (cont.)

Situation:

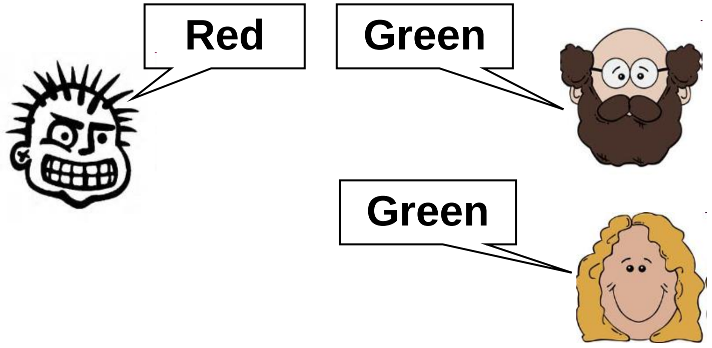
What is the color of the house?

No Failure

Fail-Silence Failure



Fail-Consistent Failure



Byzantine Fault-Tolerance (cont.)

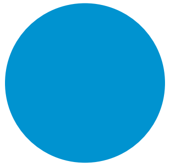
<u>Situation:</u>

What is the color of the house?

Static Situation – one Truth

<u>Situation:</u>

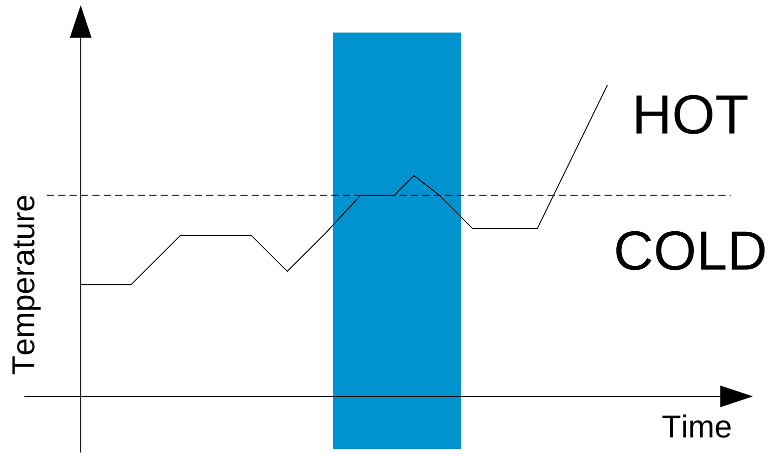
What is the color of the ball ?

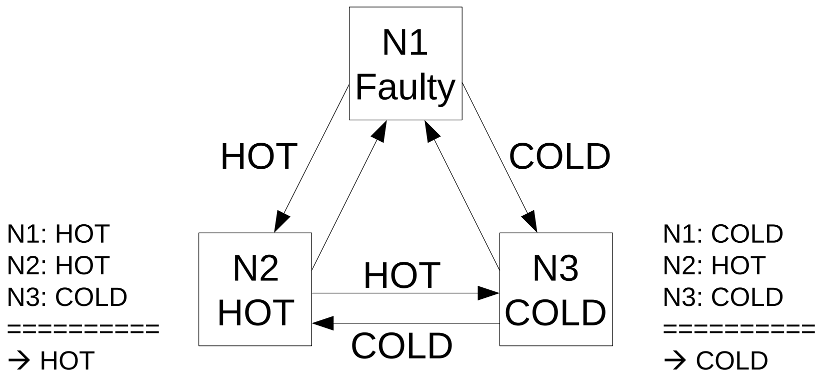


Dynamic Situation – >one Truth

Byzantine Fault-Tolerance (cont.)

A distributed system that measures the temperature of a vessel shall raise an alarm when the temperature exceeds a certain threshold.
The system shall tolerate the arbitrary failure of one node.
How many nodes are required?
How many messages are required?

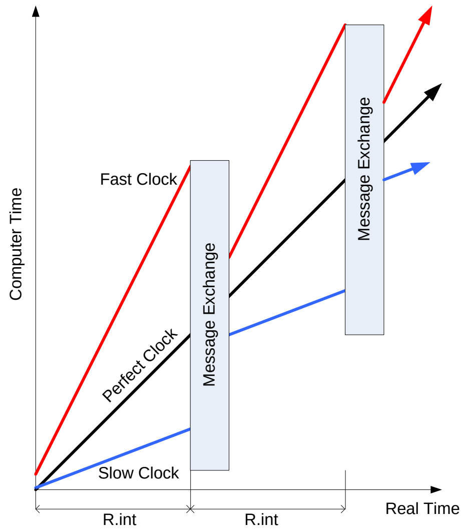


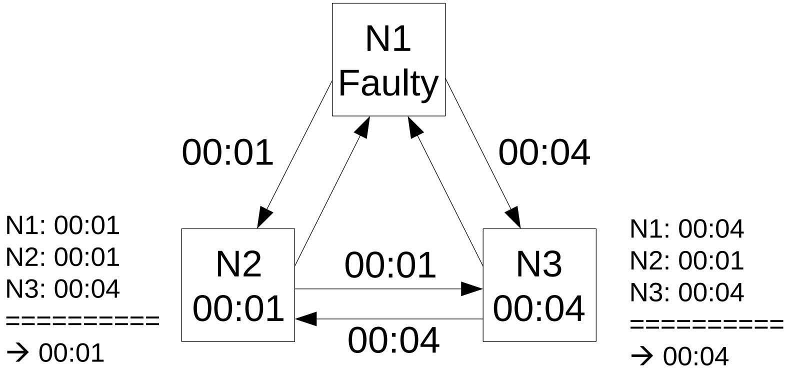


In general, three nodes are insufficient to tolerate the arbitrary failure of a single node.
The two correct nodes are not always able to agree on a value.
A decent body of scientific literature exists that address this problem of dependable systems, in particular dependable communication.

Byzantine Fault-Tolerance (cont.)

A distributed system in which all nodes are equipped with local clocks, all clocks shall become and remain synchronized.
The system shall tolerate the arbitrary failure of one node.
How many nodes are required?
How many messages are required?





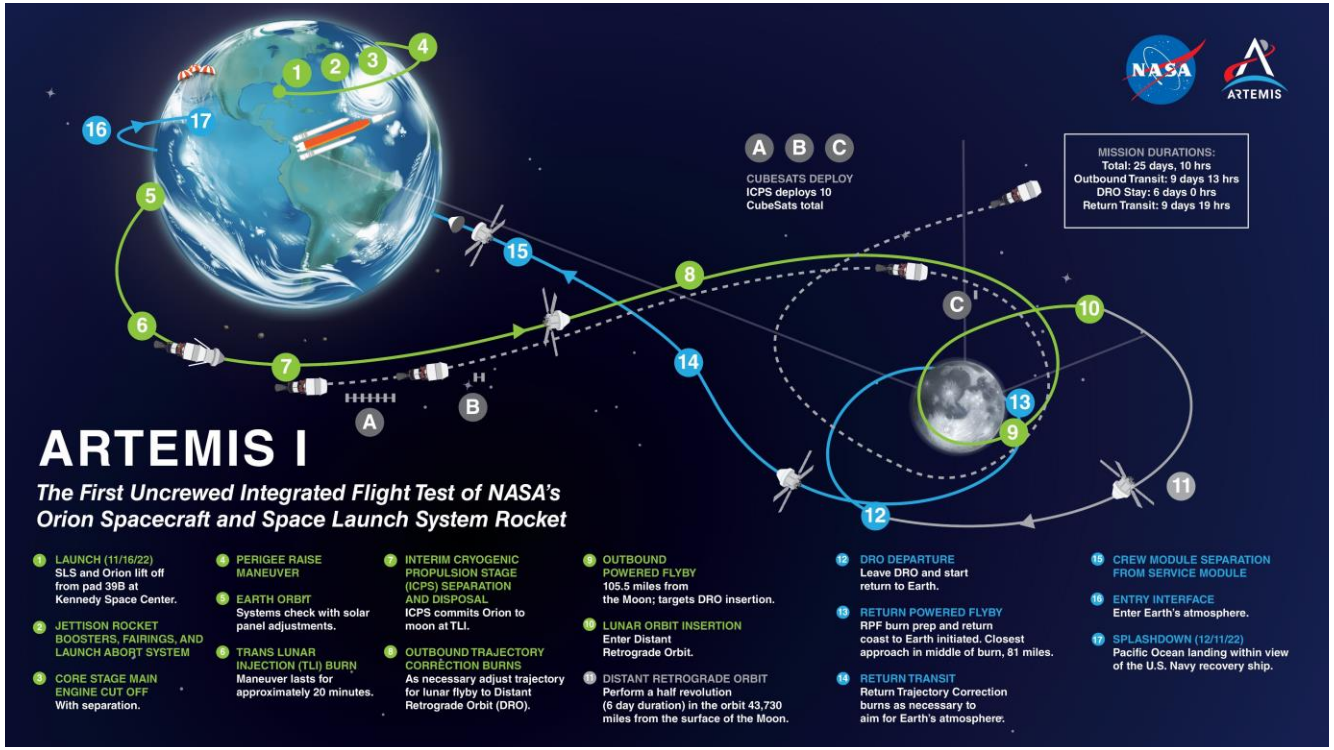
<u>In general, three nodes are insufficient</u> to tolerate the arbitrary failure of a single node.
The two correct nodes are not always able <u>to bring their clocks into close agreement</u>.
A decent body of scientific literature exists that address this problem of fault-tolerant clock synchronization.

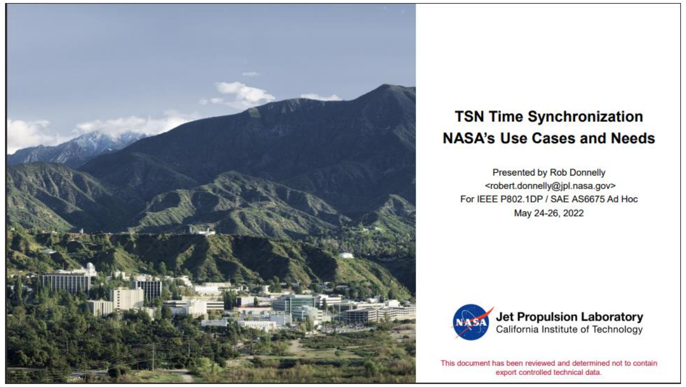
Agenda

- Byzantine Fault-Tolerance
- SAE AS6802 Byzantine Fault-Tolerant (Clock) Synchronization
- Protocol Verification for Ultrahigh Reliable Systems
- Conclusion

SAE AS6802*): Current State-of-the-Art in Fault-Tolerant (Clock) Synchronization Protocols for Avionics

First mission in 2014, last mission in 2022, next mission in 2024 (first crewed flight)



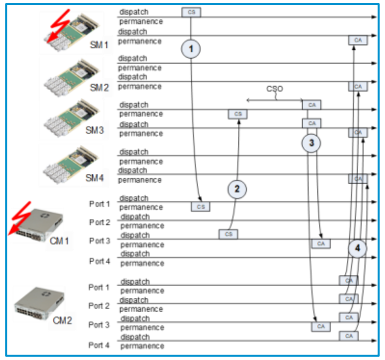


https://www.ieee802.org/1/files/public/docs2022/dp-donnelly-NASA-needs-0522-v00.pdf

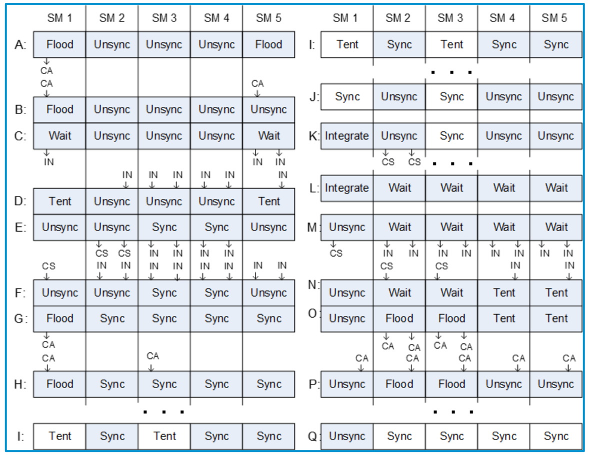
*) Standard-relevant patent families: EP2297885, EP2297886

SAE AS6802 standardizes

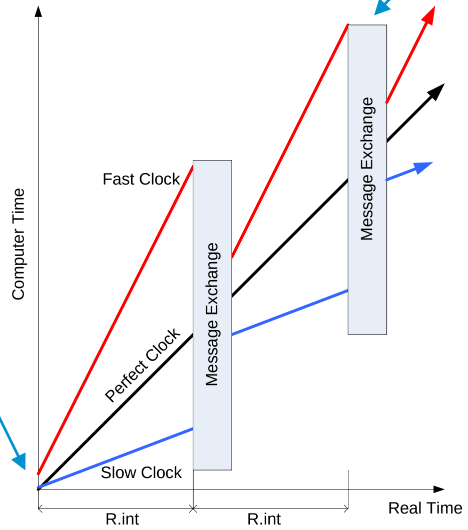
Fault-Tolerant Startup and Restart Protocol



Synchronization Masters (SM) and
Compression Masters (CM) interaction

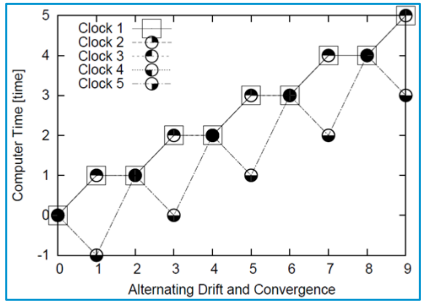


Startup w/ faulty Switch and Faulty End Station

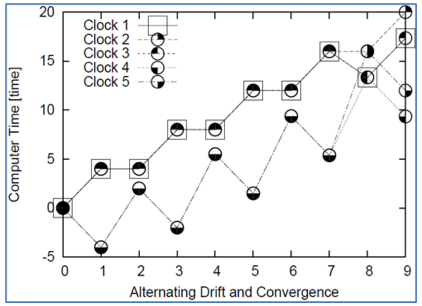


Fault-Tolerant Clock Synchronization Protocol

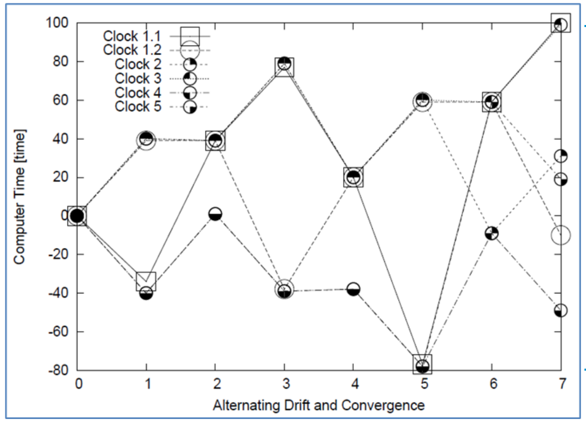
Clock



Fault-Free



Faulty Switch

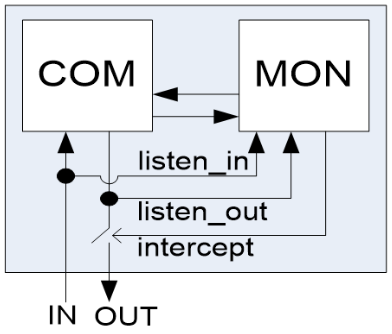


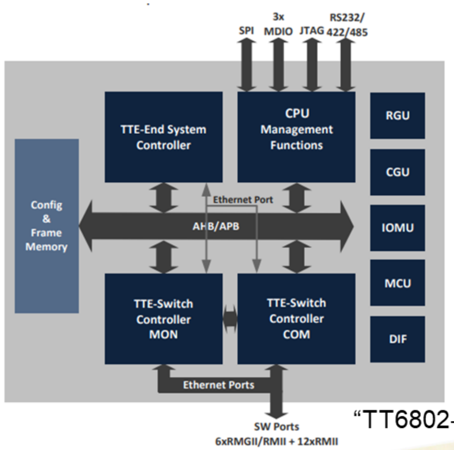
Maximum Deviation is provably bounded.

Faulty Switch and Faulty End Station

SAE AS6802 does not standardize

how to design devices to a sufficient quality level, e.g., Commander/Monitor (COM/MON) Structures





"TT6802-1-SW"

<u>Core COM/MON Assumptions:</u>

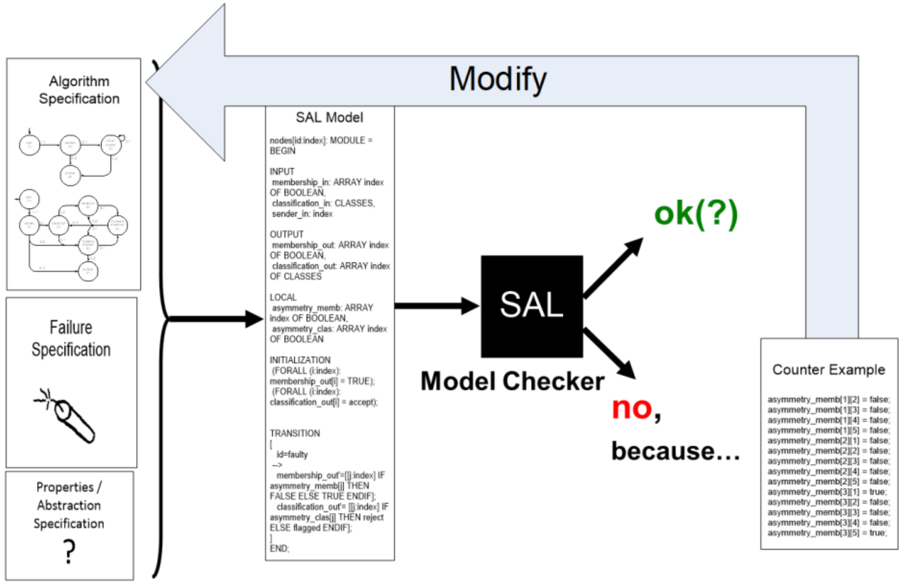
- COM and MON fail independently
- MON can intercept a faulty message produced by the COM
- COM cannot produce a valid message such that this message appears as two different messages on listen_out and OUT; though it may be valid on listen_out but detectable faulty on OUT or vice versa
- MON cannot itself generate a faulty message, neither by inverting listen_out to an output, nor by toggling the intercept signal

Agenda

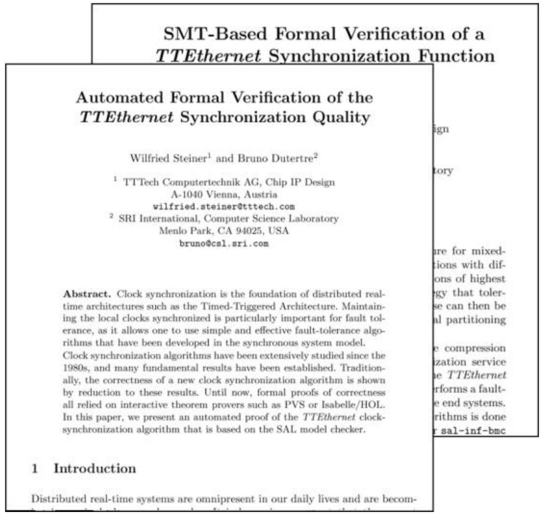
- Byzantine Fault-Tolerance
- SAE AS6802 Byzantine Fault-Tolerant (Clock) Synchronization
- Protocol Verification for Ultrahigh Reliable Systems
- Conclusion

Formal Verification of SAE AS6802 Protocols

Formal Verification (model-checking or theorem proving) is state-of-the-art for ultrahigh reliable systems.



Formal Verification w/ Model Checking



SMT-Based Formal Verification of a *TTEthernet* Synchronization Function

Automated Formal Verification of the *TTEthernet* Synchronization Quality

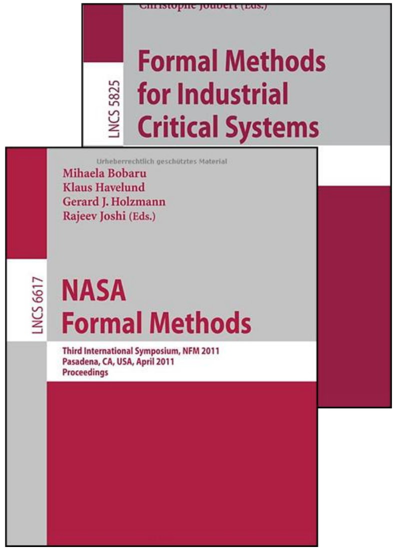
Wilfried Steiner[1] and Bruno Dutertre[2]

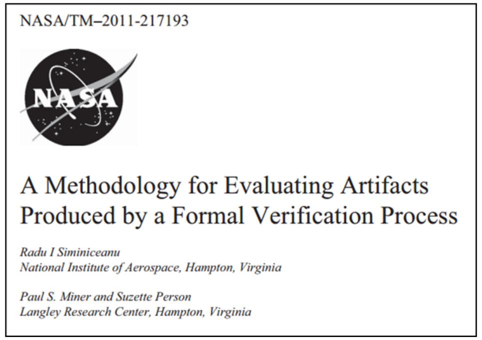
[1] TTTech Computertechnik AG, Chip IP Design
A-1040 Vienna, Austria
wilfried.steiner@tttech.com
[2] SRI International, Computer Science Laboratory
Menlo Park, CA 94025, USA
bruno@csl.sri.com

Abstract. Clock synchronization is the foundation of distributed real-time architectures such as the Timed-Triggered Architecture. Maintaining the local clocks synchronized is particularly important for fault tolerance, as it allows one to use simple and effective fault-tolerance algorithms that have been developed in the synchronous system model. Clock synchronization algorithms have been extensively studied since the 1980s, and many fundamental results have been established. Traditionally, the correctness of a new clock synchronization algorithm is shown by reduction to these results. Until now, formal proofs of correctness all relied on interactive theorem provers such as PVS or Isabelle/HOL. In this paper, we present an automated proof of the *TTEthernet* clock-synchronization algorithm that is based on the SAL model checker.

1 Introduction

Distributed real-time systems are omnipresent in our daily lives and are becom-





NASA/TM–2011-217193

A Methodology for Evaluating Artifacts Produced by a Formal Verification Process

Radu I Siminiceanu
National Institute of Aerospace, Hampton, Virginia

Paul S. Miner and Suzette Person
Langley Research Center, Hampton, Virginia

References on SAE AS6802 Formal Verification

Agenda

- Byzantine Fault-Tolerance
- SAE AS6802 Byzantine Fault-Tolerant (Clock) Synchronization
- Protocol Verification for Ultrahigh Reliable Systems
- Conclusion

Conclusions

Let’s not re-invent the wheel but build on proven solutions.

- Byzantine fault tolerance is state-of-the-art in ultra-high reliable systems.
- SAE AS6802 (Time-Triggered Ethernet) tolerates Byzantine faults and is the state-of-the-art in fault-tolerant (clock) synchronization for ultra-high reliable systems such as avionics.
- It is state-of-the-art to use formal methods like model checking and theorem proving to verify protocol correctness. All SAE AS6802 protocols have been formally verified.
- SAE AS6802 has been standardized in 2011 and concluded a revision in 2023. It has been proven in use for ultra-high reliable systems.

- Proposal 1: work towards using SAE AS6802 synchronization protocols as default FTM.
  - Maybe prepare an SAE AS6802 „Appendix D“ for „Time-Triggered Ethernet Realization on IEEE802.1 (TSN)“ (today Appendix B is for IEEE 802.3 and Appendix C is for ARINC 664-p7).
- Proposal 2: let’s define a „slimmed-down“ version of gPTP to simplify a chip-only implementation.
  - E.g., make peer-delay-measurements and clock rate measurements optional.

TTTech