IFAC Workshop on Real-Time Programming (WRTP'99)

Experimental Evaluation of High-Accuracy Time Distribution in a COTS-based Ethernet LAN

Ulrich Schmid

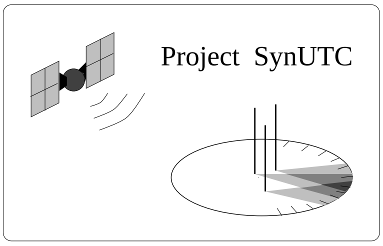
Technische Universität Wien
Department of Automation
Treitlstraße 1, A-1040 Vienna
s@auto.tuwien.ac.at

Herbert Nachtnebel

Technische Universität Wien
Department of General Electrical Engineering and Electronics
Gußhausstraße 25–29, A-1040 Vienna
nachtneb@iaee.tuwien.ac.at

May 26, 1999

About our SynUTC-Project



http://www.auto.tuwien.ac.at/Projects/SynUTC

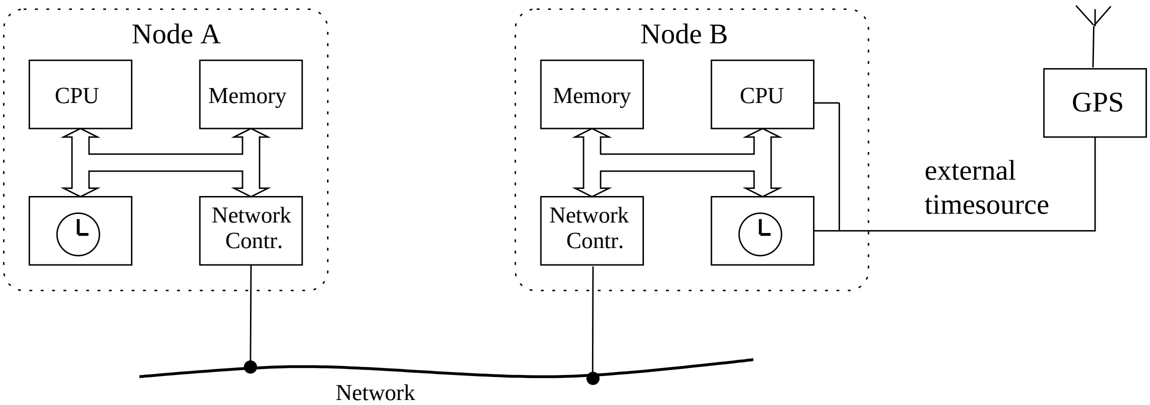
Starting from a WRTP'94 paper [Schmid] ...

our co-workers and a lot of **supporters'** money ...

Gerhard Cadek	FWF-project P10244-ÖMA
Martin Horauer	BMWV Zl.601.577/2-IV/B/9/96
Nikolaus Kerö	OeNB "Jubiläumsfonds" 6454
Johann Klasek	START-Programme Y41-MAT
Dietmar Loy	Austrian GMe
Thomas Mandl	ESPRIT Europractice
Gerda Richter	
Michael Schmidt	
Klaus Schossmaier	

... helped us to pop-up at the WRTP'99 again!

Distributed Systems Architecture



Major components of a node:

- CPU
- Network controller
- Shared memory
- Local clock device (UTCSU-ASIC)

Global issues:

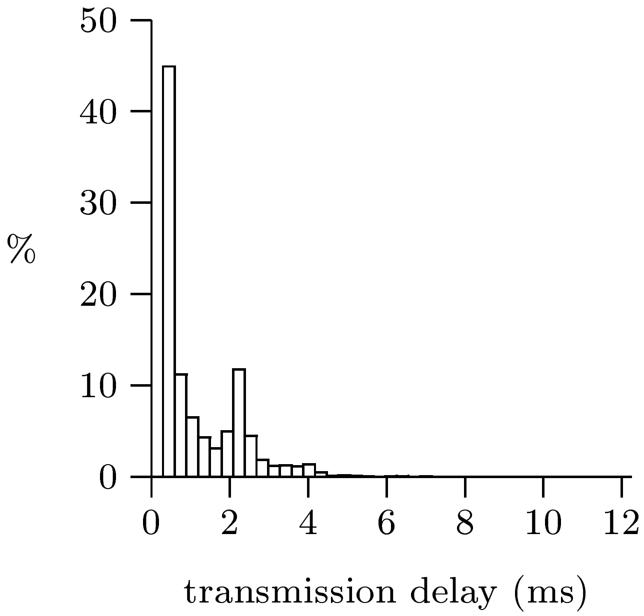
- Nodes solely interconnected via data network
- External timesources (GPS-receivers) at certain nodes

Time Distribution

Problem:

- How to distribute a node's time with high accuracy?
- How to do this via data packets?

Histogram of transmission delays δ:



Minimum	0.30 ms
99%-minimum	0.30 ms
95%-minimum	0.30 ms
95%-maximum	4.17 ms
99%-maximum	9.24 ms
Maximum	37.00 ms
Average	1.35 ms
Std.Dev.	1.63 ms

High-accuracy time distribution impossible due to

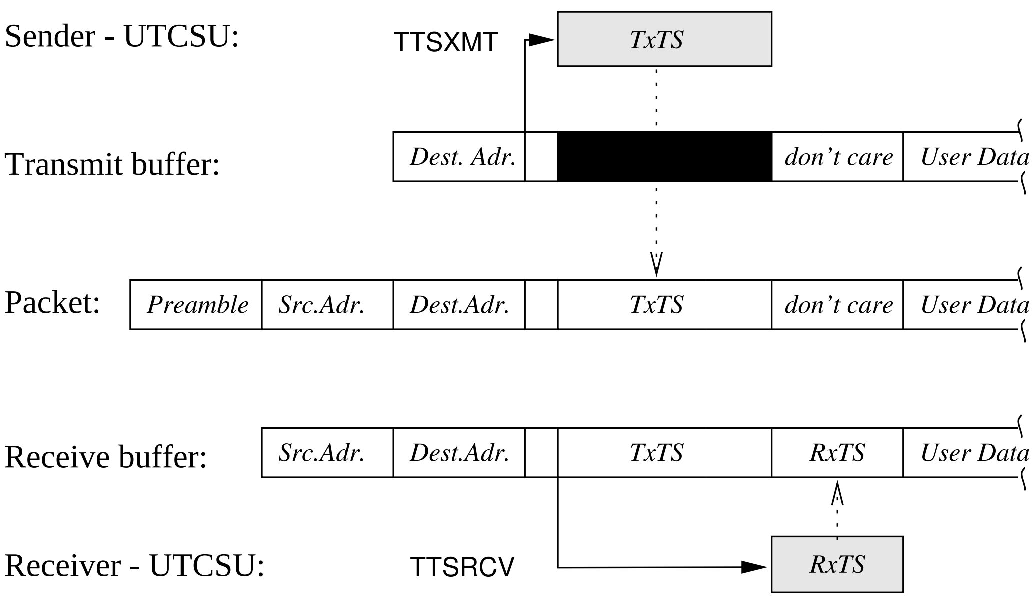
- Max. transmission delay uncertainty $\varepsilon = \delta_{\max} - \delta_{\min}$ in the 10 ms-range
- ε_{95} in the ms-range
- strong dependence upon node and network load

Hardware-Assisted Packet Timestamping

Reduce max. transmission delay uncertainty ε by hardware-timestamping.

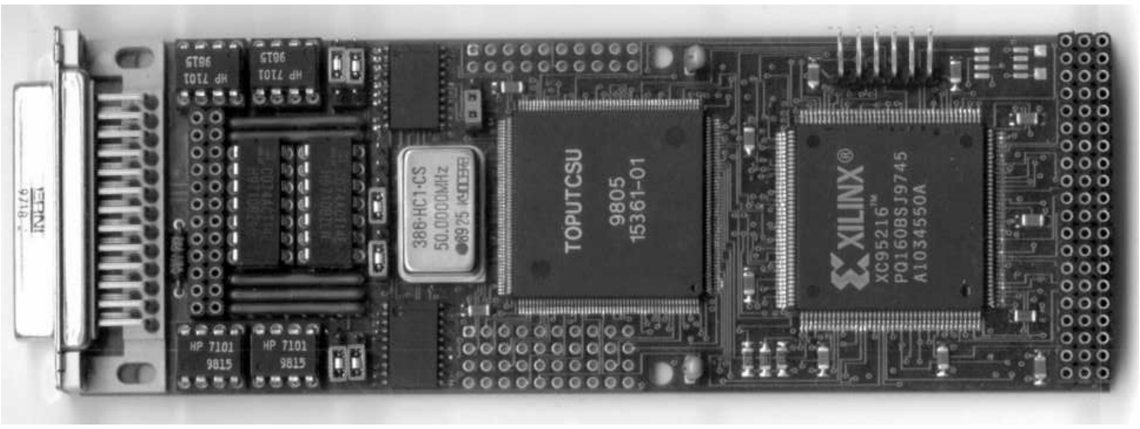
Memory-based timestamping á la [Kopetz & Ochsenreiter, 1987]:

- Reading/Writing a certain byte in the packet's transmit/receive buffer triggers timestamping
- Transmit timestamp is transparently mapped into the transmit buffer



[Ideal: Network controller triggers timestamping]

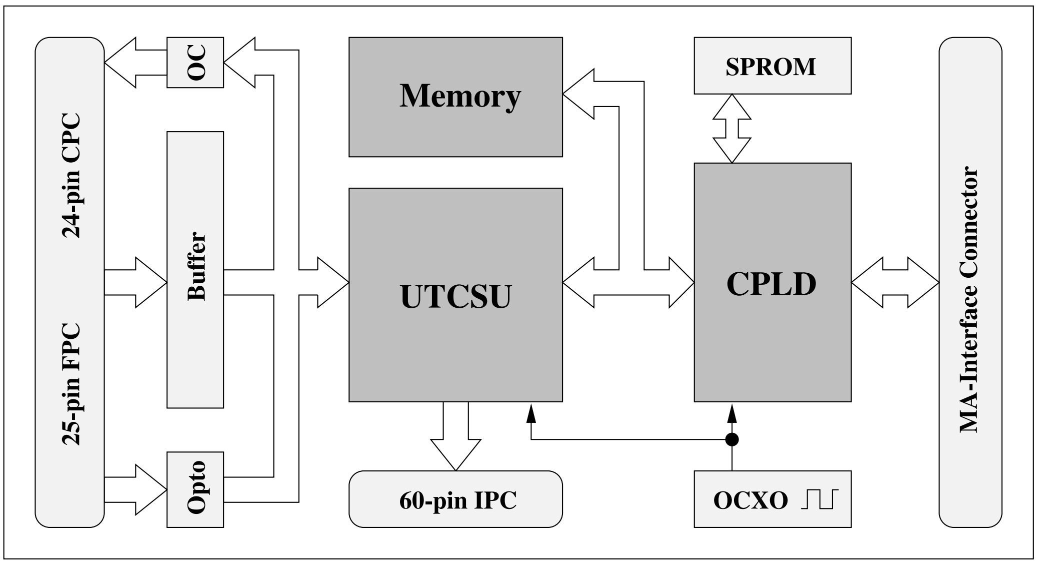
Network Time Interface M-Module (NTI)



Extends COTS-based hardware architectures by providing

- memory-based packet timestamping
- elaborate local clock device

Functional blocks on-board the NTI:

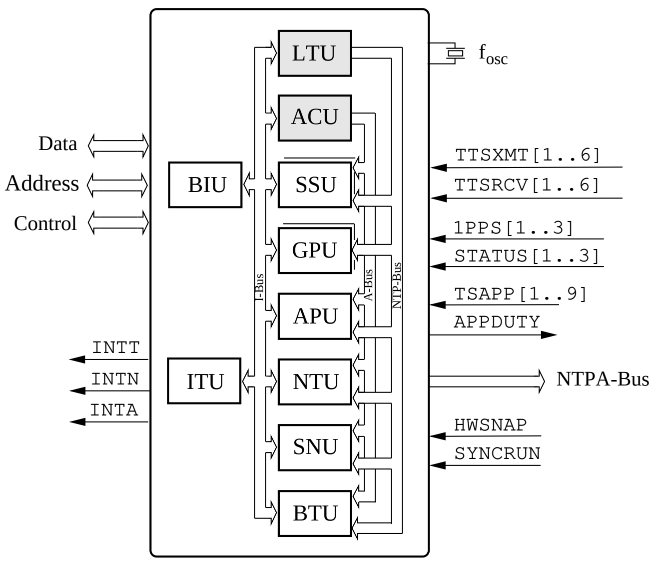


Features of the UTCSU-ASIC (1)

Chip data:

- Complexity 80,000 gates
- 100 mm^2 die size (0.7 μm CMOS process)
- 208-pin high-density PQFP package
- Produced by ES2 (ESPRIT Europractice)
- Manufacturing costs US$ 50,000,–

Block diagram of the UTCSU:



unit	full name
ACU	Accuracy Unit
APU	Application Unit
BIU	Bus Interface Unit
BTU	Built-In Test Unit
GPU	GPS Unit
ITU	Interrupt Unit
LTU	Local Time Unit
NTU	Network Time Interface Unit
SNU	Snapshot Unit
SSU	Synchronization Subnet Unit

Features of the UTCSU-ASIC (2)

Flexible bus interface

- 32, 16 or 8 bit bus width
- Dynamic bus-sizing
- Little/big endian support

Adder-based local clock (LTU)

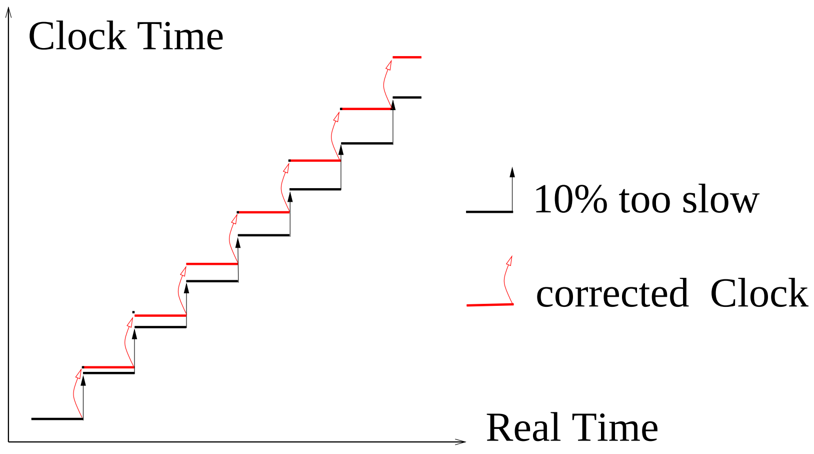
- 56-bit NTP-time (60 ns resolution) + 8-bit checksum
- arbitrary oscillator frequency 1 . . . 25 MHz
- fine-grained clock rate adjustment (10 ns/s)
- continuous amortization and leap second insertion

Local accuracy intervals (ACU)

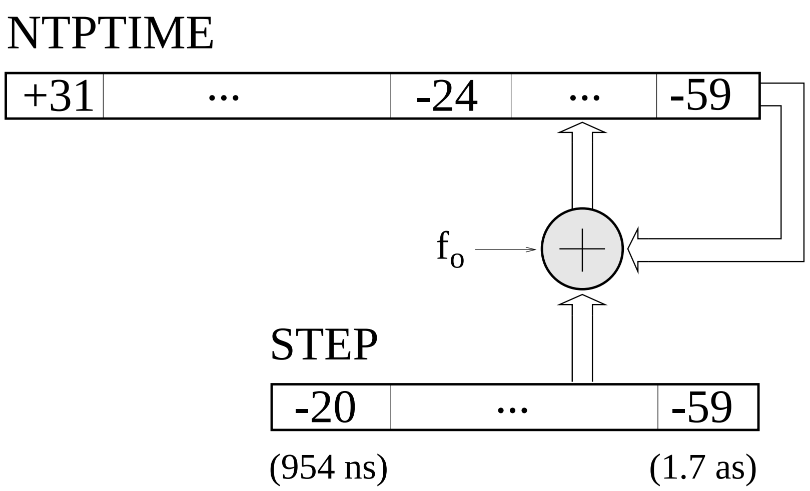
- Maintain local bound upon instantaneous accuracy
- Facilitate interval-based clock synchronization

Excursion: Adder-Based Clocks

Principle of an adder-based clock:



Adder-based clock design:



Excursion: Interval-Based Paradigm

Replace node p's ordinary clock $T = C_p(t)$ by an interval clock

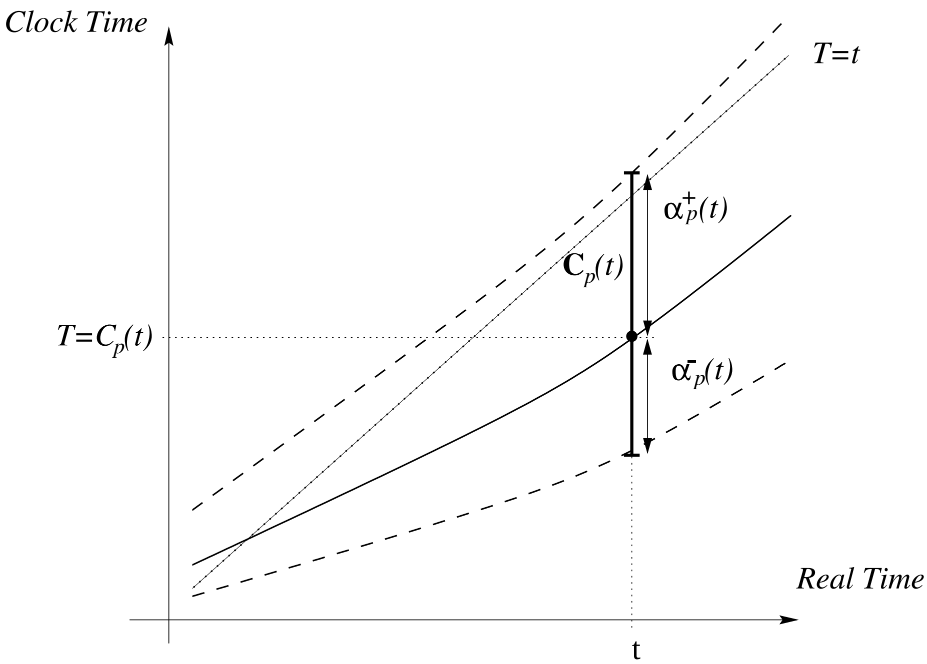
$$\boldsymbol{C}_p(t) = [C_p(t) - \alpha_p^-(t), C_p(t) + \alpha_p^+(t)]$$

incorporating

- a usual local clock $C_p(t)$
- an interval of accuracies $\boldsymbol{\alpha}_p(t) = [-\alpha_p^-(t), \alpha_p^+(t)]$

and enforce $t \in \boldsymbol{C}_p(t)$ at any non-faulty node p.

Example building block: drift compensation



Features of the UTCSU-ASIC (3)

Atomic time+accuracy-stamping for

- packet timestamping (6 SSUs)
- 1pps timing output of a GPS-receiver (3 GPUs)
- application-specific purposes (APU)

Duty-timers for

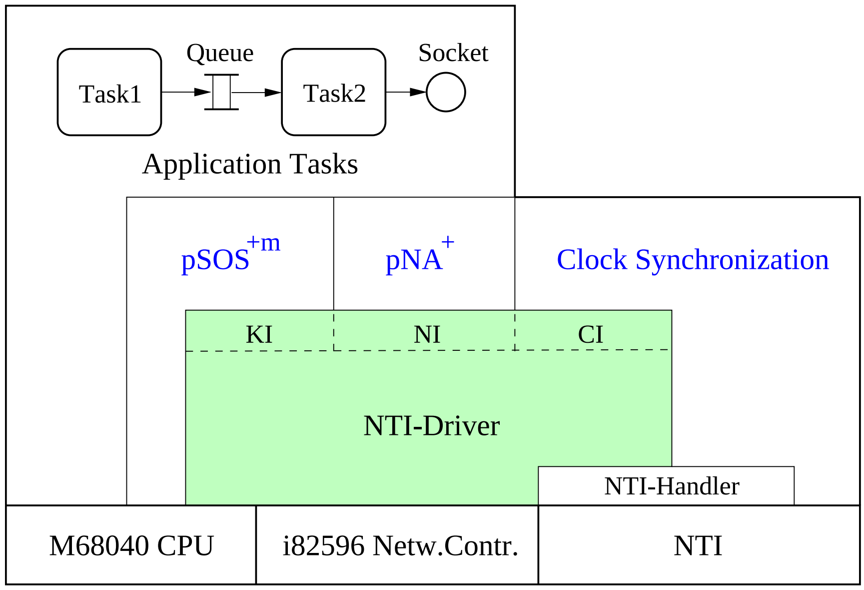
- triggering execution of clock synchronization algorithm
- starting continuous amortization
- inserting/deleting leap seconds
- generating application-specific events

Various (on-line) test and debugging features:

- Scan-test
- Signature and blocksum generation
- Snapshot features
- Synchronous (re-)start of redundant UTCSUs

i82596 NTI Device-Driver for pSOS

Transparent integration into ISI's multitasking/multiprocessing RTOS pSOS^{+m} on the MVME-162 CPU:



NTI-Handler provides

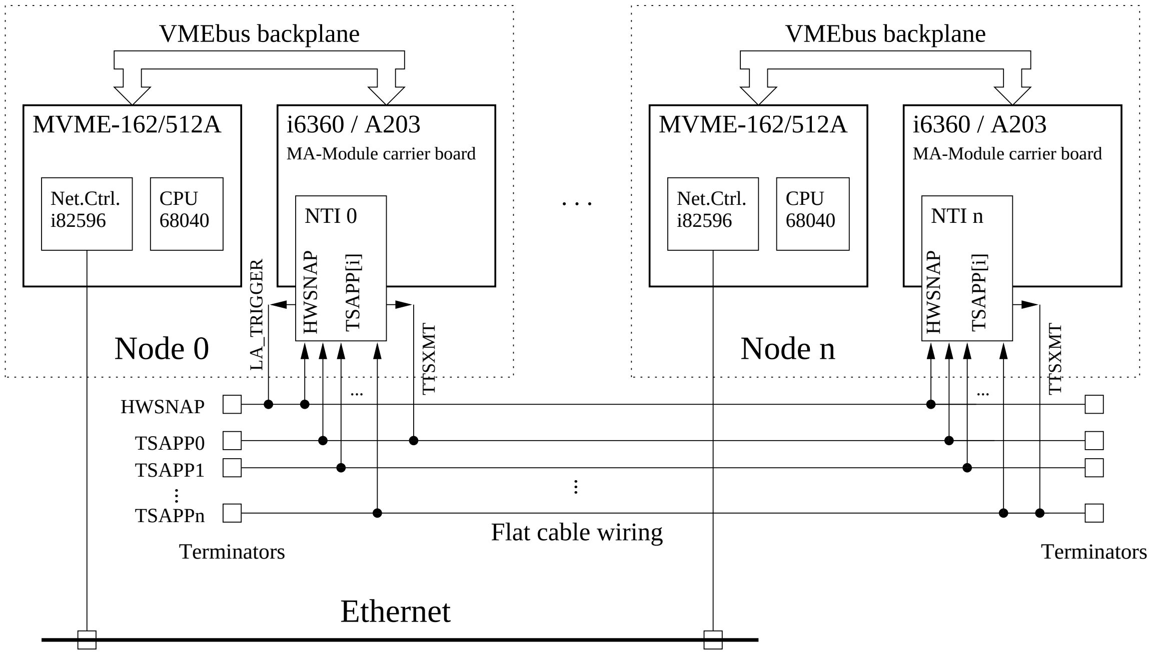
- configuration of NTI/UTCSU registers & memory
- low-level interrrupt handling

i82596 NTI-Driver provides

- Kernel Interface (KI) to pSOS^{+m} kernel
- Network Interface (NI) to TCP/IP protocol stack pNA^{+}
- Clock Interface (CI) to clock synch. algorithm

Evaluation System for Transmission Delay Measurements

Hardware architecture:

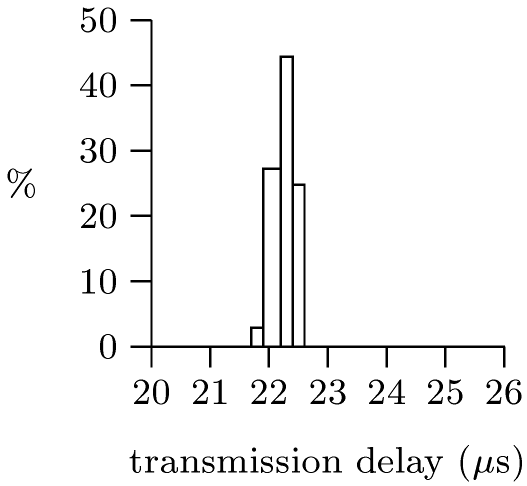


Evaluation system software implemented as a pSOS^{+m} application:

- Made easy by the powerful pSOS^{+m} features
- System integration of NTI and COTS hardware VERY difficult

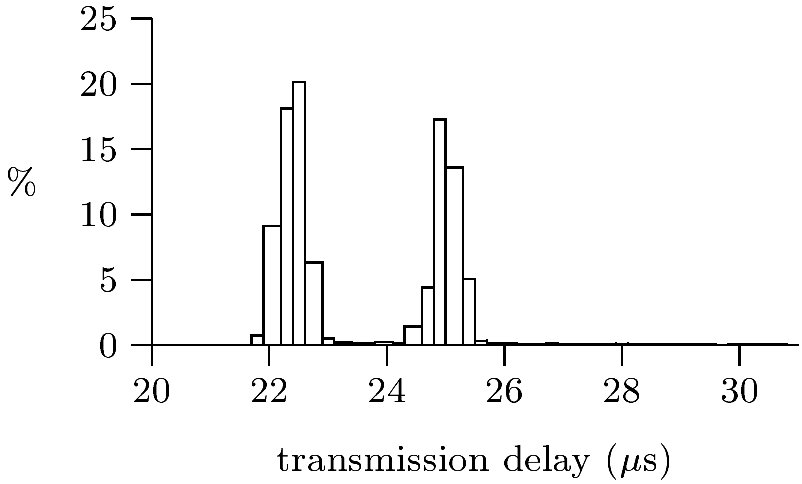
Measurement Results (1)

Collision-free Ethernet operation:



Minimum	21.7 μs
99%-minimum	21.7 μs
95%-minimum	21.7 μs
95%-maximum	22.6 μs
99%-maximum	22.9 μs
Maximum	22.9 μs
Average	22.3 μs
Std.Dev.	0.3 μs

Unconstrained Ethernet operation:



Minimum	20.4 μs
99%-minimum	21.7 μs
95%-minimum	21.9 μs
95%-maximum	25.5 μs
99%-maximum	31.0 μs
Maximum	36.8 μs
Average	23.6 μs
Std.Dev.	1.6 μs

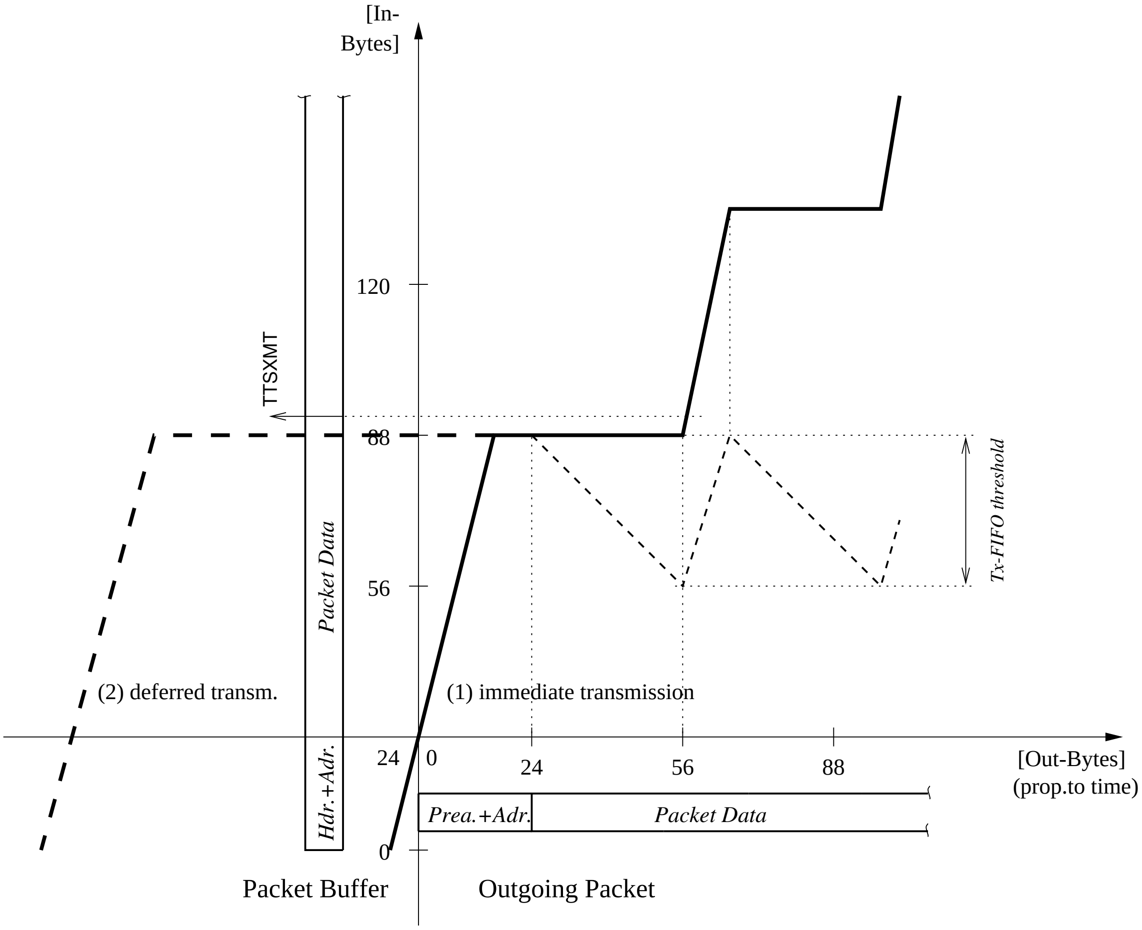
Accordingly,

- ε in the 10 μs-range
- ε_{95} in the μs-range
- no dependence upon node and network load

Measurement Results (2)

Major impact of i82596 FIFOs:

- 64-byte Tx-FIFO with 32-byte threshold
- 128-byte Rx-FIFO with 64-byte threshold



Choosing TxTS trigger offset properly is vital:

- "Data-sheet guess" (byte 28) resulted in $\varepsilon \approx 1.2$ ms
- Experiments revealed right setting $>$ byte 92

Conclusions

We presented some results of the experimental evaluation of our NTI M-Module in an Ethernet-based system.

The NTI supports

- μs-range time distribution via memory-based packet timestamping
- interval-based clock synchronization via the UTCSU-ASIC

Major findings:

- NTI decreases ε at least by 10^{-3} w.r.t. software-based timestamping
- Proper TxTS triggering offset selection vital
- Bus-speed also important

NTI limitations:

- Restricted to network controllers with small FIFOs (or exported trigger signals) only
- Assessment of ε usually requires measurements