



TECHNISCHE UNIVERSITÄT
CAROLO-WILHELMINA
ZU BRAUNSCHWEIG



SEP SS2007

Distributed and Ubiquitous Systems

Prof. Dr.-Ing. Michael Beigl

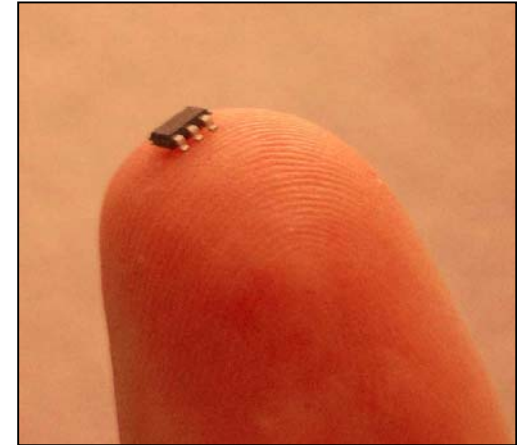


Institut für Betriebssysteme und Rechnerverbund
Technische Universität Braunschweig
Mühlenpfordtstr. 23, 1st floor
38106 Braunschweig



Termine

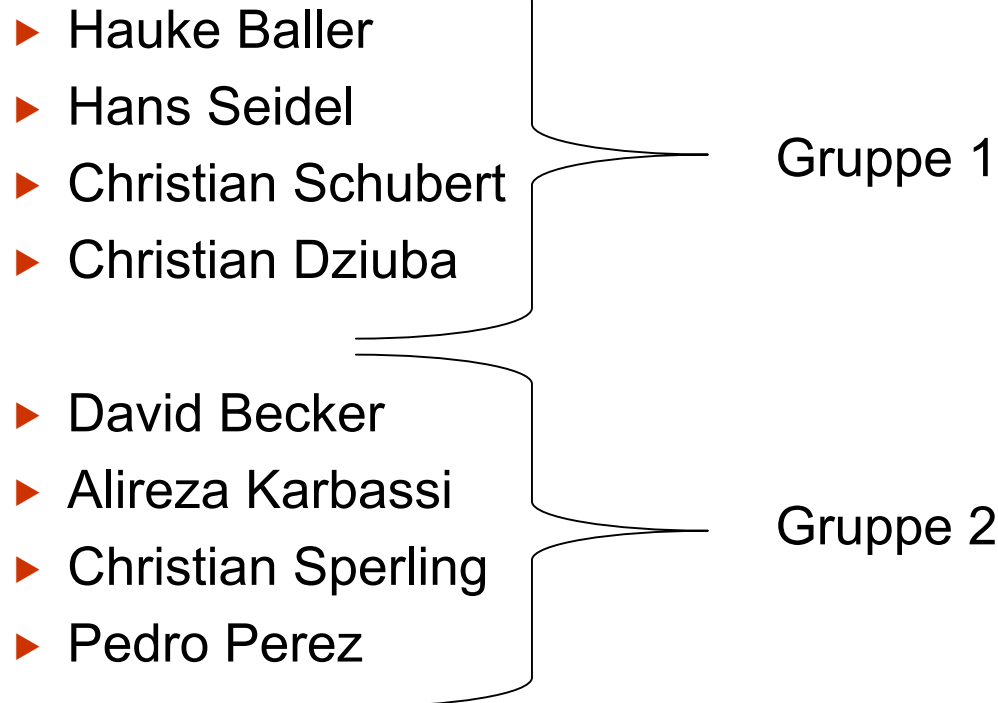
- × 10.04. Initiales Treffen
- × 16.04. Abgabe Pflichtenheft
- × 30.04. Abgabe Grobentwurf
- × 14.05. Abgabe Feinentwurf
- × 21.05. Abgabe Präsentationsfolien
- × 22.05. Zwischenpräsentation
- × 04.06. Zwischenabgabe Implementierung
- × 18.06. Abgabe Implementierung
- × 09.07. Abgabe Testbericht und fertige Implementierung
- × 16.07. Abgabe Plazapräsentation
- × 19.07. Plazapräsentation





Anmeldungen

× Teilnehmer



× 2 Gruppen a 4 Mitglieder



Vorgaben SSE

- × **Fokus liegt auf den Dokumenten (Pflichtenheft, Entwürfe, Präsentationen, Testberichte...)**
- × **Ablauf strikt nach dem Wasserfallmodell**
- × **Template für Pflichtenheft auf Webseite (Nutzung Pflicht!)**

- × **Zwischenpräsentationen**
 - ▶ Finden zentral statt
 - ▶ Verteilt auf mehrere Tage (Die, Mi, Do – Termine im Netz)
 - ▶ Jede Gruppe präsentiert eigenen Ergebnisse
 - ▶ 10 Minuten Präsentation / 10 Minuten Rückfragen

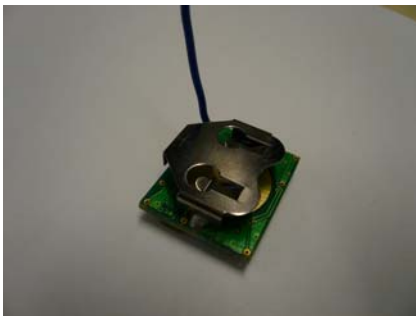
- × **Bewertungen**
 - ▶ Intern
 - ▶ Kriterien zentral festgelegt (Pflichtenheft 10%, Implementierung 33% ... etc)
 - ▶ Neues Kriterium: „allgemeine Beteiligung“ des Einzelnen

- × **Subversionsystem**
 - ▶ SVN
 - ▶ Zentral aufgesetzt (SSE)
 - ▶ Benutzung obligatorisch



Particle Computer Platform

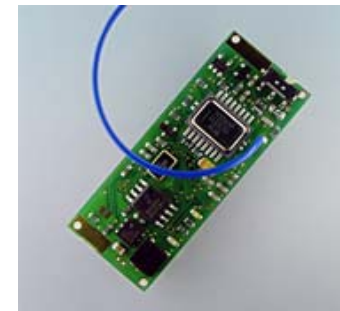
- ✖ Based on the EU Smart-Its project
 - ▶ **Universal platform for research projects in ubicomp**
 - ▶ **Easy to use, cheap, reusable platform**
 - ▶ **Modular hardware**
 - **Flexible adaption to changing project requirements**
 - **Contains simple, cost-efficient sensors**
- ✖ Particle Computer
 - ▶ **μ Parts, cParts, pParts**



μ Part

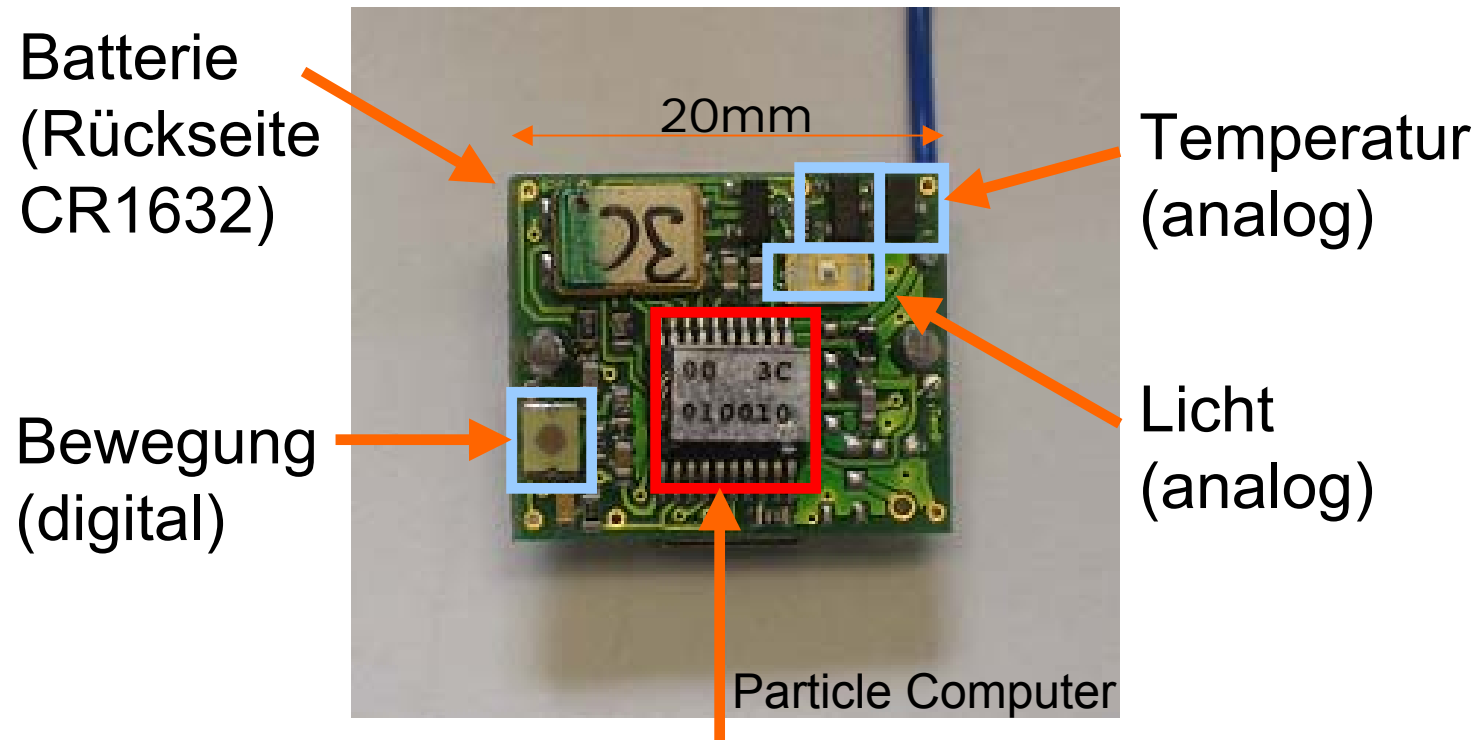


cPart



pPart

Sensoren am Beispiel des μ Part



Microcontroller und Kommunikation

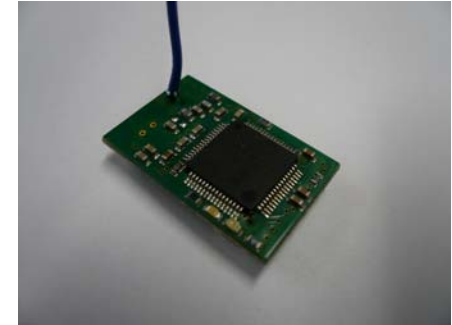
- Analog-Digital Konverter
- Digital I/O



Particle Computer Platform (3)

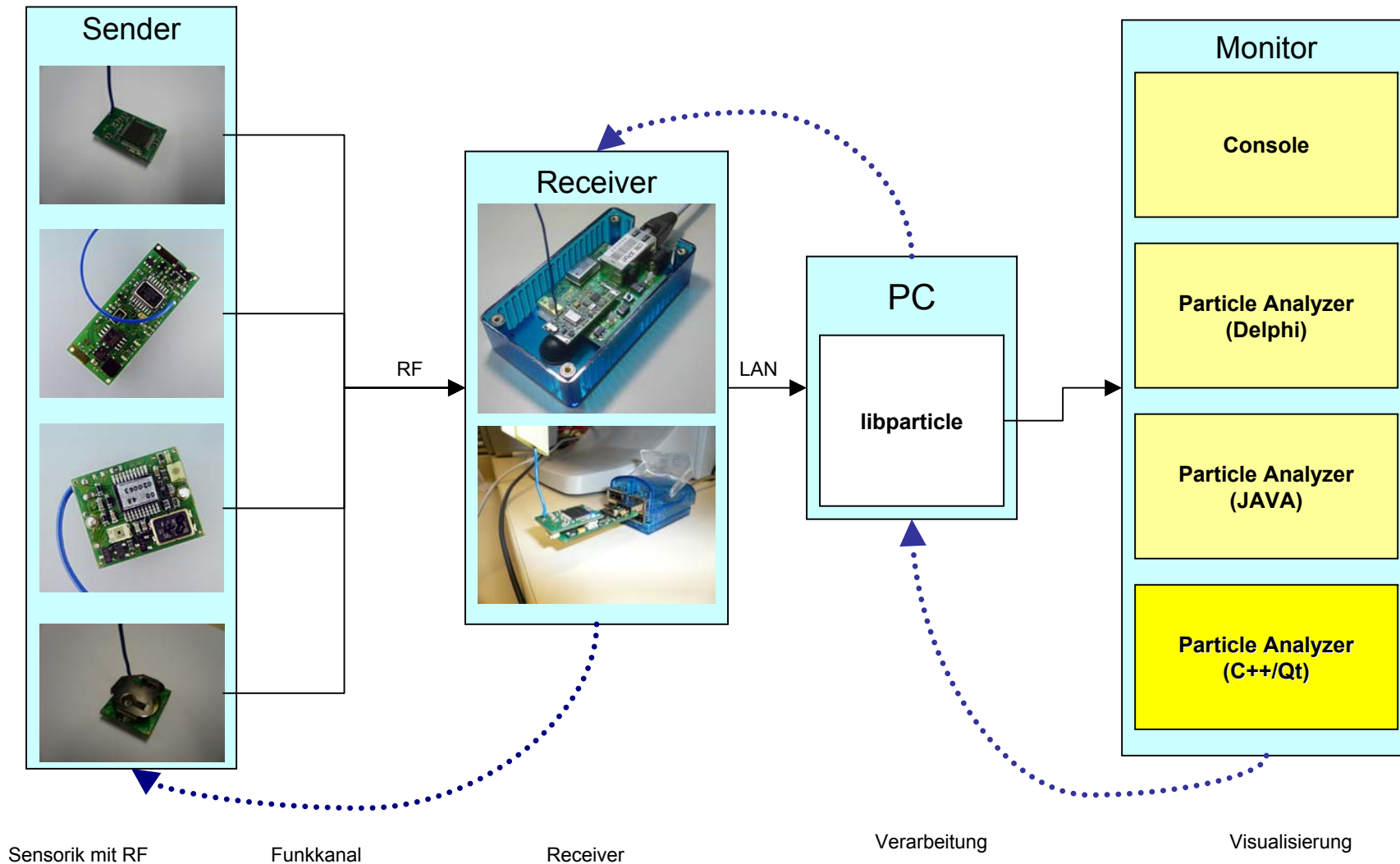
cParts

- ✗ 18x30mm Particle including
 - ▶ **Processor (8051)**
 - ▶ **RAM (4kbyte), Program Flash (32kbyte)**
- ✗ 868, 914 MHz Band Communication (software selectable channels) or 433, 310 MHz Band
- ✗ Field strength regulation
- ✗ Range: 1-30 meters in house
- ✗ 4 A/D channels
- ✗ SPI, I2C, serial communication
- ✗ Transmitt and receive
- ✗ Real-time clock / automatic time synchronization via network protocol
- ✗ 2 LED output
- ✗ 3.3-3.6 V supply voltage
- ✗ Language: C
- ✗ Uses Free Compiler (SDCC, see TecO-webpage)





Particle Computer Platform (4)



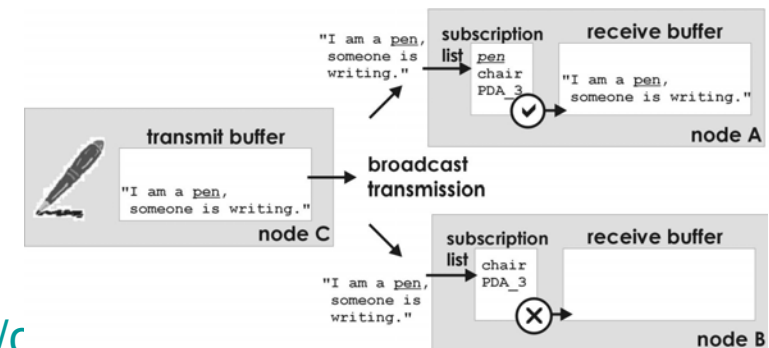
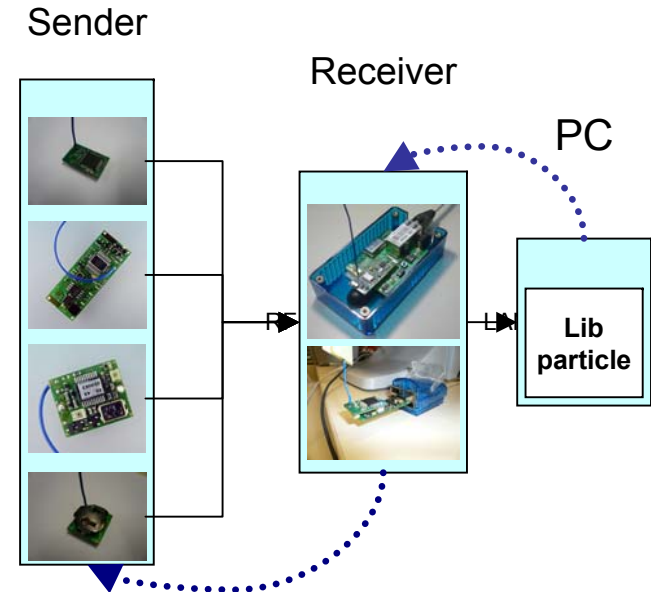


× USB-Bridge

- ▶ Schnittstelle zwischen Particle und Computer
- ▶ Kommunikation zu/von Particles via RF
- ▶ Broadcasted empfangene Pakete auf alle Netzwerkdevices (default: Port 5555/5556)

× ConCom

- ▶ ACL-Typen
 - Sprache, um Informationen insbesondere Kontext ähnlich einer gesprochenen Sprache zu übertragen
 - Spezifiziert von TecO, Uni Karlsruhe
 - <http://particle.teco.edu/documentation/content/concom.html>





Motivation

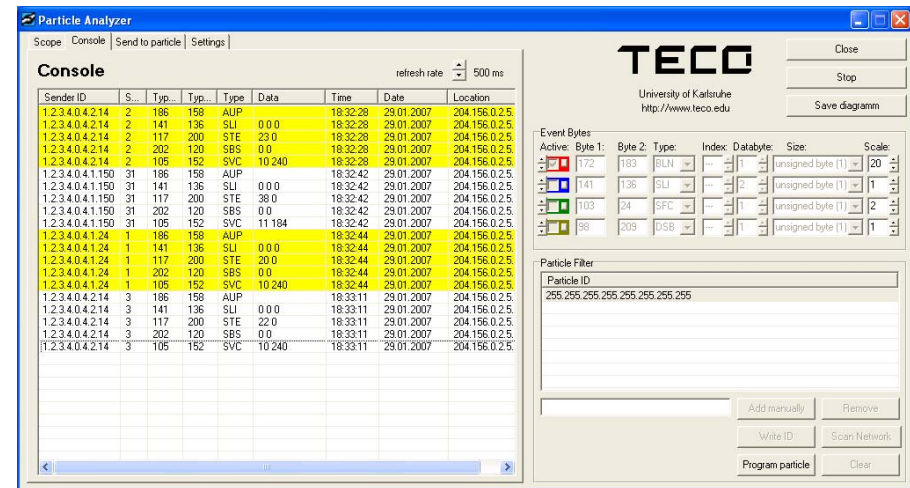
Particle Analyzer (Delphi)

✖ Vorteile

- ▶ Stabil (!!!)
- ▶ Intuitiv
- ▶ Keine Installation/VM/Interpreter notwendig
- ▶ Plots

✖ Nachteile

- ▶ Featuremangel
- ▶ Bedienung mitunter unhandlich (Bsp. Copy/Paste von IDs...)
- ▶ Ergebnis kann nicht aufgezeichnet/replayed werden
- ▶ Nur eindimensionale Filter (kann z.B. nicht nach ID und ACL filtern)





Motivation (3)

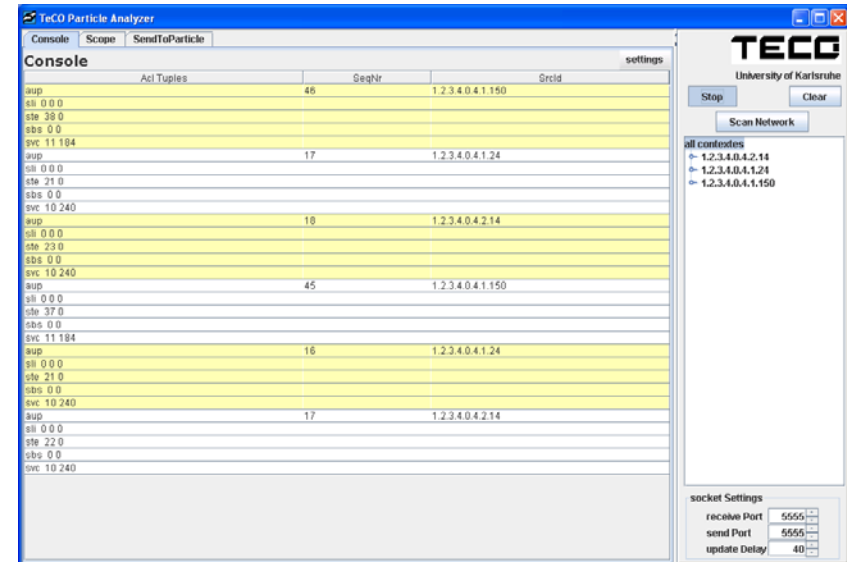
Particle Analyzer (Java)

✗ Vorteile

- ▶ Intuitiv
- ▶ Zusätzliche Features implementiert
- ▶ Bsp: Copy/Paste von IDs

✗ Nachteile

- ▶ INSTABIL (!!!)
- ▶ Java VM notwendig





Motivation (4)

Konkurrenzprodukte

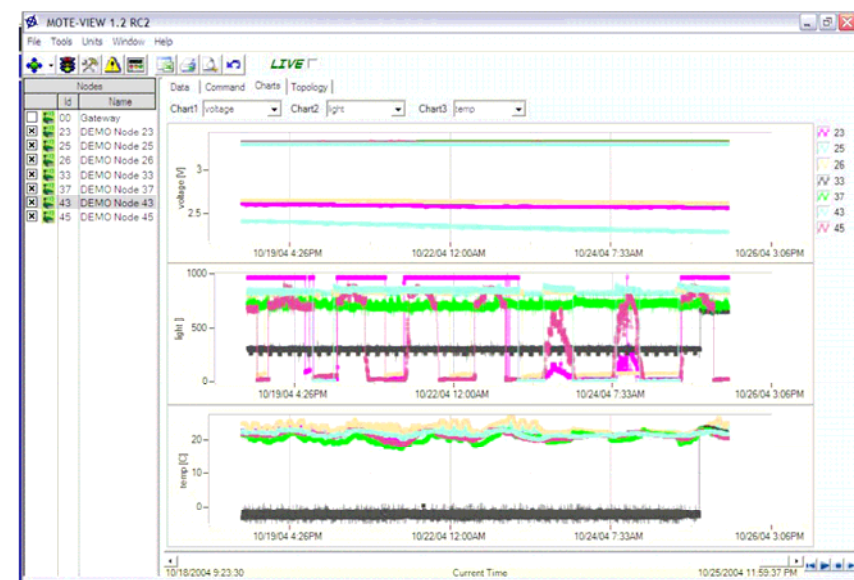
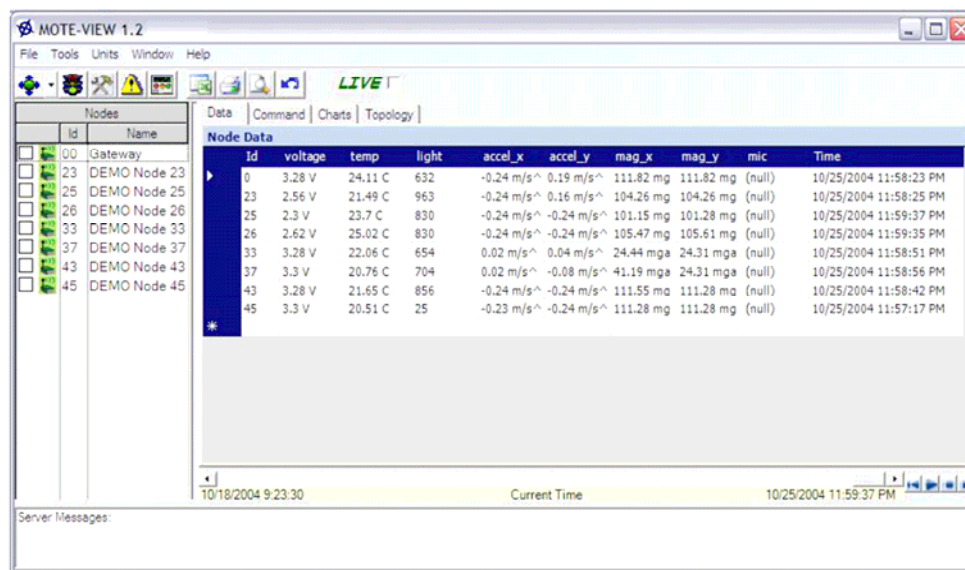
× MIT

- ▶ http://architecture.mit.edu/house_n/publications.html

× Crossbow „Mote View“

- ▶ Visualisierungstool für „Mote“-Sensor Nodes
- ▶ <http://www.xbow.com/Technology/UserInterface.aspx>
- ▶ http://www.willow.co.uk/MOTE-VIEW_Users_Manual_.pdf

× ... Unser Tool soll besser sein !!!





Motivation (5)

Mote-View (Data view)

MOTE-VIEW 1.2

File Tools Units Window Help

LIVE

Nodes

	Id	Name
<input type="checkbox"/>	00	Gateway
<input type="checkbox"/>	23	DEMO Node 23
<input type="checkbox"/>	25	DEMO Node 25
<input type="checkbox"/>	26	DEMO Node 26
<input type="checkbox"/>	33	DEMO Node 33
<input type="checkbox"/>	37	DEMO Node 37
<input type="checkbox"/>	43	DEMO Node 43
<input type="checkbox"/>	45	DEMO Node 45

Data Command Charts Topology

Node Data

	Id	voltage	temp	light	accel_x	accel_y	mag_x	mag_y	mic	Time
▶	0	3.28 V	24.11 C	632	-0.24 m/s^	0.19 m/s^	111.82 mg	111.82 mg	(null)	10/25/2004 11:58:23 PM
	23	2.56 V	21.49 C	963	-0.24 m/s^	0.16 m/s^	104.26 mg	104.26 mg	(null)	10/25/2004 11:58:25 PM
	25	2.3 V	23.7 C	830	-0.24 m/s^	-0.24 m/s^	101.15 mg	101.28 mg	(null)	10/25/2004 11:59:37 PM
	26	2.62 V	25.02 C	830	-0.24 m/s^	-0.24 m/s^	105.47 mg	105.61 mg	(null)	10/25/2004 11:59:35 PM
	33	3.28 V	22.06 C	654	0.02 m/s^	0.04 m/s^	24.44 mga	24.31 mga	(null)	10/25/2004 11:58:51 PM
	37	3.3 V	20.76 C	704	0.02 m/s^	-0.08 m/s^	41.19 mga	24.31 mga	(null)	10/25/2004 11:58:56 PM
	43	3.28 V	21.65 C	856	-0.24 m/s^	-0.24 m/s^	111.55 mg	111.28 mg	(null)	10/25/2004 11:58:42 PM
*	45	3.3 V	20.51 C	25	-0.23 m/s^	-0.24 m/s^	111.28 mg	111.28 mg	(null)	10/25/2004 11:57:17 PM

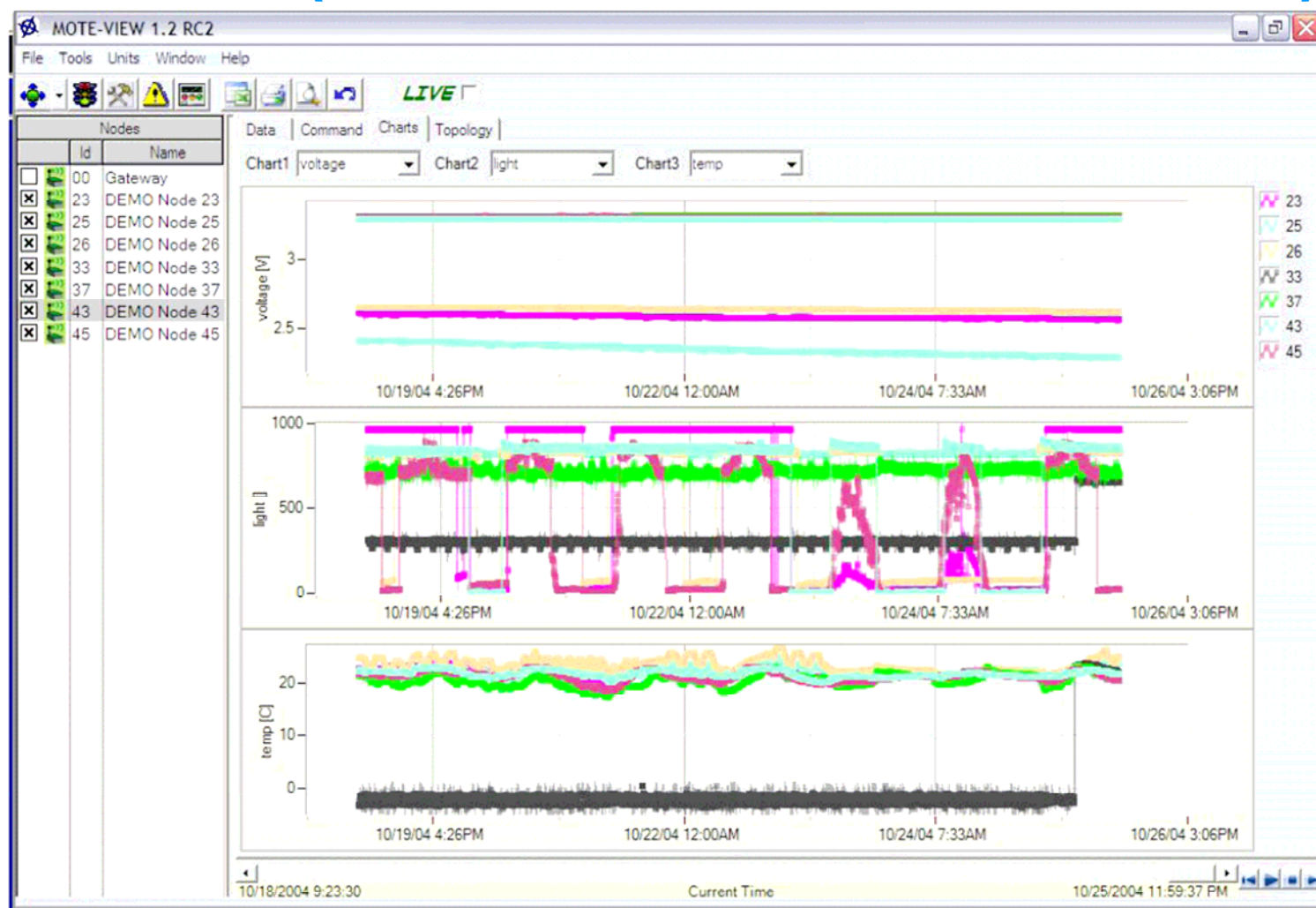
10/18/2004 9:23:30 Current Time 10/25/2004 11:59:37 PM

Server Messages:



Motivation (6)

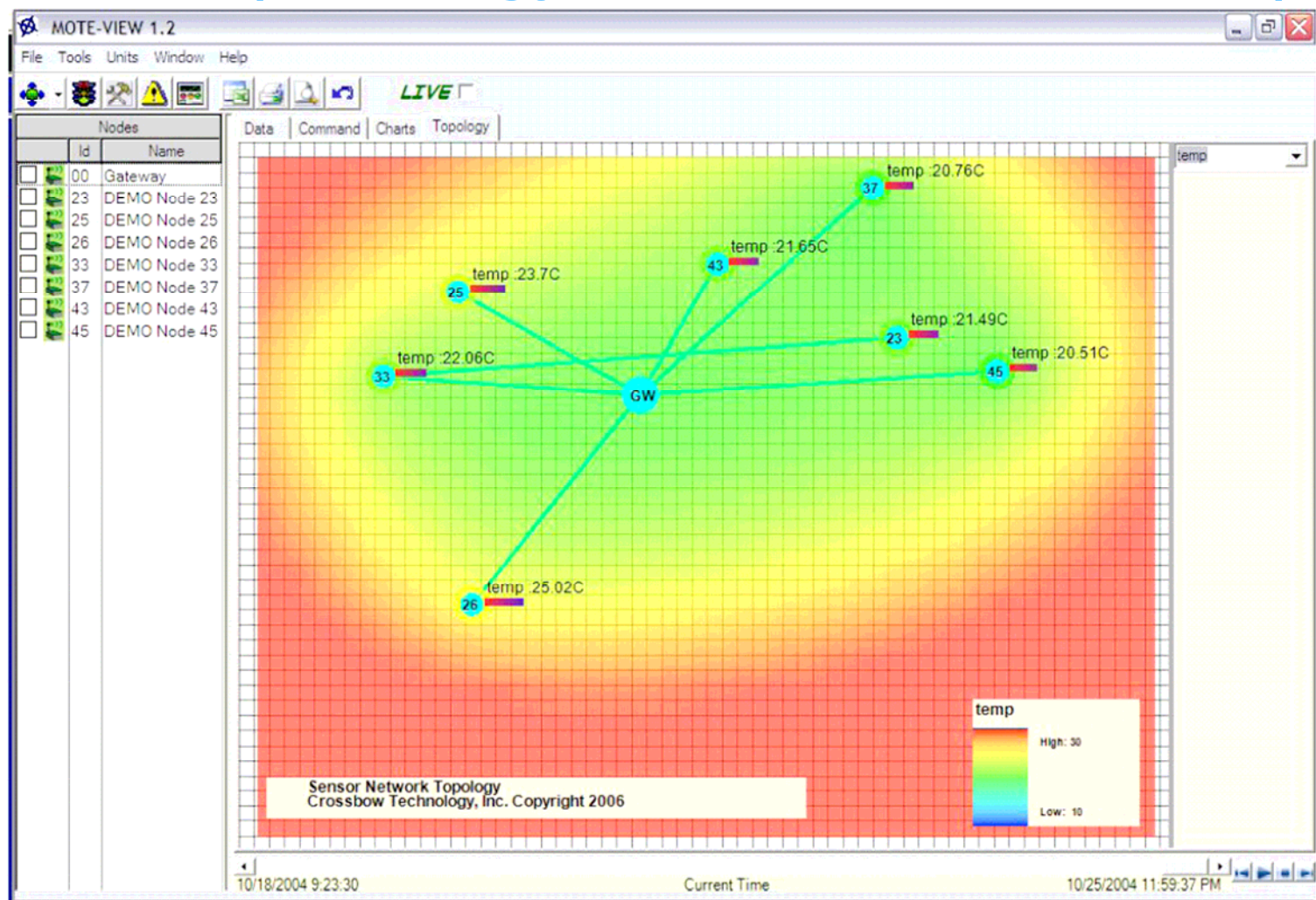
Mote-View (Chart view, demo-database)





Motivation (7)

Mote-View (Topology-view, demo-database)





Vorgehen

- ✗ **Zunächst Fokus auf Implementierung eines Particle Analyzers mit folgenden Eigenschaften**
 - ▶ Minimalsystem
 - ▶ Stabil (!!!)
 - ▶ Klar strukturiert (Benutzeroberfläche und Code)

- ✗ **Die Implementierung wird auf Herz und Nieren getestet**
 - ▶ Belastungstest
 - ▶ Test auf Echtzeitfähigkeit
 - ▶ Benutzerfreundlichkeit

- ✗ **Anschließend Implementierung der Zusatzfeatures**

- ✗ **Obligatorisch**
 - ▶ Benutzerdokumentation
 - ▶ Entwicklerdokumentation



Allgemein

- × **Programmiersprache C/C++**
- × **Graphische Oberfläche in Qt**
- × **Entwerfen eines User-Interface-Konzepts**
- × **Bedingung: executable oder „one-click-installation“**
 - ▶ ... die sofort lauffähig ist
 - ▶ Keine Abhängigkeiten von Webserver
 - ▶ Keine Abhängigkeiten von Datenbanken
 - ▶ Keine Installation weiterer dlls notwendig
 - ▶ Maximal eine Installationsroutine, die sich per Mausklick starten läßt
- × **Reguläre Ausdrücke**
 - ▶ z.B. Darstellung der Temperaturen aller Sensorknoten
 - ▶ Oder Multiplikation zweier Messwerte... usw.



Das Minimalsystem

- ✗ **Datenempfang von Paketen via Libparticle**
- ✗ **Implementierung eines Minimalsystems**
 - ▶ Modularer Aufbau
 - ▶ Filter (ACL, ID...)
 - ▶ Paketansicht in Tabellenform
 - ▶ Plot / Scope
 - ▶ Send to Particle: Broadcast, Adressed, Acked
 - ▶ Unterstützung von regulären Ausdrücken zum Auswerten der Daten (z.B. multipliziere byte 0 und 1 eines ACL Typen)



Erweiterte Funktionen

- × **Protokollierung/Replay von Daten**
- × **Modulübergreifende Kommunikation**
- × **Modulinterface mit Hierarchien**
- × **Verschiedene Sende-/Empfangsports in einem Analyzer**
- × **Daten in Datenbank schreiben (Datenbank einlesen)**
 - ▶ ABER: Keine Abhängigkeiten von Datenbanken/Webserver...
 - ▶ Tool muss auch ohne WS/DB laufen
- × **Nice-to-have**
 - ▶ Signalflußgraph zum Erstellen von Filtern (auch kaskadiert)



Vorgabe

- × **Lastenheft**
- × **cPart, μ Part oder pPart, der kontinuierlich Pakete verschickt**
- × **Kurzanleitung „Howto install a cPart-System“**
- × **Testroutine**
 - ▶ Analyzer mit speziellen Parametern
 - ▶ Pakete werden in hoher Geschwindigkeit aus einer Testdatenbank geladen und mit dem Analyzer visualisiert
- × **μ Part mit Beschleunigungssensor**
- × **Particle Analyzer (Console, Delphi, Java)**



Belastungstest

✖ Pakete in Datenbank pipen / aus Datenbank lesen

- ▶ Tool: Konsolenanalyser: `particle_analyzer.exe`
- ▶ Input: Port 5555
- ▶ Output: `mysql_new://[user]:[pass]@host:3306/[db_name]`
- ▶ Parameter:
 - ◆ -i – Input
 - ◆ -o – Output
 - ◆ -b – Ausgabe der Daten ab Datum_Uhrzeit
 - ◆ -e – Ausgabe der Daten bis Datum_Uhrzeit
 - ◆ -d – Verzögerungszeit zwischen den einzelnen Paketen

✖ Bsp: Pakete in Datenbank speichern

- ▶ `particle_analyzer -i 5555 -o mysql_new://[user]:[pass]@host:3306/[db_name]`

✖ Bsp: Pakete aus Datenbank auf Port 5556

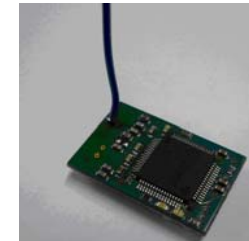
- ▶ `particle_analyzer -o 5556 -i mysql_new://[user]:[pass]@host:3306/[db_name] -b 2007/03/01_12:00:00 -e 2007/03/02_12:00:00 -d 50`



SEP Hardware

× Jede Gruppe erhält:

- ▶ 2 USB Bridges (Bridge+cPart)
- ▶ 2 cParts mit SEP Demo
- ▶ 1 uPart
- ▶ Batteriepacks als Stromversorgung für die cParts



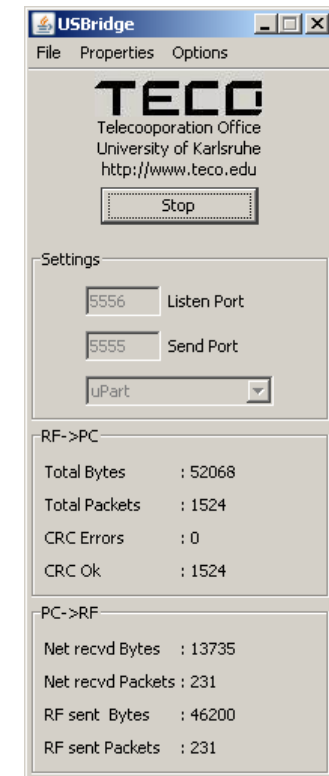
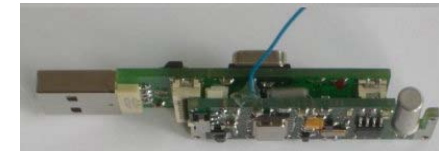


USB Bridge

- ✗ Broadcastet alle über RF empfangenen Pakete ins lokale Netz (auf Port 5555)
- ✗ Broadcastet alle im lokalen Netz empfangenen Pakete (Port 5556) über RF
- ➔ Ports des Particle Analysers umgekehrt!
- ✗ USBBridge Software

- ✗ Anleitung zur Installation und Nutzung:

<http://www.ibr.cs.tu-bs.de/~roehr/cpart/index.html>





cPart SEP Demo

✖ Stromversorgung:

- ▶ Batteriepack
- ▶ USB Bridge (Conan Connector)
- ▶ Extern über Molex

✖ Funktionen

- ▶ Sendet alle 1,2 Sekunden ein SEP Paket:

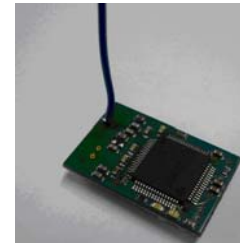
[ACL (ACL_ID) | payload | data]

[sep (184, 48) | 4 | 2 0 0 7] DUMMY

[cts (205, 56) | 4 | 0 0 0 201] TIMESTAMP

[adi (140, 54) | 4 | 7 25 0 0] DEBUG INFO

- ▶ Konfigurierbar über Pakete





✗ [sep (184, 48) | 4 | 2 0 0 7] **SUBJECT**

- ▶ Subject des Pakets, hat immer den gleichen Inhalt (4 byte)

✗ [cts (205, 56) | 4 | 0 0 0 201] **TIMESTAMP**

- ▶ Timestamp seitdem der cPart gestartet wurde (48 ms Auflösung)
- ▶ $\text{Timestamp} = \text{byte1} * 255^3 + \text{byte2} * 255^2 + \text{byte3} * 255 + \text{byte4}$
- ▶ Im Beispiel: $(0 * 255^3 + 0 * 255^2 + 0 * 255 + 201) * 48 \text{ms} = 9648$ (9,648 Sekunden)

✗ [adi (140, 54) | 4 | 7 25 0 0] **DEBUG INFO**

- ▶ byte1: # gesendeter Pakete (sollte der Seq. Nummer entsprechen)
- ▶ byte2: aktuelles timing (alle $48 \text{ms} * \text{byte2}$ wird ein Paket gesendet)
- ▶ byte3 & byte4: Informationen über zu sendende Pakete
 - byte3: aktuelles Paket
 - byte4: maximale Anzahl zu sender Pakete



cPart SEP Demo - Steuerung

✗ Häufiger Senden über [ACM]|[TIM], z.B.

[acm (165, 14) | 0 |] Subject
[tim (166, 17) | 1 | 10] Timing (48ms*timer),
hier alle: 48ms*10

→ 0 bedeutet kein Paket senden!!!

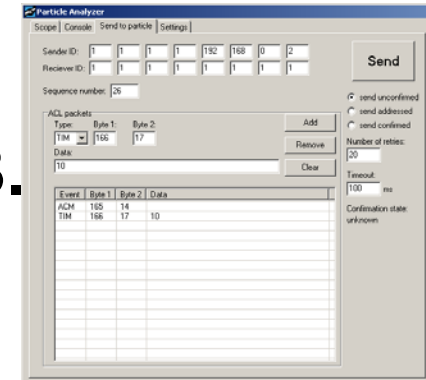
✗ Anzahl an Paketen senden (for Schleife) über [ACM][FOR], z.B.:

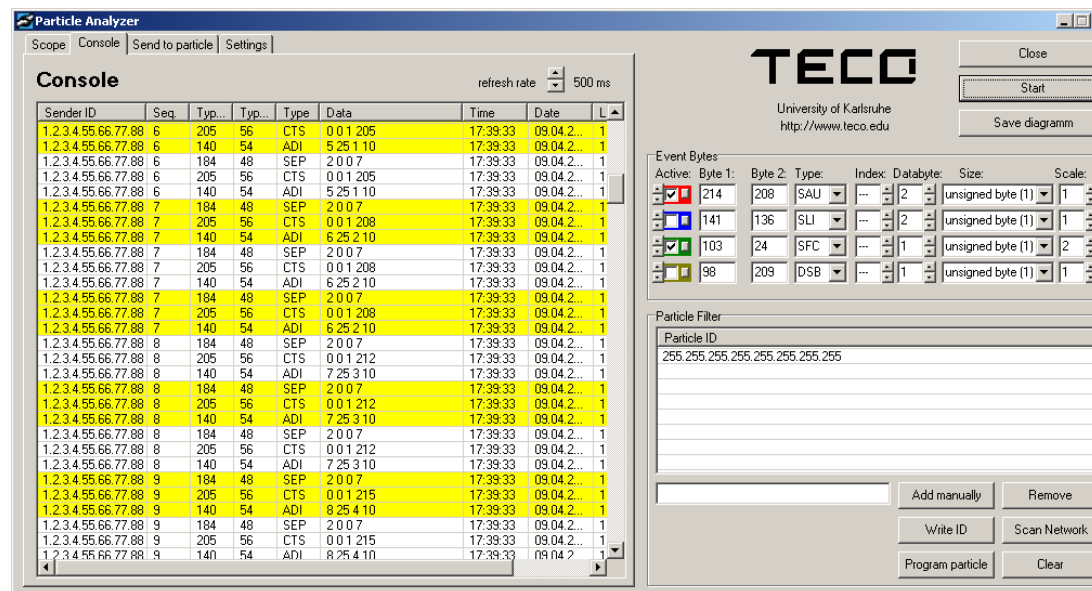
[acm (165, 14) | 0 |] Subject
[for (198,51) | 1 | 10] # Pakete
hier: 10 Pakete verschicken

Antwort z.B.:

[sep (184, 48) | 4 | 2 0 0 7]
[cts (205, 56) | 4 | 0 0 1 164]
[adi (140, 54) | 4 | 20 25 6 10]

Laufzeit: $419 \cdot 48\text{ms} = 20112$
byte3: aktuelles Paket: 6
byte4: insgesamt zu senden: 10







Links

× SEP-Seite im Netz

- ▶ <http://www.ibr.cs.tu-bs.de/courses/ss07/sep-dus/index.html>
- ▶ Gruppe DUS
 - Lehrveranstaltungen
 - SEP

× Particle Analyzer

- ▶ http://particle.teco.edu/software/development_tools/index.html

× Howtos cParts

- ▶ <http://www.ibr.cs.tu-bs.de/~roehr/cpart/>
- ▶ <http://www.ibr.cs.tu-bs.de/~roehr/cpart2/>
- ▶ <\\dus-server\users\droehr\sep2007>