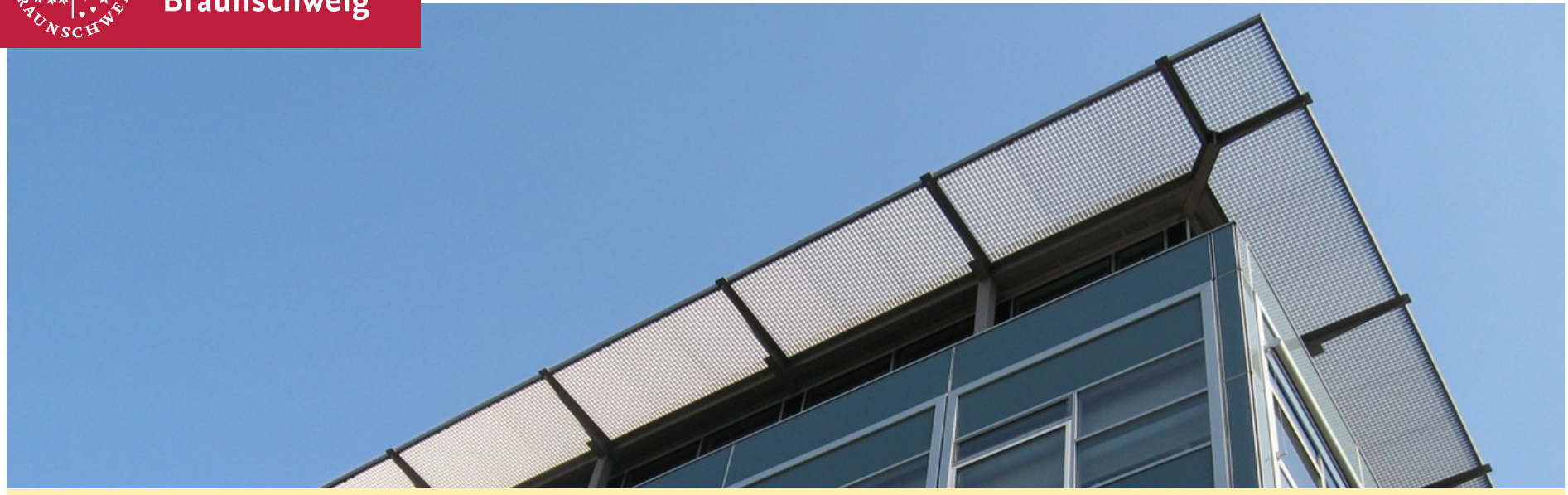
Technische Universität Braunschweig

Institut für Betriebssysteme und Rechnerverbund



Architektur eines universell einsetzbaren Sensorknotens

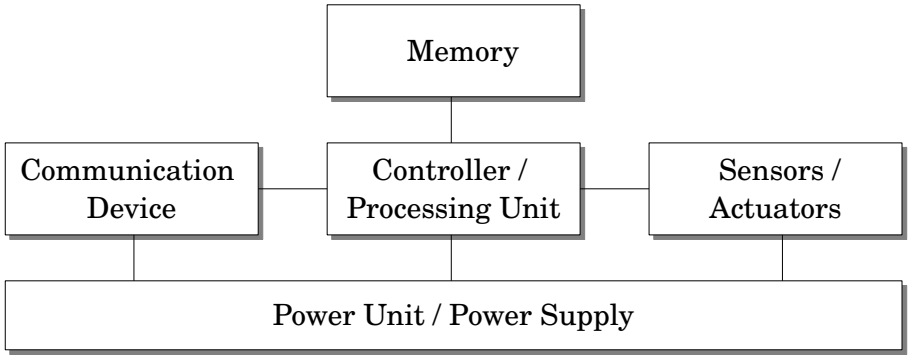
INGA (Inexpensive Node for General Applications)

Ulf Kulau

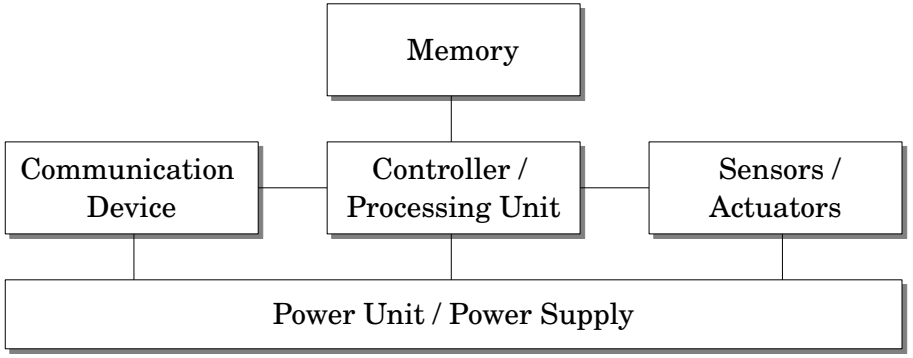
FGSN 2011

Inhalt

Gemeinsamkeit: Allgemeine Architektur von Sensorknoten

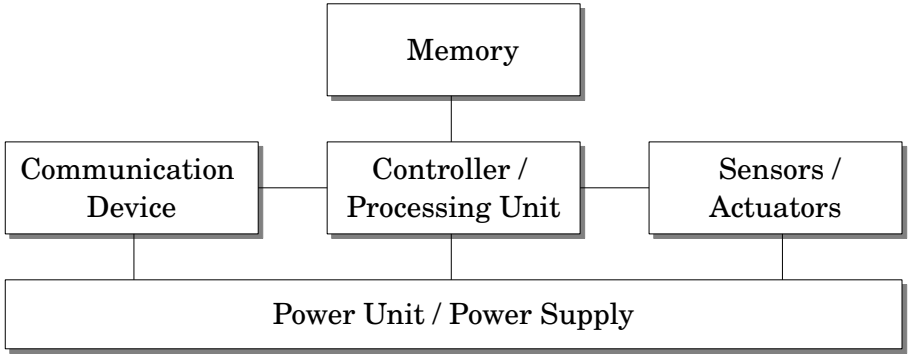


Gemeinsamkeit: Allgemeine Architektur von Sensorknoten



- Beliebige / Beliebig komplexe Realisierung der einzelnen Komponenten
- Energieeffizienz als dominierende 'Design Rule' für das Gesamtsystem

Gemeinsamkeit: Allgemeine Architektur von Sensorknoten

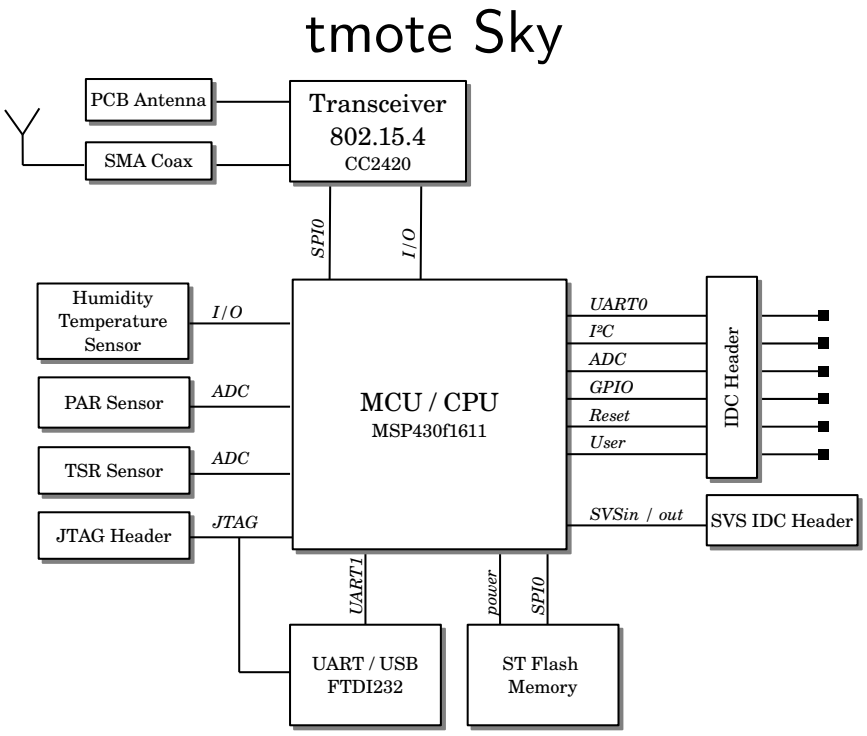


- Beliebige / Beliebig komplexe Realisierung der einzelnen Komponenten
- Energieeffizienz als dominierende 'Design Rule' für das Gesamtsystem

Entwicklungsziel INGA

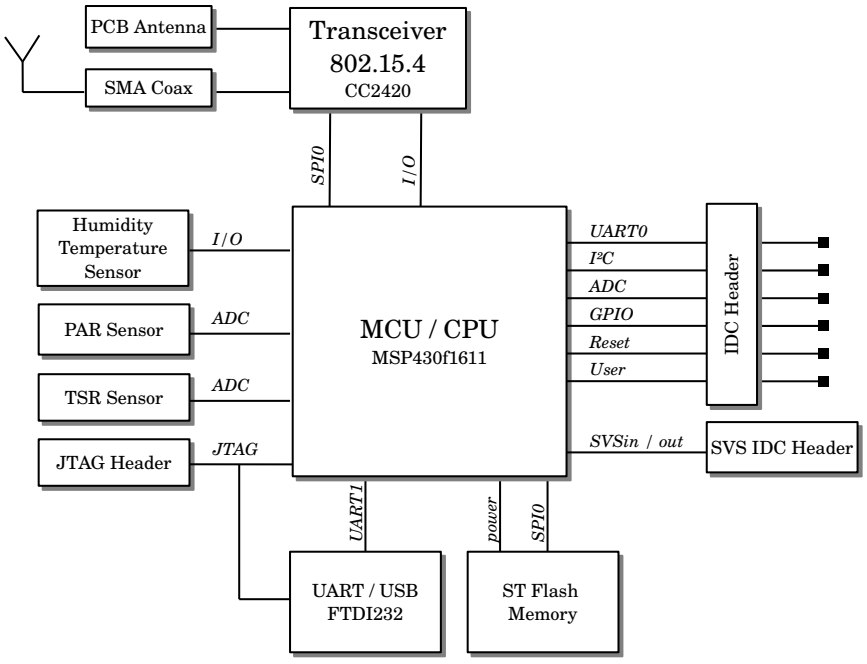
- Vorteile existierender Architekturen nutzen und optimieren
- Günstige Open-Hardware mit Kompatibilität zu Contiki

Sensorknoten Architekturen

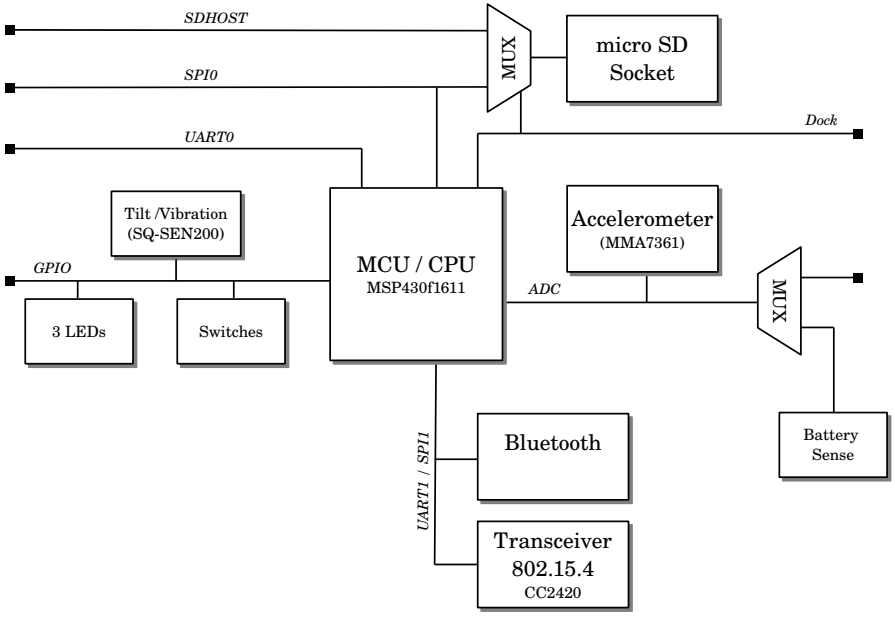


Sensorknoten Architekturen

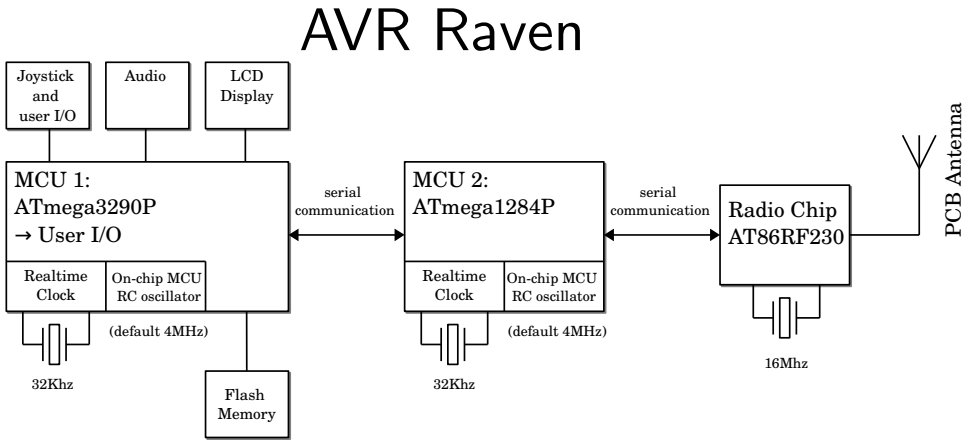
tmote Sky



Shimmer Sensor

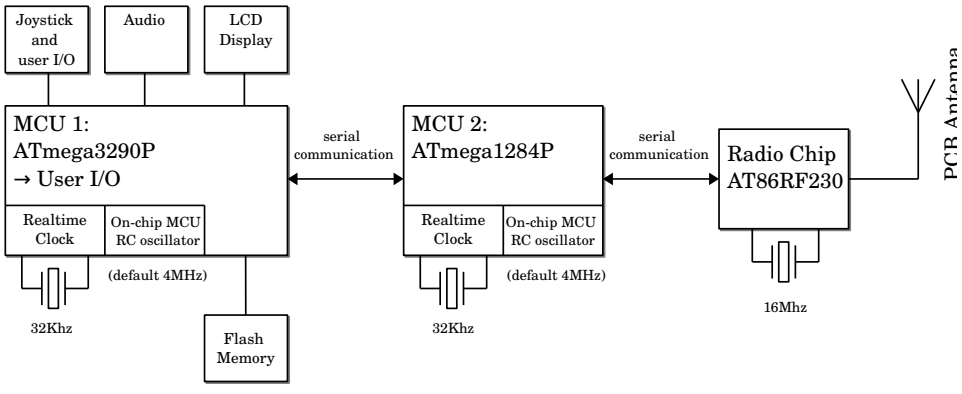


Sensorknoten Architekturen

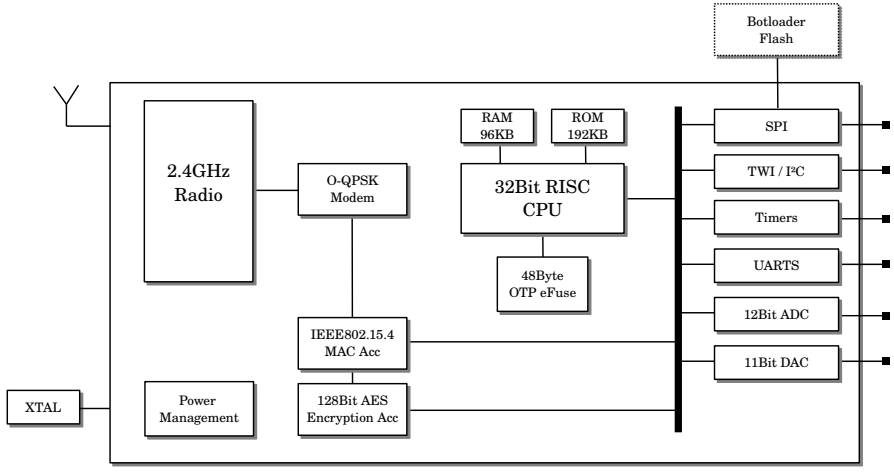


Sensorknoten Architekturen

AVR Raven

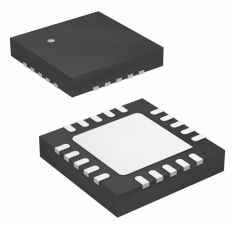


iSense



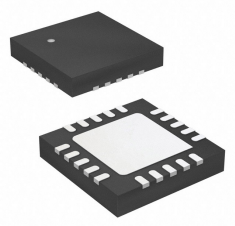
- Aufgrund der geforderten Kompatibilität zu Contiki OS ergeben sich folgende mögliche Basis-Architekturen für den INGA

Sensorknoten	Processing Unit	Communication Unit
Tmote Sky	Texas Instruments MSP430F1611	Texas Instrumets cc2420
AVR Raven	Atmel ATmega1284p	Atmel AT86RF230 Atmel AT86RF231



- Aufgrund der geforderten Kompatibilität zu Contiki OS ergeben sich folgende mögliche Basis-Architekturen für den INGA

Sensorknoten	Processing Unit	Communication Unit
Tmote Sky	Texas Instruments MSP430F1611	Texas Instrumets cc2420
AVR Raven	Atmel ATmega1284p	Atmel AT86RF230 Atmel AT86RF231



Kriterien zur Bewertung

- Leistungsaufnahme der Basisarchitektur
- Rechenleistung
- On-Chip Peripheriekomponenten (Kommunikationsinterfaces)

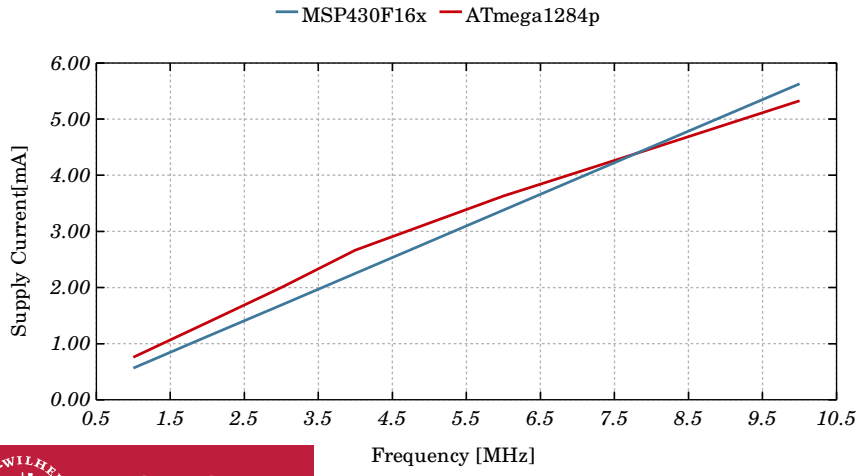
Leistungsaufnahme der Basisarchitektur

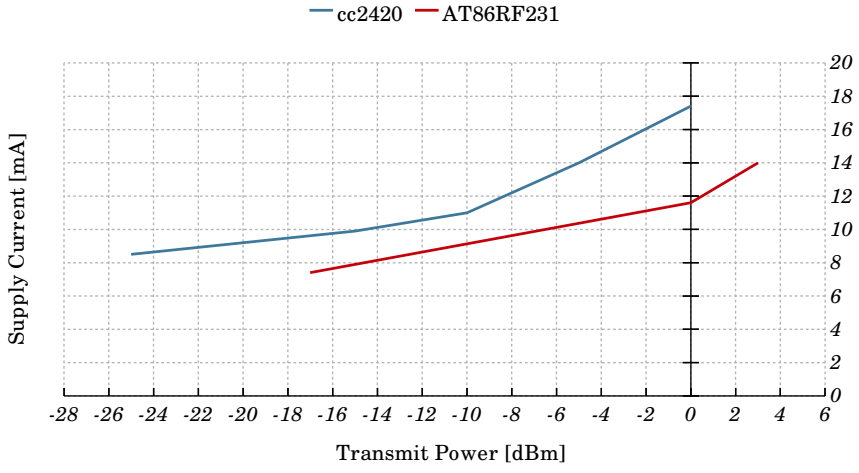
- Dynamische Verlustleistung dominiert die Leistungsaufnahme bei CMOS Technologie $P_{dyn} = C_l \cdot f \cdot U_b^2$
- Prozessorarchitektur (RISC)
- Verschiedenste Sleep-Modi

Leistungsaufnahme der Basisarchitektur

- Dynamische Verlustleistung dominiert die Leistungsaufnahme bei CMOS Technologie $P_{dyn} = C_l \cdot f \cdot U_b^2$
- Prozessorarchitektur (RISC)
- Verschiedenste Sleep-Modi

Stromaufnahme der Basisarchitekturen





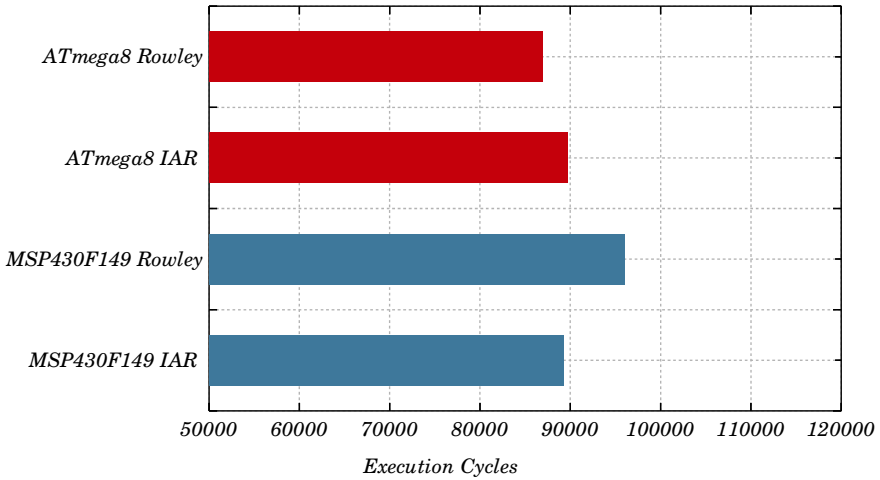
Rechenleistung

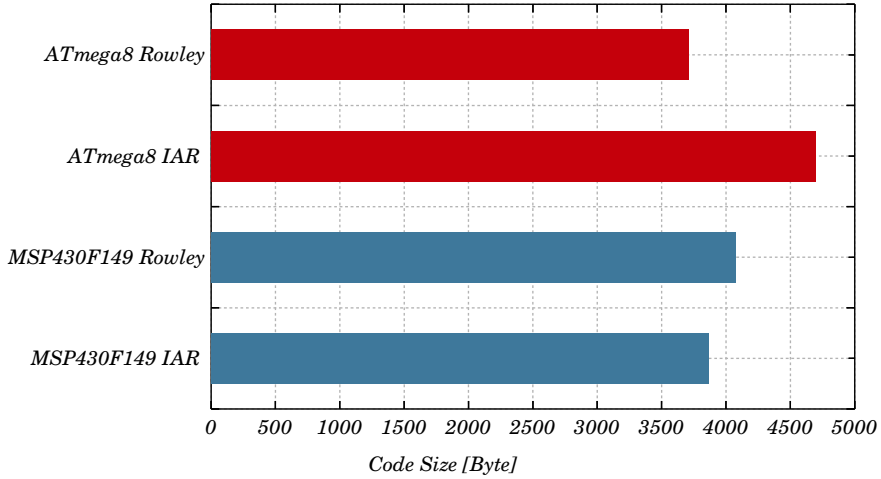
- Leistungsfähigkeit $\Lambda = \frac{1}{t_{ex}}$
- $MIPS = \frac{Clockrate}{CPI \cdot 10^6}$ ungeeignet da $CPI = 1$
- Benchmarks zur Bewertung (Vorsicht: tendenziös!)

Rechenleistung

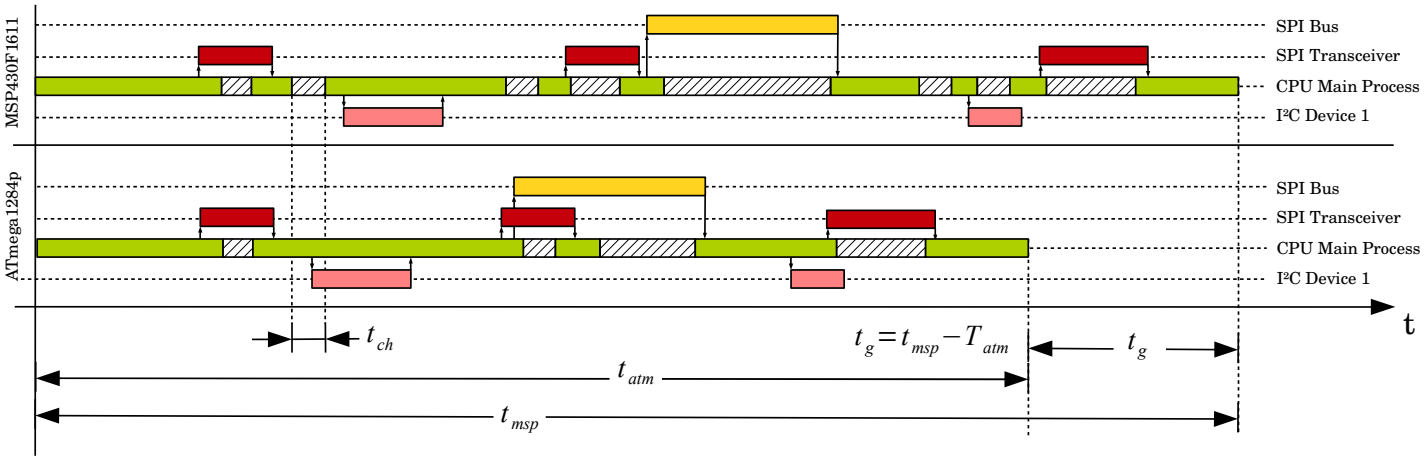
- Leistungsfähigkeit $\Lambda = \frac{1}{t_{ex}}$
- $MIPS = \frac{Clockrate}{CPI \cdot 10^6}$ ungeeignet da $CPI = 1$
- Benchmarks zur Bewertung (Vorsicht: tendenziös!)

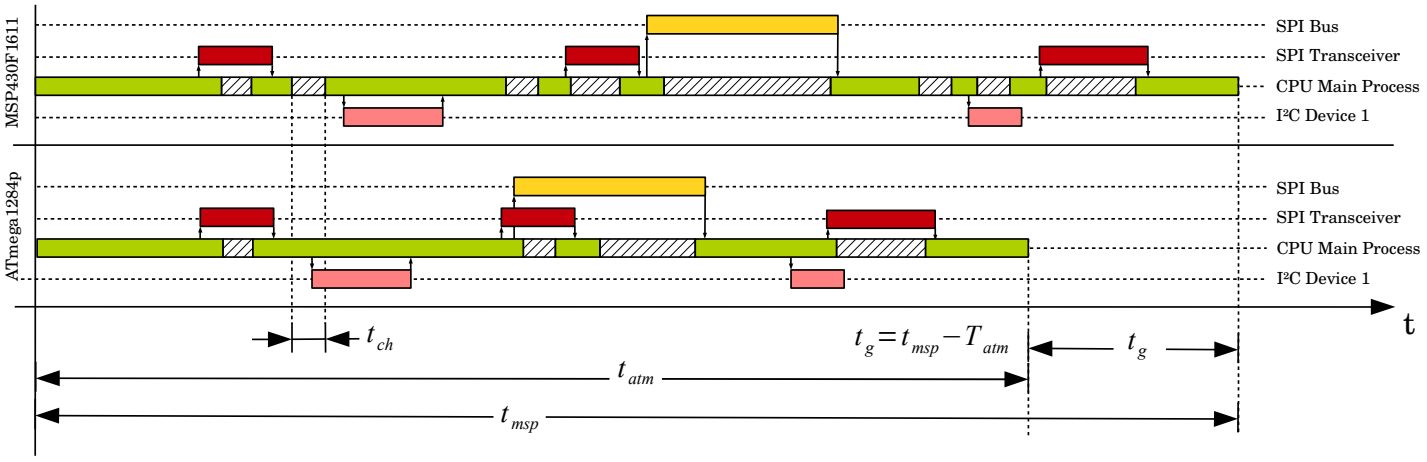
Auswertung (bedingt aussagekräftig)





On-Chip Peripheriekomponenten: ATmega1284p vs. MSP430





On-Chip Peripheriekomponenten (Kommunikationsinterfaces)

- Kein Umkonfigurieren des USART ($\Rightarrow t_{ex} \downarrow$)
- Parallelisierung in einem sequentiellen System
- Erhöhte Stromaufnahme durch zusätzliche Hardwarekomponenten, jedoch Verringerung der Leistungsaufnahme bei Kompression $\geq 2\%$
- Dedizierte Trennung von Komponenten auf Hardware-Ebene

Auswirkung auf die Verlustleistung

$$P_{ges} = \frac{1}{T} \cdot \int_0^T U_b(t) \cdot I_{cc}(t)\, dt \tag{1}$$

$$= \frac{1}{T} \cdot \left(\int_0^{T_{active}} P_{active}(t)\, dt + \int_{T_{active}}^{T} P_{sleep}(t)\, dt \right) \tag{2}$$

$$\text{aus } t_{ex} \downarrow = T_{active} \downarrow \quad \text{mit } P_{active} \gg P_{sleep} \Rightarrow P_{ges} \downarrow \tag{3}$$

Auswirkung auf die Verlustleistung

$$P_{ges} = \frac{1}{T} \cdot \int_0^T U_b(t) \cdot I_{cc}(t)\, dt \tag{1}$$

$$= \frac{1}{T} \cdot \left(\int_0^{T_{active}} P_{active}(t)\, dt + \int_{T_{active}}^{T} P_{sleep}(t)\, dt \right) \tag{2}$$

$$\text{aus} \quad t_{ex} \downarrow = T_{active} \downarrow \quad \text{mit} \quad P_{active} \gg P_{sleep} \Rightarrow P_{ges} \downarrow \tag{3}$$

Weiterer Faktor

- Große AVR Community (erleichtert den Einstieg)

Auswirkung auf die Verlustleistung

$$P_{ges} = \frac{1}{T} \cdot \int_0^T U_b(t) \cdot I_{cc}(t)\, dt \quad (1)$$

$$= \frac{1}{T} \cdot \left(\int_0^{T_{active}} P_{active}(t)\, dt + \int_{T_{active}}^{T} P_{sleep}(t)\, dt \right) \quad (2)$$

$$\text{aus } t_{ex} \downarrow = T_{active} \downarrow \quad \text{mit } P_{active} \gg P_{sleep} \Rightarrow P_{ges} \downarrow \quad (3)$$

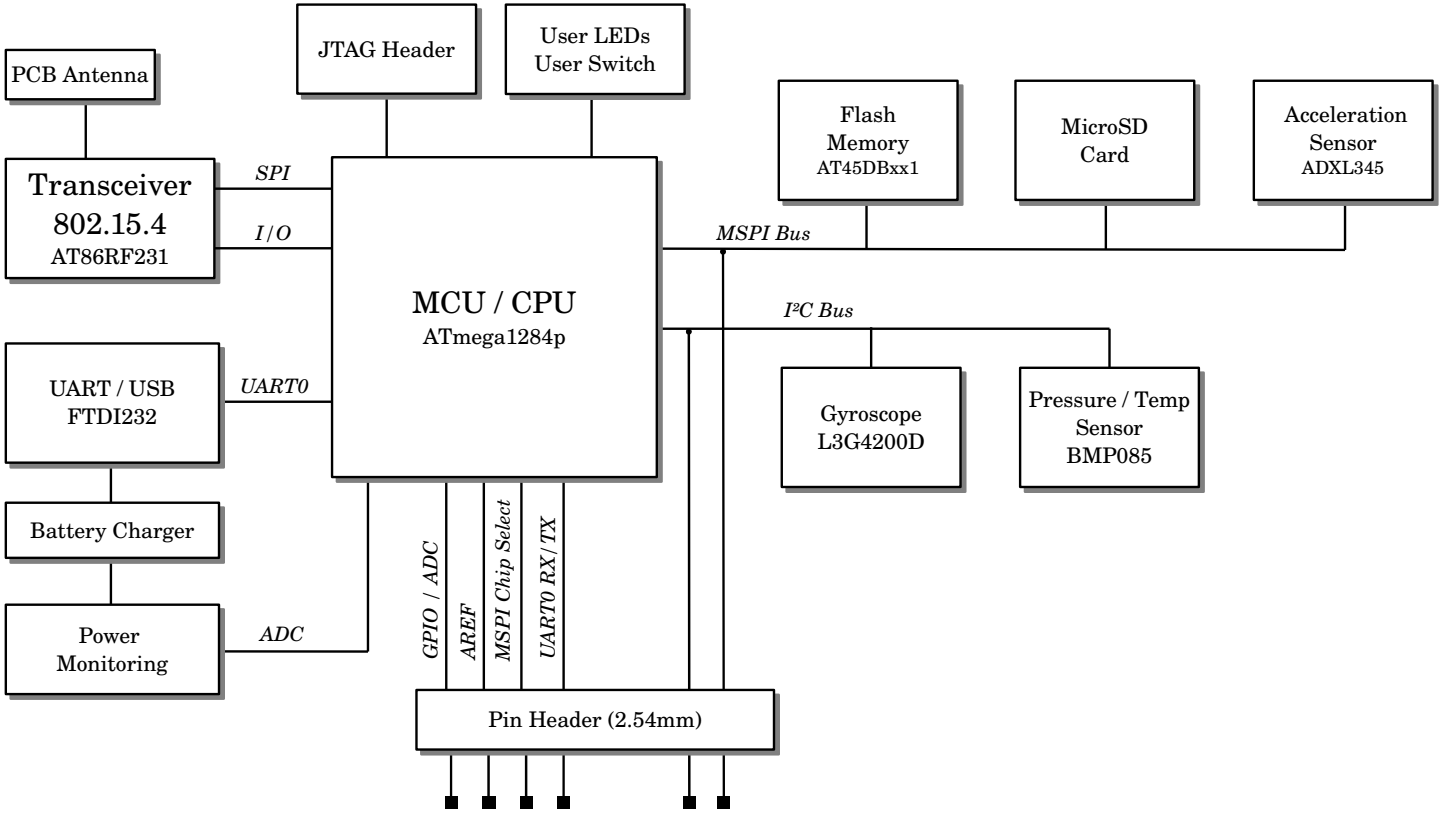
Weiterer Faktor

- Große AVR Community (erleichtert den Einstieg)

Resultierende Basis-Architektur

ATmega1284p + AT86RF231

INGA Design und Hardware Implementierung

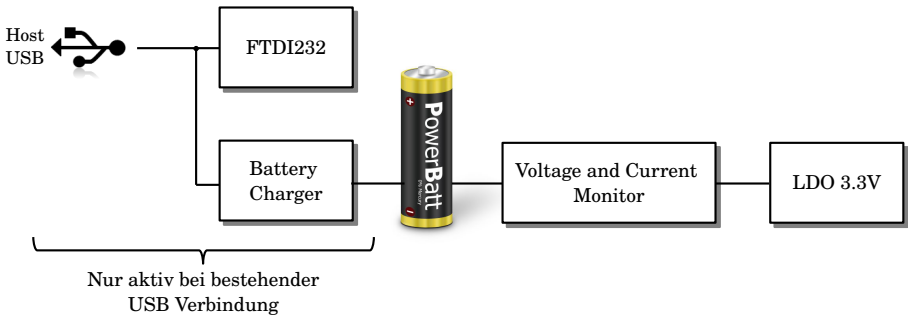


USB

- USB Unterstützung durch FTDI232 (UART to USB)
- Kontrollierter Reset via FTDI232

USB

- USB Unterstützung durch FTDI232 (UART to USB)
- Kontrollierter Reset via FTDI232

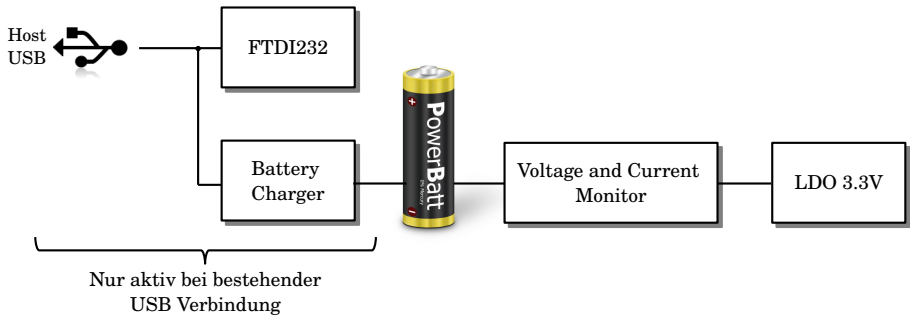


Power Management und Monitoring

- Diskreter Aufbau zur Reduzierung des Overheads $\overline{\omega_{pm}(I_{cc})} \approx 35\mu A$
- Online Spannungs- und Strommessung
- Laden eines LiIon Akku via USB
- Betrieb über USB, LiIon oder normale Batterie

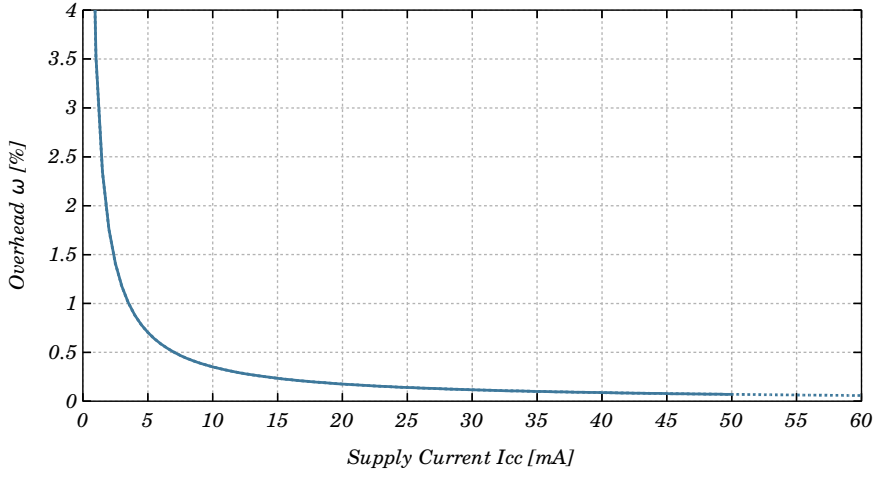
USB

- USB Unterstützung durch FTDI232 (UART to USB)
- Kontrollierter Reset via FTDI232

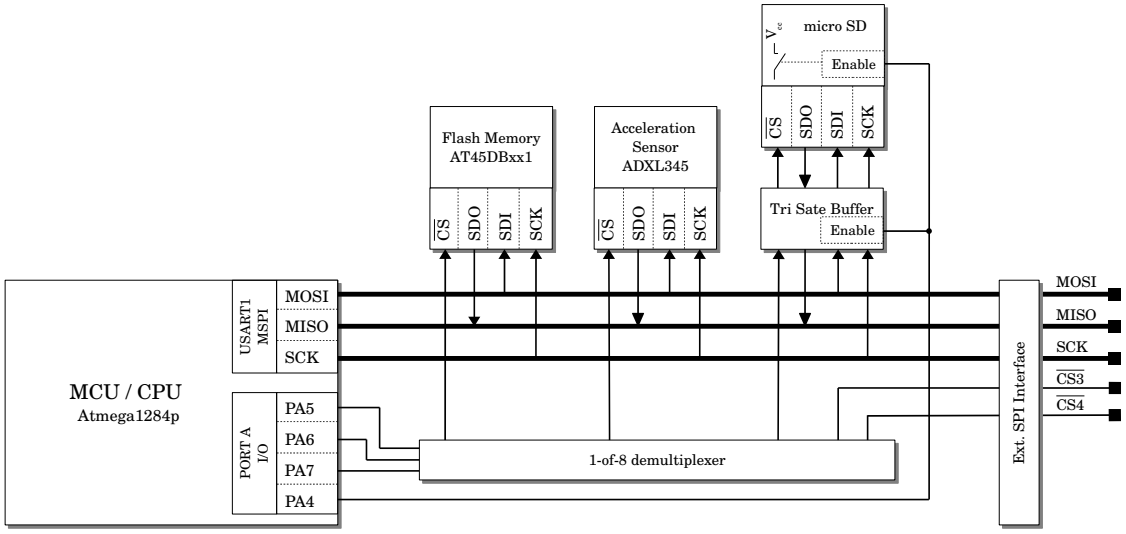


Power Management und Monitoring

- Diskreter Aufbau zur Reduzierung des Overheads $\overline{\omega_{pm}(I_{cc})} \approx 35\mu A$
- Online Spannungs- und Strommessung
- Laden eines LiIon Akku via USB
- Betrieb über USB, LiIon oder normale Batterie

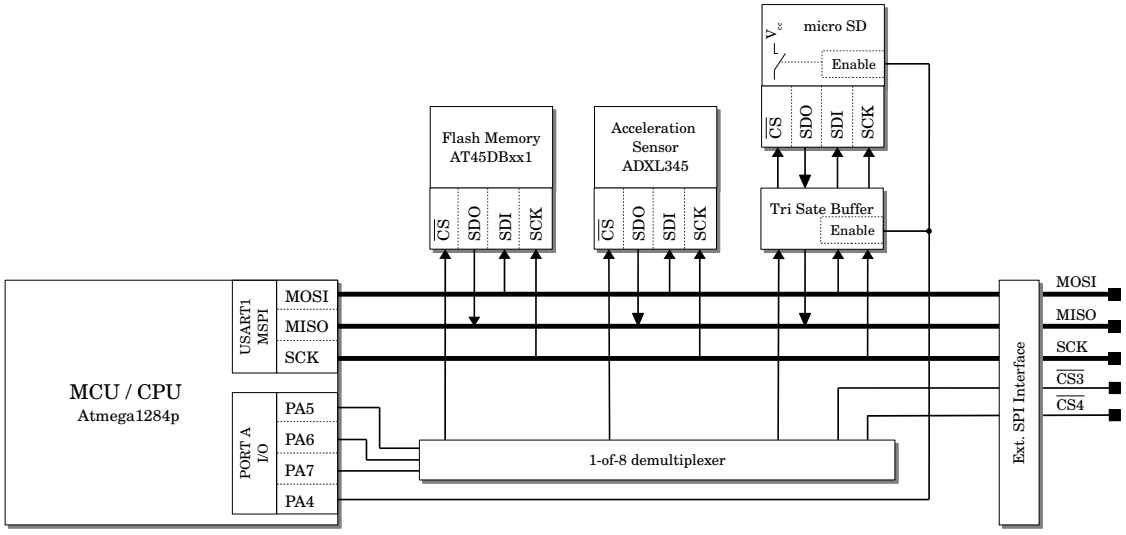


MSPI-Bus



- Eigener Hardware-SPI
- Multiplexer für Chip-Select
- Einfache Integration externer Komponenten

MSPI-Bus

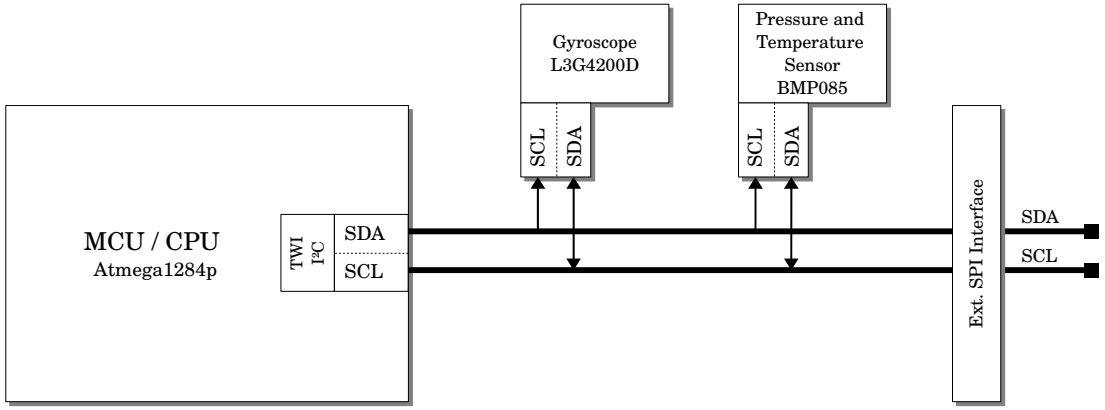


- Eigener Hardware-SPI
- Multiplexer für Chip-Select
- Einfache Integration externer Komponenten

Integrierte Komponenten

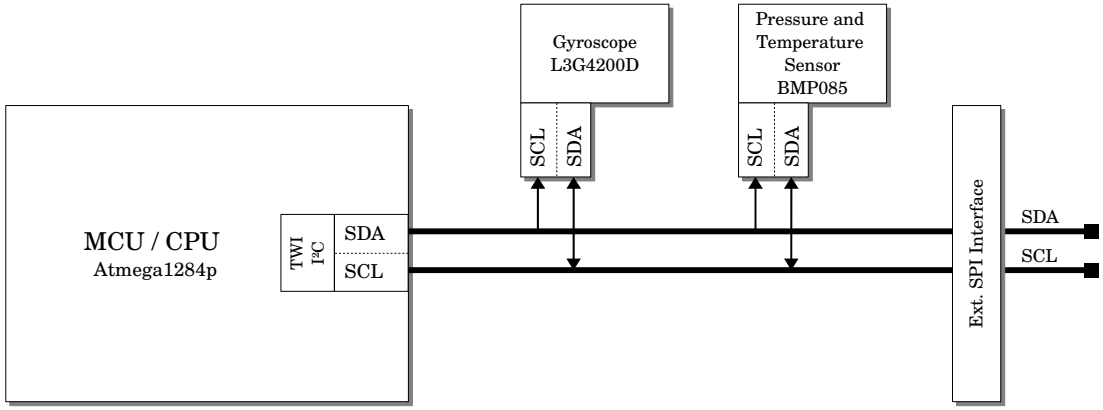
- 16Mbit Flash (Dual Buffer)
- ADXL345 Accelerometer (Rauscharm, sehr geringe Verlustleistung)
- microSD Slot (de-/aktivierbar)

I^2C-Bus



- Eigener Hardware-I^2C
- Einfache Integration externer Komponenten

I^2C-Bus

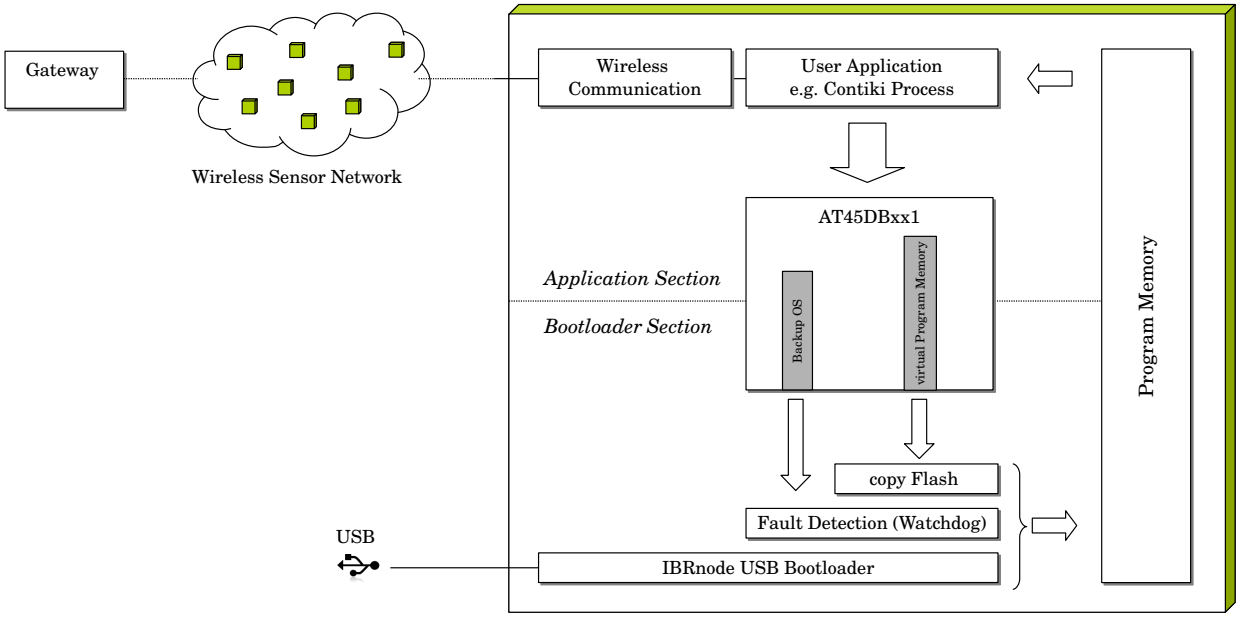


- Eigener Hardware-I^2C
- Einfache Integration externer Komponenten

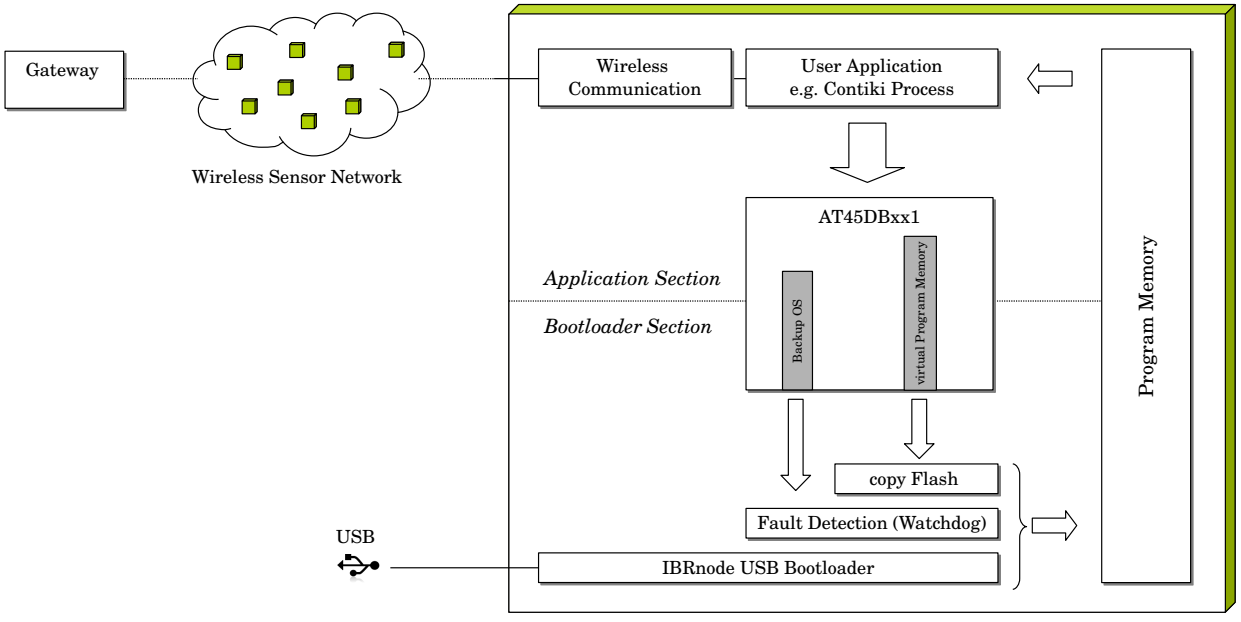
Integrierte Komponenten

- Gyroskop mit integriertem Temperatursensor
- Luftdrucksensor mit integriertem Temperatursensor

Bootloader


Gateway
Wireless Sensor Network
Wireless Communication
User Application e.g. Contiki Process
AT45DBxx1
Application Section
Bootloader Section
Backup OS
virtual Program Memory
Program Memory
copy Flash
Fault Detection (Watchdog)
USB
IBRnode USB Bootloader

Bootloader



- UART als Programmierschnittstelle
- Simples AVR109 Bootloader Protokoll (AVRDUDE kompatible)
- Future Work: Erweiterbar auf drahtloses Programmieren

Anpassung des RC Oszillatorfrequenz

- Ziel: Verringerung der Übertragungsfehler bei hohen Baud-Raten
- Iterative Näherung einer passenden RC-Frequenz (zwei Counter)
- Erzielter Speedup: 6

Übertragungsfehler:

$$UBBRn = \frac{f_{cpu}}{16 \cdot Baud} - 1 \tag{4}$$

$$\epsilon_u = \left(\frac{\lfloor UBBRn \rceil + 1}{UBBRn_{exakt}} - 1 \right) \cdot 100\% \tag{5}$$

Anpassung des RC Oszillatorfrequenz

- Ziel: Verringerung der Übertragungsfehler bei hohen Baud-Raten
- Iterative Näherung einer passenden RC-Frequenz (zwei Counter)
- Erzielter Speedup: 6

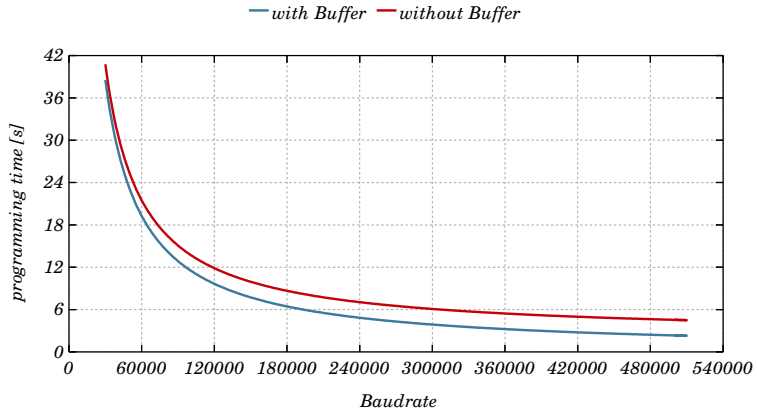
Übertragungsfehler:

$$UBBRn = \frac{f_{cpu}}{16 \cdot Baud} - 1 \quad (4)$$

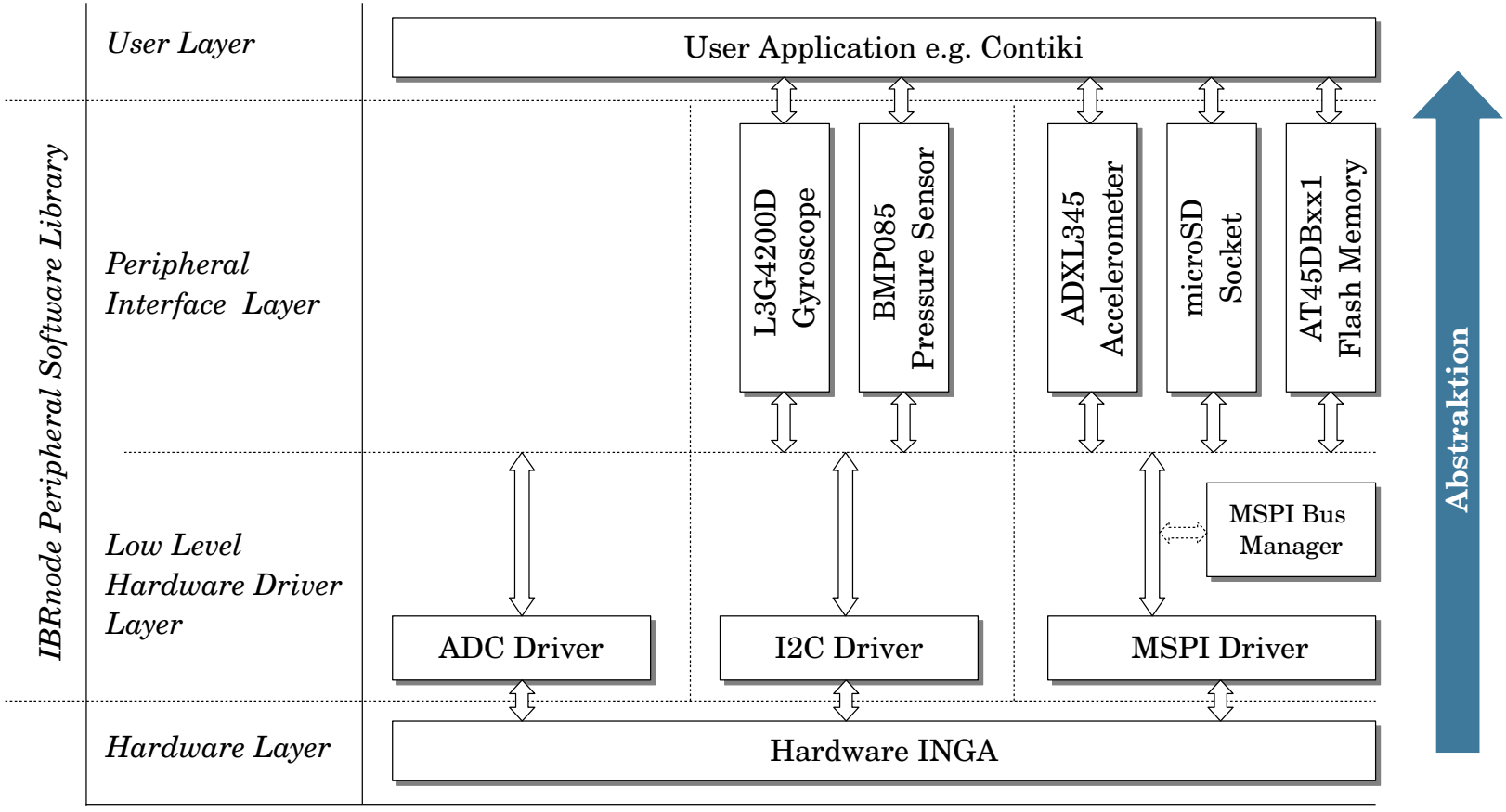
$$\epsilon_u = \left(\frac{\lfloor UBBRn \rceil + 1}{UBBRn_{exakt}} - 1 \right) \cdot 100\% \quad (5)$$

Parallelisierung der Flash-Schreiboperationen

- Prinzip: Zeit mit Platz substituieren
- Zusätzlicher Speicher verhindert Flash Delay
- Effekt bei hohen Übertragungsraten



Peripherie Software Bibliothek



MSPI-Bus Manager

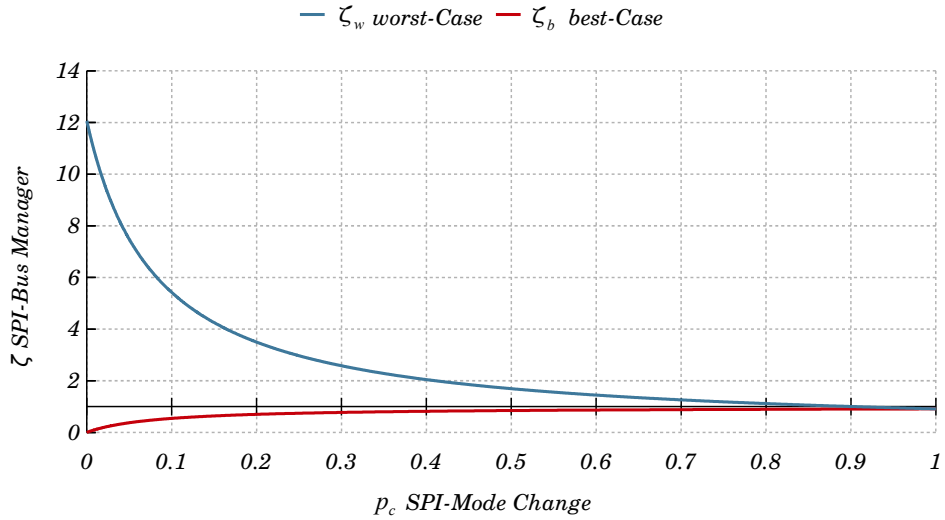
- 4 verschiedene SPI Modi
- Ziel: Abstraktion hardwarespezifischen Eigenschaften auf der obersten Software-Ebene
- MSPI-Bus Manager koordiniert den Mode-Wechsel
- Wahlweise de-aktivierbar

MSPI-Bus Manager

- 4 verschiedene SPI Modi
- Ziel: Abstraktion hardwarespezifischen Eigenschaften auf der obersten Software-Ebene
- MSPI-Bus Manager koordiniert den Mode-Wechsel
- Wahlweise de-aktivierbar

Theoretische Betrachtung

Basiert auf den Daten einer AVR Studio Simulation



Evaluation

Die Evaluation dieser Projektarbeit beschränkt sich auf folgende Punkte:

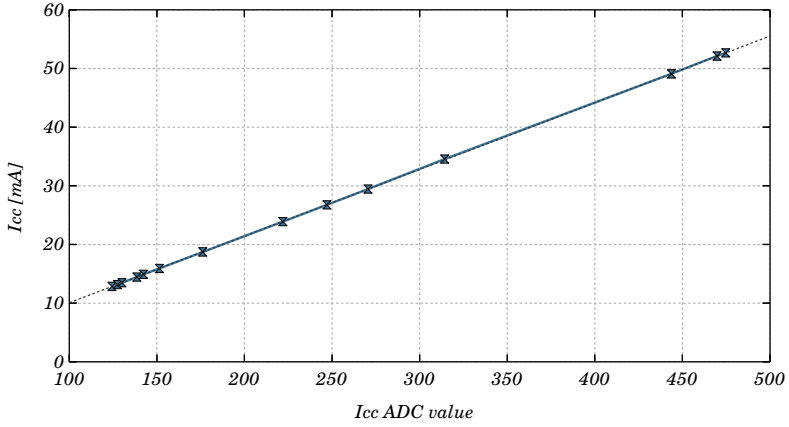
- Allgemeiner Funktionstest sämtlicher Komponenten
- Lese-/Schreibgeschwindigkeit der Speicher
- Kommunikationsreichweite
- Linearität der Strom- und Spannungsmessung

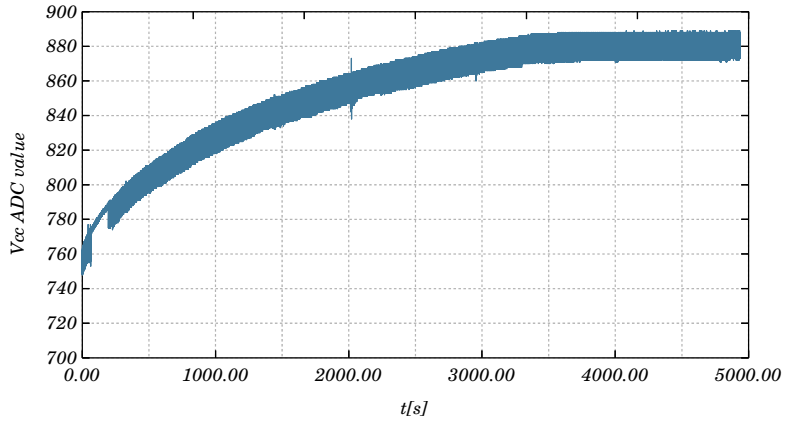
Evaluation

Die Evaluation dieser Projektarbeit beschränkt sich auf folgende Punkte:

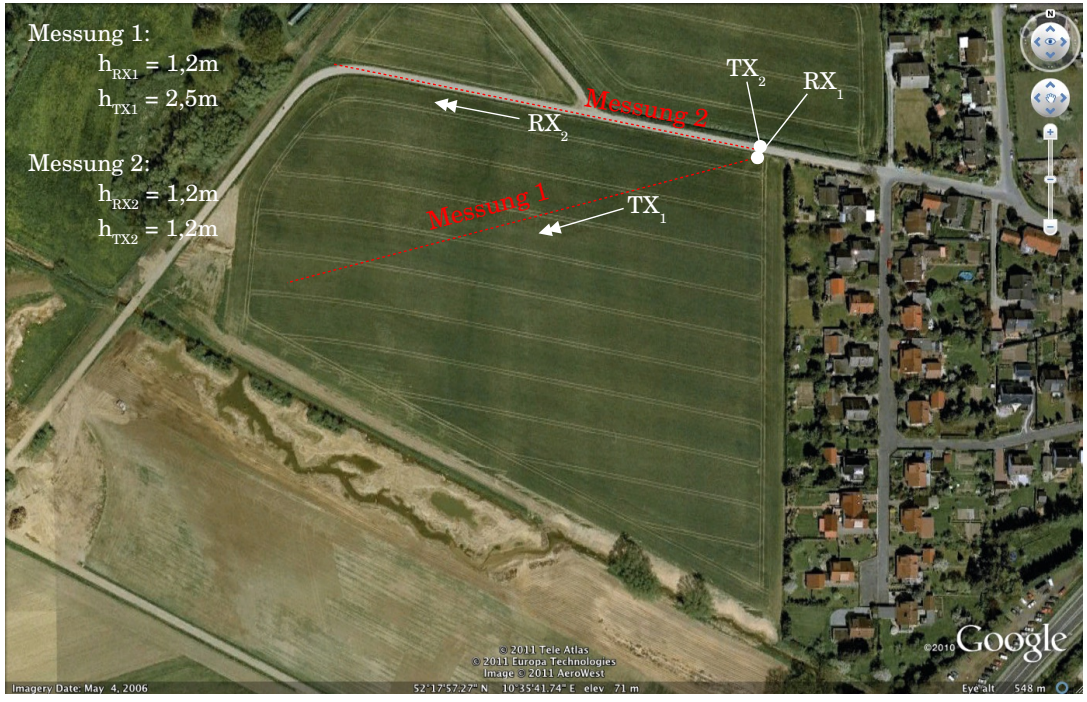
- Allgemeiner Funktionstest sämtlicher Komponenten
- Lese-/Schreibgeschwindigkeit der Speicher
- Kommunikationsreichweite
- Linearität der Strom- und Spannungsmessung

Linearität der Strom- und Spannungsmessung

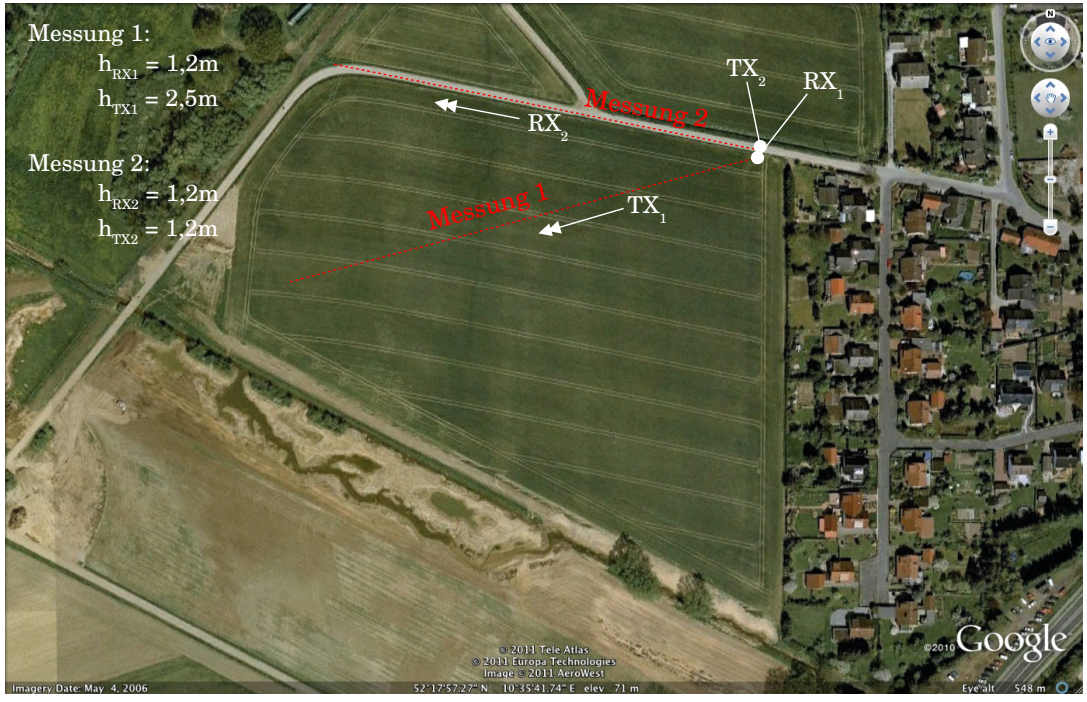




Kommunikationsreichweite: (Sensorknoten ⇒ USB Stick)

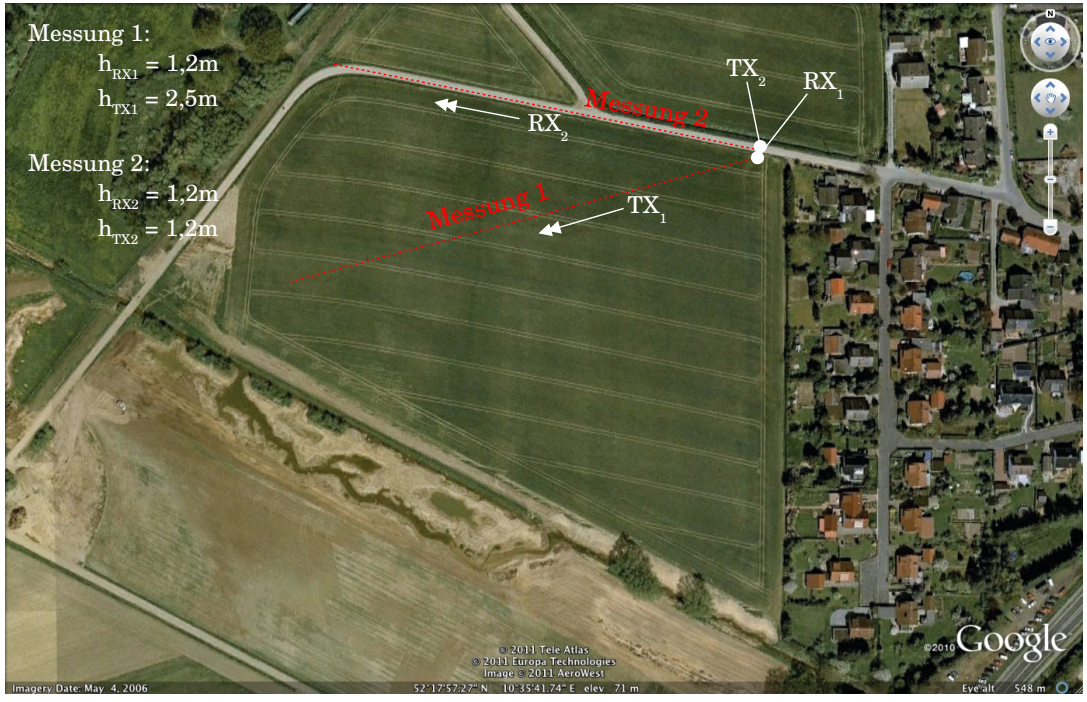


Kommunikationsreichweite: (Sensorknoten ⇒ USB Stick)



Node	$\overline{\phi_p(50m)}$[%]	$\overline{\phi_p(100m)}$[%]	$\overline{\phi_p(150m)}$[%]	$\overline{\phi_p(200m)}$[%]	$\overline{\phi_p(250m)}$[%]
INGA	0.0000	0.0000	0.0000	0.0000	0.0763
AVR Raven	0.4000	0.0380	0.1153	0.0000	0.8000

Kommunikationsreichweite: (Sensorknoten ⇒ USB Stick)



Node	$\overline{\phi_p(50m)}$[%]	$\overline{\phi_p(100m)}$[%]	$\overline{\phi_p(150m)}$[%]	$\overline{\phi_p(200m)}$[%]	$\overline{\phi_p(250m)}$[%]
INGA	0.0000	0.0000	0.0000	0.0000	0.0763
AVR Raven	0.4000	0.0380	0.1153	0.0000	0.8000

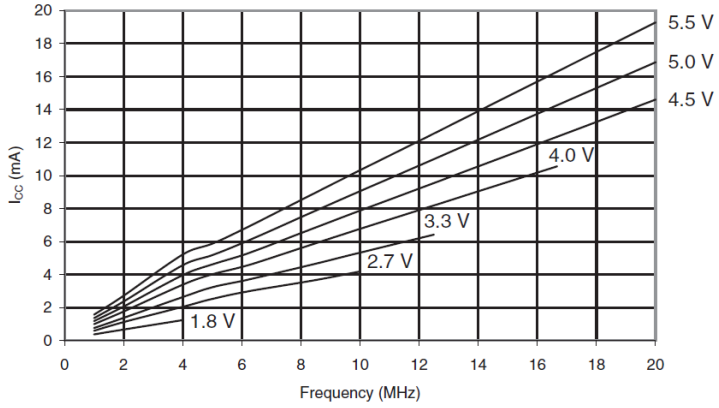
Node	$r[m]$
INGA	194
AVR Raven	219

Dynamische Anpassung der Versorgungsspannung

Zur Erinnerung:

$$P_{dyn} = C_l \cdot f \cdot U_b^2 \qquad (6)$$

Beispiel: ATmega1284p (active mode)

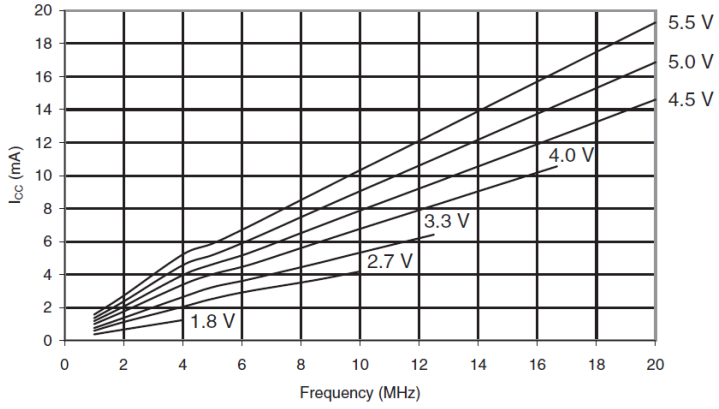


Dynamische Anpassung der Versorgungsspannung

Zur Erinnerung:

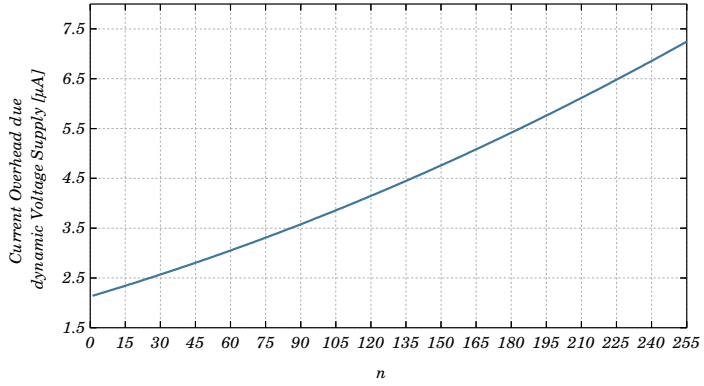
$$P_{dyn} = C_l \cdot f \cdot U_b^2 \qquad (6)$$

Beispiel: ATmega1284p (active mode)

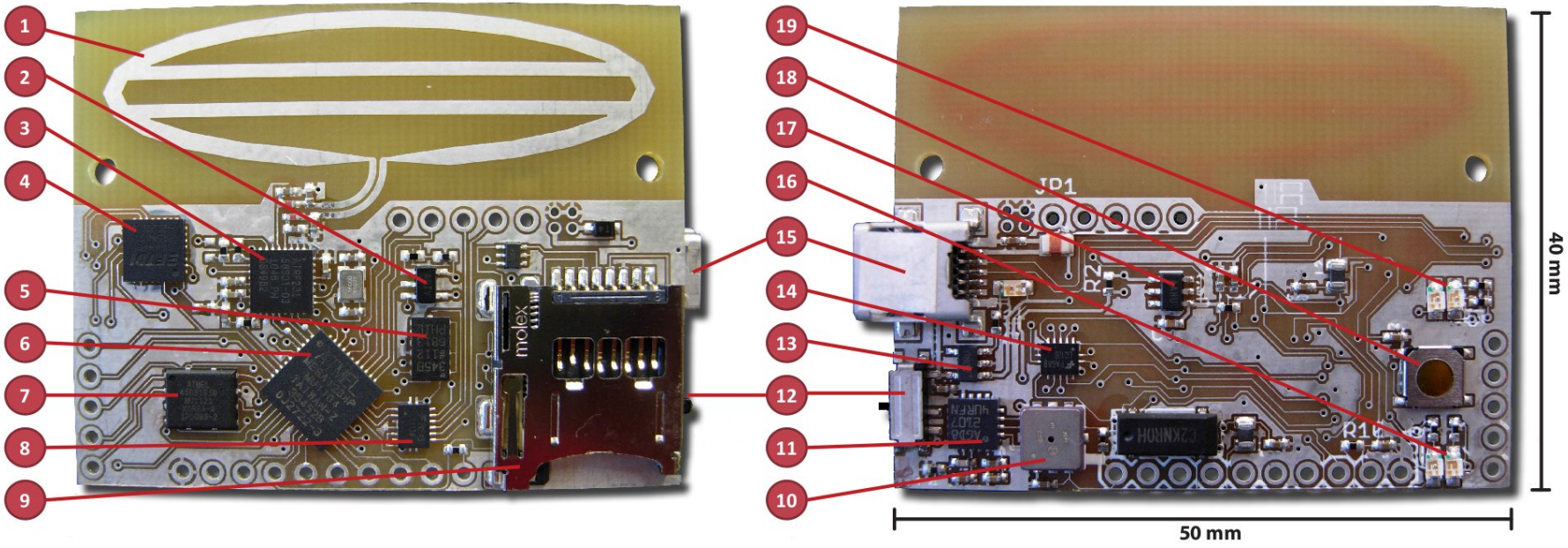


Lösungen:

- Bauteile gruppieren, fixe Spannungen
- Dynamische Spannungsanpassung
  ⇒ Simple Erweiterung von INGA
  ⇒ LDO mit Digitalpoti via i^2c
  ⇒ Online Optimierung



Vielen Dank für die Aufmerksamkeit



Ulf Kulau

kulau@ibr.cs.tu-bs.de