MICROTEC 2019 – 11. September 2019

Intelligente Wundpflaster

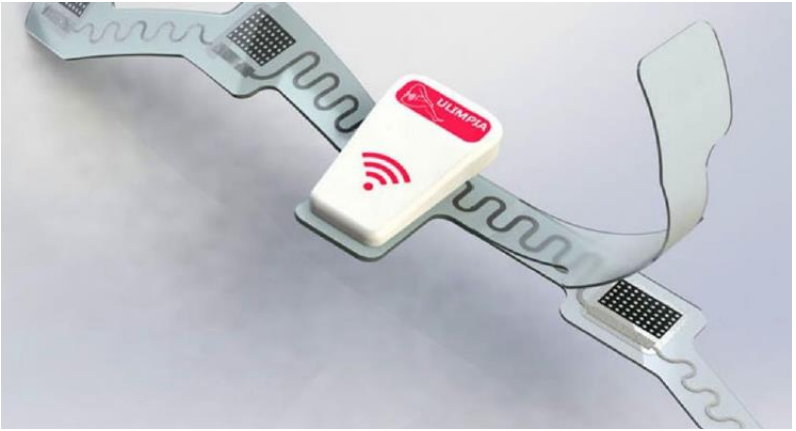
Algorithmenentwicklung in heterogenen Konsortien

Thomas Suwald

NXP Semiconductors Germany GmbH

BL Smart Transactions & Identity

Hamburg



ULIMPIA Smart Body Patches

EUREKA innovation across borders

Penta

Gefördert durch Bundesministerium für Bildung und Forschung

Themenauswahl

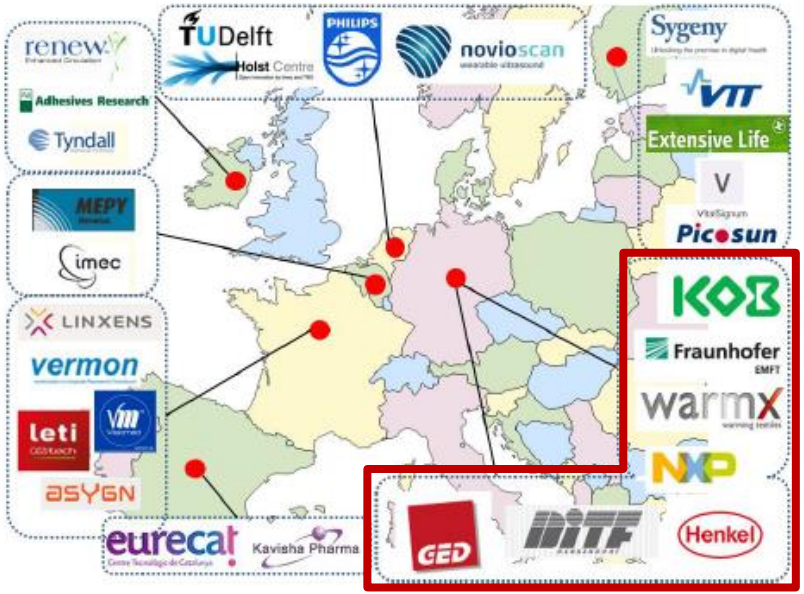
- Grundproblematik / Zielsetzung
- Systemanforderungen
- Aufbaukonzept
- Architektur
- BLE Controller KW41Z
- Arduino als Meta-Plattform
- Untersuchte Messverfahren
- Zusammenfassung

Grundproblematik

Hoher Wundpflegeaufwand

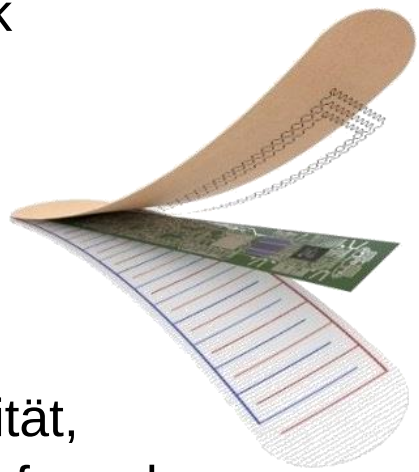
- Wundkontrolle in zu großen Zeitabständen
- Pflasterwechsel
  - zeitaufwendig
  - häufig prophylaktisch
- Zu lange Stationäraufenthalte

Zielsetzung (Deutscher Konsortialteil)

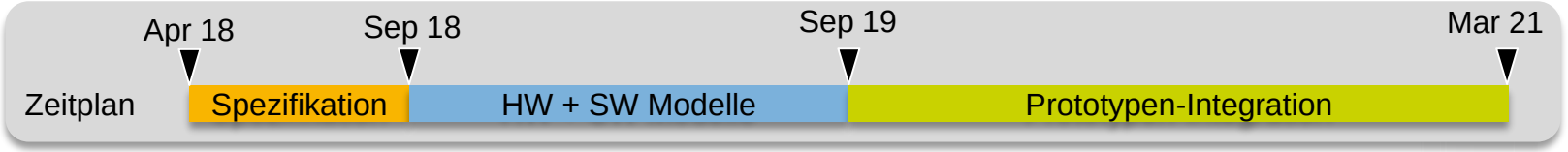


Wundpflaster mit integrierter Sensorik

- Temperatur
- Feuchtigkeit
- pH-Wert



Erwartung: bessere Überwachungsqualität, reduzierter Pflegeaufwand

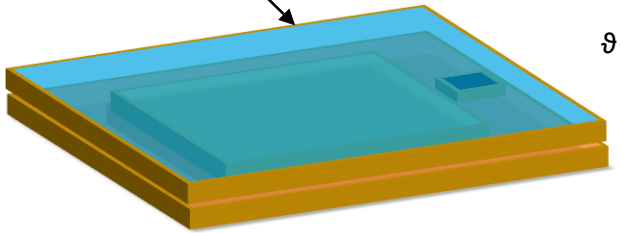


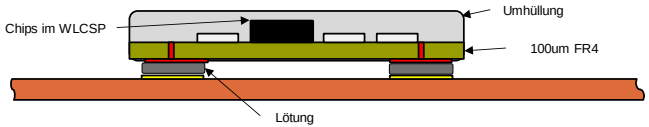
Systemanforderungen

Funktional	
Versorgung	3V Lithium Knopfzelle
Laufzeit	>1 Woche
Monitoring	ϑ 35-42°, Feuchte 0-100%, pH 3-9, Spannung
Messverfahren	Kapazitätsmessung, Widerstandsmessung, Spannungsmessung
Kommunikation	BLE
Aufbau	Substratrahmen

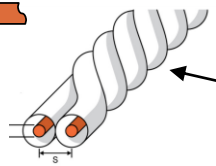
Nicht-Funktional	
Verkaufspreis	<10€
wiederverwendbar	nein
Entsorgung	umweltfreundlich
Bedienungsaufwand	gering
Tragekomfort	Nicht drückend Nicht auftragend
Warnfunktionen	Temperatur/ Feuchtigkeits-Profil
Größe	100mm x 100mm

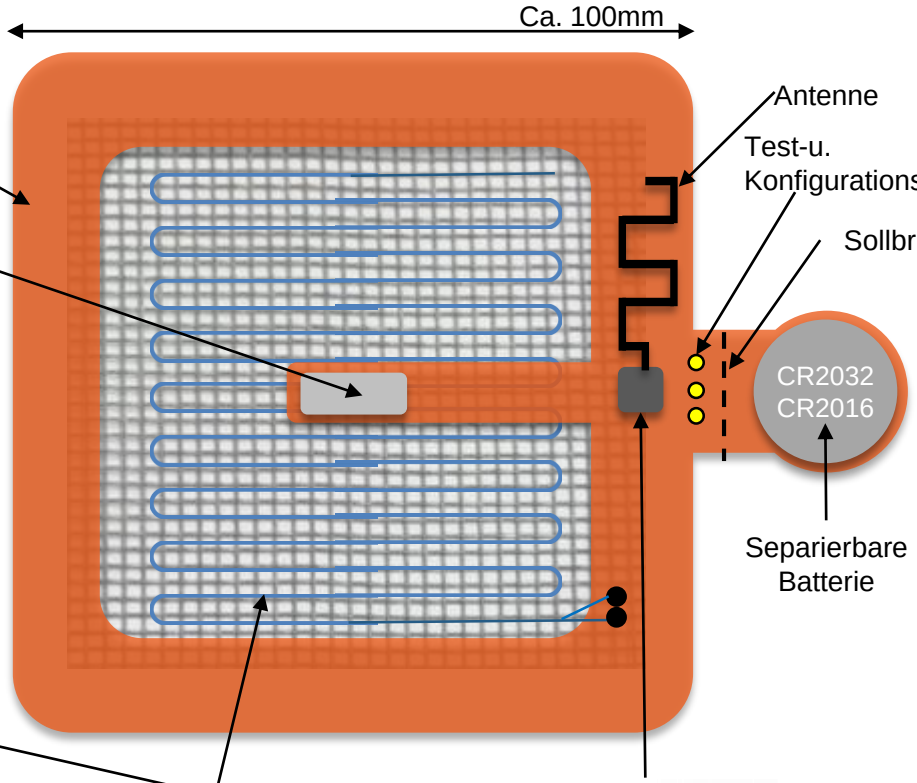
Aufbaukonzept



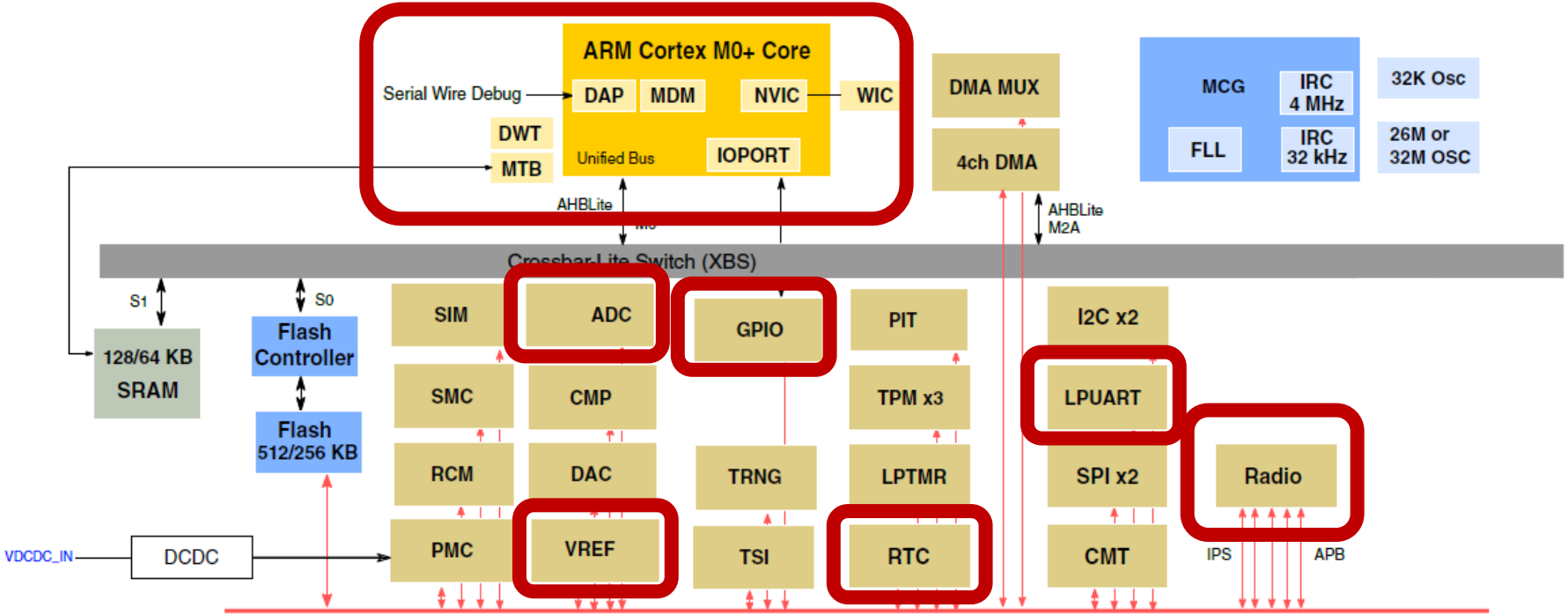


BLE Controller Modul



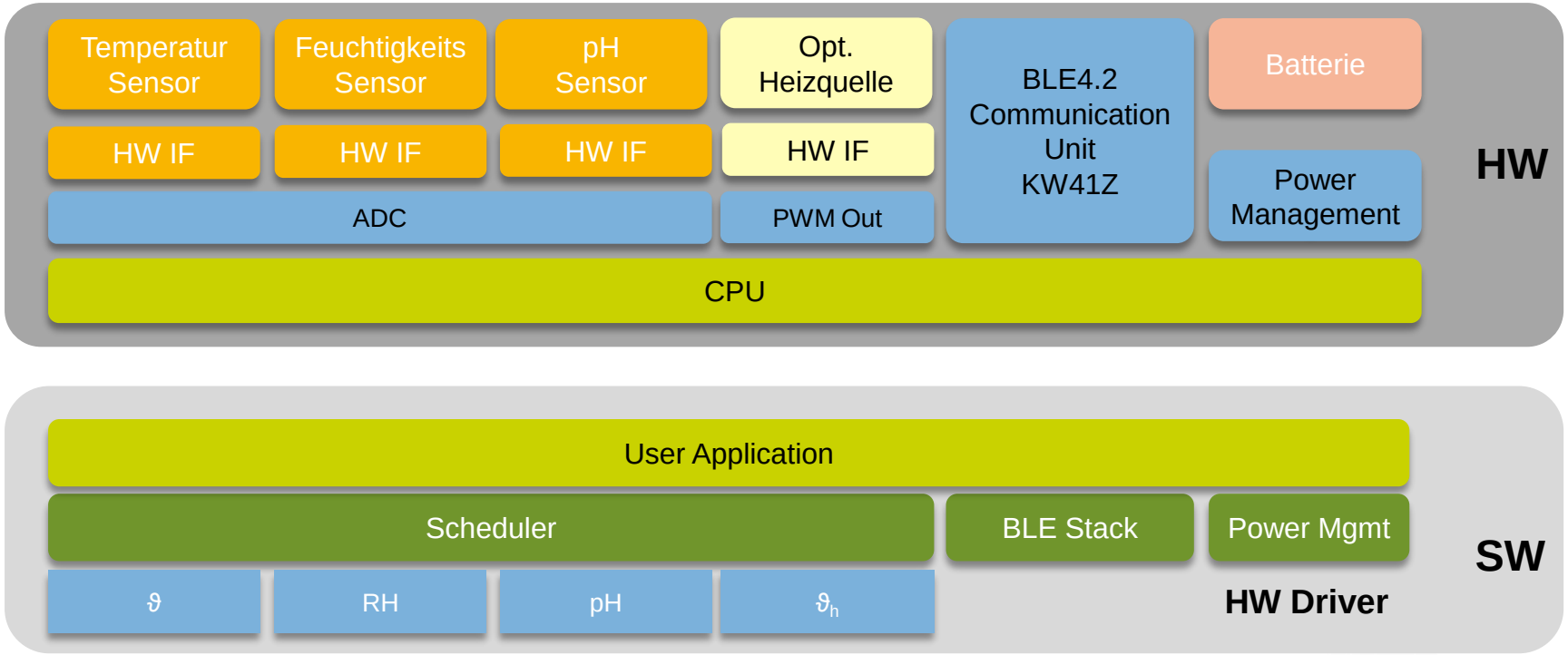


Ziel-Plattform BLE Controller KW41Z



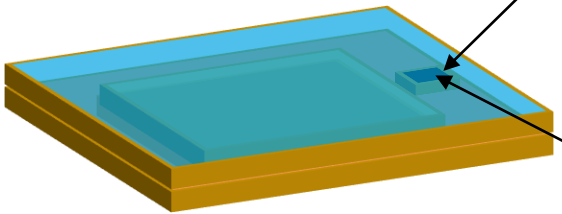
https://www.nxp.com/docs/en/data-sheet/MKW41Z512.pdf

Architektur

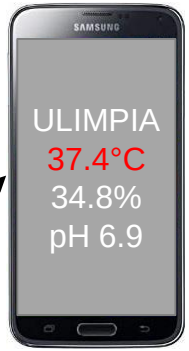


Systemanbindung

ULIMPIA
Wundpflaster



BLE



Patient

Warnfunktion

- Warnung bei Grenzwert-überschreitung
- Vorgabe von Temperatur /Feuchtigkeits-Profilen

BLE-to-LAN
Bridge

PoE LAN
(Klinik-LAN)

Stations-Monitor

Besonderheit: Heterogenes Entwicklungs-Konsortium

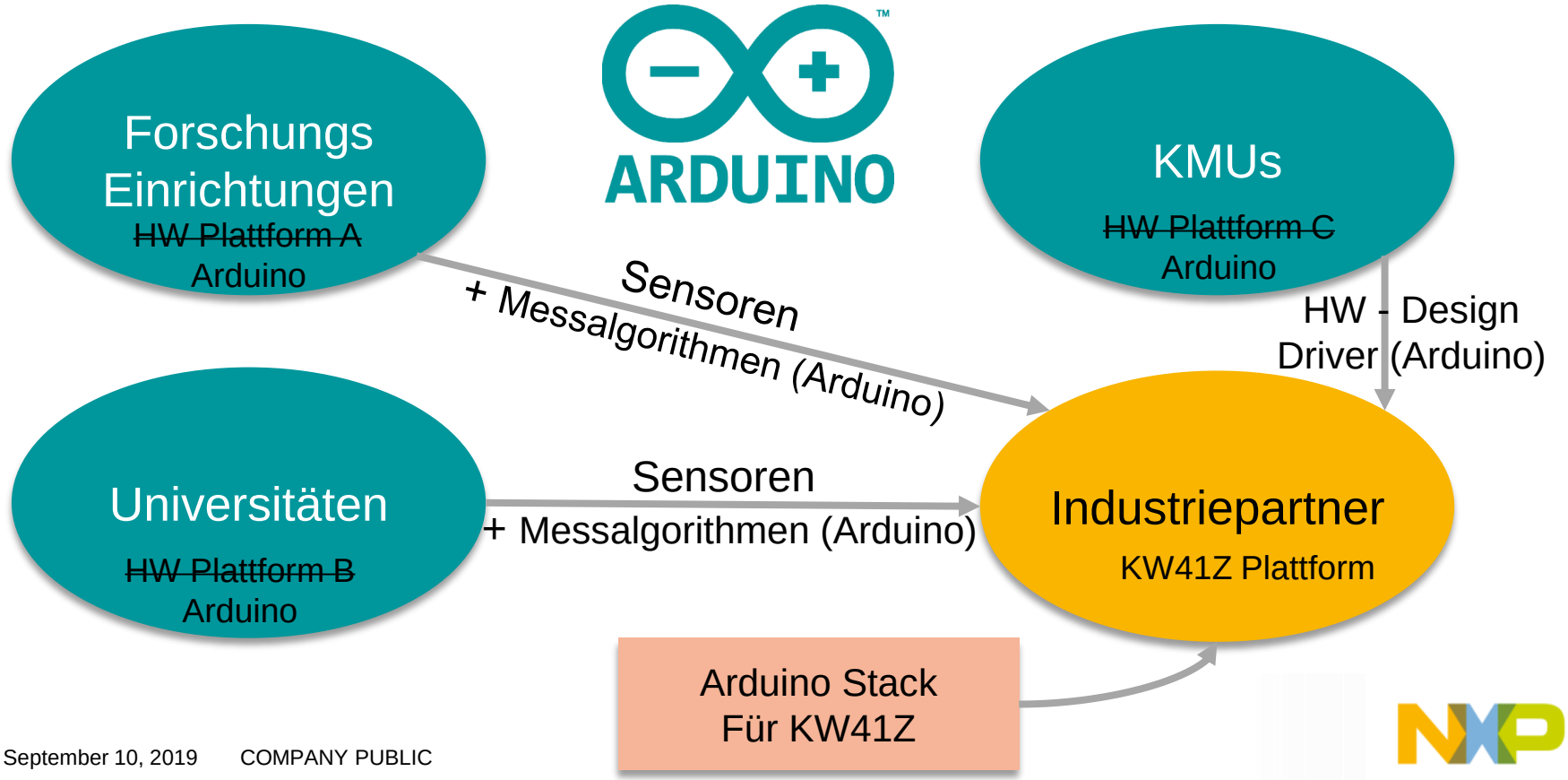
Herausforderung:

Einarbeitung in Ziel-Plattform teilweise sehr aufwendig

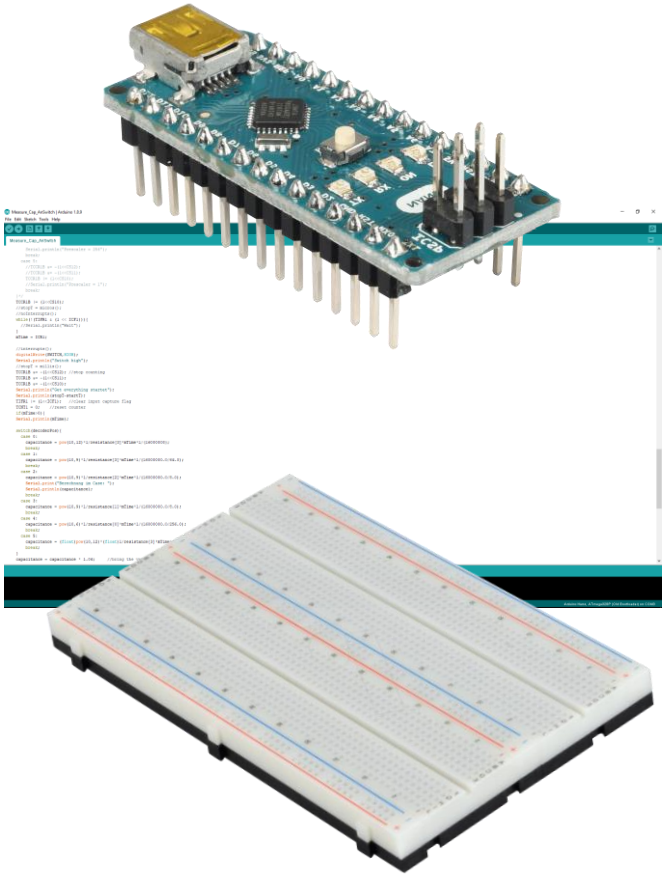
- Hoher Einarbeitungsaufwand
- Hoher Supportaufwand
- mit Kosten verbunden

Ideen für die Algorithmen sind vorhanden, deren Realisierung ist aber komplex

Lösung: Arduino als Meta-Plattform



Vorteile der Arduino Meta-Plattform



Arduino Nano

- Ausreichende Leistung
- Einfache (Hardware-)Programmierung
- 8-Bit Register erleichtern Einstieg

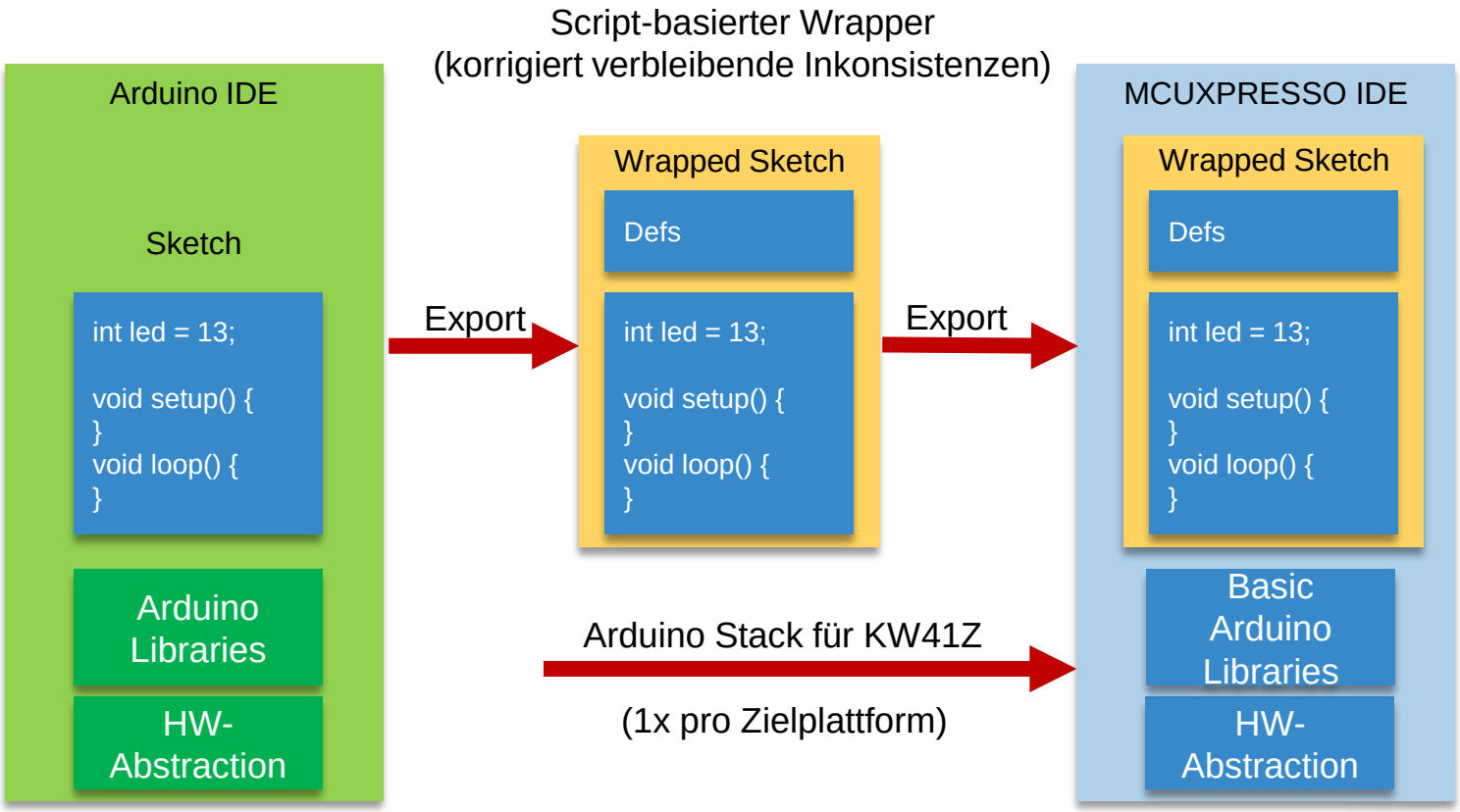
Datenzugriff per UART-Schnittstelle

- Einfaches Debuggen von Algorithmen

Werkzeugloses Arbeiten

- Per Steckbrettsystem

Portierung von Meta-Plattform auf Zielplattform



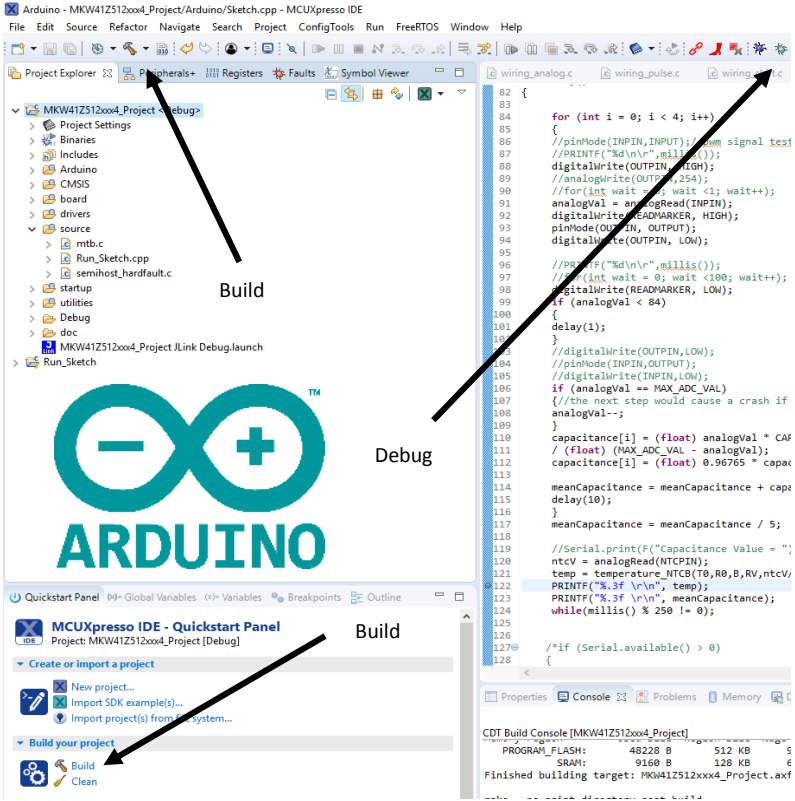
Verzicht auf Integration in Arduino IDE, Kommunikationsroutinen nicht übertragbar

~50 unterstützte Arduino API-Funktionen für KW41Z

pinMode()	pulseInLong()	map()	isAlphaNumeric()	isUpperCase()	lowByte()
digitalWrite()	shiftIn()	max()	isAscii()	isWhiteSpace()	attachInterrupt()*
digitalRead()	shiftOut()	min()	isControl()	random()	detachInterrupt()
analogRead()	~~Tone()~~	pow()	isDigit()	randomSeed()	interrupts()
analogWrite()	delay()	sq()	isGraph()	bit()	noInterrupts()
analogReference()	delayMicroseconds()	sqrt()	isHexadecimalDigit()	bitClear()	~~Serial~~
analaogReadResolution()	micros()	cos()	isLowerCase()	bitRead()	~~Stream~~
~~analogWriteResolution()~~	millis()	sin()	isPrintable()	bitSet()	~~Keyboard~~
~~noTone()~~	abs()	tan()	isPunct()	bitWrite()	~~Mouse~~
~~pulseIn()~~	constrain()	isAlpha()	isSpace()	bighByte()	

* funktioniert mit allen digitalen Pins.

Transfer Ablauf



Entwicklung auf Arduino

↓

Quellcode (Sketch) exportieren

↓

Skript-basierter Transfer nach C++

↓

Projekt unter Ziel-IDE erstellen

↓

Quellcode unter Ziel-IDE einfügen

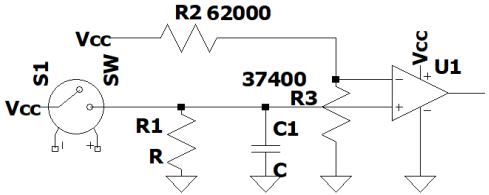
↓

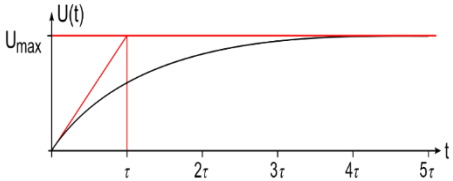
Projekt debuggen

Transfer Erfahrungen

- Sehr schnell vorweisbare Ergebnisse
- Unkompliziertes Testen unter Arduino
- Support durch große Arduino-Community
- Grundkenntnisse der Ziel-IDE hilfreich
- Peripherie der Ziel-MCUs problemlos ansteuerbar
- Ziel-Debugger vereinfacht Problemsuche
- **Nur geringe händische Anpassung erforderlich**
- Ersparte Zeit kann in den Kreativteil der Entwicklung investiert werden

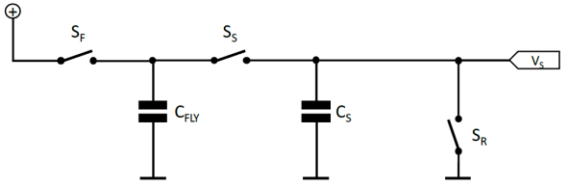
Untersuchte Kapazitätsmessverfahren (Arduino)





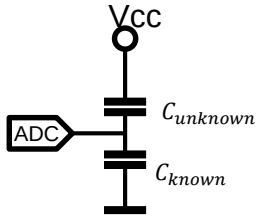
RC-Integrator

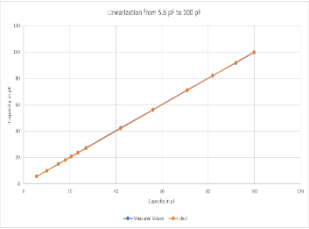
C-Messung über Integrationszeit



Switched-Cap-Integrator

C-Messung über Ladezyklenzahl





Kapazitiver Spannungsteiler

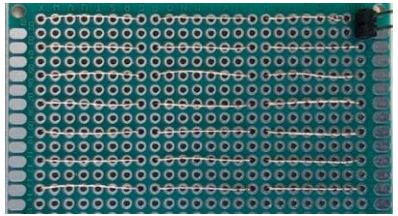
C-Messung über Spannungsmessung

Vergleich Kapazitätsmessverfahren

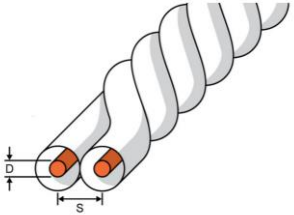
	1. RC-Integrator	2. Switched-Cap Integrator	3. Spannungsteiler
Komponenten	1… 3 externe R's	1 externer C	1 externer C
Linearität	Linear	Linearisierung per SW	Linearisierung per SW
Komplexität	mittel	hoch	Gering
Algorithmus	einfach	komplex	einfach
µC Komponenten	Timer + Komparator	GPIO + Komparator	ADC + GPIO
Messbereich/Auflösung	≈10pF – ….	≈25pF - ….	1pF - …..
Genauigkeit	Genauigkeit abhängig von der R/C Kombination Befriedigend bis Gut	Genauigkeit abhängig von der C/C Kombination Nicht für kleine C Befriedigend bis Gut	Genauigkeit abhängig vom ADC und dem Teilerverhältnis, Gut bis Sehr Gut
Bewertung	++	+	+++

Spannungsteiler: Höchste Genauigkeit, verlässliche Ergebnisse, einfache Implementierung

Kapazitive Sensoren



Feuchtesensor Prototyp (Arduino)

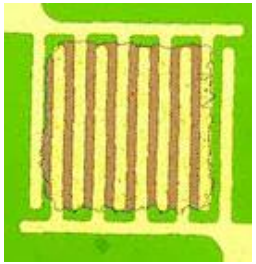


Feuchtigkeit

Verdrillter Kupferlackdraht (100 µm verwebbar)

Gesamtkapazität ≈420pF bei RH*=0%

Foto: Fraunhofer EMFT München



pH

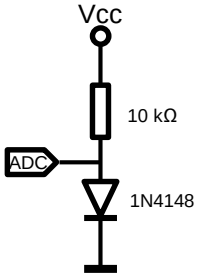
IDK*-Struktur mit pH sensitivem Gel

pH-abhängige Feuchtigkeitsaufnahme

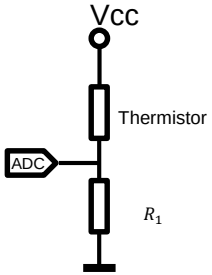
feuchtigkeitsabhängige Kapazitätsänderung

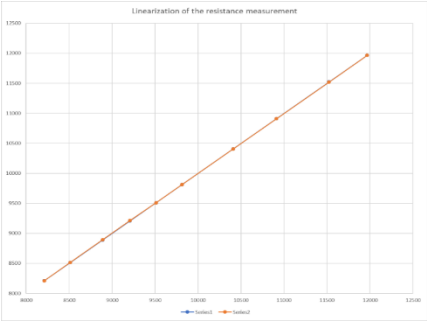
*IDK = Interdigitalkondensator *RH=Relative Humidity

Untersuchte Temperaturmessverfahren (Arduino)



Messung **Diodenschwellspannung**





Spannungsteiler mit **Thermistor** (NTC)

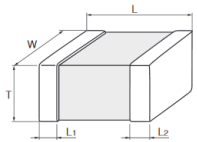
→ Klassische Fieberthermometermessung

Vergleich Temperaturmessverfahren

	1. Diode Threshold	2. Thermistor
Komponenten	1 R, 1 D	1 R, 1 NTC
Linearität	Linearisierung per SW	Linearisierung per SW
Komplexität	mittel	gering
Algorithmus	Kalibrierung erforderlich	Einfache Berechnung
µC Komponenten	ADC + GPIO	ADC + GPIO
Messbereich/Auflösung	0.1° C Auflösung kaum erreichbar	0.1° C Auflösung sicher erreichbar
Genauigkeit	Genauigkeit abhängig vom ADC und der Diode	Genauigkeit abhängig vom ADC und dem Teilerverhältnis
Bewertung	+	+++

Multilayer NTC Thermistors

Dimensions in mm (not to scale)



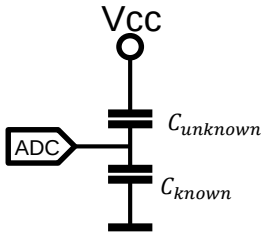
(Unit : mm)

Size Code (EIA)	L	W	T	L_1, L_2
Z(0201)	0.60±0.03	0.30±0.03	0.30±0.03	0.15±0.05
0(0402)	1.0±0.1	0.50±0.05	0.50±0.05	0.25±0.15
1(0603)	1.60±0.15	0.8±0.1	0.8±0.1	0.3±0.2

Thermistor: Hohe Genauigkeit, verlässliche Ergebnisse, einfache Implementierung, geringe Kosten

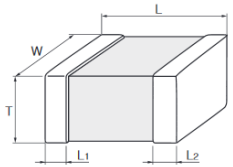
Zusammenfassung

Algorithmenentwicklung auf Arduino + Transfer auf Ziel-Plattform ist sinnvoll -- schafft mehr Zeit für kreative Entwicklung



Kapazitive Spannungsteilung am effektivsten für Feuchtigkeitssensor und pH-Sensor

Spannungsteiler mit Thermistor am effektivsten für Temperaturmessung



Das vorgestellte Monitoring-System fügt sich in bestehende Krankenhaus-Infrastrukturen ein

Vielen Dank für Ihre Aufmerksamkeit

Haben Sie Fragen ?

SECURE CONNECTIONS
FOR A SMARTER WORLD