

Technologie-Evaluierung von Drehgebern und deren Anbindung an eine Java-basierte Potiboard-App

Table of Contents

| | |
|---|----|
| Einbindung dieser Arbeit in das laufende projekt FCC Digital Potiboard | 2 |
| Rückblick und Motivation | 2 |
| Skizze des neuen Potiboard-Encoder-Device | 3 |
| Vom alten Potiboard-Prototyp übernommende Technologieentscheidungen und neue Wege | 4 |
| Generelle Hardware-Komponenten-Architekturüberlegungen | 4 |
| Komponentengruppen | 4 |
| Inkrementzähler-Übertragung über Netzwerk oder USB | 5 |
| Netzwerkbasierte System-Architekturen und Technologien | 6 |
| Netzwerkbasiertes Testsystem mit der Technologie Spring Webflux | 6 |
| Netzwerkbasierte Testsysteme mit den Technologien ZeroMQ und MQTT | 8 |
| USB-basierte System-Architekturen und Technologien | 8 |
| Testsystem mit der Technologie USB-Serial | 8 |
| Testsystem mit der Technologie USB-HID | 8 |
| Testsystem mit der Technologie USB-MIDI | 8 |
| Grundlegende zentrale Anforderungen (Stichwortliste) | 9 |
| Tabelle Technologiebewertung | 10 |
| Fazit | 10 |
| Anex | 11 |

Einbindung dieser Arbeit in das laufende projekt FCC Digital Potiboard

Die hier beschriebene Technologie-Evaluierung dient als Grundlage zur Entscheidungsfindung zu den in der [Abbildung: Status FCC Digital Potiboard](#) markierten *Major Milestones* und *Subprojects / Tasks*.



| Project | FCC Digital Potiboard | | | | | | | | | | | | | | | | | | | | | |
|--|---|--|----------------------|---------------------------|------|---|---|------|---|----|------|------------------------|---|------|------------------|---|------|---|---|------------|--------------------|-------------------------|
| Project Lead | S. Reimann | | | | | | | | | | | | | | | | | | | | | |
| Status Date | 08.07.2022 | | | | | | | | | | | | | | | | | | | | | |
| Project Description | | | | | | | | | | | | | | | | | | | | | | |
| Development of a digital potentiometer board control for FCC - Focus: UNILAC operation - But also generally as an option for other linear accelerators or any parameters, where a rotating controller makes sense. The project includes the incremental encoder hardware and the software application. | Subprojects / Tasks <ul style="list-style-type: none"> • Specify the system ✓ • Specification approval -> open • technology decision for incremental encoder -> in progress ↗ • technology decision software stack -> in progress (FESA, LSA or else) • general decision on UI -> open (existing or new app + developing group) ↗ • build hardware prototype -> open • build software prototype -> open • FAT → serial production -> open | | | | | | | | | | | | | | | | | | | | | |
| Project Goals | | | | | | | | | | | | | | | | | | | | | | |
| 1. Replace the outdated UNILAC Potiboard control with a modern version 2. compatible with fully digital control room (FCC) 3. Full control system integration 4. Be ready on time for the move to FCC 5. Production of relevant spare parts | Ressource Profile <table border="1"> <thead> <tr> <th>Year</th> <th>Estimated costs [k€]</th> <th>Personnel [person months]</th> </tr> </thead> <tbody> <tr> <td>2022</td> <td>1</td> <td>2</td> </tr> <tr> <td>2023</td> <td>4</td> <td>12</td> </tr> <tr> <td>2024</td> <td>15 (initial equipment)</td> <td>3</td> </tr> <tr> <td>2025</td> <td>10 (spare parts)</td> <td>3</td> </tr> <tr> <td>2026</td> <td>0</td> <td>0</td> </tr> <tr> <td>Sum</td> <td>30.000 Euro</td> <td>20 person months</td> </tr> </tbody> </table> | Year | Estimated costs [k€] | Personnel [person months] | 2022 | 1 | 2 | 2023 | 4 | 12 | 2024 | 15 (initial equipment) | 3 | 2025 | 10 (spare parts) | 3 | 2026 | 0 | 0 | Sum | 30.000 Euro | 20 person months |
| Year | Estimated costs [k€] | Personnel [person months] | | | | | | | | | | | | | | | | | | | | |
| 2022 | 1 | 2 | | | | | | | | | | | | | | | | | | | | |
| 2023 | 4 | 12 | | | | | | | | | | | | | | | | | | | | |
| 2024 | 15 (initial equipment) | 3 | | | | | | | | | | | | | | | | | | | | |
| 2025 | 10 (spare parts) | 3 | | | | | | | | | | | | | | | | | | | | |
| 2026 | 0 | 0 | | | | | | | | | | | | | | | | | | | | |
| Sum | 30.000 Euro | 20 person months | | | | | | | | | | | | | | | | | | | | |
| Major Milestones | | | | | | | | | | | | | | | | | | | | | | |
| Q3/2022 | Specification revision and approval | | | | | | | | | | | | | | | | | | | | | |
| Q4/2022 | technology decision - incremental encoder hardware and communication protocol | | | | | | | | | | | | | | | | | | | | | |
| Q4/2022 | decision regarding use of control system stack (FESA or LSA or else) to be able to ensure the required performance | | | | | | | | | | | | | | | | | | | | | |
| Q3/2023 | Hardware prototype ready | | | | | | | | | | | | | | | | | | | | | |
| Q1/2024 | Software prototype ready | | | | | | | | | | | | | | | | | | | | | |
| Q3/2024 | Live test in HKR (e.g. for HEST magnets) | | | | | | | | | | | | | | | | | | | | | |
| Risks, Boundary Conditions and Comments | | | | | | | | | | | | | | | | | | | | | | |
| Major Risks: | <ul style="list-style-type: none"> - turnaround time too slow for adequate UNILAC control | Concerned departments: <ul style="list-style-type: none"> - ACO (FE, AP, IN) - OPE (APS) | | | | | | | | | | | | | | | | | | | | |
| Boundary Condition: | <ul style="list-style-type: none"> - must be functional before move from HKR to FCC - existing potiboard must kept functional until move to FCC is complete | Comments: <ul style="list-style-type: none"> - to be clarified: Responsible group for hardware maintenance | | | | | | | | | | | | | | | | | | | | |

Figure 1. Status FCC Digital Potiboard

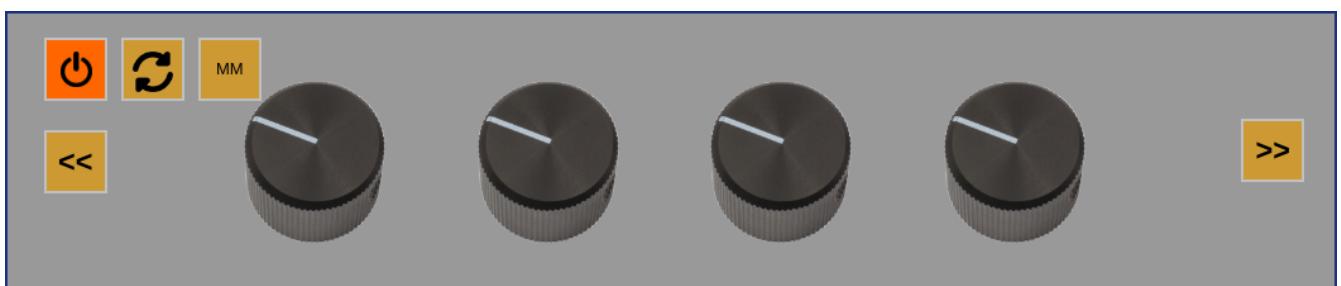
Rückblick und Motivation

Der vor ca. 6 Jahren entwickelte Potiboard-Prototyp wurde in einem der vergangenen Potiboard-Meetings als unzureichend in seiner Technologieauswahl eingestuft. Insbesondere der Einsatz einer "closed source library" des benutzten Phidget-Mikrocontroller, der zur Verarbeitung der Encodersignale dient, wurde kritisiert. Dessen Treiber läuft zudem nicht im "Userspace" auf dem zu benutzenden Linuxderivat, was aus system-administrativer Sicht ein Nachteil ist.

Eine weitere Anforderung der Zukunft (FCC) **könnte sein**, dass die Drehgeber ihre Inkremente an eine Potiboard-App über eine nicht unerhebliche Entfernung übermittelt werden müssen. Diese Anforderung wurde beim [alten Potiboard-Prototyp](#) über eine hohe Integration der beteiligten Komponenten gelöst. Drehgeber wie Potiboard-App, gesteuert über einen Touchscreen, befanden sich in einem Gehäuse mit Netzwerkanschluss.



Skizze des neuen Potiboard-Encoder-Device



Funktionsüberblick aus der Spezifikation

- Button functions are:
 - scroll back and forth (per button press by 4 devices to the right or left.
In case of individual assignment, move one device to the right or left)
 - deactivate the encoders (deactivation deselects all devices and triggers the LSA data supply)
 - change the parameter view ⇒ volts, BRHO et cetera
 - activate and deactivate the master mode (first partner knob controls both devices, second (third, fourth) partner is ignored)
 - change the increment

— Spezifikation: F-FO-CMD-en-0009_DS_Potiboard_v4_inprogress.docx

Vom alten Potiboard-Prototyp übernommende Technologieentscheidungen und neue Wege

- Optische "Rotary Quadrature Encoder" wurden wieder wegen ihrer Signalqualität, Zuverlässigkeit und Verfügbarkeit eingesetzt. Auf kugelgelagerte Modelle wurde diesmal verzichtet (Haptikgründe wegen zu hoher Leichtgängigkeit). Merkmale sind 16-128 Pulse pro 360-Rotation, keine Zahnung, 5 V. 3.3V Modelle waren auf dem Markt nicht erhältlich.
 - Beispiel-Encoder sind:
 - Grayhill 63R128, Stückpreis: 100 \$
 - Bourns ENA1J-B28-L00128L, Stückpreis: 60 \$

Durch den Einsatz von modernen Mikrocontrollern, deren Spannung an ihren I/O Kanälen häufig auf 3,3 V limitiert ist (anstatt 5V), schränkt sich die Auswahl der möglichen Endcoder-Modelle deutlich ein. Eventuell müssten die Encoder-Ausgangsspannungen an den Eingängen der Mikrocontroller mit Pegelumsetzern (Level-Shifter) angepasst werden, wenn 5 V Encoder-Modelle eingesetzt werden müssen.

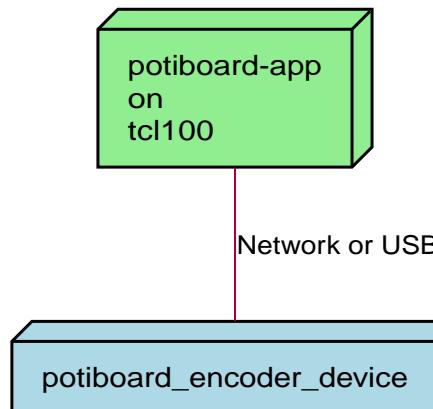
- Es wurden wieder Mikrocontroller evaluiert, die die Inkremente der bis zu acht (8!) Encoder, ohne spürbare Zeitverzögerung, weiterverarbeiten können sollen. Statt des im alten Prototypen verwendeten Phidget-Mikrocontroller (1047) wurden folgende Mikrocontroller stattdessen betrachtet:
 - Raspberry Pi 4, Stückpreis: 70 \$
 - STM32H7, STM32F7, Stückpreis: 70 \$
 - Teensy 4.1 (Arduino kompatibel), Stückpreis: 40 \$
 - Raspberry Pi Pico, Stückpreis: 8 \$

Alle Systeme stellen nachbaubare Hardware dar (Ersatzteilversorgung scheint gesichert) und lassen sich mit Open-Source Software betreiben. Pro Einheit bewegen sie sich in einem Kostenrahmen von 6-80 US \$. Der Mikrocontroller-Code zur Weiterverarbeitung der Encoder-Inkremente muss bzw. musste in C, Python oder Assembler geschrieben und gewartet werden. Das gleiche gilt für die verschiedenen Übertragungstechnologien zur Java-basierten Potiboard-Applikation.

Generelle Hardware-Komponenten-Architekturüberlegungen

Komponentengruppen

General topology for connecting a potiboard encoder device

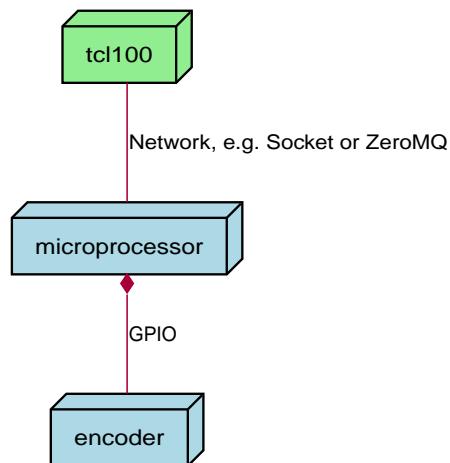


General Deployment Diagram, A. Bloch-Späth, M. Stein

Inkrementzähler-Übertragung über Netzwerk oder USB

Network based encoder counter transmission

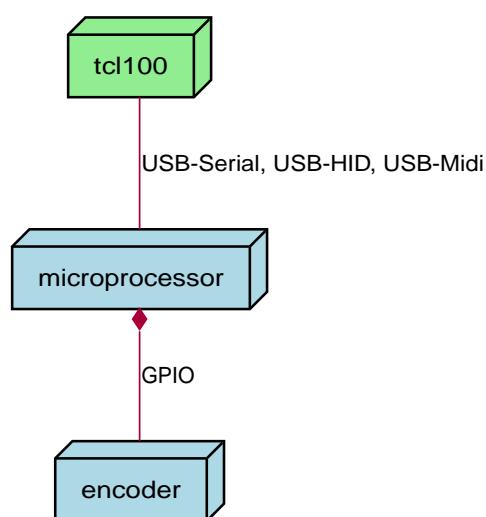
GSI/FAIR 03.11.2022



Deployment Diagram Potboard Network, A. Bloch-Späth, M. Stein

USB based encoder counter transmission

GSI/FAIR 03.11.2022



Deployment Diagram Potboard USB, A. Bloch-Späth, M. Stein

USB Nachteil



USB ist auf eine **maximale Kabellänge** von 5m spezifiziert. Mit guten Kabel und/oder Repeatern sind vielleicht bis zu 10m möglich.

USB Vorteil



USB ist prinzipiell schneller, die Protokolle haben keinen Adressierungs-Overhead. Vieles ist dadurch einfacher, z.B. müssen Sender (Potiboard) und Empfängeradresse (Potiboard-App) nicht konfigurierbar programmiert werden.

USB Vorteil



USB liefert out of the box ausreichend Strom für Drehgeber und Mikroprozessor. Beim Netzwerk müßte zusätzliche Hardware (z.B. PoE) hinzugefügt werden, wenn

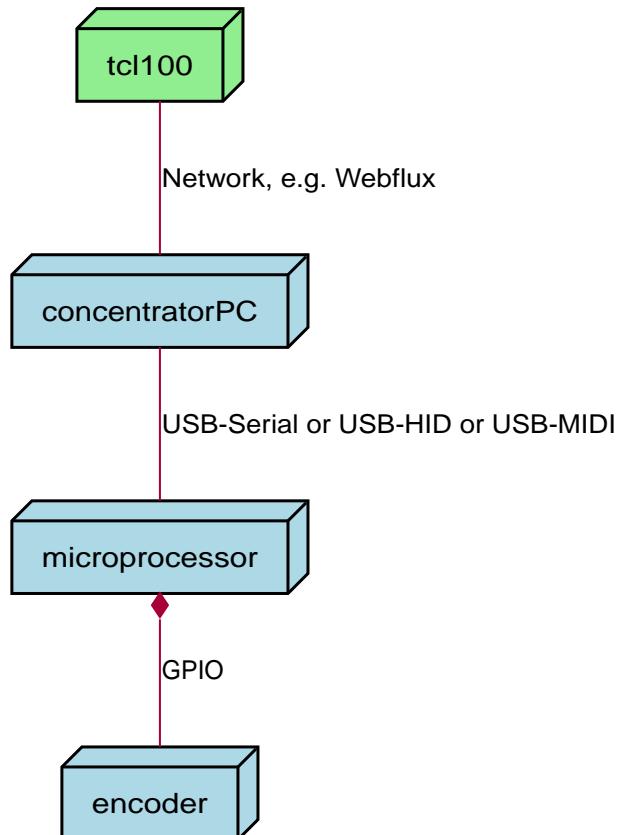
ein Stromnetz-Anschluss vermieden werden soll (USB als nur zum Stromanschluss ginge natürlich auch).

Netzwerkbasierte System-Architekturen und Technologien

Netzwerkbasierter Testsystem mit der Technologie Spring Webflux

GSI/FAIR 08.07.2022

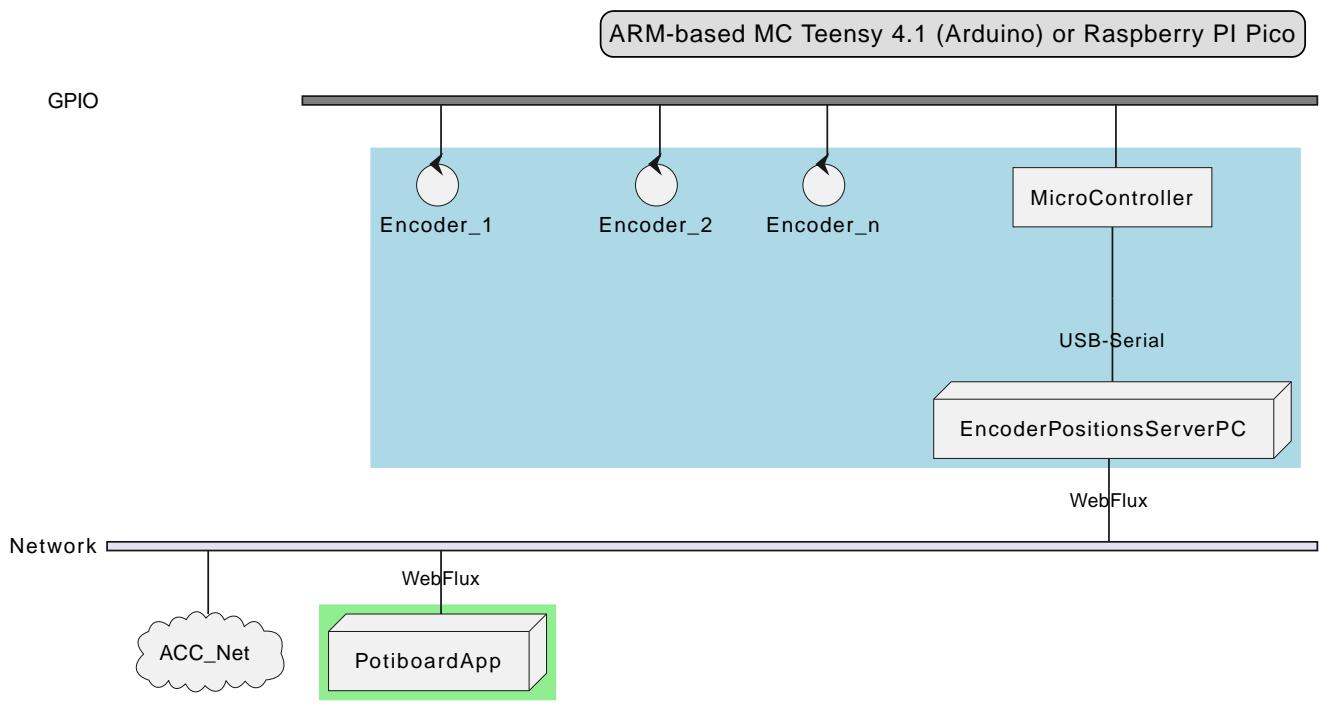
Reference Implementation with Spring WebFlux



Deployment Diagram Potiboard Webflux, A. Bloch-Späth, M. Stein

Figure 2. UML-Komponenten Diagram Network Connection with Spring Webflux

Reference Implementation with USB-Serial and Spring WebFlux



System/Network Diagram Potibord 1., A. Bloch-Späth, M. Stein

Figure 3. Test-Implementation 1

Es wurde ein Referenzsystem, wie im oberen Bild dargestellt, auf Basis eines Teensy 4.1 Mikrocontrollers entwickelt, der die Inkrementen der Encoder in hoher Geschwindigkeit bis in eine Beispiel-JavaFX-Applikation weiterreicht.

Die im Referenzsystem eingesetzte Datenübermittlungstechnologie basiert auf der Technologie **Spring Webflux** und dem "Reactive Toolkit" **Project Reactor**. Sie wurde ausgewählt, da sie der "GSI Controls Applicationsservice-Technologieauswahl" entspricht, die für die Operating-Applicationen im FCC und HKR eingesetzt werden soll und teilweise schon eingesetzt wird.

Ein Nachteil und in mancherlei Hinsicht sicher auch Vorteil dieser Architektur ist die Einführung eines java-basierten (Spring-) Webflux-Servers (siehe Bild **EncoderPositionsServerPC**), der ein PC-System mit Controls-konformen OS sein sollte. Es ist also eine Schicht (**Tier**) notwendig, um die Inkrementen der verschiedenen Encoder im **WebFlux**-Format zu versenden.

Auf der Habenseite dieser Architektur steht die Anpassbarkeit und Wartbarkeit nach den Richtlinien der Controls-Abteilung und damit eine sichere, kontrollierbare Netzwerkkommunikation im ACC-Netzwerk auf lange Sicht und keine Insellösung im ACC-Netz.

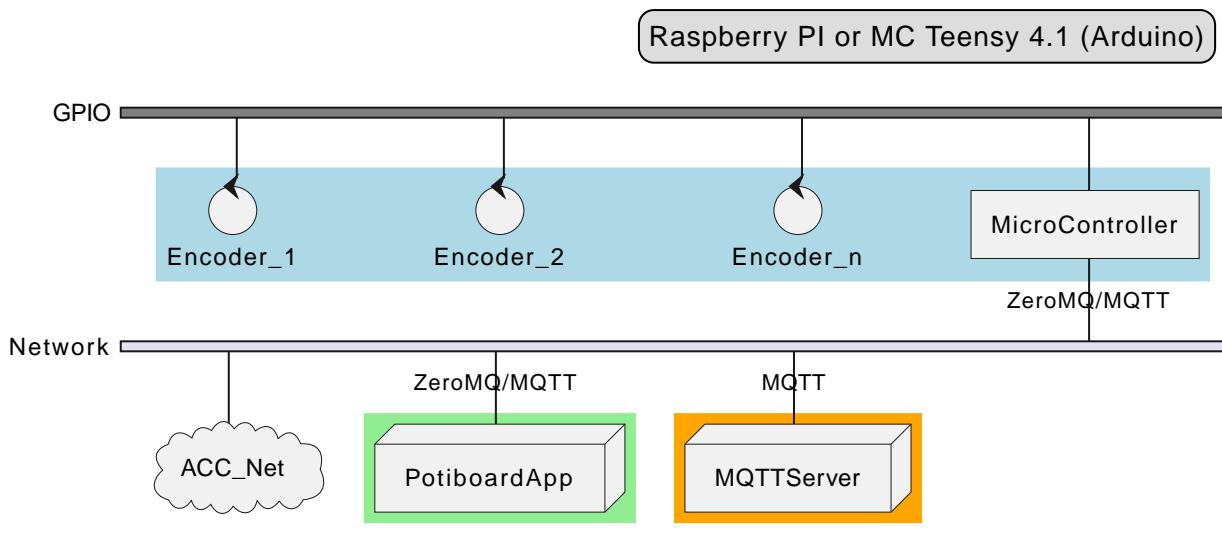
Eine vereinfachte Architektur könnte den Einsatz eines weiteren Rechners, wie der des Konzentrator-PCs, eingesetzt für als **Webflux**-Server, überflüssig machen. Die Instandhaltung des Rechners so wie die Wartung des Betriebssystems (z.B. Rocky Linux) erzeugt wiederkehrende Kosten. Deshalb wurden weitere netzwerk-basierte Technologien in Betracht gezogen.

Netzwerkbasierte Testsysteme mit den Technologien ZeroMQ und MQTT

Test-Implementation 2

GSI/FAIR 06.11.2022

Implementation with ZeroMQ or other messaging libraries



System/Network Diagram Potiboard 2., A. Bloch-Späth, M. Stein

Ein Kandidat für eine einfachere Architektur ist zum Beispiel die Technologie **ZeroMQ**, die sich mit einem Raspberry Pi 4, wie getestet, leicht einsetzen lässt.

Sehr interessant ist auch die **MQTT**-Technologie, die allerdings die Notwendigkeit des Aufsetzens eines **MQTT**-Servers nach sich ziehen würde und somit den Vorteil der Kostenersparnis zumindestens teilweise wieder verliert.

USB-basierte System-Architekturen und Technologien

Testsystem mit der Technologie USB-Serial

Testsystem mit der Technologie USB-HID

Testsystem mit der Technologie USB-MIDI

Grundlegende zentrale Anforderungen (Stichwortliste)

Komplexität, Lebensdauer und Wartbarkeit Hardware

Die Funktion der eingesetzten Drehgeber und Mikrocontroller muss durch Verfügbarkeit am Markt oder durch Reserveteile-Einlagerung für möglichst mehrere Jahrzehnte mit finanziell überschaubarem Aufwand absicherbar sein. Komplexe Systeme oder eine hohe Anzahl von verschiedenen benötigten Hardwarekomponenten sollte wenn möglich vermieden werden.

Komplexität, Lebensdauer und Wartbarkeit Software

Die eingesetzte Software auf Mikrocontroller und auf Potiboard-Applikationsseite sollte aus möglichst gut gepflegten und verständlichen Open-Source Projekten mit hoher Verbreitung stammen. Dies kann auch Auswirkungen auf die Wahl des Mikrocontrollers haben. Der notwendige selbst geschriebene Source-Code sollte möglichst einfach wartbar sein. Auf dem Mikrocontroller kommen die Programmiersprachen Assembler, C und Python in Frage, auf der Potiboard-Applikationsseite werden Java-basierte Lösungen preferiert.

Administrations-, Konfigurationsaufwand

Der Aufwand für zusätzliche Hardware und Software, wie z.B. der KonzentratorPC für [Webflux](#) oder ein [MQTT-Server](#) (Linux-Administration, Hardwarepflege) oder zusätzliche Stromversorgungswege als auch der Aufwand für Konfigurationen (Netzwerk-Adressen-Pflege) sollte minimal gehalten werden. Unter diesen Punkt fallen auch notwendige Linux-Anpassungen z.B. auf den tcl100 Rechnern für den HKR.

Geschwindigkeit Signalübertragung der Inkremente der Encoder

Die vom Nutzer über den Drehgeber zum Mikrocontroller und dann in das Java-Programm sollte zwischen max. bei 10 ms (100 Hz) liegen, besser deutlich niedriger.

Duplex-Signalübertragung, nicht nur für die Inkremente der Encoder in eine Richtung, sondern auch in der Gegenrichtung von der Portiboard-App zurück zum Potiboard-Encoder-Device.

Um die Benutzererfahrung am Potiboard-Encoder-Device zu verbessern, sollte es technisch möglich sein, Informationen wie Status der Verbindung, oder auch Magnet-Nomenklaturen an das Potiboard-Encoder-Device zu übertragen.

Tabelle Technologiebewertung

Table 1. Versuch der Einordnung der Stärken und Schwächen der verschiedenen Technologien

| Eigenschaft — Technologie | USB | Netzwerk | Hardware | Software | Administration | Geschwindigkeit | Duplex | ∑ * |
|--|-----|----------|----------|----------|----------------|-----------------|--------|---------|
| Webflux | | X | * | ** | * | ** | *** | 9 |
| MQTT | | X | * | * | * | ** | *** | 8 |
| ZeroMQ | | X | ** | ** | ** | ** | *** | 11 |
| Socket | | X | ** | ** | ** | *** | ** | 11 |
| USB-Serial | X | | *** | *** | ** | *** | ** | 13 |
| USB-HID | X | | *** | ** | ** | ** | ** | 11 |
| USB-MIDI | X | | *** | *** | *** | * | * | 11 |
| RS232/RS485 | | | | | - | | | |
| MIDI (DIN) | | | | | - | | | |
| X = gehört zu, - = ungenügend, * = ausreichend , ** = gut , *** = sehr gut | | | | | | | | |

Die Tabelle dient nur als Diskussionsgrundlage für die verschiedenen Technologien. Für einen Vergleich wären die verschiedenen Eigenschaften (Spalten) zu gewichten. Die ∑ * Spalte dient nicht der objektiven Bewertung;

Fazit

Wenn USB als Datenübertragungssystem für Potiboard-Prototypentwicklungen vorerst als ausreichend bewertet wird, wäre der technische Vorschlag, für den ersten Prototypen die Encoder-Signale mit einem Arduino kompatiblen Mikrocontroller der Art Teensy 4.1 zu verarbeiten und von diesem aus die Inkrementenzählerwerte über das USB-MIDI-Protokoll an die java-basierte Potiboard-Applikation weiterzuleiten.

Der Teensy 4.1 ist ein kosteneffizienter, gut verfügbarer und hoch performanter 600 MHz ARM Cortex M7 Mikrocontroller. Seine über die Arduino-IDE leicht einbindbaren Open-Source Bibliotheken sind verbreitet und gut unterstützt. Die in den Tests eingesetzten Bibliotheken für Encoder sowie die USB-Serial-, USB-HID- und USB-Midi Bibliotheken funktionierten schnell und problemlos.

Das USB-Midi Protokoll bietet als einzige USB-Datenübertragungstechnologie echtes Plug-and-Play an einem Linux-basierten Host (wie z.B. TCL100). Auf der Java-Seite, also bei der Entwicklung und Wartung der Potiboard-Applikation, wird

MIDI direkt von der JRE unterstützt durch die Java Sound API. D.h. es werden wahrscheinlich nie zusätzliche Bibliotheken oder Abhängigkeiten einzubinden sein. Diesen Vorteilen stehen gegenüber eine leicht erhöhte Komplexität bei der Programmierung der Übertragungsdatenpakete und eine niedrigere aber noch ausreichende Datenübertragungsrate.

Wenn USB als Datenübertragungssystem als möglicherweise nicht ausreichend bewertet wird, müsste die Evaluierung der netzwerk-basierten Technologien weitergeführt werden. Eine rein *socket-basierte* Verbindung von einem netzwerk-fähigen Mikrocontroller zur java-basierten Potiboard-Applikation wäre ein begehbarer Weg oder eine auf das *ZeroMQ-Messaging* basierende Übertragung zwischen Mikrokontroller und der Potiboard-Applikation..

Anex