**Projekt-Dokumentation**

**1. Start**

**Basisarchitektur des Arduino MEGA2560**

* **Ziel:** Entwicklung und Implementierung der grundlegenden Architektur.

**Module und Funktionen**

* **Berechnungs-Module:** Welche Berechnungen wurden implementiert und wie.
  + Methoden für verschiedenste Statistische Berechnungen, sowie Methoden für die Optimierung von Berechnungen
* **JSON-Verarbeitung:** Details zur Integration und Nutzung von JSON.
  + Verpacken, Serialisieren/Deserialisierung von Daten bzw JsonStrings, für die Kommunikation mit dem HAS und die erleichterte Einpflegung der Daten in die MongoDB Instanz
* **Serielles Menü:** Aufbau und Struktur.
  + Einfaches Menü zur Steuerung von Basisfunktionen der eSW und des ArduinoMEGA2560
* **Zeit-Handling:** Nutzung von Zeit
  + Einfaches Modul, um genaue Zeitstrings zu erstellen, basierend auf der Compile/Upload Zeit
* **Error-Handling:** Vorgehen und Strategien.
  + Zentrale Überwachung und Error Handling in der Embedded Software, um frühzeitige Fehler zu erkennen und bei Problemen schnell zu handeln.
* **Schnittstellen:** Kommunikation mit verschiedenen Geräten
  + **Ethernet:** Beschreibung der Integration und Nutzung.
    - **Kommunikation mit HAS**
    - **Kommunikation mit VAT-Slave uC**
  + **SPI:** Funktionalität und Einsatzzwecke.
    - **Kommunikation mit Sensoren/Aktoren**
  + **I2C:** Implementierung und Vorteile im Projekt.
    - **Kommunikation mit Sensoren/Aktoren**
  + **Serial:** Verwendung für Debugging und Datenübertragung.
    - **Debugging und Datenübertragung in der DIE, sowie mit anderen Tools**
* **Sensor Modul:**
  + Druck- und Temperatursensoren zur Überwachung, die Sensoren wurden dann in die Files Pressure und Temperature unterteilt.
* **Pointer Utilities:**
  + Diese Modul Kümmert sich hauptsächlich um das Memorymangement und Pointermangement

Zusammengefasst haben wir also folgende Module:

* **comModule**
  + ETH
  + SER
  + SPI
  + I2C
* **sensModule**
  + pressure
  + temperature
* **jsonModule**
* **calcModule**
* **timeModule**
* **reportSystem**
* **serialMenu**
* **ptrUtils**
* **config**

**Auswahl der Software & Tools**

* Nach Evaluierung der Tools haben wir und für Sloeber IDE entschieden, besser für Projektstruktur und passende Plugins für Eclipse. Da besser und umfangreicher als Herkömmliche Arduino IDE.

**GitHub und Versionsverwaltung**

* Anlegen der Repos für EmbeddedSoftware, des HardwareAccessService und der Dockerfiles. Sowie aufsetzten eines Projektes in Github zur Verwaltung von Tasks.

**Aufsetzten des Basic Dockerfile und Compose für unseren HAS**

* Aus Praxisgründen und wegen MongoDB Unterstützung, haben wir uns für Ubuntu entschieden, welches in dem Dockercontainer läuft. Mit diesem Betriebssystem führen wir dann auch unserer wichtigsten Applikationen aus (HAS, Grafana, MongoDB)
  + Nach einigen Optimierungen und tryouts habe ich mit Felix Latzer eine funktionierende Version des Dokcerfiles und Dockercompose entwickelt.
  + Integration des 3Third Party Plugin für die Verarbeitung von MongoDB bzw json daten.
  + Tryout der Dockercontainer auf verschieden Hosts um Kontinuität zu garantieren, wir haben das Setup auf meinem privaten RasperryPi4B und auf dem RasperryPi4B des Projektes ausprobiert.

**Meeting und Emailverkehr mit VAT bezüglich der Hardware**

* Nach einigen Mails und einem Meeting in der Schweiz mit unseren Partnern von VAT haben wir entschlossen, den uC von VAT als Slave in unserem System anzusprechen. Vorerst wollten wir hierfür aus Performancegründen und wegen den Industriestandards EtherCAT verwenden, da aber unsere MEGA2560 mit einem Ethernetschield mit dem HAS über einen Ethernet-Switch kommuniziert, war die Kommunikation via EtherCAT keine option mehr für uns. Ein Betreiben von EtherCAT und Ethernet über denselben „normalen“ Switch ist nicht möglich. Daher haben wir uns für ein Ethernetmodul von VAT für den uC entschieden, diese ermöglicht es uns das ethernetModul des MEGA2560 anzupassen und dann it simplen GET/SET commands mit dem Slave zu kommunizieren.

**2. Meetings und Zusammenarbeit**

**Meetings mit Felix**

* **Thema:** Kommunikation zwischen HAS (Hardware Access Service) und ESW (Embedded Software).
* **Inhalt:**
  + Zielsetzung der Meetings.
  + Besprochene Lösungen und Fortschritte.
  + Kommunikation zwischen HAS und eSW
  + Spikes und Tryouts für ein Simples End-To-End Testing
    - Sensoren auf eSW erfassen Messdaten, eSW verarbeitet diese und verpackt sie in passendem Format, Versendet via ethernetModule an HAS. HAS nimmt daten entgegen, speichert diese in die MongoDB und verarbeitet die Daten mithilfe von Grafana

**Meetings mit Dominik**

* **Thema:** Einschulung in die Architektur.
* **Inhalt:**
  + Erklärung der Funktionsweise.
  + Anleitungen zur Erweiterung der Architektur.
  + Erstellung von Spikes, Debugging und Tryouts für sein HighCurrent/Flyback module
  + Rategeber und Brainstorming für Problemlösungen und neue Ansätze

**Meetings mit Bonelli**

* **Thema:** Kommunikation von/zu Manual-Board mit eSW
* **Inhalt:**
  + Erklärung der Funktionsweise
  + Besprechung zu eventueller Erweiterung der Architecture
  + Verwendete bzw. zu verwendende Hardware und deren Interfaces

**3. Debugging**

* **Methode:**
  + Serial Prints
  + StackOverflow
  + Blogposts
  + LLMs
  + Austausch mit den Kollegen
* **Ergebnisse:**
  + Verbesserungen und Lösungen, die implementiert wurden.
  + Änderungen von Kommunikationsinterfaces, updates für verschiedenste Modules

**4. Programmierung und Libraries**

* **Sprache:** C/C++.
* **Externe Libraries:** Beschreibung der verwendeten Libraries und ihrer Anpassung.

**5. Dokumentation**

**Code-Dokumentation**

* **Doxygen:**
  + Einrichtung auf GitHub.
  + Generierung von HTML- und LaTeX-Dokumenten.

**Allgemeine Dokumentation**

* Strukturierte Dokumentation der Codebase.
* Zusätzliche Informationen und Anleitungen.