HTWK Hochschule für Technik, Wirtschaft und Kultur Leipzig

Einschub: Abschlussprojekt

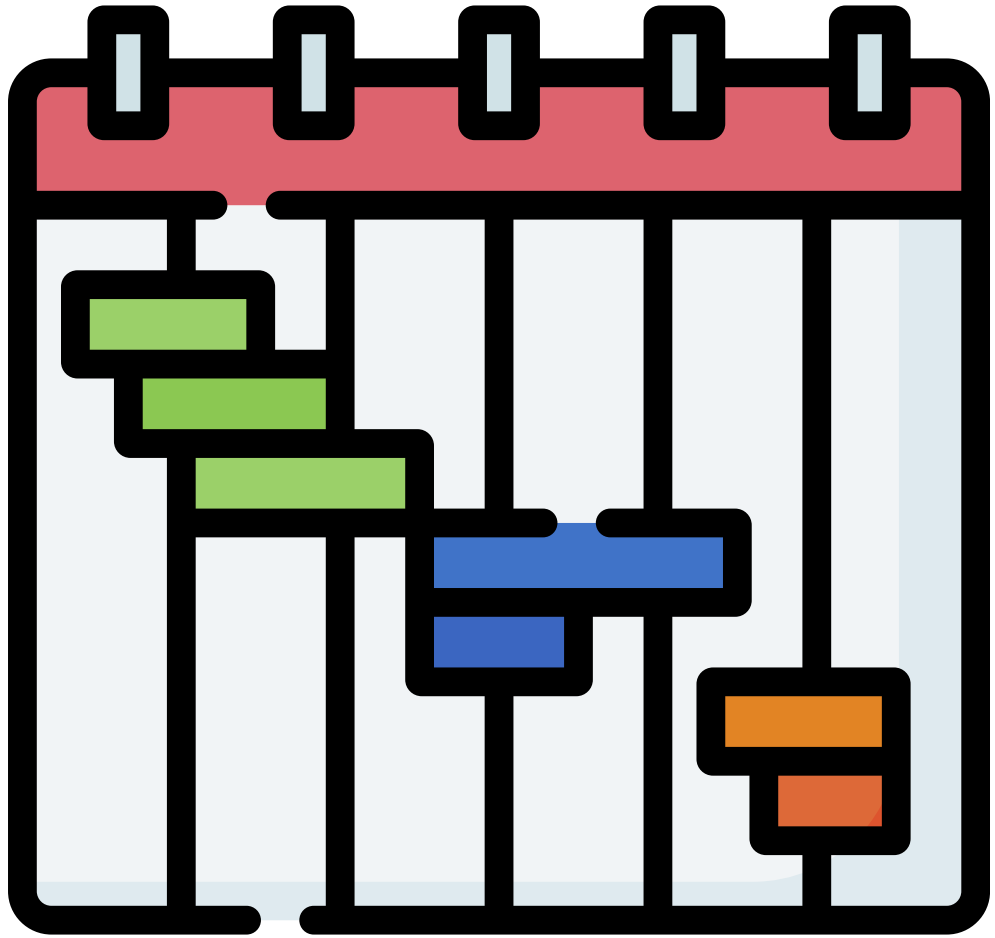
Abschlussprojekt

Überblick

- 6 bis 8 Wochen für Projekt Zeit
- ~18.07.205 Präsentation
- 31.07.2025 Projektbericht (Spätestens)
- Teams 2 - 4 Studierende

Projekt

- Programmierarbeit
  - Sprache: C/C++, Python, Assembly
  - ChatGPT / LLM erlaubt und erwünscht
- Ordentlicher Entwurf
- Code-Testen (Unit Tests, z. B. TDD)



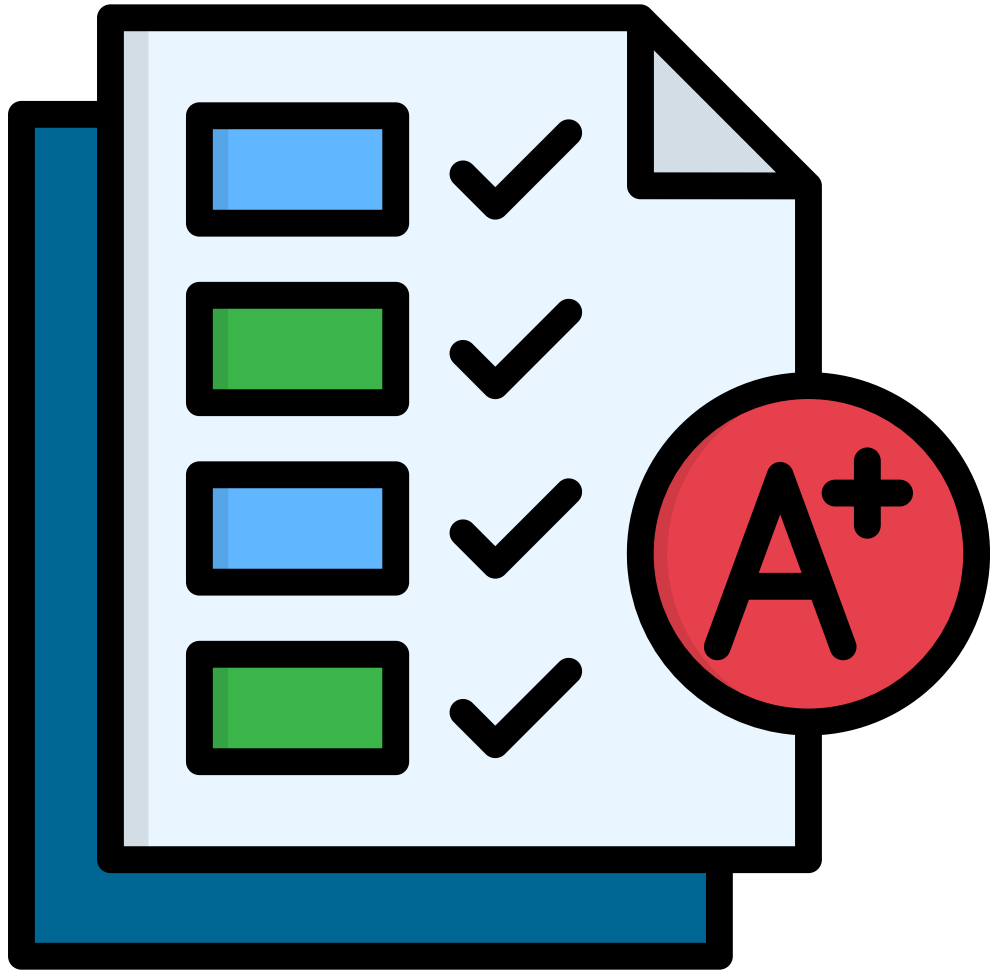
Benotung

Zusammensetzung

- 50 % Code
- 25 % Dokumentation
- 25 % Vortrag

Grundlage

- Inhalte der VL mit Projekt verknüpfen
  - DMA, IRQ, Assembler, Security, IoT, ML
  - Theorie einfließen lassen
- Schwere und Relevanz des Themas



Abschlussprojekt

Präsentation

- Zeit
  - 15 min Vortrag
  - 5 min Diskussion
- Ein Präsentator
- Folien am Beamer

Inhalt der Präsentation

1. Titel und Vorstellung (1 Min)
   - Projekttitel,
   - Vortragende, Gruppe

2. Motivation und Zielstellung (2 Min)
   - Motivation in einem Satz
   - Was ist das Problem bzw. die Herausforderung?
   - Ziel: Was soll erreicht werden?

3. Projektidee und Konzept (3 Min)
   - Grundidee, Lösungsansatz
   - Technologiewahl (kurz und verständlich)
   - Was macht das Projekt besonders ?

4. Umsetzung (5 Min)
   - Architektur / Design (ggf. Diagramm)
   - Wichtigste tech. Komponenten oder Module
   - Verwendete Soft- und Hardwarewerkzeuge
   - Herausforderungen und deren Lösung

5. Demonstration / Ergebnisse (2 – 3 Min)
   - Live-Demo, Video oder Screenshots
   - Welche Ergebnisse konnten erzielt werden
   - Vergleich mit ursprünglicen Zielsetzung

6. Fazit und Ausblick (1 – 2 Min)
   - Was habt Ihr gelernt
   - Was würdet ihr verbessern oder erweitern
   - Gesellschaftlicher und prakt. Nutzen (optional)

Abschlussprojekt

Projektbericht

- Im gitlab Repository (HTWK)
- Code
- Projektbericht
- Anleitung zur Inbetriebnahme

```
projektname/
├── README.md          ← Kurzfassung
├── docs/
│   └── abschlussbericht.md
├── src/               ← Quellcode
├── test/              ← Testfälle
├── .gitignore
├── LICENSE
└── assets/            ← Bilder etc.
```

Inhalt des Repository

Projekttitel
- Prägnanter Titel
- Autoren
- Datum

1. Einleitung
- Motivation und Problemstellung
- Ziele des Projekts
- Relevanz / Anwendungsgebiet

2. Technischer Hintergrund
- Relevante Technologien
- Verwendete Frameworks, Hardware, Protokolle ..

3. Projektidee und Anforderungen
- Kurze Beschreibung des Konzepts
- Zielgruppe / User
- Funktionale und Nicht-Funktionale Anforderunge (Performance, Zeit, Speicher)
- Erste Skizzen und Diagramme

4. Architektur und Umsetzung
- Übersicht der Systemarchitektur
- Modulaufbau / Komponenten
- Wichtige Schnittstellen
- Begründung von Entscheidungen (Warum X anstatt Y)

Inhalt des Repository

5. Implementierung
- Beschreibung zentraler Programmteile (mit Codebeispielen)
- Eingesetzte Tools und Sprachen

6. Tests und Ergebnisse
- Was wurde getestet und wie?
- Ergebnisse, Screenshots, Output-Beispiele
- Optional: Benchmark oder Metriken

7. Fazit und Ausblick
- Was lief gut, was war schwierig?
- Erfüllung der Ziele
- Lessons Learned
- Ideen für die Weiterentwicklung

8. Repository-Überblick
- Aufbau des Repos
- Setup-Anleitungen
- Beispiele zur Nutzung

9. Lizenz und Danksagung
- MIT, BSD, Apache 2.0
- Verweis auf genutzte Libraries, OSS-Komponenten
- Danksagung (optional)

Projektidee 1

Sprachgesteuertes Gerät mit TensorFlow Lite (Kommandoerkennung)

Ziel: Ein kompaktes Embedded-System, das **gesprochene Kommandos** wie „on“, „off“, „start“, „stop“ erkennt und entsprechend Hardware steuert (z. B. LEDs, Relais, Motoren).

Plattform: ESP32-S3 (mit integriertem DSP + TensorFlow Lite Unterstützung)

Highlights:

- **Modelltraining** z. B. mit Google's Speech Commands Dataset, Meta
- **Quantisierung** und Deployment mit TensorFlow Lite Micro
- **Wakeword-Erkennung** („Hey Pico“)
- **Echtzeit-Audioaufnahme**, FFT und Klassifikation

Projektidee 2

IoT-Umweltsensor mit Anbindung an einen Cloud-Dienst

Ziel: Ein **IoT-Gerät**, das z. B. Temperatur, Feuchtigkeit und Luftqualität misst und diese Werte regelmäßig an **IoT-Plattformen** wie ThingSpeak, ThingsBoard, Adafruit IO, AWS IoT oder Blynk sendet.

Plattform: Raspberry Pi Pico W2 mit Sensor

Highlights:

- **WLAN-Verbindung** per Pico SDK
- **MQTT-Protokoll** für Datenübertragung (oder anderes)
- Visualisierung in einem **IoT-Dashboard**
- Optional: Remote-Control möglich

Projektidee 3

LoRa-Kommunikation + Gateway-Bridge zu WiFi/Internet

Ziel: Ein Gerät mit Pico W2 kommuniziert per **LoRa** mit einem entfernten Sensor (z. B. auf einem Feld), und stellt die Daten über ein Gateway ins Internet bereit.

Plattform:

- Sensor Node: Pico W2 / ESP32 + RFM95 LoRa-Modul
- Gateway: Pico W2 / ESP32 + WiFi + LoRa

Highlights

- **Punkt-zu-Punkt** oder Mesh-Netzwerk
- **MQTT-Bridge** für die Weiterleitung
- **Akkubetriebene** Sensoren (Low Power Mode)
- Einsatz in Smart Agriculture / Monitoring

Projektidee 4

Buffer Overflow Angriff mit Binärcode

Ziel: Demonstration einer klassischen Stack-basierten **Buffer Overflow**-Schwachstelle auf einem Mikrocontroller durch die Ausführung von injiziertem Schellcode (Binärcode).

Plattform:

- Raspberry Pi Pico W2 (RP2350)

Highlights

- Entwicklung eines unsicheren C-Programms mit z. B. strcpy-Fehler (Webapp, Terminal-App)
- Eingabe eines **manuellen Payloads** (Padding + Shellcode)
- Ausführung von **Assembler-Shellcode** (z. B. LED-Blink oder UART-Ausgabe)
- Demonstration von möglichen **Gegenmaßnahmen**: (fstack-protector, strncpy, PMP)