

Herausforderungen an das Softwaredesign moderner Embedded-Systeme

In den heute eingesetzten Embedded-Systemen spielt die Software eine zunehmend zentrale Rolle. Systeme basierend auf komplexen Mikrocontroller- und Prozessorarchitekturen müssen mehr Funktionalitäten bieten – und das in kürzerer Entwicklungszeit als je zuvor. Dies stellt Entwickler vor die Herausforderung, qualitativ hochwertige Software zu entwerfen, die sowohl effizient als auch zuverlässig ist.

Früher wurde oft die VHIT-Methode (vom Hirn ins Terminal) angewandt, bei der Entwickler ohne großen Plan direkt mit der Implementierung begannen. In Zeiten wachsender Komplexität und steigender Anforderungen an die Softwarequalität ist diese Methode jedoch nicht mehr praktikabel. Stattdessen rückt ein strukturiertes Vorgehen beim Softwaredesign in den Vordergrund.

Erfolgreiche Methoden für das Softwaredesign mit C

Eine strukturierte und methodische Herangehensweise an das Embedded-Softwaredesign ist entscheidend. Dabei kommen verschiedene Design Patterns und Best Practices zum Einsatz, die speziell für die Programmierung mit C geeignet sind. Mit diesen Methoden lässt sich Software wartbar, erweiterbar und effizient gestalten.

Design Patterns für Embedded-Software

Design Patterns sind bewährte Lösungen für wiederkehrende Probleme in der Softwareentwicklung. Einige der wichtigsten Patterns für Embedded-Software umfassen:

Adapter Pattern

Ermöglicht die Aufruf-Anpassung von neuen zu bestehenden Software-Anteilen. Das Adapter Pattern wird häufig auch als Wrapper bezeichnet. Ein Beispiel für dessen Anwendung ist die Implementierung eines OSAL (Operating System Abstraction Layer).

Facade Pattern

Die Fassade repräsentiert ein Software-Interface, über das Software-Architekturelemente miteinander zusammenarbeiten und gleichermaßen entkoppelt sind.

Strategy Pattern

Ermöglicht zur Laufzeit den kontextabhängigen Wechsel zwischen ähnlichen Algorithmen bei gleichbleibendem Aufruf.

Observer Pattern

Ermöglicht es einem Objekt, andere Objekte über Änderungen seines Zustands zu informieren. Dies ist besonders hilfreich in Systemen, die auf Ereignissen basieren.

[Übersichtsmap „Embedded Software Design Patterns“ \(PDF\) zum Download](#)

Normen und sicherheitskritische Aspekte

Für viele Embedded-Systeme gelten strenge Standards und Normen. Aus diesen leiten sich Anforderungen zur Erfüllung der Qualitätsmerkmale *Funktionale Sicherheit* (Safety) und *Manipulationssicherheit* (Security) ab. Dies ist besonders in Branchen wie der Automobilindustrie, Medizintechnik und Luftfahrt der Fall. Hier spielen Richtlinien wie MISRA-C eine zentrale Rolle, die spezifische Regeln für die sichere Programmierung in C vorgeben.

Wichtige Eigenschaften von Embedded-Software

Bei der Entwicklung von Embedded-Software müssen mehrere Schlüsselmerkmale berücksichtigt werden:

- **Echtzeitfähigkeit:**
Die Software muss in der Lage sein, innerhalb vorgegebener Zeitrahmen zu reagieren, was deterministische Abläufe in der Hardware und Software erfordert.
- **Wiederverwendbarkeit:**
Gut strukturierte und modulare Software erleichtert die Wiederverwendung von Code in verschiedenen Projekten und Anwendungen.
- **Anpassbarkeit:**
Die Software muss flexibel an veränderte Anforderungen und Rahmenbedingungen angepasst werden können.
- **Lesbarkeit:**
Gut lesbarer und verständlicher Code ist entscheidend für Wartung und Weiterentwicklung. Dies wird durch klare Strukturierung und aussagekräftige Benennungen erreicht. Ein zum Programmcode konsistentes Softwaremodell, modelliert mit der UML (Unified Modeling Language), erleichtert die Entwicklung und repräsentiert gleichzeitig die Dokumentation der Software.

Fazit

Die Entwicklung von Embedded-Software erfordert heute eine systematische Herangehensweise, die über das einfache „Vom Hirn ins Terminal“-Denken hinausgeht. Durch den Einsatz bewährter Methoden und Design Patterns sowie die Berücksichtigung von Standards und Normen können Entwickler qualitativ hochwertige Software entwerfen, die den Anforderungen moderner Embedded-Systeme gerecht wird.

Eine umfassende Weiterbildung in diesem Bereich bietet die Möglichkeit, diese Methoden und Techniken praxisnah zu erlernen und anzuwenden, um den Herausforderungen der heutigen Embedded-Entwicklung erfolgreich zu begegnen.

MicroConsult bietet Ihnen dazu professionelle [Trainings und Coachings](#) rund um das Thema Embedded-Softwaredesign an.

Weiterführende Informationen

1. [MicroConsult Fachwissen zum Thema Embedded SW-Entwicklung](#)
2. [MicroConsult-Trainings zum Thema Embedded SW-Entwicklung](#)
3. [Alle Trainings & Termine auf einen Blick](#)

Der MicroConsult-Newsletter



Wir informieren Sie mehrmals jährlich über Trends und Best Practices im Embedded Systems Engineering.

Erhalten Sie wertvolles Fachwissen und Tipps aus erster Hand von unseren Embedded-Experten!

[Jetzt abonnieren!](#)