Trends in der Mikroelektronik
Neue Perspektiven und Anforderungen für mobil agierende Geräte

Jedes Jahr werden neue, noch leistungsfähigere Mikrocontroller-Architekturen angekündigt. Dies ebnet den Weg für immer komfortablere und sogar autonom arbeitende Transportgeräte für unseren Alltag. Neben der gesteigerten Rechenleistung – MIPS per Watt – sind die enormen Anforderungen an Safety und Security eine der größten Herausforderungen.

Autonom fahrende Systeme in geschlossenen Räumen, wie z.B. Transportroboter in Fabrikationseinrichtungen, sind heute bereits im Einsatz. Damit solche autonom agierenden Fahrzeuge in unserem öffentlichen Raum aktiv werden können, müssen die Designer zahlreiche Sicherheitsmaßnahmen implementieren. So gibt es z.B. auf Flughäfen schon Schienenfahrzeuge, die Fluggäste fahrerlos zwischen den Terminals befördern.

Auch laufen eine Reihe von Studienprojekten zur Auslieferung von Päckchen und Paketen mittels Drohnen, und an der Realisierung von Lufttaxis im Umfeld größerer Einzugsgebiete wird z.B. in Berlin, München, Stuttgart und Frankfurt gearbeitet, damit diese Transportmittel unseren Alltag begleiten können.

In unseren Autos werden zunehmend Elektroantriebe verbaut, und das autonome Fahren steht bereits in den Startlöchern. Die verfügbaren Systeme, die unsere Fahrzeuge sicherer und komfortabler machen sollen, sind aktuell noch auf dem Stand eines „Schönwettersystems“. Bei Regen oder Schnee fallen jedoch noch regelmäßig die Abstandsregelsysteme aus und übergeben die Verantwortung an den Fahrer. Bei autonom fahrenden Fahrzeugen ist dies später im Einsatz natürlich undenkbar.

Wenn diese autonom fahrenden Systeme an unserem realen Verkehr, also außerhalb des Schutzraumes eines geschlossenen Systems, teilnehmen sollen, müssen sie imstande sein, selbst bei widrigen Witterungseinflüssen bzw. bei nicht regelkonformem Verhalten anderer Verkehrsteilnehmer die Fahrzeuge sicher und ohne Gefährdung zu steuern. Dazu müssen diese Systeme eine Vielzahl an Mikrocontrollern, Sensoren, Kameras, Aktuatoren (Motoren) und zahlreiche Softwareprogramme für Steuerungs-, Regelungs- und Kommunikationsaufgaben enthalten.

Die Entwicklung und das Design solcher Systeme erfordern also ein Höchstmaß an Sicherheit. Hinter dem deutschen Wort Sicherheit stehen im Englischen **Safety** und **Security**.

Safety

Natürlich erwarten wir, dass sich autonom bewegende Geräte ein Höchstmaß an Sicherheit gewährleisten. Da im Hintergrund kein Mensch steht, der beim nicht erwarteten Eintritt eines Fehlers das "Ruder" übernehmen kann, muss das System alle möglichen und notwendigen Maßnahmen ergreifen und ständig alles überwachen. Im Fehlerfall muss dann innerhalb kürzester Zeit entschieden werden, wie das System in einen sicheren Zustand überführt werden kann.

Neue Safety-Anforderungen (z.B. die Klassifizierung einzelner Projektteile in unterschiedliche SIL- oder ASIL-Klassen) unter Berücksichtigung CPU-privater Speicherressourcen und gemeinsam verwendeter Ressourcen, wie globale RAM-Speicher, müssen eingehalten werden. Ferner sind die Datenintegrität und der Schutz vor unberechtigten Zugriffen auf Peripheriekomponenten, die sicherheitsrelevante Abläufe steuern, zu gewährleisten.

Zum Erreichen der geforderten Sicherheit gibt es bei den neuesten Mikrokontroller-Generationen zahlreiche kontinuierlich laufende Safety-Überwachungen und -Funktionen. Das Gesamtsystem eines Mikrocontrollers beinhaltet dazu Safety-Mechanismen.

Am Anfang stehen die essentiellen Voraussetzungen für die Arbeit eines Mikrokontrollers: Durch Spannungs- und Taktüberwachung wird garantiert, dass die Steuerung eines Systems erst anlaufen kann, wenn die Mindestanforderungen erfüllt sind. Dazu sind diese beiden Überwachungen mit dem Baustein-Reset verbunden und geben diesen erst frei, wenn Spannung und Taktung in den voreingestellten Arbeitsbereichen liegen.

Mit freigegebenem Reset und der notwendigen Taktung kann die Grundinitialisierung des Bausteins erfolgen. Alle Programm- und Applikationsdaten in den Mikrocontroller-Speichern (Flash und RAM) werden über Sicherungscodes (Error Correction Codes, ECC) vor ihrer Verwendung überwacht. Selbst bei der internen Kommunikation dieser Daten zwischen Speicher und CPU erfolgt eine Überwachung der Adresse und Daten.
CPUs zur Abarbeitung des Programms haben zur Einhaltung der Safety-Anforderungen die Möglichkeit, die Aufgabe z.B. um 2 Takte versetzt in zwei identischen CPU-Kernen abzuarbeiten. Das Ergebnis eines Programmbefehls wird in der Haupt-CPU verarbeitet und direkt an das System weitergegeben. Eine zweite CPU (Checker-Core bzw. Lock-Step-Core) verarbeitet alles mit einem Zeitversatz von z.B. zwei Takten. Im Anschluss werden beide Ergebnisse verglichen. Wird eine Diskrepanz der Ergebnisse erkannt, kann diese Fehlermeldung an eine Safety-Steuereinheit (Safety Management Unit, SMU) gemeldet werden.

Eine weitere Überwachungsmöglichkeit bei CPUs ist die Implementierung von Speicherzugriffs-Freigabeeinheiten (sog. Memory Protection Units, MPU). In einer MPU lässt sich für das jeweilige Programm einstellen, auf welche Speicherbereiche lesend oder schreibend zugegriffen werden darf. Damit kann beispielsweise ein unberechtigter Speicherzugriff auf ein sicherheitsrelevantes Software-Modul verhindert werden. Die Datenintegrität einzelner Softwarekomponenten wird so gewährleistet.

Selbst für die Peripheriemodule mit externer Kommunikation über ein Bussystem und die Steuermodule, welche die Arbeit von Motoren regeln und überwachen sollen, gibt es Safety-Funktionalitäten. So können z.B. bei einem Mehrrechnerkern (Multicore-CPU) die Zugriffe einer einzelnen CPU freigegeben oder Zugriffe nicht berechtigter CPUs gesperrt werden. Ferner gibt es für die sicherheitsrelevanten Peripherie-Komponenten jeweils zwei identisch designte Module. Beiden Modulen kann die gleiche Aufgabe zugeteilt werden. Die Ausgangsinformationen, z.B. PWM-Signale zur Geschwindigkeitsregelung eines Elektromotors, können in einer Compare-Unit verglichen werden. Bei Erkennung von Signalabweichungen kann die Ausgabe abgestellt werden und eine Fehlersignalisierung an die SMU erfolgen.

Alle Fehlerindikationen (Spannungsfehler, Taktabweichungen, Speicher-, Kommunikations-, CPU-Fehler), die innerhalb eines Mikrokontrollers erkannt werden, sind an die zentrale Safety-Steuereinheit angeschlossen.

Soweit sind die bestehenden Hardware-Voraussetzungen für die Safety kurz beschrieben.

Nun beginnt die Aufgabe der Softwareentwickler: Die Reaktion auf jeden einzelnen erkannten und gemeldeten Fehler muss in der Safety-Steuereinheit individuell freigegeben werden, und die erforderliche Fehlerreaktion ist abhängig von der möglichen Fehlerauswirkung auszulösen. Eine SMU bietet hierzu folgende Reaktionen an:

- Auf einen behebbaren Fehler kann wahlweise eine Software-Routine in Form einer Interrupt-Service-Routine oder eine Exception-Routine (nicht maskierbarer Interrupt, NMI) aufgerufen werden.
- Wurde ein nicht behebbarer Fehler erkannt, muss ein Reset ausgelöst werden.

Es liegt also in der Verantwortung der Programmierer, wie ein System später auf einen erkannten Fehler reagieren soll oder muss.

Security

Die Sicherheit intern gespeicherter Daten und die Möglichkeit einer geschützten bzw. verschlüsselter Datenübertragung lassen sich durch Security-Module (z.B. High Security Module, HSM) gewährleisten.

Bei diesen Modulen eines Mikrokontrollers handelt es sich um eine eigenständig arbeitende CPU, die geschützt hinter einer Firewall liegt. Der Rest des Mikrocontrollersystems kann nicht auf die HSM-Ressourcen zugreifen.

Diese „Safety-Welt“ hat ihre private CPU, eigene Speicher (Flash und RAM), verschiedene Kryptoprozessoren und einen Random-Number-Generator. Das gesicherte System hat Zugriff auf das Mikrocontroller-Bussystem. So kann es alle internen Komponenten ansprechen und über Kommunikationsperipherie (z.B. CAN, Ethernet, etc.) mit der Außenwelt kommunizieren.

Essentielle Kommunikationskomponenten, wie das Ethernet-Modul, welche systemrelevante und somit verschlüsselte Datenübertragungen z.B. durch das High Security Modul (HSM) sicherstellen sollen, müssen also besonders vor unberechtigten internen und externen Zugriffen geschützt werden. Software-Updates werden in komplexen Projekten mit einer großen Menge an Codezeilen (Lines of Code) immer häufiger vonnöten sein. Es ist demnach zwingend erforderlich, Software-Updates Over-The-Air (SOTA) zu ermöglichen. Diese Methode wird heute schon beispielsweise bei Tesla-Fahrzeugen angewendet.

Der Einsatz von autonomen/mobilen Systemen

Damit mobile Systeme entwickelt werden können, sind alle beteiligten Entwickler mit völlig neuen Aufgabenstellungen konfrontiert.

Alle autonom arbeitenden Systeme erfordern ein Höchstmaß an Kenntnissen auf dem Gebiet der Safety und Security, da diese Geräte ihre Aufgaben ohne das Backup eines Operators erfüllen müssen. Dabei darf durch ihr Wirken kein Mensch gefährdet, verletzt oder gar getötet werden.

Multicore-Test-Knowhow für Safety-relevante Systeme

Hier ist Wissen darüber erforderlich, wie Softwarekomponenten, welche

- auf verschiedenen CPUs abgearbeitet werden sollen,
- unterschiedlichen Safety-Klassen (z.B. ASLI- A bis ASIL_D) zuzuordnen sind oder
- individuellen Schutz durch Memory Protection Units haben (MPUs mit Lese- und/oder Schreibschutz auf Software-Task-Ebene)

den Software-Anforderungen entsprechend, richtig und umfassend zu testen sind.

Neue Herausforderungen für Entwickler und Tester

Jeder in der Entwicklungskette solcher autonom arbeitenden Systeme muss ganzheitlich denken können, damit am Ende wirklich sichere Geräte im Einsatz sind. Dazu gehören Fertigkeiten und Kenntnisse u.a. aus folgenden Bereichen:

- Anforderungsmanagement / Requirements Engineering
- Architektur-Design
- Safety und Security
- verschiedenste Programmiersprachen
- Embedded-Multicore-Softwareentwicklung und -design
- Embedded-Multicore-Mikrocontroller mit modernen Architekturen, die Safety-Support gewährleisten und Security-Module enthalten
- Multicore-Betriebssysteme (OS/-RTOS)
- Debugging und Trace
- Hypervisor-Anwendung
- Software-Simulation und -Test

Die genannten Anforderungen betreffen die beteiligten Projekt-Designer, Embedded-Software-Entwickler und -Tester. Um den neuen Anforderungen gerecht werden zu können, benötigen diese tiefergehende und umfangreichere Kenntnisse.

Machen Sie sich fit für die neuen Herausforderungen - vertiefen Sie Ihre Kenntnisse mit den maßgeschneiderten Trainingsangeboten von MicroConsult:

- Requirements Engineering und Management für Embedded-Systeme
- Embedded-Software-Design und Patterns mit C
- Software-Architekturen für Embedded- und Echtzeitsysteme
- Software-Qualität im Programmcode
- Software-Sicherheit (Safety and Security)
- Objektorientierte Softwareentwicklung - Der Weg zum Clean Code für C++
- Embedded-Multicore-Mikrocontroller in der Praxis
- Real-Time-Betriebssysteme für Embedded-Anwendungen - RTOS

Weiterführende Informationen

- MicroConsult Fachwissen: Multicore & Mikrocontroller
- MicroConsult Fachwissen: Embedded SW-Entwicklung
- MicroConsult Fachwissen: Safety & Security
- Alle MicroConsult Trainings & Coachings

Autor

Ingo Pohle ist Mitgründer und Geschäftsführer der MicroConsult GmbH und international anerkannter Spezialist für Embedded-Lösungen, mit einem reichen Erfahrungsschatz rund um den Einsatz von Embedded-Mikrocontrollern, Bussystemen und RTOS.