Zusammenfassung

Die Ansprüche an die Leistungsfähigkeit und Sicherheit technischer Systeme, wie Autos oder Flugzeuge, steigt stetig. Um den wachsenden Anforderungen gerecht zu werden ist es notwendig, dass maschinenbauliche, elektrotechnische und Softwarekomponenten zusammenarbeiten. Ein System, dass aus solchen Komponenten zusammengesetzt ist, wird als mechatronisches System bezeichnet.

Ein solches System kann über Sensoren Informationen über seine Umwelt sammeln. Es hat aber auch die Möglichkeit, mit anderen mechatronischen Systemen zu kommunizieren oder zu kooperieren. Die Software, die für eine solche Interaktion erforderlich ist, ist sicherheitskritisch, d.h. ein Fehlverhalten der Software kann einen großen finanziellen Schaden verursachen und im schlimmsten Fall auch Menschenleben kosten. Da die Software gleichzeitig aber auch sehr komplex ist und zumeist einen unendlichen Zustandsraum hat, reicht Testen alleine nicht aus, um die Korrektheit der Software nachzuweisen. Automatische Ansätze zur formalen Verifikation wie Model Checking können nur Systeme mit einem endlichen Zustandsraum verifizieren. Theorembeweiser, die einen Korrektheitsnachweis auch für solche Systeme führen können, benötigen die Interaktion mit einem Benutzer, der mit formalen Methoden vertraut ist.

Deshalb wird in dieser Arbeit ein kompositionaler Ansatz vorgestellt, der die Software, die zur Interaktion zwischen mehreren mechatronischen Systemen notwendig ist, automatisch formal verifizieren kann. Der Ansatz baut auf den existierenden Ansatz der MECHATRONIC UML auf. Er nutzt dabei die Tatsache aus, dass ein Systemzustand in der MECHATRONIC UML, charakterisiert durch die mechatronischen Systeme und deren laufende Interaktion, als Graph dargestellt werden kann. Zustandsübergänge wie beispielsweise das Starten oder Beenden einer Interaktion können dann als Graphtransformationsregel beschrieben werden. Zudem können die hier betrachteten strukturellen Sicherheitseigenschaften lokal nachgewiesen werden. Für eine Menge von Graphtransformationsregeln und eine Menge von Sicherheitseigenschaften wird dann gezeigt, dass die Regeln niemals einen korrekten Graphen, also einen Zustand, der alle Sicherheitseigenschaften erfüllt, in einen inkorrekten Graphen überführen können. Die Erreichbarkeit der Graphen wird dabei jedoch nicht berücksichtigt. Somit wird bei der Verifikation nachgewiesen, dass die Sicherheitseigenschaften induktive Invarianten des Systems darstellen.