Code-Generator und Framework für synthetisierte Sicherheitsmechanismen in SPS-Programmen

(Informatik-Bachelorarbeit)

Kontext

Bei einer Synthese gemäß der *Supervisory Control Theory* wird aus einer abstrakten Modellierung eines zu steuernden Systems und einer formalen Spezifikation des gewünschten Verhaltens ein sogenannter Supervisor erzeugt, der Aussage darüber gibt welche Aktionen der Steuerung zu welchem Zeitpunkt erlaubt sind.

Diese Bachelorarbeit wird im Rahmen der Entwicklung eines praxistauglichen Tools zur Synthese auf Grundlage der *Supervisory Control Theory* für SPS-Steuerungsprogramme verfasst. Dieses Tool mit dem Projektnamen *SynTACS* wird am Lehrstuhl Informatik 11 entwickelt. Es soll dazu dienen, Supervisor zu generieren, welche zum einen Befehle der Steuerung blockieren und zum anderen sicherheitsrelevante Aktionen eigenständig ausführen können, mit dem Ziel dem Auftreten bestimmter sicherheitskritischer Ereignisse vorzubeugen.

Zielsetzung

Ziel dieser Bachelorarbeit ist die Implementierung und Analyse eines Code-Generators als Bestandteil des Synthesetools *SynTACS*, welcher die Aufgabe besitzt ein Steuerungsprogramm mit einem entsprechend synthetisierten Supervisor zu verknüpfen und ausführbaren SPS-Code zu erzeugen. Dieser soll die Aktionen des ursprünglichen Steuerungsprogramms auf solche Weise einschränken oder erweitern, dass die Anforderungen des Supervisors zu jeder Zeit erfüllt werden.

Vorgehensweise

Zunächst werde ich im Bereich der Code-Generierung und SPS-Programmierung recherchieren, um ein besseres Verständnis der Sachlage zu erhalten und die Struktur der Implementierung auf bestehendem Wissen aufbauen zu können.

Daraufhin folgt die Implementierung des Code-Generators, welche in *Java* als Teil eines Eclipse-Plugins erfolgen wird. Als Programmiersprache des generierten Codes ist vorerst *Strukturierter Text* (ST) gemäß IEC 61131-3 vorgesehen. Hierzu werde ich in der weiteren Entwicklung noch evaluieren inwiefern die zusätzliche Verwendung von *Ablaufsprache* (AS) zur Vereinfachung oder *Anweisungsliste* (AWL) zur Effizienzsteigerung sinnvoll ist.

Im letzten Schritt der Arbeit werde ich generierten Code aus verschiedenen Varianten der Implementierung im Hinblick auf seine Effizienz untersuchen. Dies bezieht sich in erster Linie auf die Nutzung des Programmspeichers und die Zykluszeit der SPS.

Bearbeiter

Thomas Timmermanns

Betreuer: Florian Göbe
08.12.2014

Informatik 11
Embedded Software
RWTH AACHEN UNIVERSITY