Peketeinräumerding

Michael Thomas Andreas Glatz Simon Weitzel Felix Baral-Weber
Stand: 8. Juli 2016 16:00

Inhaltsverzeichnis

1 Einführung

1.1 Zweck des Dokuments

Dieses Dokument beschreibt die Anforderungen und Implementierungsdetails an eine Echtzeitsystemanwendung die das Abschlussprojekt des Moduls Echtzeitbetriebssysteme der EAH Jena. Dieses Dokument richtet sich dabei an den Vorgaben für ein Software Requirements Specification (SRS) nach dem IEE Standard 830-1998. Der Modulverantwortliche ist Oliver Jack aus dem Fachbereich Elektrotechnik und Informationstechnik.

2 Allgemeine Beschreibung

2.1 Aufgabenstellung und Umsetzungsansatz

Die Aufgabestellung des Semesterprojektes ist es eine Hochregallagersimulation unter zuhilfenahme eines Echtzeitbetriebssystems zu erstellen. Diese soll das gesamte System mit einer statisch gewählten Regaldimension sowohl steuern als auch simulieren. Das Simulationmodell umfasst ein Regallagersystem mit $5 * 10$ Regalfächern, ein Ein- und ein Ausgabeslot und einen Turm zum be- und entladen. Dieser Turm ist auf einer Achse montiert auf welcher er sich in X-Richtung bewegen kann, desweiteren kann der Ausleger am Turm in Y-Richtung herauf und herab und in Z-Richtung rein und raus gefahren werden. Diese Bewegungen und die daraus resultierende Position des Turms werden von Tastern auf der Schiene ermittelt.
Auf der X-Achse und auf der Y-Achse sind jeweils 10 Taster und auf der Z-Achse 3.

Dem Anwender steht als Eingabe- und Ausgabemedium die Konsole zur Verfügung.

Ihm stehen folgende Befehle zur Verfügung:

- getspace → Zeigt aktuelle Belegung des Hochregallagers an
- vsetspace[x][y] → Definiert einen Platz als schon belegt
- clearspace[x][y] → Definiert einen Platz als frei
- insert[x][y] → Holt ein Klötzchen vom Eingabe-Slot und legt es an gewünschter Position im Hochregallager ab
- remove[x][y] → Holt ein Klötuchen von der gewünschten Position ab und legt es an den Ausgabe-Slot

Sollte ein Befehl nicht möglich sein, da zum Beispiel ein Hochregallagerplatz oder der Ausgabe-Slot bereits belegt ist, wird der Anwender durch eine Fehlermeldung in der Konsole darauf aufmerksam gemacht und der Befehl nicht ausgeführt.
Die Visualisierung des Hochregallagers erfolgt ebenfalls in der Konsole, in welcher sowohl das Hochregal und dessen Belegung als auch die Position des Turms und dessen Auslegers, durch ASCI-Zeichen stilisiert dargestellt werden. Dabei liegt der Ursprung der Koordinaten und somit der Regalplatz (0,0) in der linken unteren Ecke. Die Darstellung des Ein- und Ausfahren des Auslegers wird unterhalb dargestellt. Dabei stellt die linke Position den zum Ein-/Ausgabe-Slot gefahrenen Arm da, die rechte Position die in das Regal hereingefahrene. TODO: Bild der Konsole einfügen

2.2 Entwicklungsumgebung und Programmierung

Als Entwicklungs- und Testumgebung wurde Windriver Workbench 3.3 genutz und die Programmiersprache C verwendet. Es wurden für das Echtzeitbetriebssystem Vxworks Libaries inkludiert.

3 Sp9ezifische Anforderungen

3.1 Funktionale Anforderungen

3.1.1 DFD1 Simulation

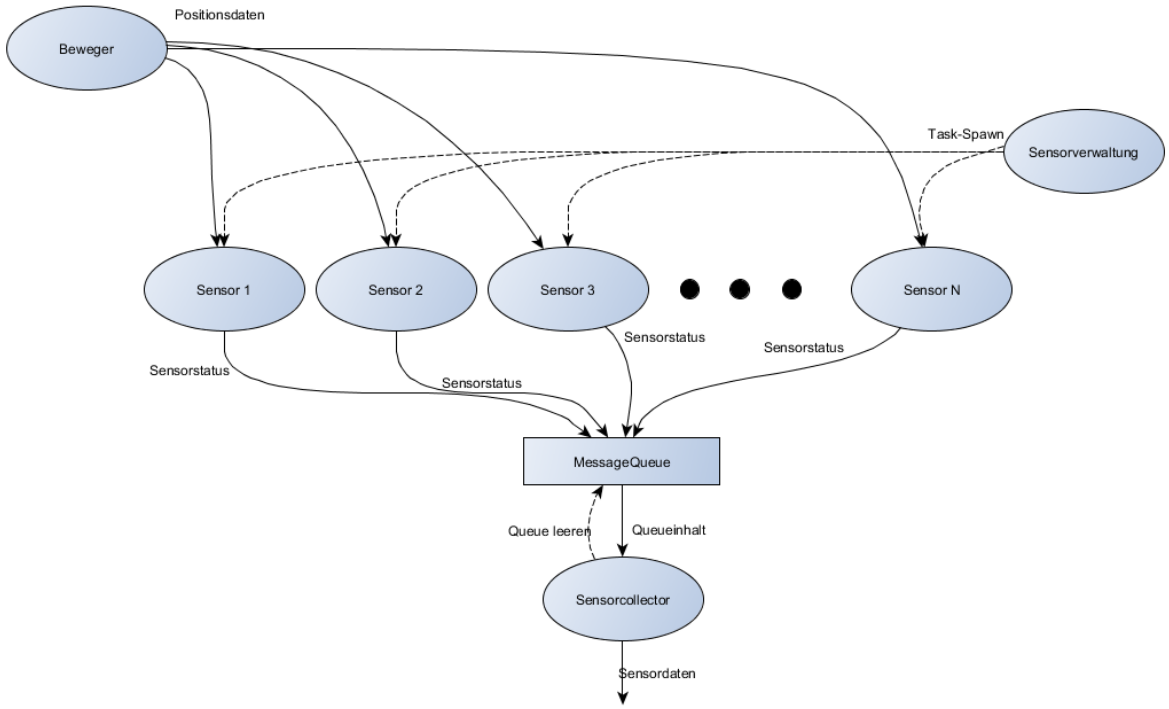

Abbildung 1: DFD1 Simulation

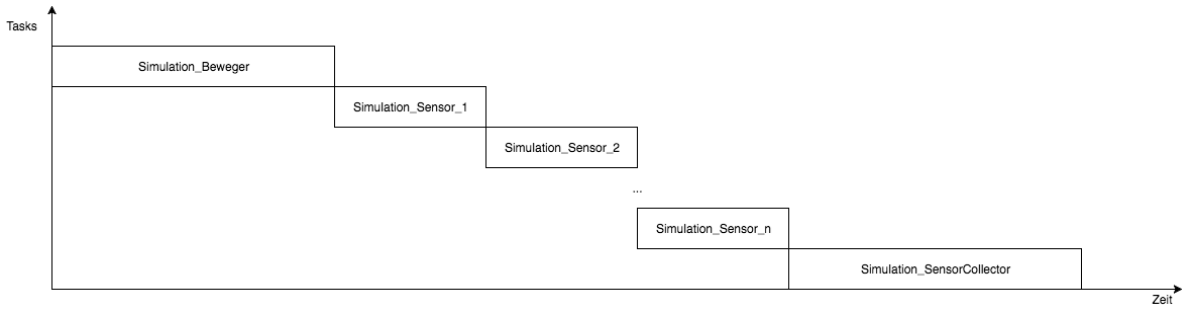

Abbildung 2: Gantt Diagramm der Task der Simulation

Beweger

Wie in Abb.2 zu sehen ist läuft der **Beweger** als erster Task eines Simulationsschrittes. Zuerst aktualisiert er die Aktorwerte aus der gloablen Variable **AktorBusData**.

Diese entspricht den Busdaten TODO: entspricht was? und ist durch einen Semaphor geschützt. Da die Simulation minimal blockiert werden soll implementiert der Semaphor Prioritätsvererbung. Daraufhin „verschiebt“er den Turm entsprechend den Aktordaten.

Sensorcollector

Der **Sensorcollector** sammelt aus der **MessageQueue** die Einträge aller Sensoren. Wenn ein Sensor ausfällt bleibt der Wert des Sensors auf 1.

Simulation Sensor x

Jeder Sensor bekommt eine id übergeben die seine Funktion und ihn eindeutig definiert.

1. **Tastsensoren** überprüfen ob die Turmposition mit der Sensorposition übereinstimmt.

2. **Lichtschranken** Simulieren entweder das eintreffen eines Klötzchens am Eingabeslot oder das Entfernen eines Klötzchens am Ausgabeslot, jeweils mit einem vordefinierten Delay. TODO: Das könnte besser sein

3. Der **Lichttaster** im Turm wird dann ausgelöst wenn die vorige y-Position des Turms im vorigen Schritt über (beim Ablegen eines Klötzchens) bzw. unter (beim Aufnehmen) der jetzigen Position ist und (beim Ablegen) ein Klötzchen im Turm ist bzw. (beim Aufnehmen) ein Klötzchen an der Aufnahmeposition ist.

Taster X[0-9]	[gedrückt|gehalten]
Taster Y[0-9]	[gedrückt|gehalten]
Taster Z[0-3]	[gedrückt|gehalten]
Lichtschranke Eingabe	[unterbrochen|nicht unterbrochen]
Lichtschranke Ausgabe	[unterbrochen|nicht unterbrochen]
Turmlichtschalter	[bedeckt|nicht bedeckt]

Tabelle 1: Requirements Dictionary