Modul-Design S88-Treiber

Für das studentische Projekt *Sichere Eisenbahnsteuerung*

Datum	17.06.2010
Quelle	Google Code → Dokumente → 02_Design → 02.02_Moduldesign
Autoren	Kai Dziembala
	Norman Nieß
Version	1.0
Status	freigegeben

1 Historie

Version	Datum	Autor	Bemerkung
0.1	24.11.2009	Jan-Christopher Icken	Initial
0.2	30.11.2009	Jan-Christopher Icken	Einleitung, Architektur, Referenzierte Dokumente
0.3	07.12.2009	Jan-Christopher Icken	Ports festgelegt, Zeiten für Abfragen festgelegt
0.4	14.12.2009	Jan-Christopher Icken	Kapitel 6 erstellt
0.5	21.12.2009	Jan-Christopher Icken	Anregungen aus dem Plenum eingefügt
0.6	29.04.2010	Kai Dziembala Norman Nieß	Kapitel 3 – 6 überarbeitet; Kapitel 7 entfernt; Layout-Anpassung
0.7	06.05.2010	Kai Dziembala Norman Nieß	Korrektur von Rechtschreibfehlern; neuer Dokumentenstatus: freigegeben
1.0	17.06.2010	Kai Dziembala	Freigabe des Moduldesign 'S88-Treiber'

2 Inhaltsverzeichnis

3 Einleitung

Um die an den S88-Rückmeldemodulen anliegenden Sensordaten auszulesen, wird ein S88-Treiber benötigt. Dieses Modul befindet sich in der Treiberschicht und stellt die Daten über den Shared Memory dem Modul Befehlsvalidierung zur Verfügung.

Außer der reinen Abfrage der Sensordaten, findet auch noch eine Validierung der ankommenden Daten statt.

Um die Rückmeldemodule abzufragen, ist es notwendig, ein Taktsignal (CLOCK) für deren Ansteuerung zu erzeugen. Des Weiteren müssen die Signale für RESET und LOAD erzeugt werden. Die Daten werden seriell auf der Leitung DATA eingelesen.

4 Referenzierte Dokumente

Software-Design:	Google Code → Dokumente → 02_Design → 02.01_Subsystemdesign → Software-Design
S88-Bus:	Aulis → F4 TI PROJEKT Bredereke WiSe0910 → Projektverzeichnis → Schnittstellendokumentation → S88-Bus

5 Architektur

5.1 Funktionshierarchie

Der S88-Treiber befindet sich in der Treiberschicht und greift auf einen definierten Speicherbereich im Shared Memory zu. Damit keine Daten anderer Module beschädigt werden, ist auf die Ansteuerung des Speichers besondere Aufmerksamkeit zu legen.

5.2 Daten

2 Byte Sensordaten (1 Bit pro Sensor)

Jedes Bit in den 2 Bytes steht für den Zustand eines Sensor wobei das MSB für den Sensor Nr. 16 und das LSB für den Sensor Nr.1 steht. Ein gesetztes Bit steht hierbei für ein Ereignis am Sensor, ein nicht gesetztes Bit dementsprechend für kein Ereignis am Sensor.

5.3 Abhängigkeiten von anderen Modulen

Damit der Software-Watchdog das Modul S88-Treiber eindeutig zuweisen kann, hat dieses die Modul-ID 3. Gibt das Modul die Ressourcen des Mikrocontrollers frei, obwohl eine Aufgabe noch nicht abgearbeitet ist, so wird dem Watchdog eine Nachricht mit einem Statusbyte gesendet, das den Fortschritt der Abarbeitung wiedergibt.

Dies geschieht über die Schnittstelle hello(byte module_id, byte status) des Watchdogs.

5.4 Beschreibung interner Funktionen

Interne Funktionen innerhalb des Modul sind:

- validate_sensor_data()
- write_sensor_data(int sensor_number)
- get_sensor_data()
- wait(byte times)

Die Funktion get_sensor_data() ruft die Daten von den S88-Rückmeldemodulen ab. Hierfür ist es notwendig, dass mehrere Signale erzeugt werden. Zur Erzeugung des Taktsignals wird die eigene Funktion 'wait(byte times)' genutzt.

Bei der Generierung des Taktsignals ist darauf zu achten, dass hierbei eine Taktperiode von 30µs nicht unterschritten wird. Um dies zu gewährleisten, ist eine Taktperiode von 40µs sinnvoll.

Die Abfrage des ersten Sensors, benötigt zwecks Generierung des Signals PS und Reset eine Gesamtzeit von ~280µs, jeder weitere Sensor benötigt ~40µs. Dadurch ergibt sich eine Gesamtlesezeit von ~860µs. (Quelle: http://www.digital-bahn.de/info_tech/s88.htm#timing)

Die Funktion 'validate_sensor_data()' dient zum Erkennen einer positiven Flanke der Hall-Sensoren. Ist dies der Fall, wird in dem Befehlsvalidierungs-Byte an entsprechender Stelle eine eins gesetzt.

Die Funktion 'write_sensor_data(int sensor_number)' speichert die Signale der Hall-Sensoren für das anschließende Validieren zwischen.

Name:	get_sensor_data()
Vorbedingung:	Aufruf der Funktion durch die BV: workS88()
Parameter:	keine
Rückgabe:	keine
Nachbedingung:	interner Aufruf der Funktionen: write_sensor_data(int sensor_number) validate_sensor_data()
Fehlerfall:	keine

Name:	validate_sensor_data()
Vorbedingung:	Sensoren müssen über get_sensor_data abgefragt worden sein
Parameter:	2 Byte Sensordaten
Rückgabe:	keine
Nachbedingung:	Zugriff auf die Parameter: S88_BV_sensordaten.Byte0 S88_BV_sensordaten.Byte1
Fehlerfall:	keine

Name:	write_sensor_data(int sensor_data)
Vorbedingung:	Aufruf der Funktion get-sensor-data()
Parameter:	2 Byte Sensordaten
Rückgabe:	keine
Nachbedingung:	keine
Fehlerfall:	keine

6 Dynamisches Verhalten

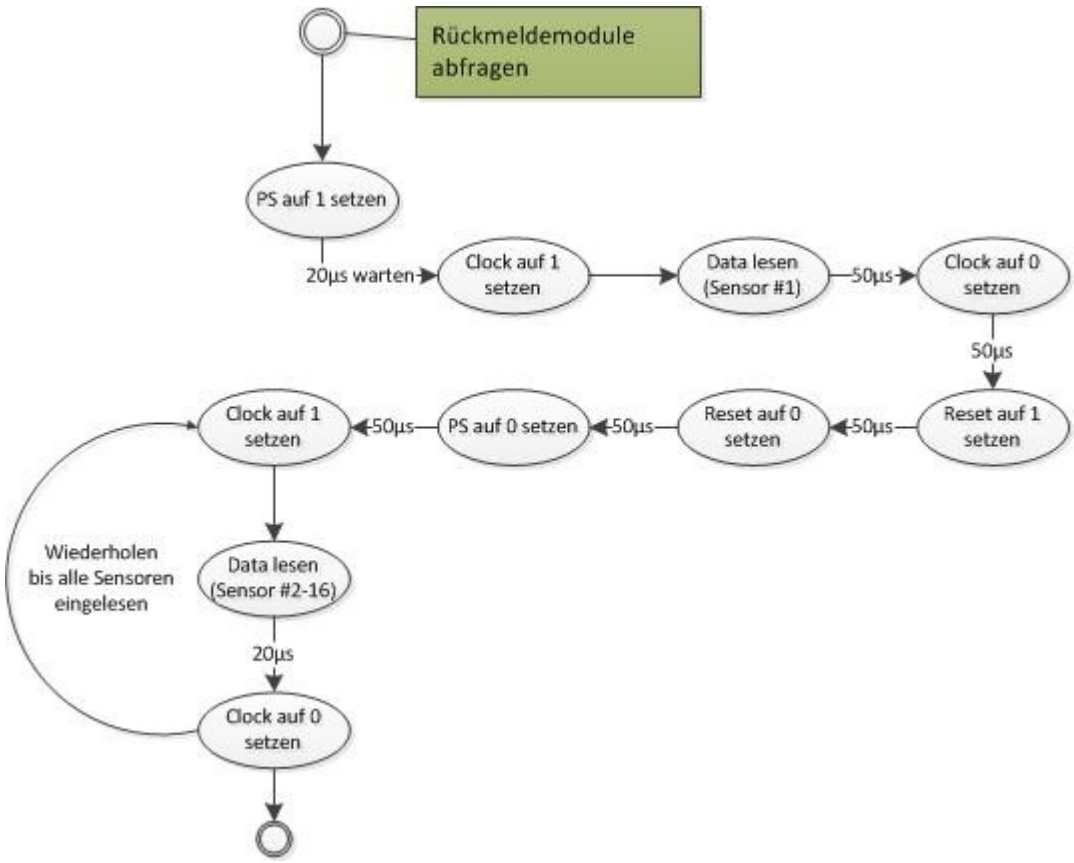

Abbildung 1: Rückmeldemodule abfragen