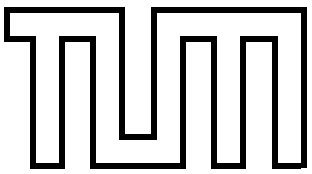


Technische Universität
München

Fakultät für Informatik
Forschungs- und Lehreinheit Informatik VI

Prozessrechner-Praktikum Echtzeitsysteme

Aufgabe 5 - Aufzugsteuerung

Matthias Regensburger
regensbu@in.tum.de

Christian Buckl
buckl@in.tum.de

Dr. Gerhard Schrott
schrott@in.tum.de

Sommersemester 2008

1 Aufgabenstellung

Für den im Praktikumsraum 03.05.012 dreifach aufgebauten Fischertechnik-Aufzug soll eine Einknopf-Umlaufsteuerung unter VxWorks implementiert werden:

Die Aufzugkabine bewegt sich solange in einer Fahrtrichtung, bis alle Anforderungen von außerhalb der Kabine (Stocktaster) und Fahrbefehle vom Innenraum der Kabine (Tastfeld) in dieser Fahrtrichtung erfüllt sind. Liegen keine Anforderungen und Befehle mehr vor, so wartet die Kabine bei offener Tür im letzten angefahrenen Stockwerk. Bei den Anforderungen an den Türen wird bei der Einknopf-Umlaufsteuerung die gewünschte Fahrtrichtung nicht berücksichtigt; die Kabine hält in jeder Fahrtrichtung.

Die beiden Aufzugsmodelle werden über den in Kapitel 2 näher beschriebenen CAN-Bus von den VxWorks-Targetrechnern "atknoll74-76" gesteuert (siehe Abbildung 1). Pro Aufzug gibt es 6 Tastfelder, die jeweils einen eigenen Mikrocontroller MC68HC11 besitzen. Um den CAN-Bus anzusprechen ist in jedem Rechner ein CAN-Controller Intel 82526 verbaut.

1. Befassen Sie sich mit dem in Kap. 2 beschrieben CAN-Bus und dem CAN-Monitor auf dem Windows-Rechner. Verschicken Sie mit dem Monitor einfache Steuernachrichten, wie z.B. "rote Lampe im Erdgeschoß einschalten", und verfolgen Sie die Bus-Aktivitäten bei Betätigung von Tastern am Aufzug.
2. Laden Sie die Dateien aus `atknoll133/Praktikum/PCI200/` in Ihr Verzeichnis und studieren Sie das bereitgestellte Programm `Tx_Rx_Frame_Test.c`. Testen Sie diesen Prozess durch einfache Steueraufgaben und beobachten Sie die CAN-Bus-Aktionen mit dem Monitor.
3. Schreiben Sie eine Empfange-Task, welche Botschaften des CAN-Bus empfängt, dekodiert und verwaltet.
4. Vervollständigen Sie Ihr bisheriges Programm, so dass die oben genannte Funktionalität einer Einknopf-Umlaufsteuerung erreicht wird.
5. Geben Sie das kommentierte Programmlisting im DIN A4-Format geheftet ab.

2 Der CAN-Bus

CAN steht für Controller Area Network und ist ein serielles Bussystem, das ursprünglich für den Einsatz in Kraftfahrzeugen entwickelt wurde, aber inzwischen aufgrund seiner Eigenschaften auch in der industriellen Automatisierungstechnik immer stärker eingesetzt wird. In KFZs gehen seine Einsatzgebiete von einfachen Steuerungen, wie z.B. der Scheibenwaschanlage bis hin zum Motormanagement, ABS oder der Katalysatorsteuerung.

In den internationalen Standardvorschlägen ISO/DIS 11898 und ISO/DIS 11519-1 sind die physikalische Schicht sowie die Sicherungsschicht des CAN-Protokolls spezifiziert. Die

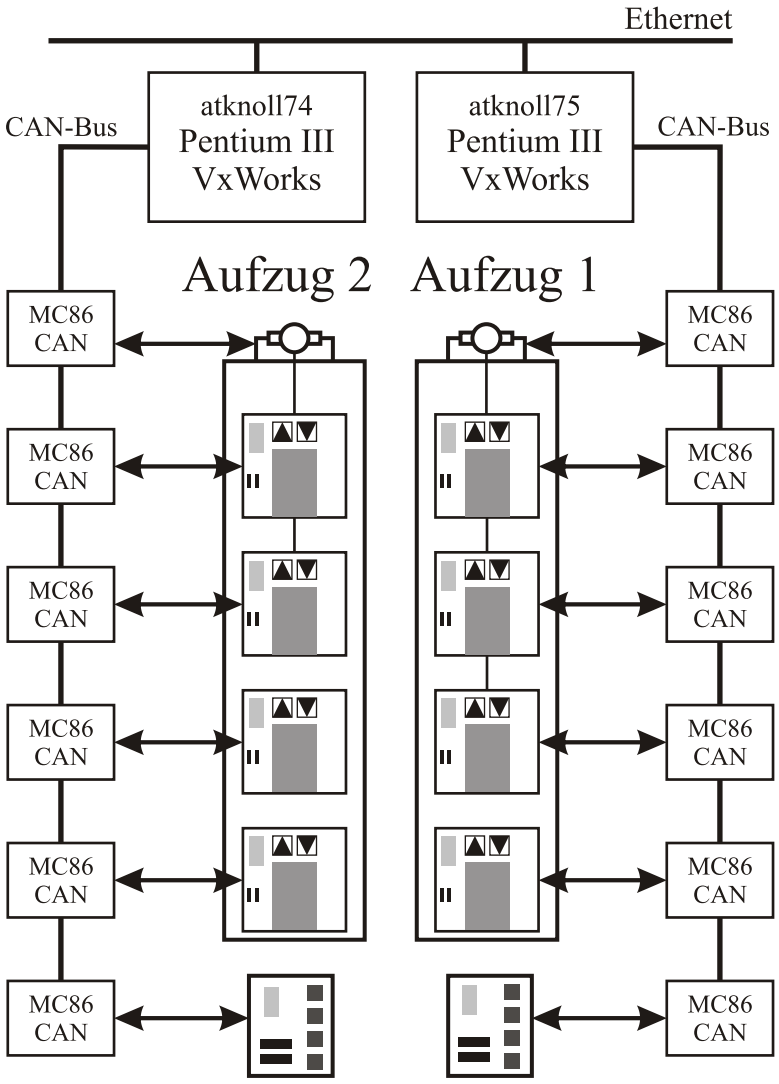


Abbildung 1: CAN-Bus-Struktur für Doppelaufzug

Übertragungsrate kann demnach zwischen 1MBit/s bei 40m Buslänge und 50kBit/s bei 1000m Buslänge variieren. Die Signalübertragung erfolgt über eine Zweidrahtleitung nach RS-485.

Über den CAN-Bus werden Objekte mit einer genau definierten und normierten Objektstruktur übertragen. Ein solches Objekt besitzt zwei Byte mit Statusinformationen für dieses Objekt und bis zu acht Byte Nutzinformation (vgl. Abbildung 2). Ein CAN-Objekt kann also maximal 10 Byte groß sein. Die ersten 11 Bit der Statusinformation bilden den Objekt-Identifikator. Demnach sind 2048 verschiedene *Identifikatoren* möglich. Das 12. Bit bestimmt, ob es sich um einen *Transmit* oder einen *Remote Transmit Request* handelt und die verbleibenden 4 Bit der Statusinformation ergeben die Länge der Nutzdaten.

Über den CAN-Bus werden beim Versand von Botschaften keine Empfangs- und Sendestationen adressiert. Jede Station kann jede über den CAN-Bus laufende Botschaft empfangen.

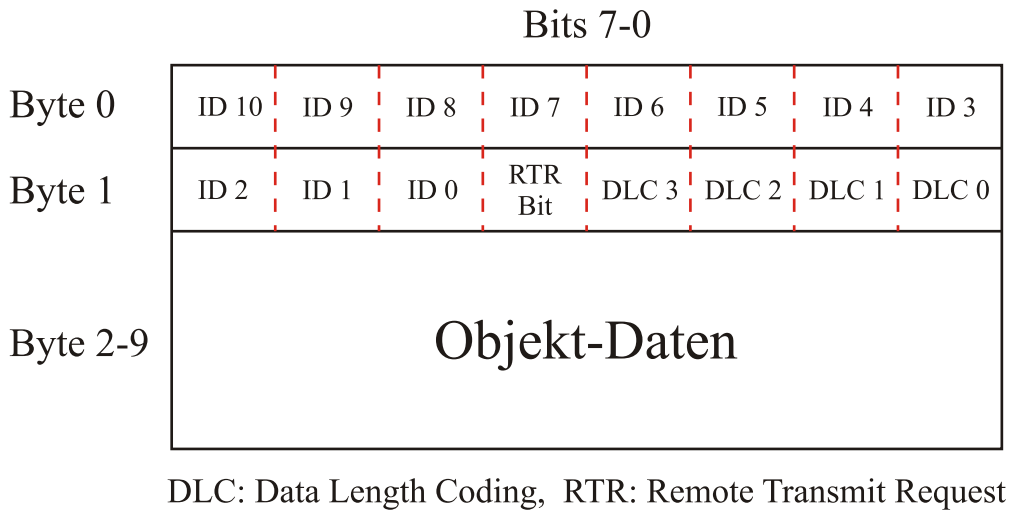


Abbildung 2: Aufbau eines CAN-Objekts

Der Absender einer Botschaft ist unbekannt. Lediglich der Identifikator des Objekts auf dem Bus ist relevant. Der CAN-Bus ist somit rein botschaftsorientiert !
Stationen können Datenobjekte auch nur bereitstellen, sie also nicht sofort übertragen. Durch einen empfangenen *Remote Transmit Request*, der keine eigenen Daten enthält (enthalten darf) aber den gleichen Objektidentifikator trägt, wird nun diese Station veranlaßt das geforderte Objekt zu übermitteln.
Auf der CAN-Bus-Leitung wird zwischen dominanten und rezessiven Bits unterschieden. Ein nicht gesetztes Bit wird als dominanter Pegel übertragen und ein gesetztes Bit als rezessiver Pegel. Bei Zugriffskonflikten auf dem Bus können rezessive Pegel durch dominante überlagert werden, umgekehrt jedoch nicht.
Nach der Definition des CAN-Protokolls besitzt jede Botschaft einen eindeutigen Identifikator. Dieser Identifikator stellt beim Buszugriff nicht nur den Namen der Botschaft dar, sondern auch die Priorität der Botschaft.
Falls zwei oder mehrere CAN-Controller gleichzeitig versuchen, ein Objekt über den Bus zu senden, wird der Konflikt durch bitweise Arbitrierung gelöst. Die Sendelogik jedes Transmitters vergleicht die übertragenen Bitpegel mit den Bitpegeln auf dem Bus. Ist ein rezessives Bit auf dem Bus in ein dominantes Bit umgeschlagen, so bedeutet dies, dass gerade ein anderes Objekt mit höherer Priorität auf dem Bus übertragen werden soll. Die Sendelogik unterbricht hier den eigenen Übertragungsvorgang, bis das Objekt mit höherer Priorität gesendet wurde. Der Identifikator mit dem niedrigsten Binärwert hat so die höchste Priorität.

Aber Achtung: Die Arbitrierung erstreckt sich nur über die ersten 12 Bits. Das heißt, sollten zwei Stationen Objekte mit gleichem Identifikator aber unterschiedlichem Inhalt senden, kann dies zu Fehlern führen, die nicht aufgelöst werden können. Es wird ein Fehlerprotokoll initiiert, das die eben erfolgte Übertragung für ungültig erklärt. Beide Stationen versuchen

nun anschließend ihre Nachricht zu wiederholen, was folglich wieder scheitert. Es liegt daher in der Verantwortung der Anwender solche Kollisionen zu vermeiden. (z.B. keine zwei *Sende*stationen dürfen Objekte mit gleichen Identifikatoren besitzen.)

3 Implementierungshinweise

3.1 Zuteilung der Objektidentifikatoren zur Steuerung der Aufzugfunktionen

Die Nachrichten sprechen die Register der CAN-Platinen über eindeutige Identifikatoren an. Jede Nachricht enthält ein Byte Information, den Wert des entsprechenden Registers. Die Zuordnung der Identifikatoren erfolgt nach folgendem Schema aus drei Hex-Ziffern <*aufzug*><*platine*><*register*>. Um z.B. einen Wert in das Register für die Anzeigen im Stockwerk 2 des 1. Aufzugs zu verschicken, verwendet man den Identifikator 16#132#.

1. **Hex-Ziffer <*aufzug*>: Unterscheidung der Aufzüge**

Aufzug 1	<*aufzug*>=1
Aufzug 2	<*aufzug*>=2
Aufzug 3	<*aufzug*>=3

2. **Hex-Ziffer <*platine*>: Bezeichner für die Platinen der Aufzüge**

Kabinenmotor	<*platine*>=0
Erdgeschoss	<*platine*>=1
1. Stock	<*platine*>=2
2. Stock	<*platine*>=3
3. Stock	<*platine*>=4
Tastfeld	<*platine*>=7

3. **Hex-Ziffer <*register*>: Bezeichner für die E/A-Register der Platinen**

Taster	<*register*>=0
Motor	<*register*>=1
Anzeigen	<*register*>=2
LED-Anzeige	<*register*>=3

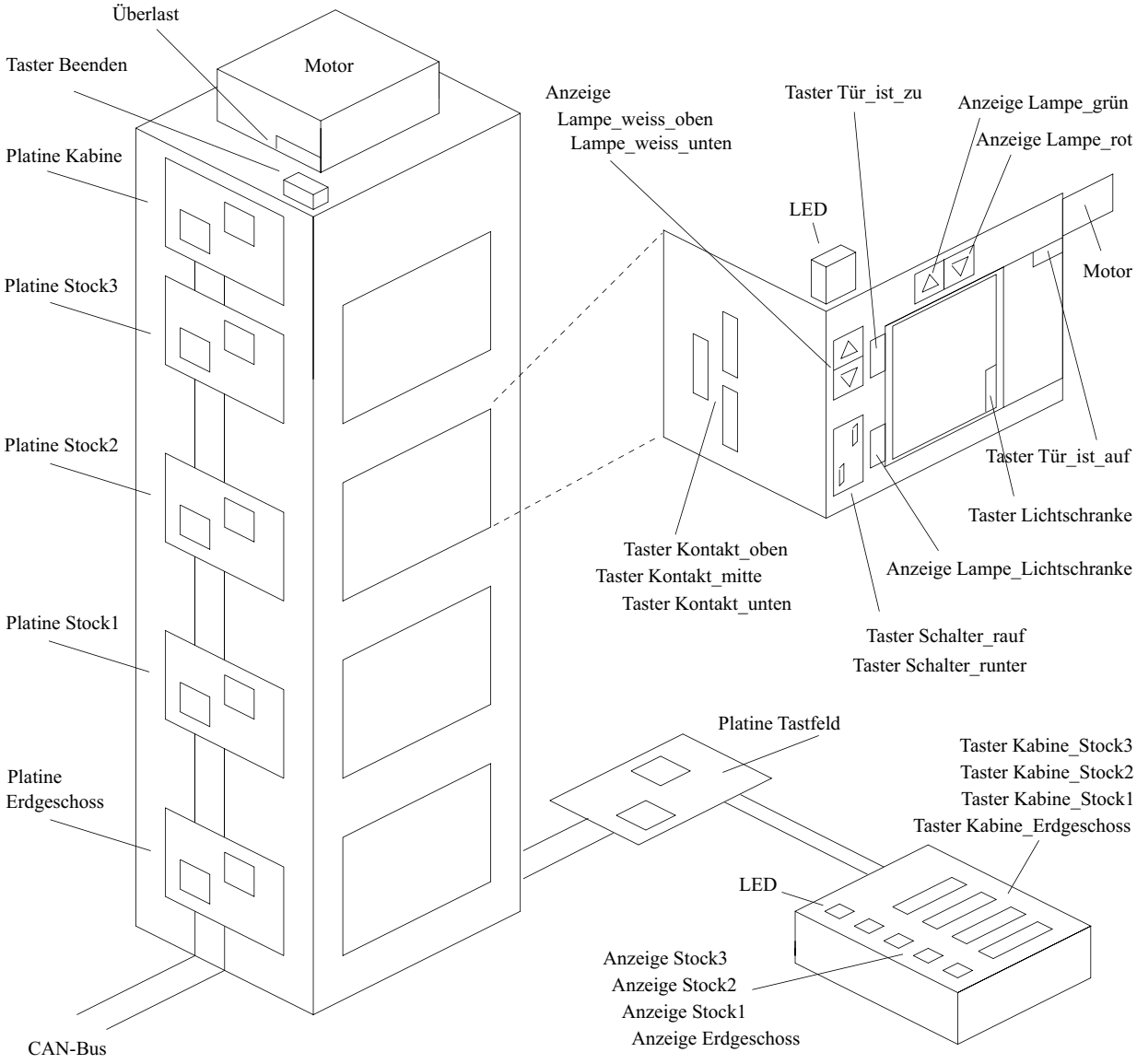


Abbildung 3: Schematische Darstellung des Modell-Aufzugs

3.2 Belegung der E/A-Register der CAN-Platinen

Zum Schalten der Motoren bzw. Lampen muss eine CAN-Nachricht mit dem entsprechenden Identifikator und einem Datenbyte gesendet werden; das Bitmuster des gesendeten Bytes steuert die einzelnen Motoren und Lämpchen. Jede Veränderung der Eingabewerte der Taster bzw. Lichtschranken bewirkt, dass eine Nachricht mit dem entsprechenden Identifikator über den CAN-Bus geschickt wird, die ein Datenbyte mit allen aktuellen Werten enthält. Dabei wird nicht nur das geänderte Bit gesendet, sondern das Abbild sämtlicher Eingabewerte der zugeordneten Platine. Es gilt folgende Zuordnung der E/A-Register zu den Aktoren und Sensoren des Aufzugs. In Abbildung 3 ist die Bedeutung der einzelnen Aktoren und Sensoren im Aufzugsmodell schematisch dargestellt:

3.2.1 Platine für den Kabinenmotor:

Register	Identifikator	Hex-Wert	Aktor/Sensor
Taster	16#<*aufzug*>00#	1	Überlast
		4	Beenden
Motor	16#<*aufzug*>01#	1	Kabine rauf
		3	Kabine langsam rauf
		5	Kabine runter
		7	Kabine langsam runter

3.2.2 Platinen für Stockwerke (<*platine*>=1,2,3,4):

Register	Identifikator	Hex-Wert	Aktor/Sensor
Taster	16#<*aufzug*><*platine*>0#	1	Lichtschranke
		2	Tür ist auf
		4	Schalter rauf
		8	Schalter runter
		10	Tür ist zu
		20	Kontakt oben
		40	Kontakt mitte
		80	Kontakt unten
Motor	16#<*aufzug*><*platine*>1#	1	Tür auf
		5	Tür zu
Anzeigen	16#<*aufzug*><*platine*>2#	1	Lampe grün
		4	Lampe rot
		10	Lampe weiß oben
		40	Lampe weiß unten
		80	Lampe Lichtschranke

3.2.3 Platine für Steuerung in der Kabine:

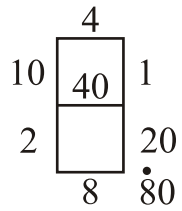
Register	Identifikator	Hex-Wert	Aktor/Sensor
Taster	16#<*aufzug*>70#	1	Fahrziel 3. Stock
		2	Fahrziel 2. Stock
		4	Fahrziel 1. Stock
		8	Fahrziel Erdgeschoß
		10	Alarm
		20	Schalter Nothalt
		40	Schalter Innensteuerung
Anzeigen	16#<*aufzug*>72#	1	3. Stock
		4	2. Stock
		8	weisse Lampe unten
		10	1. Stock
		20	Alarmhupe
		40	Erdgeschoß
		80	rote Lampe

3.2.4 Beispiel:

Bekommt man aus denm CAN-Bus folgende Nachricht: 16#130# mit Hex-Wert: 54, bedeutet dies, dass der Fahrstuhl 1 in Mittelstellung, bei geschlossener Tür ist und „Schalter runter" gedrückt ist.

3.2.5 LED-Anzeige:

Die LED Anzeige erhält in jedem Stockwerk und in der Kabine den einheitlichen Identifikator 16#<*aufzug*>73#.
Bitzuordnug zu den Segmenten der Anzeige:



3.3 Weitere Hinweise

- In Ihrem Projektverzeichnis muss zum Zugriff auf den CAN-Treiber die Datei *PCI200.c* enthalten sein.
- Es ist ratsam, eine Initialisierungs-Prozedur zu schreiben, die den Aufzug bei Programmstart in einen definierten Anfangszustand überführt.
- Um Ihr Programm korrekt zu verlassen, verwenden Sie bitte den Beenden-Taster oben-links am Aufzugmodell. Wenn Ihr Programm die Betätigung dieses Tasters feststellt, sollten sie sicherstellen das keiner Ihrer Aktoren – vor allem Motoren – noch in Betrieb ist. Sie sollten diese abschalten (Hex-Wert 0), bevor Sie Ihr Programm endgültig verlassen.

3.4 Achtung:

Gehen Sie bitte vorsichtig und behutsam mit dem Modell um. Bei Hardware–Problemen informieren Sie bitte einen der Praktikumsbetreuer und probieren nicht, die Probleme selbst zu beheben. Lassen Sie die Aufzüge nur in Anwesenheit eines Praktikumbetreuers laufen.