

Eigenschaften mobiler und eingebetteter Systeme:

CAN Programmierung in der Praxis

EMES

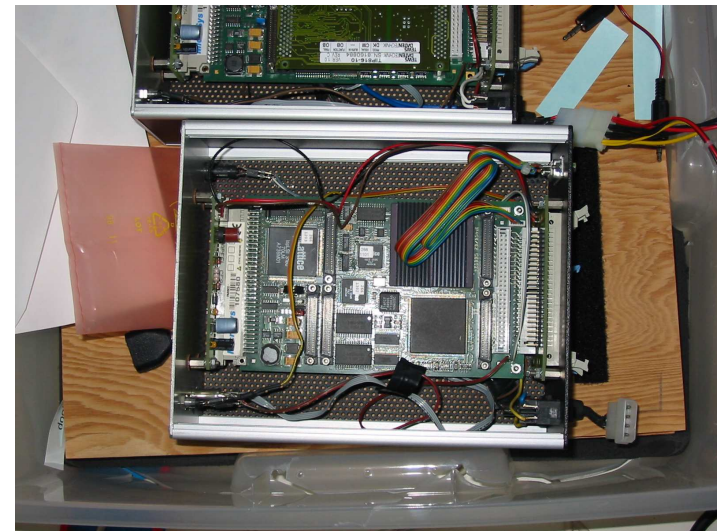
Dipl.-Inf. Jan Richling
Jens-Martin Loebel
Wintersemester 2002/2003

Übersicht

- Eingebettete Systeme - Theorie und Praxis
- CAN Programmierung
 - Die Technik
 - Die Anbindung
 - Die Fallstricke
 - Beispiele

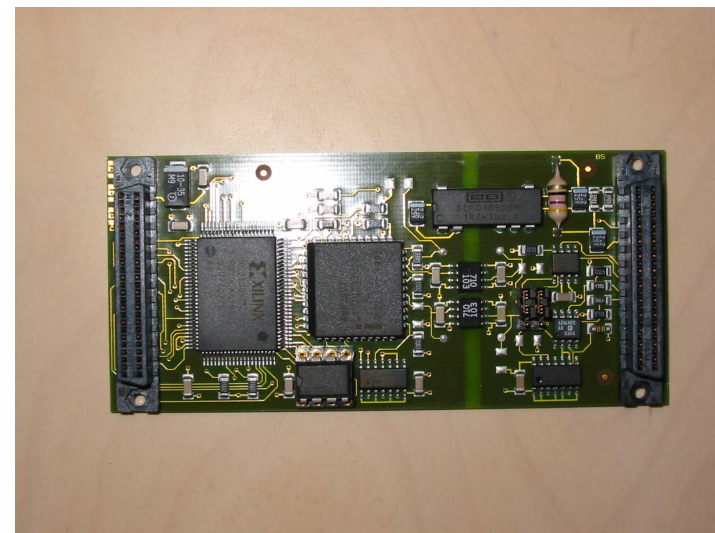
MicroSys Entwickler Boards

- IP 460 Board, für Prototypenbau
 - IP = Industry Pack
- 68040 Prozessor, kein OS
- 2 IP Steckplätze
- Direkte Pinouts
- Ausgänge für 2x RS232
- Reset Taster
- 4 MB Speicher
- Serieller Bootloader



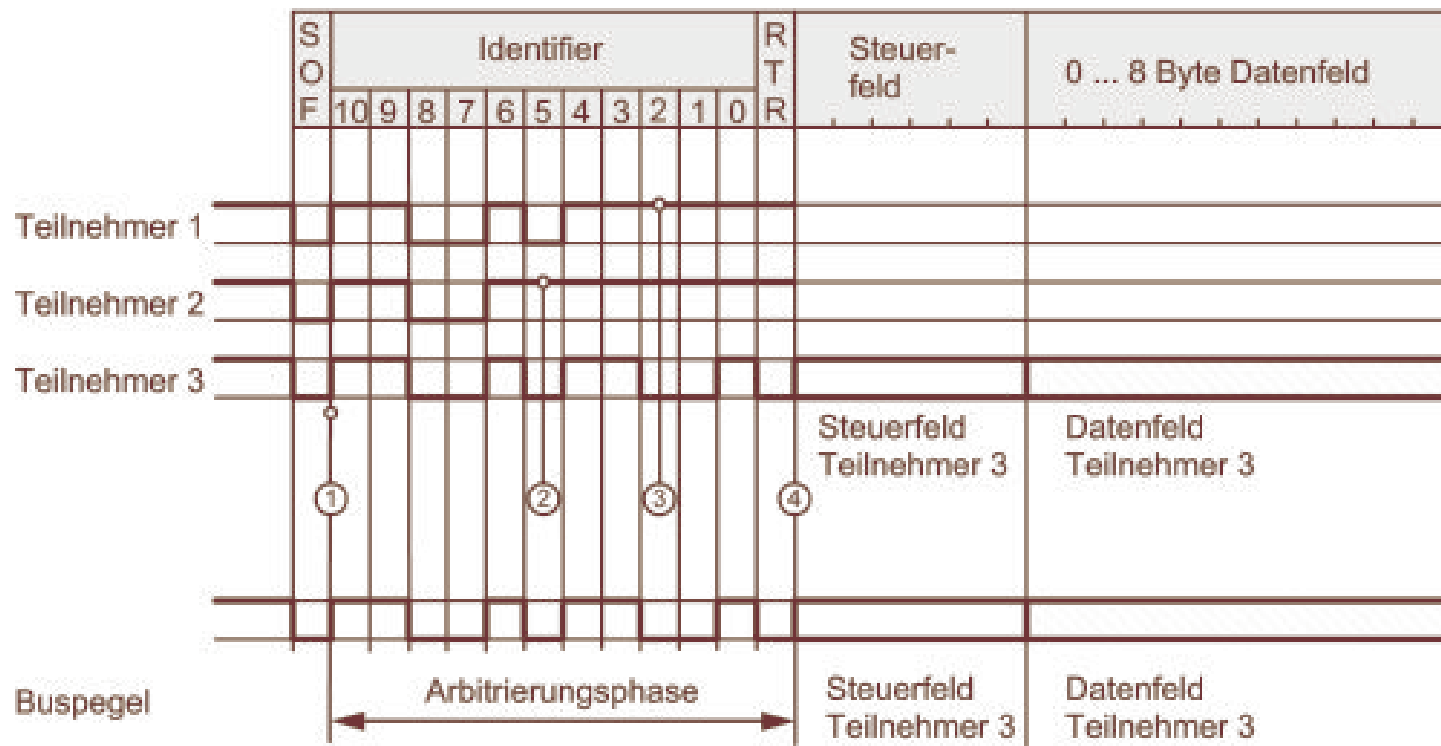
TEWS CAN Modul

- Intel 82527 Controller
- IP Modul Format
- Bis 1,6 Mbit (Abweichung vom Standard!)
- Direkte Bus-Anbindung
- CAN High Speed / mod. CAN
- Register Zugriff



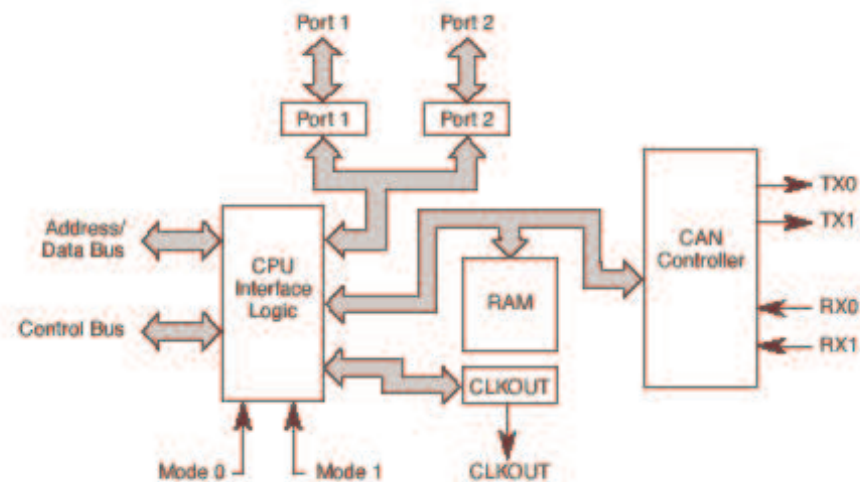
CAN (Wiederholung)

- Prioritäten-basiertes Protokoll bit 1 MBit
 - Gantierte minimale Antwortzeit
 - Arbitrierung, Nachrichten werden adressiert



Intel CAN Controller

- Unterstützt CAN 2.0
- Programmierbare Globale Maske
 - Standard (11 bits) / Extended Identifier (29 bits)
- 15 Message Objekte zur Kommunikation
 - 14 Rx/Tx Buffer
 - 1x Buffer mit eigener Maske
- 2 Ports
 - verbunden



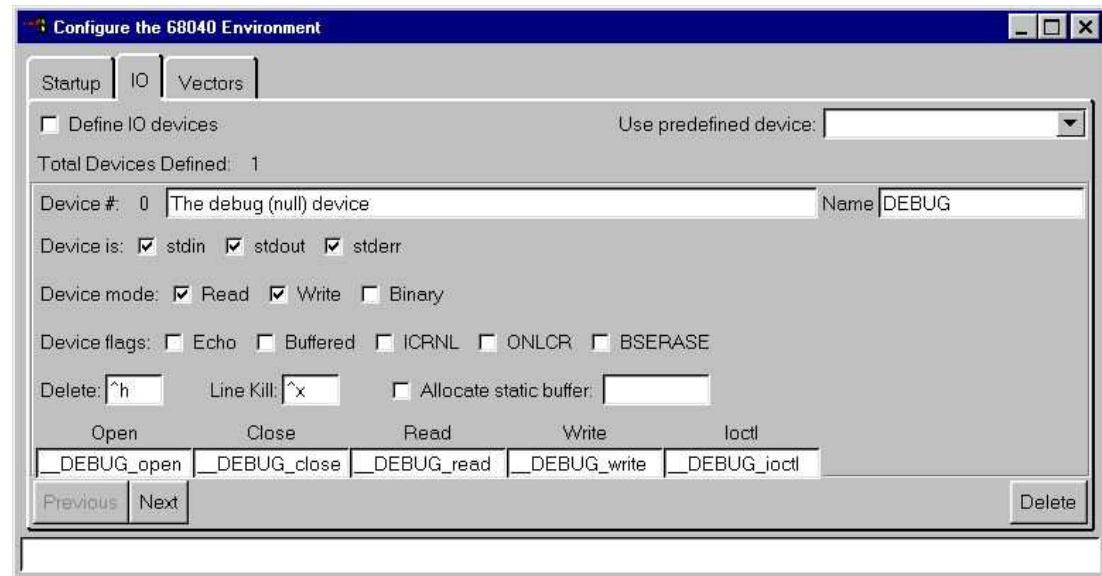
Dokumentation

- Offizielle Intel Spezifikation mit Update
 - Elektrische Eigenschaften
 - Pinouts
 - Register
 - Timing Diagramme
 - Programmablaufpläne
 - Message Aufbau
- TEWS Dokumentation
 - Elektrische Eigenschaften
 - Pinouts
 - Speicherlayout

IDE?

- Introl Compiler
 - TCL/TK basierende IDE
 - Konfigurierbar für 68040
 - IDE ruft Kommandozeilen-Compiler auf
 - Intel HEX Converter

- Fazit
 - Inkompatibel
 - stürzt ab
 - unbrauchbar



CAN Anbindung

- Dokumentation
 - Verdrehter Klingeldraht
 - Maximale Länge abhängig von Geschwindigkeit (max 40 m)
 - Terminierung
- Anschluß
 - Kein Connector vorhanden --> Pinouts
 - Keine Spezialkabel
 - Terminierung

Schnittstellen zum Board

- Direktes Mapping in den Speicher
- Pinouts für IP Connector
- Reservierter Speicherblock
 - Register
 - Message Objekte
 - Daten
 - Direktes Bus Leseregister

Programmierung I

- CAN Status Register
- CAN Control Register
- CPU Interface
- High Speed Read

4.3 Status Register (01H)

7	6	5	4	3	2	1	0
BOFF	Warn	Wake	RXOK	TXOK	LEC2	LEC1	LEC0
r	r	r	r/w	r/w	r/w	r/w	r/w

4.2 Control Register (00H)

7	6	5	4	3	2	1	0
0	CCE	0	0	EIE	SIE	IE	Init
r/w	r/w	r	r	r/w	r/w	r/w	r/w

r – readable
w – writable

4.4 CPU Interface Register (02H)

7	6	5	4	3	2	1	0
RstST	DSC	DMC	PwD	Sleep	MUX	0	CEh
r	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Programmierung II

- Objekte 1-14
 - 2 control Register
 - Max 8 Datenbytes
 - Status Indikatoren
- Objekt 15
 - Zusätzliche Maske

Message Object Structure

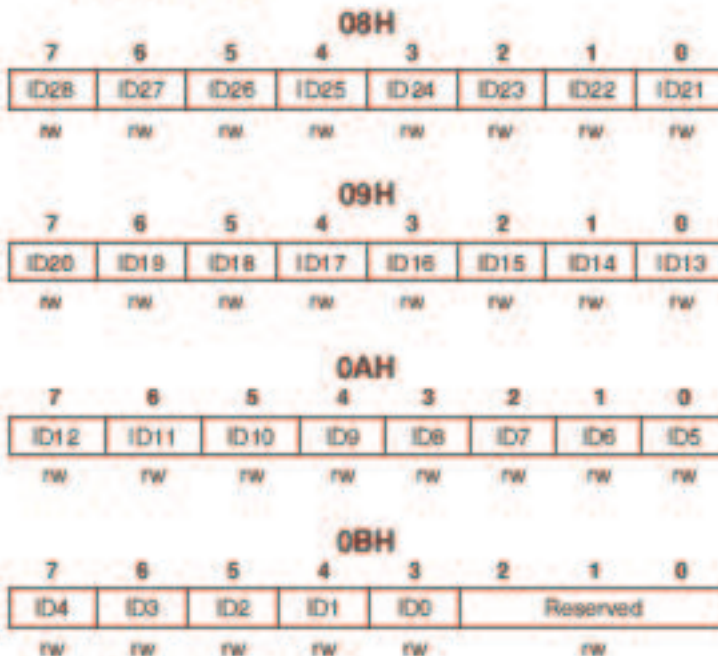
Base Address	+0	Control 0
	+1	Control 1
	+2	Arbitration 0
	+3	Arbitration 1
	+4	Arbitration 2
	+5	Arbitration 3
	+6	Mess. Conf.
	+7	Data 0
	+8	Data 1
	+9	Data 2
	+10	Data 3
	+11	Data 4
	+12	Data 5
	+13	Data 6
	+14	Data 7

Programmierung III

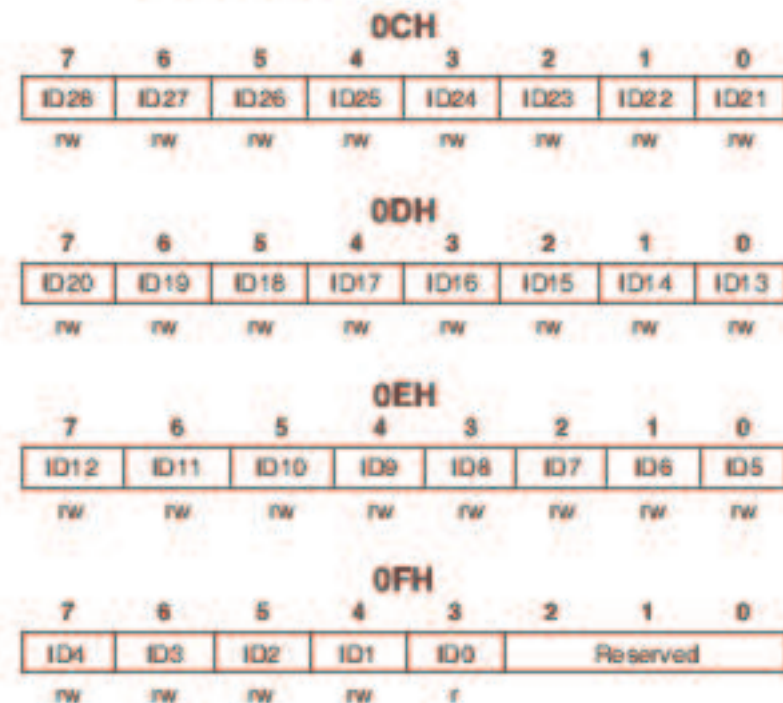
- Globale Masken

Lokale Maske

4.8 Global Mask—Extended Register (08–0BH)



4.10 Message 15 Mask Register (0C–0FH)



Theorie & Praxis

- Initialisierung
- Geschwindigkeit errechnen
- Bus Anschluß
- Message Arbitrierung

Programmierung Beispiel

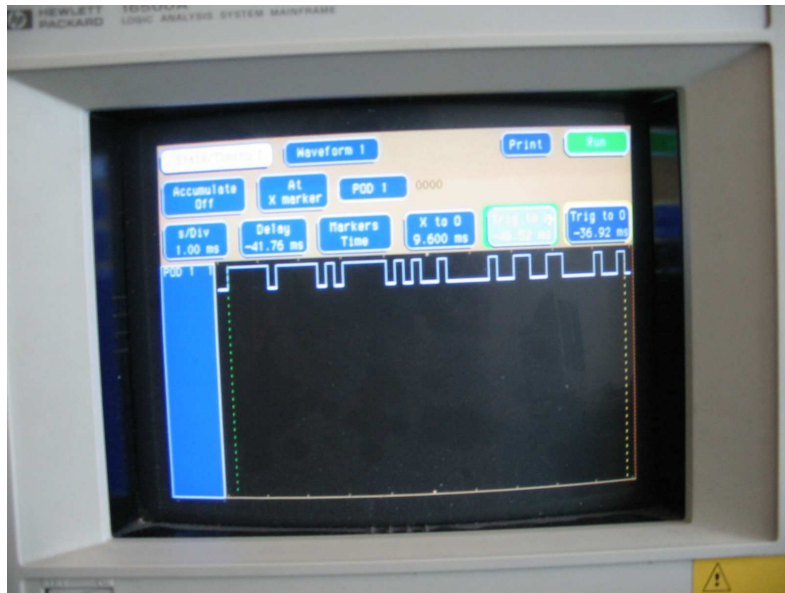
- Init
 - `writeb(0x07,CANStatusReg); writeb(0x00,CANCPUReg);`
`writeb((64+16+4+1),CANMessage1+1);`
`writeb(16,CANMessage1+6);`
- Empfangen
 - `if ((readb(CANStatusReg)&128)) while (1) printf("BUS`
`OFF!\r\n"); if ((readb(CANStatusReg)&&16) return 1;`

Debugging

- Schwer bei eingebetteten Systemen
 - Code wird nicht auf Entwicklungs-PC ausgeführt
- Lösung: Hardware Möglichkeiten des Systems nutzen
 - LEDs, RS232
- Verfahren
 - Interrupts
 - Single Step modus des Prozessors
 - Variablen überwachen
 - Testroutinen

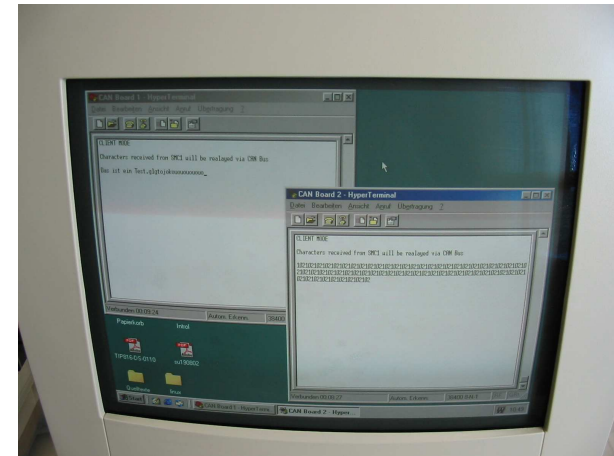
Fehlersuche mit Logic Analyzer

- HP Analyzer mit Touchscreen
- Messung von Bit Timing, Bus Status, Signalqualität



Fallstricke

- Orientierung an altem Treiber
 - (Speicherlecks, Interrupt Probleme)
- CAN falsch angeschlossen
 - (keine Fehlermeldung)
- Kommunikation mit Board
 - (kein Hardware Handshake)
- Pufferüberlauf
 - Empfohlene Einstellungen falsch



Problemsuche I

- Problemsuche am Beispiel:
 - Board stürzt sporadisch ab, Verhalten nicht reproduzierbar. Ein anderes Board meldet manchmal „Bus Error“ nach dem Upload / bei der Ausführung.
- Problemanalyse:
 - 1. Gedanke: Das Programm (CAN Treiber) ist fehlerhaft.
 - Schritte:
 - Programm debuggen
 - Variablen überwachen
 - Interrupts deaktivieren
 - Schlußfolgerung: Annahme falsch!

Problemsuche II

- Problemanalyse:
 - 2. Gedanke: Ein Hardware Fehler. Der Prozessor wird zu heiß.
 - Schritte:
 - Messung der Temperatur
 - Für Kühlung sorgen
 - Vergleich: Verhalten nach dem Einschalten / bei Dauerbetrieb

Schlußfolgerung: Annahme falsch!

Problemsuche III

- Problemanalyse:
 - 3. Gedanke: Spannungsversorgung zu niedrig oder Kurzschluß.
 - Schritte:
 - Spezifikation anschauen
 - notwendige Bedingungen für Reset feststellen
 - Logic Analyzer
 - andere Spannungsquelle verwenden

Schlußfolgerung: Annahme falsch!

Problemsuche Auflösung

- Rat einholen
 - 3 Daimler Ingenieure verzweifeln
- Entdeckung durch Zufall
 - Stromaufnahme des Boards zu hoch
 - „Notlösung“ durch Zusatzkabel



Zusammenfassung

- CAN Programmierung macht Spaß
- Die Welt ist eine Scheibe

E-mail an
Jens@WASTE.informatik.hu-berlin.de

