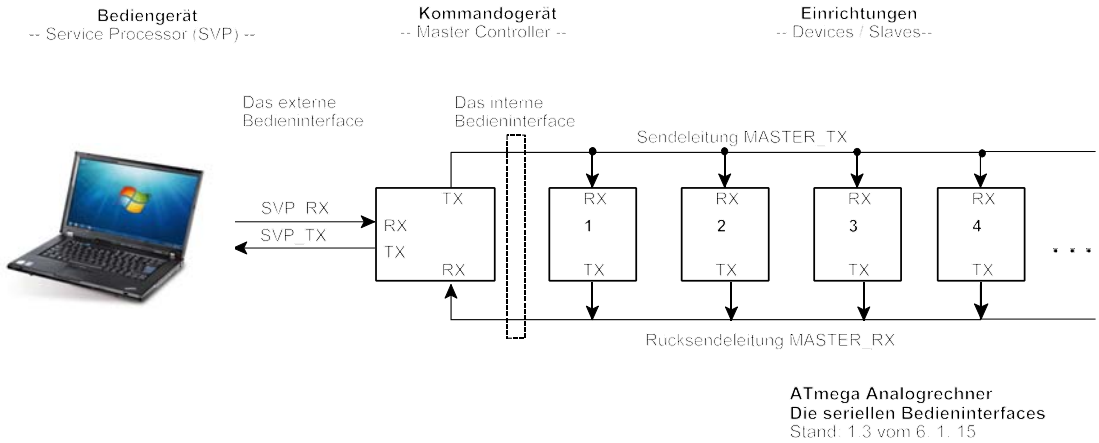
Projekt Analogrechner
Das interne Bedieninterface

Stand: 10. 3. 2015

1. Grundlagen

Das Bedieninterface nutzt die seriellen Schnittstellen der Mikrocontroller. Es gibt zwei Interfaces: das externe und das interne Bedieninterface. Jedes dieser Interfaces hat zwei Signalwege: die Sendeleitung (TX) und die Empfangsleitung (RX). Das externe Bedieninterface verbindet das externe Bediengerät mit dem Kommandogerät, das interne Bedieninterface verbindet das Kommandogerät mit den anderen Einrichtungen. Das externe Interface ist eine Punkt-zu-Punkt-Schnittstelle, das interne Interface ist eine Busschnittstelle. Alle anderen Einrichtungen empfangen die Daten vom Kommandogerät. Zu einer Zeit kann aber nur eine Einrichtung senden. Das Kommandogerät ist der Master, die anderen Einrichtungen sind die Slaves. Die Sendesignale der Einrichtungen werden nach dem Prinzip des Wired OR auf die Rücksendeleitung (MASTER RX) aufgeschaltet. Die Zeichen, die das Kommandogerät sendet, lösen in den anderen Einrichtungen Unterbrechungsbehandlungsroutinen aus. Alle Einrichtungen sehen alle Bytes, die vom Kommandogerät kommen. Nur das Kommandogerät sieht die Bytes, die die jeweils ausgewählte Einrichtung sendet.



Das Kommandogerät steuert das interne Bedieninterface nur dann an, wenn über das externe Bedieninterface entsprechende Kommandos gegeben werden. Das Kommandogerät ist dabei nur Durchreiche oder Signalwandler. Die eigentlichen Bedienfunktionen werden im externen Bediengerät implementiert. Wenn das Bedieninterface arbeitet, ruht das Kommandointerface und umgekehrt. Der Chef des Kommandogeräts ist das Bediengerät, der Chef der Slaves ist das Kommandogerät.

Die Signale des internen Bedieninterfaces:

Bezeichnung	Quelle	Nutzung
MASTER_TX	Kommandogerät	Sendedaten vom Kommandogerät, Empfangsdaten für alle anderen Einrichtungen
MASTER_RX	die jeweils ausgewählte Einrichtung (Wired OR)	Sendedaten von der ausgewählten Einrichtung, Empfangsdaten für das Kommandogerät
SVCH/FRAME#	Kommandogerät	reserviert

Das Signal SVCH/FRAME# dient als Zusatzsignal vom Kommandogerät. Es kann in den Controllern Unterbrechung auslösen.

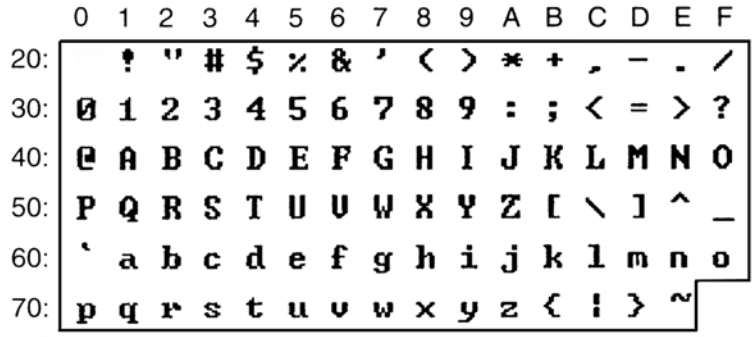
Hinweis: Die erste Implementierung nutzt SVCH/FRAME# nicht. Sie beschränkt sich ausschließlich auf das Übertragen von ASCII-Zeichen.

Typische Ursachen für das Entstehen von Bedienanforderungen sind:

- das Betätigen des Anforderungsschalters der jeweiligen Einrichtung,
- das Auftreten interner Bedingungen (z. B. Vergleichsstop),
- eine Bedienhandlung am externen Bediengerät.

Nur das Bediengerät kann Aktionen an den Bedieninterfaces von sich aus auslösen. Das Kommandogerät wird am internen Bedieninterface nicht autonom wirksam. Das Bediengerät kann das Kommandogerät veranlassen, Kommandos und Daten vom externen zum internen Bedieninterface (und umgekehrt) 1:1 durchzureichen (Direktsteuermodus).

Datenübertragung
Die Datenübertragung beruht auf dem ASCII-Code. Binäre Daten werden hexadezimal übertragen (mit zwei Hexadezimalzeichen je Byte). Weitere Kommandos unterstützen die direkte (uncodierte) Übertragung von ASCII-Zeichen. Einige ASCII-Steuerzeichen werden zur Übertragungssteuerung verwendet (reservierte Steuerzeichen). Sie dürfen nicht im Datenstrom vorkommen. Alle anderen ASCII-Steuerzeichen sind im Datenstrom an sich zugelassen. Über das interne Bedieninterface sollten aber nur darstellbare ASCII-Zeichen übertragen werden (ASCII-Codes 21H...7FH).



Reservierte Steuerzeichen des internen Bedieninterfaces:

Zeichen	Code	Wirkung
CR	0DH	Allgemeines Abschlußzeichen.
CAN	18H	Alle Einrichtungen rücksetzen und trennen (Puffer und Auswahlzustände).
EOT	04H	Den ausgewählten Slave rücksetzen, aber nicht trennen.
ENQ	05H	Maschinenfehlerreport senden.
ETB	17H	Abschließen und Trennen vom aktuellen Slave.
EM	19H	Hart rücksetzen.
BEL	07H	Allgemeinauswahl (Broadcast).
DC2	12H	Übergang in den Bedienbetrieb.
DC3	13H	Übergang in den Rechenbetrieb.
ACK	06H	Positive Antwort auf Anfrage (Anforderung anhängig) oder Kommandoende.
NAK	15H	Negative Antwort auf Anfrage (keine Anforderung anhängig) oder Kommandoende mit Fehlermeldung.

Übertragungsreihenfolge:

1. Hexadezimalzeichen: das erste Zeichen betrifft die höherwertige, das zweite die niederwertige Tetrade. C1H = 1100 0001B.
2. Wörter, Adressen usw.: das niedrigstwertige Byte wird als erstes übertragen (Little Endian).

Byteübetragung (8 Bits) mit zwei Hexadezimalzeichen:

1. Byte	2. Byte
Bits 7 ... 4	Bits 3 ... 0

Ein 16-Bit-Wort:

15 12	11 8	7 4	3 0

Die hexadezimalen Bytes:

1. Byte	2. Byte	3. Byte	4. Byte
7 4	3 0	15 12	11 8

Slave-Adressen:

- Die Adressen 00H und FFH dürfen nicht vergeben werden.
- Die Bitmuster der reservierten Steuerzeichen CR, CAN, EOT, ENQ, ETB, EM, BEL, DC1, DC1, DC2, DC3, DC4, ACK, NAK dürfen nicht vergeben werden.
- Demzufolge sind 256 – 13 = 243 Einrichtungsadressen zugelassen. Diese Adressen müssen fest zugeteilt werden.

Einrichtungstypen:
Jede Einrichtung hat einen bestimmten Typ. Er ist in einem Byte codiert. Dieser Typcode ist in den Einrichtungen fest gespeichert. Er kann über Kommandos abgerufen werden.

Slave-Adressen:

Adresse	Einrichtung
0	nicht genutzt bzw. kein Slave ausgewählt
1...20 = 01H bis 14H	Rechengerät (ein Controller unterstützt 2 Rechenfunktionen und hat demgemäß 2 aufeinander folgende Adressen)
F1H	Ausgabewandler, Hauptprozessor
F2H	Ausgabewandler, Hilfsprozessor
F5H	Eingabewandler, Hauptprozessor
F6H	Eingabewandler, Hilfsprozessor
F8H	Funktionsgenerator
FFH	reserviert

Einrichtungstypen:

Typcode (HEX)	Einrichtung
1x	Rechengerät. Aktueller Typ 10H
4x	Ausgabewandler. Aktueller Typ Hauptprozessor: 40H, Hilfsprozessor 41H
7x	Eingabewandler. Aktueller Typ Hauptprozessor: 70H, Hilfsprozessor 71H
Ax	Funktionsgenerator. Aktueller Typ A0H
Dx	Kommandogerät. Aktueller Typ D0H.

Daten im Speicher: Register, Objekte und Speicherbereiche

Die Einrichtungen am Bedieninterface sind verschiedenartig. Die Programme werden sich immer wieder ändern. Damit ist auch stets mit Änderungen der Speicherbelegung zu rechnen. Deshalb müssen die anwendungsseitigen Zugriffe von den konkreten Adressen in der Einrichtung unabhängig sein. Schreib- und Lesekommandos mit direkten Adressen sind eigentlich nur zu Debugging-Zwecken vorgesehen. Die Awendungsprogrammierung sollte ausschließlich mit Objektzugriffen arbeiten. Die wichtigsten Objekte sind vordefiniert und mit fest zugeordneten Zeichen auswählbar. Dies sind die Parameter, die Variablen, die Register sowie die Festwerte der Maschinenidentifikation. Zusätzlich können bis zu 255 Objekte anwendungs- und einrichtungsspezifisch definiert werden. Sie sind über Objektnummern (Object Identifiers) auswählbar, die in der Einrichtung über eine Objekttabelle (Object Reference Table ORT) in Adressen umgesetzt werden. Die Größe der ORT ist fest, die ORT-Inhalte können über Kommandos geändert werden.

Register

Wenn in diesem Text von Registern die Rede ist, sind damit Bytes oder Wörter im Speicher gemeint. Jedes Register ist über einen Identifier aufrufbar, unabhängig von seiner Placierung im RAM.

Parameter

Es werden bis zu 16 Parameter unterstützt. Jeder Parameter kann über eine Hexadezimalzahl (0...F) angesprochen werden, unabhängig von seiner Placierung im RAM. Die Anzahl der Parameter und die Parameterlänge werden über einen Festwertparameter vorgegeben. Parameter 0 ist stets der Funktionscode, der bestimmt, welche Funktionen in der Kommandoschleife und bei Ausführung des Kommandos “Funktionsauslösung” (X – X) auszuführen sind. Das niedrigstwertige Byte dieses

Parameters enthält den primären Funktionscode als Binärzahl[1]. Weitere Parameterbytes können ergänzende Codes, Steuerbits usw. enthalten. Funktionscode 00H bewirkt ganz elementare Abläufe zu Prüf- und Diagnosezwecken.

Variable
Es werden bis zu 16 Variable unterstützt. Jede Variable kann über eine Hexadezimalzahl (0...F) angesprochen werden, unabhängig von ihrer Placierung im RAM. Es gibt Eingangsvariable und Ausgangsvariable. Die Eingangsvariablen werden von 0 an gezählt, die Ausgangsvariablen folgen nach. Die Anzahl der Variablen insgesamt, die Anzahl der Eingangsvariablen und die Variablenlänge werden über Festwertparameter vorgegeben.

1. Beispiel: 6 Variablen insgesamt, davon 4 Eingangsvariable: die Eingangsvariablen haben die Identifier 0...3, die Ausgangsvariablen demgemäß die Identifier 4 und 5.

2. Beispiel: 5 Variablen und 0 Eingangsvariablen: Alle Variablen 0...4 sind Ausgangsvariable.

Objekte als frei definierbare Speicherbereiche
Die Objektdefinition erfolgt in der Objekttabelle (Object Reference Table ORT). Die maximale Anzahl der Objekte wird über eine Festwertparameter vorgegeben. Jedes Objekt wird über eine Objektnummer (Object Identifier) im Bereich von 1 bis 255 angesprochen (1 Byte oder 2 Hexadezimalzeichen). Objekt 0 ist reserviert. Die Objekttabelle enthält Objektdeskriptoren. Jeder Objektdeskriptor ist 5 Bytes lang. Er gibt die Anfangsadresse, die Länge und den Typ des Objekts an.

Ein Objektdeskriptor:

1. Byte	2. Byte	3. Byte	4. Byte	5. Byte
Adresse, Bits 7...0	Adresse, Bits 15...8	Länge, Bits 7...0	Länge, Bits 15...8	Typkennung

Über den Objekttyp könne auch schreibgeschützte Objekte und Objekte in anderen Speichern (Flash, EEPROM) definiert werden. Anfänglich wird nur Objekttyp 0 unterstützt (RAM, kein Schreibschutz).

Fehlerbehandlung
Fehler werden angezeigt (Blinken der LED) und registriert (Fehlerregister). Manche Fehler der seriellen Kommandoausführung werden durch Rücksenden von NAK gemeldet. Die Fehlerbehandlung obliegt dem Bediengerät. Fehlerregister werden nur über Kommandos gelöscht.

Fehleranzeige
Eine Einrichtung, die einen Fehler erkannt hat, läßt ihre LED schnell rot-grün blinken. Dieses Blinken hat Vorrang vor allen anderen LED-Signalen und kann nur mit Kommandos ausgeschaltet werden.

ACK und NAK
Diese Antwortzeichen dienen dazu, die Annahme oder Abweisung eines Zugriffs oder Kommandos anzuzeigen. NAK kann auch gegeben werden, um eine Fehlerbedingung zu signalisieren (Kommando wird abgebrochen oder nicht ausgeführt).

Fehlermeldung
Alle Einrichtungen können Fehler über Bedienanforderungen ans Bediengerät melden..

1: Damit dieser Code auf einfache Weise ausgewertet werden kann (Vergleichen mit Festwerten, Funktionsverzweigung).

Fehlererkennung seitens des Bediengerätes
Von den Fehlermeldungen der Einrichtungen abgesehen gibt es drei grundsätzliche Möglichkeiten der Fehlererkennung:

1. Durch Zeitkontrolle (Timeout).
2. Durch zyklisches Abfragen. So können im Rahmen von Bedienhandlungen ENQ-Kommandos gegeben werden, um die Fehlerregister abzufragen.
3. Durch Bedienereingriff (z. B. Auslösen der Fehlerbehandlung, wenn LEDs blinken und sonst nichts geschieht (Hängenbleiben)).

Maschinenzustände im Slave
Im Slave kann jeweils eine von zwei Grundsteuerschleifen laufen: die Kommandoschleife oder die Serviceschleife.

Die Kommandschleife wartet auf STROBE, die Serviceschleife auf die Anforderung, ein CR-Kommando auszuführen (Bedienanforderung; SERVICE REQUEST).

Das Kommandogerät schaltet hart zwischen beiden Schleifen um. Hierzu dienen Kommandos, die über das serielle Interface gegeben werden. Vom Umschalten sind alle Einrichtungen gleichzeitig betroffen. Es wird nur dann umgeschaltet, wenn sich die jeweils aktive Schleife im Warte-Umlauf befindet. Eine Kontextrettung beim Umschalten ist somit unnötig.

Die einzige Möglichkeit einer Einrichtung, eine Bedienanforderung anzuzeigen, besteht darin, das ATN-Signal auf Low zu ziehen. Zuvor muß ein entsprechender Anforderungscode in das Byte ATN_REQUEST geladen werden. Das ATN-Signal bleibt so lange auf Low, bis ein entsprechendes Ausschaltkommando empfangen wird.

Bovor der Slave ATN auf Low zieht, muß er den zugehörigen Anforderungscode speichern. Es sind 255 primäre Anforderungscodes möglich. Bedarfsweise können primäre Anforderungscodes mit zusätzlichen daten ergänzt werden. Solche Datenbereiche werden zweckmäßigerweise als Objekte organisiert.

ATN# und STOP#
ATN# dient zum Signalisieren von Bedienanforderungen. STOP# dient nur dazu, ggf. das Ende des Rechendurchgangs anzuzeigen. Debugging-Funktionen, wie beispielsweise Vergleichsstop, müssen somit über ATN# implementiert werden.

2. Kommandos am internen Bedieninterface

Kommandoübersicht:

Zeichen	Funktion	Erledigung
DC2	Übergang in die Serviceschleife (Bedienmodus). Serielle Puffer löschen. Anforderungen löschen. Auswahl löschen. Rücksendeleitung freigeben.	Unterbrechung. Gilt für alle
DC3	Übergang in die Kommandoschleife (Rechenmodus). Serielle Puffer löschen. Anforderungen löschen. Auswahl löschen. Rücksendeleitung freigeben.	Unterbrechung. Gilt für alle
BEL	Alle Einrichtungen als Slaves auswählen (Allgemeinauswahl, Rundempfang, Broadcast). Rücksendeleitung freigeben. Serviceschleife erzwingen.	Unterbrechung. Gilt für alle
CAN	Die serielle Eingabe initialisieren (Puffer und Kommandoanforderung löschen, Slaveauswahl löschen (Trennen)). Ähnlich DC2, aber keine Schleifenumschaltung. Kommandocode wird nicht gespeichert.	Unterbrechung. Gilt für alle.
EOT	Serielle Kommunikation im Slave rücksetzen, aber nicht trennen. Wirkt nur im ausgewählten Slave. Puffer und Kommandoanforderung löschen. Neuanlauf der Serviceschleife erzwingen. Kommandocode wird nicht gespeichert.	Unterbrechung
ETB	Löschen der Slaveauswahl (Trennen) und bestätigen. Die Rücksendeleitung wird freigegeben, nachdem ACK gesendet wurde. Serielle Kommunikation im Slave rücksetzen. Wirkt nur im ausgewählten Slave. Wird eine Datenübertragung mit ETB abgeschlossen, wirkt ETB wie CR, veranlaßt aber zusätzlich das Löschen der Slaveauswahl (Trennen) nach dem Senden des Abschlußzeichen.	Unterbrechung oder Serviceschleife (Kommandos mit Datenübertragung). ETB kann anstelle von CR gegeben werden, um nach der Kommandoausführung den Slave zu trennen.
EM	Totales Rücksetzen. Programmstart von Anfang an.	Unterbrechung. Gilt für alle
ENQ	Maschinenfehlerreport senden	Unterbrechung. Gilt nur im ausgewählten Slave
CR	Kommandos mit Datenübertragung	Serviceschleife. Gilt nur in ausgewählten Slaves. Kommandos mit Rückantwort nicht bei Allgemeinauswahl

Zurücksetzen der Zustände der seriellen Kommunikation:
Seriellen Puffer löschen (Inhaltszähler auf 0; Inhalt bleibt erhalten). Bedienanforderung aus.

Rückantwort:
Ist entweder nichts oder ACK (o.k., Bestätigung, Annahme) oder NAK (Abweisung). Ein Slave kann nur antworten, wenn er sich auf die Rücksendeleitung (MASTER RX) aufgeschaltet hat. Alle Kommandos, die nicht auf einen einzigen ausgewählten Slave wirken, können keine Rückantwort liefern.

Slaveauswahl:
Eine Einrichtung, die sich als nicht ausgewählt erkennt, muß eine ggf. vorhandene Auswahl rückgängig machen und die Rücksendeleitung freigeben.

Problem:
Der bisherige Slave muß die Rücksendeleitung freigeben. Dann erst darf der neue Slave aufschalten. Die Abläufe in den einzelnen Einrichtungen werden aber zu unterschiedlichen Zeiten wirksam (Latenzzeiten der Interruptauslösung, Abfrageschleife usw.). Wenn alles auf dasselbe Zeichen hin geschieht, kann es sein, daß die Einrichtung, die aufschalten will, eher am Interface wirksam wird als jene, die freigeben muß. Deshalb wird die Reihenfolge trennen – neu aufschalten über eine Programmiervorschrift erzwungen. Alle Einrichtungen müssen ein Trennzeichen gesehen haben, ehe die neue Auswahl kommt.

Programmiervorschrift:
Eine bestehende Slaveauswahl zuerst löschen (trennen), dann den neuen Slave auswählen. Trennen mit CAN oder ETB, entweder allein oder als Endezeichen eines Datenübertragungskommandos. Slaveauswahlkommandos dürfen nur dann gegeben werden, wenn kein Slave aktiv ist.

- CAN wirkt immer, sendet aber keine Bestätigung. Latenzzeit abwarten. Richtwert: 10 ms.
- ETB darf nur gegeben werden, wenn ein Slave ausgewählt ist. Dieser sendet eine Bestätigung (ACK).

Der serielle Puffer
Alle Zeichen, die nicht direkt als Kommdos interpretiert werden, werden in den seriellen Puffer eingetragen. Die Kommandos CR, ETB und EM werten den Pufferinhalt aus. Nach der Auswertung wird der Puffer gelöscht. Das Löschen erfolgt, indem der zugehörige Füllstandszähler auf 0 gesetzt wird. Der Pufferinhalt bleibt erhalten und kann ggf. analysiert werden. Richtwert der Puffergröße: 200 Bytes (Beschränkung mit Rücksicht auf die ATmega16 in den Rechengeräten[2]).

Direkte Schreibzugriffe
Manche Schreibkommandos umgehen den seriellen Puffer und tragen die Schreibdaten direkt in den RAM ein. Das Schreibkommando liefert zunächst die Anfangsadresse, die in den seriellen Puffer eingetragen wird. Diese Übertragung wird mit CR abgeschlossen. Die Einrichtung antwortet darauf mit ACK oder, im Fehlerfall, mit NAK. Dann werden die Schreibdaten geliefert. Die Übertragung wird mit CR oder ETB abgeschlossen.

3. Allgemeine Kommandos

DC2
Übergang in die Serviceschleife (Bedienmodus). Die Zustände der seriellen Kommunikation und die Slaveauswahl werden zurückgesetzt. Die Rücksendeleitung (MASTER_RX) wird freigegeben. Danach wird der Übergang in allen Einrichtungen erzwungen. Antwort: keine.

DC3
Übergang in die Kommandoschleife (Rechenmodus). Die Zustände der seriellen Kommunikation und die Slaveauswahl werden zurückgesetzt. Die Rücksendeleitung (MASTER_RX) wird ggf. freigegeben. Danach wird der Übergang in allen Einrichtungen erzwungen. Antwort: keine.

2: Der Pufferfüllstandszähler sind nur ein Byte lang. Deshalb Puffergröße nicht über 255 Bytes. Es gibt Lese- und Schreibkommandos, die beliebig viele Bytes übertragen können. Sie nutzen den Puffer nur zur Parameterübergabe und greifen ansonsten direkt auf den Speicher zu.

BEL
Alle Einrichtungen auswählen (Rundempfang, Broadcast). Die Zustände der seriellen Kommunikation werden zurückgesetzt. Die Rücksendeleitung (MASTER_RX) wird ggf. freigegeben. Alle Einrichtungen nehmen den BROADCAST-Auswahlzustand ein (BROADCAST, SLAVE SELECTED = 1, 1). Der Übergang in die Sserviceschleife wird erzwungen. Antwort: keine.

CAN
Die seriellen Puffer und die Bedienanforderungen in allen Einrichtungen löschen. Die Slaveauswahl wird zurückgesetzt. Antwort: keine. CAN dient vor allem dazu, ggf. den Anfangszustand der Zeicheneingabe herzustellen (gelöschter Puffer).

EOT
Serielle Kommunikation im Slave rücksetzen, aber nicht trennen. Das Kommando wird nur vom ausgewählten Slave ausgeführt. Die Zustände der seriellen Kommunikation werden zurückgesetzt. Die seriellen Puffer und die Bedienanforderungen werden gelöscht. Es wird ein Übergang an den Anfang der Serviceschleife erzwungen. Nutzung: Ggf. zur Fehlerbehandung (Wiederanlauf).

ETB
Serielle Kommunikation im Slave rücksetzen und trennen. Das Kommando wird nur vom ausgewählten Slave ausgeführt. Die Zustände der seriellen Kommunikation werden zurückgesetzt. Der Auswahlzustand wird gelöscht. Die Einrichtung ist nicht mehr ausgewählt. Es wird ein Übergang an den Anfang der Serviceschleife erzwungen. Nutzung: Zum Trennen des ausgewählten Slaves. Ggf. auch zur Fehlerbehandung (Wiederanlauf). Antwort: ACK. Nach dem Senden von ACK wird die Rücksendeleitung (MASTER_RX) freigegeben. ETB kann auch anstelle von CR am Kommandoende gegeben werden. Das Kommando wird dann wie ein CR-Kommando ausgeführt. Nach Ausführung des Kommandos wird aber die Slaveauswahl beendet (Trennen). *Hinweis:* Es wird immer getrennt, auch wenn NAK kommt.

ENQ
Maschinenfehlerlogout senden. Das Kommando wirkt nur im jeweiligen Slave. Der Slave sendet einen fest formatierten Maschinenfehlerreport (Logout). Jedes Byte wird mit zwei Hexadezimalzeichen übertragen. Das erste Byte gibt den Typ der Einrichtung an. Die Übertragung wird mit ACK abgeschlossen.

EM
Totales Maschinenrücksetzen. Das Kommando wirkt nur in ausgewählten Einrichtungen, also in allen, wenn zuvor ein BEL-Kommando gegeben wurde, oder im jeweiligen Slave. Die Einrichtung wird vollständig neu initialisiert. Das Kommando wirkt nur, wenn es im seriellen Puffer eine Sicherheitskennung vorfindet (Füllstand + besondere Zeichenfolge). Antwort: keine. Die erste Sicherheitskennung: s – l – a – v – e.

CR
Kommandos mit Datenübertragung. Das erste Datenzeichen ist der Kommandocode. Abhängig vom Kommandocode wirkt das Kommando in allen Einrichtungen oder nur im ausgewählten Slave. CR allein (ohne Daten im seriellen Puffer) wird nicht ausgeführt. Ein korrekt ausgeführtes CR-Kommando wird mit ACK abgeschlossen. wurde ein Fehler erkannt, wird das Kommando mit NAK beendet.

4. Kommandos zum Abfragen und Auswählen

r – ah – al – CR: sense request & release
Die ausgewählte Einrichtung schaltet sich auf die interne Rücksendeleitung auf. Ist keine Anforderung anhängig, sendet sie NAK und gibt die Rücksendeleitung wieder frei. Ist eine Anforderung anhängig, sendet sie zunächst das Ereignisbyte (EVENTS) in zwei Hexadezimalzeichen zurück. Dann sendet sie ACK und gibt die Rücksendeleitung wieder frei (Trennen).

Hinweis: Dieses Kommando ermöglicht es dem Bediengerät, die Anforderungen aller Einrichtungen abzufragen und dann zu entscheiden, welche Einrichtung zwecks Anforderungsbehandlung ausgewählt wird. Die Art der Anforderung geht aus der niederwertigen Tetrade (Bits 3...0) des Ereignisbytes (EVENTS) hervor.

g – ah – al – CR: sense request & grant
Die ausgewählte Einrichtung schaltet sich auf die interne Rücksendeleitung auf. Ist keine Anforderung anhängig, sendet sie NAK und gibt die Rücksendeleitung wieder frei. Ist eine Anforderung anhängig, sendet sie zunächst das Ereignisbyte (EVENTS) in zwei Hexadezimalzeichen zurück. Dann sendet sie ACK. Die Einrichtung bleibt ausgewählt und auf der Rücksendeleitung aufgeschaltet (kein Trennen). Damit ist sie zum aktuellen Slave geworden.

s – ah – al – CR: slave select
Die ausgewählte Einrichtung wird zum aktuellen Slave und schaltet sich auf die interne Rücksendeleitung auf. Sendet ACK.

p – ah – al – CR: slave presence detect
Die ausgewählte Einrichtung wird zum aktuellen Slave und schaltet sich auf die interne Rücksendeleitung auf. Dann sendet sie den Einrichtungstyp in zwei Hexadezimalzeichen zurück, sendet ACK und gibt die Rücksendeleitung wieder frei (Trennen).

Hinweis: Dieses Kommando ermöglicht es dem Bediengerät, die Maschinenkonfiguration zu ermitteln, indem es alle Einrichtungsadressen abfragt.

P – CR: slave poll
Die Anforderungen im aktuellen Slave werden abgefragt. Antwort: Anforderungsbytes + ACK. Es werden Bytes zurückgesendet, die die Anforderung näher kennzeichnen. Jedes Byte wird mit zwei Hexadezimalzeichen übertragen.

- Das erste Byte ist das Ereignisbyte (EVENTS).
- Ist eine ATN-Anforderung anhängig, folgt der Anforderungscode nach (ATN CODE).
- Ist eine STOP-Anforderung anhängig, folgt der STOP Code nach.
- Ist eine Maschinenfehlerbedingung anhängig, folgen drei Fehlerbytes nach: 1. Fehler der seriellen Schnittselle (OPCOM CHECKS), 2. interne Fehler (INTERNAL CHECKS), 3. Fehlercode (ERROR CODE).

Wenn nur eine Schalteranforderung anhängig ist, so kommt das Ereignisbyte allein zurück. In diesem Byte ist das Bit ATN SWITCH gesetzt.

q – Anforderungsnummer – CR: slave quit
Die von der Anoforderungsnummer (1 HEX-Zeichen) ausgewählte Anforderung wird gelöscht. Antwort: ACK, bei Anforderungsnummer 1 auch NAK.

Die Anforderungsnummern betreffen:

1: Schalteranforderung. Das Kommando sendet NAK zurück, wenn der Schalter noch betätigt ist. Die Anforderung wird erst dann gelöscht, nachdem der Schalter losgelassen wurde.
2: ATN-Anforderung + Anforderungscode.
3: Stopanforderung + Stopcode.
4: Maschinenfehlerbedingung + die drei Fehlerbytes.
F: Alles zuvor Genannte.

X – Kommandocode – Parameter – CR: Ein Steuerkommando ausführen (Execute)
Die Zeichenkette aus Kommandocode und (falls erforderlich) Parametern beschreibt ein Kommando, das im Kommandogerät selbst ausgeführt oder über das interne Bedieninterface ausgelöst wird. Die Anzahl der Parameter hängt vom Kommando ab. Manche Kommandos werden immer ausgeführt, manche nur unter bestimmten Bedingungen der Slaveauswahl. Kommandos, die das Zurücksenden von Daten bewirken, werden nur dann ausgeführt, wenn ein einziger Slave ausgewählt ist. Bei erfolgreicher Ausführung wird ACK gesendet, ansonsten NAK.

X-Kommandos:

Kommando	Wirkung
X – RST	Auslösen eines Hardwarerücksetzens in den anderen Einrichtungen. ACK kommt, nachdem der Rücksetzimpuls wieder inaktiv geworden ist.
X – X	Starten eines beliebigen Ablaufs, der über Parameter P0 (Funktionscode) ausgewählt wird.
X – x	Starten eines beliebigen Ablaufs. Parameter P0 dient dabei als Startadresse.

5. Kommandos zum Schreiben und Lesen

Q – Testdaten – CR: **Zurücksenden der Testdaten (Ping)**
Kommunikation mit dem Slave. Rücksendung: Alles, was gesendet wurde einschließlich des Kommandozeichens (Q). Abschluß mit ACK.

W – L – x – CR
Die Anforderungs-LED des Slaves schalten. x ist ein Steuerzeichen. Die zulässigen Steuerzeichen und deren Wirkungen:

- o = aus
- r = rot
- g = grün
- R = rot blinken
- G = grün blinken
- F = rot/grün blinken (schnell)

Das Kommando wird auch im Rundempfangszustand (BROADCAST) ausgeführt. Dann gibt es keine Antwort.

W – X – Adresse – CR – Schreibdaten (HEX) – CR
Schreiben hexadezimal auf Adresse. Die Adresse ist 16 Bits lang. Sie wird mit vier HEX-Zeichen übertragen. Nach dem ersten CR antwortet die Einrichtung mit ACK. Dann folgen die hexadezimalen Schreibdaten. Zwei aufeinander folgende HEX-Zeichen ergeben ein binäres Byte. Das erste Zeichen enthält die höherwertige Tetrade, das zweite Zeichen die niederwertige. Es ist möglich, in den gesamten RAM zu schreiben. Wird versucht, auf ungültige Adressen zu schreiben, so werden die Daten angenommen. Es wird aber nichts geschrieben. Der Schreibzugriff wird aber mit NAK beendet, und die Fehlerbedingung LIMIT_CHECK wird gesetzt.

W – x – Adresse – CR – Schreibdaten – CR
Schreiben Zeichen auf Adresse. Die Adresse ist 16 Bits lang. Sie wird mit vier HEX-Zeichen übertragen. Nach dem ersten CR antwortet die Einrichtung mit ACK. Dann folgen die Schreibdaten. Die Bytes werden direkt (ohne weitere Prüfung) in den Speicher geschrieben. Es ist möglich, in den gesamten RAM zu schreiben. Wird versucht, auf ungültige Adressen zu schreiben, so werden die Daten angenommen. Es wird aber nichts geschrieben. Der Schreibzugriff wird aber mit NAK beendet, und die Fehlerbedingung LIMIT_CHECK wird gesetzt.

W – Y – Adresse – ein Datenbyte (HEX) – CR
Ein einzelnes Byte headezimal schreiben. Dem Unterkommandocode folgen vier Hexadezimalzeichen für die Adresse und zwei Hexadezimalzeichen für das Datenbyte.

W – y – Adresse – ein Datenbyte – CR
Ein einzelnes Byte schreiben. Dem Unterkommandocode folgen vier Hexadezimalzeichen für die Adresse und das Datenbyte.

W – Q – Adresse – vier Datenbytes (HEX) – CR
Vier Bytes hexadezimal schreiben. Dem Unterkommandocode folgen vier Hexadezimalzeichen für die Adresse und acht Hexadezimalzeichen für die vier Datenbytes. Die Datenbytes werden auf aufeinander folgende Adressen geschrieben (Adreßerhöhung jeweils um 1).

W – q – Adresse – vier Datenbytes – CR
Vier Bytes schreiben. Dem Unterkommandocode folgen vier Hexadezimalzeichen für die Adresse und vier Datenbytes. Die Datenbytes werden auf aufeinander folgende Adressen geschrieben (Adreßerhöhung jeweils um 1).

W – O – Objektnummer – CR – Schreibdaten (HEX) – CR
Schreiben von hexadezimalen Objektdaten. Die Objektnummer ist ein binäres Byte (Wertebereich 0...255). Sie wird mit zwei Hexadezimalzeichen übertragen. Objektadresse und Länge ergeben sich aus der Objekttabelle (Object Reference Table ORT) der Einrichtung. Der Datenbereich des Objekts wird stets von Anfang an überschrieben. Jedes zu schreibende Byte wird mit zwei Hexadezimalzeichen übertragen.

W – o – Objektnummer – CR – Schreibdaten – CR
Schreiben von Objektdaten. Die Objektnummer ist ein binäres Byte (Wertebereich 0...255). Sie wird mit zwei Hexadezimalzeichen übertragen. Objektadresse und Länge ergeben sich aus der Objekttabelle (Object Reference Table ORT) der Einrichtung. Der Datenbereich des Objekts wird stets von Anfang an überschrieben. Die Bytes werden direkt (ohne weitere Prüfung) in den Speicher geschrieben.

W – r – Objektnummer – Objektdeskriptor – CR
Schreiben Objektdeskriptor. Die Objektnummer ist ein binäres Byte (Wertebereich 0...255). Sie wird mit zwei Hexadezimalzeichen übertragen. Der Objektdeskriptor steht in der Objekttabelle (Object Reference Table ORT) der Einrichtung. Er besteht aus fünf Bytes, die mit zehn Hexadezimalzeichen übertragen werden.

R – X – Adresse – Byteanzahl – CR
Lesen hexadezimal von Adresse. Es werden so viele Bytes gelesen, wie als Byteanzahl angegeben. Adresse und Byteanzahl sind jeweils 16 Bits lang. Sie werden mit jeweils vier HEX-Zeichen übertragen. Jedes gelesene Datenbyte wird mit zwei Hexadezimalzeichen zurückgesendet.

R – x – Adresse – Byteanzahl – CR
Lesen von Adresse. Es werden so viele Bytes gelesen, wie als Byteanzahl angegeben. Adresse und Byteanzahl sind jeweils 16 Bits lang. Sie werden mit jeweils vier HEX-Zeichen übertragen. Die gelesenen Datenbytes werden direkt zurückgesendet.

R – Y – Adresse – CR
Ein einzelnes Byte hexadezimal lesen. Dem Unterkommandocode folgen vier Hexadezimalzeichen für die Adresse. Das gelesene Datenbyte wird mit zwei Hexadezimalzeichen zurückgesendet.

R – y – Adresse – CR
Ein einzelnes Byte lesen. Dem Unterkommandocode folgen vier Hexadezimalzeichen für die Adresse. Das gelesene Datenbyte wird direkt zurückgesendet.

R – Q – Adresse – CR
Vier aufeinander folgende Bytes hexadezimal lesen. Dem Unterkommandocode folgen vier Hexadezimalzeichen für die Adresse. Die gelesenen Datenbytes werden mit acht Hexadezimalzeichen zurückgesendet.

R – q – Adresse – CR
Vier aufeinander folgende Bytes lesen. Dem Unterkommandocode folgen vier Hexadezimalzeichen für die Adresse. Die gelesenen Datenbytes werden direkt zurückgesendet.

R – O – Objektnummer – CR
Einen kompletten Objektinhalt hexadezimal lesen. Die Objektnummer ist ein binäres Byte (Wertebereich 0...255). Sie wird mit zwei Hexadezimalzeichen übertragen. Objektadresse und Länge ergeben sich aus der Objekttabelle (Object Reference Table ORT) der Einrichtung. Jedes gelesene Datenbyte wird mit zwei Hexadezimalzeichen zurückgesendet.

R – o – Objektnummer – CR
Einen kompletten Objektinhalt lesen. Die Objektnummer ist ein binäres Byte (Wertebereich 0...255). Sie wird mit zwei Hexadezimalzeichen übertragen. Objektadresse und Länge ergeben sich aus der Objekttabelle (Object Reference Table ORT) der Einrichtung. Die gelesenen Datenbytes werden direkt zurückgesendet. Achtung: Nur Textobjekte können so gelesen werden.

R – r – Objektnummer – CR
Lesen Objektdeskriptor. Die Objektnummer ist ein binäres Byte (Wertebereich 0...255). Sie wird mit zwei Hexadezimalzeichen übertragen. Der Objektdeskriptor steht in der Objekttabelle (Object Reference Table ORT) der Einrichtung. Er besteht aus fünf Bytes, die mit zehn Hexadezimalzeichen übertragen werden.

X – Auswahlzeichen – Schreibdaten – CR: Parameter schreiben
Es können ein ausgewählter Parameter oder alle Parameter geschrieben werden.

- Auswahlzeichen = 0...F (hexadezimal): Parameter 0...15 (ein einzelner Parameter).
- Auswahlzeichen = t: Alle Parameter. Wieviele Parameter es sind, ist einrichtungsspezifisch.

Die Schreibdaten sind Hexadezimalzeichen. Jedes Byte wird mit zwei Hexadezimalzeichen übertragen. Die Anzahl der Bytes je Parameter entspricht der vorgegebenen Parameterlänge. Das niedrigstwertige Byte kommt jeweils als erstes.

x – Auswahlzeichen – Schreibdaten – CR: Variable schreiben
Es können eine ausgewählte Variable oder mehrere Variablen geschrieben werden.

- Auswahlzeichen = 0...F (hexadezimal): Variablen 0...15 (eine einzelne Variable).
- Auswahlzeichen = t: Alle Variablen.
- Auswahlzeichen = i: Alle Eingangsvariablen.
- Auswahlzeichen = o: Alle Ausgangsvariablen.

Wieviele Variablen es jeweils sind, ist einrichtungsspezifisch.

Die Schreibdaten sind Hexadezimalzeichen. Jedes Byte wird mit zwei Hexadezimalzeichen übertragen. Die Anzahl der Bytes je Variable entspricht der vorgegebenen Variablenlänge. Das niedrigstwertige Byte kommt jeweils als erstes.

Y – Auswahlzeichen – CR: Parameter lesen
Es können ein ausgewählter Parameter oder alle Parameter gelesen werden.

- Auswahlzeichen = 0...F (hexadezimal): Parameter 0...15 (ein einzelner Parameter).
- Auswahlzeichen = t: Alle Parameter. Wieviele Parameter es sind, ist einrichtungsspezifisch.

Die Lesedaten sind Hexadezimalzeichen. Jedes Byte wird mit zwei Hexadezimalzeichen übertragen. Die Anzahl der Bytes je Parameter entspricht der vorgegebenen Parameterlänge. Das niedrigstwertige Byte kommt jeweils als erstes.

y – Auswahlzeichen – CR: Variable lesen
Es können eine ausgewählte Variable oder mehrere Variablen gelesen werden.

- Auswahlzeichen = 0...F (hexadezimal): Variablen 0...15 (eine einzelne Variable).
- Auswahlzeichen = t: Alle Variablen.
- Auswahlzeichen = i: Alle Eingangsvariablen.
- Auswahlzeichen = o: Alle Ausgangsvariablen.

Wieviele Variablen es jeweils sind, ist einrichtungsspezifisch.

Die Lesedaten sind Hexadezimalzeichen. Jedes Byte wird mit zwei Hexadezimalzeichen übertragen. Die Anzahl der Bytes je Variable entspricht der vorgegebenen Variablenlänge. Das niedrigstwertige Byte kommt jeweils als erstes.

Z – Auswahlzeichen – Schreibdaten – CR: Register schreiben (selektiver Log-In)
Das Auswahlzeichen wählt ein Register bzw. eine Informationsstruktur der Einrichtung aus, unabhängig von der internen Adresse. Alle Ziffern und Buchstaben des ASCII-Codes sind als Auswahlzeichen zugelassen. Aus dieser Auswahl ergibt sich auch, wieviele Bytes Schreibdaten nachfolgen müssen. Jedes Byte wird mit zwei Hexadezimalzeichen übertragen. Hinweis: Ein Schreiben in Bereiche, die länger sein können als 255 Bytes, wird in dieser Version nicht unterstützt[3].

z – Auswahlzeichen – CR: Register lesen (selektiver Log-Out)
Das Auswahlzeichen wählt ein Register bzw. eine Informationsstruktur der Einrichtung aus, unabhängig von der internen Adresse. Alle Ziffern und Buchstaben des ASCII-Codes sind als Auswahlzeichen zugelassen. Aus dieser Auswahl ergibt sich auch, wieviele Bytes Lesedaten zurückgesendet werden. Jedes Byte wird mit zwei Hexadezimalzeichen übertragen.

Auswahlzeichen:

Zeichen	Bezeichnung	Erläuterung	Hinweise
0	EVENTS	Ereignisregister	
1	ATN CODE	Die Ursache der ATN-Anforderung	
2	STOP CODE	Die Ursache der Stopanforderung	
3	OPCOM CHECKS	Fehlerbits der seriellen Übertragung	
4	INTERNAL CHECKS	Interne Fehlerbits	
5	ERROR CODE	Fehlercode. Ergänzt manche internen Fehlerbits	
6	COMMON STATS	Allgemeine Zustandsbits	
7	CURRENT COMMAND	Das aktuelle Kommando am Kommandointerface	
8	SERIAL REQUESTS	Anforderungs- und Zustandsbits der seriellen Übertragung	
9	SERIAL COMMAND CODE	Das aktuelle Kommando am Bedieninterface	
A	SERBUFFCOUNT	Der aktuelle Füllstand des seriellen Puffers	
B	BYTEBUFF	Das zwischengepufferte Hexadezimalzeichen beim seriellen Schreiben	
C	LEDSTATE	Die Zustandsbits der LED-Anzeige	
D	POINTER	Adreßzeiger in Schreibkommandos	2 Bytes
E	BYTECOUNT	Bytezähler in Schreibkommandos	2 Bytes
O	OBJECT TABLE	Objekttabelle	Festwerte no_of_objects * descriptor_length. Kein Schreiben
S	STACK AREA	Stackbereich	Festwert stacksize. Kein Schreiben

3: Abhilfe: mit Funktion z die jeweilige Adresse beschaffen und das Schreiben mit W-Kommandos erledigen.

Zeichen	Bezeichnung	Erläuterung	Hinweise
s	SP	Stackpointer	2 Bytes. Kein Schreiben
z	MACHINE IDENT	Maschinenidentifikation	5 Bytes. Kein Schreiben

Die Maschinenidentifikation:

Byte	Bereich	HEX	Angabe	Name im Programm
1	00H...FFH	2	Typ der Einrichtung	device_type
3	15	1	Parameteranzahl	no_of_parms
4	15	1	Variablenanzahl	no_of_vars
5	15	1	Anzahl der Eingangsvariablen	no_of_ins
6	255	2	Anzahl der Objekte	no_of_objects
8	255	2	Länge des Maschinenfehlerreports	mc_reportlength
10	0...255*4	2	Größe des seriellen Puffers	serbuffsize
12	255	2	Wortlänge am Kommandointerface (in Bits)	wordlength
14	15	1	Variablenlänge (in Bytes)	varlength
15	15	1	Parameterlänge (in Bytes)	paramlength
16	15	1	Deskriptorlänge (in Bytes)	descriptor_length
17	0...64k-1	4	die Anfangsadresse der Objekttabelle	object_table
21	0...255*4	2	Länge des Stackbereichs	stacksize
23	255	2	die Anfangsadresse des RAM	sram_start
25	0...64k-1	4	die zugängliche RAM-Speicherkapazität außer dem Stackbereich	ram_arena

6. Register im Slave

Übersicht:

Bezeichnung	Offset	Bytes	Inhalt
EVENTS	0	1	Ereignisse
ATN CODE	1	1	Die aktuelle externe Anforderung an das Bediengerät (binär codiert)
STOP CODE	2	1	Die Ursache der STOP-Signalisierung (binär codiert)
OPCOM CHECKS	3	1	Fehlerbits der seriellen Schnittstelle und der Kommandoausführung
INTERNAL CHECKS	4	1	Interne Fehlerbits
ERROR CODE	5	1	Näheres zur Fehlerursache (binär codiert)
COMMON STATS	6	1	Interne Zustandsbits
CURRENT COMMAND	7	1	Das aktuelle Kommando am Kommandointerface
SERIAL REQUESTS	8	1	ATN-Anforderung und Slaveauswahl
SERIAL COMMAND CODE	9	1	Das aktuelle serielle Kommando vom Bediengerät (zwecks Fehleranalyse). CAN und ENQ werden nicht gespeichert
SERBUFFCOUNT	10	1	Füllstand des seriellen Puffers
LEDSTATE	11	1	Die Zustände der programmseitg angesteuerten LEDs
LEDCOUNT ATN	12	2	Blinkzeitzähler der Anforderungs-LED
BYTEBUFF	14	1	Puffer für das erste Hexadezimalzeichen beim direkten Schreiben
POINTER	15	2	Adreßzeiger beim Schreiben und Lesen
BYTECOUNT	17	2	Bytezähler beim Schreiben und Lesen
MAXLENGTH	19	2	Bereichslänge beim Schreiben und Lesen

EVENTS

7	6	5	4	3	2	1	0
SERVICE REQUEST				MACHINE CHECK	STOP COND	ATN COND	ATN SWITCH

- ATN_SWITCH: Schalter betätigt (Schalteranforderung).
- ATN COND. Stopbedingung anhängig. In der Kommandoausführung zu setzen. Das Kommando muß zudem den zugehörigen Anforderungscode in ATN_CODE-Register laden. Das Bit bewirkt, daß am Ende des Kommandozyklus die ATN#-Leitung aktiviert wird (Low).
- STOP COND. Stopbedingung anhängig. In der Kommandoausführung zu setzen. Das Bit bewirkt, daß am Ende des Kommandozyklus die STOP#-Leitung aktiviert wird (Low).
- MACHINE CHECK. Fehlerbedingung. Näheres in den Fehlerregistern OPCOM CHECKS, INTERNAL CHECKS und ERROR CODE.
- SERVICE REQUEST: Bedienanforderung für Serviceschleife anhängig.

STOP CODE

7	6	5	4	3	2	1	0

ATN CODE

7	6	5	4	3	2	1	0

OPCOM CHECKS

7	6	5	4	3	2	1	0
	CR CHECK	DATA CHECK	CMD CHECK	BUFFER EMPTY	BUFFER FULL	FRAMING ERROR	OVERRUN

- OVERRUN: neues Byte empfangen, aber das vorherige noch nicht abgeholt (Fehlermeldung vom USART).
- FRAMING ERROR: falsches Stopbit (Fehlermeldung vom USART).
- BUFFER FULL: CR-Kommando und serieller Puffer voll (Überlauf).
- BUFFER EMPTY: CR-Kommando und serieller Puffer leer.
- CMD CHECK: inkorrekter (primärer) Kommandocode (Kommandos mit CR).
- DATA_CHECK: inkorrekter Füllstand des seriellen Puffers (zu viele oder zu wenige Zeichen).
- CR CHECK: Fehler beim Ausführen eines CR-Kommandos (falsche Codes, falsche Werte usw.).

INTERNAL CHECKS

7	6	5	4	3	2	1	0
LIMIT CHECK	ADRS CHECK					COMMAND CHECK	SPURIOUS ITRP

- SPURIOUS ITRP: Unerwartete Unterbrechung.
- COMMAND CHECK: Inkorrektes Kommando am Kommandointerface.
- ADRS CHECK: Adressierungsfehler beim Schreiben oder Lesen.
- LIMIT CHECK: Bereichsüberschreitung beim Schreiben.

ERROR CODE

7	6	5	4	3	2	1	0
ERROR CODE							

COMMON STATS

7	6	5	4	3	2	1	0
SERVICE MODE					CHARAC-TER MODE	DATA WRITE	BYTEBUFF FULL

- BYTEBUFF FULL: Der Bytepuffer zur Wandlung der einlaufenden Daten ist mit einem Hexadezimalzeichen gefüllt.
- DATA WRITE: die einlaufenden Daten werden ins Binäre gewandelt und nicht in den seriellen Puffer, sondern direkt in den Speicher geschrieben.

- CHARACTER MODE: direkte Datenzugriffe (nicht hexadazimal).
- SERVICE MODE: Die Einrichtung läuft in der Serviceschleife.

SERIAL REQUESTS

7	6	5	4	3	2	1	0
						BROAD-CAST	SLAVE SELECTED

- SLAVE SELECTED: Die Einrichtung ist als Slave ausgewählt
- BROADCAST: Rundempfang, Allgemeinauswahl.

Anhang

Die Anforderungs-LED der Einrichtungen:
Interruptbetrieb über Counter/Timer.

- Aus: Nach dem Rücksetzen.
- Rot: Initialisierung. Ggf. auch rechnend (ausprobieren).
- Grün: Betriebsbereit.
- Rot blinkend: Anforderung anhängig.
- Grün blinkend: Anforderung wird bearbeitet.
- Rot-Grün blinkend (schnell): Fehlerzustand.

Primäre Anforderungscodes der Einrichtungen (Slaves):

Anforderungscode	Anforderung
0	Keine
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	

Weitere Codes betreffen interne Rechen- oder Fehlerzustände (TBD).

Die ASCII Steuerzeichen:

Dez.	HEX	Zeichen	Bedeutung	Dez.	HEX	Zeichen	Bedeutung
0	0	NUL	Null	16	10	DLE	Data Link Escape
1	1	SOH	Start of Heading	17	11	DC1	Device Control 1
2	2	STX	Start of Text	18	12	DC2	Device Control 2
3	3	ETX	End of Text	19	13	DC3	Device Control 3
4	4	EOT	End of Transmission	20	14	DC4	Device Control 4
5	5	ENQ	Enquiry	21	15	NAK	Negative Acknowledge
6	6	ACK	Acknowledge	22	16	SYN	Synchronous Idle
7	7	BEL	Bell (Klingel)	23	17	ETB	End of Transmission Block
8	8	BS	Backspace (Rückschritt)	24	18	CAN	Cancel (ungültig machen)
9	9	TAB	Horizontal Tabulation	25	19	EM	End of Medium
10	A	LF	Line Feed (Zeilenvorschub)	26	1A	SUB	Substitute Character
11	B	VT	Vertical Tabulation	27	1B	ESC	Escape (Umschaltung)
12	C	FF	Form Feed (New Page; neue Seite)	28	1C	FS	File Spearator
13	D	CR	Carriage Return (Zeilenrücklauf)	29	1D	GS	Group Separator
14	E	SO	Shift Out (Dauerumschaltung)	30	1E	RS	Record Separator
15	F	SI	Shift In (Rückschaltung)	31	1F	US	Unit Separator
				127	7F	DEL	Delete

Die einfachsten Diagnoseabläufe:

1. Ausgaben: Festwerte der Art Einrichtungstyp + laufende Nummer (von 1 an)
Werte werden in den Ausgangsvariablen bereitgestellt

2. Eingaben: werden abgeholt und in den Eingangsvariablen gespeichert.

Wenn Parameter 0 = alles 0: s. oben
Wenn Parameter 0 = 0100H: Anzeige der Eingaben auf EBT 02/10.