

Application Note AN098

CQmax für X-MAX-1, X-MAX-E und X-COM-4

1. Hinweise	1
2. Funktion	2
3. Basiskommunikation.....	3
3.1. Die Grundstruktur der Basiskommunikation	3
3.2. Handshake	3
3.3. Senden und Empfangen.....	4
4. Das Kommunikationsprogramm X1PA008.LIB.....	4
4.1. Initialisierung.....	5
4.2. Empfangen.....	5
4.3. Senden	6
4.4. Die Parameter des Programms X1PA008.LIB	8
4.5. Tabelle der möglichen Fehler für einen Programmabbruch:.....	9
4.6. Die Funktionen und Prozeduren des Basiskommunikationsprogramms.....	10
4.7. Fehlerrückgabecodes von Funktionen des Programms X1PA008	12
5. Historie dieses Dokumentes.....	13

1. Hinweise

Das Treiberprogramm CQmax läuft auf einem X-MAX-1 bzw. X-MAX-E unter dem Betriebssystem OsX. Es ermöglicht die Benutzung der seriellen Schnittstellen der Module X-MAX-1, X-MAX-E und X-COM-4.

Für das X-COM-4 kann alternativ dazu auch der **Modul-Device-Treiber** verwendet werden (s. Handbuch). Dessen Einsatz wird **für das X-COM-4 unbedingt empfohlen**, da beim X-COM-4 mit CQmax Interruptkonflikte mit anderen Modulen auftreten können.

Alles in dieser Application Note bezieht sich sowohl auf das X-COM-4 als auch auf die serielle Schnittstelle des X-MAX-1 bzw. X-MAX-E, auch wenn nicht jedesmal alle Modultypen genannt sind.

2. Funktion

Die Intelligenz des MAX-PC wird zum einen für die Pufferung der Daten und zum anderen für die Abarbeitung beliebiger Protokolle benutzt. Beides läuft komplett unabhängig vom Host-PC. Der Host-PC gibt die Nutzdaten zu beliebiger Zeit mit hoher Geschwindigkeit zum MAX-PC, der sich von da ab nur um das Senden der Daten und um die Erfüllung aller Protokollanforderungen kümmert.

Die Echtzeit-Programme für die serielle Kommunikation lassen sich in zwei Teile zerlegen. Der eine Teil übernimmt das Senden und das Empfangen von Zeichen aus dem bzw. in den Speicher des MAX-PC. Er setzt direkt auf der Hardware auf und stellt alle zeitkritischen Funktionen zur Verfügung. Er ist so programmiert, dass auch bei hohen Baudraten keine Zeichen verloren gehen. Dieser Teil wird als **Basiskommunikation** bezeichnet. Der zweite Teil ist eher nicht zeitkritisch. Er übernimmt das gesamte **Protokollhandling**, also die Erzeugung und Überprüfung von Steuerzeichen und -signalen, Anforderung von Wiederholungen und alle anderen Aktionen, die zur Erfüllung der Protokollspezifikationen nötig sind. Natürlich kann auch in diesem Teil des Programms die Zeit eine wichtige Rolle spielen, wenn zum Beispiel Reaktionstelegramme innerhalb einer bestimmten Zeitspanne versandt werden müssen.

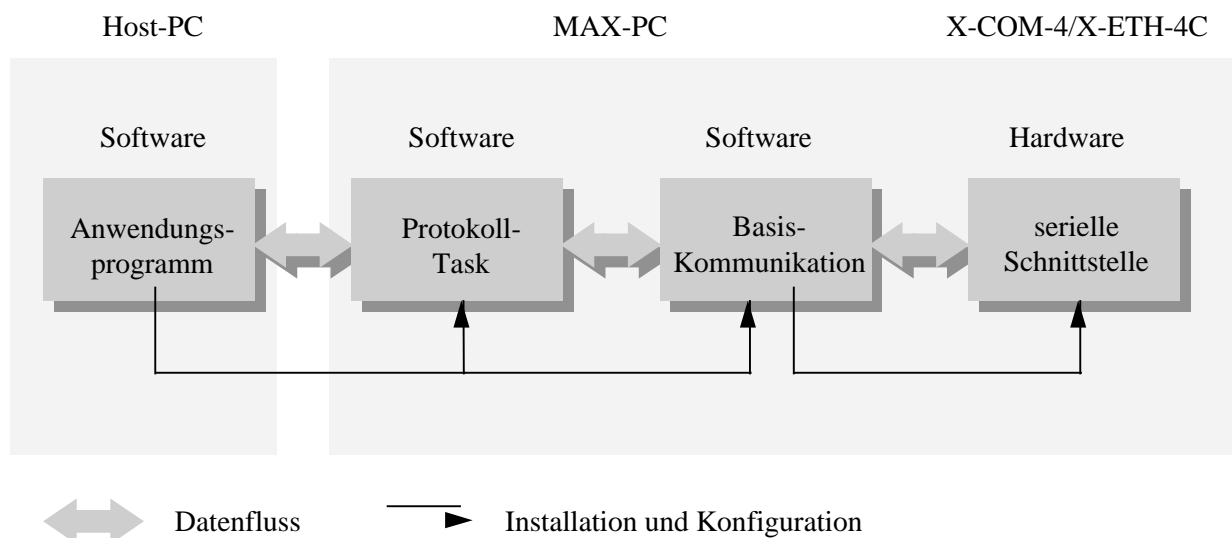


Abb. 1-1: Grundstruktur der asynchronen seriellen Kommunikation

Der PC tauscht mit dem Teil bzw. mit der Task, die das Protokollhandling übernimmt, die Sende- und Empfangsdaten aus. Wenn ganz ohne Protokoll oder nur mit einem einfachen Handshake (RTS/CTS, XON/XOFF) kommuniziert wird, entfällt die Protokolltask, und der PC tauscht die Daten direkt mit den Tasks der Basiskommunikation aus.

3. Basiskommunikation

Die Programme der Basiskommunikation lösen, wie bereits erwähnt, den zeitkritischen Teil der Kommunikation und arbeiten direkt mit der Hardware. Die Basiskommunikation für eine serielle Schnittstelle setzt sich aus mehreren Programmen bzw. Tasks zusammen.

3.1. Die Grundstruktur der Basiskommunikation

Jede der 4 seriellen Schnittstellen auf dem Modul kann aus mehreren Gründen Interrupts auslösen. Für jede Schnittstelle und jeden ihrer Interrupts wird eine eigene Task installiert. Da aber jedes MAX-Modul X-COM-4 nur eine einzige Interruptleitung zum MAX-PC hat, muss zunächst entschieden werden, von welcher Schnittstelle und aus welchem Grund der Interrupt ausgelöst wurde, um zu bestimmen, welche Task den Interrupt bearbeiten muss. Diese Aufgabe wird von einer Task übernommen, die unter dem jeweiligen Interrupt des X-COM-4 installiert und als Interruptmanager bezeichnet wird. Bei der Verwendung der seriellen Schnittstelle des CPU-Moduls muss der Interruptmanager aus Kompatibilitätsgründen ebenfalls installiert werden.

Die Basiskommunikationsprogramme bestehen aus einem Interruptmanager (Programm X1PA007.LIB) und einem Kommunikationsprogramm (Programm X1PA008.LIB), die für die Schnittstellen des MAX-Moduls X-COM-4 geeignet sind. Pro X-COM-4 Modul muss eine Interruptmanager-Task installiert sein und für jeden verwendeten Kanal eine Kommunikations-Task.

Nur mit den Kommunikationstasks werden Daten ausgetauscht, der Interruptmanager wird nie aus der Anwendungsebene heraus angesprochen.

3.2. Handshake

Für asynchrone Kommunikation bietet die Basiskommunikation zwei einfache Handshakemechanismen (RTS/CTS und XON/XOFF), mit denen verhindert werden kann, dass der Empfangspuffer überläuft. In beiden Fällen meldet die Empfangsstation, wenn der Empfangspuffer einen bestimmten (einstellbaren) 'Füllstand' erreicht hat. Der Sender sendet der Gegenstation dann vorübergehend keine Zeichen mehr, bis gemeldet wird, dass der 'Füllstand' wieder unter eine (einstellbare) Grenze gefallen ist.

Beim RTS/CTS-Handshake muss die RTS-Leitung des Empfängers mit der CTS-Leitung des Senders verbunden werden. Wenn ein bestimmter Füllstand (Meldegrenze für 'Puffer voll') erreicht ist, setzt der Empfänger RTS auf Null, der Sender unterbricht dann den laufenden Sendevorgang. Sobald der Puffer wieder leer ist (Meldegrenze für 'Puffer leer'), wird RTS wieder auf eins gesetzt, und der Sender fährt mit dem Senden von Zeichen fort.

Der XON/XOFF-Handshake benötigt im Gegensatz zu RTS/CTS keine zusätzlichen Steuerleitungen. Der Empfänger sendet, sobald die obere Meldegrenze erreicht ist, das Steuerzeichen XOFF zum Sender. XOFF ist ein beliebiges Zeichen, das auf Empfänger- und Senderseite gleich sein muss (üblicherweise 13h). Sobald auf der Sendeseite XOFF empfangen wird, wird das Senden weiterer Zeichen unterbunden, bis der Empfänger durch das Senden von XON wieder Empfangsbereitschaft meldet. XON ist ebenfalls ein frei definierbares Zeichen (üblicherweise 11h). Beim Arbeiten mit XON/XOFF ist zu beachten, dass sowohl XON als auch XOFF reservierte Steuerzeichen sind und in den Nutzdaten nicht vorkommen dürfen. Sie werden immer als Steuerzeichen interpretiert und gelangen nie in den Empfangspuffer.

3.3. Senden und Empfangen

Die für Senden und Empfangen benutzten Parameter, Prozeduren und Funktionen sind für alle von SORCUS gelieferten Basiskommunikationsprogramme gleich. Je nachdem, ob die Sendedaten vom PC oder von einer übergeordneten Protokolltask kommen, müssen folgende Schritte mit den Bibliotheken durchgeführt werden. Alle in diesem Kapitel beschriebenen Aufrufe beziehen sich auf die Kommunikationstask des jeweiligen Kanals. Die Tasknummern können im Prinzip frei gewählt werden; empfohlen wird jedoch folgende Zuordnung:

Interruptkanal	Tasknummer des Interrupt-Managers	Tasknummer der Kommunikationstask
PIRQ-6 (7ch)	301h	318h + Kanalnummer (0..3)
PIRQ-7 (7dh)	302h	320h + Kanalnummer (0..3)
PIRQ-2 (91h)	303h	328h + Kanalnummer (0..3)
PIRQ-3 (92h)	304h	330h + Kanalnummer (0..3)

Tab. 3-1: *Empfohlene Verteilung von Tasknummern und Interruptnummern*

4. Das Kommunikationsprogramm X1PA008.LIB

Das Kommunikationsprogramm X1PA008.LIB bedient jeweils eine serielle Schnittstelle. Dies betrifft sowohl die Initialisierung der Schnittstelle als auch den laufenden Betrieb. Es ermöglicht den Zugriff auf alle für eine serielle Schnittstelle relevanten Einstellungen und Zustände (Fehlerstatus, Statusleitungen, etc.).

Das Basiskommunikationsprogramm enthält auch die für die beiden Datenflussprotokolle XON/XOFF und RTS/CTS nötigen Funktionen, die sich durch Parameter aktivieren lassen.

4.1. Initialisierung

Zunächst müssen die Kommunikationsparameter im Parameterbereich der für den Kanal zuständigen Kommunikationstask eingestellt werden. Anschließend wird die Task durch Aufruf von Prozedur 2 gestartet, so dass sie zum Senden und Empfangen bereit ist. Die Datenpuffer sind dadurch angelegt. Falls Sie aber Parameter (wie z.B. Baudrate) ändern oder die Kommunikation neu beginnen möchten, muss das Programm reinitialisiert werden. Nach dem Ändern von Parametern muss Prozedur 2 aufgerufen werden, damit die Änderungen wirksam werden. Mit den Prozeduren 5 und 15 können Sende- und Empfangsbereitschaft ein- und ausgeschaltet sowie Sende- und Empfangspuffer gelöscht werden.

4.2. Empfangen

Die empfangenen Daten stehen im Empfangspuffer der Basiskommunikationstask, sie können mit der Funktion 17 (11h) der Kommunikationstask abgeholt werden. Dazu müssen Sie zuerst eine Datenstruktur reservieren, die die empfangenen Daten aufnehmen soll, und die Adresse dieser Struktur übergeben. Werten Sie unbedingt den von der Funktion zurückgelieferten Fehlercode aus. Er gibt zum Beispiel an, ob die übergebenen Daten gültig sind.

Das folgende Beispiel kopiert empfangene Zeichen in einen String. Wenn das Funktionsergebnis 0 ist, sind die Daten gültig. Der beim Aufruf der Funktion übergebene String muss für mindestens 255 Zeichen Speicher reserviert haben.

Pascal:

```
FUNCTION rcv_string (channel: BYTE; VAR data: STRING): MAX_ERROR;
VAR
  dummyout: BYTE;
  insize: WORD;
  outsize: WORD;
  error: MAX_ERROR;
  task: WORD;
BEGIN
  task := $318 + channel;
  insize := 0;
  outsize := 255;
  { Die Empfangsdaten werden an data[1] geschrieben, }
  { da das erste Byte von Data die Stringlänge enthält. }
  error := max_call_func(hModul,task,17,insize,dummyout,outsize,data[1]);
  data[0] := chr(outsize);           {Stringlänge einstellen}
  rcv_string := error;
END;
```

C:

```
MAX_ERROR rcv_string (UCHAR channel, char *data)
{
    void      *dummyout;
    USHORT    insize, outsize;
    UCHAR     error;
    USHORT    task;

    task = 0x318 + channel;
    insize = 0;
    outsize = 255;
    error = max_call_func(hModul, task, 17, &insize, &dummyout, &outsize, data);
    data[outsize] = 0; /* String mit 'Null' beenden */
    return(error);
}
```

Die Funktion überprüft dann den Status des Empfangspuffers, kopiert die empfangenen Zeichen in den angegebenen Puffer und liefert die Anzahl der tatsächlich gelesenen Zeichen zurück oder gegebenenfalls eine Fehlermeldung (siehe 4.7).

Um den Empfangsstatus zu ermitteln, liest man die zugehörigen Parameter (Nummern 262 und 264), in denen Fehlermeldungen sowie die Anzahl der Zeichen im Puffer enthalten sind.

4.3. Senden

Um Daten zu senden, müssen Sie die Funktion 7 der Basiskommunikation aufrufen und dabei die Sendedaten (max. 256 Bytes) übergeben. Werten Sie unbedingt die von der Funktion zurückgegebene Fehlermeldung aus! Sie enthält zum Beispiel Informationen darüber, ob die Daten in den Puffer eingetragen werden konnten.

Das folgende Beispiel zeigt eine Funktion zum Senden eines Strings. Der Rückgabewert ist 0, wenn der String in den Sendepuffer eingetragen werden konnte.

Pascal:

```
FUNCTION send_string (channel: BYTE; data: STRING): MAX_ERROR;
VAR
  dummyin: BYTE;
  error: MAX_ERROR;
  task, datalen, retlen: WORD;
BEGIN
  task := $318 + channel;

  { Die Sendedaten werden mit data[1] übergeben, da bei einer Übergabe mit
  { data auch das erste Byte (= Stringlänge) übertragen würde. }
  datalen := length(data);
  retlen := 0;
  error := max_call_func(hModul,task,7,datalen,data[1],retlen,dummyin);
  send_string := error;
END;
```

C:

```
MAX_ERROR send_string (UCHAR channel, char *data)
{
  void *dummyin;
  USHORT datalen, retlen;
  MAX_ERROR error;
  USHORT task;

  task = 0x318 + channel;
  datalen = strlen(data);
  retlen = 0;
  error = max_call_func(hModul,task,7,&datalen,&data,&retlen,dummyin);
  return(error);
}
```

Das Basiskommunikationsprogramm sendet dann die Zeichen. Eine Statusabfrage, ob die Zeichen gesendet oder in den Sendepuffer übernommen werden können, wird durch die Funktion vorgenommen. Im Fehlerfall wird ein Fehlercode zurückgeliefert (siehe 4.7).

Um den Sendestatus zu ermitteln (z.B. ob alle Zeichen gesendet wurden), liest man die zugehörigen Parameter (256 und 258), die Statusmeldungen (2 Byte) und die Anzahl der im Sendepuffer (4 Byte) enthaltenen Zeichen.

4.4. Die Parameter des Programms X1PA008.LIB

Alle Parameter, die in der folgenden Tabelle nicht beschrieben sind, sind reserviert und dürfen nicht geändert werden.

Nr.	Typ	Init	Zugr. ¹	Bedeutung des Parameters
0	UCHAR	0	R	Genereller Programmstatus: 0 = geladen, 1 = läuft, 2 = gestoppt, 3 = gestoppt nach Fehler
1	UCHAR	0	R	Fehlerinformation: gültig, wenn Parameter 0 = 3, siehe 4.5
4	USHORT	0	R/W	Programmnummer des zugehörigen Interruptmanagers
6	USHORT	0	R/W	Tasknummer des zugehörigen Interruptmanagers
8	UCHAR	0	R/W	Hier muss die Slot^Layer-Nr. angegeben werden, auf dem das MAX-Modul steckt, dessen serielle Schnittstelle verwendet werden soll (Bit 7..4 = Slot, Bit 3..0 = Layer).
9	UCHAR	0	R/W	Kanalnummer: Kanal A = 0, B = 1, C = 2 und D = 3
10	UCHAR	1	R/W	Physikalische Verbindung: 1 = RS-232, 3 = RS-422, 4 = RS-485
24	LONG	9600	R/W	Baudrate für Senden ²
28	UCHAR	8	R/W	Zeichenlänge für Senden in Anzahl Bit: 5, 6, 7, 8
29	UCHAR	1	R/W	Anzahl Stopbit (Senden und Empfangen) 1 = ein Stopbit, 2 = zwei Stopbits, 3 = 1,5 Stopbits
30	UCHAR	0	R/W	Paritätsbit-Generierung (Senden und Empfangen): 0 = keine, 1 = ungerade, 2 = gerade
34	LONG	9600	R/W	Baudrate für Empfangen ²
38	UCHAR	8	R/W	Zeichenlänge für Empfangen in ³ Anz. Bit: 5, 6, 7, 8
39	UCHAR	0FFh	R	Bitmaske für Zeichenlänge ⁴
40	LONG	0	R/W	Sendepuffer-Größe (in Byte)
48	LONG	0	R/W	Empfangspuffer-Größe (in Byte)
54	UCHAR	0	R/W	Empfangenes Byte bei Paritäts-Fehler verwerfen oder speichern: 0 = speichern, 1 = verwerfen

¹ Zugriff auf Parameter: R= Nur Lesen, R/W = Lesen und Schreiben

² Es sind beliebige Baudraten einstellbar. Nach Aufruf der Prozedur 2 wird hier die nächste realisierbare Baudrate eingetragen.

³ Standardmäßig liefert der Kommunikationsbaustein immer 8 Bit zurück, wobei bei Zeichenlängen kleiner 8 das höchstwertige Bit entsprechend der Parität gesetzt wird. Das Programm liefert normalerweise nur die gewünschte Anzahl von Bits/Zeichen. Alle anderen Bits werden = 0 gesetzt. Soll das Paritätsbit mitgeliefert werden, kann das über eine Bitmaske definiert werden (siehe Parameter 39)

⁴ Bitmaske, die definiert, welche Bits zurückgeliefert werden sollen. Bit = 1 bedeutet: Bit wird nicht gelöscht.

Nr.	Typ	Init	Zugr. ¹	Bedeutung des Parameters
256	USHORT	0	R	Sendestatus (Bitmap) Bit 0 = Sendepuffer (Software) ist leer Bit 1 = alle Zeichen gesendet (Hardware) Bit 2 bis 14 sind reserviert Bit 15 = Sendeteil angehalten (durch PC oder Protokoll)
258	LONG	0	R	Anzahl Zeichen im Sendepuffer
262	USHORT	0	R	Empfangsstatus (Bitmap) Einmaliges Auftreten der Bedingung setzt das zugehörige Bit (Ausnahme Bit 15). Rücksetzung Bit 0 bis 4 durch Funktion 16 Bit 0 = Zeichenverlust wegen übergelaufenem Empfangspuffer Bit 1 = Zeichenverlust wegen Überlastung im UART Bit 2 = Paritäts-Fehler Bit 3 = Frame-Fehler Bit 4 = Break-Detection Bit 5 bis 14 sind reserviert Bit 15 = Empfangsteil angehalten (durch PC oder Protokoll)
264	LONG	0	R	Anzahl Byte im Empfangspuffer
268	USHORT	0	R	Zustand der Eingangs-Steuerleitungen (Bitmap) Bit 0 = CTS (Clear To Send) Bit 1 = DCD (Data Carrier Detected) Bit 2 = RI (Ring Indicator) Bit 3 bis 15 = reserviert
318	USHORT	0	R/W	Protokolltyp 0 = kein Protokoll 1 = XON/XOFF 2 = RTS/CTS

4.5. Tabelle der möglichen Fehler für einen Programmabbruch:

(Wenn Parameter 0 = 3 ist, dann steht in Parameter 1 die Fehlerursache.)

Fehlernr. in Par. 1	Erklärung
0	Reserviert
1	Nicht genügend Platz für Sendepuffer-Reservierung
2	Falscher Parameter für Sendepuffer-Reservierung
3	Unbekannter Fehler während Sendepuffer-Reservierung
4	Falscher Parameter für Sendepuffer-Reservierung
7	Ungültige Sendepuffer-Nr.
8	Sendepuffer wird gerade benutzt (locked)

Fehlernr. in Par. 1	Erklärung
11	Nicht genügend Platz für Empfangspuffer-Reservierung
12	Falscher Parameter für Empfangspuffer-Reservierung
13	Unbekannter Fehler während Empfangspuffer-Reservierung
14	Falscher Parameter für Empfangspuffer-Reservierung
17	Ungültige Puffernummer für Empfangspuffer
18	Empfangspuffer wird gerade benutzt (locked)
30	Fehler beim Aufruf einer externen Funktion (aufgerufen durch Aktions-Filter)
40	Fehler bei Interrupt-Service-Aufruf
50	Keine Quarzfrequenz in EEPROM eingetragen
99	Unbekannter Fehler

4.6. Die Funktionen und Prozeduren des Basiskommunikationsprogramms

Das Basiskommunikationsprogramm umfasst globale Prozeduren (ohne Übergabeparameter und Antwort) und globale Funktionen (mit Übergabe von Parametern und Antwort). In der folgenden Tabelle sind die Prozeduren in der Spalte 'Typ' mit P gekennzeichnet, die Funktionen mit F. Beim Aufruf von Funktionen müssen eine Reihe von Parametern übergeben werden. Sie sind in der folgenden Tabelle mit den Bezeichnern angegeben, mit der die Funktion über die Bibliotheken aufgerufen wird.

Nr.	Typ	Bedeutung der Funktion
2	P	Start/Restart der Kommunikation (Konfiguration, Speicher allokiieren)
4	P	Baudrate aus Parameter 24 und 34 übernehmen und einstellen
5	F	Sendebereitschaft ein- und ausschalten , optional Sendepuffer löschen Hin: insize = 2 outsize = 0 indata = <i>Kontroll-Wort</i> (s.u.) <i>Kontroll-Wort:</i> 0 = Senden anhalten, Puffer nicht löschen 1 = Senden starten, Puffer nicht löschen 2 = Senden anhalten, Puffer löschen 3 = Senden starten, Puffer löschen

Nr.	Typ	Bedeutung der Funktion
6	F	<p>Sendestatus melden</p> <p>Hin: insize = 0 outsize = 6 outdata = <i>Rückgabe-Struktur</i> (s.u.)</p> <p>Rück: outsize = 6</p> <p><i>Rückgabe-Struktur:</i> Sendestatus (USHORT), Anzahl Zeichen im Sendepuffer (LONG), Bedeutung wie Parameter 256 und 258</p>
7	F	<p>Zeichenkette an Sendepuffer übergeben</p> <p>Hin: insize = Anzahl zu übergebender Zeichen outsize = 0 indata = <i>Übergabe-Puffer</i> (s.u.)</p> <p><i>Übergabe-Puffer:</i> Datenstruktur, die <i>insize</i> zu sendende Bytes enthält.</p>
15	F	<p>Empfangsbereitschaft ein- und ausschalten, optional Empfangspuffer löschen</p> <p>Hin: insize = 2 outsize = 0 indata = <i>Kontroll-Wort</i> (s.u.)</p> <p><i>Kontroll-Wort:</i> 0 = Empfangen anhalten, Puffer nicht löschen 1 = Empfangen starten, Puffer nicht löschen 2 = Empfangen anhalten, Puffer löschen 3 = Empfangen starten, Puffer löschen</p>
16	F	<p>Empfangsstatus melden</p> <p>Hin: insize = 2 outsize = 6 outdata = <i>Rückgabe-Struktur</i> (s.u.)</p> <p>Rück: outsize = 6</p> <p><i>Rückgabe-Struktur:</i> Empfangsstatus (USHORT), Anzahl Zeichen im Empfangspuffer (LONG), Bedeutung wie Parameter 262 und 264</p>
17	F	<p>Zeichenkette aus Empfangspuffer übernehmen</p> <p>Hin: insize = 0 outsize (1) = Anzahl angeforderter Zeichen outdata = <i>Rückgabe-Puffer</i> (s.u.)</p> <p>Rück: outsize (2) = Anzahl tatsächlich gelesener Zeichen</p> <p><i>Rückgabe-Puffer:</i> Datenstruktur, die die gelesenen Zeichen aufnehmen kann, also mindestens <i>outsize</i> (1) Byte umfasst. Nach dem Aufruf sind die ersten <i>outsize</i> (2) Byte gültig.</p>

Nr.	Typ	Bedeutung der Funktion
33	F	<p>Steuerleitungs-Ausgänge setzen</p> <p>Hin: insize = 4 outsized = 0 indata = <i>Übergabe-Struktur</i> (s.u.)</p> <p><i>Übergabe-Struktur:</i> Wort 1: Maske für die betroffenen Steuerleitungen (USHORT), Bit = 1: Steuerleitung soll geändert werden Wort 2: Information, wie die Steuerleitung gesetzt werden soll</p> <p>Bitmap (Wort 1 und 2): Bit 0 = RTS Bit 1 = DTR Bit 2 = TMT-Break Bit 3 bis 15 reserviert</p>
34	F	<p>Zustand der Steuerleitungs-Eingänge lesen</p> <p>Hin: insize = 0 outsized = 2 outdata = <i>Rückgabe-Struktur</i> (s.u.)</p> <p><i>Rückgabe-Struktur:</i> Information, wie die Steuerleitung gesetzt wurde (USHORT) Bitmap: Bit 0 = CTS Bit 1 = DCD Bit 2 = RI Bit 3 = DSR Bit 4 bis 15 reserviert</p>

4.7. Fehlerrückgabecodes von Funktionen des Programms X1PA008

Alle zurückgelieferten Fehler sind Meldungen vom Betriebssystem. Die folgende Tabelle zeigt, welche Fehler bei den einzelnen Funktionen auftauchen können und was sie bedeuten:

Funktion	Fehlernummer	Bedeutung
5	22h oder 23h	Sendepuffer gesperrt
	26h	Sendepuffer-Nummer ungültig
6	26h	Sendepuffer-Nummer ungültig
7	22h oder 23h	Sendepuffer gesperrt
	24h	Sendepuffer voll
	26h	Sendepuffer-Nummer ungültig
15	22h oder 23h	Empfangspuffer gesperrt
	26h	Empfangspuffer-Nummer ungültig
16	26h	Empfangspuffer-Nummer ungültig
17	22h oder 23h	Empfangspuffer gesperrt
	26h	Empfangspuffer-Nummer ungültig

Bei den Fehlermeldungen 'Puffer voll' bzw. 'Puffer gesperrt' können Sie den Funktionsaufruf zu einem späteren Zeitpunkt wiederholen. Damit sich der Pufferzustand ändern kann, müssen Sie die Kontrolle nach dem ersten Funktionsaufruf (der den Fehler gemeldet hat) zuerst wieder an das Betriebssystem zurückgeben.

5. Historie dieses Dokumentes

Datum	Autor	Änderungen
29.04.2005	DH	Kleine Schönheitsreperaturen
28.04.2005	JD	Erstellung der AN (Übernahme vom Handbuch 2. Auflage)