

## 13. Serielle Kommunikation

Das MODULAR-4 System bietet eine breite Palette serieller Schnittstellen, das Multi-COM System 6 universelle serielle Schnittstellen. Die Intelligenz der Multi-COM/MODULAR-4 Karte wird in diesem Aufgabenfeld zum einen für die Pufferung der Daten und zum anderen für die Abarbeitung beliebiger Protokolle benutzt. Beides läuft komplett unabhängig vom PC. Der PC gibt die Nutzdaten zu beliebiger Zeit mit hoher Geschwindigkeit zur Karte, die sich von da ab allein 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 der Multi-COM/MODULAR-4 Karte. Er setzt direkt auf die Hardware auf und stellt alle zeitkritischen Funktionen zur Verfügung. Er muß so programmiert sein, daß auch bei hohen Baudraten keine Zeichen verlorengehen. Dieser Teil wird als **Basiskommunikation** bezeichnet. Der zweite Teil ist eher zeitunkritisch. 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.

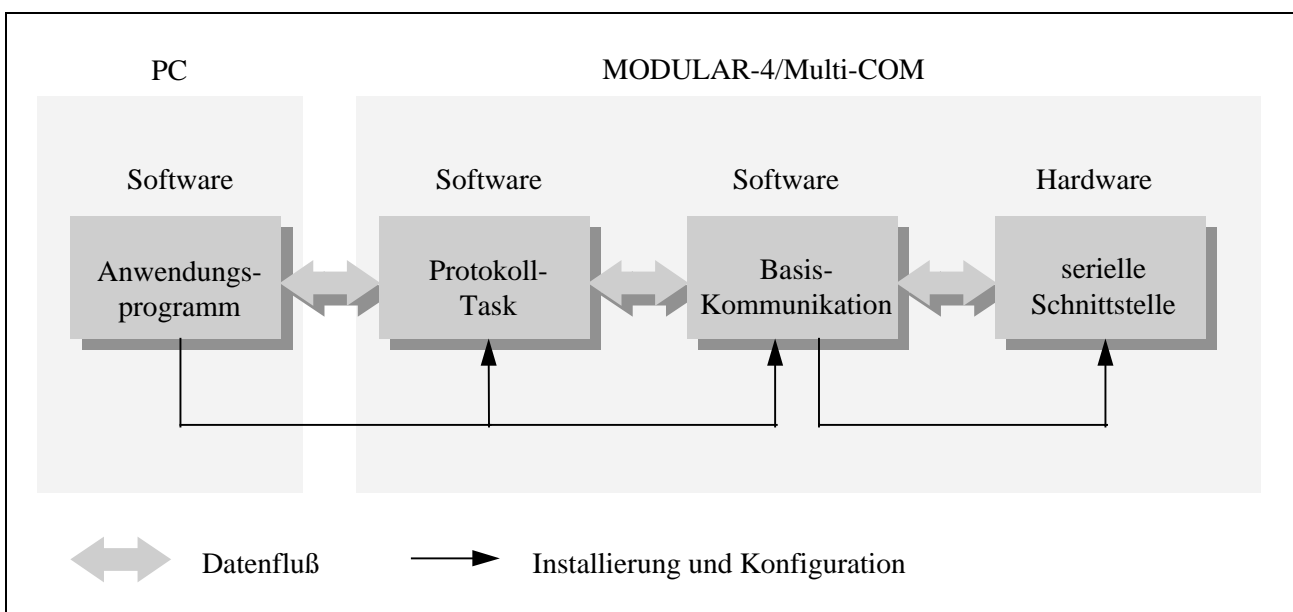


Abb. 13-1: Grundstruktur der asynchronen seriellen Kommunikation

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.

Um die mitgelieferten Kommunikationsprogramme verwenden zu können, muß auf der Multi-COM/MODULAR-4 Karte ein Betriebssystem mit der Version ML6-1A.01E (bzw. ML6-1A.01R) / ML8-3A.00E (bzw. ML8-3A.00R) oder größer laufen. Diese Beschreibung bezieht sich auf die Version 2.I der Kommunikationsprogramme oder spätere Versionen.

## **13.1. Basiskommunikation**

Die Programme der Basiskommunikation lösen, wie bereits erwähnt, den zeitkritischen Teil der Kommunikation und arbeiten direkt mit der Hardware. Die Basiskommunikation für ein serielle Schnittstelle setzt sich aus mehreren Programmen bzw. Tasks zusammen. Die Installierung der Programme geschieht am einfachsten mit SNW6/SNW oder SNW32.

### **13.1.1. Datenpuffer**

Die Kommunikationstasks arbeiten mit zwei unterschiedlichen Datenpuffern.

Der **Sendepuffer** darf nur mit den entsprechenden Funktionen der Basiskommunikationstask beschrieben werden.

Der **Empfangspuffer** enthält die empfangenen Zeichen. Auch er darf nur über den Aufruf von Funktionen der Empfangstask ausgelesen werden.

Für beide Datenpuffer muß beim Installieren angegeben werden, wie groß sie sein sollen.

### **13.1.2. Handshake (asynchrone Kommunikation)**

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.

### 13.1.3. Andere Protokolle

Die Basiskommunikation von SORCUS bietet sog. Aktions-Filter an, die zur Implementierung eigener Protokolle verwendet werden können. Bei bestimmten Zuständen von Steuerleitungen oder bei Empfang spezieller Zeichen können beliebige Funktionen aufgerufen werden. Das XON/XOFF-Handshaking basiert z.B. auf einem Aktions-Filter.

Kapitel 13.6 beschreibt die Verwendung von Aktions-Filtern.

### 13.1.4. 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 Befehlen der Bibliothek ML6BIB oder mit Betriebssystemaufrufen von ML6RTBIB 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, die standardmäßig auch beim Installieren mit SNW6/SNW bzw. SNW32 eingehalten wird:

<b>Schnittstelle</b>	<b>Interruptkanal</b>	<b>Tasknummer des Interrupt-Managers</b>	<b>Tasknummer der Kommunikationstask</b>
A	IRQ-D (7eh)	301h	318h
B			319h
C	IRQ-C (7dh)	302h	320h
D			321h
E	IRQ-B (7ch)	303h	328h
F			329h
A <sup>1</sup> (optional)	SCC (90h)	300h	310h
B (optional)			311h

*Tab. 13-1: Verteilung von Tasknummern und Interruptnummern bei Installation der Basiskommunikation mit SNW6/SNW bzw. SNW32*

<b>Schnittstelle</b>	<b>SCC-Baustein (Modul<sup>1, 2</sup>)</b>	<b>SCC-Kanal</b>
A	1	0
B	1	1
C	2	0
D	2	1
E	3	0
F	3	1

*Tab. 13-2: Zuordnung der Kanäle zu den Kommunikationsbausteinen.*

<sup>1</sup> Die Basiskommunikation für die Schnittstellen A und B kann auch unter Interrupt 90h installiert werden (SCC0, Kompatibilität mit MODULAR-4/486).

<sup>2</sup> SNW6 verwendet die Bezeichnung 'Modul' für die Auswahl eines SCC-Bausteins.

## 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 Null, wenn der String in den Sendepuffer eingetragen werden konnte.

### Pascal:

```
FUNCTION send_string (scc, scc_channel: BYTE; data: STRING): BYTE;
VAR
  dummyin, dummysize : BYTE;
  error : BYTE;
  task : WORD;
BEGIN
  task := $310 + $08 * scc + scc_channel;

  { Die Sendedaten werden mit data[1] übergeben, da bei einer }
  { Übergabe mit data auch das erste Byte der Stringvariablen }
  { (= Stringlänge) übertragen würde. }

  ml6_call_func(task,7,LENGTH(data),data[1],0,dummysize,dummyin,error);

  send_string := error;
END;
```

### C:

```
byte send_string (byte scc, byte scc_channel, char *data)
{
  void *dummyin;
  ushort dummysize;
  byte error;
  ushort task;

  task = 0x310 + 0x08 * scc + scc_channel;

  ml6_call_func(task,7,strlen(data),data,0,&dummysize,dummyin,&error);

  return(error);
}
```

## 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 Null ist, sind die Daten gültig. Der beim Aufruf der Funktion übergebene String muß für mindestens 255 Zeichen Speicher reserviert haben.

### Pascal:

```
FUNCTION rcv_string (scc, scc_channel: BYTE; VAR data: STRING): BYTE;
VAR
  dummyout : BYTE;
  insize    : WORD;
  error     : BYTE;
  task      : WORD;
BEGIN
  task := $310 + $08 * scc + scc_channel;

  { Die Empfangsdaten werden nicht an data, sondern an data[1] }
  { geschrieben, da das erste Byte von Data die Stringlänge   }
  { enthält.                                                  }

  ml6_call_func(task,17,0,dummyout,255,insize,data[1],error);

  data[0] := chr(insize);          {Stringlänge einstellen}
  rcv_string := error;
END;
```

### C:

```
byte rcv_string (byte scc, byte scc_channel, char *data)
{
  void *dummyout;
  ushort insize;
  byte error;
  ushort task;

  task = 0x310 + 0x08 * scc + scc_channel;

  ml6_call_func(task,17,0,dummyout,255,&insize,data,&error);

  data[insize] = 0; /* String mit ,Null' beenden */
  return(error);
}
```

## 13.2. Protokollhandling

Die von SORCUS lieferbaren Protokollprogramme unterstützen alle seriellen Schnittstellen der Multi-COM/MODULAR-4 Karte. Sie setzen auf der Basiskommunikation auf und können teilweise auch mit SNW6/SNW bzw. SNW32 installiert werden. Das Senden und Empfangen von Daten wird bei jedem Protokoll unterschiedlich gehandhabt.

## 13.3. Installieren mit SNW6/SNW

Mit Hilfe von SNW6/SNW kann das Installieren der Basiskommunikationsprogramme und zum Teil auch der Protokollprogramme sehr einfach und komfortabel geschehen. Voraussetzungen dafür sind lediglich, daß sich die Multi-COM/MODULAR-4 Karte im PC befindet und daß alle Kommunikationsprogramme in einem gemeinsamen Verzeichnis stehen.

Alle Hilfsmittel zur Installierung finden Sie unter dem Hauptmenüpunkt **COM**.

Kapitel 13.5. beschreibt die Installation der Basiskommunikation ohne SNW6/SNW bzw. SNW32.

### 13.3.1. Gesamtkonfiguration

Die Gesamtkonfiguration wird in einem Fenster angezeigt, das mit **Konfiguration öffnen** geöffnet wird (siehe Abb. 13-2). Es enthält die Einstellungen aller verfügbaren Schnittstellen einer Karte. Zum Konfigurieren und Installieren einzelner Kanäle muß immer eine Gesamtkonfiguration geöffnet sein. Die meisten Menüpunkte von COM lassen sich auch nur dann aktivieren, wenn ein Fenster mit einer Gesamtkonfiguration angewählt ist (doppelt umrandet).

Das Fenster enthält eine Tabelle, in der alle verfügbaren Schnittstellen aufgelistet sind. Wenn ein Kanal verwendet wird, steht in der Tabelle ein Name. Unbenutzte Kanäle werden mit einem waagerechten Strich gekennzeichnet. Innerhalb der Tabelle ist immer ein Kanal angewählt (invers dargestellt). In der Box links unterhalb des Fensters werden seine Konfigurationsdaten angezeigt. Die Kanalanwahl kann mit den Cursortasten oder mit der Maus vorgenommen werden.

Die Gesamtkonfiguration wird mit dem Menüpunkt **Konfiguration speichern** in einer Datei mit der Erweiterung „.SCF“ (Serial ConFfiguration) gespeichert. **Konfiguration öffnen** dient sowohl dem Laden einer vorhandenen Datei als auch dem Er-

stellen einer neuen. Wenn Sie eine neue Datei erstellen wollen, müssen Sie bei **Konfiguration öffnen** einen neuen Namen angeben.

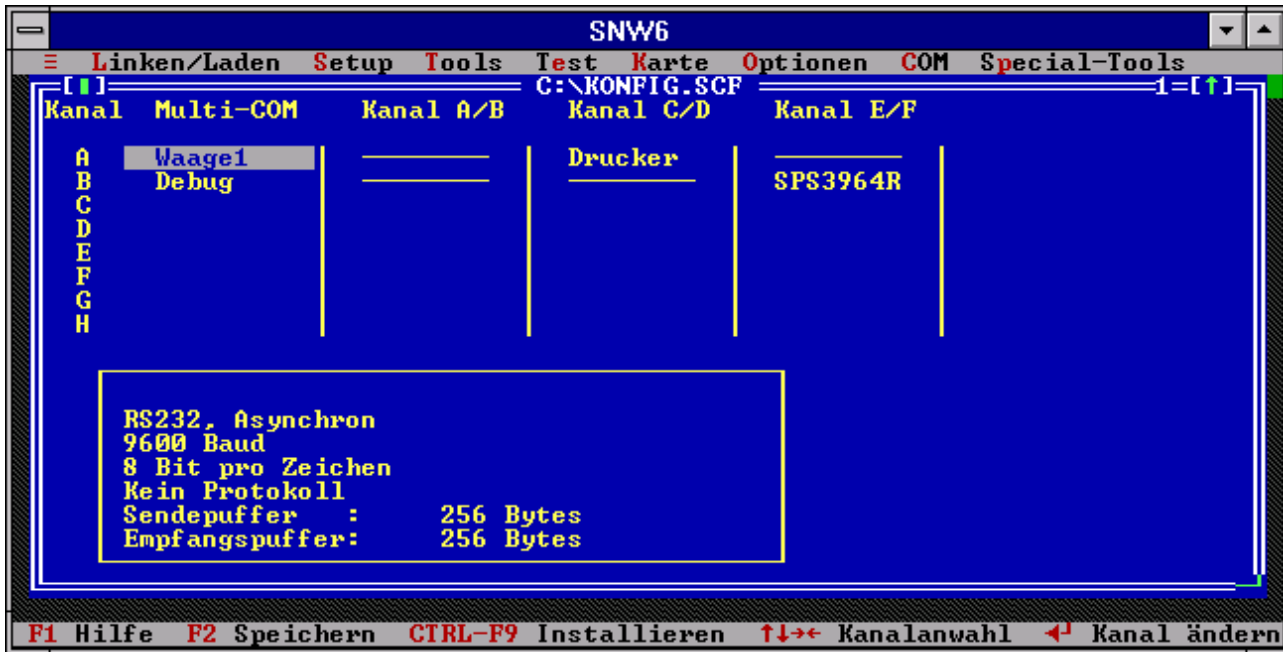


Abb. 13-2: Konfigurationsfenster in SNW6/SNW

### 13.3.2. Kanaleinstellungen

Die für die Datenübertragung relevanten Parameter werden als *Kanaleinstellungen* bezeichnet. Sie werden für jeden Kanal getrennt eingestellt und gespeichert. Um die Einstellungen vorzunehmen, wählt man zunächst den gewünschten Kanal an, drückt die [Return]-Taste oder wählt den Menüpunkt **Kanaleinstellung ändern**. Alternativ genügt ein Doppelklick auf den gewünschten Kanal mit der Maus. Die Einstellungen werden für jeden Kanal in einer eigenen Datei mit der Bezeichnung <Name>.chc (**C**hannel-**C**onfiguration) gespeichert.



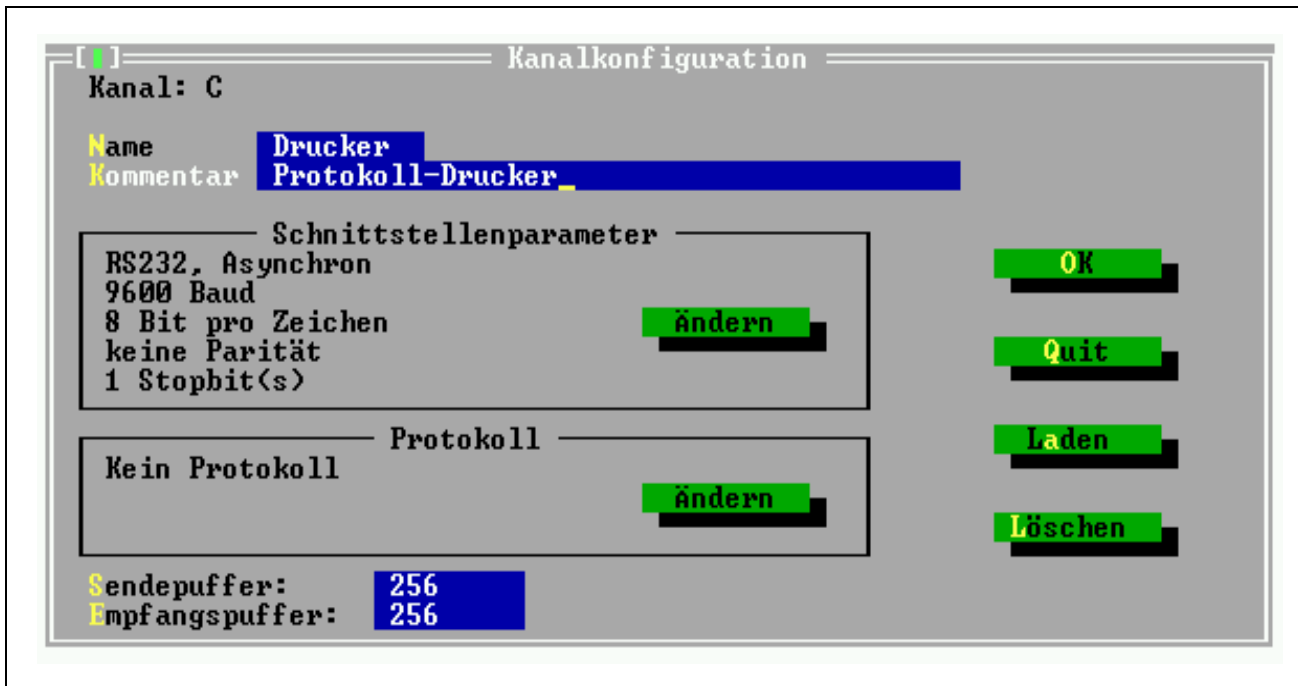


Abb. 13-3: Kanaleinstellungen in SNW6/SNW

- **Name**

Jeder Kanal-Parametersatz erhält einen Namen, der innerhalb einer Gesamtkonfiguration nur einmal vorkommen darf. Der Name dient zum einen der Übersichtlichkeit in der Gesamttabelle, wenn er eindeutig beschreibt, womit die Schnittstelle verbunden ist. Zum anderen dient er zum einfachen Austauschen und Kopieren von Konfigurationsdaten (siehe ‚Laden‘). Der in diesem Feld eingetragene Text muß den Ansprüchen eines Dateinamens (ohne Erweiterung) genügen. Vorsicht ist geboten, wenn mehrere Gesamtkonfigurationen denselben Namen (und damit dieselbe Parameterdatei) verwenden. In diesem Fall wirkt sich die Änderung der Parameter natürlich auch auf alle anderen Konfigurationen aus.

- **Kommentar**

Das Kommentarfeld darf bis zu 40 beliebige Zeichen enthalten. Es dient der Beschreibung, z.B. der möglichst genauen Zuordnung der installierten Schnittstelle bzw. der Gegenstation.

- **Schnittstellenparameter**

Als Schnittstellenparameter gelten die Parameter, die zur Übertragung eines Zeichens eingestellt werden müssen (Schnittstellentyp, Baudrate, Bit pro Zeichen, Paritätsprüfung und Stopbits). In der Box wird die aktuelle Einstellung gezeigt. Wenn sie geändert werden soll, muß der zugehörige ‚Ändern‘-Schalter betätigt werden. Dadurch wird eine Dialogbox zur Einstellung der Parameter geöffnet. Die einzige Besonderheit dieser Dialogbox ist die Eingabe der Baudrate. Neben dem Eingabefeld befindet sich ein Abwärtspfeil, der mit der Maus angeklickt oder mit der [↓]-Taste aktiviert werden kann. Damit erhalten Sie eine Liste aller für diesen Kanal einstellbaren Baudraten.

- **Protokoll**

Unter diesen Punkt fallen sowohl die einfachen Handshakemechanismen XON/XOFF und RTS/CTS als auch komplexe, von SORCUS lieferbare Protokolle wie z.B. 3964/R. Die Box zeigt die aktuellen Einstellungen, die mit dem zugehörigen ‚Ändern‘-Schalter geändert werden können. Je nach eingestelltem Protokoll können die Dialogboxen zur Eingabe der Protokollparameter stark variieren.

- **Sendepuffer**

In dieses Feld wird die Größe des Sendepuffers in Byte eingetragen.

- **Empfangspuffer**

Hier wird die Größe des Empfangspuffers in Byte angegeben.

- **Laden**

Laden dient zum Übernehmen der Parameter (Schnittstellen-, Protokollparameter, ...) anderer Schnittstellen. Nach dem Betätigen des Schalters muß der Name des Parametersatzes angegeben werden, der kopiert werden soll. Wenn es ein Parametersatz ist, der in der aktuellen Gesamtkonfiguration bereits enthalten ist, dann muß er anschließend umbenannt werden. Auf diese Weise können sehr schnell Installationen zur Ansteuerung vieler gleichartiger Geräte durchgeführt werden.

- **Löschen**

Dieser Schalter entfernt die Kanaleinstellung aus der Gesamtkonfiguration, der Kanal wird als unbenutzt markiert. Die Parameterdatei <Name>.chc wird aber nicht gelöscht, die Einstellungen können also später mit ‚Laden‘ wieder zurückgeholt werden.

- **Sonderfunktionen**

Unter diesem Punkt sind Funktionen und Parameter zusammengefaßt, die nicht für alle Schnittstellen bzw. Multi-COM/MODULAR-4 Karten zur Verfügung stehen (z.B. unterschiedliche Baudraten für Senden und Empfangen) und die üblicherweise nicht verwendet oder geändert werden. Wenn keine Sonderfunktionen zur Verfügung stehen, fehlt der Schalter in der Dialogbox.

### 13.3.3. Installieren

Als Installieren wird das Laden und Parametrieren der Kommunikationsprogramme bezeichnet. Die Multi-COM/MODULAR-4 Karte ist anschließend bereit zum Senden und Empfangen von Daten; alle Schnittstellen werden der Gesamtkonfiguration entsprechend eingerichtet. Für das Installieren wird immer eine Installationsdatei (siehe Kapitel 4) benötigt. Sie kann mit dem Menüpunkt ‚Installationsdatei erzeugen‘ erstellt werden. Wenn Sie ‚Installieren‘ wählen, dann wird ebenfalls eine Installationsdatei erzeugt und anschließend installiert. Die Installationsdatei wird in dem Verzeichnis abgelegt, in dem sich die Gesamtkonfigurationsdatei befindet. SNW6/SNW geht beim Laden davon aus, daß sich hier auch die Kommunikationsprogramme der Multi-COM/MODULAR-4 Karte befinden. Die Installationsdatei erhält den gleichen Namen wie die Konfigurationsdatei, allerdings mit der Erweiterung ‚.INS‘.

In der Installationsdatei sind alle Angaben über die Konfiguration der Schnittstellen enthalten, so daß sie auch alleine verwendbar ist. Sie kann also später auch direkt mit einem Kommandozeilenaufruf geladen werden, ohne daß die komplette SNW6/SNW-Umgebung gestartet wird (z.B. in AUTOEXEC.BAT):

**SNW6/SNW /i:<name>**

Unter dem Menüpunkt ‚Optionen‘ können Sie auswählen, ob vor der Installation der Kommunikationsprogramme ein Reset durchgeführt werden soll. Das ist dann sinnvoll, wenn außer den Kommunikationsprogrammen keine anderen Programme auf der Karte laufen sollen, oder die Kommunikationsprogramme immer zuerst installiert werden. **Standardmäßig wird ein Reset in die Installationsdatei eingefügt.**

### **13.3.4. Testen von Schnittstellen**

Sobald die Kommunikationsprogramme für eine Schnittstelle installiert sind, können sie innerhalb von SNW6/SNW getestet werden. Dazu muß der entsprechende Kanal im Konfigurationsfenster angewählt und im ‚COM‘-Untermenü ‚Kanal testen‘ aktiviert werden. Getestet wird immer die Gesamtinstallation eines Kanals. Wenn ein Protokoll installiert wurde, werden die Daten mit der übergeordneten Protokolltask und nicht mit der Basiskommunikation ausgetauscht. Von den Protokolltasks erzeugte Steuerzeichen und Telegrammköpfe gelangen nicht zum PC. Für jeden installierten Kanal kann ein Testfenster geöffnet werden, auch gleichzeitig mit anderen Fenstern.

Zeichen, die gesendet werden sollen, können direkt über die Tastatur eingegeben werden. Mit [Return] werden sie in den Sendepuffer der Karte übertragen. Das Fenster gibt allerdings keine Auskunft darüber, ob die Zeichen tatsächlich gesendet worden sind oder ob das Kommunikationsprogramm vielleicht noch auf ein Handshake-Signal zum Senden der Zeichen (z.B. XON) wartet. Die Sendedaten können entweder direkt als ASCII-Zeichen oder hexadezimal (ohne ‚H‘, ‚0x‘ oder sonstige Kennung) eingegeben werden. Welches Format von beiden verwendet wird, können Sie in ‚COM/Optionen‘ einstellen.

Etwa viermal pro Sekunde (bei jedem Blinken der linken oberen Fensterecke) überprüft SNW6/SNW, ob Zeichen empfangen worden sind. Wenn das der Fall ist, werden bis zu achtzig Byte abgeholt und im unteren Fensterteil angezeigt. Das bedeutet, daß innerhalb des Testprogrammes von einer Schnittstelle maximal 320 Zeichen pro Sekunde zum PC übertragen werden können. Andere PC-Programme können natürlich mit wesentlich höheren Datenraten arbeiten.

### **13.3.5. Abhilfe bei Fehlern**

In der Regel werden alle erkennbaren Fehler gemeldet. Solange eine Fehlermeldung auf dem Bildschirm angezeigt wird, erhalten Sie mit F1 weitere Informationen über die Fehlerursache und mögliche Abhilfen.

Wenn keine Zeichen gesendet oder empfangen werden, sollten Sie zunächst alle Kabel überprüfen. Denken Sie daran, daß die meisten seriellen Verbindungen ein sogenanntes Nullmodemkabel benötigen (mindestens TxD und RxD überkreuz miteinander verbunden). Stellen Sie sicher, daß Sender und Empfänger mit denselben Parametern (Baudrate ...), demselben Handshakemechanismus und demselben Protokoll arbeiten.

## 13.4. Die Grundstruktur der Basiskommunikation

Jeder Kommunikationsbaustein (SCC) hat 2 serielle Schnittstellen, von denen wiederum jede aus mehreren Gründen Interrupts auslösen kann. Für jede Schnittstelle und jeden ihrer Interrupts wird eine eigene Task installiert. Da aber jeder SCC nur eine einzige Interruptleitung zur Multi-COM/MODULAR-4 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 SCC installiert und als **Interruptmanager** bezeichnet wird.

Die Basiskommunikationsprogramme bestehen aus einem Interruptmanager (Programm M6P0500.LIB) und einem Kommunikationsprogramm (Programm M6P0520/M7P0520/M8P0520.LIB), die für die Schnittstellen der Multi-COM/MODULAR-4 Karte geeignet sind. Pro SCC muß eine Interruptmanager-Task installiert sein und für jeden verwendeten Kanal eine Kommunikations-Task.

Die Installierung aller notwendigen Tasks nimmt SNW6/SNW vor. Anschließend werden nur mit den Kommunikationstasks Daten ausgetauscht, der Interruptmanager wird nie aus der Anwendungsebene heraus angesprochen. Im folgenden werden die wichtigsten Parameter und Prozeduren des Kommunikationsprogramms dargestellt.

### 13.4.1. Das Kommunikationsprogramm M6P0520 / M7P0520 / M8P0520.LIB

Das Kommunikationsprogramm M6P0520 / M7P0520 / M8P0520.LIB bedient **eine** serielle Schnittstelle des seriellen Schnittstellenbausteins auf der Multi-COM/MODULAR-4. 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 Datenfluß-Protokolle XON/XOFF und RTS/CTS nötigen Funktionen, die sich durch Parameter aktivieren lassen.

### **13.4.1.1. Initialisierung**

Mit SNW6/SNW wird das Basiskommunikationsprogramm den Einstellungen des Anwenders entsprechend parametrisiert und gestartet, so daß es zum Senden und Empfangen bereit ist (z. Zt nur asynchrone serielle Kommunikation, die Initialisierung für synchrone Kommunikation muß vom Anwenderprogramm entsprechend der Parameterliste durchgeführt werden). 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.

### **13.4.1.2. Empfangen**

Empfangene Zeichen stehen im Empfangspuffer und können mit der Funktion 17 ausgelesen werden (siehe Beispiel Seite 13-6). Dabei wird die gewünschte Anzahl an Zeichen und ein Pointer auf einen Puffer, in den die Zeichen geschrieben werden sollen, an die Funktion übergeben. 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 Tabelle der Fehlerrückgabecodes, Seite 13-25).

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.

### **13.4.1.3. Senden**

Zeichen, die gesendet werden sollen, können mit der Funktion 7 an den Sendepuffer übergeben werden (siehe Beispiel auf Seite 13-5). Das Basiskommunikationsprogramm sendet dann die Zeichen. Der Funktion wird die Anzahl der zu sendenden Zeichen und der Zeiger auf einen Puffer, in dem die Zeichen stehen, übergeben. 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 Tabelle der Fehlerrückgabecodes, Seite 13-25).

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.



<b>Nr.</b>	<b>Typ</b>	<b>Init</b>	<b>Zugr</b>	<b>Bedeutung des Parameters</b>
16	Byte	1	R/W	Kommunikationstyp: 1 = asynchron 2 = Mono-Sync 3 = Bi-Sync 4 = external Sync 5 = SDLC <sup>1</sup> 6 = SDLC Loop Mode <sup>1</sup> 7 = 9 Bit Übertragung <sup>2</sup> 8-255 = reserviert
17	Byte	1	R/W	Daten-Codierung Senden und Empfangen: 1 = NRZ (Non-Return to Zero) 2 = NRZI <sup>1</sup> (Non-Return to Zero Inverted) 3 = FM1 <sup>1</sup> (Biphase Mark) 4 = FM0 <sup>1</sup> (Biphase Space) 5 = Manchester <sup>1</sup> (Biphase Level) 6 bis 255 = reserviert
18*	Byte	1	R/W	Clock-Quelle Senden: 1 = Standard Baudratengenerator 2 = Von extern über Clock-Leitung an RTxC <sup>3</sup> 3 = Intern, Ausgabe auf Clock-Leitung <sup>3</sup> 4 = Autogeneration (DPLL) <sup>1</sup> 5 = Von extern über Clock-Leitung an TRxC <sup>3</sup> 6-255 = reserviert
19*	Byte	1	R/W	Clock-Quelle Empfangen (Bedeutung siehe Parameter 18)
24*	Long (4)	9600	R/W	Baudrate für Senden <sup>4</sup>
28*	Byte	8	R/W	Zeichenlänge für Senden in Anzahl Bit: 5, 6, 7, 8

<sup>1</sup> Noch nicht oder nicht vollständig implementiert

<sup>2</sup> Dazu werden die Daten wortweise übertragen: Das 1. Byte enthält die Bit 0 bis 7, das 2. Byte das Bit 8. Das 2. Byte kann nur 0 oder 1 sein.

<sup>3</sup> Taktein- bzw. Ausgang entspricht dem Baudratentakt

<sup>4</sup> Es sind beliebige Baudraten einstellbar. Nach Aufruf der Prozedur 2 wird hier die tatsächlich realisierbare Baudrate eingetragen



Nr.	Typ	Init	Zugr	Bedeutung des Parameters
29*	Byte	1	R/W	Anzahl Stopbit (Senden und Empfangen) 0 = Sync-Mode Enable 1 = ein Stopbit, 2 = zwei Stopbits, 3 = 1,5 Stopbits
30*	Byte	0	R/W	Paritätsbit-Generierung (Senden und Empfangen): 0 = keine, 1 = ungerade, 2 = gerade
32	Byte	0	R/W	Sync Character 1 (Low-Byte, WR6) s.u.
33	Byte	0	R/W	Sync Character 2 (High-Byte, WR7)s.u.
34*	Long (4)	9600	R/W	Baudrate für Empfangen <sup>4</sup>
38*	Byte	8	R/W	Zeichenlänge für Empfangen <sup>1</sup> in Anz. Bit: 5, 6, 7, 8
39	Byte	0	R	Bitmaske für Zeichenlänge <sup>2</sup>
40*	Long (4)	0	R/W	Sendepuffer-Größe (in Byte)
48*	Long (4)	0	R/W	Empfangspuffer-Größe (in Byte)
52	Word (2)	ffffh	R	Interrupt-Kontrolle (Bitmap, reserviert)
54	Byte	0	R/W	Empfangenes Byte bei Parity-Error verwerfen oder speichern: 0 = speichern, 1 = verwerfen
55	Byte	0	R/W	Sync-Features: <div> <div> Bit 0 = 1: TxCRC Enable  Bit 1 = 1: SDLC/CRC-16  Bit 2 = 1: CRC Preset I/O  Bit 3 = 1: 6-Bit/8-Bit Sync  Bit 4 = 1: RxCRC Enable  Bit 5 = 1: Sync Character Load Inhibit    Bit 6 = 1: Enter Hunt Mode  Bit 7 = 1: Address Search Mode (SDLC) </div> <div> Bit 0 von WR5  Bit 2 von WR5  Bit 7 von WR10  Bit 0 von WR10  Bit 3 von WR3    Bit 1 von WR3  Bit 4 von WR3  Bit 2 von WR3 </div> </div>

<sup>1</sup> 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)

<sup>2</sup> Bitmaske, die definiert, welche Bits zurückgeliefert werden sollen. Bit = 1 bedeutet: Bit wird nicht gelöscht.

Nr.	Typ	Init	Zugr	Bedeutung des Parameters
256	Word (2)	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 (4)	0	R	Anzahl Zeichen im Sendepuffer
262	Word (2)	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 ser. Controller Bit 2 = Parity-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 (4)	0	R	Anzahl Byte im Empfangspuffer
268	Word (2)	0	R	Zustand der Eingangs-Steuerleitungen (Bitmap) Bit 0 = CTS (Clear To Send) Bit 1 = DCD (Data Carrier Detected) Bit 2 = RI (Ring Indicator)
290	Word (2)	0	R	Zähler, Sendepuffer war voll <sup>1</sup>
292	Word (2)	0	R	Zähler, Empfangspuffer übergelaufen <sup>1</sup>
294	Word (2)	0	R	Zähler, Zeichenverlust (Overrun) <sup>1</sup>
296	Word (2)	0	R	Zähler, Parity-Fehler <sup>1</sup>
298	Word (2)	0	R	Zähler, Framing-Fehler <sup>1</sup>
300	Word (2)	0	R	Zähler, Break-Fehler <sup>1</sup>

<sup>1</sup> Parameter wird durch Aufruf der Funktion 2 auf Null gesetzt

Nr.	Typ	Init	Zugr	Bedeutung des Parameters
318	Word (2)	0	R/W	Protokolltyp 0 = kein Protokoll 1 = XON/XOFF 2 = RTS/CTS
457	Long (4)	0	R	Nummer des Puffers, der bei SDLC die Längen der empfangenen Frames enthält.
461	Byte	0	R/W	Anzahl der Worte im Userbereich, die verwendet werden sollen für die Initialisierung des SCC
462	Word (2) 127x	0	R/W	Userbereich: Low-Byte = SCC-Register, High-Byte = Data

### Hinweis zu Parameter 32 und 33:

#### Parameter 32:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
Sync7	Sync6	Sync5	Sync4	Sync3	Sync2	Sync1	Sync0	Monosync, 8 Bits
Sync1	Sync0	Sync5	Sync4	Sync3	Sync2	Sync1	Sync0	Monosync, 6 Bits
Sync7	Sync6	Sync5	Sync4	Sync3	Sync2	Sync1	Sync0	Bisync, 16 Bits
Sync3	Sync2	Sync1	Sync0	1	1	1	1	Bisync, 12 Bits
ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0	SDLC
ADR7	ADR6	ADR5	ADR4	x	x	x	X	SDLC (Adress Range)

#### Parameter 33:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
Sync7	Sync6	Sync5	Sync4	Sync3	Sync2	Sync1	Sync0	Monosync, 8 Bits
Sync5	Sync4	Sync3	Sync2	Sync1	Sync0	x	x	Monosync, 6 Bits
Sync15	Sync14	Sync13	Sync12	Sync11	Sync10	Sync9	Sync8	Bisync, 16 Bits
Sync11	Sync10	Sync9	Sync8	Sync7	Sync6	Sync5	Sync4	Bisync, 12 Bits
0	1	1	1	1	1	1	0	SDLC

**Hinweis zur Nutzung des SDLC-Protokolls:**

Folgende Parameter müssen gesetzt werden:

Parameter 11: =2, wenn ESCC verwendet werden soll

Parameter 16: = 5 für Kommunikationstyp SDLC

Parameter 29: = 0 für Sync-Mode Enable

Parameter 32: = Adresse für SDLC-Geräte (s.o.)

Parameter 33: = SDLC-Flag (= 7eh) (s.o.)

Parameter 55: = z.B. 55h (TxCRC Enable, CRC Preset I/O, Rx CRC Enable, Enter Hunt Mode)

## Tabelle der möglichen Fehler für einen Programmabbruch

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

<b>Fehlernummer in Parameter 1</b>	<b>Erklärung</b>
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)
11	Nicht genügend Platz für Empfangspuffer-Reservierung
12	Falscher Parameter für Empfangspuffer-Reservierung
13	Unbekannter Fehler während Empfangspuffer-Reservierung
14	Falscher Parameter für Empfangspuffer-Rerservierung
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

### 13.4.1.5. Die Funktionen und Prozeduren des Basiskommunikationsprogrammes

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 in den Hochsprachen-Bibliotheken ML6BIB und ML6RTBIB aufgerufen wird.

Nr.	Typ	Bedeutung der Funktion
2	P	Start/Restart der Kommunikation (Konfiguration, Speicherreserv.)
5	F	<b>Sendebereitschaft ein- und ausschalten</b> , optional Sendepuffer löschen Hin: outsize = 2 maxinsize = 0 data_out = <i>Kontroll-Wort</i> (s.u.) Rück: error = Fehlernummer  <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
6	F	<b>Sendestatus melden</b> Hin: outsize = 0 maxinsize = 6 indata_var = <i>Rückgabe-Struktur</i> (s.u.) Rück: insize = 6 error = Fehlernummer  <i>Rückgabe-Struktur:</i> Sendestatus (Word), Anzahl Zeichen im Sendepuffer (Long), Bedeutung wie Parameter 256 und 258

Nr.	Typ	Bedeutung der Funktion
7	F	<b>Zeichenkette an Sendepuffer übergeben</b> Hin:    outsize    = Anzahl zu übergebender Zeichen maxinsize = 0 data_out    = <i>Übergabe-Puffer</i> (s.u.) Rück: error       = Fehlernummer  <i>Übergabe-Puffer:</i> Datenstruktur, die <i>outsize</i> zu sendende Bytes enthält.
15	F	<b>Empfangsbereitschaft ein- und ausschalten</b> , optional Empfangspuffer löschen Hin:    outsize    = 2 maxinsize = 0 data_out    = <i>Kontroll-Wort</i> (s.u.) Rück: error       = Fehlernummer  <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
16	F	<b>Empfangsstatus melden</b> Hin:    outsize    = 2 maxinsize = 6 indata_var = <i>Rückgabe-Struktur</i> (s.u.) Rück: insize       = 6 error       = Fehlernummer  <i>Rückgabe-Struktur:</i> Empfangsstatus (Word), Anzahl Zeichen im Empfangspuffer (Long), Bedeutung wie Parameter 262 und 264

Nr.	Typ	Bedeutung der Funktion
17	F	<p><b>Zeichenkette aus Empfangspuffer übernehmen</b></p> <p>Hin: outsize = 0  maxinsize = Anzahl angeforderter Zeichen  indata_var = <i>Rückgabe-Puffer</i> (s.u.)</p> <p>Rück: insize = Anzahl gelesener Zeichen  error = Fehlernummer</p> <p><i>Rückgabe-Puffer:</i>  Datenstruktur, die die gelesenen Zeichen aufnehmen kann, also mindestens <i>maxinsize</i> Byte umfaßt. Nach dem Aufruf sind die ersten <i>insize</i> Byte gültig.</p>
33	F	<p><b>Steuerleitungs-Ausgänge setzen</b></p> <p>Hin: outsize = 4  maxinsize = 0  data_out = <i>Übergabe-Struktur</i> (s.u.)</p> <p>Rück: error = Fehlernummer</p> <p><i>Übergabe-Struktur:</i>  Wort 1: Maske für die betroffenen Steuerleitungen (Word),  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><b>Zustand der Steuerleitungs-Eingänge lesen</b></p> <p>Hin: outsize = 0  maxinsize = 2  indata_var = <i>Rückgabe-Struktur</i> (s.u.)</p> <p>Rück: error = Fehlernummer</p> <p><i>Rückgabe-Struktur:</i>  Information, wie die Steuerleitung gesetzt wurde (Word)  Bitmap: Bit 0 = CTS  Bit 1 = DCD  Bit 2 = RI  Bit 3 = DSR  Bit 4 bis 15 reserviert</p>



## Fehlerrückgabecodes von Funktionen des Programmes M6P0520 / M7P0520 / M8P0520

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 26h	Sendepuffer gesperrt Sendepuffer-Nummer ungültig
6	26h	Sendepuffer-Nummer ungültig
7	22h oder 23h 24h 26h	Sendepuffer gesperrt Sendepuffer voll Sendepuffer-Nummer ungültig
15	22h oder 23h 26h	Empfangspuffer gesperrt Empfangspuffer-Nummer ungültig
16	26h	Empfangspuffer-Nummer ungültig
17	22h oder 23h 26h	Empfangspuffer gesperrt 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.

### **13.5. Installation der Basiskommunikation ohne SNW6/SNW**

Für die Installation der Basiskommunikation ohne SNW6/SNW sind alle Parameter des Programms 520 die mit '\*' gekennzeichnet sind (siehe Kapitel 13.4.1.4.) entsprechend zu setzen. Teilweise sind die Parameter vorinitialisiert.

Für synchrone Kommunikation sind zusätzlich die Parameter 16, 17, 32, 33 und 55 relevant.

#### **Hinweise:**

- Für jeden verwendeten SCC der Multi-COM/MODULAR-4 Karte muß jeweils ein Interrupt-Manager (Programm 500) unter dem von dem SCC verwendeten Interrupt installiert (II-Task, Installationsflags: 0809h, kein Datenbereich benötigt) werden.
- Für jeden verwendeten Kanal ist jeweils eine Kommunikationstask (Programm 520, NI-Task, Installationsflags: 0808h, Datenbereich wird vom Programm selbst reserviert) zu installieren.
- Die Task-Nummer des zugehörigen Interruptmanagers muß in Parameter 6 jeder Kommunikationstask gesetzt werden. Es empfiehlt sich die Zuordnung wie in Tab. 13.1 vorzunehmen.
- Nach Setzen der gewünschten Parameter der Kommunikationstask und Aufruf der Prozedur 2 ist das Basiskommunikations-Programm konfiguriert und der Speicherplatz für die Puffer entsprechend der in den Parametern angegebenen Größe reserviert.
- Um Zeichen empfangen oder senden zu können, müssen noch die Funktionen 5 (Sendebereitschaft) und 15 (Empfangsbereitschaft) aufgerufen werden.
- Hinweise zum Empfangen: siehe Kapitel 13.1.4 und 13.4.1.2.
- Hinweise zum Senden: siehe Kapitel 13.1.4 und 13.4.1.3.

## 13.6. Die Aktions-Filter des Programms M6P0520 / M7P0520 / M8P0520.LIB

Das Basis-Kommunikations-Programm M6P0520 / M7P0520 / M8P0520 kann unter bestimmten, einstellbaren Bedingungen Funktionen anderer Tasks aufrufen. Die Bedingungen stellen Filter für die Ausführung einer Aktion dar. Wir sprechen daher von Aktions-Filtern. Die Übergabe-Konvention für die aufgerufene Funktion entspricht der in Kapitel 10 (System-Subroutinen) bei CALL\_FUNC und in Kapitel 9 bei ml6rt\_call\_func beschriebenen. Daher können Funktionen einer anderen Task abhängig von Ereignissen, die in der Basiskommunikation aufgetreten sind, aufgerufen werden. Dieser Mechanismus ist in erster Linie zur Implementierung von Protokollen vorgesehen. Zur Zeit sind 10 Filter-Aktionen definiert. Sie können mit dem Parameter "Aktions-Filter-Maske" vom Programm einzeln aktiviert oder unterdrückt werden. Jedem Aktions-Filter ist ein Parameter-Satz von 12 Byte Länge zugeordnet (Parameter siehe Kapitel 13.6.2). Dieser enthält die Nummer (2 Byte) der Task, die die aufzurufende Funktion enthält, und die Nummer der Funktion (2 Byte), die bei zutreffender Bedingung aufgerufen werden soll. Es folgt ein 4 Byte langes Argument, das von Filter-Aktion zu Filter-Aktion unterschiedliche Bedeutung hat. Danach kommt die Funktionsadresse der aufzurufenden Funktion im Format Segment:Offset (4 Byte). Diese Adresse wird beim Aufruf der Startfunktion (Prozedur 2) vom Programm M6P0520/M7P0520/M8P0520 berechnet, wenn das zugehörige Bit in der Aktions-Filter-Maske gesetzt ist. Die Task, die die aufzurufende Funktion enthält, muß zu diesem Zeitpunkt bereits installiert sein.

Der aufgerufenen Funktion wird eine Struktur mit diversen Status-Informationen übergeben (s.u.). Je nach Filter-Aktion muß eine Funktion einen Rückgabewert liefern. Der Aufbau dieser Funktion kann wie folgt aussehen (siehe auch Kapitel 9):

in C:           **VOID**                   **PFAR** name (USHORT task, USHORT insize, VOID \*inptr, USHORT outsize, VOID \*outptr);

in Pascal:      **PROCEDURE** name (task, insize : Word; VAR inptr; outsize: Word; VAR outptr);

Parameter	<i>task:</i>	Nummer der eigenen Task.
	<i>insize:</i>	Anzahl der Daten, die der Funktion übergeben werden (= 19).
	<i>inptr:</i>	Zeiger auf die Übergabedaten (Struktur <i>p0520_status</i> ).
	<i>outsize:</i>	Maximale Anzahl der Daten, die zurück erwartet werden (in Byte).

*outptr*:        Zeiger auf den Bereich, in dem die Task ihre Daten  
                  zurückgibt (Rückgabewert).

Bei Aufruf der Funktion aus einem Assembler-Programm sind die Register wie folgt vorbesetzt:

dx     = Tasknummer der aufzurufenden Task  
di     = Länge der Übergabedaten (=19)  
si     = Maximale Anzahl der Daten, die zurück erwartet werden (in Byte).  
eax    = Adresse (physikalisch) der Übergabedaten (Struktur *p0520\_status*)  
ecx    = Adresse (physikalisch) des Rückgabewertes  
         (Aktions-Filter 0 und Aktions-Filter 1)

Die Struktur *p0520\_status* ist ein Ausschnitt aus dem Parameterbereich des Programms M6P0520 / M7P0520 / M8P0520.LIB. Die genaue Bedeutung der einzelnen Variablen entnehmen Sie bitte der Tabelle der Parameter des Programms M6P0520 / M7P0520 / M8P0520.LIB (Parameter 256 bis 271) . Hier die Struktur-Definition in C-Syntax:

```
struct _p0520_status {  
    ushort sstatus;            // Sendestatus  
    ulong  scout;             // Anzahl Zeichen im Sendepuffer  
    ushort rstatus;           // Empfangsstatus  
    ulong  rcount;            // Anzahl Zeichen im Empfangspuffer  
    ushort control_status;    // Zustand der Steuerleitungen  
    byte   last_received;     // zuletzt empfangenes Zeichen(nur  
                             // Filter-Aktion 1 + 2)  
    ulong  s_ex_count;        // Anzahl Zeichen im Sende-Expreßpuffer  
} p0520_status;
```

### 13.6.1. Aktions-Filter und zugehörige Filter-Argumente

In Parameter 56 von Programm 520 (siehe Kapitel 13.6.2) können z.Zt. 10 verschiedene Aktions-Filter ausgewählt werden.

---

#### Aktions-Filter 0

Bei jedem empfangenen Zeichen soll die zugeordnete Funktion aufgerufen werden.

Das Argument hat keine Bedeutung.

Im Rückgabewert (Byte) der Funktion wird festgelegt, ob das empfangene Zeichen nachträglich noch im Empfangs-Puffer gespeichert wird (= 1) oder nicht (= 0).

---

#### Aktions-Filter 1

Die zugeordnete Funktion wird aufgerufen wenn ein empfangenes Zeichen sich in einem Vergleichsstring befindet. Das erste Zeichen im Vergleichsstring enthält die Anzahl der nachfolgenden zu vergleichenden Zeichen.

Das Argument enthält den Zeiger (Format Segment:Offset) auf den Vergleichsstring.

Im Rückgabewert (Byte) der Funktion wird festgelegt, ob das empfangene Zeichen nachträglich noch im Empfangs-Puffer gespeichert wird (= 1) oder nicht (= 0).

---

#### Aktions-Filter 2

Die zugeordnete Funktion wird bei Empfangsfehlern aufgerufen.

Im Argument (untere 2 Byte) können Empfangsfehler maskiert werden. Die Bedeutung der Maske entspricht der Beschreibung des Parameters "Empfangsstatus". Ein gesetztes Bit (=1) führt zum Aufruf der Funktion, ein nicht gesetztes unterbindet den Aufruf.

Es gibt kein Rückgabewort.

---

#### Aktions-Filter 3

Funktionsaufruf bei Empfangs-Füllstands-Überschreitung.

Das Argument enthält die obere Empfangs-Füllstands-Grenze.

Es gibt kein Rückgabewort.

**Wenn dieses Filter benutzt wird, muß auch Filter 4 benutzt werden.**

---

---

**Aktions-Filter 4**

Funktionsaufruf bei Empfangs-Füllstands-Unterschreitung.

Das Argument enthält die untere Empfangs-Füllstands-Grenze.

Es gibt kein Rückgabewort.

**Wenn dieses Filter benutzt wird, muß auch Filter 3 benutzt werden.**

---

**Aktions-Filter 5**

Funktionsaufruf bei Sende-Füllstands-Überschreitung

Das Argument enthält die obere Sende-Füllstands-Grenze.

Es gibt kein Rückgabewort.

**Wenn dieses Filter benutzt wird, muß auch Filter 6 benutzt werden.**

---

**Aktions-Filter 6**

Funktionsaufruf bei Sende-Füllstands-Unterschreitung.

Das Argument enthält die untere Sende-Füllstands-Grenze.

Es gibt kein Rückgabewort.

**Wenn dieses Filter benutzt wird, muß auch Filter 5 benutzt werden.**

---

**Aktions-Filter 7**

Funktionsaufruf, wenn alle Zeichen physikalisch gesendet sind.

Das Argument hat keine Bedeutung.

Es gibt kein Rückgabewort.

---

**Aktions-Filter 8**

Funktionsaufruf, wenn sich eine Eingangs-Steuerleitung ändert. Abhängig von der verwendeten Hardware handelt es sich z.B. um die Steuerleitungs-Eingänge CTS, DCD, RI, DSR etc.

Das Argument enthält eine Maske für Steuerleitungs-Eingänge, die der Bedeutung des Parameters "Status der Steuerleitungen" entspricht. Ein gesetztes Bit (=1) führt zum Aufruf der Funktion, ein nicht gesetztes unterbindet den Aufruf.

Es gibt kein Rückgabewort.

---

---

**Aktions-Filter 9**

Funktionsaufruf bei nicht behebbarem Laufzeitfehler.

Die Fehlerursache muß durch Lesen des Parameter 1 des Programms M6P0520/M7P0520/M8P0520.LIB ermittelt werden.

Das Argument hat keine Bedeutung.

Es gibt kein Rückgabewort.

---

**13.6.2. Erweiterte Parameterstruktur von Programm 520**

---

Nr.	Typ	Init	Zugr	Bedeutung des Parameters
56	Long (4)	0	R/W	Aktions-Filter-Maske (Bitmap), Aufruf bei Bit 0 = 1: jedem empfangenen Zeichen Bit 1 = 1: empfangenem Zeichen im Vergleichs-string enthalten Bit 2 = 1: Fehler beim Empfang (Overflow, Parity, etc.) Bit 3 = 1: Empfangs-Füllstands-Überschreitung Bit 4 = 1: Empfangs-Füllstands-Unterschreitung Bit 5 = 1: Sende-Füllstands-Überschreitung Bit 6 = 1: Sende-Füllstands-Unterschreitung Bit 7 = 1: leerem Empfangspuffer Bit 8 = 1: Änderung der Steuerleitungs-Eingänge (z.B. CTS, DCD, RI, DSR). Die Verfügbarkeit der Steuerleitungen ist von der eingesetzten Hardware abhängig (siehe Hardwarebeschreibung). Bit 9 = 1: nicht behebbarem Laufzeitfehler Bit 10 bis 31 sind reserviert

---

Nr.	Typ	Init	Zugr	Bedeutung des Parameters
60				<b>Aktions-Filter 0:</b> bei jedem empfangenen Zeichen Aktions-Task
	Word (2)	0	R/W	Aktions-Funktion
	Word (2)	0	R/W	Argument: ohne Bedeutung
	Long (4)	0	-	Funktionsadresse (Segment:Offset)
	Long (4)	0	R	Rückgabewert (Byte): 0 = Zeichen nach Aufruf des Filters noch im Empfangspuffer speichern, 1 = nicht speichern
72				<b>Aktions-Filter 1:</b> empfangenes Zeichen ist im Vergleichsstring (siehe Parameter 320)
	Word (2)	0	R/W	Aktions-Task
	Word (2)	0	R/W	Aktions-Funktion
	Long (4)	0	R/W	Argument: Zeiger auf Vergleichsstring (Segment:Offset), Aufbau s.u.
	Long (4)	0	R	Funktionsadresse (Segment:Offset) Rückgabewert (Byte): 0 = Zeichen nach Aufruf des Filters noch im Empfangspuffer speichern, 1 = nicht speichern
84				<b>Aktions-Filter 2:</b> Empfangsfehler
	Word (2)	0	R/W	Aktions-Task
	Word (2)	0	R/W	Aktions-Funktion
	Long (4)	0	R/W	Argument: Fehlermaske (Bit 0 bis 15, Bitmap), Bedeutung wie Parameter "Empfangsstatus"
	Long (4)	0	R	Funktionsadresse (Segment:Offset)
96				<b>Aktions-Filter 3:</b> Empfangs-Füllstands-Überschreitung
	Word (2)	0	R/W	Aktions-Task
	Word (2)	0	R/W	Aktions-Funktion
	Long (4)	0	R/W	Argument: obere Empfangs-Füllstands-Grenze
	Long (4)	0	R	Funktionsadresse (Segment:Offset)
108				<b>Aktions-Filter 4:</b> Empfangs-Füllstands-Unterschreitung
	Word (2)	0	R/W	Aktions-Task
	Word (2)	0	R/W	Aktions-Funktion
	Long (4)	0	R/W	Argument: untere Empfangs-Füllstands-Grenze
	Long (4)	0	R	Funktionsadresse (Segment:Offset)



Nr.	Typ	Init	Zugr	Bedeutung des Parameters
120				<b>Aktions-Filter 5:</b> Sende-Füllstands-Überschreitung
	Word (2)	0	R/W	Aktions-Task
	Word (2)	0	R/W	Aktions-Funktion
	Long (4)	0	R/W	Argument: obere Sende-Füllstands-Grenze
	Long (4)	0	R	Funktionsadresse (Segment:Offset)
132				<b>Aktions-Filter 6:</b> Sende-Füllstands- Unterschreitung
	Word (2)	0	R/W	Aktions-Task
	Word (2)	0	R/W	Aktions-Funktion
	Long (4)	0	R/W	Argument: untere Sende-Füllstands-Grenze
	Long (4)	0	R	Funktionsadresse (Segment:Offset)
144				<b>Aktions-Filter 7:</b> Alle Zeichen gesendet
	Word (2)	0	R/W	Aktions-Task
	Word (2)	0	R/W	Aktions-Funktion
	Long (4)	0	R/W	Argument: ohne Bedeutung
	Long (4)	0	R	Funktionsadresse (Segment:Offset)
156				<b>Aktions-Filter 8:</b> Änderung einer Steuerleitung
	Word (2)	0	R/W	Aktions-Task
	Word (2)	0	R/W	Aktions-Funktion
	Long (4)	0	R/W	Argument: Maske für Steuerleitungen, Bedeutung wie bei Status der Steuerleitungen
	Long (4)	0	R	Funktionsadresse (Segment:Offset)
168				<b>Aktions-Filter 9:</b> nicht behebbbarer Laufzeitfehler
	Word (2)	0	R/W	Aktions-Task
	Word (2)	0	R/W	Aktions-Funktion
	Long (4)	0	R/W	Argument: ohne Bedeutung
	Long (4)	0	R	Funktionsadresse (Segment:Offset)
180	6 * 12 = 72 Byte	0	R	<b>Aktions-Filter 10 bis 15:</b> reserviert Task, Funktion, Argument, Funktionsadresse (Auf- bau wie Filter 0)

Nr.	Typ	Init	Zugr	Bedeutung des Parameters
318	Word (2)	0	R/W	<b>Protokolltyp</b> 0 = kein Protokoll 1 = XON/XOFF : Aktions-Filter 1, 3 und 4 werden verwendet, die Funktionen 38, 36 und 37 werden aufgerufen 2 = RTS/CTS: Aktions-Filter 3, 4 und 8 werden verwendet, die Funktionen 40, 41 und 42 werden aufgerufen
320	16 Byte	0	R/W	<b>Vergleichsstring für Filteraktion 1</b> Aufbau: Das erste Zeichen enthält die Anzahl der zu vergleichenden Zeichen im String, danach folgen die Zeichen (maximal 15), bei Protokolltyp = 1 werden automatisch die XON/XOFF Zeichen (11H, 13H) eingetragen.