isel - TMO 4403
(PICMIC)

OEM - Schrittmotormodul
mit DIN-Meßbus-Ankopplung

Anleitung zur Programmierung

Hardwarebeschreibung

Die in dieser Druckschrift enthaltenen Informationen, technischen Daten und Maßangaben entsprechen dem neuesten technischen Stand zum Zeitpunkt der Veröffentlichung. Etwa dennoch vorhandene Druckfehler und Irrtümer können jedoch nicht ausgeschlossen wer-den. Für Verbesserungsvorschläge und Hinweise auf Fehler sind wir dankbar.

Es wird darauf hingewiesen, dass die in unseren Druckschriften verwendeten Soft- und Hardwarebezeichnungen der jeweiligen Firmen im allgemeinen warenzeichen-, marken- oder patentrechtlichem Schutz unterliegen.

Alle Rechte vorbehalten. Kein Teil unserer Druckschriften darf in irgendeiner Form (Druck, Fotokopie oder einem anderen Verfahren) ohne schriftliche Genehmigung der Firma *iselautomation* reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, ver-vielfältigt oder verbreitet werden.

Stand: Mai 1999

Hersteller: Fa. *iselautomation* KG
Im Leibolzgraben 16
D-36132 Eiterfeld
Fax: (06672) 898-888
e-mail: automation@isel.com
http://www.isel.com

Inhaltsverzeichnis

1. Einleitung

1.1. Vorwort

Seit vielen Jahren ist die Firma ***isel***-automation bekannt für Antriebssysteme und Automatisierungslösungen auf der Basis von Schrittmotorsteuerungen. Weltweit haben sich Schrittmotoren im kleinen und mittleren Leistungsbereich durchgesetzt. In Antriebssystemen mit mittleren Genauigkeits- und Dynamikanforderungen sind sie quasi nicht mehr wegzudenken. Schrittmotorsteuerungen arbeiten im Allgemeinen nicht mit geschlossenen Regelkreisen, so daß teure Sensoriken und Auswerteelektroniken entfallen. Dadurch ergibt sich neben einfachem Aufbau und Inbetriebnahme auch ein günstiges Preis-Leistungs-Verhältnis.
Im Bereich der Automatisierungstechnik sind Einachslösungen zur Automatisierung wiederkehrender Bewegungsabläufe weit verbreitet. Dem Stand der Technik folgend geht der Trend heute eindeutig hin zu vernetzbaren Antriebssystemen mit intelligenten Aktoren. Diesem Trend entsprechend sind wir mit unserer intelligenten Einachsschrittmotorsteuerung mit DIN-Meßbus-Ankopplung einen ersten Schritt in Richtung vernetzte Antriebssyteme gegangen. Unser Modul ist in erster Linie als Automatisierungsbaustein für den OEM - Anwender gedacht, welcher, in mit DIN - Meßbus vernetzten Systemen, Aktoren für einfache Bewegungsabläufe benötigt. Das ***isel***-PICMIC-Modul stellt auf der Basis des DIN-Meßbus entsprechende Funktionalitäten zu einem sehr günstigen Preis-Leistungs-Verhältnis bereit.
Diese Anleitung soll einen Einblick in die Funktionen und Funktionalitäten des ***isel***-PICMIC-Moduls und dessen Ankopplung über den DIN-Meßbus geben. Darüber hinaus soll diese Beschreibung dem OEM-Anwender und Programmierer als Handbuch zur Programmierung des ***isel***-PICMIC-Moduls dienen.

1.2. Vernetzungen in der Automatisierungstechnik

Feldbussysteme haben in der Automatisierungstechnik inzwischen weite Verbreitung gefunden. Gegenüber herkömmlichen Übertragungssystemen, wo die einzelnen Geräte parallel verdrahtet sind, bringen seriell verdrahtete Systeme, so auch Feldbussysteme, dem Anwender neben geringeren Kosten auch eine höhere Leistungsfähigkeit. Die Vorteile derartiger Systeme liegen auf der Hand. Dies sind neben dem geringeren Verkabelungsaufwand vor allem bessere Überschaubarkeit, einfachere Projektierung, einfachere Erweiterbarkeit und kostengünstigere Wartung. Besonders hervorzuheben ist die Modularität eines solchen Systems. Insbesondere der DIN-Meßbus ist einfach erweiterbar und kann ohne großen Aufwand modifiziert werden. Der PC kann dabei mit einer Vielzahl von Geräten, auch in großen Entfernungen, verbunden werden, ohne daß ein derartiges Projekt an fehlenden Einsteckplätzen für Schnittstellenkarten scheitern könnte. Feldbussyteme sind für den Einsatz in der Industrie konzipiert und orientieren sich im Gegensatz zu lokalen Netzwerken (LAN) in der Bürokommunikation an Meß- und Regelaufgaben vor Ort. Somit sind Feldbussyteme Grundlage und Voraussetzung für die Vernetzung von dezentralen, intelligenten Aktoren und Sensoren in der Automatisierungstechnik.

2. DIN-Meßbus nach DIN 66348 Teil 2

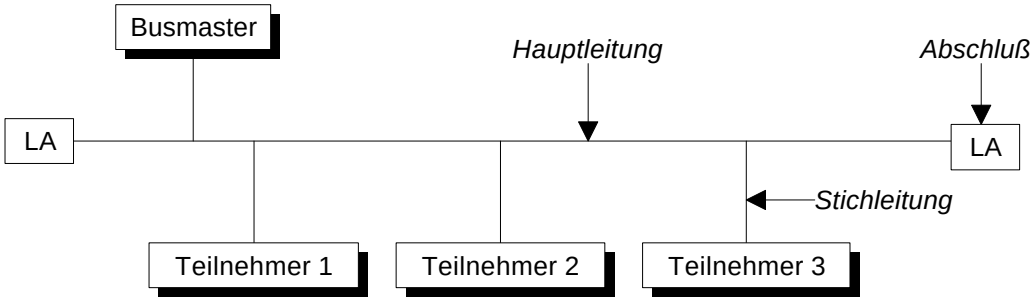
2.1. Allgemeines

Der DIN-Meßbus entstand in Zusammenarbeit zwischen Herstellern im Bereich der Fertigungsmeßtechnik, Anwendern aus der Automobilindustrie und der Physikalisch - Technischen Bundesanstalt als typische Anwendernorm. Er ist das erste genormte und eichfähige Feldbussystem für sichere Datenübertragung in der Produktion. Der DIN-Meßbus wurde unter der DIN 66348 Teil 2 genormt und ist gemäß der Richtlinie 50.20 der Physikalisch-Technischen Bundesanstalt ausdrücklich für das Meßwesen zugelassen. Zur optimalen Nutzung grenzt der DIN-Meßbus sein Einsatzgebiet sinnvoll ab. Es liegt zwischen der sogenannten Sensor/Aktor- und der LAN-Ebene und beschränkt sich auf die Kommunikation zwischen intelligenten Sensoren, Meß- und Prüfgeräten, Steuerungen, BDE/DNC-Terminals sowie anderen Datenein- und ausgabestationen.
Im Vergleich mit anderen Feldbussystemen weist der DIN-Meßbus entscheidende Vorzüge auf, die Ergebnis des durchdachten und praktikablen, von Anwendern entwickelten Konzeptes sind. Diese beinhaltet Vierdrahttechnik, galvanische Trennung, Master-Slave-Struktur und Geräteanschluß über Stichleitungen. Mit diesen Festlegungen, die eine ganze Reihe von Leistungsvorteilen und erhöhte Sicherheit mit sich bringen, ist der DIN-Meßbus einzigartig unter den Feldbussystemen. Besonderen Wert legt der DIN-Meßbus auf sichere Datenübertragung, einfache Handhabung und flexiblen Einsatz. Die Norm DIN 66348 erzwingt Gerätekompatibilität, ist bewußt einfach gehalten und jedermann zugänglich. Sie unterscheidet sich auch in diesem Punkt von anderen auf dem Markt befindlichen und firmenspezifischen Systemen.
Für den Anwender und Gerätehersteller ergeben sich mehrfache Kosten- und Leistungsvorteile. Zum einen können benötigte Sondergeräte auch von kleineren Speziallieferanten kostengünstig mit der DIN-Meßbus-Schnittstelle ausgerüstet und zum Aufbau hoch effizienter, heterogener Systeme herangezogen werden. Zur Ankopplung bereits vorhandener Geräte mit V.24- und anderen gängigen Schnittstellen stehen Schnittstellenkonverter verschiedenster Hersteller auf dem Markt zur Verfügung. Zum anderen sind auch komplexe Systeme in kurzen Zeiten zu projektieren, was der Leistungsfähigkeit und Angebotsvielfalt zugute kommt.
Der DIN-Meßbus ist entsprechend den Anforderungen vor Ort streng hierarchisch gegliedert und besitzt nur eine Leitstation, die bis zu 31 Teilnehmer überwacht. Die Leitstation kann jederzeit steuernd in das System eingreifen. In Verbindung mit getrennten Sende- und Empfangsleitungen, Verwendung finden hier herkömmliche verdrillte Kupferdrahtleitungen, gewährleistet der DIN-Meßbus eine hervorragende Verfügbarkeit und hohe Datensicherheit.
Vorteile bestehen auch in der Installation. Beim DIN-Meßbus können unter Verwendung einfacher Zwischenverstärker Haupt- und Stichleitungen beliebig verlängert und Baumstrukturen aufgebaut werden. Dies kommt bestehenden Betriebsstrukturen, in die der Feldbus eingeflochten werden muß, in idealer Weise entgegen und ermöglicht eine räumlich ungebundene Geräteaufteilung.

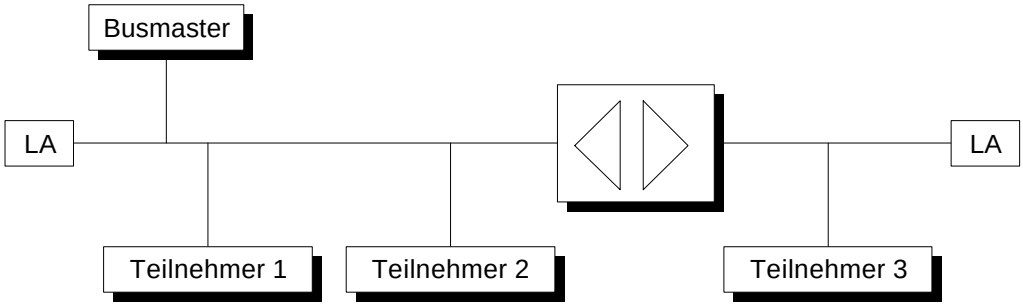
2.2. Eigenschaften

2.2.1. Struktur des Bussystems

Der DIN-Meßbus wird in einer Linienstruktur realisiert, die als Vierdrahtleitung ausgeführt wird. An die Hauptleitung, welche beidseitig mit Busabschlüssen versehen ist, werden alle Teilnehmer über kurze Stichleitungen angeschlossen.



Die maximal zulässige Länge der Hauptleitung ist in der DIN 66348 auf 500m festgelegt, größere Entfernungen können jedoch problemlos durch zwischengeschaltete Verstärker überbrückt werden. Als Stichleitungslänge zu den einzelnen Teilnehmern sind maximal 5m zulässig.



Als elektrisches Übertragungsverfahren wird beim DIN-Meßbus die symmetrische Spannungsschnittstelle nach EIA RS485 benutzt, woraus eine maximale Teilnehmerzahl von 32 resultiert.

2.2.2. Physikalische Eigenschaften

- Übertragung mit Differenzspannungssignal nach EIA RS 485
- Galvanische Trennung, Schirmung und Potentialausgleich
- Bus-Leitungslänge bis zu 500m bei max. Übertragungsrate, beliebig verlängerbar
- Stichleitungslänge bis 5m, beliebig verlängerbar
- Übertragungsraten 110bit/s bis 1Mbit/s (darunter alle PC-Bitraten), einstellbar
- Full Duplex Betrieb (4-Draht), dadurch hohe Busverfügbarkeit und einfache Koppelschaltungen wie Repeater und Ankoppler an Lichtwellenleiter
- Start-Stop-Übertragung, ASCII Zeichensatz mit gerader Parität (7-Bit-Code)
- geringe Material- und Entwicklungskosten für die Schnittstelle

2.2.3. Protokolleigenschaften

- zentralgesteuertes Zugriffsverfahren (Master-Slave) für bis zu 31 Teilnehmer
- quittierter Datenaustausch in drei Schritten: Aufforderungsphase, Datenübermittlungsphase, Abschlußphase
- gesicherte Blockübertragung mit bis zu 128 Bytes Länge
- Datensicherung durch Querparität und Blockprüfzeichen (HD=4)
- Block- und Quittierungswiederholung bei Störungen (max. 3-fach)
- Zeitüberwachung T_A für Quittungen und T_C für Teilnehmersendezeit
- Querverkehr unter Kontrolle der Leitstation
- Rundrufmöglichkeit durch feste Rundrufadresse für alle Teilnehmer (Broadcast)
- Abbruch einer Datenübermittlung jederzeit durch die Leitstation möglich (Störung oder Eilmeldung)
- flexibles Busmanagement, frei konfigurierbares Polling
- schnelle Ergebnisbearbeitung durch sehr kurze Statusabfrage

2.2.4. Übertragungsformat

Zur Datenübertragung benutzt der DIN-Meßbus 7-Bit-ASCII-Zeichen, wobei die einzelnen Zeichen folgendermaßen aufgebaut sind:

- Startbit low
- 7-Bit-Zeichen, niederwertigstes Bit beginnt
- Paritätsbit, mit gerader Parität
- Stopbit high

Grundzustand	Start	Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Parity	Stop	

Der verwendete 7-Bit-Zeichencode umfaßt alle ASCII-Zeichen von 0 - 127. Folgende Zeichen werden dabei vom DIN-Meßbus als Steuerzeichen verwendet:

Dezimal	ASCII	HEX	Binär
001	SOH	01	00000001
002	STX	02	00000010
003	ETX	03	00000011
004	EOT	04	00000100
005	ENQ	05	00000101
006	ACK	06	00000110
016	DLE	10	00010000
021	NAK	15	00010101
023	ETB	17	00010111

Alle anderen Zeichen dürfen zur Informationsübertragung benutzt werden.

Für die Datenübertragung sind in der DIN-Meßbus-Norm folgende Baudraten festgelegt:

110, 300, 600, 1200, 2400, 4800, 9600 und 19200 Baud.

Dabei muß die Übertragungsrate von 9600 Baud von jedem Busteilnehmer realisiert werden können. Übertragungsraten oberhalb von 19200 Baud sind zwar in der Norm nicht festgelegt, aber erlaubt. Dabei werden ganzzahlige Vielfache von 9600 oder 64000 Baud ausdrücklich empfohlen.

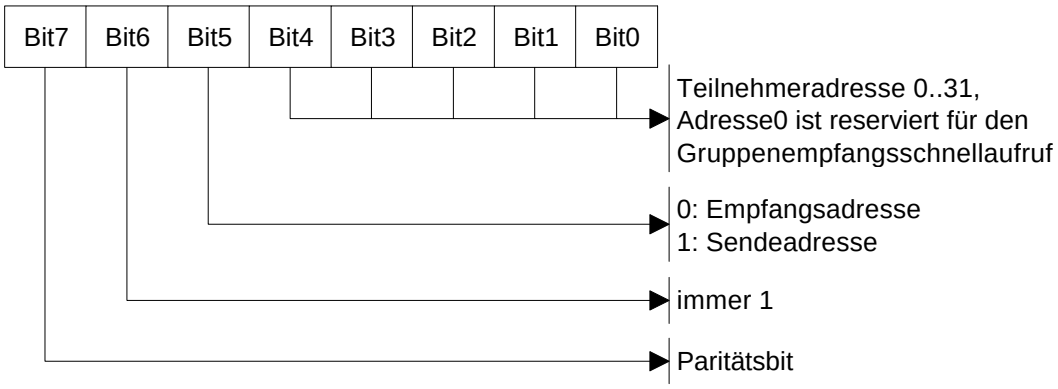
2.2.5. Bedeutung der Steuerzeichen

Zur Steuerung der Datenübertragung zwischen der Leitstation und den Teilnehmern werden vom DIN-Meßbus-Protokoll verschiedene Steuerzeichen verwendet, die innerhalb eines Datenblocks nicht vorkommen dürfen. Diese Steuerzeichen finden folgende Anwendungen:

Steuerzeichen	HEX	Verwendung	Bedeutung
<SADR>ENQ	xx05	Aufforderungsphase	Polling, Sendeaufruf der Leitstation an den Teilnehmer mit entsprechender Adresse
<EADR>ENQ	xx05	Aufforderungsphase	Auswahl, Empfangsaufruf der Leitstation an den Teilnehmer mit der entsprechenden Adresse
		Übermittlungsphase	Aufforderung an die Datensenke, eine Rückmeldung zu senden
<SADR>DLE30	xx1030	Aufforderungsphase	Positive Rückmeldung des Teilnehmers mit der angegebenen Adresse
<EADR>DLE30	xx1030	Aufforderungsphase	Positive Rückmeldung des Teilnehmers mit der angegebenen Adresse
DLE31	1031	Übermittlungsphase	Positive Rückmeldung
<SADR>NAK	xx15	Aufforderungsphase	Negative Rückmeldung des Teilnehmers mit der angegebenen Adresse
<EADR>NAK	xx15	Aufforderungsphase	Negative Rückmeldung des Teilnehmers mit der angegebenen Adresse
NAK	15	Übermittlungsphase	Negative Rückmeldung
SOH	01	Aufforderungsphase	Bei Querverkehr markiert SOH den Anfang des Kopfes. Die Empfangsadresse des Zieles folgt direkt

			dahinter.
STX	02	Übermittlungsphase	Beginn eines Datenblocks
ETB	17	Übermittlungsphase	Ende eines Datenblocks
ETX	03	Übermittlungsphase	Ende eines Datenblocks, gleichzeitig Ende eines Textes
EOT	04	zu jeder Zeit	Ende der Datenverbindung. EOT kann zu jedem Zeitpunkt auftreten und muß von jedem Teilnehmer zu jeder Zeit erkannt werden.

Die in den Telegrammen verwendeten Teilnehmeradressen (<SADR> und <EADR>, für Sende- bzw. Empfangsadresse) haben folgende Aufbau:



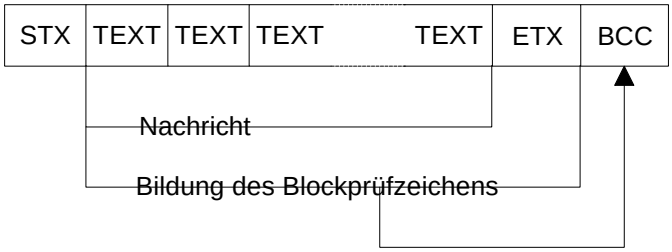
2.2.6. Datensicherungsverfahren

Zur Sicherung der Datenübertragung werden die übertragenen Nutzdaten durch Prüfzeichen ergänzt. Zwei unterschiedliche Verfahren kommen dabei zur Anwendung:

- Alle Zeichen, also auch sämtliche Steuerzeichen, werden mit einem Paritätsbit (Bit7) übertragen. Das Paritätsbit wird so gebildet, daß die Modulo-2-Summe des Zeichens einschließlich Paritätsbit 0 ergibt (gerade Parität).

- Alle Nutzdatenblöcke werden mit einem Blockprüfzeichen BCC ergänzt. Dabei repräsentieren Bit0 bis Bit6 des BCC je ein Paritätsbit, welches aus sämtlichen Bits der Nutzdatenbytes an gleicher Bitposition gewonnen wird. Dieses Paritätsbit wird ebenfalls mit gerader Parität erzeugt. Bit7 des BCC wird, wie bei den Nutzdatenbytes, als Paritätsbit für das BCC erzeugt und sichert so das Paritätsbit ab.

Damit erhält ein Datenblock folgendes Aussehen:



Zur Übertragung eines Datenblocks werden von der Datenquelle alle notwendigen Prüfbits und das Blockprüfzeichen erzeugt und zusammen mit der Nachricht gesendet. Die Überprüfung durch die Datensenke nach Empfang des Datenblocks erfolgt in der Art, daß alle Prüfzeichen und Prüfbits von der Datensenke ein zweites Mal erzeugt und mit den empfangenen Prüfdaten verglichen werden. Ist keine Übereinstimmung vorhanden, führt dies zu einem Fehler und somit zu einer negativen Rückmeldung (NAK). Die übertragene

Nachricht ist nur gültig, wenn die Überprüfung erfolgreich war und eine positive Rückmeldung (DLE31) erzeugt wurde. Erst nach Empfang dieser positiven Rückmeldung darf die Datenquelle die Nachricht mit neuen Daten überschreiben.

2.3. Realisierung der Datenübertragung

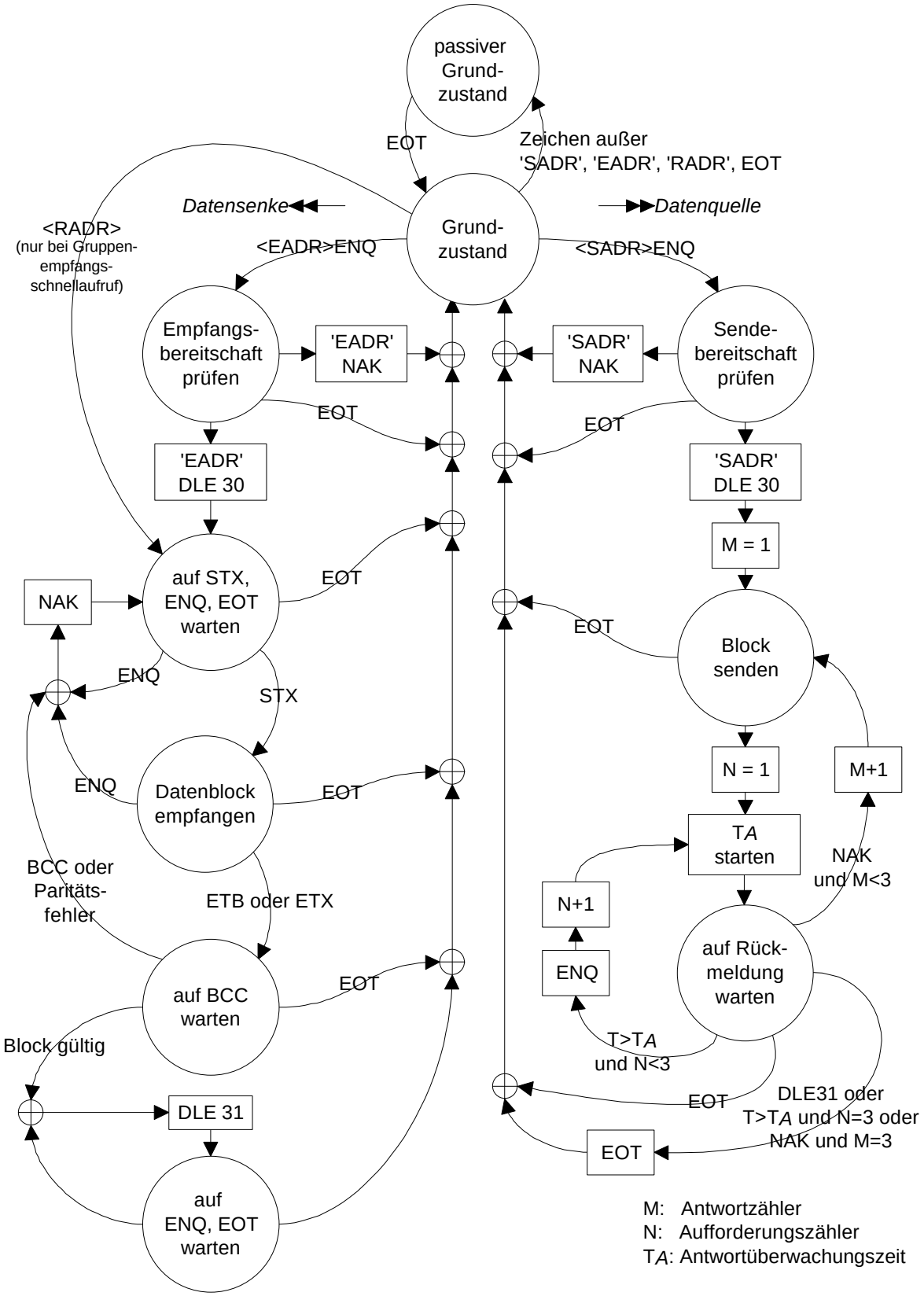
2.3.1. DIN-Meßbus-Protokoll

Das zur Datenübertragung notwendige Protokoll für den DIN-Meßbus kann sehr einfach softwaremäßig erzeugt werden. Eine Implementation der notwendigen Routinen auf einem Mikroprozessor, einem PC oder einem Rechnersystem stellt keine größere Schwierigkeit dar. Auch dies ist ein Vorteil gegenüber anderen Feldbussystemen. Grundsätzlich sind für jeden Teilnehmer zwei Programmteile erforderlich, ein Empfangs- und ein Sendeprogramm. Die gesamte Steuerung und Überwachung des Datenverkehrs wird durch die Leitstation, die in jedem DIN-Meßbus-System vorhanden sein muß, realisiert. Auch der Querverkehr zwischen zwei Teilnehmerstationen, welcher nur selten realisiert wird, ist nur unter Kontrolle der Leitstation möglich. Die Leitstation ist also für den Datenverkehr auf dem Bus verantwortlich. Sie verteilt die Informationen und gibt jedem Teilnehmer in regelmäßigen Abständen die Möglichkeit Daten zu senden. Reihenfolge und Priorität der einzelnen Teilnehmer können durch ein entsprechendes Leitstationenprogramm den jeweiligen Anwendungen beliebig angepaßt werden. Zur Adressierung der Teilnehmer erhält jeder eine Sende- und Empfangsadresse. Diese muß an jedem Teilnehmer einstellbar sein (durch Hard- oder Software). Zur Realisierung eines Datenaustausches wird der Teilnehmer von der Leitstation entsprechend aufgefordert. Folgende zwei grundlegende Aufrufe müssen dabei realisierbar sein:

Sendeaufruf: Die Leitstation sendet <SADR>ENQ und wartet auf die Reaktion des Teilnehmers. Dieser antwortet entweder mit einer negativen Rückmeldung (<SADR>NAK), falls keine Sendedaten vorliegen, oder mit einer positiven Rückmeldung (<SADR>ENQ) gefolgt von den vorhandenen Daten.

Empfangsaufruf: Die Leitstation sendet <EADR>ENQ und wartet auf die Reaktion des Teilnehmers. Dieser antwortet mit einer negativen Rückmeldung (<EADR>NAK), falls er nicht empfangsbereit ist, oder mit einer positiven Rückmeldung (<EADR>DLE30), worauf die Leitstation sofort mit der Übertragung der Daten beginnt.

Die Datenübertragung wird mit dem Steuerzeichen EOT beendet. Folgende Darstellung soll die notwendigen Zustände einer Teilnehmerstation verdeutlichen:



Grundlegend besteht jeder Datenverkehr aus drei Phasen:

- Aufforderungsphase

- Datenübermittlungsphase
- Abschlußphase

Diese drei Phasen werden bei einer ordnungsgemäßen Datenübertragung nacheinander durchlaufen.

2.3.2. Aufforderungsphase

Die Aufforderungsphase wird immer von der Leitstation eingeleitet. Dabei gibt es nur folgende drei Varianten: den Sendeaufruf, den Empfangsaufruf und den Gruppenempfangsschnellaufruf.

1. Sendeaufruf: Die Leitstation übergibt mit dem Sendeaufruf <SADR>ENQ das Senderecht an den Teilnehmer mit der entsprechenden Adresse und startet die Antwortüberwachungszeit T_A. Darauf hin können sich folgende Reaktionen ergeben:
   - Positive Rückmeldung <SADR>DLE30. Der Teilnehmer verfügt über Daten und übernimmt das Senderecht. Die Leitstation startet die Betriebsüberwachungszeit T_C, innerhalb der die Datenübertragung beginnen muß. Wird diese Zeit nicht eingehalten, so bricht die Leitstation die Datenverbindung durch senden von EOT ab.
   - Negative Rückmeldung <SADR>NAK. Der Teilnehmer verfügt nicht über aktuelle Daten. Statt der Datenübermittlungsphase folgt hier direkt die Abschlußphase.
   - Keine, fehlerhafte oder unzulässige Rückmeldung: Nach Ablauf der Antwortüberwachungszeit geht die Leitstation direkt in die Abschlußphase über.

2. Empfangsaufruf: Mit dem Empfangsaufruf <EADR>ENQ fragt die Leitstation den gewünschten Teilnehmer an, ob er für den Empfang eines Datenblocks bereit ist. Wie beim Sendeaufruf startet die Leitstation direkt nach der Übertragung die Antwortüberwachungszeit T_A. Die möglichen Reaktionen des Teilnehmers sind hier:
   - Positive Rückmeldung <EADR>DLE30. Der Teilnehmer ist bereit und geht direkt in die Datenübermittlungsphase über. Die Leitstation sendet die entsprechenden Daten direkt nach Empfang der Rückmeldung.
   - Negative Rückmeldung <EADR>NAK. Der Teilnehmer ist zur Zeit nicht empfangsbereit. Die Leitstation geht direkt in die Abschlußphase über.
   - Keine, fehlerhafte oder unzulässige Rückmeldung: Nach Ablauf der Antwortüberwachungszeit geht die Leitstation direkt in die Abschlußphase über.

3. Gruppenempfangsschnellaufruf: Zur direkten Adressierung sämtlicher Teilnehmer am Bus ist die Geräteadresse 0 reserviert. Die Leitstation sendet dabei ein spezielles Telegramm, welches von keinem Teilnehmer bestätigt wird: <RADR> STX Text ETX BCC EOT.

2.3.3. Datenübermittlungsphase

Jede Datenübermittlungsphase wird entweder mit dem Zeichen STX oder dem Zeichen SOH mit folgenden Bedeutungen eingeleitet:

- STX: Der folgende Datenblock ist für den adressierten Teilnehmer bzw. die Leitstation bestimmt.
- SOH: Der folgende Datenblock wird von einem Teilnehmer an die Leitstation gesendet und soll von dieser an einen anderen Teilnehmer weitergeleitet werden (Querverkehr). Dabei wird die Geräteadresse des gewünschten Teilnehmers direkt nach SOH angegeben, die komplette Zeichenfolge lautet SOH <EADR> STX.

Die maximale Länge eines Datenblocks darf 128 Zeichen betragen. Längere Datensätze müssen entsprechend in einzelne Blöcke zu je maximal 128 Zeichen zerlegt und einzeln übertragen werden. Zur Markierung einer Datenübertragung mit mehreren Blöcken wird dabei als letztes Zeichen jeweils ETB (statt ETX) übergeben. Das Ende der Blockübertragung wird im letzen Block mit ETX (statt ETB) gekennzeichnet.
Die ordnungsgemäße Übertragung eines Datenblocks wird mit der positiven Rückmeldung DLE31 quittiert. Jede Datenübertragung wird mit EOT beendet.
Wird nach Übertragung eines Datenblocks statt der positiven Rückmeldung DLE31 eine negative Rückmeldung NAK empfangen, dann bedeutet dies, daß der Empfänger einen Fehler innerhalb des Datenblocks gefunden hat und nun auf die Wiederholung des Datenblocks wartet. Kann trotz mehrfacher Wiederholung keine positive Rückmeldung empfangen werden, sollte die Übertragung mit EOT beendet werden, damit der Bus wieder für andere Teilnehmer frei wird. Die Anzahl der zulässigen Wiederholungen kann anwendungsspezifisch festgelegt werden, die Norm empfiehlt maximal 2 zulässige Wiederholungen.
Wird nach Übertragung eines Datenblocks innerhalb der Antwortüberwachungszeit T_A keine oder eine fehlerhafte Rückmeldung empfangen, so wird die Rückmeldung erneut mit ENQ angefordert. Diese erneute Anforderung wird maximal zweimal wiederholt. Bei negativem Ergebnis wird die Datenübertragung mit EOT beendet.

In der Datenübermittlungsphase fehlerhaft übertragene Steuerzeichen führen bei Erkennen durch die Leitstation zu einer Beendung des Datenverkehrs mit EOT.

2.3.4. Abschlußphase

Durch die Abschlußphase wird Datenübertragung zwischen Leitstation und Teilnehmer beendet. Die Datenquelle sendet dazu EOT und geht in den Grundzustand über. Die Datensenke geht nach dem Empfang von EOT ebenfalls in den Grundzustand über. Wird der Empfang der Meldung EOT gestört, so führt der Ablauf der Betriebsüberwachungszeit T_C ebenfalls zum Grundzustand der Leitstation.
Die Betriebsüberwachungszeit T_C ist ebenso wie die Antwortüberwachungszeit T_A zur Erkennung von Störungen des Datenverkehrs vorgesehen. Beide Zeiten sind von der eingestellten Datenübertragungsrate direkt abhängig:

- Antwortüberwachungszeit T_A: Sie wird von der jeweiligen Datenquelle mit der Aussendung des Zeichens ENQ oder BCC gestartet. Wird innerhalb dieser Zeit keine gültige Rückantwort von der Datensenke gesendet, so kann die Datenquelle durch senden der Zustandsabfrage ENQ die Anforderung der Rückantwort maximal zweimal wiederholen. Die Zeit T_A berücksichtigt vor allem die beim Empfänger einer Nachricht benötigte Reaktionszeit auf eine Datenübertragung und die notwendige Zeit zur Berechnung und Überprüfung des Blockprüfzeichens BCC. Sie beträgt:

$$T_A = 20 * t_{ZEICHEN} = 20 * \frac{10}{\text{Baudrate}} = \frac{200}{\text{Baudrate}}$$

- Betriebsüberwachungszeit T_C: Sie beginnt nach der Übergabe des Senderechts an den adressierten Teilnehmer. Innerhalb dieser Zeit muß die gesamte Nachricht des Teilnehmers übertragen worden sein. nach Ablauf von T_C nimmt die Leitstation das Senderecht durch Aussenden von EOT zurück. Die Betriebsüberwachungszeit beträgt:

$$T_C = 4 * t_{BLOCK} = 4 * 128 * t_{ZEICHEN} = \frac{4 * 128 * 10}{\text{Baudrate}} = \frac{5120}{\text{Baudrate}}$$

Innerhalb eines Datenblocks darf der Abstand zwischen zwei aufeinanderfolgenden Zeichen maximal $0{,}25*T_A$ betragen.

2.4. Softwareprotokoll

2.4.1. Teilnehmer

Die Teilnehmer am DIN-Meßbus verhalten sich im Grundzustand passiv. Sie sind jederzeit empfangsbereit, hören jede Aufforderung der Leitstation mit und entscheiden nach Adressenvergleich. Ob eine Nachricht für sie bestimmt ist. Ist dies der Fall werden entsprechende Reaktionen eingeleitet, d.h. der Teilnehmer aktiviert sein Sende- oder Empfangsprogramm. Vorhandene Daten können nur nach Erteilung des Senderechts durch die Leitstation weitergeleitet werden. Es ergeben sich also für jeden Teilnehmer zwei grundlegende Programmteile, das Sende- und das Empfangsprogramm.

2.4.2. Empfangsprogramm für einen Teilnehmer

Das Empfangsprogramm eines Teilnehmers am DIN-Meßbus kann nur durch zwei Aufrufe der Leitstation aktiviert werden:

- durch einen direkten Empfangsaufruf bei Übertragung von <EADR>ENQ,
- durch einen Gruppenempfangsschnellaufruf bei Übertragung von <RADR>.

Direkter Empfangsaufruf: Nach Empfang der Aufforderung <EADR>ENQ überprüft der Teilnehmer seine Empfangsbereitschaft. Ist er nicht empfangsbereit (z.B. weil die vorhergehenden Daten noch verarbeitet sind) antwortet er mit einer negativen Quittung <EADR>NAK. Liegt Empfangsbereitschaft vor, so teilt er dies der Leitstation durch Übertragung einer positiven Quittung <EADR>DLE30 mit und aktiviert seine Empfangswarteschleife. Wird innerhalb dieser Zeit ein Steuerzeichen EOT empfangen, so bricht der Teilnehmer sein Empfangsprogramm ab und geht zurück in den passiven Grundzustand.

Gruppenempfangsschnellaufruf: Der Beginn eines Gruppenempfangsschnellaufrufes wird von der Leitstation durch Übertragung der dafür reservierten Teilnehmeradresse 0 eingeleitet. Dieser Aufruf gilt also allen Busteilnehmern gleichermaßen. Eine Überprüfung der Empfangsbereitschaft findet hier nicht statt, die Teilnehmer aktivieren sofort ihre Empfangswarteschleife. Auch hier kann EOT diesen Vorgang beenden und zum passiven Grundzustand der Teilnehmer führen.

Nach erfolgreicher Aufforderung durch die Leitstation befindet sich der Teilnehmer in der Empfangswarteschleife und wertet hier empfangene Steuerzeichen aus, bis die Übertragung eines Datenblocks beginnt. Folgende Aktivitäten auf Steuerzeichen sind in der Empfangswarteschleife möglich:

EOT: Abbruch aller Aktivitäten und Rückkehr zum Grundzustand;
ENQ: der Leitstation wird eine negative Quittung NAK gesendet und die Empfangswarteschleife wird erneut aktiviert; Dieser Zustand kann eintreten, wenn das Startzeichen für den Datenblock STX nicht korrekt empfangen werden konnte und der Teilnehmer somit in seiner Empfangswarteschleife festhängt. Die Leitstation erkennt dies, wenn die Quittung für den übetragenen Datenblock ausbleibt und erzeugt eine erneute Aufforderung ENQ. Auf diese reagiert der Teilnehmer mit einer negativen Quittung NAK, was die Leitstation zur erneuten Übertragung des Datenblocks, beginnend mit STX, veranlaßt.
STX: kennzeichnet den Beginn eines Datenblocks, der Teilnehmer geht zum Empfang der Daten über;

Der Datenblock wird nun zeichenweise empfangen und in den internen Puffer des Teilnehmers übernommen. Dabei werden folgende Steuerzeichen ausgewertet:

EOT: Abbruch aller Aktivitäten und Rückkehr zum Grundzustand;
ENQ: der Leitstation wird eine negative Quittung NAK gesendet und die Empfangswarteschleife wird erneut aktiviert; Dieser Zustand kann eintreten, wenn die Steuereichen für das Datenblockende, ETX oder ETB, nicht korrekt empfangen werden konnte und der Teilnehmer somit in seiner Empfangsschleife festhängt. Die Leitstation erkennt dies, wenn die Quittung für den übertragenen Datenblock ausbleibt und erzeugt eine erneute Aufforderung ENQ. Auf diese reagiert der Teilnehmer mit einer negativen Quittung NAK, was die Leitstation zur erneuten Übertragung des Datenblocks, beginnend mit STX, veranlaßt.
ETB: markiert das Ende eines Datenblocks und weist darauf hin, daß die Nachricht aus mehreren Datenblöcken besteht; Ist eine Nachricht länger als 128 Zeichen, so wird sie in einzelne Datenblöcke zerlegt, die nacheinander durch mehrere Empfangsaufrufe übertragen werden. Der letzte Datenblock der Nachricht erhält als Endekennzeichen ein ETX. Nach Empfang von ETB aktiviert der Teilnehmer eine Warteschleife zum Empfang des Blockprüfzeichens BCC.
ETX: markiert das Ende eines Datenblocks; Der Teilnehmer aktiviert eine Warteschleife zum Empfang des Blockprüfzeichens BCC.

Wird nun statt des Blockprüfzeichens BCC ein EOT empfangen so führt dies zum Abbruch der Datenübertragung. Der Teilnehmer geht in den passiven Grundzustand über.
Empfängt der Teilnehmer ein gültiges BCC, werden die empfangenen Daten durch Berechnung eines eigenen BCC und Vergleich mit dem empfangenen BCC überprüft. Wenn beide übereinstimmen und während der Datenübertragung kein Paritätsfehler aufgetreten ist erzeugt der Teilnehmer eine positive Rückmeldung DLE31 und wartet auf die Bestätigung der Leitstation. Antwortet diese mit ENQ so wird die positive Rückmeldung wiederholt, lautet die Antwort EOT so ist die Datenübertragung erfolgreich abgeschlossen und der Teilnehmer geht in den passiven Grundzustand über.
Trat bei der Überprüfung des BCC ein Fehler auf, erzeugt der Teilnehmer eine negative Rückmeldung NAK und geht zurück in die Empfangswarteschleife, damit die Datenübertragung wiederholt werden kann.

2.4.3. Sendeprogramm für einen Teilnehmer

Das Sendeprogramm eines Teilnehmers am DIN-Meßbus kann nur durch Übergabe des Senderechts von der Leitstation an den Teilnehmer aktiviert werden. Die Leitstation sendet dazu <SADR>ENQ. Nach Empfang eines solchen für den Teilnehmer bestimmten Sendeaufrufes überprüft dieser seine Sendebereitschaft. Ist der Teilnehmer nicht sendebereit, z.B. wenn die Daten noch nicht entsprechend aufbereitet sind, reagiert er mit einer negativen Quittung <SADR>NAK und gibt somit das Senderecht zurück an die Leitstation. Bei Sendebereitschaft wird vom Teilnehmer eine positive Quittung <SADR>DLE30 erzeugt und zur Datenübertragung übergegangen. Ein von der Leitstation generiertes EOT beendet alle Aktivitäten des Teilnehmers, welcher daraufhin in den passiven Grundzustand übergeht.

Wird die Datenübertragung gestartet, muß zunächst der Antwortzähler neu initialisiert werden. Dieser dient zur Zählung der Wiederholungen bei fehlerhafter Datenübertragung. Die Anzahl der Wiederholungen wird

begrenzt, um die Busbelastung durch einen defekten Teilnehmer möglichst klein zu halten. Der zu übertragende Datenblock muß vom Teilnehmer vor der Übertragung entsprechend aufbereitet, also mit Steuerzeichen STX, SOH, ETX, ETB und Blockprüfzeichen BCC versehen, in einem Pufferbereich des Teilnehmers abgelegt werden, da das Sendeprogramm nur für die Übertragung, nicht aber für die Richtigkeit der Daten verantwortlich ist. Ein von der Leitstation erzeugtes EOT während der Datenübertragung beendet diese und führt zum passiven Grundzustand des Teilnehmers. Nach Übertragung des Datenblocks wird vom Teilnehmer dessen Afforderungszähler neu initialisiert und die Antwortüberwachungszeit T_A gestartet, um die Reaktionen der Leitstation auf die Datenübertragung einer entsprechenden Kontrolle unterziehen zu können. Ist nach Ablauf von T_A keine Quittung von der Leitstation vorhanden, wird der Aufforderungszähler inkrementiert und T_A erneut gestartet. Nach maximal drei Versuchen wird dieser Vorgang durch Senden von EOT abgebrochen und die Datenübertragung ist ungültig. Bei einer rechtzeitigen Quittierung der Datenübertragung durch die Leitstation sind zwei Fälle möglich:

- Positive Quittierung mit DLE31; Der Datenblock wurde von der Leitstation richtig empfangen, die Überprüfung des BCC war erfolgreich. Der Teilnehmer erzeugt daraufhin EOT um die Datenübertragung zu beenden. Das Senderecht geht damit wieder an die Leitstation über, der Teilnehmer geht wieder in den passiven Grundzustand.
- Negative Quittierung mit NAK; Der Teilnehmer inkrementiert seinen Antwortzähler und wiederholt die Datenübertragung falls die maximale Anzahl der Wiederholungen noch nicht erreicht ist. Anderenfalls bricht der Teilnehmer die Datenübertragung durch Senden von EOT ab und geht in den passiven Grundzustand über.

Die Leitstation kann während der Antwortüberwachungszeit T_A durch Senden von EOT die Datenübertragung jederzeit abbrechen.

2.4.4. Leitstation

Die Leitstation fungiert im Bussystem als Busmaster und organisiert und überwacht somit den gesamten Datenverkehr auf dem Bus. Dazu sind wie bei einer Teilnehmerstation zwei grundlegende Programmteile, ein Sende- und ein Empfangsprogramm, notwendig. Alle anderen Aufgaben der Leitstation, wie Visualisierung oder Kommunikation mit übergeordneten Systemen, sind stark anwendungsabhängig und sollen hier nicht näher betrachtet werden. Auch der Querverkehr zwischen Teilnehmerstation soll hier keine weitere Berücksichtigung finden, da er eher selten auftritt und nur in wenigen Anwendungsfällen sinnvoll ist. Die Leitstation sollte ständig über Aktivitäten und Zustand der Busteilnehmer informiert sein, so daß auch entsprechende Pollingroutinen zur Statusabfrage sinnvoll und notwendig scheinen. Diese sind aber auch anwendungsabhängig und nur bei entsprechend durch die Teilnehmer zur Verfügung gestellten Funktionalitäten realisierbar. Somit wird auch dieser Punkt in unserer Beschreibung zum DIN-Meßbus vernachlässigt.

2.4.5. Empfangsprogramm für eine Leitstation

Die Leitstation muß in der Lage sein die für eine Anwendung notwendigen Daten von einem Teilnehmer zu empfangen. Dazu muß zunächst das Senderecht durch einen entsprechenden Sendeaufruf <SADR>ENQ an eine Teilnehmerstation übertragen werden. Nach Senden von <SADR>ENQ startet die Leitstation die Antwortüberwachungszeit T_A und eine Warteschleife zum Empfang der Quittierung durch den angesprochenen Teilnehmer. Innerhalb dieser Schleife werden folgende Reaktionen der Teilnehmerstation ausgewertet:

- Keine Quittung innerhalb von T_A; Der angesprochene Teilnehmer konnte nicht oder nicht richtig auf die Sendeaufforderung der Leitstation reagieren. Die Leitstation bricht den Vorgang durch Senden von EOT ab und signalisiert, daß der Teilnehmer nicht reagiert. Dies sollte entsprechend von übergeordneten Programmteilen ausgewertet werden.
- Negative Quittung <SADR>NAK; Es stehen keine Daten zur Verfügung, die Leitstation bricht den Datenverkehr mit EOT ab und signalisiert dies entsprechend übergeordneten Programmteilen.
- Positive Quittung <SADR>DLE30; Die angesprochene Teilnehmerstation hat ihre Daten entsprechend aufbereitet und beginnt sofort mit der Übertragung des Datenblocks.

Nach einer positiven Quittung startet die Leitstation sofort die Betriebsüberwachungszeit T_C, innerhalb welcher der gesamte Datenblock übertragen werden muß, und die Empfangswarteschleife, die auf den Start der Datenübertragung wartet. Bei Ablauf von T_C ohne gültige Datenübertragung, z.B. wenn STX nicht erkannt werden konnte und die Leitstation somit in der Empfangsschleife festhängt, wird der Vorgang von der Leitstation durch Senden von EOT beendet.

Das Empfangsprogramm kann in der Empfangsschleife folgende Steuerzeichen auswerten:

EOT: Die Datenübertragung wird durch den Teilnehmer abgebrochen, die Leitstation erzeugt ebenfalls ein EOT, um das Ende der Datenübertragung zu bestätigen.
ENQ: Die positive Quittung der Leitstation an den Teilnehmer ist ausgeblieben und der Teilnehmer veranlaßt die Leitstation nun zu einer negativen Quittung und zum Neustart der Empfangsschleife, damit die Übertragung des Datenblocks wiederholt werden kann. Dies kann eintreten, wenn STX nicht korrekt übertragen wurde, die Leitstation also noch in der Empfangsschleife festhängt, T_c noch nicht abgelaufen ist, der Teilnehmer aber bereits seinen gesamten Datenblock übertragen hat und nun auf eine Quittung der Leitstation wartet. Vor allem bei der Übertragung kurzer Datenblöcke kann dies der Fall sein, da die Antwortüberwachungszeit T_A wesentlich kleiner als die Betriebsüberwachungszeit T_c ist.
STX: kennzeichnet den Beginn des Datenblocks, die Datenübertragung gilt damit als gestartet.
SOH: kennzeichnet den Beginn eines Datenblockes für den Querverkehr. Nach SOH folgt immer die Zieladresse und STX für den Beginn des eigentlichen Datenblockes.

Nach STX beginnt die Übertragung des Datenblocks. Die Leitstation aktiviert dazu eine entsprechende Empfangsschleife, die für den Empfang der Daten verantwortlich ist und folgende Steuerzeichen auswerten kann:

EOT: Die Datenübertragung wird durch den Teilnehmer abgebrochen, die Leitstation erzeugt ebenfalls ein EOT, um das Ende der Datenübertragung zu bestätigen.
ENQ: Die positive Quittung der Leitstation an den Teilnehmer ist ausgeblieben und der Teilnehmer veranlaßt die Leitstation nun zu einer negativen Quittung und zum Neustart der Empfangsschleife, damit die Übertragung des Datenblocks wiederholt werden kann. Dies kann eintreten, wenn ETX oder ETB nicht korrekt übertragen wurde, die Leitstation also noch in der Empfangsschleife festhängt, T_c noch nicht abgelaufen ist, der Teilnehmer aber bereits seinen gesamten Datenblock übertragen hat und nun auf eine Quittung der Leitstation wartet. Vor allem bei der Übertragung kurzer Datenblöcke kann dies der Fall sein, da die Antwortüberwachungszeit T_A wesentlich kleiner als die Betriebsüberwachungszeit T_c ist.
ETB: markiert das Ende eines Datenblocks und weist darauf hin, daß die Nachricht aus mehreren Datenblöcken besteht; Ist eine Nachricht länger als 128 Zeichen, so wird sie in einzelne Datenblöcke zerlegt, die nacheinander durch mehrere Sendeaufrufe der Leitstation übertragen werden müssen. Der letzte Datenblock der Nachricht erhält als Endekennzeichen ein ETX. Nach Empfang von ETB aktiviert die Leitstation eine Warteschleife zum Empfang des Blockprüfzeichens BCC.
ETX: markiert das Ende eines Datenblocks; Die Leitstation aktiviert eine Warteschleife zum Empfang des Blockprüfzeichens BCC.

Die Warteschleife zum Empfang des Blockprüfzeichens kann durch Empfang eines EOT abgebrochen werden, die Leitstation kehrt dann in den Grundzustand zurück. Wird ein gültiges BCC empfangen, beginnt die Überprüfung der Daten. Dazu berechnet die Leitstation ein eigenes BCC und vergleicht dies mit dem übertragenen. Sind beide gleich und trat während der Übertragung kein Paritätsfehler auf, quittiert die Leitstation die Übertragung der Daten mit einer positiven Rückmeldung, welche vom Teilnehmer quittiert werden muß. Dazu aktiviert die Leitstation eine Warteschleife auf die Quittierung des Teilnehmers. Empfängt die Leitstation nun ein ENQ vom Teilnehmer, so wiederholt sie die positive Rückmeldung und wartet erneut auf die Quittierung durch den Teilnehmer. Wird durch die Leitstation ein EOT empfangen, gilt die Datenübertragung als erfolgreich abgeschlossen und die Daten können übergeordneten Programmteilen zur Verfügung gestellt werden. Sollte die Überprüfung des BCC nach Empfang des Datenblockes negativ ausfallen, wird von der Leitstation eine negative Quittung NAK erzeugt und erneut die Empfangswarteschleife aktiviert, damit die Datenübertragung wiederholt werden kann.

2.4.6. Sendeprogramm für eine Leitstation

Sollen Daten von der Leitstation an einen Teilnehmer übertragen werden, so muß die Leitstation zunächst die Empfangsbereitschaft des entsprechenden Teilnehmers überprüfen. Dazu generiert die Leitstation eine entsprechende Aufforderung <EADR>ENQ und startet die Antwortüberwachungszeit T_A. Die darauf folgende Warteschleife wertet folgende Reaktionen des angesprochenen Teilnehmers aus:

- Negative Rückmeldung <EADR>NAK; Der angesprochene Teilnehmer ist nicht empfangsbereit und antwortet mit einer negativen Quittung <EADR>NAK. Die Leitstation kann den Aufruf auf einen späteren Zeitpunkt verschieben, oder auf einen Fehler des Teilnehmers schließen und dies übergeordneten Programmteilen mitteilen.
- Positive Rückmeldung <EADR>DLE30: Der angesprochene Teilnehmer ist empfangsbereit und erwartet nun den zu übertragenden Datenblock.
- Ablauf der Antwortüberwachungszeit T_A; Bei Ablauf von T_A ohne gültige Rückmeldung wird der Vorgang von der Leitstation durch Senden von EOT abgebrochen.

Nach erfolgter positiver Rückmeldung und der Neuinitialisierung des Antwortzählers der Leitstation beginnt die eigentliche Übertragung des Datenblocks. Der Antwortzähler dient dabei im Fehlerfall der Begrenzung der maximalen Anzahl der Wiederholungen. Der zu übertragende Datenblock muß von der Leitstation vor der Übertragung entsprechend aufbereitet, also mit Steuerzeichen STX, SOH, ETX, ETB und Blockprüfzeichen BCC versehen, in einem Pufferbereich der Leitstation abgelegt werden, da das Sendeprogramm nur für die Übertragung, nicht aber für die Richtigkeit der Daten verantwortlich ist.
Nach der Übertragung des Datenblocks startet die Leitstation sofort die Antwortüberwachungszeit T_A und initialisiert den Aufforderungszähler neu. Ist nach Ablauf von T_A keine gültige Rückantwort empfangen worden fordert die Leitstation die Rückantwort nochmals durch Senden von ENQ an, inkrementiert den Aufforderungszähler und startet die Antwortüberwachungszeit T_A erneut. Nach maximal drei Aufforderungen muß eine gültige Rückantwort des Teilnehmers vorliegen, ansonsten wird die Übertragung durch die Leitstation beendet, indem ein EOT erzeugt wird. Wurde hingegen eine gültige Rückantwort empfangen, so kann diese zwei Formen annehmen:

- Positive Rückmeldung DLE31; Die Übertragung der Daten und deren Überprüfung mittels Parität und Blockprüfzeichen war erfolgreich. Das Sendeprogramm der Leitstation beendet daraufhin die Übertragung mit EOT.
- Negative Rückmeldung NAK; Bei der Übertragung des Datenblocks ist ein Fehler aufgetreten. Die Leitstation inkrementiert daraufhin den Antwortzähler und startet die Übertragung des Datenblocks neu, falls die maximale Anzahl der Wiederholungen noch nicht erreicht ist. Wird die maximale Anzahl der Wiederholungen überschritten bricht die Leitstation die Datenübertragung ab und signalisiert dies übergeordneten Programmteilen.

Im Gegensatz zum normalen Sendeprogramm wird beim Gruppenempfangsschnellaufruf keine Überprüfung der Reaktionen der Teilnehmer durchgeführt. Nach <RADR> folgt hier direkt die Übertragung des Datenblocks. Die Übertragung wird auch hier von der Leitstation durch EOT beendet.

3. Programmierung des *isel* OEM Schrittmotormoduls PICMIC

3.1. Funktionalitäten und Anwendungsgebiete von PICMIC

3.1.1. Allgemeines

Das ***isel***-Schrittmotormodul-PICMIC ist in erster Linie als Automatisierungsbaustein für den OEM-Anwender gedacht, welcher, in mit DIN-Meßbus vernetzten Systemen, Aktoren für einfache Bewegungsabläufe benötigt. Auf der Basis des DIN-Meßbus stellt das Modul entsprechende Funktionalitäten zu einem sehr günstigen Preis-Leistungs-Verhältnis bereit. Grundlegend sind zwei Betriebsmodes, Positionierbetrieb und Drehzahlbetrieb, in zwei Ansteuerungsvarianten, DIN-Meßbus-Ansteuerung und Stand-Alone-Betrieb, implementiert. Neben der Ansteuerung des Motors sind auf dem Modul 4 digitale Eingänge und 1 digitaler Ausgang, sowie ein serielles E^2PROM zur Abspeicherung von Funktionsabläufen vorhanden. Für die Ansteuerung über den DIN-Meßbus wird eine galvanisch getrennte vollduplex Schnittstelle nach EIA RS485 zur Verfügung gestellt. Weitere Merkmale des Moduls sind geringer Platzbedarf, weiter Versorgungsspannungsbereich und einstellbare Motorströme. Daraus resultiert ein breites Anwendungsspektrum in verschiedensten Applikationen. Neben Meßtechnik, Prozeß- und Laborautomation, Schul- und Ausbildungsbereich sind auch Anwendungen im Heim- und Hobbybereich denkbar.

3.1.2. Betriebsmodes

Um ein möglichst breites Anwendungsspektrum abdecken zu können sind zwei Betriebsmodes vorgesehen:

Positionierbetrieb: Der Motor kann über absolute und relative Positionierbefehle beliebige Positionen anfahren. Die Positionierung erfolgt mit der entsprechend eingestellten Geschwindigkeit. Neben den Befehlen für absolute und relative Bewegungen sind auch die Befehle Referenzfahrt und Bewege bis Impuls implementiert. Die Befehle können sowohl über den DIN-Meßbus an die Steuerung übertragen und direkt ausgeführt, als auch im E^2PROM für den Stand-Alone-Betrieb abgelegt werden.

Drehzahlbetrieb: Beim Drehzahlbetrieb versucht die Steuerung die jeweils vorgegebene Geschwindigkeit zu erreichen. Die Drehrichtung wird dabei aus dem Vorzeichen der Geschwindigkeitsangabe abgeleitet. Auch dieser Betriebsmode kann über den DIN-Meßbus, oder Stand-Alone realisiert werden.

3.1.3. Ansteuerungsvarianten

Das Modul kann sowohl direkt am DIN-Meßbus, als auch Stand-Alone betrieben werden. Folgende Eigenschaften ergeben sich damit:

DIN-Meßbus: Das Modul wird als Slave am Bus betrieben. Alle Befehle werden dabei über den Bus übertragen und direkt abgearbeitet. Die Überprüfung des Zustands der Steuerung ist ständig durch entsprechende Statusabfragen möglich. Durch die softwaremäßig einstellbare Slaveadresse können bis zu 31 Module am Bus betrieben werden. Der relativ große Befehlsvorrat der Steuerung erlaubt hier eine Vielzahl von Anwendungen.

Stand-Alone: Im seriellen E^2PROM des Moduls können bis zu 64 Befehle abgelegt werden, die als eigenständiges Programm abgearbeitet werden können. Damit ist das Modul für zyklische Funktionsabläufe in Stand-Alone-Betrieb bestens geeignet. Die Abspeicherung der Befehle erfolgt im Busmode. Während der Abarbeitung eines Programms bleiben die Zugriffsmöglichkeiten über den DIN-Meßbus erhalten. Somit ist auch eine aktive Kontrolle der Programmabarbeitung möglich.

3.2. Realisierung der Programmierschnittstelle

3.2.1. Grundlagen

Zur Realisierung der Programmierschnittstellle wurde auf dem ***isel***-PICMIC-Modul ein entsprechendes Sende- und Empfangsprogramm für den DIN-Meßbus implementiert. Die serielle Datenübertragung wird mit einer fest eingestellten Baudrate von 9600 Baud, dies ist sinnvoll, da alle DIN-Meßbus Teilnehmer diese Baurate beherrschen müssen, betrieben. Die Befehle werden als Zeichenketten in das Protokoll des DIN-Meßbus als Nachricht an einen Teilnehmer eingebettet. Die vom Modul erzeugten Rückmeldungen entsprechen ebenfalls diesem Protokoll. Da auf das DIN-Meßbus-Protokoll und den Aufbau der notwendigen Sende- und Empfangsprogramme bereits ausführlich eingegangen wurde (siehe 2. DIN-Meßbus nach DIN 66348 Teil 2) beschränken wir uns nun auf die Beschreibung der eigentlichen Befehle und deren Funktionalitäten.
Die Befehle beginnen immer mit einem Zeichen ‘p’ als Kennung für das ***isel***-PICMIC-Modul, gefolgt von einem Großbuchstaben, der als Befehlscode dient. Alle nachfolgenden Zeichen beinhalten die für den Befehl notwendigen Parameter. Alle Zahlenwerte werden dabei als Hexwerte in ASCII-Notation entsprechend ihres Wertevorrates dargestellt.
Die Rückmeldungen von der Steuerung sind ähnlich aufgebaut. Auch hier wird mit einem Zeichen ‘p’ als Kennung für das ***isel***-PICMIC-Modul begonnen, anschließend folgt ein ASCII-Zeichen als Fehlercode und ein Großbuchstabe mit dem entsprechenden Befehlscode. Alle nachfolgenden Zeichen beinhalten die Information in Abhängigkeit vom Befehlscode als Hexwerte in ASCII-Notation oder als Zeichenkette.

3.2.2. Befehlsübersicht für direkt ausführbare Befehle

Befehlscode	Kurzbeschreibung
A	Vorteiler für Beschleunigung einstellen
B	Absolute Bewegung
C	Sollstrom und Stromabsenkung einstellen
D	Defaultwerte einstellen
E	Befehl im E^2PROM speichern
F	Sollgeschwindigkeit einstellen
G	Bewegung/Programmablauf starten
H	Bewegung/Programmablauf anhalten
I	Eingänge lesen
J	Ausgänge rücklesen
K	Programm im E^2PROM löschen
L	Startbedingungen festlegen
M	Betriebsmodus einstellen
N	Absolutposition auf Null setzen
O	Ausgänge setzen
P	Position abfragen
Q	Softwarereset
R	Wert aus E^2PROM lesen
S	Statusinformation lesen
T	Stop- und Breakbedingungen festlegen
U	Befehl aus E^2PROM rücklesen
V	Versionsabfrage
W	Wert ins E^2PROM schreiben
X	Relative Bewegung
Y	Bewege bis Impuls
Z	Referenzfahrt

3.2.3. Befehlsübersicht für speicherbare Befehle

Befehlscode HEX	Kurzbeschreibung
00	Programmende
01	Bewege Absolut
02	Bewege Relativ
03	Referenzfahrt
04	Bewege bis Impuls
05	Nullpunkt setzen
06	Sollgeschwindigkeit einstellen
07	Vorteiler für Beschleunigung einstellen
08	Sollstrom und Stromabsenkung einstellen
09	Ausgänge setzen
0A	Eingänge lesen, verzweigen
0B	Schleife, Verzweigung
0C	Verzögerung
0D	Betriebsmodus einstellen
0E	Geschwindigkeit erhöhen (nur im Drehzahlbetrieb)
0F	Geschwindigkeit vermindern (nur im Drehzahlbetrieb)

3.2.4. Fehlercodes der Programmierschnittstelle

Nach einem Sendeaufruf an den jeweiligen Teilnehmer antwortetet dieser stets mit einer entsprechenden Nachricht, die entweder den aktuellen Status oder die zuletzt angeforderten Informationen (z.B. Portbelegung oder Position). In diese Nachrichten wird immer ein Fehlercode in Form eines ASCII-Zeichens eingefügt (siehe Beschreibungen der einzelnen Rückmeldungen der Befehle), der den letzten bekannten Fehlerstatus signalisiert. Folgende Fehlercodes sind hierbei möglich:

Fehlercode	Fehler	Beschreibung
'0'	kein Fehler	der letzte Befehl wurde richtig abgearbeitet
'1'	unbekannter Befehl	die Steuerung hat einen unbekannten Befehlscode empfangen, der Befehl wurde ignoriert
'2'	Syntaxfehler	ein Befehl wurde mit falscher Syntax verwendet, der Befehl konnte nicht ausgeführt werden
'3'	Parameterfehler	ein notwendiger Parameter wurde nicht oder mit falschem Wertebereich übergeben, der Befehl wurde ignoriert
'4'	In-Bewegung-Fehler	ein Befehl konnte nicht ausgeführt werden, weil der Motor in Bewegung ist
'5'	Typkennung-Fehler	es wurde ein Befehl mit falscher Typkennung übergeben, der Befehl wurde nicht ausgeführt
'6'	Programmende	ein gespeichertes Programm wurde bis zum Ende abgearbeitet
'7'	Stopfehler	eine Bewegung oder ein Programm wurde angehalten
'8'	Breakfehler	ein Breakbefehl wurde ausgeführt, die Steuerung sollte neu initialisiert werden

3.2.5. Speicheraufteilung im E^2PROM, Defaultwerte

Das ***isel***-PICMIC-Modul verfügt über ein serielles E^2PROM, in welchem sowohl Defaultwerte und speicherbare Befehle, als auch Nutzerdaten abgelegt werden können. Dieses E^2PROM hat eine Größe von 4K-Bit. Es ist in 2 Speicherbänke a 2K-Bit eingeteilt und wird vom Prozessor seriell über ein I^2C-Protokoll bedient.
In jeder Speicherbank können also 256 Byte Daten abgelegt werden. Bank0 ist dabei für Defaultwerte und Nutzerdaten vorgesehen, Bank1 dient der Speicherung von Programmabläufen.

Bank0 des E^2PROMS: Adresse `00h – 7fh` Default- und Initialisierungswerte
Adresse `80h – ffh` frei für Nutzerdaten

Nach dem ersten Einschalten des Moduls hat die Bank 0 folgendes Aussehen:

Adresse	00	01	02	03	
Wert	1fh	55h	aah	ffh	Teilnehmeradresse (31 dezimal)
Adresse	04	05	06	07	
Wert	feh	ffh	ffh	ffh	Beschleunigung (feh)
Adresse	08	09	0a	0b	
Wert	00h	00h	ffh	ffh	Strom (Sollstrom und abgesenkter Strom 00h)
Adresse	0c	0d	0e	0f	
Wert	03h	e8h	ffh	ffh	Geschwindigkeit (3e8h - 1kHz Halbschritt)
Adresse	10	11	12	13	
Wert	00h	ffh	ffh	ffh	Steuermodus (00h – Punkt zu Punkt)
Adresse	14	15	16	17	
Wert	00h	00h	ffh	ffh	Startbedingungen (00h keine)
Adresse	18	19	1a	1b	
Wert	00h	00h	00h	00h	Stop- und Breakbedingungen (00h keine)
Adresse	1c	.....		7f	
Wert	reserviert				steuerungsinterne Werte, Erweiterungen
Adresse	80	.....		ff	
Wert	Nutzerdaten				für Nutzer frei verfügbar

Auf die Bank0 kann über folgende Befehle:

- Defaultwerte einstellen (siehe 3.3.4. Defaultwerte einstellen)
- Startbedingungen festlegen (siehe 3.3.12. Startbedingungen festlegen)
- Wert aus E^2PROM lesen (siehe 3.3.18. Wert aus E^2PROM lesen)
- Stop- und Breakbedingungen festlegen (siehe 3.3.20. Stop- und Breakbedingungen festlegen)
- Wert ins E^2PROM schreiben (siehe 3.3.23. Wert nach E^2PROM schreiben)

zugegriffen werden.

Bank1 des E^2PROMS: Adresse 00h – ffh Programmablauf, max. 64 Befehle

Adresse	00	01	02	03	
Wert	Befehl 1				Befehl 1, 4 Byte
Adresse	04	05	06	07	
Wert	Befehl 2				Befehl 2, 4 Byte
Adresse	08	09	0a	0b	
Wert	Befehl 3				Befehl 3, 4 Byte
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
.	.				.
Adresse	fc	fd	fe	ff	
Wert	Befehl 64				Befehl 64, 4 Byte

Auf die Bank1 kann über folgende Befehle:

- Befehl im E^2PROM speichern (siehe 3.3.5. Befehl im E^2PROM speichern)
- Programm im E^2PROM löschen (siehe 3.3.11. Programm im E^2PROM löschen)
- Befehl aus E^2PROM rücklesen (siehe 3.3.21. Befehl aus E^2PROM rücklesen)

zugegriffen werden. Die Adressierung erfolgt hierbei befehlsweise, also von Befehl 0 bis Befehl 64.

3.3. Direkt ausführbare Befehle

3.3.1. Vorteiler für Beschleunigung einstellen

Die Beschleunigung wird vom Modul über einen Timerinterrupt mit einer Grundfrequenz von 900Hz, einem Offset von 100Hz und einer Start-Stop-Frequenz von 300Hz erzeugt. Die Änderung der Beschleunigung wird über einen 8-Bit-Software-Upcounter ermöglicht. Mit dieser Funktion kann der Relaodwert für diesen Upcounter festgelegt werden.

Befehlsaufbau:	"pAxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	A	Befehlscode
	xx	Hexwert für Vorteiler, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		FF = maximale Beschleunigung
		00 = minimale Beschleunigung
Beispiel:	"pAfe"	Der Reloadwert für die Beschleunigung wird auf fe hex eingestellt d.h. die Geschwindigkeit wird in einem Zeitraster von 2.2ms (450Hz) verändert. Bei einem Offset von 100Hz ergibt sich somit eine Beschleunigung von 45 Hz/ms.

3.3.2. Absolute Bewegung

Die Ausführung einer absoluten Positionierung ist eine Grundfunktionalität jeder Einachssteuerung. Diese Funktion ermöglicht die Positionierung auf eine beliebige absolute Position mit der aktuell eingestellten Geschwindigkeit. Die Ausführung der Bewegung kann über die Statusabfrage überwacht und über Start-, Stop- und Breakfunktionen beeinflußt werden. Auch die aktuelle Position kann während der Bewegung abgefragt werden, da der Zugriff über den Bus erhalten bleibt.

Befehlsaufbau:	"pBxxxxxxxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	B	Befehlscode
	xxxxxxxx	Hexwert für Zielposition, 32-Bit (4Byte, 8Hexziffern)
Kommentar:		Die Angabe der Zielposition erfolgt in Halbschritten. Da modulintern alle Positionen in 16-tel Schritten gerechnet und das höchstwertige Bit als Vorzeichen benutzt werden stehen als Wertevorrat nur 28Bit d.h. +/- 268435455 Halbschritte zur Verfügung. Die Überschreitung dieser Werte wird nicht überprüft und führt somit zu falschen Ergebnissen.
Beispiel:	"pB00001000"	Es wird eine absolute Positionierung auf die Position 1000hex (4096 dezimal) mit der aktuell eingestellten Geschwindigkeit in Auftrag gegeben.

3.3.3. Sollstrom und Stromabsenkung einstellen

Bei Schrittmotorsteuerungen werden i.A. bei Motorstillstand die Phasenströme abgesenkt um eine unnötige Erwärmung der Motoren und Endstufen zu verhindern. Für das ***isel***-PIMIC-Modul können die Phasenströme softwaremäßig eingestellt werden. Der Mikrokontroller der Steuerung erzeugt zur Ansteuerung der Endstufen dazu ein PWM-Signal mit 8-Bit Auflösung. Diese Funktion ermöglicht die Einstellung von zwei Stromsollwerten, getrennt für Betriebsstrom und abgesenkten Strom.

Befehlsaufbau:	"pCxxyy"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	C	Befehlscode
	xx	Hexwert für Sollstrom, 8-Bit (1Byte, 2Hexziffern)
	yy	Hexwert für abgesenkten Strom, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		FF = maximaler Strom
		00 = minimaler Strom
Beispiel:	"pCf27f"	Der Sollstrom wird auf 95%, der abgesenkte Strom auf 50% des Maximalstromes eingestellt. Die Stromabsenkung erfolgt automatisch bei Stillstand des Motors. Soll keine Stromabsenkung erfolgen müssen beide Werte gleich gesetzt werden.

3.3.4. Defaultwerte einstellen

Jede Steuerung benötigt nach dem Einschalten oder Reset Defaultwerte für die Initialisierung. Oft sind diese fest durch das Betriebssystem vorgegeben und können nicht verändert werden. Da das ***isel***-PICMIC-Modul ein serielles E^2PROM besitzt können hier notwendige variable Defaultwerte abgelegt werden. Diese Werte werden dann nach dem Einschalten oder Reset ausgelesen und für die Initialisierung der entsprechenden Variablen genutzt. Für das Modul wichtige Initialisierungswerte sind der Betriebsmode, die Phasenströme, Beschleunigung und Geschwindigkeit und vor allem die Teilnehmeradresse für den DIN-Meßbus. Alle Defaultwerte werden in der Bank0 des E^2PROM abgelegt.

Befehlsaufbau:	`"pDcval"`	
Erläuterung:	`p`	Kennung für ***isel***-PICMIC-Modul
	`D`	Befehlscode
	`c`	Code für den Defaultwert
	`val`	Parameter für den Defaultwert
Kommentar:		Dieser Befehle kann sehr unterschiedliches Aussehen haben, da der Wertebereich der Parameter abhängig von der Art des Defaultwertes ist.

3.3.4.1. Einstellen des Defaultwertes für den Beschleunigungsvorteiler

siehe auch 3.3.1. Vorteiler für Beschleunigung einstellen

Befehlsaufbau:	`"pDAxx"`	
Erläuterung:	`p`	Kennung für ***isel***-PICMIC-Modul
	`D`	Befehlscode
	`A`	Defaultreloadwert für Beschleunigung einstellen
	`xx`	Hexwert für Vorteiler, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		`FF` = maximale Beschleunigung
		`00` = minimale Beschleunigung
Beispiel:	`"pDAfe"`	Der Defaultreloadwert für die Beschleunigung wird auf fe hex eingestellt d.h. die Geschwindigkeit wird in einem Zeitraster von 2.2ms (450Hz) verändert. Bei einem Offset von 100Hz ergibt sich somit eine Beschleunigung von 45 Hz/ms.

3.3.4.2. Einstellen der Defaultwerte für Sollstrom und Stromabsenkung

siehe auch 3.3.3. Sollstrom und Stromabsenkung einstellen

Befehlsaufbau:	`"pDCxxyy"`	
Erläuterung:	`p`	Kennung für ***isel***-PICMIC-Modul
	`D`	Befehlscode
	`C`	Default für Sollstrom und Stromabsenkung einstellen
	`xx`	Hexwert für Sollstrom, 8-Bit (1Byte, 2Hexziffern)
	`yy`	Hexwert für abgesenkten Strom, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		`FF` = maximaler Strom
		`00` = minimaler Strom
Beispiel:	`"pDCf27f"`	Der Sollstrom wird auf 95%, der abgesenkte Strom auf 50% des Maximalstromes eingestellt. Die Stromabsenkung erfolgt automatisch bei Stillstand des Motors. Soll keine Stromabsenkung erfolgen müssen beide Werte gleich gesetzt werden.

3.3.4.3. DIN-Meßbus-Teilnehmeradresse einstellen

Am DIN-Meßbus können bis zu 31 Teilnehmer angeschlossen werden. Die Einstellung einer neuen Teilnehmeradresse für das ***isel***-PICMIC-Modul kann softwaremäßig erfolgen, wenn die alte Teilnehmeradresse bekannt ist. Im Lieferzustand besitzen alle Module die Adresse 31. Sollte dies nicht der Fall sein, so kann diese Einstellung durch setzen eines Jumpers (siehe 4.5. Steckerbelegung) auf der Platine und anschließendes Einschalten erzwungen werden. Der Jumper muß danach unbedingt wieder entfernt werden. Nach diesem Rücksetzen kann das Modul auf Adresse 31 angesprochen werden. Eine Änderung der Teilneh-

meradresse wird erst nach dem Neueinschalten des Moduls gültig. Achten Sie bitte darauf, daß keine Adresse im Bussystem zweimal vergeben werden darf.

Befehlsaufbau:	"pDDxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	D	Befehlscode
	D	DIN-Meßbus-Teilnehmeradresse einstellen
	xx	Hexwert für Adresse, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Adressen im Bereich 00-1F hex (0 - 31 dezimal)
Beispiel:	"pDD01"	Der Teilnehmer erhält die neue Adresse 1. Diese Einstellung wird nach dem Neueinschalten gültig.

3.3.4.4. Einstellen der Defaultgeschwindigkeit

siehe auch 3.3.6. Sollgeschwindigkeit einstellen

Befehlsaufbau:	"pDFxxxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	D	Befehlscode
	F	Default für Geschwindigkeit einstellen
	xxxx	Hexwert für Geschwindigkeit, 16-Bit (2Byte, 4Hexziffern)
Kommentar:		max. 24000Hz Halbschritt (0000 - 5dc0 hex)
Beispiel:	"pDF03e8"	Die Defaultgeschwindigkeit wird auf 1000Hz Halbschritt (03e8hex) eingestellt. Diese Einstellung ist nach dem Einschalten oder Reset bis zur Einstellung einer neuen Geschwindigkeit gültig.

3.3.4.5. Einstellen des Defaultbetriebsmode

siehe auch 3.3.13. Betriebsmodus einstellen

Befehlsaufbau:	"pDMxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	D	Befehlscode
	M	Default für Betriebsmode einstellen
	xx	Hexwert für Betriebsmode, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		00 = Positionierbetrieb
		01 = Drehzahlbetrieb
Beispiel:	"pDM00"	Der Defaultbetriebsmode ist der Positioniermode. Diese Einstellung ist nach dem Einschalten oder Reset bis zur Einstellung eines neuen Betriebsmode gültig.

3.3.5. Befehl im E^2PROM speichern

Das ***isel***-PICMIC-Modul besitzt zur Abspeicherung von Befehlen ein serielles E^2PROM. Dieses E^2PROM ist in zwei Speicherbänke aufgeteilt, in Bank0 werden alle Defaultwerte und in Bank1 alle speicherbaren Befehle abgelegt. Ein Programm für den Stand-Alone-Mode der Steuerung kann bis zu 64 Befehle (siehe 3.4. Speicherbare Befehle) umfassen. Diese Befehle werden im Busmodus übertragen und entsprechend in der Bank1 des E^2PROM abgelegt. Da E^2PROM's nichtflüchtige Speicher sind bleibt ein abgelegtes Programm bis zum Überschreiben oder Löschen gültig.

Befehlsaufbau:	"pEnnxxxxxxxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	xxxxxxxx	Hexwert für Befehlscodierung, 32-Bit (4Byte, 8Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex)
		Befehlscodierung immer 1Byte Befehlscode und 3Byte Parameter
		siehe auch 3.4. Speicherbare Befehle

Beispiel: "pE010603e800" Ein Befehl zum Setzen der Geschwindigkeit wird als Satznummer 1 abgespeichert. Im Programmablauf wird dann an dieser Stelle die Geschwindigkeit auf 1000Hz Halbschritt (03e8hex) eingestellt.

3.3.6. Sollgeschwindigkeit einstellen

Zur Ausführung einer Bewegung wird immer eine Geschwindigkeit benötigt. Wurde keine Geschwindigkeit übergeben, wird eine auszuführende Bewegung mit der zuletzt eingestellten (Default-) Geschwindigkeit abgearbeitet. Im Drehzahlbetrieb wird die neue Geschwindigkeit mit der aktuellen Beschleunigung eingestellt, die Drehrichtung wird dabei vom Vorzeichen der Geschwindigkeit abgeleitet. Im Positionierbetrieb gilt die Geschwindigkeit für die nächste Bewegung, die Drehrichtung ergibt sich hier aus dem Positionierbefehl, so daß die Geschwindigkeitsangabe hier immer positiv sein sollte.

Befehlsaufbau:	"pFxxxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	F	Befehlscode
	xxxx	Hexwert für Geschwindigkeit, 16-Bit (2Byte, 4Hexziffern)
Kommentar:		maximale Geschwindigkeit steuerungsseitig ist 24000Hz Halbschritt, der Wertebereich ist also 0-24000Hz (0000 - 5dc0 hex) im Positioniermode und +/-24000Hz (a240 - 5dc0 hex) im Drehzahlmode. Die tatsächlich erreichbare Geschwindigkeit in einer Applikation kann hier nicht angegeben werden, da dies stark vom Motor selbst, dessen Lastverhalten und der angeschlossenen Mechanik abhängig ist.
Beispiel:	"pF03e8"	Die Geschwindigkeit wird auf 1000Hz Halbschritt (03e8hex) eingestellt. Diese Einstellung ist bis zur Einstellung einer neuen Geschwindigkeit gültig.

3.3.7. Bewegung/Programmablauf starten

Dieser Befehl hat zwei Aufgaben, erstens startet er ein evtl. vorhandenes speicherbares Programm und zweitens startet er eine evtl. angehaltene Bewegung.

Befehlsaufbau:	"pG"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	G	Befehlscode
Kommentar:		Befindet sich die Steuerung im Stopmode (d.h. eine Bewegung wurde angehalten) wird die angehaltene Bewegung weitergeführt, egal ob sie aus einem direkten oder speicherbaren Befehl stammt. Andernfalls wird überprüft, ob ein speicherbares Programm vorhanden und aktiv ist. Ist dies der Fall wird dieses an der aktuellen Stelle fortgesetzt.
Beispiel:	"pG"	Die angehaltene Bewegung oder das speicherbare Programm werden weitergeführt.

3.3.8. Bewegung/Programmablauf anhalten

Auch dieser Befehl hat zwei Aufgaben, erstens eine evtl. gerade aktive Bewegung anzuhalten und zweitens die Programmabarbeitung eines evtl. gerade aktiven speicherbaren Programms anzuhalten.

Befehlsaufbau:	"pH"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	H	Befehlscode
Kommentar:		Es wird immer zuerst überprüft, ob eine Bewegung aktiv ist. Ist dies der Fall, so wird diese angehalten, egal ob sie aus einem direkten oder speicherbaren Befehl stammt. Ist ein speicherbares Programm aktiv, so wird die Ausführung dieses Programms angehalten.
Beispiel:	"pH"	Die aktive Bewegung oder ein aktives speicherbares Programm werden angehalten.

3.3.9. Eingänge lesen

Das isel-PICMIC-Modul verfügt über 4 frei verwendbare Eingänge. Diese können z.B. als Referenzschalter, als Starttaster oder auch zur Synchronistation mit anderen Prozessen genutzt werden. Mit dieser Funktion kann der aktuelle Zustand der Eingänge abgefragt werden.

Befehlsaufbau:	"pI"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	I	Befehlscode
Kommentar:		Nach Empfang dieses Befehls wird der aktuelle Zustand der Eingänge gelesen und als Nachricht für den nächsten Sendeaufruf der Leitstation aufbereitet. Die Rückgabe der Zustandsinformation erfolgt dann beim nächsten Sendeaufruf der Leitstation.
Rückgabe:	"peIxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	e	Fehlercode
	I	Befehlscode
	xx	Hexwert mit aktuellem Zustand, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Die Bitbelegung innerhalb des Hexwertes ist folgende: Bit0: nicht benutzt Bit1: nicht benutzt Bit2: Eingang 1 Bit3: Eingang 2 Bit4: Eingang 3 Bit5: Eingang 4 Bit6: nicht benutzt Bit7: nicht benutzt
Beispiel:	"pI"	Die aktuelle Belegung der Eingänge wird abgefragt. Die Rückantwort könnte folgendermaßen aussehen:
	"p0I3c"	Die Eingänge sind derzeit alle auf High-Pegel.

3.3.10. Ausgänge rücklesen

Oft können bei I/O-Ankopplungen die aktuellen Zustände der Ausgänge nicht rückgelesen werden und müssen so in eigenen internen Puffern gespiegelt werden. Beim ***isel***-PICMIC-Modul übernimmt dies der Mikrokontroller der Steuerung, denn er kennt den aktuellen Zustand der Ausgänge und kann diese Information so zurückliefern.

Befehlsaufbau:	"pJ"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	J	Befehlscode
Kommentar:		Nach Empfang dieses Befehls wird der aktuelle Zustand der Ausgänge als Nachricht für den nächsten Sendeaufruf der Leitstation aufbereitet. Die Rückgabe der Zustandsinformation erfolgt dann beim nächsten Sendeaufruf der Leitstation.
Rückgabe:	"peJxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	e	Fehlercode
	J	Befehlscode
	xx	Hexwert mit aktuellem Zustand, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Die Bitbelegung innerhalb des Hexwertes ist folgende: Bit0: Zustand der Endstufen Ein/Aus Bit1: Zustand Ausgang1 Bit2-7: nicht benutzt
Beispiel:	"pJ"	Die aktuelle Belegung der Ausgänge wird abgefragt. Die Rückantwort könnte folgendermaßen aussehen:
	"p0J03"	Die Endstufen sind eingeschaltet, der Ausgang1 ist auf High-Pegel.

3.3.11. Programm im E^2PROM löschen

Ein im E^2PROM abgelegtes Programm muß gelöscht oder als ungültig gekennzeichnet werden, damit es nicht versehentlich gestartet werden und so zu Fehlverhalten führen kann. Diese Funktion erledigt dies.

Befehlsaufbau:	"pK"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	K	Befehlscode
Kommentar:		Durch diesen Befehl wird einfach die Satznummer 0 des abgelegten Programms mit einem Programmende belegt, damit wird die Ausführung der gespeicherten Befehle verhindert. Ein komplettes Überschreiben des E^2PROM's würde hier zu lange dauern.
Beispiel:	"pK"	Das aktuelle abgespeicherte Programm wird als ungültig erklärt.

3.3.12. Startbedingungen festlegen

Da das isel-PICMIC-Modul auch Stand-Alone betrieben werden kann ist es sinnvoll, Bedingungen zu definieren die zu einem Startbefehl führen. Solche Bedingungen können direkt der Reset oder das Neueinschalten sein, oder aber auch eine definierte Belegung von Eingangsports. Dieser Befehl definiert solche Bedingungen in Form einer Aktivmaske und eines Sollwertes. Ist eine Bedingung aktiviert vergleicht die Steuerung den Istwert mit dem Sollwert und bei Gleichheit wird steuerungsintern ein Startbefehl erzeugt. Sind mehrere Bedingungen aktiviert so werden diese mit Oder verknüpft. Damit wird eine aktive Zugriffsmöglichkeit auf die Befehls- und Programmabarbeitung im Modul geschaffen. Diese Bedingungen werden im E^2PROM abgelegt und stehen der Steuerung somit auch nach Reset oder Neueinschalten zur Verfügung.

Befehlsaufbau:	"pLaabb"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	L	Befehlscode
	aa	Hexwert für Maske, 8-Bit (1Byte, 2Hexziffern)
	bb	Hexwert für Wert, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		als Bedingungen für einen Startbefehl werden für Maske und Wert folgende Bits definiert: Bit0 nicht benutzt Bit1 Reset als Bedingung Bit2 Eingang1 als Bedingung Bit3 Eingang2 als Bedingung Bit4 Eingang3 als Bedingung Bit5 Eingang4 als Bedingung Bit6 nicht benutzt Bit7 nicht benutzt
Beispiel:	"pL0400"	Der Eingang1 wird als aktive Bedingung für einen Startbefehl definiert. Der Startbefehl wird erzeugt, wenn der entsprechende Eingang Low-Pegel besitzt.

3.3.13. Betriebsmodus einstellen

Um ein möglichst breites Anwendungsspektrum abdecken zu können sind zwei Betriebsmodes vorgesehen der Positionierbetrieb, in welchem beliebige relative und absolute Positionen angefahren werden können, und der Drehzahlbetrieb, in welchen die Steuerung immer versucht die vorgegebene Drehzahl zu erreichen. Dieser Befehl erlaubt die Einstellung des gewünschten Betriebsmodes.

Befehlsaufbau:	"pMxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	M	Befehlscode
	xx	Hexwert für Betriebsmode, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		00 = Positionierbetrieb 01 = Drehzahlbetrieb
Beispiel:	"pM00"	Als Betriebsmode wird der Positioniermode eingestellt. Diese Einstellung ist bis zur Einstellung eines neuen Betriebsmode gültig.

3.3.14. Absolutpostion auf Null setzen

Das Rücksetzen der Positionsregister ist eine Grundfunktion jeder Positioniersteuerung. Damit kann der Nullpunkt für eine Positionierung an einen beliebigen Punkt im Bewegungsbereich der Mechanik gesetzt werden.

Befehlsaufbau:	"pN"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	N	Befehlscode
Kommentar:		Ein Rücksetzen der Position erfolgt ebenfalls nach einem Reset oder nach einer Referenzfahrt.
Beispiel:	"pN"	Die aktuelle Absolutposition wird auf 0 rückgesetzt..

3.3.15. Ausgänge setzen

Zur Synchronisation mit übergeordneten Prozessen oder einfach zur Erzeugung eines Schaltvorganges besitzt das ***isel***-PICMIC-Modul einen frei verwendbaren Ausgang, welcher über diese Funktion gesetzt oder gelöscht werden kann. Desweiteren können mit diesem Befehl die Endstufen definiert ein- oder ausgeschaltet werden.

Befehlsaufbau:	"pOxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	O	Befehlscode
	xx	Hexwert mit aktuellem Sollzustand, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Die Bitbelegung innerhalb des Hexwertes ist folgende: Bit0: Endstufen Ein/Aus Bit1: Ausgang1 Bit2-7: nicht benutzt
Beispiel:	"pO03"	Sowohl die Endstufen, als auch der frei verwendbare Ausgang werden eingeschaltet.

3.3.16. Position abfragen

Zur Feststellung des aktuellen Zustandes einer Positioniersteuerung gehört unbedingt die Abfrage der aktuellen Position. Beim isel-PICMIC-Modul ist dies auch während der Bewegung möglich.

Befehlsaufbau:	"pP"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	P	Befehlscode
Kommentar:		Nach Empfang dieses Befehls wird die aktuelle Position der Steuerung als Nachricht für den nächsten Sendeaufruf der Leitstation aufbereitet. Die Rückgabe der Positionsinformation erfolgt dann beim nächsten Sendeaufruf der Leitstation.
Rückgabe:	"pePxxxxxxxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	e	Fehlercode
	P	Befehlscode
	xxxxxxxx	Hexwert mit aktueller Position, 32-Bit (4Byte, 8Hexziffern)
Kommentar:		Die Rückgabe der Istposition erfolgt in Halbschritten. Da modulintern alle Positionen in 16-tel Schritten gerechnet und das höchstwertige Bit als Vorzeichen benutzt werden stehen als Wertevorrat nur 28Bit d.h. +/- 268435455 Halbschritte zur Verfügung.
Beispiel:	"pP"	Die aktuelle Position der Steuerung wird abgefragt. Die Rückantwort könnte folgendermaßen aussehen:
	"p0P00001000"	Die aktuelle Istposition ist 1000hex, d.h. 4096 dezimal.

3.3.17. Softwarereset

Durch die Softwareresetfunktion ist es möglich die Steuerung in einen definierten Ausgangszustand zu bringen. Nach Softwarereset werden alle Variablen neu initialisiert.

Befehlsaufbau: "pQ"

Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	Q	Befehlscode
Kommentar:		Nach Empfang dieses Befehls wird steuerungsintern ein Softwarereset durchgeführt. Damit werden auch die Defaultwerte aus dem E^2PROM neu gelesen und die Variablen entsprechend initialisiert.
Beispiel:	"pQ"	Die Steuerung führt einen Softwarereset aus.

3.3.18. Wert aus E^2PROM lesen

Das ***isel***-PICMIC-Modul verfügt über ein serielles E^2PROM, in welchem sowohl Defaultwerte und speicherbare Befehle, als auch Nutzerdaten abgelegt werden können. Dieses E^2PROM hat eine Größe von 4K-Bit und ist in 2 Speicherbänke eingeteilt. Bank0 ist dabei für Defaultwerte und Nutzerdaten vorgesehen, Bank1 dient der Speicherung von Programmen. Mit diesem Befehl können alle Werte aus Bank0 des E^2PROM ausgelesen werden.

Befehlsaufbau:	"pRaa"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	R	Befehlscode
	aa	Hexwert für Byteadresse, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Nach Empfang dieses Befehls wird der aktuelle Wert von Adresse aa aus Bank0 des E^2PROM gelesen und von der Steuerung als Nachricht für den nächsten Sendeaufruf der Leitstation aufbereitet. Die Rückgabe der Information erfolgt dann beim nächsten Sendeaufruf der Leitstation. Für Nutzerdaten ist dabei der Bereich von 80-FFhex vorgesehen, der Bereich von 0-7Fhex ist für steuerungsinterne Werte reserviert und darf vom Anwender nicht verändert werden.
Rückgabe:	"peRxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	e	Fehlercode
	R	Befehlscode
	xx	Hexwert mit aktuellem Wert, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Die Rückgabe des gespeicherten Wertes erfolgt als Hexwert. Für Nutzerdaten ist dabei der Bereich von 80-FFhex vorgesehen, der Bereich von 0-7Fhex ist für steuerungsinterne Werte reserviert und darf vom Anwender nicht verändert werden.
Beispiel:	"pR80"	Der Wert auf Adresse 80hex wird abgefragt. Die Rückantwort könnte folgendermaßen aussehen:
	"p0RAA"	Auf Adresse 80hex steht derzeit der Wert AAhex.

3.3.19. Statusinformation lesen

Die Statusinformation gibt dem Anwender einen aktuellen Überblick über den Zustand der Steuerung. Hier werden alle wichtigen Informationen in Form von einzelnen Bits zusammengestellt und dem Anwender zur Verfügung gestellt.

Befehlsaufbau:	"pS"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	S	Befehlscode
Kommentar:		Nach Empfang dieses Befehls wird der aktuelle Status von der Steuerung als Nachricht für den nächsten Sendeaufruf der Leitstation aufbereitet. Die Rückgabe der Information erfolgt dann beim nächsten Sendeaufruf der Leitstation.
Rückgabe:	"peSxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	e	Fehlercode
	R	Befehlscode
	xx	Hexwert mit aktuellem Status, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Die Rückgabe des aktuellen Status erfolgt als Hexwert. Folgende Bedeutung der Bits ist dabei festgelegt: Bit0 Drehrichtung (0-positiv/1-negativ) Bit1 Betriebsmode (0-Positioniermode/1-Drehzahlmode)

		Bit2	Programm (0-inaktiv/1-aktiv)
		Bit3	Stopmode (0-inaktiv/1-aktiv)
		Bit4	Runflag (0-Motor im Stillstand/1-Motor in Bewegung)
		Bit5	AccelFlag (1-Beschleunigungsphase/0-sonst)
		Bit6	Move Flag (1-Konstantphase/0-sonst)
		Bit7	DecelFlag (1-Bremsphase/0-sonst)
Beispiel:	"pS"	Der aktuelle Status der Steuerung wird abgefragt. Die Rückantwort könnte folgendermaßen aussehen:	
	"p0S52"	Der Betriebsmode Drehzahlbetrieb ist aktiv, der Motor ist mit konstanter Geschwindigkeit in Bewegung.	

3.3.20. Stop- und Breakbedingungen festlegen

Da das ***isel***-PICMIC-Modul auch Stand-Alone betrieben werden kann ist es sinnvoll, Bedingungen zu definieren die zu einem Stop- oder Breakbefehl führen. Solche Bedingungen können durch eine definierte Belegung von Eingangsports erzeugt werden. Dieser Befehl definiert solche Bedingungen in Form einer Aktivmaske und eines Sollwertes. Ist eine Bedingung aktiviert vergleicht die Steuerung den Istwert mit dem Sollwert und bei Gleichheit wird steuerungsintern ein Stop- oder Breakbefehl erzeugt. Sind mehrere Bedingungen aktiviert so werden diese mit Oder verknüpft. Damit wird eine aktive Zugriffsmöglichkeit auf die Befehls- und Programmabarbeitung im Modul geschaffen. Diese Bedingungen werden im E^2PROM abgelegt und stehen der Steuerung somit auch nach Reset oder Neueinschalten zur Verfügung. Ein Stopbefehl bewirkt das Anhalten einer aktiven Bewegung oder eines aktiven speicheren Programms. Wurde eine Bewegung gestoppt, so geht das Modul in den Stopmode über. Eine Breakfunktion beendet eine aktive Bewegung oder ein aktives speicherbares Programm.

Befehlsaufbau:	"pTaabbccdd"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	T	Befehlscode
	aa	Hexwert für Maske Stop, 8-Bit (1Byte, 2Hexziffern)
	bb	Hexwert für Wert Stop, 8-Bit (1Byte, 2Hexziffern)
	cc	Hexwert für Maske Break, 8-Bit (1Byte, 2Hexziffern)
	dd	Hexwert für Wert Break, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		als Bedingungen für einen Stop- oder Breakbefehl werden für Maske und Wert folgende Bits definiert:
		Bit0 nicht benutzt
		Bit1 nicht benutzt
		Bit2 Eingang1 als Bedingung
		Bit3 Eingang2 als Bedingung
		Bit4 Eingang3 als Bedingung
		Bit5 Eingang4 als Bedingung
		Bit6 nicht benutzt
		Bit7 nicht benutzt
Beispiel:	"pL08001000"	Der Eingang2 wird als aktive Bedingung für einen Stopbefehl und der Eingang3 als aktive Bedingung für einen Breakbefehl definiert. Beide Befehle werden erzeugt, wenn der entsprechende Eingang Low-Pegel besitzt.

3.3.21. Befehl aus E^2PROM rücklesen

Das ***isel***-PICMIC-Modul besitzt zur Abspeicherung von Befehlen ein serielles E^2PROM. Dieses E^2PROM ist in zwei Speicherbänke aufgeteilt, in Bank0 werden alle Defaultwerte und in Bank1 alle speicherbaren Befehle abgelegt. Ein Programm für den Stand-Alone-Mode der Steuerung kann bis zu 64 Befehle (siehe auch 3.4. Speicherbare Befehle) umfassen. Diese Befehle werden im Busmodus übertragen und entsprechend in der Bank1 des E^2PROM abgelegt. Da E^2PROM's nichtflüchtige Speicher sind bleibt ein abgelegtes Programm bis zum Überschreiben oder Löschen gültig. Mit dieser Funktion können die abgelegten Befehle aus dem E^2PROM rückgelesen werden.

Befehlsaufbau:	"pUnn"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	U	Befehlscode
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)

Kommentar:		Nach Empfang dieses Befehls wird der speicherbare Befehl mit der Satznummer nn von der Steuerung als Nachricht für den nächsten Sendeaufruf der Leitstation aufbereitet. Die Rückgabe der Information erfolgt dann beim nächsten Sendeaufruf der Leitstation.
Rückgabe:	`"peUnnaabbccdd"`	
Erläuterung:	`p`	Kennung für ***isel***-PICMIC-Modul
	`e`	Fehlercode
	`U`	Befehlscode
	`nn`	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	`aa`	Befehlscode als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	`bb`	Parameterbyte1 als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	`cc`	Parameterbyte2 als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	`dd`	Parameterbyte3 als Hexwert, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex) Befehlscodierung immer 1Byte Befehlscode und 3Byte Parameter. siehe auch 3.4. Speicherbare Befehle
Beispiel:	`"pU01"`	Der Befehl auf Satznummer 1 wird abgefragt. Die Rückantwort könnte folgendermaßen aussehen:
	`"p0U010603e800"`	Ein Befehl zum Setzen der Geschwindigkeit ist als Satznummer 1 abgespeichert. Im Programmablauf wird an dieser Stelle die Geschwindigkeit auf 1000Hz Halbschritt (03e8hex) eingestellt.

3.3.22. Versionsabfrage

Diese Funktion ermöglicht die Abfrage der Betriebssystemversion des ***isel***-PICMIC-Moduls.

Befehlsaufbau:	`"pV"`	
Erläuterung:	`p`	Kennung für ***isel***-PICMIC-Modul
	`V`	Befehlscode
Kommentar:		Nach Empfang dieses Befehls wird die aktuelle Betriebssystemversion von der Steuerung als Nachricht für den nächsten Sendeaufruf der Leitstation aufbereitet. Die Rückgabe der Information erfolgt dann beim nächsten Sendeaufruf der Leitstation.
Rückgabe:	`"peVstring"`	
Erläuterung:	`p`	Kennung für ***isel***-PICMIC-Modul
	`e`	Fehlercode
	`V`	Befehlscode
	`string`	Zeichenkette mit aktueller Betriebssystemkennzeichnung
Kommentar:		Die Rückgabe der Betreibssystemversion erfolgt als Zeichenkette. Die Anzahl der Zeichen ist dabei unbestimmt.
Beispiel:	`"pV"`	Die Betriebssystemversion der Steuerung wird abgefragt. Die Rückantwort könnte folgendermaßen aussehen:
	`"p0VpV1.00"`	Die Steuerung hat das Betriebssytem V1.00.

3.3.23. Wert nach E^2PROM schreiben

Das ***isel***-PICMIC-Modul verfügt über ein serielles E^2PROM, in welchem sowohl Defaultwerte und speicherbare Befehle, als auch Nutzerdaten abgelegt werden können. Dieses E^2PROM hat eine Größe von 4K-Bit und ist in 2 Speicherbänke eingeteilt. Bank0 ist dabei für Defaultwerte und Nutzerdaten vorgesehen, Bank1 dient der Speicherung von Programmen. Mit diesem Befehl können Daten in Bank0 des E^2PROM abgelegt werden.

Befehlsaufbau:	`"pWaaxx"`	
Erläuterung:	`p`	Kennung für ***isel***-PICMIC-Modul
	`W`	Befehlscode
	`aa`	Hexwert für Byteadresse, 8-Bit (1Byte, 2Hexziffern)
	`xx`	Hexwert für Datenbyte, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Nach Empfang dieses Befehls wird das Datenbyte xx auf Adresse aa der Bank0 des E^2PROM abgelegt.

		Für Nutzerdaten ist dabei der Bereich von 80-FFhex vorgesehen, der Bereich von 0-7Fhex ist für steuerungsinterne Werte reserviert und darf vom Anwender nicht verändert werden.
Beispiel:	"pW80AA"	Auf Adresse 80hex des E^2PROM Bank0 wird der Wert AAhex abgelegt.

3.3.24. Relative Bewegung

Die Ausführung einer relativen Positionierung ist eine Grundfunktionalität jeder Einachssteuerung. Diese Funktion ermöglicht die Positionierung um einen beliebigen Betrag mit der aktuell eingestellten Geschwindigkeit. Die Ausführung der Bewegung kann über die Statusabfrage überwacht und über Start-, Stop- und Breakfunktionen beeinflußt werden. Auch die aktuelle Position kann während der Bewegung abgefragt werden, da der Zugriff über den Bus erhalten bleibt.

Befehlsaufbau:	"pXxxxxxxxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	X	Befehlscode
	xxxxxxxx	Hexwert für Zielposition, 32-Bit (4Byte, 8Hexziffern)
Kommentar:		Die Angabe des Betrages erfolgt in Halbschritten. Da modulintern alle Positionen in 16-tel Schritten gerechnet und das höchstwertige Bit als Vorzeichen benutzt werden stehen als Wertevorrat für Positionen nur 28Bit d.h. +/-268435455 Halbschritte zur Verfügung. Beachten Sie bitte. daß die Überschreitung dieser absoluten Positionen nicht überprüft wird und somit zu falschen Ergebnissen führt.
Beispiel:	"pX00001000"	Es wird eine relative Positionierung um den Betrag 1000hex (4096 dezimal) mit der aktuell eingestellten Geschwindigkeit in Auftrag gegeben.

3.3.25. Bewege bis Impuls

Dieser Befehl ermöglicht das definierte Anhalten einer Bewegung nach einer bestimmten Anzahl von Impulsen. So kann z.B. ein Index genutzt werden, um den Motor eine bestimmte Anzahl von Umdrehungen zu bewegen oder es kann definiert ein Schalter im Bewegungsbereich der Mechanik angetastet werden. Voraussetzung hierfür ist jedoch immer, daß genügend Auslauf für den Bremsweg des Motors vorhanden ist. Als Impulseingang kann ein beliebiger Eingang der vier frei verwendbaren Anwendereingänge genutzt werden.

Befehlsaufbau:	"pYxxyy"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	Y	Befehlscode
	xx	Hexwert für Impulsdaten, 8-Bit (1Byte, 2Hexziffern)
	yy	Hexwert für Downcounter, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Nach Empfang dieses Befehls wird die Steuerung in den Drehzahlbetrieb geschaltet und in die definierte Richtung bewegt bis die eingehenden Impulse den Downcounter auf Null decrementiert haben. Ist der Downcounter gleich Null wird die Bewegung wie bei einer Referenzfahrt umgekehrt, bis ein Flankenwechsel am Zähleingang eintritt. Danach wird eine definierte Anzahl von Schritten in diese Richtung weitergefahren, um sicher außerhalb eines Schalters zu stehen und damit einen evtl. vorhanden Schalter gleichzeitig zu entprellen. Die Impulsdaten sind wie folgt aufgebaut: Bit0 Eingang LSB Bit1 Eingang MSB (00 = Eingang1, 11 = Eingang4) Bit2 Richtung (0-positiv/1-negativ) Bit3 HighLowAktivFlag (0-Lowaktiver/1-Highaktiver Eingang) Bit4 StepsOut LSB Bit5 StepsOut Bit6 StepsOut Bit7 StepsOut MSB (4Bit Schritte aus Schalter heraus) Die Angabe der Schritte (0-Fhex) erfolgt in Halbschritten.
Beispiel:	"pYA305"	Es wird eine Bewegung in positiver Richtung ausgeführt, bis am Eingang4 5 Highaktive Signale gezählt wurden. Anschließend wird in umgekehrter Richtung verfahren bis Eingang4 wieder Low-Pegel hat. Zusätzlich wird 10 Schritte in gleicher Richtung weitergefahren.

3.3.26. Referenzfahrt

Das Ausführen einer Referenzfahrt ist nach dem Einschalten einer Anlage meist notwendig, um die Mechanik mit der Steuerung zu synchronisieren. Beim ***isel***-PICMIC-Modul sind die Parameter für die Referenzfahrt relativ frei einstellbar. Als Referenzschalter kann dabei einer der vier freien Anwendereingänge dienen. Voraussetzung hierfür ist auch hier, daß genügend Auslauf für den Bremsweg des Motors vorhanden ist.

Befehlsaufbau:	"pZxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	Z	Befehlscode
	xx	Hexwert für Referenzdaten, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Nach Empfang dieses Befehls wird die Steuerung in den Drehzahlbetrieb geschaltet und in die definierte Richtung bewegt bis ein Flankenwechsel am definierten Eingang auftritt. Die Bewegung wird dann umgekehrt, bis ein erneuter Flankenwechsel am Eingang eintritt. Danach wird eine definierte Anzahl von Schritten in diese Richtung weitergefahren, um sicher außerhalb eines Schalters zu stehen und damit einen evtl. vorhanden Schalter gleichzeitig zu entprellen. Die Referenzdaten sind wie folgt aufgebaut: Bit0 Eingang LSB Bit1 Eingang MSB (00 = Eingang1, 11 = Eingang4) Bit2 Richtung (0-positiv/1-negativ) Bit3 HighLowAktivFlag (0-Lowaktiver/1-Highaktiver Eingang) Bit4 StepsOut LSB Bit5 StepsOut Bit6 StepsOut Bit7 StepsOut MSB (4Bit Schritte aus Schalter heraus) Die Angabe der Schritte (0-Fhex) erfolgt in Halbschritten.
Beispiel:	"pZA3"	Es wird eine Bewegung in positiver Richtung ausgeführt, bis am Eingang4 ein Highaktives Signal erkannt wurde. Anschließend wird in umgekehrter Richtung verfahren bis Eingang4 wieder Low-Pegel hat. Zusätzlich wird 10 Schritte in gleicher Richtung weitergefahren.

3.4. Speicherbare Befehle

3.4.1. Allgemeines

Einachssteuerungen werden in der Automation oft eingesetzt, um im Stand-Alone-Betrieb wiederkehrende Funktionsabläufe zu realisieren. Dazu muß eine Möglichkeit vorhanden sein entsprechende Programmabläufe dauerhaft abzuspeichern. Im ***isel***-PICMIC-Modul wurde diese Möglichkeit durch ein 4K-Bit serielles E^2PROM geschaffen. Dieses E^2PROM ist in zwei Speicherbänke unterteilt. Bank0 wird für Defaultwerte, steuerungsinterne Werte und Anwenderdaten benutzt, Bank1 steht für die Abspeicherung von Programmen zur Verfügung. Es können maximal 64 Befehle im E^2PROM abgespeichert werden.

3.4.2. Befehlsaufbau für speicherbare Befehle

Alle speicherbaren Befehle werden mit je 4Byte kodiert, wobei das erste Byte jeweils den Befehlscode und alle weitern die entsprechenden Parameter enthalten. Dabei müssen auch nicht benutzte Parameterbytes immer erzeugt und an die Steuerung übergeben werden. Die speicherbaren Befehle werden über den DIN-Meßbus mit dem Befehl "E"-"Befehl im E^2PROM speichern" an die Steuerung übetragen und können mit dem Befehl "U"-"Befehl aus E^2PROM rücklesen" zurückgelesen werden. Es stehen 16 Befehle für speicherbare Programme zur Verfügung:

Code	Kurzbeschreibung	Parameter	
00	Programmende	keine	
01	Bewege Absolut	Zielposition	3Byte
02	Bewege Relativ	Zielposition	3Byte
03	Referenzfahrt	Referenzdaten	1Byte
04	Bewege bis Impuls	Impulsdaten, Impulszahl	2Byte
05	Nullpunkt setzen	Keine	
06	Sollgeschwindigkeit einstellen	Sollgeschwindigkeit	2Byte
07	Vorteiler für Beschleunigung einstellen	Beschleunigungsvorteiler	1Byte
08	Sollstrom und Stromabsenkung einstellen	Sollstrom, Stromabsenkung	2Byte
09	Ausgänge setzen	Maske, Wert	2Byte
0A	Eingänge lesen, verzweigen	Maske, Wert, Sprungadresse	3Byte
0B	Schleife, Verzweigung	Schleifenzahl, Sprungadresse	2Byte
0C	Verzögerung	Verzögerungszeit	1Byte
0D	Betriebsmodus einstellen	Betriebsmode	1Byte
0E	Geschwindigkeit erhöhen (nur Drehzahlbetrieb)	Maske, Wert, Offset	3Byte
0F	Geschwindigkeit vermindern (nur Drehzahlbetrieb)	Maske, Wert, Offset	3Byte

3.4.3. Beschreibung der speicherbaren Befehle

3.4.3.1. Programmende

Ein Programmende muß prinzipiell als letzter Satz eines Programms geschrieben werden.

Befehlsaufbau: `"pEnn00aabbcc"`

Erläuterung:		
	`p`	Kennung für ***isel***-PICMIC-Modul
	`E`	Befehlscode, Befehl im E^2PROM speichern
	`nn`	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	`00`	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	`aa`	Parameterbyte1, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
	`bb`	Parameterbyte2, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
	`cc`	Parameterbyte3, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex) Ein Programmende muß unbedingt als letzter Befehl eines Programmes im Speicher abgelegt sein.
Beispiel:	`"pE0100000000"`	Ein Befehlsende wird auf Satznummer 01 geschrieben.

3.4.3.2. Bewege Absolut

Die Ausführung einer absoluten Positionierung ist eine Grundfunktionalität jeder Einachssteuerung. Diese Funktion ermöglicht die Speicherung eines Befehls für die Positionierung auf eine beliebige absolute Position mit der aktuell eingestellten Geschwindigkeit. Die Ausführung der Bewegung kann über die Statusabfrage überwacht und über Start-, Stop- und Breakfunktionen beeinflußt werden.

Befehlsaufbau:	"pEnn01xxxxxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	01	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	xxxxxx	Zielposition, 24-Bit (3Byte, 6Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex) Die Angabe der Zielposition erfolgt in Halbschritten. Da der Parameter hier auf 3Byte begrenzt ist und das höchstwertige Bit als Vorzeichen benutzt wird, stehen als Wertevorrat für speicherbare Zielpositionen nur 23Bit d.h. +/-8388607 Halbschritte zur Verfügung.
Beispiel:	"pE0101001000"	Eine absolute Bewegung wird auf Satznummer 01 geschrieben. Die Zielposition ist 1000hex (4096 dezimal).

3.4.3.3. Bewege Relativ

Auch die Ausführung einer relativen Positionierung ist eine Grundfunktionalität jeder Einachssteuerung. Diese Funktion ermöglicht die Speicherung eines Befehls für die Positionierung um einen beliebigen Betrag mit der aktuell eingestellten Geschwindigkeit. Die Ausführung der Bewegung kann über die Statusabfrage überwacht und über Start-, Stop- und Breakfunktionen beeinflußt werden.

Befehlsaufbau:	"pEnn02xxxxx"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	02	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	xxxxxx	Weg, 24-Bit (3Byte, 6Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex) Die Angabe des Weges erfolgt in Halbschritten. Da der Parameter hier auf 3Byte begrenzt ist und das höchstwertige Bit als Vorzeichen benutzt wird, stehen als Wertevorrat nur 23Bit d.h. +/-8388607 Halbschritte zur Verfügung.
Beispiel:	"pE0102001000"	Eine relative Bewegung wird auf Satznummer 01 geschrieben. Der Weg beträgt 1000hex (4096 dezimal).

3.4.3.4. Referenzfahrt

Die Referenzfahrt dient meist am Beginn eines Programmablaufes zur Synchronisation der Mechanik mit der Steuerung. Beim ***isel***-PICMIC-Modul sind die Parameter für die Referenzfahrt relativ frei einstellbar. Als Referenzschalter kann dabei einer der vier freien Anwendereingänge dienen. Voraussetzung hierfür ist auch hier, daß genügend Auslauf für den Bremsweg des Motors vorhanden ist.

Befehlsaufbau:	"pEnn03xxbbcc"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	03	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	xx	Referenzdaten, 8-Bit (1Byte, 2Hexziffern)
	bb	Parameterbyte2, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
	cc	Parameterbyte3, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex) Die Speicherung dieses Befehls bewirkt im Programmablauf eine Referenzfahrt. Bei Abarbeitung dieses Befehls wird die Steuerung in den Drehzahlbetrieb geschaltet und in die definierte Richtung bewegt bis ein Flankenwechsel am definierten Eingang auftritt. Die Bewegung wird dann

umgekehrt, bis ein erneuter Flankenwechsel am Eingang eintritt. Danach wird eine definierte Anzahl von Schritten in diese Richtung weitergefahren, um sicher außerhalb eines Schalters zu stehen und damit einen evtl. vorhanden Schalter gleichzeitig zu entprellen. Die Referenzdaten sind wie folgt aufgebaut:

Bit0 Eingang LSB
Bit1 Eingang MSB (00 = Eingang1, 11 = Eingang4)
Bit2 Richtung (0-positiv/1-negativ)
Bit3 HighLowAktivFlag (0-Lowaktiver/1-Highaktiver Eingang)
Bit4 StepsOut LSB
Bit5 StepsOut
Bit6 StepsOut
Bit7 StepsOut MSB (4Bit Schritte aus Schalter heraus)

Die Angabe der Schritte (0-Fhex) erfolgt in Halbschritten.

Beispiel: `"pE0103A30000"` Bei Abarbeitung des Befehls wird eine Bewegung in positiver Richtung ausgeführt, bis am Eingang4 ein Highaktives Signal erkannt wurde. Anschließend wird in umgekehrter Richtung verfahren bis Eingang4 wieder Low-Pegel hat. Zusätzlich wird 10 Schritte in gleicher Richtung weitergefahren.

3.4.3.5. Bewege bis Impuls

Die Abspeicherung dieses Befehls ermöglicht das definierte Ahnhalten einer Bewegung nach einer bestimmten Anzahl von Impulsen. So kann z.B. ein Index genutzt werden, um den Motor eine bestimmte Anzahl von Umdrehungen zu bewegen oder es kann definiert ein Schalter im Bewegungsbereich der Mechanik angetastet werden. Voraussetzung hierfür ist jedoch immer, daß genügend Auslauf für den Bremsweg des Motors vorhanden ist. Als Impulseingang kann ein beliebiger Eingang der vier frei verwendbaren Anwendereingänge genutzt werden.

Befehlsaufbau: `"pEnn04aabbcc"`

Erläuterung:
- `p` Kennung für ***isel***-PICMIC-Modul
- `E` Befehlscode, Befehl im E^2PROM speichern
- `nn` Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
- `04` Befehlscode, 8-Bit (1Byte, 2Hexziffern)
- `aa` Impulsdaten, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
- `bb` Downcounter, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
- `cc` Parameterbyte3, ungenutzt, 8-Bit (1Byte, 2Hexziffern)

Kommentar: Satznummern von 0 - 63 dezimal (00 - 3f hex)

Die Speicherung dieses Befehls bewirkt im Programmablauf, daß die Steuerung in den Drehzahlbetrieb geschaltet und der Motor in die definierte Richtung bewegt wird, bis die eingehenden Impulse den Downcounter auf Null dekrementiert haben. Ist der Downcounter gleich Null wird die Bewegung wie bei einer Referenzfahrt umgekehrt bis ein Flankenwechsel am Zähleingang eintritt. Danach wird eine definierte Anzahl von Schritten in diese Richtung weitergefahren, um sicher außerhalb eines Schalters zu stehen und damit einen evtl. vorhanden Schalter gleichzeitig zu entprellen. Die Impulsdaten sind wie folgt aufgebaut:

Bit0 Eingang LSB
Bit1 Eingang MSB (00 = Eingang1, 11 = Eingang4)
Bit2 Richtung (0-positiv/1-negativ)
Bit3 HighLowAktivFlag (0-Lowaktiver/1-Highaktiver Eingang)
Bit4 StepsOut LSB
Bit5 StepsOut
Bit6 StepsOut
Bit7 StepsOut MSB (4Bit Schritte aus Schalter heraus)

Die Angabe der Schritte (0-Fhex) erfolgt in Halbschritten.

Beispiel: `"pE0104A30500"` Bei Abarbeitung des Befehls wird eine Bewegung in positiver Richtung ausgeführt, bis am Eingang4 5 Highaktive Signale gezählt wurden. Anschließend wird in umgekehrter Richtung verfahren bis Eingang4 wieder Low-Pegel hat. Zusätzlich wird 10 Schritte in gleicher Richtung weitergefahren.

3.4.3.6. Nullpunkt setzen

Das Rücksetzen der Positionsregister ist eine Grundfunktion jeder Positioniersteuerung. Damit kann der Nullpunkt für eine Positionierung an einen beliebigen Punkt im Bewegungsbereich der Mechanik gesetzt werden.

Befehlsaufbau:	"pEnn05aabbcc"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	05	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	aa	Parameterbyte1, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
	bb	Parameterbyte2, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
	cc	Parameterbyte3, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex) Ein Rücksetzen der Positionsregister erfolgt ebenfalls nach Reset oder nach einer Referenzfahrt.
Beispiel:	"pE0105000000"	Ein Nullpunkt setzen wird auf Satznummer 01 geschrieben.

3.4.3.7. Sollgeschwindigkeit setzen

Zur Ausführung einer Bewegung wird immer eine Geschwindigkeit benötigt. Wurde keine Geschwindigkeit übergeben, wird eine auszuführende Bewegung mit der zuletzt eingestellten (Default-) Geschwindigkeit abgearbeitet. Im Drehzahlbetrieb wird die neue Geschwindigkeit mit der aktuellen Beschleunigung eingestellt, die Drehrichtung wird dabei vom Vorzeichen der Geschwindigkeit abgeleitet. Im Positionierbetrieb gilt die Geschwindigkeit für die nächste Bewegung, die Drehrichtung ergibt sich hier aus dem Positionierbefehl, so daß die Geschwindigkeitsangabe hier immer positiv sein sollte.

Befehlsaufbau:	"pEnn06xxxxcc"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	06	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	xxxx	Sollgeschwindigkeit, 16-Bit (2Byte, 4Hexziffern)
	cc	Parameterbyte3, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex) maximale Geschwindigkeit steuerungsseitig ist 24000Hz Halbschritt, der Wertebereich ist also 0-24000Hz (0000 - 5dc0 hex) im Positioniermode und +/-24000Hz (a240 - 5dc0 hex) im Drehzahlmode. Die tatsächlich erreichbare Geschwindigkeit in einer Applikation kann hier nicht angegeben werden, da dies stark vom Motor selbst, dessen Lastverhalten und der angeschlossenen Mechanik abhängig ist.
Beispiel:	"pE010603e800"	Bei Abarbeitung des Befehls wird die Geschwindigkeit auf 1000Hz Halbschritt (03e8hex) eingestellt. Diese Einstellung ist bis zur Einstellung einer neuen Geschwindigkeit gültig.

3.4.3.8. Vorteiler für Beschleunigung einstellen

Die Beschleunigung wird vom Modul über einen Timerinterrupt mit einer Grundfrequenz von 900Hz, einem Offset von 100Hz und einer Start-Stop-Frequenz von 300Hz erzeugt. Die Änderung der Beschleunigung wird über einen 8-Bit-Software-Upcounter ermöglicht. Mit dieser Funktion kann der Relaodwert für diesen Upcounter festgelegt werden.

Befehlsaufbau:	"pEnn07xxbbcc"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	07	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	xx	Beschleunigungsvorteiler, 8-Bit (1Byte, 2Hexziffern)
	bb	Parameterbyte2, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
	cc	Parameterbyte3, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex)

FF = maximale Beschleunigung
00 = minimale Beschleunigung

Beispiel: "pE0107fe0000" Bei Abarbeitung des Befehls wird der Reloadwert für die Beschleunigung auf fe hex eingestellt d.h. die Geschwindigkeit wird in einem Zeitraster von 2.2ms (450Hz) verändert. Bei einem Offset von 100Hz ergibt sich somit eine Beschleunigung von 45 Hz/ms.

3.4.3.9. Sollstrom und Stromabsenkung einstellen

Bei Schrittmotorsteuerungen werden i.A. bei Motorstillstand die Phasenströme abgesenkt um eine unnötige Erwärmung der Motoren und Endstufen zu verhindern. Für das ***isel***-PIMIC-Modul können die Phasenströme softwaremäßig eingestellt werden. Der Mikrokontroller der Steuerung erzeugt zur Ansteuerung der Endstufen dazu ein PWM-Signal mit 8-Bit Auflösung. Diese Funktion ermöglicht die Einstellung von zwei Stromsollwerten, getrennt für Betriebsstrom und abgesenkten Strom.

Befehlsaufbau: "pEnn08aabbcc"

Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	08	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	aa	Sollstrom, 8-Bit (1Byte, 2Hexziffern)
	bb	Stromabsenkung, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
	cc	Parameterbyte3, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex)
		FF = maximaler Strom
		00 = minimaler Strom

Beispiel: "pE0108f27f00" Bei Abarbeitung des Befehls wird der Sollstrom auf 95%, der abgesenkte Strom auf 50% des Maximalstromes eingestellt. Die Stromabsenkung erfolgt automatisch bei Stillstand des Motors. Soll keine Stromabsenkung erfolgen müssen beide Werte gleich gesetzt werden.

3.4.3.10. Ausgänge setzen

Zur Synchronisation mit übergeordneten Prozessen oder einfach zur Erzeugung eines Schaltvorganges besitzt das ***isel***-PICMIC-Modul einen frei verwendbaren Ausgang, welcher über diese Funktion gesetzt oder gelöscht werden kann. Desweiteren können mit diesem Befehl die Endstufen definiert ein- oder ausgeschaltet werden.

Befehlsaufbau: "pEnn09aabbcc"

Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	09	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	aa	Aktivmaske, 8-Bit (1Byte, 2Hexziffern)
	bb	Sollwert, 8-Bit (1Byte, 2Hexziffern)
	cc	Parameterbyte3, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex)
		Die Bitbelegung innerhalb des Sollwertes ist folgende:
		Bit0: Endstufen Ein/Aus
		Bit1: Ausgang1
		Bit2-7: nicht benutzt

Beispiel: "pE0109030000" Bei Abarbeitung dieses Befehls werden sowohl die Endstufen, als auch der frei verwendbare Ausgang eingeschaltet.

3.4.3.11. Eingang lesen, verzweigen

Als grundlegendes Element eines Programmablaufes wird oft als Kontrollstruktur die bedingte Verzweigung (if.....then.....else.....) benötigt. Dies kann beim ***isel***-PICMIC-Modul über die Verwendung der Eingänge und entsprechende Sprungziele realisiert werden. Dieser Befehl ermöglicht eine Verzweigung auf Grund einer Eingangsbelegung. Das Sprungziel ist dabei immer absolut und bezieht sich auf die Satznummern innerhalb des Programms.

Befehlsaufbau: "pEnn0Aaabbcc"

Erläuterung:		
	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	0A	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	aa	Aktivmaske, 8-Bit (1Byte, 2Hexziffern)
	bb	Sollwert, 8-Bit (1Byte, 2Hexziffern)
	cc	Sprungadresse, 8-Bit (1Byte, 2Hexziffern)

Kommentar: Satznummern von 0 - 63 dezimal (00 - 3f hex)
Für Aktivmaske und Sollwert gilt folgende Bitbelegung:

Bit0	nicht benutzt
Bit1	nicht benutzt
Bit2	Eingang1
Bit3	Eingang2
Bit4	Eingang3
Bit5	Eingang4
Bit6	nicht benutzt
Bit7	nicht benutzt

Als Sprungadressen sind Satznummern von 0-63 dezimal (00-3Fhex) zulässig. Eine Überprüfung der Sinnfälligkeit der Sprungziele erfolgt nicht.

Beispiel: "pE010A404032" Bei Abarbeitung des Befehls wird der Eingang4 auf High-Pegel getestet. Ist dies der Fall wird nach Adresse 50 (32hex) gesprungen, sonst wird mit dem nächsten Befehl im Programmablauf fortgefahren.

3.4.3.12. Schleife, Verzweigung

Neben der bedingten Verzweigung stellt das ***isel***-PICMIC-Modul als Kontrollstrukturen noch eine nichtabweisende Zählschleife (do..... while(loopcounter>0)) und einen Sprung (goto.....) zur Verfügung. Beide Kontrollstrukturen können mit dem gleichen Befehl realisiert werden. Für einen Sprung muß lediglich der Schleifenzähler mit 0 angegeben werden. In beiden Fällen werden als Sprungziel absolute Adressen (Satznummern) vorausgesetzt, so daß sowohl Vorwärts- als auch Rückwärtssprünge realisierbar sind.

Befehlsaufbau: "pEnn0Baabbcc"

Erläuterung:		
	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	0B	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	aa	Schleifenzahl, 8-Bit (1Byte, 2Hexziffern)
	bb	Sprungadresse, 8-Bit (1Byte, 2Hexziffern)
	cc	Parameterbyte3, reserviert, 8-Bit (1Byte, 2Hexziffern)

Kommentar: Satznummern von 0 - 63 dezimal (00 - 3f hex)
Die Schleifenzahl hat einen Wertebereich von 0-255 (00-FFhex) und gibt die Anzahl der Wiederholungen an. Wird als Schleifenzahl aa eine 0 programmiert ergibt sich ein unbedingter Sprung zur Sprungadresse bb. Als Sprungadressen sind Satznummern von 0-63 dezimal (00-3Fhex) zulässig. Eine Überprüfung der Sinnfälligkeit der Sprungziele erfolgt nicht.

Beispiel: "pE300B042000" Bei Abarbeitung des Befehls wird ein interner Schleifenzähler eingerichtet und bei jedem Durchlauf inkrementiert. Ist die Schleifenzahl (04hex) erreicht wird mit dem nächsten Befehl fortgefahren, sonst wird die Adresse 20hex angesprungen. In diesem Beispiel würden die Befehle von der Adresse 20hex bis zur Adresse 30hex 4 mal wiederholt, also insgesamt 5 mal abgearbeitet.

"pE300B002000" Bei Abarbeitung dieses Befehls wird immer die Adresse 20hex angesprungen, dies würde einer Endlosschleife gleichkommen.

3.4.3.13. Verzögerung

Zur Synchronisation mit anderen Prozessen werden oft Zeitverzögerungen benötigt. Diese Funktion stellt eine Verzögerung mit einer Auflösung von 1/10 Sekunde bereit.

Befehlsaufbau:	"pEnn0Cxxbbcc"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	0C	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	xx	Verzögerungszeit, 8-Bit (1Byte, 2Hexziffern)
	bb	Parameterbyte2, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
	cc	Parameterbyte3, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex) Die Verzögerungszeit wird in 1/10 Sekunden angegeben, der Wertebereich ist 0-255 (00-FFhex).
Beispiel:	"pE010C0A0000"	Bei Abarbeitung dieses Befehls wird eine Verzögerungszeit von 1 Sekunde erzeugt.

3.4.3.14. Betriebsmode einstellen

Um ein möglichst breites Anwendungsspektrum abdecken zu können sind zwei Betriebsmodes vorgesehen der Positionierbetrieb, in welchem beliebige relative und absolute Positionen angefahren werden können, und der Drehzahlbetrieb, in welchen die Steuerung immer versucht die vorgegebene Drehzahl zu erreichen. Dieser Befehl erlaubt die Einstellung des Betriebsmodes auch für speicherbare Programme.

Befehlsaufbau:	"pEnn0Dxxbbcc"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	0D	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	xx	Betriebsmode, 8-Bit (1Byte, 2Hexziffern)
	bb	Parameterbyte2, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
	cc	Parameterbyte3, ungenutzt, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex) 00 = Positionierbetrieb 01 = Drehzahlbetrieb
Beispiel:	"pE010D000000"	Bei Abarbeitung dieses Befehls wird als Betriebsmode der Positioniermode eingestellt. Diese Einstellung ist bis zur Einstellung eines neuen Betriebsmode gültig.

3.4.3.15. Geschwindigkeit erhöhen

Um im Drehzahlbetrieb eine möglichst einfache manuelle Geschwindigkeitseinstellung realisieren zu können wurde mit diesem Befehl die Möglichkeit geschaffen über eine definierte Eingangsbelegung eine Erhöhung der Geschwindigkeit zu erreichen. Somit kann im einfachsten Fall über einen Taster und ein entsprechendes speicherbares Programm eine manuelle Geschwindigkeitsanpassung im Drehzahlbetrieb realisiert werden.

Befehlsaufbau:	"pEnn0Eaabbcc"	
Erläuterung:	p	Kennung für ***isel***-PICMIC-Modul
	E	Befehlscode, Befehl im E^2PROM speichern
	nn	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	0E	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	aa	Aktivmaske, 8-Bit (1Byte, 2Hexziffern)
	bb	Sollwert, 8-Bit (1Byte, 2Hexziffern)
	cc	Offset, 8-Bit (1Byte, 2Hexziffern)
Kommentar:		Satznummern von 0 - 63 dezimal (00 - 3f hex) Für Aktivmaske und Sollwert gilt folgende Bitbelegung: Bit0 nicht benutzt Bit1 nicht benutzt Bit2 Eingang1 Bit3 Eingang2 Bit4 Eingang3 Bit5 Eingang4 Bit6 nicht benutzt

Bit7 nicht benutzt
Als Offset sind Werte von 0-255 dezimal (00-FFhex) zulässig. Der Befehl ist nur bei Drehzahlbetrieb zulässig.

Beispiel: `"pE010E40400A"` Bei Abarbeitung des Befehls wird der Eingang4 auf High-Pegel getestet. Ist dies der Fall wird die Geschwindigkeit um 10 Schritte/s (0Ahex) erhöht, sonst wird mit dem nächsten Befehl im Programmablauf fortgefahren.

3.4.3.16. Geschwindigkeit vermindern

Um im Drehzahlbetrieb eine möglichst einfache manuelle Geschwindigkeitseinstellung realisieren zu können wurde mit diesem Befehl die Möglichkeit geschaffen über eine definierte Eingangsbelegung eine Verminderung der Geschwindigkeit zu erreichen. Somit kann im einfachsten Fall über einen Taster und ein entsprechendes speicherbares Programm eine manuelle Geschwindigkeitsanpassung im Drehzahlbetrieb realisiert werden.

Befehlsaufbau: `"pEnn0Faabbcc"`

Erläuterung:

	`p`	Kennung für ***isel***-PICMIC-Modul
	`E`	Befehlscode, Befehl im E^2PROM speichern
	`nn`	Satznummer als Hexwert, 8-Bit (1Byte, 2Hexziffern)
	`0F`	Befehlscode, 8-Bit (1Byte, 2Hexziffern)
	`aa`	Aktivmaske, 8-Bit (1Byte, 2Hexziffern)
	`bb`	Sollwert, 8-Bit (1Byte, 2Hexziffern)
	`cc`	Offset, 8-Bit (1Byte, 2Hexziffern)

Kommentar: Satznummern von 0 - 63 dezimal (00 - 3f hex)
Für Aktivmaske und Sollwert gilt folgende Bitbelegung:
Bit0 nicht benutzt
Bit1 nicht benutzt
Bit2 Eingang1
Bit3 Eingang2
Bit4 Eingang3
Bit5 Eingang4
Bit6 nicht benutzt
Bit7 nicht benutzt
Als Offset sind Werte von 0-255 dezimal (00-FFhex) zulässig. Der Befehl ist nur bei Drehzahlbetrieb zulässig.

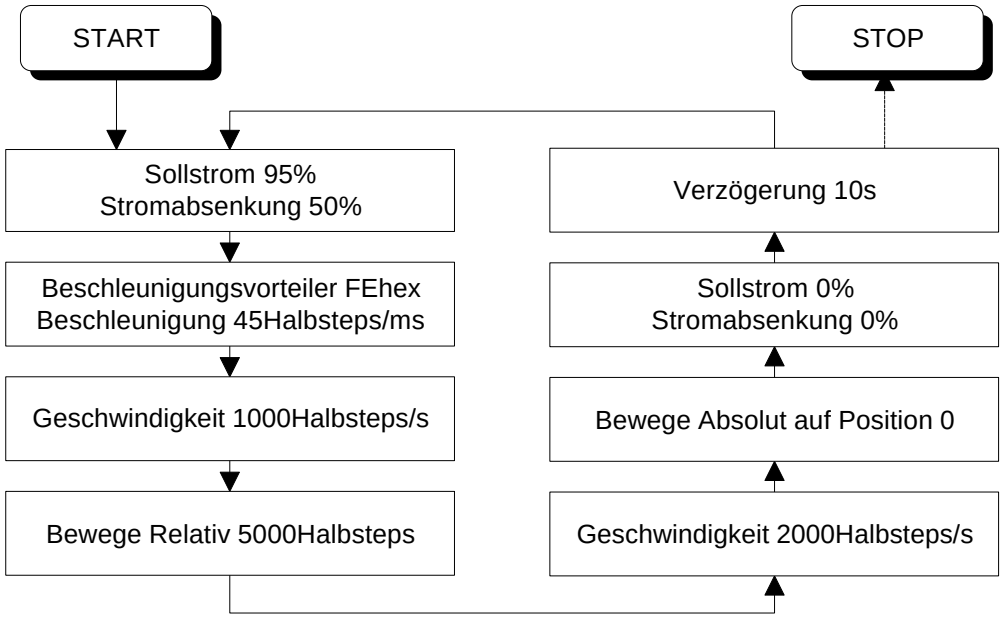
Beispiel: `"pE010F40400A"` Bei Abarbeitung des Befehls wird der Eingang4 auf High-Pegel getestet. Ist dies der Fall wird die Geschwindigkeit um 10 Schritte/s (0Ahex) vermindert, sonst wird mit dem nächsten Befehl im Programmablauf fortgefahren.

3.4.4. Ein kurzes Beispiel für ein speicherbares Programm

Folgendes kurze Beispiel soll die Anwendung speicherbarer Befehle verdeutlichen. Die einzelnen Befehle werden jeweils mit einem direkten Befehl („pEncp1p2p3“) an die Steuerung übertragen. Das Programm kann mit einem Startbefehl („pG“), oder nach Definierung einer Startbedingung („pL.....“) mit einer bestimmten Eingangsbelegung und/oder nach Reset gestartet werden.

N=Satznummer, C=Befehlscode, P1=Parameterbyte1, P2=Parameterbyte2, P3=Parameterbyte3

N	C	P1	P2	P3	Bedeutung
00	08	F2	7F	00	Sollstrom 95%, Stromabsenkung 50%
01	07	FE	00	00	Beschleunigungsvorteiler FEhex, 45 Halbschritte/ms
02	06	03	E8	00	Geschwindigkeit 1000 Halbschritte/s
03	02	00	13	88	Bewege Relativ 5000 Halbschritte
04	06	07	D0	00	Geschwindigkeit 2000 Halbschritte/s
05	01	00	00	00	Bewege Absolut auf Position 0
06	08	00	00	00	Sollstrom 0%, Stromabsenkung 0%, Strom aus
07	0C	64	00	00	Verzögerung 10s
08	0B	00	00	00	unbedingter Sprung auf Satznr. 0, Endlosschleife
09	00	00	00	00	Programmende, wird nie erreicht



4. Hardwarebeschreibung

4.1. Allgemeines

Das ***isel***-Schrittmotormodul 'PICMIC' ist als Leistungs- sowie Steuereinheit für bipolare Zweiphasenschrittmotoren vorgesehen. Die angeschlossen Schrittmotoren können mit der vorgestellten Leistungseinheit maximal im 1/16 Microschrittbetrieb betrieben werden. Weiterhin ist das Modul für den Anschluß von 4 digitalen optisch isolierten Eingängen sowie einem optisch isolierten Ausgang vorgesehen. Die Ein-/Ausgänge sind mit 24V DC Versorgungsspannung betreibbar. Für notwendige Positionieraufgaben stellt das Modul eine galvanisch getrennte vollduplex RS485-Schnittstelle mit enablebarem Ausgangstreiber zur Verfügung. In einer Option ist auch der Anschluß einer Standard-RS232 Schnittstelle möglich. Zur Abspeicherung von Bewegungsfunktionen, Parametern usw. steht auf der Baugruppe ein E^2PROM zur Verfügung. Durch den einstellbaren Phasenstrom ist die Endstufe an verschiedenste Einsatzfälle anpaßbar.

4.2. Schrittmotorleistungstreiber

Zum Treiben der bipolaren Zweiphasenschrittmotoren ist eine integrierte Leistungsendstufe eingesetzt. Diese Leistungsendstufe bietet unter anderem die Möglichkeit die angeschlossenen Schrittmotoren im 1/16-tel Mikroschrittbetrieb, je nach Softwareansteuerung, zu betreiben. Der Anschluß der Motoren erfolgt am Steckverbinder H100. Die Anschlußbelegung des Steckverbinders ist im Kapitel Steckerbelegung (siehe 4.5. Steckerbelegung) beschrieben. Die maximale Versorgungsspannung der Endstufe beträgt 45V DC, minimal 12V DC. In jedem Fall sollten Sie beachten, daß die Endstufe nach dem Konstantstromprinzip arbeitet. Der maximal mögliche Treiberstrom beträgt ca. 2.8A und ist mittels Software an verschiedenste Motoren anpaßbar. Die eingesetzte Endstufe ist bedingt kurzschlußfest, auf einen ordnungsgemäßen Anschluß Ihres Schrittmotors ist achtzugeben um eine Zerstörung auszuschließen. Die Spannung für die Motorversorung muß gleichgerichtet mit einer Restwelligkeit von <0.5V zur Verfügung gestellt werden.

Kurzübersicht der Leistungsparameter der Leistungsendstufe	
Versorgungsspannung	12V DC - 45V DC, > 2A, Einspeisung Steckverbinder H2 Pin 1, 2
Strom Endstufe	0.5A - 2.8A, kleinere Werte bedingt möglich
Steuerprinzip	Konstantstromansteuerung Chopper
Induktivität des Motors	< 10mH

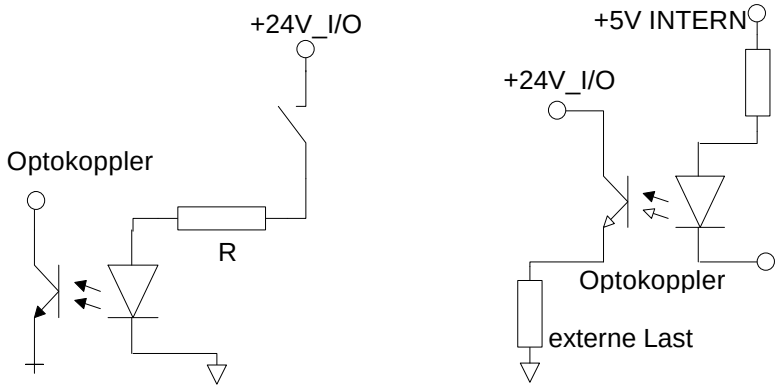
Der eingesetzte Leistungstreiber bietet, wie oben erwähnt, die Möglichkeit der Mikroschrittansteuerung mit einstellbarem Maximalstrom. Zur Realisierung dieser Funktionalität sind am eingesetzten Mikroprozessor Portpins reserviert deren Beschreibung im Abschnitt Prozessorteil (siehe 4.4. Prozessorteil) erfolgt. Die mit einem Netzteil zur Verfügung gestellte Spannungsversorgung der Endstufe muß mit einer entsprechenden Sicherung gegen Kurzschluß abgesichert sein, da aufgrund der Größe der Baugruppe eine Sicherung nicht mehr integrierbar war. Im Netzteil sollte ein ausreichend großer Glättungskondensator vorhanden sein, um den Stromrippel auf den Versorgungsleitungen zur PICMIC-Baugruppe klein zu halten.

4.3. Digitale Ein-/Ausgabe

Die vorgestellte Hardwareeinheit stellt 4 digitale, optisch isolierte Eingänge sowie einen digitalen, optisch isolierten Schaltausgang für diverse Schaltvorgänge zur Verfügung. Die Eingänge sind für den Betrieb mit industrieüblichen 24V Ein-/Ausgabespannungen vorgesehen. Die auf der Schaltung vorhandenen Vorwiderstände sind für eine Schaltspannung von 24V DC ausgelegt. Ein extern mit +24V beaufschlagter Eingang wird intern als LOW am entsprechenden Portpin gelesen, d.h. am Portpin der Baugruppe ist die Anode des Optokopplers verfügbar. Der vorhandene Schaltausgang ist ebenfalls galvanisch getrennt und als ' HIGH SIDE - Schalter ' ausgeführt, zu schaltende Verbraucher müssen daher mit einem Pol an GND_24V_I/O liegen. Nach einem RESET ist der Ausgang abgeschaltet. LOW am Prozessorpin führt zum Einschalten des Ausgangstreibers.

Kurzübersicht digitale Ein-/Ausgänge	
Vesorgungsspannung	20 - 27V DC, ca. 0.6A, Restwelligkeit <0.5V
Eingänge	U_{IN} < 1V entspricht am Prozessor HIGH U_{IN} > 19V entspricht am Prozessor LOW **ACHTUNG**: Da intern keine Schwellenspannungen auswertbar sind, müssen die Eingangsschaltspannungen in den oben genannten Grenzen liegen.
Ausgang	HIGH-SIDE Schalter, Kurzschlußgeschützt, maximaler Schaltstrom in dieser Applikation --> 0.3A !

Im Text dieser Beschreibung wird die Spannungsversorgung der Ein-/Ausgänge als 24V_I/O bezeichnet. Der negativere Pol dieser Spannungsversorgung wird als GND_24V_I/O bezeichnet. Die 24V_I/O Spannung sollte in jedem Fall als separate Spannungsversorgung mit getrennter Masse (galvanisch getrennt) in Ihrer Anwendung zur Verfügung gestellt werden. Nur so können Sie einen ausreichend störsicheren Betrieb garantieren. Die beschriebenen Eingänge sind **nicht** gegen Verpolung der Betriebsspannung geschützt. Das Einspeisen einer negativeren Spannung als -5.0V gegenüber dem GND_24V_IO-Versorgungspin in den Eingang führt zu einer Zerstörung des ensprechenden Eingangs. Die Ein-/Ausgänge sind auf den Steckverbindern H1.1 und H1.2 verfügbar. Die zur Verfügung gestellte 24V_IO Schaltspannung ist extern mit einer entsprechenden Sicherung zu schützen.



4.4. Prozessorteil

Um einen möglichst universellen Einsatz der vorgestellten Baugruppe zu ermöglichen wurde als zentraler Kern ein maskenprogrammierter Mikroprozessor eingesetzt. Nachfolgend beschriebene PIN's des Prozessors sind für die interne Schaltungsrealisierung verwendet:

PortB		
Pin	Richtung	Verwendung
RB0	OUT IN	Endstufe D/A-Wandler M1 Phase A (LSB) Schließer Grenzwertmelder Übertemperatur
RB1	OUT	Endstufe M2 Phase A
RB2	OUT	Endstufe M3 Phase A
RB3	OUT	Endstufe M4 Phase A (MSB)
RB4	OUT	Endstufe M1 Phase B (LSB)
RB5	OUT	Endstufe M2 Phase B
RB6	OUT	Endstufe M3 Phase B
RB7	OUT IN	Endstufe M4 Phase B (MSB) gesteckter Jumper 1, Initialisierung der Busadresse -> sinnvollerweise wird bei gestecktem Jumper 1 die Busadresse **31hex** automatisch eingestellt

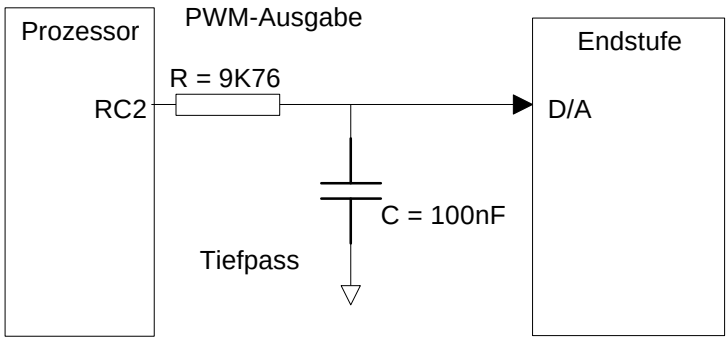
PortA		
Pin	Richtung	Verwendung
RA0	OUT	BRAKE der Endstufen
RA1	OUT	OUT_1 Ausgangstreiber, HIGH-SIDE-Schalter [1)]
RA2	IN	INPUT_1 [1)]
RA3	IN	INPUT_2 [1)]
RA4	IN	INPUT_3 [1)]
RA5	IN	INPUT_4 [1)]

[1)] siehe Digitale Ein-/Ausgabe

PortC		
Pin	Richtung	Verwendung
RC0	OUT	Direction Endstufe Phase A
RC1	OUT	Direction Endstufe Phase B
RC2	OUT	PWM-OUT Referenzspannung für DA-Wandler der Phasen A und B
RC3	OUT	Clock I^2C-Bus
RC4	OUT IN	Daten nach EEPROM I^2C -Bus Daten vom EEPROM I^2C -Bus
RC5	OUT	Treiberfreigabe RS485-BUS / DIN-Meßbus, LOW --> Treiberfreigabe
RC6	OUT	asynchrone Schnittstelle Transmit
RC7	IN	asynchrone Schnittstelle Receive

Die gewählte Prozessorquarzfrequenz beträgt 7.3728 MHz. Als Speichermedium für Parameter und interpretierbare Daten, wie z.B. Bewegungsziele, Verknüpfung von Eingängen usw. steht auf der Baugruppe ein E^2PROM vom Type 24LC04B mit einer Speicherkapazität von 4K-BIT zur Verfügung.

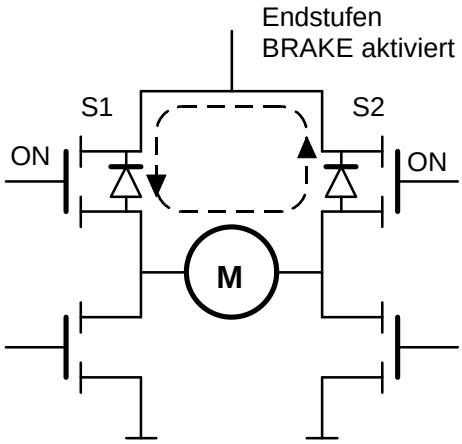
Zur Anpassung der Endstufe an verschiedene Motorströme ist über den am Prozessor verfügbaren PWM-Ausgang die Referenzspannung für den in der Endstufe integrierten DA-Wandler einstellbar. Die Realisierung des ausgangsseitigen Tiefpasses ist aus folgendem Bild ersichtlich.



Die PWM-Grundfrequenz sollte so gewählt werden, das sie größer/gleich 5 τ des angegebenen Tiefpasses ist. Mit dieser Dimensionierung wird eine annähernd rippelfreie Referenzspannung am DA-Wandler über den ganzen PWM-Bereich von 0x00 bis 0x0FF erzielt.

Für den Fall, daß die Hardwareeinheit im BUS-Modus betrieben wird, kann mittels gestecktem Jumper **J1** die Busadresse nach einem POWER-ON Reset auf Adresse 31hex erzwungen werden, d.h. nach dem Einschalten muß der Prozessorpin RB7 als Eingang programmiert sein um die Belegung des Jumpers J1 einlesen zu können. Mittels dieser 'Globaladresse' ist das Modul dann auf die gewünschte Busadresse konfigurierbar, die eingestellte Busadresse wird im E^2PROM abgespeichert, wo sie dann nach einem POWER-ON Reset zur Verfügung steht (Jumper **J1** wieder vorher entfernen).

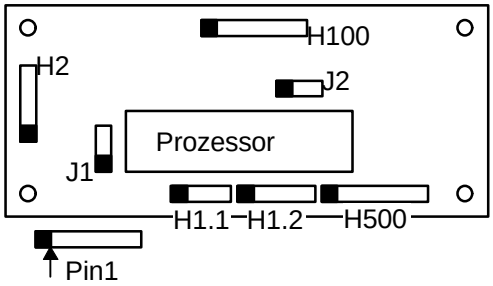
Der zur Verfügung stehende Schaltausgang OUT_1 (HIGH-SIDE-Schalter an RA1) ist nach einem RESET abgeschaltet. RA1 = LOW --> Ausgang eingeschaltet !!!



Die eingesetzten Endstufen verfügen über einen **BRAKE**-Eingang welcher zum definierten Sperren der Endstufe dient, BRAKE = HIGH --> Endstufe gesperrt. Konkret wird dies nach einem **RESET** des Prozessors verwendet um die Endstufen in einen definierten Anfangszustand zu bringen.. Für Anwendungen wo eine enablete Endstufe von Nachteil ist, wie zum Beispiel Meßanwendungen, ist die Verwendung der **BRAKE**-Funktion auch denkbar.

4.5. Steckerbelegung

Auf der Grundplatine sind 4 Stück einreihige Steckverbinder zum Anschluß der notwendigen Verbindungsleitungen vorgesehen. Im nachfolgenden Text werden die vorhandenen Steckverbinder beschrieben. Die Zielposition auf der Platine ist aus folgendem Bild ersichtlich:



Stecker H100		
Pin	Bedeutung	
1	Phase 1 Anfang	1A
2	Phase 2 Anfang	2A
3	Phase 1 Ende	1B
4	Phase 2 Ende	2B

Stecker H2		
Pin	Bedeutung	Richtung
1	+U_Motorversorgung 12V...45V	INPUT
2	GND (Masse) Motorversorgung	INPUT
3	GND (Masse) Proessorversorgung	INPUT
4	+U_Prozessorversorgung 10V...24V	INPUT

Die Massen der Prozessorversorgung und der Motorversorgung sind auf der Baugruppe zusammengeführt. Es ist denkbar eine Brücke zwischen H2-PIN1 und H2-PIN4 zu realisieren. Für diesen Fall beachten Sie die maximal zulässigen Versorgungsspannungen.

Stecker H1.1		
Pin	Bedeutung	Richtung
1	+24V I/O-Versorgung	OUTPUT
2	Anode Optokoppler (RA2)	INPUT
3	+24V I/O-Versorgung	OUTPUT
4	Anode Optokoppler (RA3)	INPUT

Stecker H1.2		
Pin	Bedeutung	Richtung
1	+24V I/O-Versorgung	OUTPUT
2	Anode Optokoppler (RA4)	INPUT
3	Anode Optokoppler (RA5)	INPUT
4	+24V I/O-Versorgung	OUTPUT
5	Leistungsschalter (RA1)	OUTPUT
6	GND (Masse) 24V I/O-Vers.	INPUT

Die Ein/ Ausgänge der Steckerverbinder H1.1 und H1.2 sind optoisoliert ausgeführt.

Stecker H500		
Pin	Bedeutung	Richtung
1	GND_ISO Busschnittstelle	OUTPUT
2	PIN-NEG. RS485 Treiber	OUTPUT
3	PIN-POS. RS485 Treiber	OUTPUT
4	PIN-POS. RS485 Empfänger	INPUT
5	PIN-NEG. RS485 Empfänger	INPUT
6	+5V_ISO Busschnittstelle, max 5mA !!	OUTPUT

Bitte beachten Sie, daß die auf dem Steckverbinder H500 zur Verfügung gestellte 5V Spannungsversorgung nicht kurzschlußfest ist. Ein Kurzschluß führt zur Zerstörung des auf der Grundplatine vorhandenen DC/DC-Wandlers !!!

Jumper J1: Jumper 1 ist für die Einstellung der DIN-MeßbusTeilnehmeradresse vorgesehen. Der Jumper darf nur für Initialisierungseinstellungen gesteckt werden. Alternativ ist die Rekonfigurierung der Einheit über Softwarepolling (nur ein Teilnehmer angesteckt) denkbar. In jedem Fall wird nach der Initialisierung der Grundparameter, wie Maximalstrom des angeschlossenen Schrittmotors, die Teilnehmeradresse im DIN-Meßbus vergeben und im E^2PROM abgespeichert. Nach einem Wiedereinschalten der Einheit ist diese dann unter der angegebenen Adresse erreichbar.

Jumper J2: Jumper 2 ist für den Anschluß eines Temperaturfühlers vorgesehen.
(Übertemperatur → Schalter des Temperaturfühlers geschlossen)

4.6. Grenzwerte des ***isel***-PICMIC-Moduls

Motorversorgungsspannung:	12V...45V DC, maximale Welligkeit 0.5V, >2A
Motorstrom:	maximal 2.8A, softwareseitig einstellbar
Prozessorversorgungsspannung:	10V...24V DC, maximale Welligkeit 0.2V, I_{MAX} ca. 300mA
24V-I/O-Versorgung:	20V...27V, 0.6A, maximale Welligkeit 0.2V
Ausgangstreiber 24V-I/O:	maximal 0.3A HIGH-SIDE
Eingänge 24V-I/O:	LOW <= 1V HIGH >= 19V Eingangsstrom Eingang bei 27V <= 6.6mA
maximale Schrittfrequenz:	12kHz Vollschritt
Datenübertragung:	9600 BAUD RS485/422 gemäß DIN-Meßbus

4.7. Kühlung des ***isel***-PICMIC-Moduls

Damit ein kontinuierlicher Betrieb des Schrittmotormoduls bzw. der OEM-Baugruppe **xMM4403** möglich ist, muß auf eine ausreichende Kühlung geachtet werden. Um einen Betrieb mit maximalen Parametern bei einer Umgebungstemperatur von 313 KELVIN zu ermöglichen, muß ein Kühlkörper mit einem Wärmewiderstand von 2.0 K/W zum Einsatz kommen. Weitere Möglichkeiten zur Kühlung bestehen in einer aktiven Kühlung mit einem entsprechenden Lüfter und/oder das flächige Anflanschen an eine Mechanik..

Die Reduzierung der Verlustleistung ist auch durch die Nutzung der programmierbaren Stromabsenkung im Stillstand möglich (siehe 3.3.3. Sollstrom und Stromabsenkung einstellen).

4.8. Verbindungskabel

Verbindungsleitung Schnittstellenkonverter SubD25-Buchse nach SubD15 DIN-Meßbus:

SubD-25 Schnittstellenkonverter	SubD15-Stecker	TMO4403
16	11	4 Empfänger positiv
17	4	5 Empfänger negiert
18	9	3 Treiber positiv
19	2	2 Treiber negiert
7	8	1 ISO_GND

Weiterhin müssen folgende Verbindungen realisiert werden, um bei abgeschalteten Teilnehmern (Teilnehmer hochohmig) auf der Seite der Leitstation einen definierten Pegel zur Verfügung zu stellen.

Pin2 SubD25 Brücke nach Pin18 SubD25
(510Ohm Widerstand +5V-ISO -> Empfänger Pegelumsetzer, intern im Pegelumsetzer)

Pin18 SubD25 Widerstand 150Ohm nach PIN19 SubD25
Pin19 SubD25 Widerstand 510Ohm nach GND_ISO
Pin16 SubD25 Widerstand 120 .. 220Ohm nach Pin17 SubD25

Alle Verbindungen sind direkt am Schnittstellenkonverter realisiert !

Am letzten Teilnehmer in der Busstruktur muß ein Busabschluß erfolgen. Fertig vorbereitete Baugruppen sind bei *isel*-automation erhältlich. Eine manuelle Realisierung erfolgt wie beschrieben :

Widerstand 510Ohm	von ISO_5V (Pin6 am letzten TMM4403-Modul)	nach Pin9-SubD15-Buchse
Widerstand 150Ohm	von Pin9-SubD15-Buchse	nach Pin2-SubD15-Buchse
Widerstand 510Ohm	von Pin2-SubD15-Buchse	nach Pin8-SubD15-Buchse (Pin1 des TMM4403)
Widerstand 120 .. 220Ohm	von Pin4-SubD15-Buchse	nach Pin11-SubD15-Buchse

Verbindungsleitung SubD15 DIN-Meßbus Teilnehmer <-> Teilnehmer:

SubD15	SubD15
1	1
9	9
2	2
4	4
11	11
8	8
15	15

Die oben beschriebene Verbindungsleitung ist als Buchse - Steckervariante (abweichend von der DIN-66348) ausgeführt.

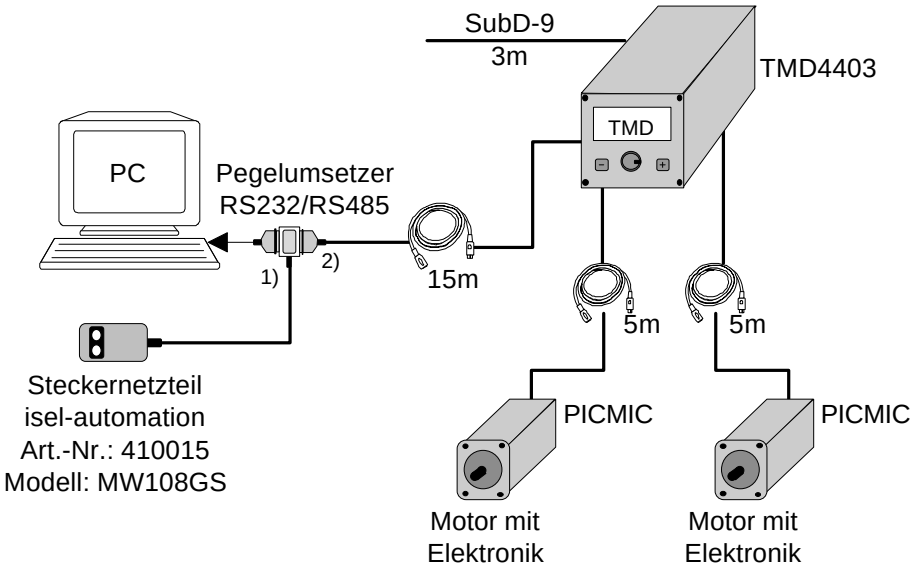
Die Schirme der Busleitung sind auf kürzestem Weg mit dem metallischen SUBD-Gehäuse zu verbinden (siehe auch 4.9. EMV-Verträglichkeit).

Die genormte Busbelegung sowie weiterführende Darlegungen sind in der DIN66348 - Teil-2 dargelegt.

4.9. EMV-Verträglichkeit

Die vorgestellte ***isel***-PICMIC-Baugruppe ist für den Einsatz im industriellen Umfeld, im Wohnbereich, in Klein-, Geschäfts- und Gewerbebetrieben vorgesehen. Die EMV-Messungen erfolgten gemäß **EN55011**-Grenzwertklasse B und **EN50082-2**.

Die EMV-Messung erfolgte in folgender Gerätekombination:



[1] Zur Einhaltung der geforderten Grenzwerte für abgestrahlte Störungen ist die in den Pegelumsetzer eingeführte Spannungsversorgung mit einem Ferrit AL=2700nH/Wdg² -> 3 Windungen zu beschalten.

[2] Weiterhin ist für die Einhaltung der Störfestigkeit nach EN50082-2 auf das vom Pegelumsetzer abgehende Buskabel ein Ferrit mit AL=3500nH/Wdg² aufzusetzen.

Zur Entstörung der leitungsgebundenen Störungen wurde im Gerät TMD4403 ein Filter der Firma Konfektronik Type HGN 339-2 eingesetzt.

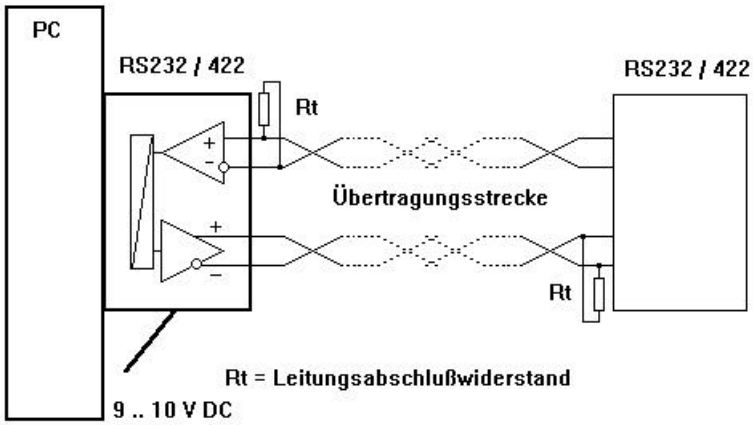
Die eingesetzten Kabel vom Pegelumsetzer nach TMD4403 und von TMD4403 zu den Motoren mit integrierter Elektronik sind geschirmt ausgeführt. Als Buskabel wurde das LAPP_Kabel UNITRONIC-BUS LD 3*2*0.22 eingesetzt.

Die oben aufgezeigte Beschaltungsvariante ist bei ***isel***-automation erhältlich. Sollten Sie für Ihren Einsatzfall eine abweichende Konfiguration wählen, dann müssen Sie in jedem Fall durch entsprechend durchzuführende EMV-Messungen die Konformität Ihrer Konfiguration nachweisen.

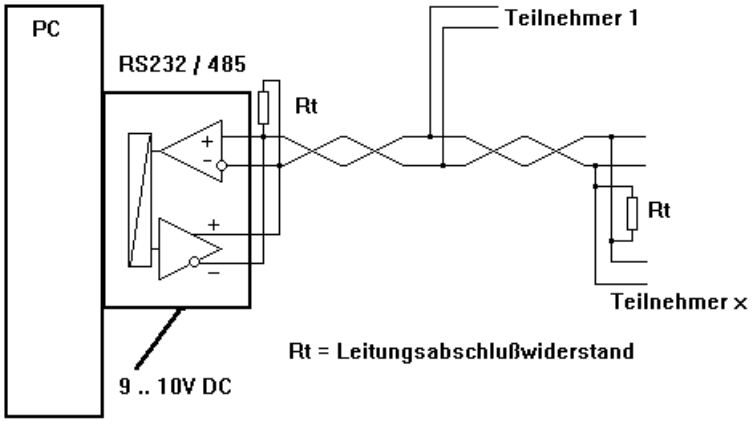
5. Schnittstellenkonverter BEC485 mit galvanischer Trennung

Der Schnittstellenumsetzer BEC485 dient dem Umsetzen der physikalischen Schnittstelle RS232 (V24), welche an jedem Personal-Computer vorhanden ist, nach der physikalischen Schnittstelle RS422/485. So ist es möglich die maximale Entfernung zwischen zwei Teilnehmern von ca. 4 Metern, welche für die Verwendung der RS232 Schnittstelle angegeben wird, auf weit über 500 Meter, je nach Datenübertragungsrate, auszudehnen. Durch die realisierte galvanische Trennung werden Potentialverschleifungen zwischen Systemen ausgeschlossen.

Bei Verwendung als Schnittstellenwandler zur RS422 Schnittstelle dient die vorgestellte Baugruppe der Überbrückung von weiten Distanzen, um wie oben erwähnt, die Nachteile der RS232 (V24) Schnittstelle zu kompensieren. Der Sendetreiber ist in dieser Betriebsart ständig freigegeben. Für diesen Zweck ist das CTS-Signal aktiv zu schalten (+ 4 ... 12V auf RS232 Seite)

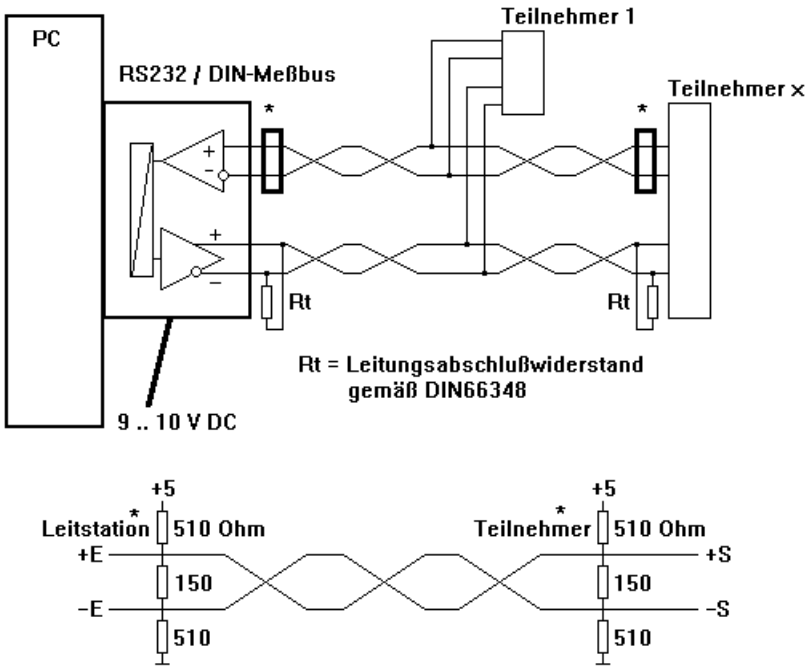


Durch eine entsprechende äußere Beschaltung ist das Modul BEC485 bedingt für den Betrieb an einer Standard-RS485 konfigurierbar. In diesem Fall sind bis zu 31 Teilnehmer an den Pegelumsetzer ansteckbar. Da diese Schnittstelle als 2-Drahtschnittstelle definiert ist, ist nur ein halbduplex Betrieb der Schnittstelle möglich. Der Treiber der Sendeleitung muß in diesem Fall mit der CTS Leitung im Falle eines Sendens aktiviert werden (Treiber Enable -> 4 ... +12V auf RS232 Seite). Der Empfänger ist ständig aktiviert !!. Die Pin's 16<->18 sowie 17<->19 sind in diesem Fall miteinander zu verbinden.

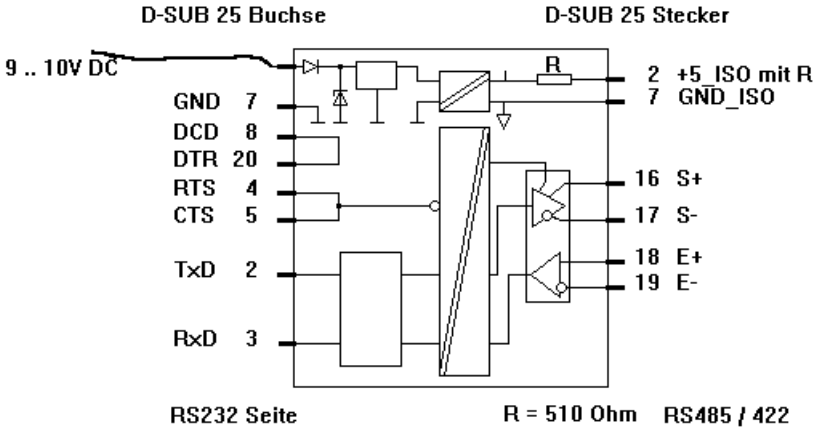


Bei Verwendung des Schnittstellenmoduls als Schnittstellenwandler zur DIN-Meßbus-Schnittstelle ist die Verwendung im 'erweiterten' RS485 Betrieb möglich. In diesem Fall sind bis zu 31 Teilnehmer an den Pegelumsetzer ansteckbar. Da diese Schnittstelle als 4 Drahtschnittstelle definiert ist, ist ein duplex Betrieb der Schnittstelle möglich. Der Treiber der Sendeleitung muß in diesem Fall mit der CTS Leitung im Falle eines Sendens aktiviert werden (Treiber Enable -> 4 ... +12V auf RS232 Seite). Der Empfänger ist ständig aktiviert. Gemäß der DIN-66348 Teil 2 muß die Sendeleitung der Leitstation auf der Leitstationseite und der am weitesten entfernten Empfängerseite mit einem Widerstand von 150

OHM abgeschlossen werden (siehe Bild). Die Empfangsleitung der Leitstation ist mit einer Widerstandskombination 510 OHM, 150 OHM, 510 OHM (siehe Bild) abzuschließen.



Im nachfolgenden Bild ist der prinzipelle Aufbau des BMC485 dargestellt.



Die Verbindung zwischen den angeschlossenen Teilnehmern sollte mit einem abgeschirmten Kabel mit paarig verdrillten Signalleitungen erfolgen. Auf diese Art ist eine Leitung mit einem definierten Wellenwiderstand realisierbar. Die Leitungsenden sollten gemäß der Anwendung mit einem Abschlußwiderstand terminiert werden.

Die erforderlichen EMV-Tests wurden in Verbindung mit der ISEL-Steuerung TMD4403 durchgeführt.Es wurden geprüft: EN55011-Grenzwert B und EN50082-2. Zur Einhaltung der Störfestigkeit ist es notwendig auf das abgehende Buskabel Type : LAPP-UNITRONIC LD 3*2*0.22 einen Ferrit mit einem AL-Wert von AL=3500 nH/WDG^2 auzusetzen.

Anschlußbelegung BEC485:

SubD25 - Buchse --> RS232 Seite		
Pin	Richtung	Bedeutung
2	IN	TxD Sendedaten RS232 vom PC
3	OUT	RxD Empfangsdaten RS232 zum PC
4	OUT	CTS
5	IN	RTS
7	IN	GND
8	OUT	DCD
20	IN	DTR

SubD25 - Stecker --> RS422/RS485 Seite		
Pin	Richtung	Bedeutung
1	IN	frei, Schirm, Schirm auf Gehäuse
2	OUT	+5V über =510 Ohm
7	OUT	GND_ISO
16	OUT	S+ Sender Ausgang
17	OUT	S- Sender Ausgang
18	IN	E+ Empfänger Eingang
19	IN	E- Empfänger Eingang

Versorgungsspannung Eingang (fest angeschlossenes Netzteil)		
Farbe	Richtung	Bedeutung
rot	IN	DC 9 .. 10 V +
blau	IN	DC GND zu rot

Das fest angeschlossene Netzteil ist auf die erforderlichen Spannungen eingestellt.

6. Anschluß an eine Fremd-RS485

6.1. Belegung der RS485 Treiber / Empfänger

Baugruppe *isel*-Pegelwandler RS232 – RS485		
SUBD-25	MAX489-PIN	Funktion
16	9	Treiberausgang nicht invertierend
17	10	Treiberausgang invertierend
18	12	Empfängereingang nicht invertierend
19	11	Empfängereingang invertierend

Baugruppe *isel*-PICMIC OEM-Baugruppe		
Header	MAX489-PIN	Funktion
2	10	Treiberausgang invertierend
3	9	Treiberausgang nicht invertierend
4	12	Empfängereingang nicht invertierend
5	11	Empfängereingang invertierend

Belegung SUBD-15 Stecker *isel*-PICMIC-Modul		
SUBD15-PIN	***isel***-PICMIC-OEM	Header
2	2	Treiberausgang invertierend
4	5	Empfängereingang invertierend
9	3	Treiberausgang nicht invertierend
11	4	Empfängereingang nicht invertierend

6.2. Beispielbeschaltung für eine handelsübliche RS485-PC-Einsteckkarte

Als Beispiel wurde eine RS485-PC-Einsteckkarte (Hersteller KDM-Ingenieurbüro) verwendet. Zum Anschluß wurde der SUBD-25 Steckverbinder des Pegelwandlers ersetzt.

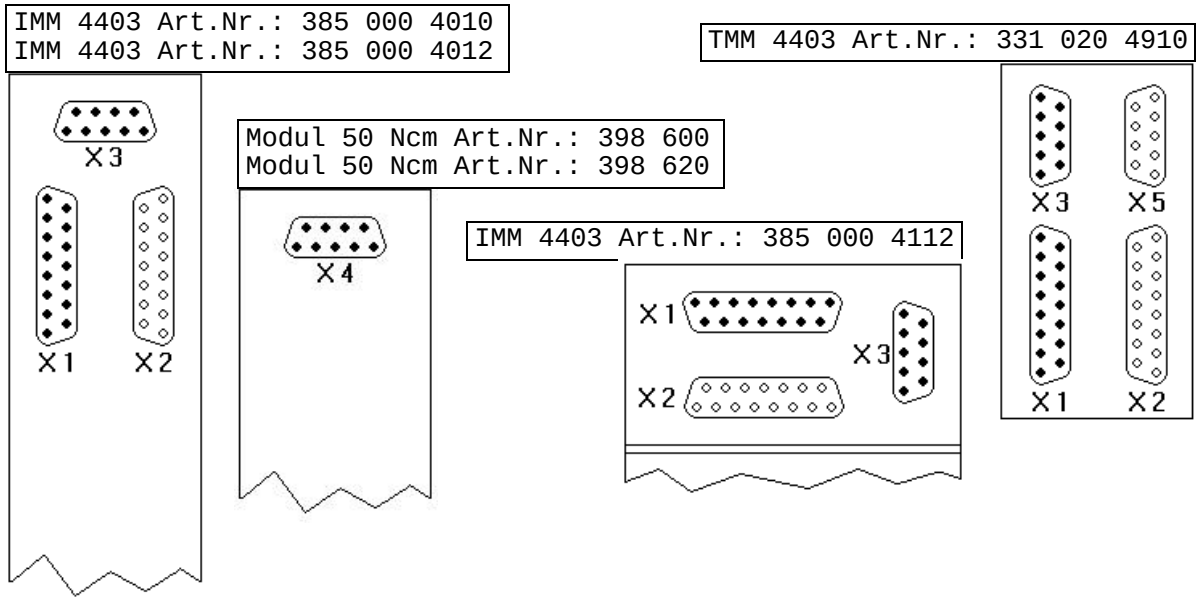
Eine Anpassung an den SUBD-37 Steckverbinder der PC-Einsteckkarte wurde wie folgt realisiert:

SUBD15 – ***isel***-DIN-Meßbus	SUBD37- PC-Karte
2	PIN7 (RS485-SIN+)
4	PIN28 (RS485-SOUT+)
9	PIN8 (RS485-SIN-)
11	PIN29 (RS485-SOUT-)
	Brücke PIN9 – PIN26
8	PIN30 (Masse isoliert extern)
	Brücke PIN10 – PIN27 (Treiber RS485 freigeschaltet !)

- Zwischen PIN 2-SUBD15 und GND-extern wurde ein Widerstand von 510 Ohm geschaltet
- Zwischen PIN 9-SUBD15 und +5V-extern wurde ein Widerstand von 510 Ohm geschaltet
- Zwischen PIN 2-SUBD15 und PIN9-SUBD15 wurde ein Widerstand von 120 Ohm geschaltet

7. Anschlußbelegungen der Modulreihe PICMIC

Module der PICMIC-Modulreihe im Überblick:



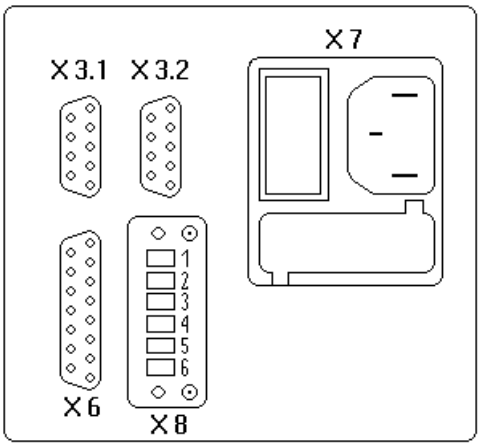
X1 DIN-Meßbus Eingang SubD15 - Stecker		
Pin	Richtung	Bedeutung
2	OUT	S- Sender RS485 Ausgang
4	IN	E- Empfänger RS485 Eingang
8	IN	GND
9	OUT	S+ Sender RS485 Ausgang
11	IN	E+ Empfänger RS485 Eingang

X2 DIN-Meßbus Ausgang SubD15 - Buchse		
Pin	Richtung	Bedeutung
2	OUT	S- Sender RS485 Ausgang
4	IN	E- Empfänger RS485 Eingang
8	OUT	GND
9	OUT	S+ Sender RS485 Ausgang
11	IN	E+ Empfänger RS485 Eingang
15	OUT	+5V, max. 5mA

X3 Spannungsversorgung Eingang SubD9 - Stecker		
Pin	Richtung	Bedeutung
1	IN	+VS 40V, Motor
2	IN	Input 4
3	IN	Input 3
4	IN	GND 10V, Prozessor
5	IN	+VS 24V, I/O
6	IN	GND 40V, Motor
7	OUT	Output 1
8	IN	+VS 10V, Prozessor
9	IN	GND 24V, I/O

X4 Motor Eingang, SubD9 Stecker X5 Motor Ausgang, SubD9 Buchse		
Pin	Richtung X4/X5	Bedeutung
1	IN/OUT	Motor, Phase 2b
2	IN/OUT	Motor, Phase 2a
3	IN/OUT	Motor, Phase 1b
4	IN/OUT	Motor, Phase 1a
5	IN/OUT	24V, Versorgung für Schalter
7	OUT/IN	Endschalter (Input2)
9	OUT/IN	Referenzschalter (Input1)

Netzteil für 2 PICMIC-Module:



X7 Netzeingangsmodul 230 Volt

X8 Sicherheitskreis (optional)

X6 I/O Signale SubD15 - Buchse		
Pin	Richtung	Bedeutung
1	IN	Input4, Achse1
2	OUT	+24V, I/O
3	IN	Input4, Achse2
4	OUT	+24V, I/O
5	IN	Input3, Achse1
6	OUT	+24V, I/O
7	IN	Input3, Achse2
8	OUT	+24V, I/O
9		nicht benutzt
10	OUT	Output1, Achse2
11	IN	GND
12		nicht benutzt
13	IN	GND
14	OUT	Output1, Achse1
15	IN	GND

X3.1 Spannungsversorgung Ausgang, Achse1 SubD9 - Stecker		
Pin	Richtung	Bedeutung
1	OUT	+VS 40V, Motor
2	OUT	Input 4
3	OUT	Input 3
4	OUT	GND 10V, Prozessor
5	OUT	+VS 24V, I/O
6	OUT	GND 40V, Motor
7	IN	Output 1
8	OUT	+VS 10V, Prozessor
9	OUT	GND 24V, I/O

X3.2 Spannungsversorgung Ausgang, Achse2 SubD9 - Stecker		
Pin	Richtung	Bedeutung
1	OUT	+VS 40V, Motor
2	OUT	Input 4
3	OUT	Input 3
4	OUT	GND 10V, Prozessor
5	OUT	+VS 24V, I/O
6	OUT	GND 40V, Motor
7	IN	Output 1
8	OUT	+VS 10V, Prozessor
9	OUT	GND 24V, I/O

Das Netzteil kann optional mit Eingängen zur Realisierung eines Sicherheitskreises ausgestattet werden. Hierfür sind folgende Eingänge vorgesehen:

X8 - Pin1 <--> Pin2 Taster „Aus“ (Öffner)
X8 - Pin3 <--> Pin4 „Not-Aus“ (Öffner)
X8 - Pin5 <--> Pin6 Taster „Ein“ (Schließer)

Stichwortindex

P

Q

R

S

T

Ü

V

Z

Wichtiger Hinweis!

Lesen Sie vor dem Öffnen der Verpackung bitte die Bestimmungen des umseitigen Lizenzvertrags genau durch.
Falls Sie mit dem Lizenzvertrag oder mit Teilen des Lizenzvertrags nicht einverstanden sind, so senden Sie das Produkt gegen Erstattung des an ***isel****automation* KG entrichteten Kaufpreises in ungeöffnetem Zustand zurück.
Mit dem Öffnen der Verpackung erklären Sie sich mit den Bedingungen des Lizenzvertrags einverstanden.

Lizenzvertrag

1 Einräumung einer Lizenz

isel *automation* KG gewährt Ihnen das Recht, Kopien des beiliegenden Programmes bzw. Programmpakets (die „Software“) auf einem oder mehreren Computern zu benutzen - solange die Nutzung im Eigengebrauch stattfindet. Die Software wird benutzt, wenn sie in den temporären Speicher (RAM) oder in einem permanenten Speicher (Festplatte, Streamer, Diskette, CD-ROM, ...) installiert wird.

2 Erweiterung der Lizenz

Falls die Software ein Sprachprodukt ist (z. B. Interpreter zur freien Programmierung), so haben Sie das Recht, die für den Betrieb der NC-Steuerdateien erforderlichen Runtime-Module der Software zu vervielfältigen und zu verbreiten, unter folgenden Voraussetzungen:

a) Sie vertreiben die Runtime-Module nur als Teil Ihres Software-Produktes und als Teil desselben.
b) Sie verwenden bei der Vermarktung Ihres Produkts weder den Namen noch das Logo noch andere Kennzeichen von ***isel*** *automation* KG.
c) Sie nehmen die Copyright-Meldung von ***isel*** *automation* KG für die Software als Teil der Bereitschaftsmeldung in Ihr Produkt auf. Falls Sie Runtime-Module von ***isel*** *automation* KG benutzen, so dürfen deren Copyright-Meldungen weder entfernt noch modifiziert werden.
d) Sie erklären sich einverstanden, ***isel*** *automation* KG bezüglich aller Ansprüche oder Rechtsstreitigkeiten, die aufgrund des Gebrauchs oder der Verbreitung Ihres Produkts entstehen können, freizustellen, schadlos zu halten und gegen solche Ansprüche zu verteidigen.

Runtime-Module sind die Dateien der Software, für die in den schriftlichen Unterlagen ausdrücklich angegeben ist, dass sie erforderlich sind für:

- den Betrieb der Hardware
- den Ablauf Ihres NC-Steuerungsprogramms

Die Runtime-Module sind auf die speicherresidenten Steuerungsprogramme (Treiber) für die Maschinensteuerung, die Objekt-Dateien für die Bahndatenerstellung und Interpreter-Module für NC-Quelldateien, deren Gebrauch und Weitergabe ausdrücklich gestattet ist, beschränkt.

3 Urheberrecht

Die Software ist Eigentum von ***isel*** *automation* KG und durch Urheberrechtsgesetze und nationale Rechtsvorschriften gegen Kopieren geschützt. Für den Eigengebrauch und für Sicherungs- und Archivierungszwecke dürfen Sie Kopien der Software anfertigen. Weder die Software noch die Handbücher oder Teile davon dürfen kopiert oder in anderer Form vervielfältigt und an Dritte weitergegeben oder Dritten in anderer Form zugänglich gemacht werden. Zurückentwicklung (Reverse Engineering), Dekompilieren und Disassemblieren der Software durch Sie oder durch Dritte ist nicht gestattet.

4 Einschränkung der Haftung und Ansprüche des Kunden

isel *automation* KG ist nicht für Schäden (uneingeschränkt eingeschlossen sind Schäden aus entgangenem Gewinn, Datenverlust, Betriebsunterbrechungen oder aus anderem finanziellem Verlust) ersatzpflichtig, die aufgrund der Benutzung der Software oder der fehlerhaften Benutzung der Software entstehen. In jedem Fall ist die Haftung von ***isel*** *automation* KG auf den Betrag beschränkt, den Sie an ***isel*** *automation* KG für die Software bezahlt haben.

Die gesamte Haftung von ***isel*** *automation* KG und Ihr alleiniger Anspruch besteht nach Wahl von ***isel*** *automation* KG entweder in der Rückerstattung des bezahlten Preises oder in der Reparatur oder dem Ersatz der Software oder der Hardware. Dies gilt nicht, wenn der Ausfall der Software auf fehlerhafte Anwendung oder Unfall zurückzuführen ist.