ROHDE&SCHWARZ

Geschäftsbereich
Kommunikationstechnik

Bedienung der Software

IEEE488-Bus-Steuerung

für RDS-CODEC DMC01

DMC01-S

2059.9007

Printed in the Federal
Republic of Germany

2059.9188.11-01 - 1

Inhalt

Anhang A

Anhang B

1 Einführung

1.1 Geräteausführung

Das Gerät 2059.9007 ist gegenüber den bisherigen Ausführungen um den IEC/IEEE488-Anschluß an der Geräterückseite erweitert (siehe Bild 1). Zusätzlich ist das Bedienmenü (siehe Abschnitt 2) um IEC/IEEE488-Einstellungen ergänzt. Eine Kassettenausführung ist nicht lieferbar.

Alle Bedienerschnittstellen (serielle Schnittstellen an Frontplatte und Geräterückseite und IEEE488-Schnittstelle) sind gleichberechtigt und mit den jeweils zugehörigen Protokollen bedienbar. Da auch bei Bedienung über den IEEE488-Bus davon keine Ausnahme gemacht wird, ist die bei diesem übliche LOCAL/REMOTE-Umschaltung nicht vorhanden.

1.2 Prinzipien

Grundzustand

Nach dem Einschalten oder nach Betätigung der Reset-Taste ist das Gerät sowohl an der Frontplatte als auch über die seriellen Schnittstellen und dem IEEE488-Bus mit den jeweiligen Protokollen bedienbar.

Fehlerbehandlung

Alle Fehler, welche das Gerät in Zusammenhang mit der Bedienung des IEC/IEEE-Bus erkennt, werden durch Setzen eines Bits (Bit 2, 4 oder 5) im Event-Status-Register angezeigt. Diese Bits bleiben erhalten, bis das Register ausgelesen oder durch die Befehle *RST oder *CLS gelöscht wird. Dies entspricht der Norm IEEE-488.2.

Bevorrechtigung

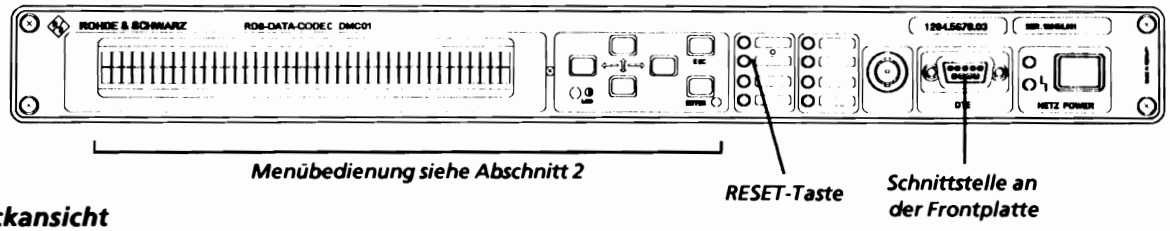
Das Gerät unterstützt eine Bevorrechtigungsmethode (BV) für die Schnittstellen, die im Betriebshandbuch DMC01, im Teil

Bedienung Coder - Leitungsprotokoll

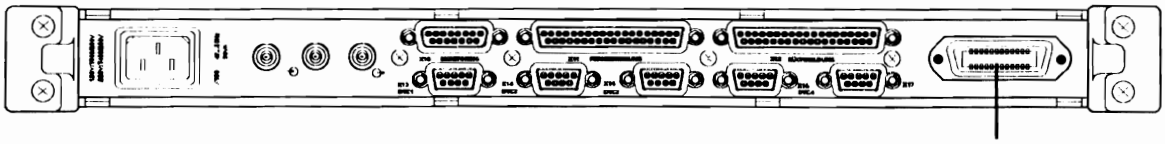
beschrieben ist.

Beachte:
Die IEC/IEEE-Schnittstelle verwendet dieselben Bevorrechtigungs-Einstellungen wie die serielle Schnittstelle an der Frontplatte.

Frontansicht



Rückansicht



Bilder zeigen ein Gerät der Bauweise 90

***Bild 1** Ansicht des Einschubs DMC01*

2 Bedienung an der Frontplatte

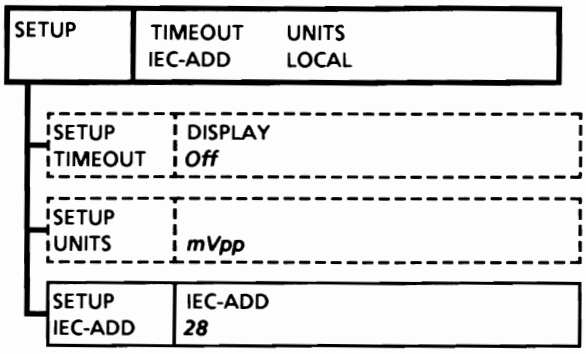
2.1 Übersicht

Nach dem Einschalten des Gerätes erscheint die Anzeige wichtiger, aktueller Daten. Mit Taste ENTER wird dann zur Gerätebedienung in das Eröffnungsmenü geschaltet.

Die im Betriebshandbuch zum DMC01 beschriebenen Menüs und Untermenüs sind weiterhin vorhanden.

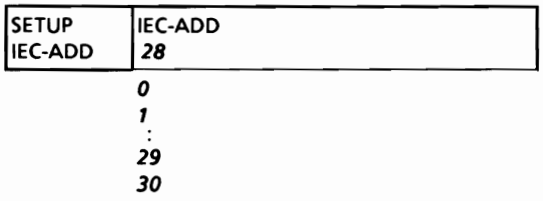
Das SETUP-Menü ist mit dem Untermenü IEC-ADD ergänzt, damit die IEC/IEEE-Bus-Adresse eingestellt werden kann. .

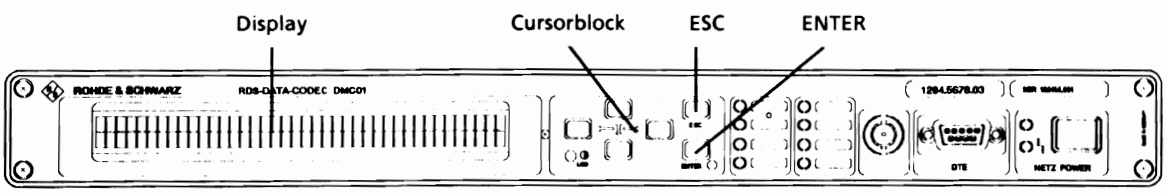
2.2 Menü "Setup"



Untermenü SETUP / IEC-ADD

Die Adresse ist ab Werk auf 28 eingestellt. Sie läßt sich verändern, wenn der Bus gerade nicht aktiv ist. Während der Abarbeitung von Programmen durch den angeschlossenen Controller ist keine Veränderung der Adresse möglich; es wird nur die aktuell eingestellte Adresse angezeigt.





3 Bedienung über IEEE488-Bus

3.1 Einführung

Das Gerät (DMC01 mit Bestell-Nr. 2059.9007...) ist serienmäßig mit einer IEC-Bus-Schnittstelle nach Norm IEC 625.1/IEEE 488.2 ausgerüstet. Die Anschlußbuchse befindet sich auf der Geräterückseite. Über sie kann ein Steuerrechner zur Fernbedienung angeschlossen werden.

Dieses Kapitel setzt Grundkenntnisse in der IEC/IEEE-Bus-Programmierung und der Bedienung des Steuerrechners voraus. Eine Beschreibung der Schnittstellenbefehle ist den entsprechenden Handbüchern des angeschlossenen IEEE-Bus-Controllers bzw. dessen IEEE-Bus-Softwarebeschreibung zu entnehmen.

Die Anforderungen zur Befehlssyntax, Fehlerbehandlung und Gestaltung der Status-Register werden ausführlich in den jeweiligen Abschnitten erläutert. Tabellen ermöglichen einen schnellen Überblick über die im Gerät realisierten Befehle und die Belegung der Bits in den Status-Registern. Die Tabellen werden durch eine umfassende Beschreibung jedes Befehls und der Status-Register ergänzt. Alle Programmbeispiele sind in QuickBASIC verfaßt.

3.2 Kurzanleitung

Die folgende kurze und einfache Bediensequenz erlaubt es, das Gerät schnell in Betrieb zu nehmen und seine Grundfunktionen einzustellen. Es wird vorausgesetzt, daß die IEC-Bus-Adresse, die werkseitig auf 28 eingestellt ist, noch nicht verändert wurde.

1. Gerät und Controller mit IEC-Bus-Kabel verbinden.
2. Am Controller folgendes QuickBASIC-Programm erstellen und starten:

```
CALL IBFIND("DEV1", generator%)        Kanal zum Gerät öffnen
CALL IBPAD(generator%, 28)             Geräteadresse dem Controller mitteilen
CALL IBWRT(generator%, "*RST;*CLS")        Gerät rücksetzen
```

3. Ein- und Ausgabe der RDS-Daten:

   Dafür gibt es zwei Befehle, einen für die Eingaben und einen für Abfragen. Im Datenteil der Befehle können alle im Gerät implementierten UECP/EBU-Befehle übertragenwerden, die dann das Gerät ausführt. UECP/Diese EBU-Befehle sind im Betriebshandbuch DMC01, Teil
   "Bedienung Coder - Universal Encoder Protokoll"
   erläutert.
   Dazu gehören folgende Dokumente:
   Universal Encoder Communication Protocol for Coder (Übersicht der Standard-Kommandos)
   Universal Encoder communication protocol (Draft SPB490 der European Broadcasting Union),
   Request protocol (Abfrage-Kommandos) und
   Manufacturer's specific protocol (Herstellerspezifische Kommandos).

   Eingaben:
   EBU #<*EBU-Kommando*>

   Abfragen:
   EBU? #<*EBU-Kommando*>

   Antwortformat:
   <*Hex-Zeichen des EBU-Kommando*> <*zugehörige Daten*>

Siehe Abschnitt 3.6.3 dieser Beschreibung.

3.3 Einstellen der Geräteadresse

Die IEC-Bus-Adresse des Gerätes ist werkseitig auf 28 eingestellt. Sie kann manuell im Menü eingestellt werden. Es sind die Adressen 0...30 erlaubt.

Manuelle Einstellung der Adresse (siehe Menübedienung in Abschnitt 2):

- Menü aufrufen (Cursor left/right und Taste [ENTER])
- Gewünschte Adresse eingeben (Cursor up/down)
- Eingabe mit Taste [ENTER] abschließen
  (oder mit Taste [ESC] verlassen ohne Änderung der Adresse)

3.4 IEC/IEEE-Bus-Nachrichten

Die Nachrichten, die auf den Datenleitungen des IEC-Bus (siehe Anhang A) übertragen werden, lassen sich in zwei Gruppen einteilen:

- **Schnittstellennachrichten** (Universalbefehle und adressierte Befehle) und
- **Gerätenachrichten** (Befehle und Geräteantworten).

3.4.1 Schnittstellennachrichten

Schnittstellennachrichten werden auf den Datenleitungen des IEC-Bus übertragen, wobei die Steuerleitung "ATN" aktiv ist. Sie dienen der Kommunikation zwischen Steuerrechner und Gerät und können nur von einem Steuerrechner, der die Controllerfunktion am IEC-Bus hat , gesendet werden.

Schnittstellenbefehle lassen sich weiter unterteilen, in

- **Universalbefehle** und
- **adressierte Befehle.**

Universalbefehle wirken ohne vorherige Adressierung auf alle am IEC-Bus angeschlossenen Geräte, adressierte Befehle nur an vorher als Hörer (Listener) adressierte Geräte. Die für das Gerät relevanten Schnittstellennachrichten sind im Anhang A aufgelistet.

3.4.2 Gerätenachrichten (Befehle und Geräteantworten)

Gerätenachrichten werden auf den Datenleitungen des IEC-Bus übertragen, wobei die Steuerleitung "ATN" nicht aktiv ist. Es wird der ASCII-Code verwendet. Gerätenachrichten werden nach der Richtung, in der sie am IEC-Bus gesendet werden, unterschieden:

- **Befehle** sind Nachrichten, die der Controller an das Gerät schickt. Sie bedienen die Gerätefunktionen und fordern Informationen an.
  Die Befehle werden wiederum nach zwei Kriterien unterteilt:

  1. Nach der Wirkung, die sie auf das Gerät ausüben:

  Einstellbefehle — lösen Geräteeinstellungen aus, z.B. Rücksetzen des Gerätes oder Setzen des Ausgangspegels auf 1 Volt.

  Abfragebefehle (Queries) — bewirken das Bereitstellen von Daten für eine Ausgabe am IEC-Bus, z.B. für die Geräteidentifikation oder die Abfrage einer Geräteeinstellung.

  2. Nach ihrer Festlegung in der Norm IEEE 488.2:

  Allgemeine Befehle (Common Commands) — sind in ihrer Funktion und Schreibweise in der Norm IEEE488.2 genau festgelegt. Sie betreffen

Funktionen, wie z.B. die Verwaltung der genormten Statusregister, Rücksetzen und Selbsttest.

Gerätespezifische Befehle — betreffen Funktionen, die von den Geräteeigenschaften abhängen, wie z.B. Frequenzeinstellung.

- **Geräteantworten** sind Nachrichten, die das Gerät nach einem Abfragebefehl zum Controller sendet. Sie können Meßergebnisse, Geräteeinstellungen oder Information über den Gerätestatus enthalten (siehe Abschnitt 3.5.4).

In Abschnitt 3.5 werden Aufbau und Syntax der Gerätenachrichten beschrieben. In Abschnitt 3.6 sind die Befehle aufgelistet und ausführlich erläutert.

3.5 Aufbau und Syntax der Gerätenachrichten

3.5.1 Aufbau eines Befehls

Die Befehle bestehen aus einem sogenannten Header und meist einem oder mehreren Parametern. Header und Parameter sind durch einen "White Space" (ASCII-Code 0..9, 11...32 dezimal, z.B. Leerzeichen) getrennt. Die Header können aus mehreren Schlüsselwörtern zusammengesetzt sein. Abfragebefehle werden gebildet, indem an den Header direkt ein Fragezeichen angehängt wird.

Common Commands — Geräteunabhängige Befehle bestehen aus einem Header, dem ein Stern "*" vorausgestellt ist, und eventuell einem oder mehreren Parametern.

Beispiele:

`*RST`	RESET, setzt das Gerät zurück
`*ESE 253`	EVENT STATUS ENABLE, setzt die Bits des Event Status Enable Registers
`*ESR?`	EVENT STATUS QUERY, fragt den Inhalt des Event-Status-Registers ab.

Gerätespezifischer Befehl — Das Gerät verfügt über einen gerätespezifischen Befehl der zur Übertragung von EBU-Kommandos dient (genaue Beschreibung im Handbuch Draft SPB 490, Universal encoder communication protocol, European Broadcasting Union).

Beispiele:
```
EBU #19xxxxxxxxx
EBU? #211xxxxxxxxxxx
```
x steht für jeweils ein Byte eines EBU-Befehls.

3.5.2 Aufbau einer Befehlszeile

Eine Befehlszeile kann einen oder mehrere Befehle enthalten. Sie wird durch ein <New Line>, ein <New Line> mit EOI oder ein EOI zusammen mit dem letzten Datenbyte abgeschlossen. QuickBASIC erzeugt automatisch ein EOI zusammen mit dem letzten Datenbyte.

Mehrere Befehle in einer Befehlszeile sind durch einen Strichpunkt ";" getrennt.

Beispiel:
```
CALL IBWRT(generator%, "*rst;*sre 16;*tst?")
```
Diese Befehlszeile beinhaltet drei Befehle. Der erste Befehl versetzt das Gerät in einen definierten Anfangszustand, der zweite setzt das Service Request enable Register auf 16 (Service Request wird ausgelöst, wenn Geräteantwort verfügbar), der dritte beinhaltet eine Selbsttestabfrage.

3.5.3 Antworten auf Abfragebefehle

Zu jedem Einstellbefehl ist, falls nicht ausdrücklich anders festgelegt, ein Abfragebefehl definiert. Er wird gebildet, indem an den zugehörigen Einstellbefehl ein Fragezeichen angehängt wird.

Der geforderte Parameter wird ohne Header gesendet.
Beispiel: `*STB?` Antwort: `4`

3.5.4 Parameter

Die meisten Befehle verlangen die Angabe eines Parameters. Die Parameter müssen durch einen "White Space" vom Header getrennt werden. Als Parametertypen sind Zahlenwerte, boolesche Parameter, Text, Zeichenketten und Blockdaten erlaubt. Der für den jeweiligen Befehl verlangte Parametertyp sowie der erlaubte Wertebereich sind in der Befehlsbeschreibung (siehe Abschnitt 3.6) angegeben.

Zahlenwerte

Zahlenwerte können in jeder gebräuchlichen Form eingegeben werden, also mit Vorzeichen, Dezimalpunkt und Exponent. Überschreiten die Werte die Auflösung des Gerätes, wird auf oder abgerundet. Die Mantisse darf bis zu 255 Zeichen lang sein,der Exponent muß im Wertebereich -32 000 bis + 32 000 liegen. Der Exponent wird durch ein "E" oder "e" eingeleitet. Die Angabe des Exponenten allein ist nicht erlaubt.

Beispiel:
`*ESE 255 = *ESE 25.5 e+001 = *ESE 2.55E2`

Blockdaten

Blockdaten sind ein Übertragungsformat, das sich für die Übertragung von Binärdaten (z.B. EBU Kommandos) eignet. Ein Befehl mit einem Blockdatenparameter hat folgenden Aufbau:

Beispiel:
`EBU #3148xxxxxxxx ...`
Das ASCII-Zeichen # leitet den Datenblock ein. Die nächste Zahl gibt an, wieviele der folgenden Ziffern die Länge des Datenblocks beschreiben. Im Beispiel geben die 3 folgenden Ziffern die Länge mit 148 Bytes an. Es folgen die 148 Datenbytes. Während der Übertragung dieser Datenbytes werden alle Ende oder sonstigen Steuerzeichen ignoriert, bis alle Bytes übertragen sind.

3.5.5 Übersicht der Syntaxelemente

Eine Übersicht der Syntaxelemente bietet folgende Zusammenstellung.

;	Der Strichpunkt trennt zwei Befehle einer Befehlszeile
?	Das Fragezeichen bildet eine Abfrageeinheit
*	Der Stern kennzeichnet ein Common Command
#	Das Doppelkreuz leitet Blockdaten ein
	Ein "White Space" (ASCII-Code 0...9, 11...32 dezimal, z.B. Leerzeichen) trennt Header und Parameter

3.6 Beschreibung der Befehle

3.6.1 Notation

In den folgenden Abschnitten werden alle im Gerät realisierten Befehle nach Befehlssystem getrennt zuerst tabellarisch aufgelistet und dann ausführlich beschrieben.

Befehlstabelle

Befehl: Die Tabelle gibt in der Spalte Befehle einen Überblick über die Befehle und ihre hierarchische Anordnung (siehe Einrückungen).

Parameter: In der Spalte Parameter werden die verlangten Parameter mit ihrem Wertebereich angegeben.

Einheit: Die Spalte Einheit zeigt die Grundeinheit der physikalischen Parameter an.

Bemerkung: In der Spalte Bemerkung wird angegeben
- ob der Befehl keine Abfrageform besitzt,
- ob der Befehl nur eine Abfrageform besitzt und
- ob dieser Befehl nur bei einer bestimmten Geräteoption realisiert ist.

Groß-/ Kleinschreibung

Das Gerät unterscheidet nicht zwischen Groß- und Kleinbuchstaben.Das gilt natürlich nicht innerhalb von Binärdaten.

3.6.2 Common Commands

Die Common Commands sind der Norm IEEE 488.2 (IEC 625.2) entnommen. Gleiche Befehle haben in unterschiedlichen Geräten gleiche Wirkung. Die Header dieser Befehle bestehen aus einem Stern"*", dem drei Buchstaben folgen. Viele Common Commands betreffen das Status-Reporting-System, das in Abschnitt 3.8 ausführlich beschrieben ist.

Befehl	*Parameter*	*Einheit*	*Bemerkung*
***CLS**			keine Abfrage
***ESE**	0...255		
***ESR?**			nur Abfrage
***IDN?**			nur Abfrage
***OPC**			
***RST**			keine Abfrage
***SRE**	0...255		
***STB?**			nur Abfrage
***TST?**			nur Abfrage
***WAI**			

***CLS** CLEAR STATUS setzt das Status Byte (STB), das Standard-Event-Register (ESR) und den EVENt-Teil des QUEStionable- und des OPERation-Registers auf Null. Der Befehl verändert die Masken- und Transition-Teile der Register nicht. Er löscht den Ausgabepuffer.

***ESE** 0...255 EVENT STATUS ENABLE setzt das Event-Status-Enable-Register auf den angegebenen Wert. Der Abfragebefehl ***ESE?** gibt den Inhalt des Event-Status-Enable-Registers in dezimaler Form zurück.

***ESR?** STANDARD EVENT STATUS QUERY gibt den Inhalt des Event-Status-Registers in dezimaler Form zurück (0...255) und setzt danach das Register auf Null.

***IDN?** IDENTIFICATION QUERY fragt die Gerätekennung ab.
Die Geräteantwort lautet zum Beispiel:
"Rohde&Schwarz, SME03,00000001, 1.03"
03 = Variantenkennung
00000001 = Seriennummer
1.03 = Firmware-Versionsnummer

***OPC** OPERATION COMPLETE setzt das Bit 0 im Event-Status-Register, wenn alle vorausgegangenen Befehle abgearbeitet sind. Dieses Bit kann zur Auslösung eines Service Requests benutzt werden (siehe Abschnitt 3.7).

***OPC?** OPERATION COMPLETE QUERY schreibt die Nachricht "1" in den Ausgabepuffer, sobald alle vorangegangenen Befehle ausgeführt sind (siehe Abschnitt 3.7).

***RST** RESET versetzt das Gerät in einen definierten Grundzustand. Der Befehl entspricht im Wesentlichen einem Druck auf die Taste [RESET]. Die Grundeinstellung ist in der Befehlsbeschreibung der Befehle angegeben.

***SRE 0...255** SERVICE REQUEST ENABLE setzt das Service Request Enable Register auf den angegebenen Wert. Bit 6 (MSS-Maskenbit) bleibt 0. Dieser Befehl bestimmt, unter welchen Bedingungen ein Service Request ausgelöst wird. Der Abfragebefehl ***SRE?** liest den Inhalt des Service Request Enable Registers in dezimaler Form aus. Bit 6 ist immer 0.

***STB?** READ STATUS BYTE QUERY liest den Inhalt des Status Bytes in dezimaler Form aus.

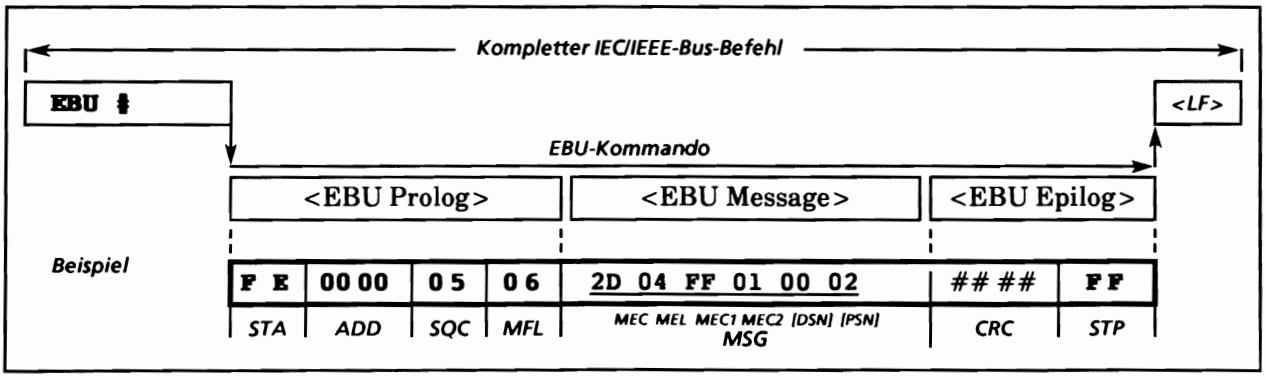
***TST?** SELF TEST QUERY löst alle in Abschnitt 4.4, Funktionstest, angegebenen Selbsttests des Gerätes aus und gibt einen Fehlercode in dezimaler Form aus.

***WA** WAIT-to-CONTINUE erlaubt die Abarbeitung der nachfolgenden Befehle erst, nachdem alle vorhergehenden Befehle durchgeführt und alle Signale eingeschwungen sind (siehe auch Abschnitt 3.7 und "*OPC").

3.6.3 Gerätespezifische Befehle

Das Gerät verfügt über einen Befehl der zur Übertragung von EBU Kommandos dient (genaue Beschreibung im 3. Teil des Handbuchs Draft SPB490, Universal Encoder communication protocol, European Broadcasting Union).

Befehl	*Parameter*	*Einheit*	*Bemerkung*
EBU	Blockdaten, EBU Kommando als Datenteil		EBU Kommandos ohne Antwort
EBU?	Blockdaten, EBU Kommando als Datenteil		EBU Kommandos mit Antwort



EBU # 0 Eingabe-Kommando **IEC EBU # <*EBU-Kommando*>**
EBU? # 0 Abfrage-Kommando **IEC EBU? # <*EBU-Kommando*>**
Anwort-Format

3.7 Gerätemodell und Befehlsbearbeitung

Das in Bild 3-1 dargestellte Gerätemodell wurde unter dem Gesichtspunkt der Abarbeitung von IEC-Bus-Befehlen erstellt. Die einzelnen Komponenten arbeiten voneinander unabhängig und gleichzeitig. Sie kommunizieren untereinander durch sogenannte "Nachrichten".

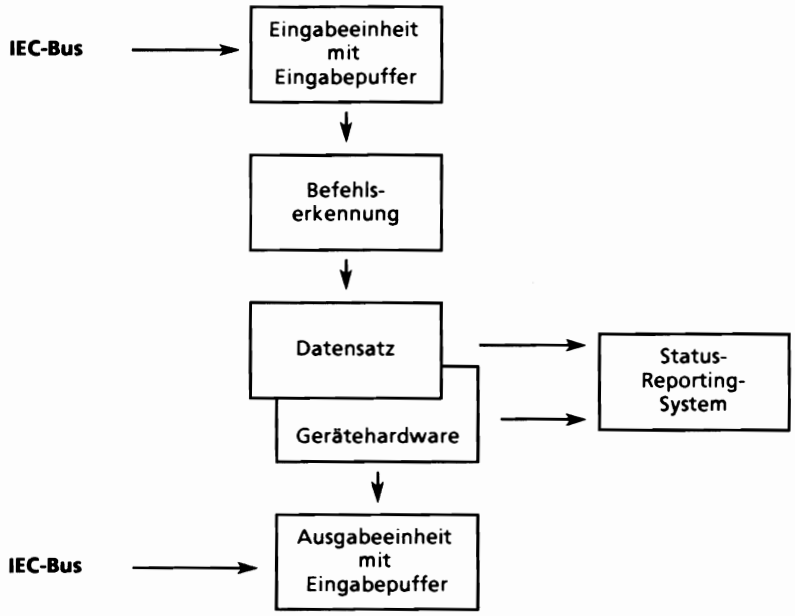


***Bild 3-1** Gerätemodell bei Fernbedienung durch den IEC/IEEE-Bus*

3.7.1 Eingabeeinheit

Die Eingabeeinheit empfängt Befehle zeichenweise vom Bus und sammelt sie im Eingabepuffer. Der Eingabepuffer ist 484 Zeichen groß. Die Eingabeeinheit schickt eine Nachricht an die Befehlserkennung, sobald der Eingabepuffer voll ist, oder sobald sie ein Endekennzeichen, <PROGRAM MESSAGE TERMINATOR>, wie in IEEE 488.2 definiert, oder die Schnittstellennachricht DCL empfängt.

Ist der Eingabepuffer voll, wird der IEC-Bus-Verkehr angehalten und die bis dahin empfangenen Daten werden verarbeitet. Danach wird der IEC-Bus-Verkehr fortgesetzt. Ist dagegen der Puffer beim Empfang des Endekennzeichens noch nicht voll, so kann die Eingabeeinheit während der Befehlserkennung und Ausführung bereits das nächste Kommando empfangen. Der Empfang eines DCL löscht den Eingabepuffer und löst sofort eine Nachricht an die Befehlserkennung aus.

3.7.2 Befehlserkennung

Die Befehlserkennung analysiert die von der Eingabeeinheit empfangenen Daten. Dabei geht sie in der Reihenfolge vor, in der sie die Daten erhält. Lediglich ein DCL wird bevorzugt abgearbeitet. Jeder erkannte Befehl wird sofort an den Datensatz weitergereicht, ohne dort allerdings sofort ausgeführt zu werden.

Syntaktische Fehler im Befehl werden hier erkannt und an das Status-Reporting-System weitergeleitet. Der Rest einer Befehlszeile nach einem Syntaxfehler wird soweit möglich weiter analysiert und abgearbeitet.

Erkennt die Befehlserkennung ein Endekennzeichen oder ein DCL, fordert sie den Datensatz auf, die Befehle jetzt auch in der Gerätehardware einzustellen. Danach ist sie sofort wieder bereit, Befehle zu verarbeiten. Das bedeutet für die Befehlsabarbeitung, daß weitere Befehle schon abgearbeitet werden können, noch während die Hardware eingestellt wird ("overlapping execution").

3.7.3 Datensatz und Gerätehardware

Der Ausdruck "Gerätehardware" bezeichnet hier den Teil des Gerätes, der die eigentliche Gerätefunktion erfüllt: Signalerzeugung, Messung etc.. Der Steuerrechner zählt nicht dazu.

Der Datensatz ist ein genaues Abbild der Gerätehardware in der Software.

IEC-Bus-Einstellbefehle führen zu einer Änderung im Datensatz. Die Datensatzverwaltung trägt die neuen Werte (z.B. Frequenz) in den Datensatz ein.

Die Daten werden erst unmittelbar bevor sie an die Gerätehardware übergeben werden auf Verträglichkeit untereinander und mit der Gerätehardware geprüft. Erweist sich dabei, daß eine Ausführung nicht möglich ist, wird ein "Execution Error" an das Status-Reporting-System gemeldet. Alle Änderungen des Datensatzes werden verworfen, die Gerätehardware wird nicht neu eingestellt.

IEC-Bus-Abfragebefehle veranlassen die Datensatzverwaltung, die gewünschten Daten an die Ausgabeeinheit zu senden.

3.7.4 Status-Reporting-System

Das Status-Reporting-System sammelt Informationen über den Gerätezustand und stellt sie auf Anforderung der Ausgabeeinheit zur Verfügung. Der genaue Aufbau und die Funktion ist im Abschnitt 3.8 beschrieben.

3.7.5 Ausgabeeinheit

Die Ausgabeeinheit sammelt die vom Controller angeforderte Information, die sie von der Datensatzverwaltung erhält. Sie bereitet sie entsprechend den IEEE 488.2 Regeln auf und stellt sie im Ausgabepuffer zur Verfügung.

Wird das Gerät als Talker adressiert, ohne daß der Ausgabepuffer Daten enthält oder von der Datensatzverwaltung erwartet, schickt setzt die Ausgabeeinheit das Query Error Bit im Event-Status-Register. Auf dem IEC-Bus werden keine Daten geschickt, der Controller wartet, bis er sein Zeitlimit erreicht hat. Dieses Verhalten ist durch IEEE 488.2 vorgeschrieben.

3.7.6 Befehlsreihenfolge und Befehlssynchronisation

Aus dem oben Gesagten wird deutlich, daß potentiell alle Befehle überlappend ausgeführt werden können. Ebenso werden Einstellbefehle innerhalb einer Befehlszeile nicht unbedingt in der Reihenfolge des Empfangs abgearbeitet.

Um sicherzustellen, daß Befehle tatsächlich in einer bestimmten Reihenfolge ausgeführt werden, muß jeder Befehl in einer eigenen Befehlszeile, d.h., mit einem eigenen IBWRT()-Aufruf gesendet werden.

Um eine überlappende Ausführung von Befehlen zu verhindern, muß einer der Befehle ***OPC**, ***OPC?** oder ***WAI** verwendet werden. Alle drei Befehle bewirken, daß eine bestimmte Aktion erst ausgelöst wird, nachdem die Hardware eingestellt und eingeschwungen ist. Der Controller kann durch geeignete Programmierung dazu gezwungen werden, auf das Eintreten der jeweiligen Aktion zu warten (siehe Tabelle 3-1).

Tabelle 3-1 Synchronisation mit *OPC, *OPC? und *WAI

Befehl	*Aktion nach Einschwingen der Hardware*	*Programmierung des Controllers*
***OPC**	Setzen des Operation-Complete Bits im ESR	- Setzen des Bit 0 im ESE - Setzen des Bit 5 im SRE - Warten auf Bedienerruf (SRQ)
***OPC?**	Schreiben einer "1" in den Ausgabepuffer	Adressieren des Gerätes als Talker
***WAI**	Fortsetzen des IEC-Bus-Handshakes	Absenden des nächsten Befehls

3.8 Status-Reporting-System

Das Status-Reporting-System (siehe Bild 3-2) speichert Informationen über den momentanen Betriebszustand des Gerätes, z.B., daß die Befehlsbearbeitung beendet ist, und über aufgetretene Fehler. Diese Informationen werden in den Statusregistern abgelegt. Die Statusregister können über IEC-Bus abgefragt werden.

Die Informationen sind hierarchisch strukturiert. Die oberste Ebene bildet das in IEEE 488.2 definierte Register Status Byte (STB) und sein zugehöriges Maskenregister Service-Request-Enable (SRE). Das STB erhält seine Information von dem ebenfalls in IEEE 488.2 definierten Standard-Event-Status-Register (ESR) mit dem zugehörigen Maskenregister Standard-Event-Status-Enable (ESE).

Der Ausgabepuffer enthält die Nachrichten, die das Gerät an den Controller zurücksendet. Er ist kein Teil des Status-Reporting-Systems, bestimmt aber den Wert des MAV-Bits im STB und ist daher in Bild 3-4 dargestellt.

3.8.1 Übersicht der Statusregister

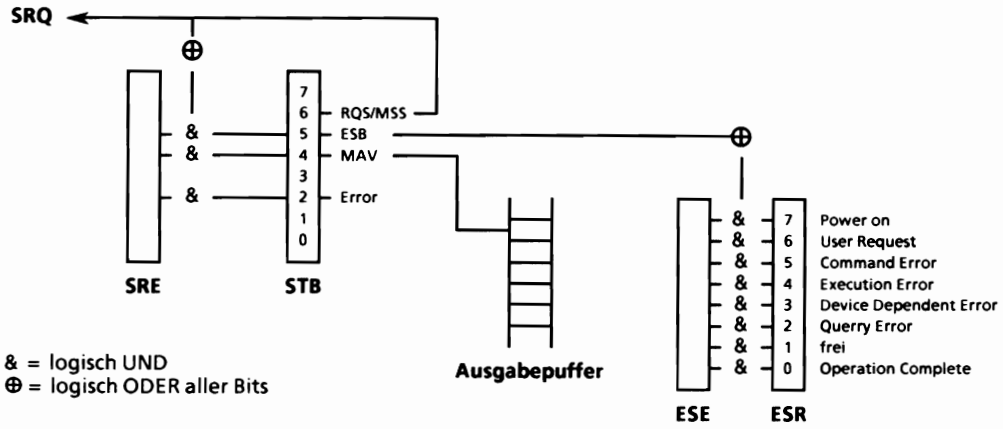


Bild 3-2 Übersicht der Statusregister

3.8.2 Beschreibung der Statusregister

3.8.2.1 Status Byte (STB) und ServiceRequestEnableRegister (SRE)

Das STB ist in IEEE 488.2 definiert. Es gibt einen groben Überblick über den Zustand des Gerätes, indem es als Sammelbecken für die Informationen der anderen, untergeordneten Register dient. Eine Besonderheit dieses Registers ist, als daß das Bit 6 als SummenBit der übrigen Bits des Status Bytes wirkt.

Das Status Byte wird mit dem Befehl ***STB?** oder einem "Serial Poll" ausgelesen.

Zum STB gehört das SRE. Jedem Bit des STB ist ein Bit im SRE zugeordnet. Das Bit 6 des SRE wird ignoriert. Wenn im SRE ein Bit gesetzt ist, und das zugehörige Bit im STB von 0 nach 1 wechselt, wird ein Service Request (SRQ) auf dem IEC-Bus erzeugt, der beim Controller einen Interrupt auslöst, falls dieser entsprechend konfiguriert ist, und dort weiterverarbeitet werden kann.

Das SRE kann mit dem Befehl ***SRE** gesetzt und mit ***SRE?** ausgelesen werden.

Tabelle 3-2 Bedeutung der benutzten Bits im Status-Byte

Bit-Nr	*Bedeutung*
2	**Error** Das Bit wird gesetzt, wenn ein Fehler aufgetreten ist. Wird dieses Bit durch das SRE freigegeben, erzeugt jeder Fehler einen Service Request. Dadurch kann ein Fehler erkannt und durch eine Abfrage des Event Status Registers genauer spezifiziert werden. Die Abfrage liefert. Diese Vorgehensweise ist zu empfehlen, da es die Probleme bei der IEC-Bus-Steuerung beträchtlich reduziert.
4	**MAV-Bit** (Message available) Das Bit ist gesetzt, wenn im Ausgabepuffer eine Nachricht vorhanden ist, die gelesen werden kann. Dieses Bit kann dazu verwendet werden, das Einlesen von Daten vom Gerät in den Controller zu automatisieren (siehe Anhang D3, Programmbeispiele)
5	**ESB-Bit** Summen-Bit des Event-Status-Registers. Es wird gesetzt, wenn eines der Bits im Event-Status-Register gesetzt und im Event-Status-Enable-Register freigegeben ist. Ein Setzen dieses Bits weist auf einen schwerwiegenden Fehler hin, der durch die Abfrage des Event-Status-Registers näher spezifiziert werden kann.
6	**MSS-Bit** (Master-Status-Summary-Bit) Dieses Bit ist gesetzt, wenn das Gerät eine Service Request auslöst. Das ist dann der Fall, wenn eines der anderen Bits dieses Registers zusammen mit seinem Maskenbit im Service-Request-Enable-Register SRE gesetzt ist.

3.8.2.2 EventStatusRegister (ESR) und Event-Status-Enable-Register (ESE)

Das ESR ist in IEEE 488.2 definiert. Das Event-Status-Register kann mit dem Befehl ***ESR?** ausgelesen werden.

Das ESE ist der zugehörige Enable-Teil. Es kann mit dem Befehl ***ESE** gesetzt und mit dem Befehl ***ESE?** ausgelesen werden.

Tabelle 3-3 Bedeutung der benutzten Bits im Event-Status-Register

Bit-Nr	*Bedeutung*
0	**Operation Complete** Dieses Bit wird nach Empfang des Befehls *OPC genau dann gesetzt, wenn alle vorausgehenden Befehle ausgeführt sind.
2	**Query Error** Dieses Bit wird gesetzt, wenn entweder der Controller Daten vom Gerät lesen möchte, aber zuvor keinen Datenanforderungsbefehl gesendet hat, oder angeforderte Daten nicht abholt und statt dessen neue Anweisungen zum Gerät schickt. Häufige Ursache ist ein fehlerhafter und daher nicht ausführbarer Abfragebefehl.
3	**Device-dependent Error** Dieses Bit wird gesetzt, wenn ein geräteabhängiger Fehler auftritt.
4	**Execution Error** Dieses Bit wird gesetzt, wenn ein empfangener Befehl zwar syntaktisch korrekt ist, aber aufgrund verschiedener Randbedingungen nicht ausgeführt werden kann. In die Error-Queue wird eine Fehlermeldung mit einer Nummer zwischen -200 und -300 eingetragen, die den Fehler näher bezeichnet (siehe Anhang B3, Fehlermeldungen)
5	**Command Error** Dieses Bit wird gesetzt, wenn ein undefinierter oder syntaktisch nicht korrekter Befehl empfangen wird. In die Error Queue wird eine Fehlermeldung mit einer Nummer zwischen -100 und -200 eingetragen, die den Fehler näher bezeichnet (siehe Anhang B3, Fehlermeldungen)
6	**User Request** Dieses Bit wird beim Druck auf die Taste [LOCAL] gesetzt, d.h., wenn das Gerät auf Handbedienung umgeschaltet wird.
7	**Power On** (Netzspannung ein) Dieses Bit wird beim Einschalten des Gerätes gesetzt.

3.8.3 Einsatz des Status-Reporting-Systems

Um das Status Reporting System effektiv nutzen zu können, muß die dort enthaltene Information an den Controller übertragen und dort weiterverarbeitet werden. Dazu existieren mehrere Verfahren, die im Folgenden dargestellt werden.

3.8.3.1 Bedienungsruf (Service Request), Nutzung der Hierarchiestruktur

Das Gerät kann unter bestimmten Bedingungen einen "Bedienungsruf" (SRQ) an den Controller schicken. Dieser Bedienungsruf löst üblicherweise beim Controller einen Interrupt aus, auf den das Steuerprogramm mit entsprechenden Aktionen reagieren kann. Wie aus Bild 3.2 (Abschnitt 3.8.2) ersichtlich, wird ein SRQ immer dann ausgelöst, wenn eines oder mehrere der Bits 2, 4 oder 5 des Status Bytes gesetzt und im SRE freigeschaltet sind. Jedes dieser Bits faßt die Information eines weiteren Registers, der Error Queue oder des Ausgabepuffers zusammen. Durch entsprechendes Setzen der ENABle-Teile der Statusregister kann erreicht werden, daß beliebige Bits in einem beliebigen Statusregister einen SRQ auslösen. Um die Möglichkeiten des Service-Request auszunutzen, sollten in den Enable-Registern SRE und im ESE alle Bits auf "1" gesetzt werden.

Beispiel (vergleiche auch Bild 3.2, Abschnitt 3.8.2):

Den Befehl `*OPC` zur Erzeugung eines SRQs verwenden

- im ESE das Bit0 setzen (Operation Complete)
- im SRE das Bit5 setzen (ESB)
  Das Gerät erzeugt nach Abschluß seiner Einstellungen einen SRQ.

Der SRQ ist die einzige Möglichkeit für das Gerät, von sich aus aktiv zu werden. Jedes Controller-Programm sollte das Gerät so einstellen, daß bei Fehlfunktionen ein Bedienungsruf ausgelöst wird. Auf den Bedienungsruf sollte das Programm entsprechend reagieren.

3.8.3.2 Serienabfrage (Serial Poll)

Bei einem Serial Poll wird, wie bei dem Befehl `*STB`, das Status Byte eines Gerätes abgefragt. Allerdings wird die Abfrage über Schnittstellennachrichten realisiert und ist daher deutlich schneller. Das Serial-Poll-Verfahren ist bereits in IEEE 488.1 definiert und war früher die einzige geräteübergreifend einheitliche Möglichkeit, das Status Byte abzufragen. Das Verfahren funktioniert auch bei Geräten, die sich nicht an IEEE 488.2 halten.

Der QuickBASIC-Befehl für die Ausführung eines Serial Poll lautet `IBRSP()`. Der Serial Poll wird hauptsächlich verwendet, um einen schnellen Überblick über den Zustand mehrerer an den IEC-Bus angeschlossener Geräte zu erhalten.

3.8.3.3 Abfrage durch Befehle

Jeder Teil jedes Statusregisters kann durch Abfragebefehle ausgelesen werden. Die einzelnen Befehle sind bei der detaillierten Beschreibung der Register in Abschnitt 3.8.2 angegeben. Zurückgegeben wird immer eine Zahl, die das Bitmuster des abgefragten Registers darstellt. Die Auswertung dieser Zahl obliegt dem Controller-Programm.

Abfragebefehle werden üblicherweise nach einem aufgetretenen SRQ verwendet, um genauere Informationen über die Ursache des SRQ zu erhalten.

3.8.4 Rücksetzwerte des Status-Reporting-Systems

In Tabelle 3-4 sind die verschiedenen Befehle und Ereignisse zusammengefaßt, die ein Rücksetzen des Status-Reporting-Systems bewirken. Keiner der Befehle, mit Ausnahme von *RST beinflußt die funktionalen Geräteeinstellungen. Insbesondere verändert DCL die Geräteeinstellungen nicht.

Tabelle 3-4 Rücksetzen von Gerätefunktionen

Ereignis / *Wirkung*	*Einschalten der Netzspannung*		Befehle	Befehle	
	Power-On-Status-Clear				
	0	1	*DCL, SDC (Device Clear, Selected Device Clear)*	*RST	*CLS
STB,ESR löschen	---	ja	---	---	ja
SRE,ESE löschen	---	ja	---	---	---
EVENt-Teile der Register löschen	---	ja	---	---	ja
Ausgabepuffer löschen	ja	ja	ja	1)	1)
Befehlsbearbeitung und Eingabepuffer löschen	ja	ja	ja		ja

1) Jeder Befehl, der als erster in einer Befehlszeile steht, d.h., unmittelbar einem <PROGRAM MESSAGE TERMINATOR> folgt, löscht den Ausgabepuffer

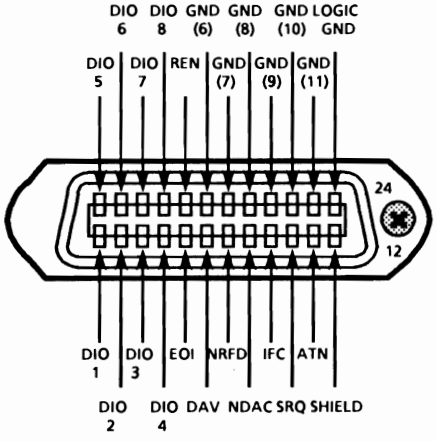
Anhang A

IEC-Bus-Schnittstelle

Das Gerät ist serienmäßig mit einem IEC-Bus-Anschluß ausgestattet. Die Anschlußbuchse nach IEEE488 befindet sich an der Geräterückseite. Über die Schnittstelle kann ein Controller zur Fernsteuerung angeschlossen werden. Der Anschluß erfolgt mit einem geschirmten Kabel.

Eigenschaften der Schnittstelle

- 8-bit-parallele Datenübertragung
- bidirektionale Datenübertragung
- Dreidraht-Handshake
- hohe Datenübertragungsrate, max. 350 kByte/s
- bis zu 15 Geräte anschließbar
- maximale Länge der Verbindungskabel 15 m (Einzelverbindung 2m)
- Wired-Or-Verknüpfung bei Parallelschaltung mehrerer Geräte.



Busleitungen

1. **Datenbus mit 8 Leitungen DIO 1...DIO 8.**
   Die Übertragung erfolgt bitparallel und byte-seriell im ASCII/ISO-Code. DIO1 ist das niedrigstwertige und DIO8 das höchstwertige Bit.

2. **Steuerbus mit 5 Leitungen.**
   IFC (Interface Clear),
   aktiv LOW setzt die Schnittstellen der angeschlossenen Geräte in die Grundeinstellung zurück.
   ATN (Attention),
   aktiv LOW meldet die Übertragung von Schnittstellennachrichten inaktiv HIGH meldet die Übertragung von Gerätenachrichten.
   SRQ (Service Request),
   aktiv LOW ermöglicht dem angeschlossenen Gerät, einen Bedienungsruf an den Controller zu senden.
   REN (Remote Enable),
   aktiv LOW ermöglicht das Umschalten auf Fernsteuerung.
   EOI (End or Identify),
   hat in Verbindung mit ATN zwei Funktionen:
   aktiv LOW kennzeichnet das Ende einer Datenübertragung bei ATN = HIGH
   aktiv LOW löst Parallelabfrage (Parallel Poll) aus bei ATN = LOW.

3. **Handshake Bus mit drei Leitungen.**
   DAV (Data Valid),
   aktiv LOW meldet ein gültiges Datenbyte auf dem Datenbus.
   NRFD (Not Ready For Data),
   aktiv LOW meldet, daß eines der angeschlossenen Geräte zur Datenübernahme nicht bereit ist .
   NDAC (Not Data Accepted),
   aktiv LOW, solange das angeschlossene Gerät die am Datenbus anliegenden Daten übernimmt.

Schnittstellenfunktionen

Über IEC-Bus fernsteuerbare Geräte können mit unterschiedlichen Schnittstellenfunktionen ausgerüstet sein. Tabelle A-1 führt die für das Gerät zutreffenden Schnittstellenfunktionen auf.

Tabelle A-1 Schnittstellenfunktionen

Steuerzeichen	Schnittstellenfunktionen
SH1	Handshake-Quellenfunktion (Source Handshake)
AH1	Handshake-Senkenfunktion (Acceptor Handshake)
L4	Listener-Funktion.
T6	Talker-Funktion, Fähigkeit zur Antwort auf Serienabfrage
SR1	Bedienungs-Ruf-Funktion (Service Request)
PP0	keine Parallel-Poll-Funktion
RL2	Remote/Local-Umschaltfunktion ohne Local Lockout
DC1	Rücksetzfunktion (Device Clear)
DT0	keine Auslösefunktion (Device Trigger)
C0	keine Controllerfunktion

Schnittstellennachrichten

Schnittstellennachrichten werden auf den Datenleitungen zum Gerät übertragen, wobei die Attentionleitung aktiv (LOW) ist. Sie dienen der Kommunikation zwischen Gerät und Steuerrechner.

Universalbefehle

Die Universalbefehle liegen im Code-Bereich 10...1F hex. Sie wirken ohne vorhergehende Adressierung auf alle an den Bus angeschlossenen Geräte.

Tabelle A-2 Universalbefehle

Befehl	*QuickBASIC-Befehl*	*Wirkung auf das Gerät*
DCL (Device Clear)	IBCMD (controller%, CHR$(20))	Bricht die Bearbeitung der gerade empfangenen Befehle ab und setzt die Befehlsbearbeitungs-Software in einen definierten Anfangszustand. Verändert die Geräteeinstellung nicht.
IFC (Interface Clear)	IBSIC (controller%)	Setzt die Schnittstellen in die Grundeinstellung zurück.
SPE (Serial Poll Enable)	IBCMD (controller%, CHR$(24))	Bereit zur Serienabfrage
SPD (Serial Poll Disable)	IBCMD (controller%, CHR$(25))	Ende der Serienabfrage

Adressierte Befehle

Die adressierten Befehle liegen im Code-Bereich 00...0F hex. Sie wirken nur auf Geräte, die als Listener adressiert sind.

Tabelle A-3 Adressierte Befehle

Befehl	*QuickBASIC-Befehl*	*Wirkung auf das Gerät*
SDC (Selected Device Clear)	IBCLR (device%)	Bricht die Bearbeitung der gerade empfangenen Befehle ab und setzt die Befehlsbearbeitungs-Software in einen definierten Anfangszustand. Verändert die Geräteeinstellung nicht.
GTL (Go to Local)	IBLOC (device%)	Übergang in den Zustand "Local" (Handbedienung)

Anhang B

Einführung

Über diese Schnittstelle kann ein IEC/IEEE-Bus-Controller angeschlossen werden.

Die Schnittstelle hat folgende Eigenschaften:

- Datenbus 8-bit-parallel, Byte seriell
- bidirektionale Datenübertragung
- hohe Datensicherheit über Dreidraht-Handshake
- hohe Datenübertragungsrate bis 400 KByte/s
- maximale Länge der Verbindungskabel bis 20m
- bis zu 15 Geräte sind pro Bussystem anschließbar
- Managementbus und Softwareprotokoll erlauben zahlreiche Sonderfunktionen

Anschluß der verschiedenen Gerätetypen

Die am Bus angeschlossenen Geräte lassen sich in drei Gruppen einteilen:

a) Controller

Er übt die Steuerfunktion auf dem Bus aus. In dieser Funktion sendet er Adressen und Befehle und ist in der Lage, Geräteabfragen durchzuführen, um festzustellen, welches Gerät bedient werden muß.

b) Hörer (Listener)

Geräte, die als Hörer arbeiten, empfangen über den Bus gerätebezogene Daten. Dazu müssen sie vorher vom Controller adressiert worden sein. Es dürfen gleichzeitig mehrere Geräte die Funktion des Hörers haben.

c) Sprecher (Talker)

Als Sprecher arbeitende Geräte können über den Bus gerätebezogene Daten an andere senden. Dies geschieht ebenfalls erst, wenn durch den Controller eine Adressierung als Sprecher erfolgt ist. Es darf zu jedem Zeitpunkt immer nur ein Gerät Sprecher sein. Um dies zu gewährleisten, wird, bevor ein neuer Sprecher adressiert wird, der alte zuvor automatisch entadressiert.

Viele der über IEC-Bus fernsteuerbaren Geräte können mehrere Funktionen (Controller, Hörer oder Sprecher) ausüben: Der Controller arbeitet, während er z.B. von einem Digitalvoltmeter Meßdaten empfängt, als Hörer. Das Digitalvoltmeter muß, bevor es als Sprecher die Meßdaten sendet, zuvor als Hörer empfangen haben, welchen Meßbereich es einstellen soll.

Der Ablauf einer Datenübertragung, etwa vom Controller zu einem Hörer geschieht prinzipiell so, daß der Controller zuerst das betreffende Gerät als Hörer adressiert, d.h., es werden über den Datenbus DIO1...DIO8 Daten gesendet, wobei gleichzeitig die Leitung ATN aktiv ist, die das empfangende Gerät als seine Höreradresse erkennt. Danach nimmt der Controller die ATN-Leitung wieder zurück und sendet die eigentlichen Gerätedaten über den Datenbus.

Der zeitliche Ablauf des Datenverkehrs wird mit den drei Handshake-Leitungen geregelt, wie es Bild B-1 zeigt. Das zugehörige Flußdiagramm ist aus Bild B-2 ersichtlich.

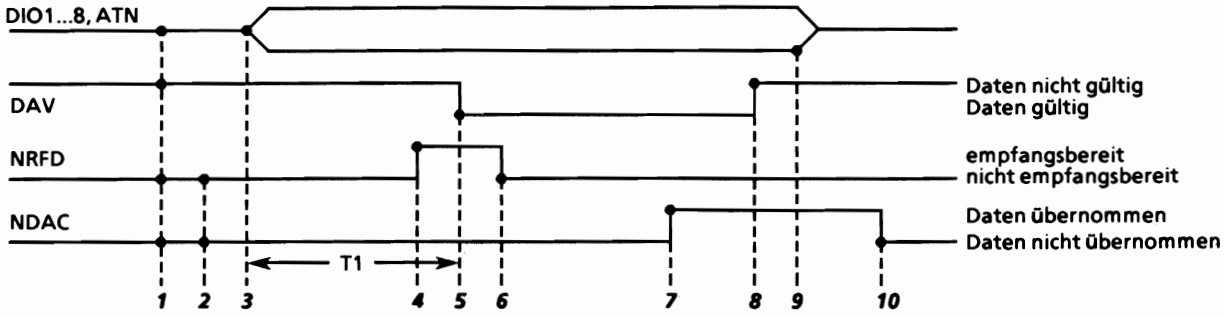
Befehlsstruktur

Jeder Befehl setzt sich zusammen aus:

Prüfsumme - Informationskurzzeichen - Parameterteil - Daten

Um Datenübertragungsfehler leichter erkennen zu können, wird jedem Befehl eine zweistellige Prüfsumme vorausgeschickt. Jeder ausgegebene Befehl wird mit dem ASCII-Zeichen <ETX> abgeschlossen. Zusammengesetzte Ausgaben, die Teillisten der alternativen Listen, werden mit dem ASCII-Zeichen <RS> aneinandergefügt und mit dem ASCII-Zeichen <ETX> abgeschlossen. Nach der Ausgabe eines Befehls erwartet der Decoder

Bild B-1 Zeitlicher Ablauf des Handshake-Prozesses für die Übertragung eines Bytes



1 Anfangszustand des Busses
2 Sprecher prüft auf Fehlerzustand NRFD und NDAC = High
3 Sprecher legt nach der Fehlerprüfung das erste Datenbyte an
4 Alle Hörer melden mit NRFD = High (falsch), daß sie zur Datenübernahme bereit sind.
5 Nachdem die Daten auf den DIO-Leitungen eingeschwungen sind und der Sprecher feststellt, daß NRFD = High ist, setzt er nach T1 DAV = Low und meldet damit die Daten als gültig.
6 Der Hörer, der als erster DAV = Low feststellt, setzt NRFD = Low. Damit ist die Bereitschaft zur Datenübernahme beendet.
7 Erst wenn der letzte Hörer Daten verarbeitet hat, gibt er die NDAC-Leitung frei, (Meldung Zustand "Daten übernommen")
8 Der Sprecher stellt NDAC = High fest und setzt DAV = High. Damit meldet er die Daten wieder als ungültig.
9 Der Sprecher schaltet das Datenbyte von den DIO-Leitungen ab.
10 Der erste Hörer, der DAV = High feststellt, setzt NDAC = Low, zur Vorbereitung des nächsten Zyklus einer Datenübernahme.

Bild B-2 Flußdiagramm des Handshake-Prozesses für die Übertragung eines Bytes