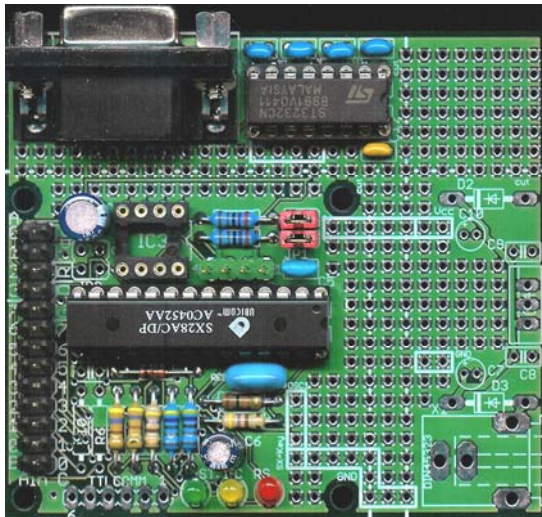


## RS-232 – I<sup>2</sup>C Adapter/Monitor – Version 3.8

Stand Oktober 2006



Der Adapter dient zur Kommunikation zwischen einem PC über eine RS-232 Schnittstelle und verschiedenen Bausteinen einer Steuerung, die an einem gemeinsamen I<sup>2</sup>C-Bus angeschlossen sind.

Der Befehlssatz des Adapters enthält die notwendigen Kommandos, um mit dem I<sup>2</sup>C-Bus zu kommunizieren. Die sogenannten "high-level"-Kommandos führen die notwendigen Aktionen für eine vollständige I<sup>2</sup>C-Kommunikation transparent für die RS-232-Seite aus, d.h. Setzen der Start-Bedingung, Senden der Adresse zusammen mit dem Schreib/Lese-Bit, Senden oder Lesen der Datenbytes, Auswerten oder Setzen des Acknowledge-Bits und abschließendes Setzen der Stop-Bedingung.

Da der Adapter die Ergebnisse von „high-level“-Kommandos mit einem führenden „O“ (oder „E“ bei Fehlern) zurückgibt, müssen die von einem I<sup>2</sup>C-Baustein gelesenen Daten in einem Puffer zwischengespeichert werden. Dieser Puffer kann maximal 16 Bytes speichern und daraus ergibt sich, dass höchstens 16 Datenbytes mit einem „high-level“-Kommando gelesen werden können.

Der Befehlssatz des Adapters enthält aber auch sogenannte „low-level“-Kommandos, um mit dem I<sup>2</sup>C-Bus zu kommunizieren. Diese Kommandos erfordern eine etwas umfangreichere Aktivität auf der RS-232-Seite. Sie ermöglichen aber auf der anderen Seite die Übertragung einer beliebigen Anzahl von Datenbytes an oder von einem I<sup>2</sup>C-Baustein. Sie sind auch nützlich, um mit Bausteinen zu kommunizieren, die andere, als die durch die „high-level“-Kommandos unterstützten Sequenzen erwarten.

Neben seiner Hauptaufgabe, Kommandos zwischen RS-232 und I<sup>2</sup>C zu übersetzen, unterstützt der Adapter auch Kommandos über RS-232, mit denen 13 parallele Ein-/Ausgangsleitungen gelesen bzw. gesteuert werden können. Acht Leitungen steuern außerdem interne 16-Bit Zähler, die mit entsprechenden Kommandos gelesen und zurückgesetzt werden können.

Zu Testzwecken kann der Adapter in den „Monitor-Modus“ gesetzt werden. In diesem Modus „beobachtet“ der Adapter passiv einen I<sup>2</sup>C-Bus und überträgt die erkannten Bus-Aktivitäten über RS-232.

Optional kann auf der Leiterplatte des Adapters ein serielles I<sup>2</sup>C EEPROM bestückt werden, das anschlusskompatibel zu den Typen 24LC01, 24LC02 usw. ist.

Der Adapter ist auf der Basis eines SX28 Mikrocontrollers aufgebaut. Daher ist es durch Änderung der Controller-Software leicht möglich, Anpassungen an spezielle Anforderungen vorzunehmen.

---

Ab Version 3.2 wird der Adapter auf einer Leiterplatte in 4-Lagen-Technik ausgeliefert, wobei die beiden inneren Lagen die Signale Vdd (positive Versorgungsspannung) bzw. Vss (Masse) führen. Durch diese Technik wird erreicht, dass die geltenden EMV-Richtlinien eingehalten werden.

**Wichtiger Hinweis:** Das Anbringen nachträglicher Bohrungen auf der Leiterplatte sollte möglichst vermieden werden, um zu verhindern, dass zwischen den beiden Innenlagen Kurzschlüsse entstehen. Sollte dies unvermeidbar sein, ist nach dem Anbringen der Bohrungen unbedingt zu prüfen, ob zwischen der positiven Versorgungsleitung und Masse evtl. ein Kurzschluss besteht. In diesem Fall müssen die Bohrungen vorsichtig aufgerieben werden, bis der Kurzschluss beseitigt ist.

## Elektrische Spezifikationen

Betriebsspannung:	3,3...5 V stabilisiert, ca. 100 mA oder 8...12 V (nicht stabilisiert), ca. 100 mA mit zusätzlichem Festspannungsregler.
Absolute Maximalspannung an allen Anschlüssen:	7 V
Parallel-Eingänge:	TTL-Pegel mit Schmitt-Trigger Verhalten, über Controller-interne Pull-up Widerstände (ca. 10 kOhm) mit +5 V verbunden.
Parallel-Ausgänge:	TTL-Pegel, maximal 30 mA je Ausgang. <b>Achtung:</b> Beim Schalten von induktiven Lasten (z.B. Relais) Freilaufdioden vorsehen.
Serielle-Schnittstelle:	+/- 12 V, entsprechend RS-232.

## Schnittstelle zum PC

Zur Kommunikation mit der PC-Seite stehen die Signale TxD und RxD der RS-232 Schnittstelle zur Verfügung. Hardware-Handshake ist nicht vorgesehen, da alle an den Adapter gesendeten Kommandos mit entsprechenden Antworten quittiert werden. Auf der PC-Seite muss die Schnittstelle wie folgt konfiguriert werden:

38400 Baud, 8 Bits, no Parity, 1 Stop-Bit, kein Handshake. (Im Monitor-Modus wird eine Baudrate von 112500 verwendet – siehe Monitor-Modus).

Die auf dem Adapter vorhandene 9-polige SUB-D Buchse muss über ein serielles Kabel mit einer COM-Schnittstelle des PCs verbunden werden. Wichtig ist, dass ein Kabel ohne getauschte Adern (d.h. kein Null-Modem Kabel) verwendet wird.

## Schnittstelle zum I<sup>2</sup>C-Bus

Zur Kommunikation über den I<sup>2</sup>C-Bus gemäß Herstellerspezifikation der Fa. Philips stehen die Signale

Serielle Daten (SDA)  
Serieller Takt (SCL)

zur Verfügung. Beide Leitungen sind open collector. Auf dem Adapter sind die SDA- und SCL-Leitungen über Pullup-Widerstände von je 2,2 kΩ mit +5 V verbunden. Wird der Adapter mit einem I<sup>2</sup>C-Bus verbunden, der über eigene Pullup-Widerstände verfügt, können die Widerstände auf dem Adapter durch Entfernen zweier Jumper deaktiviert werden.

---

## Parallel Ein-/Ausgänge, Zähler

Der Controller besitzt drei Ein/Ausgabe-Ports, von denen eines (Port A) vier Leitungen besitzt, die für die serielle Kommunikation genutzt werden (RxD und TxD für die serielle Schnittstelle und SDA/SCL für den I<sup>2</sup>C-Bus). Zwei weitere Ports (Port B und Port C) haben jeweils 8 Ein/Ausgangsleitungen, wobei drei Leitungen von Port C zur Steuerung der Status-LEDs dienen. Die verbleibenden fünf Leitungen und die acht Leitungen von Port B sind frei für allgemeine Ein-/Ausgabefunktionen. Mit dem CONFIGURE I/O-Kommando (U) können diese 13 Leitungen individuell als Ein- oder Ausgänge konfiguriert werden. Die Leitungen können über die Erweiterungs-Schnittstelle an den mit „C4...0“ und „B7...0“ markierten Kontakten angeschlossen werden.

Beim Einschalten der Spannungsversorgung, nach einem Reset oder BREAK werden alle Ein-/Ausgabeleitungen automatisch als Eingänge konfiguriert.

Mit den Kommandos IN (N) und OUT (O) können alle Ein-/Ausgabeleitungen parallel gelesen bzw. gesetzt werden. Die Kommandos READ PIN (n) und WRITE PIN (o) ermöglichen es, einzelne Ein-/Ausgabeleitungen zu lesen bzw. zu setzen.

Das OUT-Kommando erwartet zwei Parameter Bytes, wobei das erste die Port C Leitungen steuert und das zweite die Leitungen von Port B. Die oberen drei Bits des ersten Parameters werden ignoriert, da die zugeordneten Leitungen fest als Ausgänge für die Status-LEDs konfiguriert sind. Die übrigen Bits der beiden Parameter steuern nur solche Port-Leitungen, die zuvor als Ausgänge konfiguriert wurden. Als Eingänge konfigurierte Leitungen werden ignoriert.

Das IN-Kommando gibt zwei Bytes zurück, wobei das erste Byte den Status der Port C Leitungen angibt (die höheren drei Bits sind immer 0) und das zweite Byte gibt den Status der Port B Leitungen zurück. Bei Leitungen, die als Ausgang konfiguriert sind, werden die zurückgegebenen Bits entsprechend dem aktuellen Ausgangszustand der Leitungen gesetzt. Bei Eingängen zeigen die Bits den aktuellen Eingangspegel an.

Das READ PIN Kommando erwartet einen Parameter – die Anschlussnummer und es wird der Status der angesprochenen Ein-/Ausgangsleitung zurückgegeben. Die Anschlussnummern von 12...8 bezeichnen die fünf Ein-/Ausgangsleitungen der Port C und die Nummern 8...0 die acht Leitungen des Port B.

Für Leitungen, die als Augänge konfiguriert sind, wird der aktuelle Ausgangspegel zurückgegeben, bei Eingängen entspricht die Rückgabe dem aktuellen Eingangspegel des Anschlusses.

Das WRITE PIN Kommando erwartet zwei Parameter, wobei der erste Parameter wie bei READ PIN die Anschlussnummer definiert und der zweite Parameter den neuen Ausgabepegel angibt. Ist er 0, nimmt der Ausgang low-Pegel an. Jeder andere Wert setzt den Ausgang auf high-Pegel.

Leitungen, die als Eingang konfiguriert sind, werden von diesem Kommando nicht beeinflusst.

Alle Leitungen, die als Eingänge konfiguriert sind, besitzen interne pull-up Widerstände (ca. 10 k $\Omega$ ) und haben Schmitt-Trigger Eigenschaft.

Den acht Port B Leitungen sind acht 16-Bit Zähler zugeordnet. Bei jeder positiven Flanke (low→high) auf einer Leitung wird der zugeordnete Zähler inkrementiert. Es stehen Kommandos zur Verfügung, mit denen die Zähler einzeln oder gesamt gelesen und gelöscht werden können. Die Leitungen werden mit einer Rate von ca. 1 MHz abgetastet und es ist keine Entprellung vorgesehen. Falls erforderlich, müssen externe Komponenten verwendet werden, um „Spitzen“ auszufiltern, die fehlerhafte Zählergebnisse verursachen könnten.

Bei Leitungen, die als Ausgang konfiguriert sind, inkrementieren die Zähler, wenn die zugeordnete Leitung mit einem Ausgabekommando von low nach high gesetzt wird.

---

## Allgemeines zur Kommunikation über I<sup>2</sup>C

Über den I<sup>2</sup>C-Bus kommunizieren ein oder mehrere sogenannte „Master“ mit einem oder mehreren sogenannten „Slaves“. Hier übernimmt jedoch lediglich der Adapter die Master-Funktion und alle I<sup>2</sup>C-Bausteine in der Steuerung müssen als „Slaves“ agieren, d.h. sie reagieren auf entsprechende Kommandos, die vom PC über den Adapter gesendet werden.

Wenn ein Master eine Kommunikation aufbauen will, sendet er nach Erzeugen der Startbedingung (SDA 1→0 bei SCL = 1) zunächst ein Adress- und Modusbyte mit folgendem Aufbau:

```
aaaaaaam
```

wobei aaaaaa die 7-Bit Adresse des Slaves angibt und m definiert, ob eine Schreib- oder Leseoperation ausgeführt werden soll (m = 0: Schreiben, m = 1: Lesen). Der adressierte Slave zieht nach Empfang dieser 8 Bits die SDA-Leitung auf low-Pegel zur Bestätigung (Acknowledge) und der Master prüft, ob die Bestätigung erfolgte. Ist dies nicht der Fall, liegt ein Fehler vor (Slave nicht bereit, nicht vorhanden, defekt oder sonstige Störung).

Wurde vom Master eine Schreiboperation eingeleitet, sendet er nach dem ersten Adress- und Modusbyte ein oder mehrere Datenbytes, die jeweils vom Slave mit einem Acknowledge bestätigt werden müssen. Am Ende der Übertragung erzeugt der Master auf dem Bus eine Stop-Bedingung (SDA 0→1 bei SCL = 1).

Wurde vom Master eine Leseoperation eingeleitet, empfängt er vom Slave ein oder mehrere Bytes und bestätigt jedes (bis auf das letzte Byte) mit einem Acknowledge. Beim letzten Byte sendet der Master kein Acknowledge, um dem Slave zu signalisieren, dass keine weiteren Bytes gesendet werden dürfen. Danach erzeugt der Master auf dem Bus eine Stop-Bedingung (SDA 0→1 bei SCL = 1).

Weitere Möglichkeiten des I<sup>2</sup>C-Bus, wie 10-Bit Adressen, Multi-Master-Betrieb mit Behandlung von Bus-Arbitration sind hier nicht erforderlich und werden nicht unterstützt.

Die Möglichkeit, vor dem eigentlichen Adress- und Modusbyte ein so genanntes „Start Byte“ zu senden, wie es in den neueren I<sup>2</sup>C-Spezifikationen beschrieben ist, besteht ab Firmware-Version 3.7 bei den TX- und RX-Kommandos des Adapters (siehe TX1S, TXnS, RX1S und RXnS ).

**WICHTIG:** Die Kommandos des Adapters, mit denen auf I<sup>2</sup>C-Bausteine zugegriffen werden, erwarten eine 7-Bit Adresse, d.h. das höchste Bit muss grundsätzlich 0 sein. Diese Adresse wird vom Adapter intern um ein Bit nach links verschoben und als niedrigstes Bit wird automatisch je nach Kommando 1 für Lesen oder 0 für Schreiben eingefügt. Wenn z.B. der I<sup>2</sup>C-Baustein ein Adress-/Modusbyte 0xa0 zum Schreiben und 0xa1 zum Lesen erwartet, muss die an den Adapter gesendete Adresse 0x50 lauten, d.h. die oberen sieben Bits des Adress-/Modusbytes (1010000).

## Aufgabe des RS-232 – I<sup>2</sup>C Adapters

Neben der Umsetzung der elektrischen Pegel, d.h. RS-232 mit ±12 V für die Signale TxD und RxD auf der einen und 0/5 V für die Signale SCL und SDA auf der anderen Seite, besteht die Aufgabe des Protokolladapters darin, die Kommunikation über I<sup>2</sup>C mit den Komponenten möglichst transparent für die PC-Anwendung abzuwickeln. Diese kommuniziert mit dem Protokolladapter über einfache Kommandosequenzen, die z.B. in einer VisualBasic-Anwendung realisiert werden.

Hierzu sendet die Anwendung jeweils ein Kommando-Zeichen und ggf. weitere Zeichen an den Adapter. Bei Kommandos, die den Adapter selbst betreffen, wird danach sofort eine Quittung gesendet.

---

Bei I<sup>2</sup>C-Kommandos, löst der Adapter eine Kommunikation mit einem Baustein über den I<sup>2</sup>C-Bus aus. Nach dieser Kommunikation sendet der Adapter zunächst ein Statusbyte zurück, das angibt, ob die Operation erfolgreich war oder ob ein Fehler aufgetreten ist.

War die Operation erfolgreich, sendet der Adapter nach dem Statusbyte je nach Operation ein oder mehrere Bytes, die das Ergebnis enthalten. Im Fehlerfall sendet der Adapter statt dessen den Fehlercode „E“.

## **Ruhezustand**

Nach dem Einschalten der Spannungsversorgung, einem BREAK-Kommando oder nach Timeout oder Reset geht der Adapter in den Ruhezustand, d.h. der I<sup>2</sup>C-Bus bleibt im Idle-Zustand (SDA und SCL auf high-Potential) und es erfolgt keine Kommunikation über den Bus. Alle Kommandos, die über die RS-232 Schnittstelle eintreffen, werden ignoriert und mit „S“ beantwortet (bis auf INIT). Die grüne Status-LED des Adapters blinkt.

Der Ruhezustand wird durch das INIT-Kommando beendet. Mit diesem Kommando werden verschiedene Parameter des Adapters konfiguriert. Nach Empfang des Kommandos nimmt der Adapter andere Kommandos über die RS-232 Schnittstelle entgegen und führt sie entsprechend aus. Die grüne Status-LED des Adapters leuchtet dann dauernd.

Es ist auch möglich, den Adapter vom Ruhezustand direkt in den Monitor-Modus zu schalten (siehe MONITOR-Befehl).

## **Timeout**

Mit dem INIT-Kommando kann u.a. ein Timeout definiert werden. Wenn der Adapter innerhalb des angegebenen Zeitintervalls kein neues Kommando (oder ein unvollständiges Kommando) über die RS-232 Schnittstelle empfängt, geht er automatisch in den Ruhezustand. Die PC-Anwendung sollte ihrerseits eine Zeitüberwachung durchführen. Wenn innerhalb der Zeitgrenzen keine Antworten vom Adapter empfangen werden, sollte die Anwendung ein BREAK-Kommando und danach ein neues INIT-Kommando senden.

---

## Adapter-Kommandos

In der nachfolgenden Aufstellung der verfügbaren Kommandos des Adapters wird folgende Syntax verwendet:

<Zeichen> → = Zeichen, die von der Anwendung an den Adapter gesandt werden  
 → <Zeichen> = Zeichen, die der Adapter als Antwort sendet

Wenn mehrere Alternativen möglich sind, werden diese durch “|” getrennt angegeben.

Wenn für <Zeichen> ein Buchstabe, eine Ziffer oder ein Sonderzeichen angegeben ist, wird dieses als ASCII-Zeichen gesendet bzw. empfangen (*WICHTIG*: Groß- und Kleinschrift sind signifikant). Ist statt dessen der Ausdruck CHR(nnn) angegeben, wird ein Zeichen mit dem ASCII-Code nnn gesendet bzw. empfangen. CR bedeutet, dass das Zeichen Return (13) gesendet werden muss.

Werden mehrere Zeichen nacheinander gesendet bzw. empfangen, sind sie in der nachfolgenden Definition zur besseren Lesbarkeit durch Pluszeichen getrennt dargestellt, diese werden aber nicht gesendet bzw. empfangen.

Angaben in geschweiften Klammern “{“ und “}“ sind optional bzw. mit variabler Anzahl möglich.

### BREAK – Adapter zwangsweise zurücksetzen

Sequenz: Low auf TxD länger als 10 Bitzyklen (> 260 µs)  
 Pause ca. 500 ms  
 → O

Dies ist eine Sonderfunktion, über die der Adapter unabhängig von seinem aktuellen Status zurückgesetzt werden kann. Hierzu ist es erforderlich, dass die PC-Anwendung nach Senden eines Startbits die Leitung TxD für mehr als 10 Bitzyklen der Baudrate von 38400 (> 260 µs) auf low hält. Dadurch wird im Adapter zur erwarteten Zeit kein Stopbit empfangen und es tritt ein sogenannter „Framing Error“ auf, der den Adapter zu einem Reset veranlasst.

Am einfachsten kann dies in der PC-Anwendung erreicht werden, indem mit einer niedrigen Baudrate (z.B. 300 Baud) das Zeichen CHR(0) gesendet wird oder die TXD-Leitung für eine entsprechende Zeit auf low-Pegel gehalten wird. In Visual Basic steht für das MSComm-Objekt hierzu die Break-Methode zur Verfügung.

Nachdem der Adapter ein BREAK empfangen hat, sendet er als Bestätigung das Zeichen „O“ an den PC und geht danach in den Ruhezustand, solange bis ein INIT-Kommando empfangen wird.

Werden während des Ruhezustandes andere Kommandos an den Adapter gesendet, antwortet dieser mit dem Zeichen „S“.

### CLEAR ALL COUNTERS – Alle Zähler löschen

Sequenz: a →  
 → O

Funktion: Löscht alle Zähler.

Rückgabe: O = Löschen erfolgreich

---

**CLEAR COUNTER – Zähler löschen**

Sequenz:	c + CHR(<n>) → → O   E
Zulässige Werte für <n>:	0..7
Funktion:	Löscht den Zähler, der dem mit <n> angegebenen Eingang (0..7) zugeordnet ist.
Rückgabe:	O = Rücksetzen erfolgreich E = Falscher Wert für <n>

**CONFIGURE I/O PINS**

Sequenz:	U + CHR(CfgC) + CHR(CfgB)→ → O
Funktion:	Konfiguriert die Ein-/Ausgangsleitungen, wobei der CfgC-Parameter die Konfiguration von Port C und der CfgB-Parameter die von Port B angibt. Wenn ein Parameter-Bit 0 ist, wird die zugeordnete Portleitung als Ausgang konfiguriert. Die oberen drei Bits des CfgC-Parameters werden ignoriert und intern immer gelöscht, um diese drei Leitungen als Ausgänge für die Status-LEDs zu konfigurieren.  Leitungen, die als Ausgänge konfiguriert werden, nehmen nach Ausführung dieses Kommandos automatisch low-Level an.  Nach Anlegen der Spannungsversorgung, einem BREAK oder Reset werden alle Portleitungen automatisch als Eingänge konfiguriert.
Rückgabe:	O = Erfolg

**COUNTER READ – Zählerinhalt lesen**

Sequenz:	C + CHR(<n>) → → O + CHR(<h>) + CHR(<l>)   E + 0 + 0
Zulässige Werte für <n>:	0..7
Funktion:	Liest den Inhalt des Zählers, der der mit <n> angegebenen Port B Leitung (0..7) zugeordnet ist.
Rückgabe:	O = Lesen erfolgreich, <h> enthält das high-Byte und <l> das low-Byte des aktuellen Zählerstands. E = Falscher Wert für <n>

**COUNTER READ ALL – Alle Zählerinhalte lesen**

Sequenz: A →  
 → O + CHR(<h<sub>7</sub>>) + CHR(<l<sub>7</sub>>) + CHR(<h<sub>6</sub>>) + CHR(<l<sub>6</sub>>) +  
 CHR(<h<sub>5</sub>>) + CHR(<l<sub>5</sub>>) + CHR(<h<sub>4</sub>>) + CHR(<l<sub>4</sub>>) +  
 CHR(<h<sub>3</sub>>) + CHR(<l<sub>3</sub>>) + CHR(<h<sub>2</sub>>) + CHR(<l<sub>2</sub>>) +  
 CHR(<h<sub>1</sub>>) + CHR(<l<sub>1</sub>>) + CHR(<h<sub>0</sub>>) + CHR(<l<sub>0</sub>>)

Funktion: Liest Inhalte aller Zähler.

Rückgabe: O = Lesen erfolgreich, die nachfolgenden 16 Bytes enthalten die Inhalte der Zähler, beginnend mit dem Zähler für Port B Leitung 7 bis Leitung 0, jeweils zuerst das high- und dann das low-Byte.

**INIT – Adapter initialisieren**

Sequenz: I + <c> + CHR(<to>) + CR →  
 → O + <hhl> | E + 000

<c> = I<sup>2</sup>C Bitrate kBit/s  
 <to> = Timeout-Wert  
 <hhl> = Versionsnummer des Adapters (3-stellige ASCII-Zeichenfolge  
 hh = Hauptversion  
 l = Unterversion

z.B.: 038 = Version 3.8

Zulässige Werte für <c>:  
 0 = 25 kBit/s  
 1 = 50 kBit/s  
 2 = 100 kBit/s  
 3 = 200 kBit/s  
 4 = 400 kBit/s  
 5 = 3 kBit/s

Zulässige Werte für <to>: 0...255 Bei <to> = 0 ist die Zeitüberwachung ausgeschaltet, ansonsten bestimmt <to> die Wartezeit bis zu einem Timeout in 100 ms-Schritten, d.h. es können Zeiten zwischen 100 ms bis zu 25,5 s eingestellt werden.

Funktion: Initialisiert den Adapter und beendet bei erfolgreicher Initialisierung den Ruhezustand, d.h. die Kommunikation wird freigegeben. Werden innerhalb des mit <to> definierten Zeitintervalls keine weiteren Kommandos empfangen, geht der Adapter wieder in den Ruhezustand, aus dem er nur durch ein erneutes INIT-Kommando gebracht werden kann.

Rückgabe: O + <hhl> = Initialisierung erfolgreich, <hhl> enthält als 3-stellige ASCII-Zeichenfolge die Versions-Nr. des Adapters, z.B. 038 entspricht Version 3.8  
 E = Fehler - es wurde eine unzulässige Anzahl von Parametern oder der falsche Sicherheitscode übertragen.

**INPUT – Portleitungen lesen**

Sequenz: N  
→ O + CHR(<valC>) + CHR(<valB>)

Funktion: Liest die Port C und Port B Ein-/Ausgangsleitungen und gibt das entsprechende Bitmuster in <valC> und <valB> zurück. Bits 7...5 in valC sind immer 0, da diese Leitungen für die Status-LEDs des Adapters reserviert sind.

Beachten Sie, dass offene Eingangsleitungen high als Ergebnis liefern, da die Controller-internen Pull-Up Widerstände aktiviert sind.

Rückgabe: O + CHR(<valC>) + CHR(<valB>)

**MONITOR – Monitor-Modus aktivieren**

Sequenz: M →  
→ CHR<byte> + +|- + { CHR<byte> + +|- + ... }

Funktion: Dieses Kommando aktiviert den Monitor-Modus des Adapters. In diesem Modus werden alle Parallel-Ausgänge in den hochohmigen Zustand geschaltet. Die grüne Status-LED blinkt und die gelbe I<sup>2</sup>C-Aktivitäts-LED leuchtet ständig.

Der Adapter überwacht dann den I<sup>2</sup>C-Bus. Wird eine Bus-Aktivität erkannt, überträgt er das auf dem Bus gesendete Byte über die RS-32-Schnittstelle, gefolgt von einem „+“ wenn ein Acknowledge-Bit erkannt wurde oder einem „-“, wenn kein Acknowledge-Bit erkannt wurde.

Wird eine Stop-Bedingung erkannt, sendet der Adapter die spezielle Zeichenfolge CR/LF zurück. Das PC-Programm kann diese herausfiltern und so z.B. im Monitor-Protokoll eine neue Zeile zu beginnen.

Zum Beenden des Monitor-Modus muss ein BREAK-Kommando gesendet werden, alle anderen Kommandos werden ignoriert.

Um den Monitor-Modus zu aktivieren, ist es nicht erforderlich, den Adapter zuvor zu initialisieren, d.h. es kann direkt nach dem Einschalten der Stromversorgung das MONITOR-Kommando gesendet werden.

**WICHTIG:** Bei aktivem Monitor-Modus überträgt der Adapter die Daten mit 112500 Baud über die COM-Schnittstelle, um eine möglichst hohe Übertragungsrate zu erreichen. Die PC-Software muss nach dem Aufruf des Monitor-Modus die PC-seitige COM-Schnittstelle entsprechend konfigurieren.

*Anmerkung:* Der Adapter verwendet einen 32-Byte FIFO Puffer für die Bytes und Acknowledge-Zustände. Damit ist es möglich, Bussysteme mit einer Datenrate von bis zu 100 kBit/s zu überwachen, vorausgesetzt, dass nicht ständig Kommunikation über den Bus erfolgt, da in diesem Fall die Zeichen nicht schnell genug über die RS-232-Schnittstelle übertragen werden können. Wenn der FIFO-Puffer voll ist, werden einige I<sup>2</sup>C-Aktivitäten nicht erkannt.

**OUTPUT – Portleitungen setzen**

Sequenz: O + CHR(<valC>) + CHR(<valB>)  
→ O

Funktion: Setzt die unteren fünf Port C Leitungen und die acht PortB Leitungen entsprechend dem Bitmuster in <valC> und <valB>. Die oberen drei Bits von <valC> werden ignoriert, da diese Leitungen den Status-LEDs zugeordnet sind.

Es werden nur Leitungen beeinflusst, die zuvor mit CONFIGURE I/O PINS (U) als Ausgang konfiguriert wurden.

Rückgabe: O

**PING – Abfrage**

Sequenz: P →  
→ O

Funktion: Prüft, ob der Adapter vorhanden, bzw. betriebsbereit ist – wenn dies der Fall ist, antwortet er mit „O“, ansonsten erfolgt keine Antwort (dies muss in der Anwendung über einen Timeout abgefangen werden).

Rückgabe: O

**READ PIN**

Sequenz: n + CHR(<pin-#>) →  
→ O + CHR(<pin status>) | E

Funktion: Dieses Kommando liest den Status einer Ein-/Ausgabeleitung. Es erwartet die Leitungs-Nummer <pin-#> als Parameter. Gültige Werte sind 12...8 für die unteren fünf Port C Leitungen und 7...0 für die acht Port B Leitungen. Werte über 12 werden mit einem Fehlercode quittiert.

Bei Leitungen, die als Eingang konfiguriert sind, wird der auf der Leitung eingespeiste Pegel zurückgegeben, bei Ausgängen der aktuelle Ausgangszustand.

Rückgabe: O = Lesen erfolgreich, Leitungs-Status folgt (0 = low, 1 = high)  
E = Es wurde eine falsche Leitungs-Nummer übergeben.

**READ BYTE WITH ACK – Byte mit Acknowledge lesen**

Sequenz: E →  
→ CHR(<byte>)

Funktion: Dieses „low-level“-Kommando liest ein Byte (<byte>, 0...255) von dem I<sup>2</sup>C-Baustein, der zuletzt zum Lesen adressiert wurde und setzt anschließend die Acknowledge-Bedingung (Acknowledge-Bit low),

Damit mit diesem Kommando sinnvolle Daten gelesen werden, ist es erforderlich, zuerst die Adresse des Bausteins zu senden, der gelesen werden soll (siehe SEND READ ADDRESS-Kommandos). Ohne

vorherige Adressierung eines gültigen I<sup>2</sup>C-Bausteins, wird dieses Kommando immer CHR(255) oder 0xFF zurückgeben, da die SDA-Leitung von keinem Baustein gesteuert wird und damit immer auf High-Pegel liegt.

Es können danach beliebig viele READ BYTE WITH ACK-Kommandos, z.B. zum sequentiellen Auslesen eines EEPROMS ausgeführt werden. Bei den meisten Bausteinen muss das sequentielle Lesen mit einem Lesen ohne Acknowledge beendet werden (siehe READ BYTE WITHOUT ACK).

Rückgabe: <byte>

### READ BYTE WITHOUT ACK – Byte ohne Acknowledge lesen

Sequenz: e →  
→ CHR(<byte>)

Funktion: Dieses „low-level“-Kommando liest ein Byte (<byte>, 0...255) von dem I<sup>2</sup>C-Baustein, der zuletzt zum Lesen adressiert wurde und setzt anschließend keine Acknowledge-Bedingung (Acknowledge-Bit high),

Damit mit diesem Kommando sinnvolle Daten gelesen werden, ist es erforderlich, zuerst die Adresse des Bausteins zu senden, der gelesen werden soll (siehe SEND READ ADDRESS-Kommandos). Ohne vorherige Adressierung eines gültigen I<sup>2</sup>C-Bausteins, wird dieses Kommando immer CHR(255) oder 0xFF zurückgeben, da die SDA-Leitung von keinem Baustein gesteuert wird und damit immer auf High-Pegel liegt.

Dieses Kommando dient meistens zum Beenden eines sequentiellen Lesens, wobei alle Bytes mit READ BYTE WITH ACK gelesen werden, bis auf das letzte Byte, dass mit diesem Kommando gelesen wird. Durch das fehlende Acknowledge erkennt der Baustein, dass keine weiteren Daten gelesen werden sollen.

Rückgabe: <byte>

### RX1 – ein Byte empfangen

Sequenz: R + CHR(<adr>) →  
→ O + CHR(<val>) | E

Funktion: Diese „high-level“-Funktion erzeugt zunächst die Startbedingung, sendet die 7-Bit Bausteinadresse <adr> mit gesetztem R/\*W-Bit, liest dann den 8-Bit Wert <val> (0...255), setzt danach das Acknowledge-Bit (kein Acknowledge) und setzt abschließend die Stop-Bedingung.

Rückgabe: O = Lesen erfolgreich, gelesenes Byte folgt  
E = Fehler (keine Bestätigung vom I<sup>2</sup>C-Bus oder Adresse > 127)

*Anmerkung:* Dies ist eine Sonderform der RXN-Funktion, bei der die Angabe des Längenbytes entfällt, so dass die Kommunikation etwas schneller abläuft.

## RX1S – Startbyte senden und ein Byte empfangen

Sequenz: G + CHR(<adr>) →  
→ O + CHR(<val>) | E

Funktion: Diese „high-level“-Funktion sendet zunächst die Startbedingung, sendet dann ein Start-Byte (0x01) und ignoriert das Acknowledge-Bit. Danach wird erneut die Startbedingung gesetzt, die 7-Bit Bausteinadresse <adr> mit gesetztem R/\*W-Bit gesendet, und dann der 8-Bit Wert <val> (0...255) gelesen. Abschließend wird das Acknowledge-Bit gesetzt (kein Acknowledge) und abschließend die Stop-Bedingung gesendet.

Rückgabe: O = Lesen erfolgreich, gelesenes Byte folgt  
E = Fehler (keine Bestätigung vom I<sup>2</sup>C-Bus oder Adresse > 127)

*Anmerkung:* Dies ist eine Sonderform der RXnS-Funktion, bei der die Angabe des Längenbytes entfällt, so dass die Kommunikation etwas schneller abläuft.

## RXN – mehrere Bytes empfangen

Sequenz: r + CHR(<adr>) + CHR(<n>) →  
→ O + CHR(<val1>) {+ CHR(<val2>) + ...} | E

Funktion: Diese „high-level“-Funktion erzeugt zunächst die Startbedingung, sendet die 7-Bit Bausteinadresse <adr> mit gesetztem R/\*W-Bit, liest dann <n-1> 8-Bit Werte <valx> (0...255) und erzeugt jeweils ein Acknowledge. Das <n>-te Byte wird dagegen ohne Acknowledge gelesen und abschließend wird die Stop-Bedingung gesetzt.

Rückgabe: O = Lesen erfolgreich, gelesene Bytes folgen  
E = Fehler (keine Bestätigung vom I<sup>2</sup>C-Bus oder <n> < 1 oder <n> > 16 oder Adresse > 127).

## RXnS – Start-Byte senden und mehrere Bytes empfangen

Sequenz: g + CHR(<adr>) + CHR(<n>) →  
→ O + CHR(<val1>) {+ CHR(<val2>) + ...} | E

Funktion: Diese „high-level“-Funktion sendet zunächst die Startbedingung, sendet dann ein Start-Byte (0x01) und ignoriert das Acknowledge-Bit. Die Funktion erzeugt dann eine weitere Startbedingung, sendet die 7-Bit Bausteinadresse <adr> mit gesetztem R/\*W-Bit, liest danach <n-1> 8-Bit Werte <valx> (0...255) und erzeugt jeweils ein Acknowledge. Das <n>-te Byte wird dagegen ohne Acknowledge gelesen und abschließend wird die Stop-Bedingung gesetzt.

Rückgabe: O = Lesen erfolgreich, gelesene Bytes folgen  
E = Fehler (keine Bestätigung vom I<sup>2</sup>C-Bus oder <n> < 1 oder <n> > 16 oder Adresse > 127).

---

## SEND BYTE – Byte senden

Sequenz: B + CHR(<Byte>) →  
→ O | E

Funktion: Dieses "low-level"-Kommando sendet ein Byte über den I<sup>2</sup>C-Bus. Vor Ausführung dieses Kommandos ist es erforderlich, eine Adresse zu senden (siehe SEND ADDRESS-Kommandos), um den I<sup>2</sup>C-Baustein auszuwählen, der das gesendete Byte empfangen soll. Danach kann eine beliebige Anzahl von SEND BYTE-Kommandos ausgeführt werden, z.B. um einen Datenblock in ein EEPROM zu schreiben.

Rückgabe: O = Ein Baustein hat mit Acknowledge quittiert  
E = Kein Baustein hat mit Acknowledge quittiert

Wenn ständig „E“ zurückgegeben wird, wurde evtl. vergessen, zuvor eine Adresse zu senden oder es wurde die Adresse eines nicht vorhandenen Bausteins gesendet und der dabei aufgetretene Fehlercode ignoriert.

## SEND READ ADDRESS WITH START – Leseadresse mit Start-Bedingung senden

Sequenz: D + CHR(<addr>) →  
→ O | E

Funktion: Dieses „low-level“-Kommando setzt die Startbedingung auf dem Bus und sendet dann eine Leseadresse. Beachten Sie, dass <addr> zwischen 0 und 127 liegen muss. Dieser Wert wird automatisch um eine Bit-Position nach links geschoben und das niedrigste Bit (das R/\*W-Bit) wird gesetzt.

Rückgabe: O = Ein Baustein hat mit Acknowledge quittiert  
E = Kein Baustein hat mit Acknowledge quittiert

## SEND READ ADDRESS WITHOUT START – Leseadresse ohne Start-Bedingung senden

Sequenz: d + CHR(<addr>) →  
→ O | E

Funktion: Dieses „low-level“-Kommando sendet eine Leseadresse, ohne zuvor die Startbedingung zu setzen. Beachten Sie, dass <addr> zwischen 0 und 127 liegen muss. Dieser Wert wird automatisch um eine Bit-Position nach links geschoben und das niedrigste Bit (das R/\*W-Bit) wird gesetzt.

Einige I<sup>2</sup>C-Bausteine erwarten zunächst die Startbedingung, gefolgt von einer Schreibadresse und einem Konfigurationsbyte und danach eine Leseadresse ohne erneute Startbedingung. Sie können die Kommandos SEND WRITE ADDRESS WITH START, SEND BYTE und SEND READ ADDRESS WITHOUT START zu diesem Zweck nacheinander ausführen. Normalerweise folgen danach mehrere READ BYTE-Kommandos, um Daten des I<sup>2</sup>C-Bausteins zu lesen.

---

Rückgabe: O = Ein Baustein hat mit Acknowledge quittiert  
E = Kein Baustein hat mit Acknowledge quittiert

### SEND WRITE ADDRESS WITH START – Schreibadresse mit Start-Bedingung senden

Sequenz: W + CHR(<addr>) →  
→ O | E

Funktion: Dieses „low-level“-Kommando setzt die Startbedingung auf dem Bus und sendet dann eine Schreibadresse. Beachten Sie, dass <addr> zwischen 0 und 127 liegen muss. Dieser Wert wird automatisch um eine Bit-Position nach links geschoben und das niedrigste Bit (das R/\*W-Bit) wird gelöscht.

Rückgabe: O = Ein Baustein hat mit Acknowledge quittiert  
E = Kein Baustein hat mit Acknowledge quittiert

### SEND WRITE ADDRESS WITHOUR START – Schreibadresse ohne Start-Bedingung senden

Sequenz: W + CHR(<addr>) →  
→ O | E

Funktion: Dieses „low-level“-Kommando sendet eine Schreibadresse, ohne zuvor die Startbedingung auf dem Bus zu setzen. Beachten Sie, dass <addr> zwischen 0 und 127 liegen muss. Dieser Wert wird automatisch um eine Bit-Position nach links geschoben und das niedrigste Bit (das R/\*W-Bit) wird gelöscht.

Rückgabe: O = Ein Baustein hat mit Acknowledge quittiert  
E = Kein Baustein hat mit Acknowledge quittiert

### STOP – Stop-Bedingung setzen

Sequenz: S →  
→ O

Funktion: Setzt auf dem I<sup>2</sup>C-Bus die Stop-Bedingung. Dies ist eine Hilfsfunktion, die z.B. nach einem erkannten Fehler aufgerufen werden kann, um alle Slaves der Steuerung in den Ausgangszustand zu versetzen.

Rückgabe: O

### TX1 – Ein Byte senden

Sequenz: T + CHR(<adr>) + CHR(<val>) →  
→ O | E

Funktion: Dieses „high-level“-Kommando setzt die Startbedingung, sendet die 7-Bit Adresse <adr> mit gelöschtem R/\*W-Bit, dann den 8-Bit Wert <val> (0...255), liest das Acknowledge-Bit und setzt abschließend die Stop-Bedingung.

Rückgabe: O = Übertragung erfolgreich  
E = Fehler (keine Bestätigung vom I<sup>2</sup>C-Bus oder <adr> > 127)

*Anmerkung:* Dies ist eine Sonderform der TXN-Funktion, bei der die Angabe des Längenbytes entfällt, so dass die Kommunikation etwas schneller abläuft.

### TX1S – Start-Byte und ein Byte senden

Sequenz: F + CHR(<adr>) + CHR(<val>) →  
→ O | E

Funktion: Diese „high-level“-Funktion sendet zunächst die Startbedingung, sendet dann ein Start-Byte (0x01) und ignoriert das Acknowledge-Bit. Die Funktion setzt danach erneut die Startbedingung, sendet die 7-Bit Adresse <adr> mit gelöschtem R/\*W-Bit, dann den 8-Bit Wert <val> (0...255), liest das Acknowledge-Bit und setzt abschließend die Stop-Bedingung.

Rückgabe: O = Übertragung erfolgreich  
E = Fehler (keine Bestätigung vom I<sup>2</sup>C-Bus oder <adr> > 127)

*Anmerkung:* Dies ist eine Sonderform der TXnS-Funktion, bei der die Angabe des Längenbytes entfällt, so dass die Kommunikation etwas schneller abläuft.

### TXN – mehrere Bytes senden

Sequenz: t + CHR(<adr>) + CHR(<n>) + CHR(<val1>) {+ CHR(<val2>) + ...} →  
→ O | E

Funktion: Dieses „high-level“-Kommando setzt die Startbedingung, sendet die 7-Bit Adresse <adr> mit gelöschtem R/\*W-Bit, dann <n> 8-Bit Werte <valx> (0...255), liest jeweils die Acknowledge-Bit und setzt abschließend die Stop-Bedingung.

Rückgabe: O = Übertragung erfolgreich  
E = Fehler (keine Bestätigung vom I<sup>2</sup>C-Bus oder <n> < 1 oder <adr> > 127)

### TXnS – mehrere Bytes senden

Sequenz: t + CHR(<adr>) + CHR(<n>) + CHR(<val1>) {+ CHR(<val2>) + ...} →  
→ O | E

Funktion: Diese „high-level“-Funktion sendet zunächst die Startbedingung, sendet dann ein Start-Byte (0x01) und ignoriert das Acknowledge-Bit. Die Funktion setzt dann erneut die Startbedingung, sendet die 7-Bit Adresse <adr> mit gelöschtem R/\*W-Bit, dann <n> 8-Bit Werte <valx> (0...255), liest jeweils die Acknowledge-Bit und setzt abschließend die Stop-Bedingung.

Rückgabe: O = Übertragung erfolgreich  
E = Fehler (keine Bestätigung vom I<sup>2</sup>C-Bus oder <n> < 1 oder <adr> > 127)

---

## WRITE PIN

Sequenz:                   o + CHR(<pin-#>) + CHR(<setting>) →  
                                  → O | E

Funktion:                   Dieses Kommando setzt den Zustand einer Ausgangsleitung. Es erwartet zwei Parameter, wobei der erste die Leitungsnummer angibt. Gültige Werte sind 12...8 für die unteren fünf Port C Leitungen und 7...0 für die acht Port B Leitungen. Werte über 12 werden nicht akzeptiert und mit einem Fehlercode zurückgewiesen.

Der zweite Parameter gibt den Ausgangszustand an. Wenn er 0 ist, wird die Ausgangsleitung auf low Level gesetzt, jeder andere Wert bewirkt, dass die Leitung auf high Level gesetzt wird.

Wird das Kommando auf Leitungen angewandt, die als Eingänge konfiguriert sind, erfolgt keine Fehlermeldung, aber der Zustand der Leitung wird nicht verändert.

Rückgabe:                   O = Kommando erfolgreich  
                                  E = Falsche Leitungsnummer

## Fehlerhafte Kommandos

Sequenz:                   <?> →  
                                  → ?

Funktion:                   Werden nicht definierte Kommandos an den Adapter gesendet, antwortet dieser mit „?“.

## Zusammenfassung der Adapter -Funktionen

Kenner	Funktion
ohne	BREAK – Adapter zwangsweise zurücksetzen
a	CLEAR ALL COUNTERS – Alle Zähler zurücksetzen
A	COUNTER READ ALL – Alle Zähler lesen
B	SEND BYTE – Byte senden und Acknowledge-Status zurückgeben
c	CLEAR COUNTER – Zähler zurücksetzen
C	COUNTER READ – Zählerinhalt lesen
d	SEND ADDRESS FOR READ WITHOUT START – Leseadresse ohne vorherige Startbedingung senden
D	SEND ADDRESS FOR READ WITH START – Startbedingung setzen und dann eine Leseadresse senden
e	READ BYTE WITHOUT ACK – Ein Byte lesen, kein Acknowledge senden
E	READ BYTE WITH ACK – Ein Byte lesen, Acknowledge senden
f	TXnS – Start-Byte und mehrere Bytes über I <sup>2</sup> C senden
F	TX1S – Start-Byte und ein Byte über I <sup>2</sup> C senden
g	RXnS – Start-Byte senden und mehrere Bytes über I <sup>2</sup> C empfangen
G	RX1S – Start-Byte senden und ein Byte über I <sup>2</sup> C empfangen
I	INIT – Adapter initialisieren
M	MONITOR – Monitor-Modus aktivieren
n	READ PIN – Ein-/Ausgangsleitung lesen
N	INPUT – Eingangsport des Adapters lesen
o	WRITE PIN – Ausgangsleitung schreiben
O	OUTPUT – Ausgangsport des Adapters setzen
P	PING – Abfrage, ob betriebsbereit
r	RXN – Mehrere Bytes über I <sup>2</sup> C empfangen
R	RX1 – Ein Byte über I <sup>2</sup> C empfangen
S	STOP – I <sup>2</sup> C-Bus auf Stop-Bedingung setzen
t	TXN – Mehrere Bytes über I <sup>2</sup> C senden
T	TX1 – Ein Byte über I <sup>2</sup> C senden
U	CONFIGURE I/O PINS – Port Ein-/Ausgangsleitungen konfigurieren
w	SEND WRITE ADDRESS WITHOUT START – Schreibadresse ohne vorherige Startbedingung senden
W	SEND WRITE ADDRESS WITH START – Start-Bedingung setzen und Schreibadresse senden

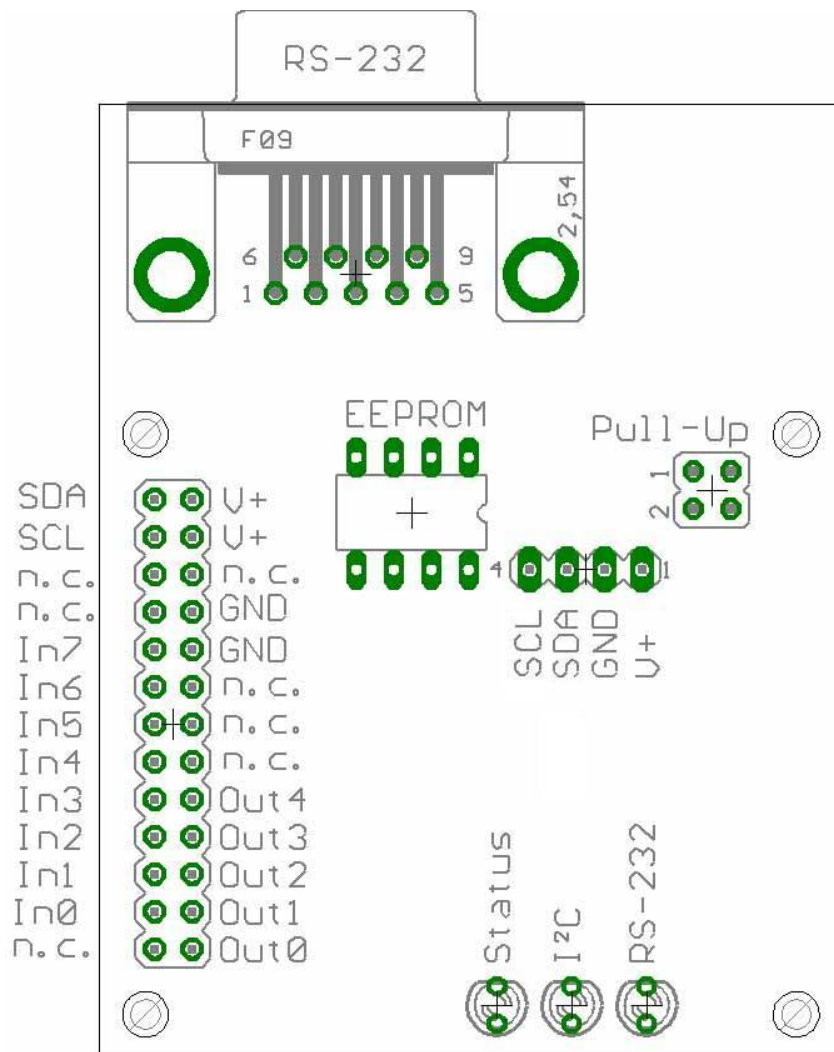
Anm.: „Low-Level“-Kommandos sind grau hinterlegt.

## Anschlussbelegung des Adapters

Alle Anschlüsse des Adapters (bis auf die der seriellen Schnittstelle) liegen auf einer 26-poligen Stiftleiste (2 \* 13) im Rastermaß 2,54 mm. Die Signale für die serielle Schnittstelle (TxD, RxD und GND) stehen an einem 9-poligen SUB-D Stecker zur Verfügung.

Zusätzlich ist ein 4-poliger Steckverbinder im Rastermaß 2,54 mm vorhanden, über den die Signale SDA und SCL des I<sup>2</sup>C Bus, sowie die Versorgungsspannungsleitungen Vdd (+3,3...5V) und Vss (Masse) zur Verfügung stehen.

Im rechten Bereich der Leiterplatte (in der folgenden Abbildung nicht gezeigt), kann bei Bedarf ein Festspannungsregler, zusammen mit den notwendigen Filterkondensatoren und Schutzdioden nachgerüstet werden. Hier kann auch eine DIN-Niederspannungsbuchse bestückt werden, über die der Adapter mit einer unregelmäßigen Gleichspannung zwischen 8 und 12 Volt versorgt werden kann. Wird der Festspannungsregler nicht benötigt, kann dieser Teil der Leiterplatte abgetrennt werden, wenn kleinere Abmessungen der Platine wichtig sind (76 \* 56 mm statt 76 \* 65 mm).



Anschlüsse des Adapters von der Bestückungsseite gesehen

---

## 26-polige Stiftleiste

Auf der 26-poligen Stiftleiste stehen alle parallelen Ein- und Ausgänge des Adapters und der I<sup>2</sup>C Bus (SDA/SCL) zur Verfügung. Über die Stifte „V+“ und „GND“ kann die Spannungsversorgung des Adapters erfolgen (3,3...5 V, ca. 100 mA, stabilisiert). Die mit „n.c.“ bezeichneten Stifte sollten nicht extern beschaltet werden, da sie mit Ein-/Ausgangsleitungen des Controllers verbunden sind, die intern genutzt werden.

## 4-polige Stiftleiste

Alternativ kann der Adapter über die 4-polige Stiftleiste mit dem I<sup>2</sup>C-Bus verbunden werden. Eine Spannungsversorgung des Adapters ist über diesen Anschluss ebenfalls möglich.

## Jumper „Pull-Up“

Wenn im 4-poligen Anschlussfeld „Pull-Up“ die beiden oberen und die beiden unteren Stifte mit Jumpern verbunden werden, sind die auf dem Adapter vorhandenen Pull-Up Widerstände für die Leitungen SDA und SCL des I<sup>2</sup>C-Bus aktiv. Ohne Jumper sind die Widerstände nicht aktiv. In diesem Fall müssen an mindestens einer anderen Stelle des I<sup>2</sup>C-Bus entsprechende Pull-Up-Widerstände vorhanden sein.

## SUB-D 9 Buchse „RS-232“

Diese Buchse dient zum Anschluss des Adapters an die COM-Schnittstelle eines PCs. Es sind lediglich drei Kontakte angeschlossen:

Stift 2: Adapter TxD → PC RxD

Stift 3: Adapter RxD → PC TxD

Stift 5: Signal Ground

## Leuchtdioden

Die grüne LED „Status“ signalisiert den Status des Adapters:

Blinken = Adapter nicht initialisiert oder im Monitor-Modus

Dauernd ein = Adapter initialisiert

Die gelbe LED „I<sup>2</sup>C“ flackert, wenn Aktivitäten auf dem I<sup>2</sup>C-Bus erfolgen. Sie leuchtet dauernd, wenn der Monitor-Modus des Adapters aktiv ist.

Die rote LED „RS-232“ flackert, wenn Aktivitäten auf der seriellen Schnittstelle erfolgen.

SUB-D 9 Buchse „RS-232“

Über dies Buchse wird der Adapter mit der COM-Schnittstelle eines PC verbunden. Es gilt folgende Belegung:

Stift 2: Adapter TxD → PC RxD (Stift 2)

Stift 3: Adapter RxD → PC TxT (Stift 3)

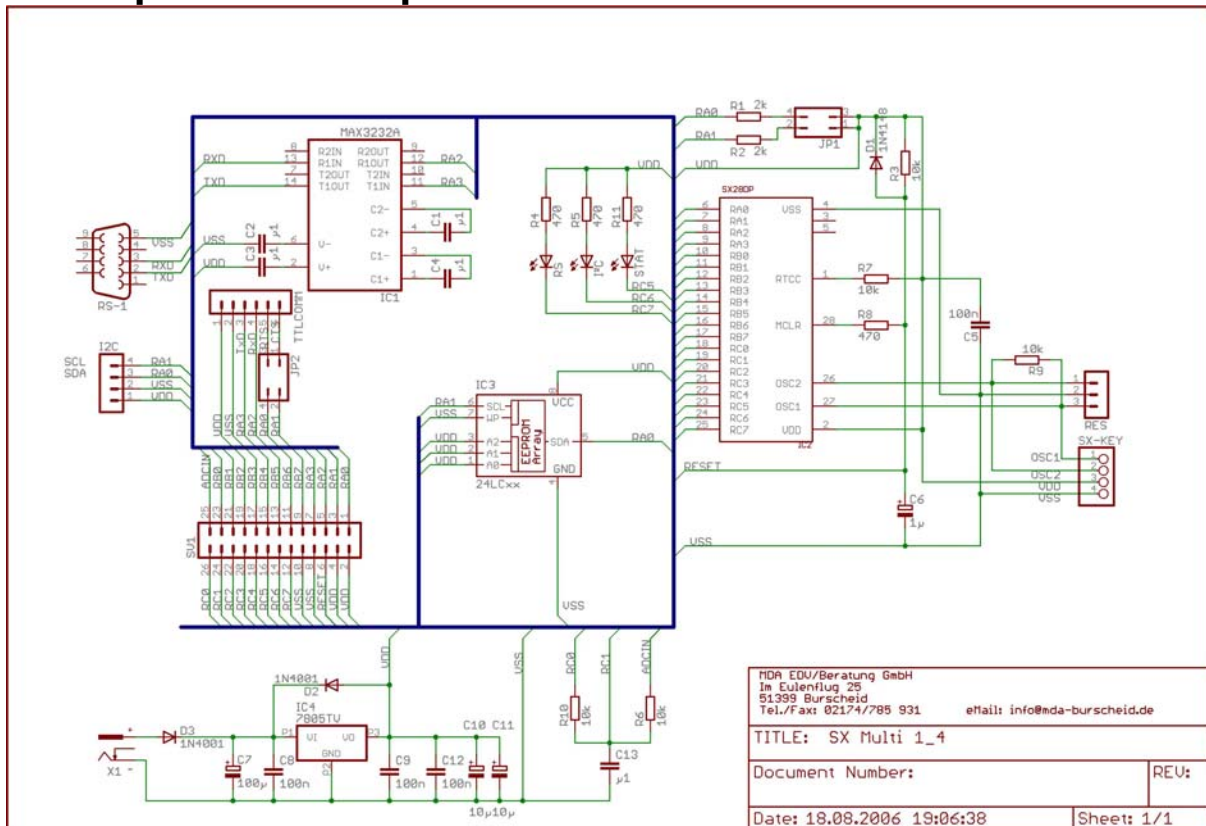
Stift 5: Adapter Signal Ground → PC Signal Ground

---

## Bestückungsposition „EEPROM“

An dieser Stelle kann optional ein serielles I<sup>2</sup>C EEPROM bestückt werden, das anschluss-kompatibel zu den Typen 24LC01, 24LC02 usw. ist. Die drei Adressbits A0, A1 und A2 (Anschlüsse 1, 2 und 3) sind fest mit Vdd (plus) verbunden, so dass ein hier eingesetzter EEPROM-Baustein die Baustein-Adresse 111 erhält.

## Schaltplan des Adapters

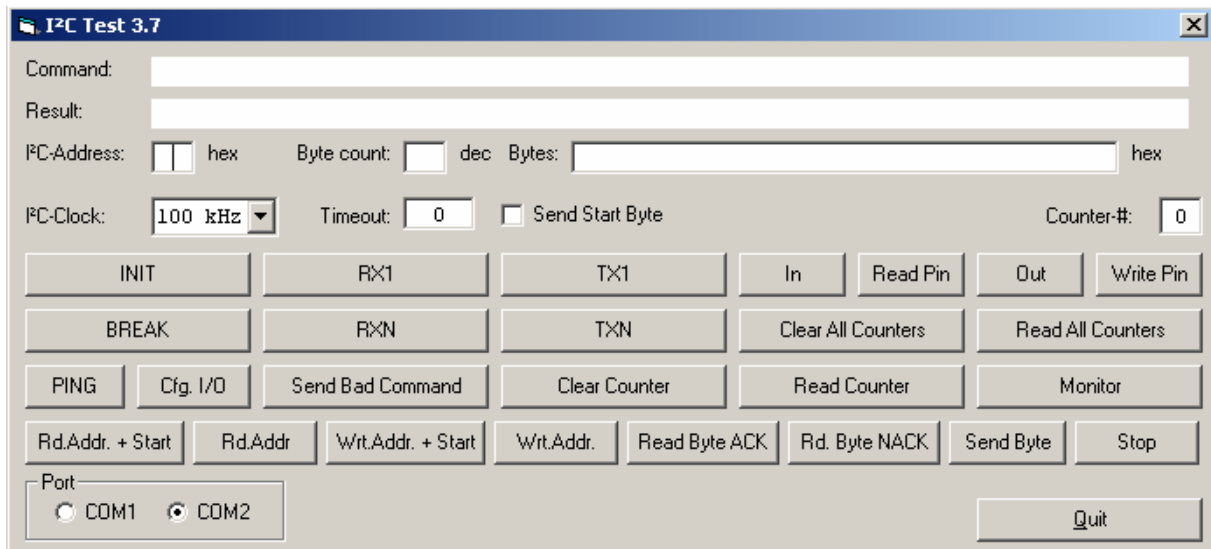


**Hinweis:** Je nach Ausführung des Adapters sind nicht alle im Schaltplan gezeigten Bauteile bestückt.

## Testprogramm für den Adapter

Das auf dieser CD enthaltene Testprogramm ermöglicht es, die Funktionen des Adapters mit einem PC zu testen. Dieses Programm wurde in Microsoft Visual-BASIC erstellt – die Projektdateien finden Sie ebenfalls auf dieser CD als Anregung für eigene Anwendungen.

Nach dem Start des Programms erscheint folgende Anzeige:



Unter der Voraussetzung, dass der Adapter mit Spannung versorgt wird (die grüne LED blinkt) und dass die serielle Schnittstelle mit einem COM-Port des PCs verbunden ist (COM1 oder COM2) können folgende Operationen ausgeführt werden:

### Auswahl des COM-Ports:

Wählen Sie im Bereich „Port“ das COM-Port des PCs aus, mit dem der Adapter verbunden ist.

Mit den nachfolgend beschriebenen Schaltflächen werden verschiedene Kommandos an den Adapter gesendet. In der Zeile „Command“ erscheinen jeweils die gesendeten Zeichen (Daten in spitzen Klammern <> sind hexadezimalwerte). In der Zeile „Result“ wird die Rückgabe des Adapters angezeigt.

### INIT

Die Schaltfläche „INIT“ löst das Senden des INIT-Kommandos aus. Sie sollten die Vorgaben in den Bereichen I2C-Clock, „Timeout“ und „Code“ beibehalten, ggf. können Sie vorher eine andere I2C-Busrate auswählen.

Nach dem Klicken auf „INIT“ sollte die grüne LED des Adapters dauernd leuchten und damit anzeigen, dass die Initialisierung akzeptiert wurde. Im Feld „Result“ wird in diesem Fall der Text „Onnn“ angezeigt, wobei „nnn“ die Versionsnummer der Adapter-Software angibt. Damit wird der Ruhezustand des Adapters beendet.

Nach der Initialisierung können alle anderen Kommandos ausgelöst werden, es ist jedoch ratsam, zunächst ein „PING“-Kommando zu senden.

---

## PING

Dieses Kommando dient zur Abfrage, ob der Adapter bereit ist.

## UN-PING

Ab Version 3.2 des Adapters ist dieses Kommando ohne Bedeutung und wird nicht mehr unterstützt. Im Testprogramm ist es weiter vorhanden, um es zusammen mit früheren Versionen des Adapters aufrufen zu können.

## BREAK

Mit dieser Schaltfläche wird das Senden eines BREAK-Kommandos an den Adapter ausgelöst. Er muss danach mit einem neuen INIT-Kommando (und ggf. mit einem PING-Kommando) erneut aktiviert werden.

## Cfg. I/O

Mit dieser Schaltfläche wird ein CONFIGURE I/O-Kommando an den Adapter gesendet. Das Kommando erwartet zwei Parameter-Bytes, mit denen die Konfiguration der Port C und Port B Ein-/Ausgangsleitungen angegeben wird. Geben Sie diese vorher in das „Bytes“-Feld als zwei Hexadezimalzahlen mit mindestens einem Leerzeichen zwischen den Zahlen ein. So bewirkt z.B. „0 0“, dass alle Leitungen als Ausgänge konfiguriert werden, während „1F FF“ alle Leitungen als Eingänge definiert.

## Senden und Empfangen von I<sup>2</sup>C-Daten

Bevor I<sup>2</sup>C-Daten gesendet oder empfangen werden können, muss im Feld „I<sup>2</sup>C-Address“ die Adresse des anzusprechenden I<sup>2</sup>C-Bausteins eingegeben werden. Die Eingabe muss in Hexadezimal-Notierung als 7-Bit Wert erfolgen. Das Programm schiebt die Adressbits automatisch um eine Position nach links und fügt das Schreib-/Lesebit je nach Kommando ein.

Wenn die Kommunikation mit dem I<sup>2</sup>C-Baustein erfolgreich ist, gibt der Adapter als erstes Zeichen ein „O“ zurück und je nach Kommando weitere Bytes. Bei fehlerhafter Kommunikation gibt der Adapter als erstes Zeichen ein „E“ zurück und je nach Kommando ein oder mehrere „0“-Zeichen.

Ein Fehler tritt auf, wenn der Adapter bei einer I<sup>2</sup>C-Kommunikation vom adressierten Baustein kein Acknowledge empfängt. Der Grund dafür kann z.B. sein, dass eine falsche Adresse angegeben wurde, der I<sup>2</sup>C-Bus unterbrochen ist oder dass der angesprochene Baustein defekt ist.

### Checkbox „Send Start Byte“

Wenn diese Box markiert ist, werden beim Klicken der Schaltflächen RX1, RXN, TX1 und TXN die entsprechenden Adapter-Kommandos erzeugt, die ein einleitendes Start-Byte (0x01 ohne Acknowledge-Prüfung) senden

### RX1

Diese Schaltfläche sendet ein Lesekommando für ein Datenbyte an den I<sup>2</sup>C-Baustein mit der angegebenen Adresse. Wenn der Baustein antwortet, erscheint danach im Feld „Result“ die Zeichenfolge „O <xx>“, wobei <xx> den gelesenen Wert in hex angibt.

### RXN

Diese Schaltfläche sendet ein Lesekommando für mehrere Datenbytes an den I<sup>2</sup>C-Baustein mit der angegebenen Adresse. Die Anzahl der zu empfangenen Datenbytes (ohne das führende „O“) muss

---

zuvor im Feld „Byte count“ eingegeben werden. Wenn der Baustein antwortet, erscheint danach im Feld „Result“ die Zeichenfolge „O <xx> <xx>...“, wobei <xx> die gelesenen Werte in hex angeben.

### **TX1**

Diese Schaltfläche sendet das Schreibkommando für ein Datenbyte an den I<sup>2</sup>C-Baustein mit der angegebenen Adresse. Wenn der Baustein mit Acknowledge antwortet, erscheint danach im Feld „Result“ die Zeichenfolge „O“. Im Feld „Bytes“ muss zuvor ein hexadezimalwert eingegeben werden (ohne besondere Kennzeichnung und nicht in spitzen Klammern, also z.B. 1d, nicht jedoch 0x1d, 1dh oder <1d>).

### **TXN**

Diese Schaltfläche sendet das Schreibkommando für mehrere Datenbytes an den I<sup>2</sup>C-Baustein mit der angegebenen Adresse. Die Anzahl der zu sendenden Datenbytes muss zuvor im Feld „Byte count“ eingegeben werden. Die zu sendenden Bytes müssen im Feld „Bytes“ als Hex-Werte, jeweils durch mindestens ein Leerzeichen getrennt eingegeben werden (ohne besondere Kennzeichnung und nicht in spitzen Klammern, also z.B. 1d, nicht jedoch 0x1d, 1dh oder <1d>). Wenn der Baustein mit Acknowledge antwortet, erscheint danach im Feld „Result“ die Zeichenfolge „O“.

### **Send Bad Command**

Diese Schaltfläche löst das Senden eines falschen Kommandos an den Adapter, um zu demonstrieren, wie er auf solche Kommandos antwortet (mit der Rückgabe „?“).

### **Parallele Ein- und Ausgabe**

Der Adapter verfügt über acht parallele Eingänge und fünf parallele Ausgänge.

### **IN**

Mit dieser Schaltfläche wird der aktuelle Zustand der Eingänge gelesen und in der Form „O cc bb“ zurückgegeben, wobei „cc“ ein Hexadezimalwert ist, der den Zustand der Port C Leitungen und „bb“ den der Port B Leitungen.

### **Read Pin**

Diese Schaltfläche sendet ein READ PIN Kommando an den Adapter. Das Kommando erwartet die Leitungsnummer als Parameter. Geben Sie diese vorher in das „Bytes“-Feld ein. Gültige Werte sind C...0 (Port C, Leitung4...Port B, Leitung 0).

### **OUT**

Diese Schaltfläche setzt die Ausgänge des Adapters auf die Zustände, die mit den zuvor im Feld „Bytes“ eingegebenen zwei Hexadezimalwerten angegeben sind.

### **Write Pin**

Diese Schaltfläche löst ein WRITE PIN Kommando aus. Es werden zwei Hexadezimalwerte im Feld „Bytes“ erwartet. Der erste Wert definiert die Leitungsnummer (C...0) und der zweite Parameter den gewünschten Ausgangszustand. Mit 0 wird der Ausgang auf low gesetzt, jeder andere Wert bewirkt dass der Ausgang auf high gesetzt wird.

---

## Zählerfunktionen

Der Adapter überwacht die acht parallelen Eingänge von Port B und bei jeder Signaländerung an einem Eingang (positive Flanke) wird ein entsprechender 16-Bit Zähler inkrementiert. Diese acht Zähler können einzeln oder gemeinsam gelöscht und ausgelesen werden.

Hierzu dienen die Schaltflächen „Clear Counter“, „Read Counter“, „Clear all Counters“ und „Read all Counters“. Bei „Clear Counter“ und „Read Counter“ wird der Zähler angesprochen, dessen Nummer im Feld „Counter-#“ angegeben ist.

Zu beachten ist, dass die Zählereingänge nicht entprellt sind. Hierzu muss ggf. durch externe Beschaltung gesorgt werden, um Fehlzählungen (z.B. durch prellende Schaltkontakte) zu vermeiden.

## „Low-Level“ I<sup>2</sup>C-Funktionen

Zur Kommunikation mit I<sup>2</sup>C-Bausteinen, die mit den „high-level“-Kommandos TX1/TXn und RX1/RXn nicht angesprochen werden können, dienen die „low-level“-Funktionen, die ebenfalls getestet werden können.

### Rd.Addr + Start

Diese Schaltfläche löst das Kommando SEND ADDRESS FOR READ WITH START aus, d.h. die Start-Bedingung wird gesetzt und die 7-Bit-Adresse im Feld „I<sup>2</sup>C-Address“ wird zusammen mit gesetztem R/\*W-Bit gesendet.

### Rd.Addr

Diese Schaltfläche löst das Kommando SEND ADDRESS FOR READ WITHOUT START aus, d.h. die 7-Bit-Adresse im Feld „I<sup>2</sup>C-Address“ wird zusammen mit gesetztem R/\*W-Bit gesendet, ohne dass zuvor die Start-Bedingung gesetzt wird.

### Wrt.Addr + Start

Diese Schaltfläche löst das Kommando SEND ADDRESS FOR WRITE WITH START aus, d.h. die Start-Bedingung wird gesetzt und die 7-Bit-Adresse im Feld „I<sup>2</sup>C-Address“ wird zusammen mit gelöschtem R/\*W-Bit gesendet.

### Rrt.Addr

Diese Schaltfläche löst das Kommando SEND ADDRESS FOR WRITE WITHOUT START aus, d.h. die 7-Bit-Adresse im Feld „I<sup>2</sup>C-Address“ wird zusammen mit gelöschtem R/\*W-Bit gesendet, ohne dass zuvor die Start-Bedingung gesetzt wird.

### Rd.Byte ACK

Diese Schaltfläche löst das Kommando READ BYTE WITH ACK aus, d.h. es wird ein Byte gelesen und anschließend das Acknowledge-Bit auf low gesetzt (Acknowledge). Bitte beachten Sie, dass zuvor mit einem der SEND READ ADDRESS-Kommandos ein I<sup>2</sup>C-Baustein ausgewählt werden muss, wenn dieses Kommando sinnvolle Rückgaben erzeugen soll. Wenn Sie als Rückgabe ständig „ff“ erhalten, wurde wahrscheinlich zuvor kein vorhandener Baustein adressiert.

### Rd.Byte NACK

Diese Schaltfläche löst das Kommando READ BYTE WITH ACK aus, d.h. es wird ein Byte gelesen und anschließend das Acknowledge-Bit auf high gesetzt (kein Acknowledge). Bitte beachten Sie, dass zuvor mit einem der SEND READ ADDRESS-Kommandos ein I<sup>2</sup>C-Baustein ausgewählt werden muss,

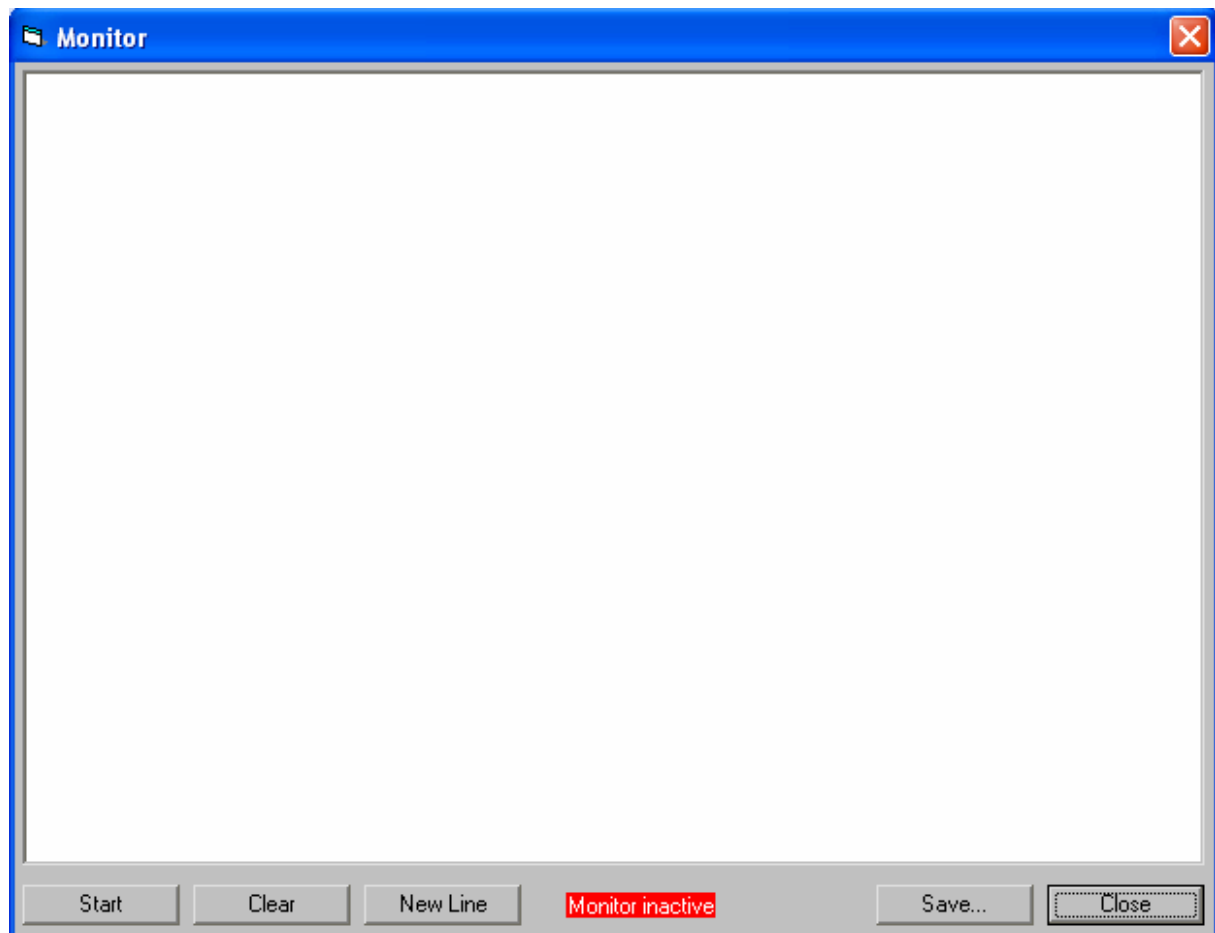
wenn dieses Kommando sinnvolle Rückgaben erzeugen soll. Wenn Sie als Rückgabe ständig „ff“ erhalten, wurde wahrscheinlich zuvor kein vorhandener Baustein adressiert.

### Send Byte

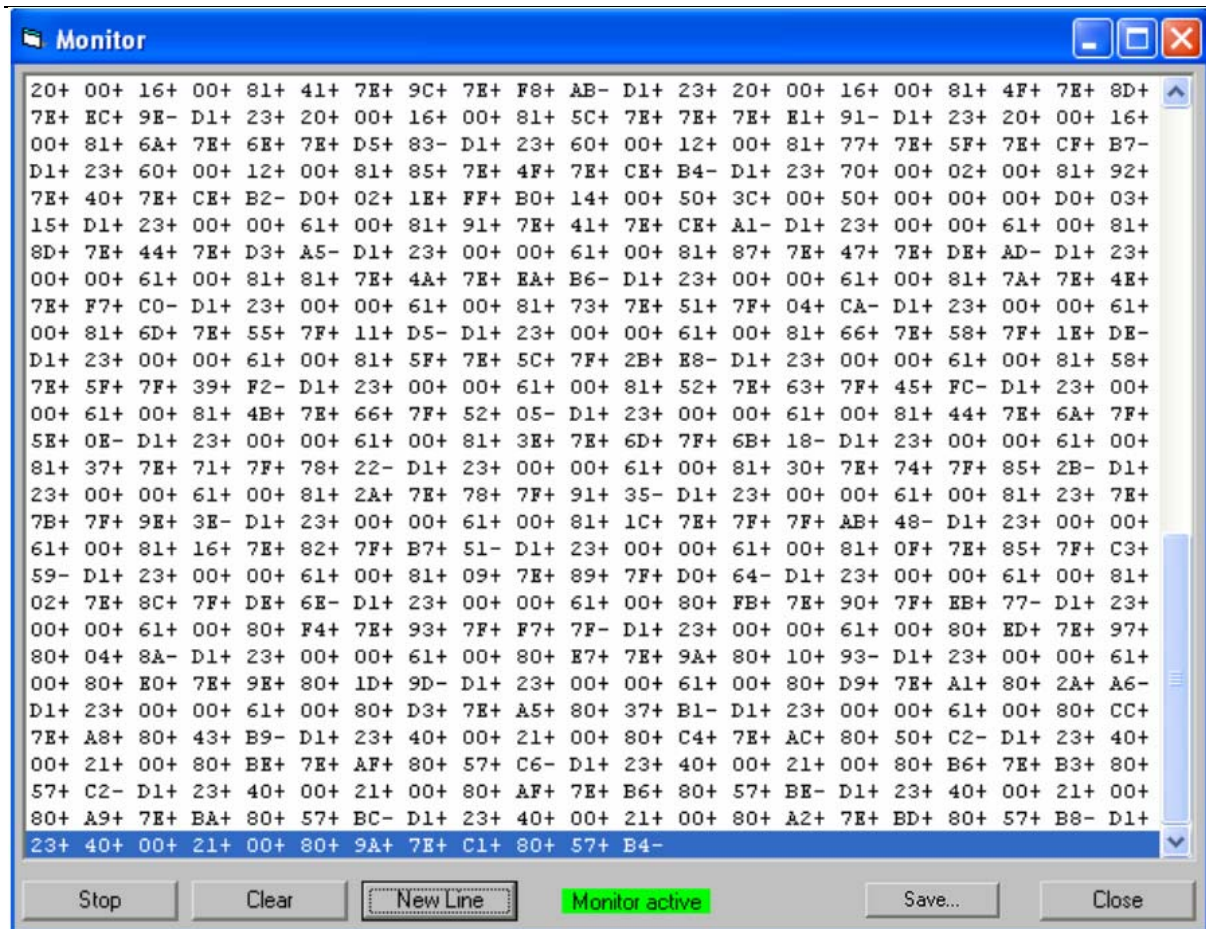
Diese Schaltfläche löst ein SEND BYTE-Kommando aus, d.h. das Byte ganz links im „Bytes“-Feld wird über den Bus gesendet. Bitte beachten Sie, dass vor der ersten Ausführung dieses Kommandos mit einem SEND WRITE ADDRESS-Kommando der I<sup>2</sup>C-Baustein ausgewählt werden muss, der das Byte empfangen soll. Wird die Schreiboperation von einem I<sup>2</sup>C-Baustein quittiert, erscheint „O“ im „Result“-Feld, sonst „E“.

### Monitor

Diese Schaltfläche öffnet das Monitor-Fenster:



Erstmaliges Klicken der Schaltfläche „Start“ sendet ein MONITOR-Kommando an den Adapter und aktiviert damit den Monitor-Modus. Die Beschriftung dieser Schaltfläche wird dann „Stop“ und die Meldung „Monitor inactive“ wechselt nach „Monitor active“. Die Baudrate der verwendeten COM-Schnittstelle wird automatisch auf 112500 Baud gesetzt.



Nun werden alle über den Bus gesendeten Bytes, die der Adapter erkennt im Listenbereich des Fensters z.B. in folgendem Format angezeigt:

```
xx+ xx+ xx- xx-
```

Hierbei steht „xx“ für den Hex-Wert der erkannten Bytes. Ein nachfolgendes „+“ kennzeichnet, dass das Byte mit Acknowledge quittiert wurde, während ein „-“ kennzeichnet, dass keine Quittung (Acknowledge) erfolgte.

Klicken der Schaltfläche „Stop“ belässt den Adapter im Monitor-Modus und er empfängt weiter Zeichen, diese werden jedoch im Listenbereich nicht angezeigt, solange bis wieder Start geklickt wird.

Die Schaltfläche „Clear“ ermöglicht das Löschen aller bisher im Listenbereich gesammelter Informationen.

Die Schaltfläche „New Line“ dient dazu, um im Listenbereich die weitere Ausgabe in einer neuen Zeile zu beginnen.

Mit der Schaltfläche „Save“ wird ein Dialog geöffnet, in dem ein Dateiname angegeben werden kann unter dem die bisher erfassten Daten als ASCII-Text gespeichert werden sollen.

Die Schaltfläche „Close“ schließt das Monitor Fenster und sendet ein BREAK-Kommando an den Adapter, um den Monitor-Modus zu beenden. Die Baudrate der verwendeten COM-Cnsittstelle wird automatisch wieder auf 38400 Baud gesetzt.

## Hinweis

*Dieses Dokument beschreibt die aktuelle Version des RS-232-PC-Adapters. Technische Änderungen auch ohne Vorankündigung vorbehalten.*