SC18IM700-Tester v1.0

1. Einleitung

Der SC18IM700-Tester ist ein mittels Visual Studio .NET und in der Programmiersprache C# entwickeltes Programm. Es lehnt sich an der Funktion eines einfachen Terminal-Programms an und erleichtert die Prüfung und Ansteuerung des SC18IM700 bzw. der angeschlossenen I2C Komponenten.

Voraussetzung für den Start der Programmdatei (.exe) ist - wie bei .NET Anwendungen üblich - die Installation des .NET Frameworks. In diesem Fall der Version 2.0.

Bei der Entwicklung stand im Vordergrund, die Ein- und Ausgabe für den Anwender verständlich und interpretierbar zu machen. Beides ist bei einfachen Terminalprogrammen i.d.R. nicht gewährleistet, da diese sich meist auf eine textbasierte Ein-/Ausgabe beschränken.

Anzumerken ist, dass Microsoft per Default .NET Klassen eine Baudrate bis 115200 Baud unterstützt. Sollen höhere Baudraten ve3rwendung finden, so sind externe Klassen zu verwenden, wie es hier nicht der Fall ist.

2. Kommunikationsmechanismen

Herkömmliche Terminalprogramme arbeiten in der Regel asynchron. Dies bedeutet, dass das Programm in zwei Teilprogrammen (Threads) arbeitet (senden und empfangen), die nicht synchron ablaufen. Wird über das Frontend eine Eingabe eine Zeichenfolge vorgenommen, so wird diese im ersten Thread an die Schnittstelle bzw. das Modul gesendet. Ein zweiter und unabhängiger Thread „lauscht" an der Schnittstelle und verarbeitet alle empfangenen Daten.

Für einen einfachen Test eines Moduls ist diese Form der Kommunikation durchaus ausreichend, da die Eingabe meist manuell erfolgt und zwischen zwei Eingaben genügend Zeit vergeht und ein Response des Moduls „scheinbar" unmittelbar erfolgt oder auch nicht.

Entwickelt man allerdings ein Programm für den praktischen Einsatz, so ist es in der Regel erforderlich, dass man einem gesendeten Befehl eine Response zuordnen möchte. Um dies zu gewährleisten, gibt es nun zwei Möglichkeiten. Zum Einen kann auf die asynchrone Kommunikation zurückgegriffen werden. Eine Zuordnung ist hier nur mit einem hohen Aufwand möglich, da hier einige Sonderfälle bedacht werden müssen. Stellen Sie sich vor, Sie fragen über das Modul unmittelbar nacheinander drei PCF8574 (I/O Port) Bausteine ab und nur eines der Module antwortet nicht. In diesem Fall haben Sie zwei Ergebnisse zu drei Anfragen. Eine Zuordnung ist nicht möglich.

Einfacher gestaltet sich eine synchrone Kommunikation. In diesem Fall wartet das sendende Programm auf einem Response. Lediglich der Fall eines Timeouts ist zu berücksichtigen. Eine Entwicklung gestaltet sich also zuverlässiger und einfacher.

Die Dauer des erforderlichen Timeouts ist einerseits von der Bus-Taktung und andererseits von der Geschwindigkeit des ausgeführten Programm-Codes abhängig. Für den SC18IM700-Tester beträgt der Default 50ms.

Der SC18IM700-Tester bietet beide Formen der Kommunikation, asynchron und synchron. Für die synchrone Kommunikation muss ein Timeout definiert werden, d.h. die Zeit, die das Programm auf einen Response warten soll.

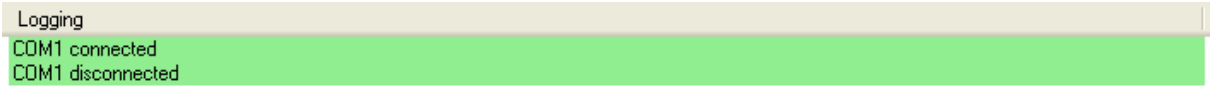
3. Verbindung herstellen

Um eine Verbindung zu Ihrem SC18IM700-Modul herzustellen, stehen Ihnen zwei Möglichkeiten zur Verfügung.

1. Manuelle Verbindung
2. Automatische Verbindung

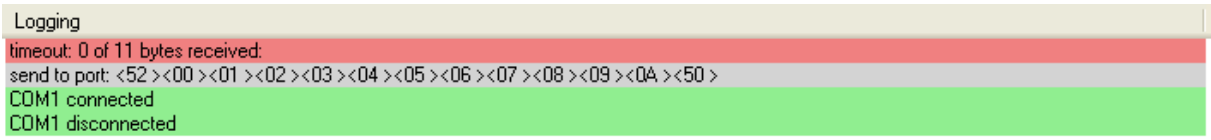
Manuelle Verbindung

Die manuelle Verbindung funktioniert analog zu herkömmlichen Terminal-Programmen und bedingt die Auswahl der Schnittstelle und der Baudrate. Mittels „connect"-Button wird die serielle Schnittstelle geöffnet.

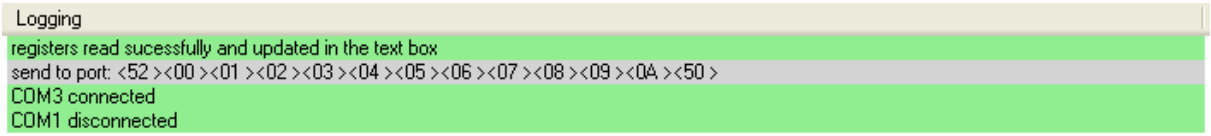


Bei diesem Vorgehen ist zu bedenken, das lediglich die Schnittstelle geöffnet und eine Baudrate definiert wird. Bis zu diesem Punkt ist noch keine Kommunikation erfolgt. Es wird nicht automatisch erkannt ob ein Modul tatsächlich an dieser Schnittstelle angeschlossen ist. Dies bedingt erst das senden eines Befehls, der eine eindeutige Response liefert.

Um zu kontrollieren, ob an der ausgewählten Schnittstelle tatsächlich ein SC18IM700-Modul angeschlossen ist, ist in diesem Fall einfach zu testen, indem z.B. mittels „read registers"-Button die Register des SC18IM700 ausgelesen werden. Ist kein Modul angeschlossen, so erfolgt ein Timeout.



Andernfalls wird eine korrekte Byte-Folge empfangen, eine Meldung im Log und die Registerwerte im Frontend dargestellt.



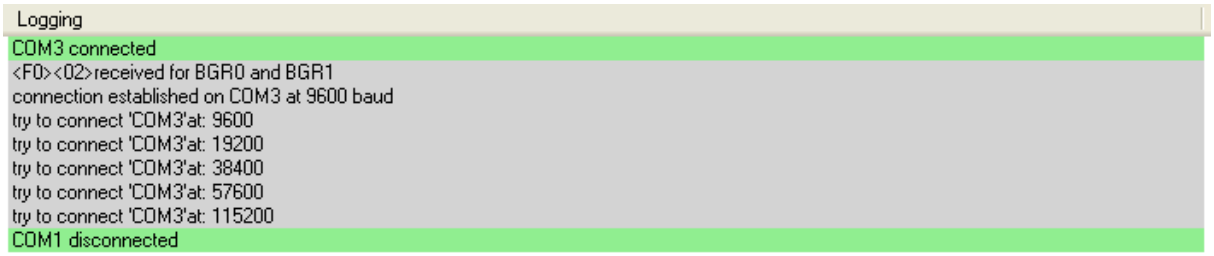
Bei dem SC18IM700 ist zu bedenken, dass nach einem Anschluss das Modul zunächst auf 9600 Baud eigestellt ist. Wird die Baudrate umgestellt und das Modul getrennt und erneut angeschlossen, so ist keine Kommunikation mehr möglich, da die Schnittstelle weiterhin versucht mit der eingestellten Baudrate zu kommunizieren, wohingegen das Modul wieder auf den Default von 9600 eingestellt ist.

In diesem Fall muss die Schnittstelle programmatisch zunächst wieder geschlossen und erneut mit 9600 Baud geöffnet werden.

Automatische Verbindung

Der SC18IM700-Tester unterstützt eine automatische Verbindung zum Modul. Mit dem Button „auto connect" werden alle COM-Schnittstellen nacheinander mit allen unterstützten Baudraten geöffnet und es werden mittels „synchroner" Kommunikation die Register des SC18IM700 ausgelesen, die die eingestellte Baudrate speichern. Antwortet das Modul auf einer Schnittstelle, d.h. kann die tatsächliche Baudrate ausgelesen

werden, wird der Port abschließen geschlossen und mit der konfigurierten Baudrate erneut geöffnet. Im Log kann der Vorgang und das Ergebnis beobachtet werden.



Ändern der Baudrate

Zu beachten ist, dass zur Änderung der in den SC18IM700-Registern konfigurierten Baudrate zunächst eine korrekte Verbindung herzustellen ist, ansonsten können die Register nicht geschrieben werden.

Besteht eine Verbindung zum Modul und wird die Baudrate in der Programmoberfläche geändert, so werden automatisch die Registerwerte übertragen und die Com-Schnittstelle mit der eingestellten Baudrate neu geöffnet.

Wird der Wert geändert ohne dass eine Verbindung existiert, so ist der eingestellte Wert die Basis für ein Öffnen der Schnittstelle mittels „connect"-Button.

4. Bytes senden und empfangen

Ist eine Verbindung zum Modul aufgebaut, so können beliebige Zeichenketten bzw. Byte-Ketten an das Modul übertragen und von diesem empfangen werden.

Für die Kommunikation ist zunächst festzulegen, ob synchron oder asynchron kommuniziert werden soll. Im synchronen Fall ist neben dem Timeout zu jedem Befehl auch die erwartete Anzahl Bytes der Response anzugeben (s.u.).

Syntax

Die Eingabe der an das Modul zu übertragenden Bytes kann in folgenden Formaten erfolgen:

- ASCII
- Hexadezimal
- Dezimal
- Binär

Soll z.B. nach einem Verbindungsaufbau das GPIO-Register ausgelesen werden, erfolgt dies mit der ASCII-Zeichenkette „IP". Gleichbedeutend ist die Angabe wie folgt:

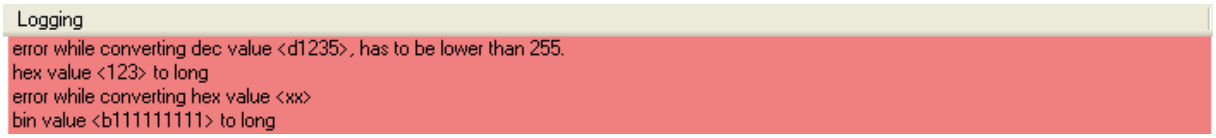
- hexadezimal: IP = <49><50>
- dezimal: IP = <d73><d80>
- binär: IP = <b1001001><b1010000> = <b**0**1001001><b**0**1010000>

Ebenso können alle Formen miteinander kombiniert werde, wie die folgenden Beispiele zeigen:

- Kombination: IP = I<40>
- Kombination: S<41><01>P = S<d65><1>P = SA<1>P

Hinweis: Vor der Konvertierung des eingegeben Textes werden zunächst alle Leerzeichen aus der Zeichenfolge entfernt. Hieraus folgt, dass ein Leerzeichen nicht interpretiert wird. Soll also das zugehörige Byte 32 übertragen werden, so ist diese explizit anzugeben, z.B. durch „<d32>" oder „<20>". Ebenso folgt daraus, dass binäre Angaben durchaus Leerstellen beinhalten dürfen, um z.B. die Übersicht zu verbessern: <b0100 1000>

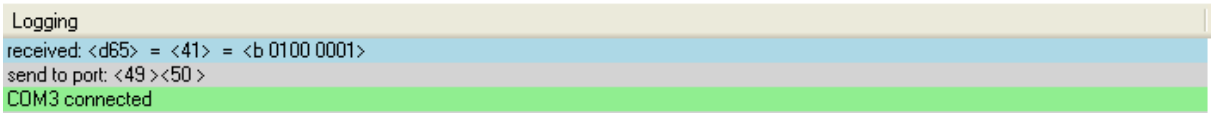
Sofern bei der Konvertierung Fehler auftreten, werden diese im Log dokumentiert und es werden „keine" Daten übertragen!



Asynchrone Kommunikation

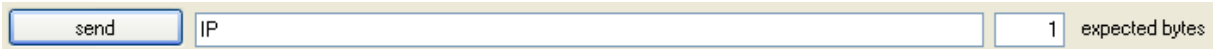
Für eine asynchrone Kommunikation werden die angegebenen Bytes mit dem Drücken des „send"-Buttons an die Schnittstelle übertragen und im Log hexadezimal angezeigt.

Sofern ein Response erfolgt, wird dies im Log ebenfalls hexadezimal dargestellt.

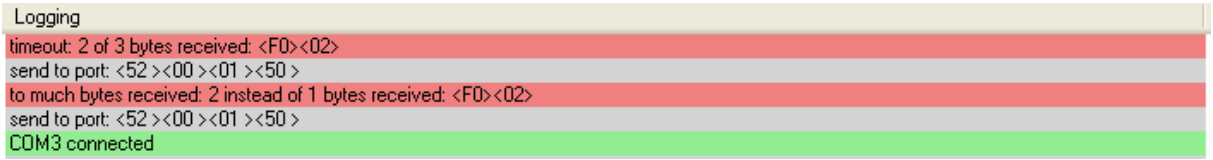


Synchrone Kommunikation

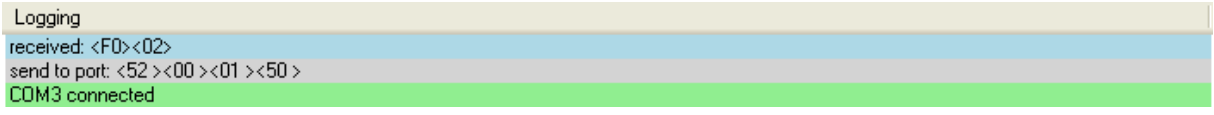
Im Falle einer Synchronen Übertragung ist die Anzahl der zu erwartenden Response-Bytes anzugeben.



Wir diese nicht korrekt angegeben, bzw. eine abweichende Anzahl Bytes empfangen, erfolgt eine entsprechende Meldung im Log.



Andernfalls wird der korrekte Empfang bestätigt und die Daten dargestellt:



Hinweis: Das Log kann jederzeit gelöscht werden. Hierzu steht der Befehl „clear log" im Context-Menü des Logs zur Verfügung.

5. Vorbereitete Funktionen

Im SC18IM700-Tester werden folgende Funktionen per Button direkt bereitgestellt:

Read GPIO

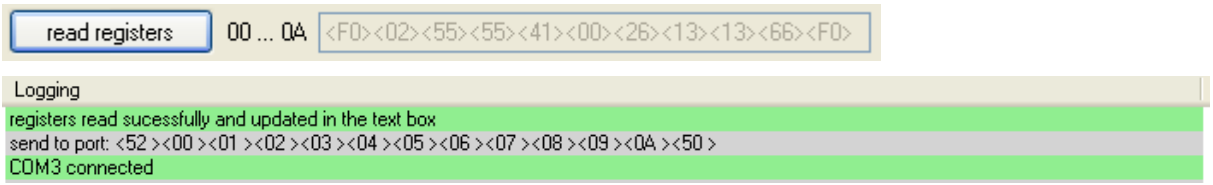
Es wird die Zeichenkette „IP" an das Modul übertragen. Dies ermöglicht einen schnellen Test, ob das Modul an verbundenen Com-Port tatsächlich angeschlossen ist und reagiert.

Antwortet das Modul korrekt, so wird ein Byte empfangen, bei dem jedes Bit einem Port-Status entspricht. Diese Status werden entsprechend der binären Auflösung des Bytes anhand von Symbolen dargestellt.

Read Registers

Mit dem Command „S<00><01><02><03><04><05><06><07><08><09><0A>P" wird das Modul aufgefordert den Inhalt der Register „<00>" bis „<0A>" zurück zu senden.

Die empfangenen Werte werden im log bestätigt und in der Programmoberfläche dargestellt.



Hinweis: Das letzte Byte, d.h. der Inhalt des Registers „0A" stellt den Status der letzten I2C-Kommunikation dar. Durch ein einfaches Abfragen der Register nach einer Kommunikation kann daher schnell auf den Status geschlossen werden:

I2C_OK	<b1111 0000> = <F0>
I2C_NACK_ON_ADDRESS	<b1111 0001> = <F1>
I2C_NACK_ON_DATA	<b1111 0010> = <F2>
I2C_TIME_OUT	<b1111 0011> = <F3>

Read PCF8574

Der PCF8574 ist ein einfacher I2C-Baustein, der 8 quasibidirektionale Ports bereitstellt. Durch das Command „S<*Moduladresse* + *Read-Bit*><01>P" sendet der mit der Moduladresse adressierte PCF8574 den Portstatus als Byte zurück. Durch Auswahl der Adresse können auf einfachem Wege die Status der Ports an den Adressen <40> … <47> eingelesen und analog zu den GPIO-Ports anhand von Symbolen dargestellt werden.