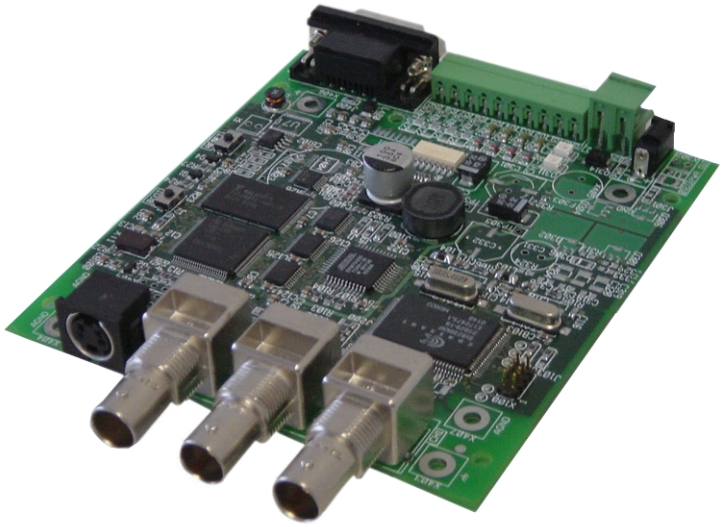
grabbMODUL-4



Softwareerweiterungen

Ausgabe: August 2004

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Allgemeines zur grabbMODUL-4 Software

In diesem Handbuch finden Sie die Beschreibung zu weiterführender Software für das Produkt grabbMODUL-4 der Firma Phytec Meßtechnik GmbH. Folgende Zusatzsoftware wird in diesem Handbuch beschrieben:

- **MODEM_COM (SO-670-X1):** Firmware zur Bildfernübertragung
  Die Firmware MODEM_COM ist eine fertige Modulsoftware, mit der Bild- und Steuerdaten über große Entfernungen - zum Beispiel über eine Telefonverbindung - einfach übertragen werden können.

- **CopyFunc.LIB:** Assember-Routinen für das grabbMODUL-4 zum schnellen Kopieren von Bilddaten aus dem Videospeicher in den RAM-Speicher des Moduls.

2. MODEM_COM: Firmware zur Bildfernübertragung

2.1 Funktionsweise

Mit der Firmware MODEM_COM stellt PHYTEC Ihnen eine fertige Modulsoftware zur Verfügung, mit der Bild- und Steuerdaten über große Entfernungen einfach übertragen werden können.
Dazu wird das grabbMODUL-4 über ein entsprechendes Modem mit dem Telefonnetz verbunden. Die Übertragung der Daten (Steuerbefehle und Bilder) erfolgt dann über das Telefonnetz.

Mit dieser Firmware kann das grabbMODUL-4 in Kombination mit einem Modem selbständig einen Anruf annehmen und über ein definiertes Protokoll mit einer Gegenstelle kommunizieren.
Als Hardwareerweiterung zu der bekannten grabbMODUL-4/Kamera-Einheit benötigen Sie auf der Modulseite ein Modem (Telefonanschluß) und eine angepaßte Gegenstelle (Leitstand oder PC inkl. Modem).

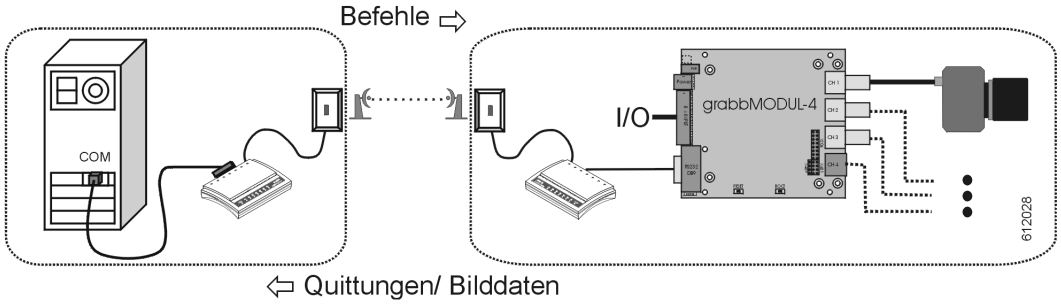


Bild 1: Gerätekonfiguration für MODEM_COM - Firmware

Die Software auf dem Modul basiert auf der bewährten und ausführlich beschriebenen Firmware „LOCAL_COM“ (siehe Handbuch grabbMODUL-4, L-612). Die „LOCAL_COM“ Software benötigt eine serielle Verbindung zwischen dem grabbMODUL-4 und einem Leitstand-/Host-PC. Auf dieser seriellen Verbindung werden Befehle und Quittungen übertragen, mit denen das grabbModul-4 eingestellt und gesteuert werden kann. Die zu einem bestimmten Ereignis oder auf Anforderung aufgenommenen Bilder, können von der Modulseite in verschiedenen Formaten übertragen werden.

Es ist dabei unwichtig ob diese serielle Verbindung direkt durch ein Kabel (siehe Bild 2)

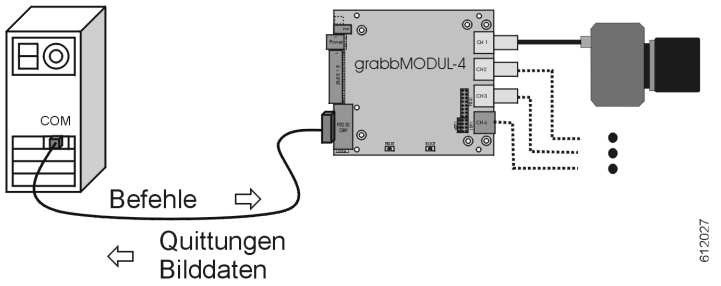


Bild 2: Direkte serielle Verbindung

oder „imaginär“ durch eine zwischengeschaltete Modem-Modem Verbindung (siehe Bild 3) bereitgestellt wird.

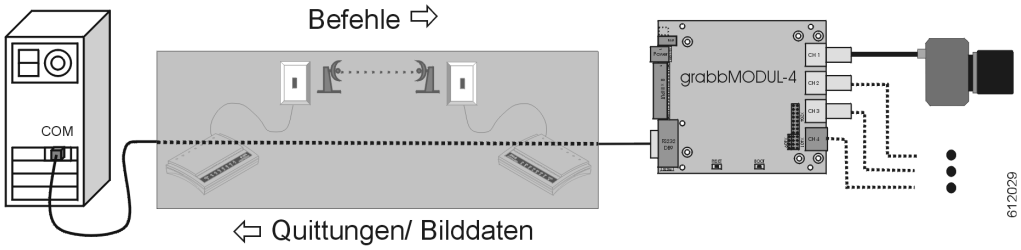


Bild 3: Imaginäre serielle Verbindung

Die Firmware „MODEM_COM“ erweitert nun das grabbMODUL-4 um die Eigenschaft, über die serielle Schnittstelle mit einem Modem zu kommunizieren und das Modem in diese „imaginäre serielle Verbindung“ zu versetzen.

Die Kommunikation grabbMODUL-4 und Modem erfolgt über AT-Befehle. Diese AT-Befehle sind standardisiert und ermöglichen den Einsatz eines beliebigen Modems mit dem Modul. Folgender Ablauf wird bei der grabbMODUL-4 – Modem Kommunikation realisiert:

- Modem initialisieren (Kommando Modus)
- warten auf Anruf und Ruf annehmen (Kommando Modus)
- nach der Rufannahme in den „transparenten“ seriellen Modus schalten (Data Modus)
- Kommunikation entsprechend des Protokolls „LOCAL_COM“
- bei einem definierten Befehl den Modus wechseln (Data-Modus in Kommando-Modus) und die Verbindung trennen
- auf weitere Anrufe warten.

Bitte beachten Sie:

- Auf dem grabbMODUL-4 muß die Firmware MODEM_COM installiert sein. Dazu müssen Sie die Datei *modem_com_Vxx.h86* mit Hilfe der PHYTEC-Flashtools in den FLASH-Speicher des Moduls programmieren. (*xx* gibt die Versionsnummer an). Im Auslieferungszustand befindet sich die Software „LOCAL_COM“ im FLASH-Speicher des grabbMODUL-4.

- Das grabbMODUL-4 und das Modem müssen in der Anwendungsumgebung über ein voll belegtes serielles Verbindungskabel verbunden werden. Benutzen Sie nach Möglichkeit ein fertig konfektioniertes Kabel, welches üblicherweise beim Modem liegt. Weiterhin muß das Modem mit einer Telefondose verbunden und eingeschaltet sein.

- Die Kommunikation des grabbMODUL-4 mit einem Host/PC über Modem ist nur möglich, wenn auch der PC mit einem Modem verbunden ist. Weiterhin müssen Sie ein PC-Programm erstellen, welches das Modem am PC konfigurieren, einen Anruf tätigen und in den Data-Modus (imaginäre serielle Verbindung) wechseln kann. Danach kann die zu erstellende PC-Software über Befehle, beschrieben in der „LOCAL_COM“, mit dem grabbMODUL-4 kommunizieren.

- Die Software LOCAL_COM kann nach dem Download in das grabbMODUL-4 mittels eines Terminal-Programms (z.B. HyperTerminal, siehe Abschnitt 2.2) oder mit Ihrer eigenen zu erstellenden Windows-Software getestet werden. Befindet sich die Firmware „MODEM_COM“ im FLASH-Speicher des grabbMODUL-4 wird beim Neustart, nach einem Modem an der seriellen Schnittstelle gesucht. Deshalb ist mit dieser Software eine direkte Kommunikation eines PCs und dem grabbMODUL-4 nicht möglich.

2.1.1 Hinweise zum Funktionsablauf der Firmware

Der Funktionsablauf ist als Flußdiagramm in Abschnitt 2.1.2 dargestellt und wird im folgenden beschrieben:

- Die Software auf dem grabbMODUL-4 initialisiert die serielle Schnittstelle mit 57600 Baud, 8 Datenbits, keine Parität, ein Stoppbit und Hardware-Handshake (CTS/RTS).

Hinweis:
Die Geschwindigkeit 57600 Baud zwischen grabbMODUL-4 und Modem steht nicht direkt im Zusammenhang mit der Geschwindigkeit auf der Modemstrecke. Sollte die Modemstrecke mit einer anderen Geschwindigkeit aufgebaut werden (z.B. 9600 Baud) so ist eine Kommunikation immer noch möglich, da die Daten von den Modems zwischengespeichert werden.

- Danach wartet die Software 2s (Power On Modem) und versucht dann mit dem Modem eine erste Kommunikation aufzunehmen.
- ① Senden von „+++“ (Beendet einen eventuellen Daten-Modus)
- Senden von „ATH0“ (Beendet eine eventuell bestehende Verbindung)
- Senden von „AT“ (Synchronisieren mit dem Modem)
- Warte auf Antwort „OK“
- Wird kein „OK“ empfangen so wird ein Software-Reset auf dem Modul ausgeführt und das grabbMODUL-4 versucht immer wieder eine Kommunikation mit dem Modem aufzubauen, Sprung zu Punkt ①.
- ② Wird ein „OK“ empfangen wartet das Modul auf einen Anruf („RING“), der vom Modem erkannt und weitergeleitet wird.
- In dieser Zeit sind die Eingangskanäle aktiv. Das heißt bei einem beliebigen Flankenwechsel an den Eingängen I/O_1 - I/O_4 wird vom dazugehörigen Kanal ein Bild gegrabbt.

Hinweis:
Nachdem eine Verbindung aufgebaut wurde, ist die automatische Eingangsabfrage inaktiv und ein Bild kann nur noch über den Grabb-Befehl „G“ aufgenommen werden

- Wird ein „RING“ erkannt, dann sendet das grabbMODUL-4 die Zeichenfolge „ATA“ damit das Modem den Ruf annimmt.
- An dieser Stelle kann der Verbindungsaufbau der Modemstrecke zeitlich variieren. Deshalb wird bis zu 50s gewartet, ob eine Antwort vom Modem zurückkommt. Da die Anwort je nach Modem und Verbindungsqualität unterschiedlich sein kann wird nur der definierte String „CONNECT“ in der Anwort kontrolliert und der Rest verworfen.

- Wird kein „CONNECT“ erkannt, so wird zu Punkt ① gesprungen und mittels „AT“ Befehl kontrolliert ob das Modem noch aktiv ist. Ist das Modem noch aktiv wird ab Punkt ② weitergearbeitet, anderenfalls wird ein Software-Reset durchgeführt.
- Wenn ein „CONNECT“ erkannt wird, dann hat das Modem in den „transparenten“ seriellen Modus (Data Modus) gewechselt. Das heißt, mit der Gegenstelle kann kommuniziert werden, als ob sie direkt seriell mit dem grabbMODUL-4 verbunden wäre.
- Das grabbMODUL-4 wartet nun auf Steuerzeichen vom Host und reagiert entsprechend der „LOCAL_COM“-Spezifikation. Das verwendete Protokoll ist ausführlich im Handbuch grabbMODUL-4, L-612, Firmware „LOCAL_COM“ beschrieben. Für den Modembetrieb ist der Befehlssatz des „LOCAL_COM“ Protokolls erweitert worden. Diese Erweiterungen sind in Abschnitt 2.1.3 dieses Handbuchs beschrieben.
- Sollten in diesem Modus für 4 min keine Zeichen ausgetauscht werden, dann wird von einer Störung ausgegangen. In diesem Fall versucht, das grabbMODUL-4 in einen definierten Zustand zu bringen. Das heißt, es wird der Befehl zum Trennen der Verbindung gesendet und die Anworten vom Modem getestet. Antwortet das Modem, wird zu Punkt ② gesprungen, andernfalls wird ein Software-Reset auf dem grabbMODUL-4 durchgeführt.

Hinweis:

- Bei der AT-Befehlskommunikation vom Modul werden die Steuerkommandos 0x0D und 0x0A sowie die von der Modem-Echofunktion zurückgesendeten Zeichen mit abgefragt.
- Das grabbMODUL-4 verwendet zur Kommunikation mit einem Modem nur Standard AT-Befehle. Somit sollte der Einsatz eines beliebigen Modemtyps gewährleistet sein. Prüfen Sie jedoch bei der Auswahl eines Modems unbedingt, ob dieses mit der LOCAL_COM – Firmware auch in allen Betriebszuständen zusammenarbeitet.
- Es gibt die Möglichkeit, einen ankommenden Anruf für 2 min. zu unterdrücken (siehe Abschnitt 2.1.4). Das heißt, das Modul bleibt trotz eines „RING“ in der Eingangsabfrage-Routine ② und nimmt den eingehenden Ruf nicht an. Dies macht den Einsatz eines weiteren Gerätes am gleichen Telefonanschluß möglich.
- Während einer Onlineverbindung ist die, im grabbMODUL-4 Handbuch L-612 (Abschnitt „Bildaufnahme durch Alarmeingänge“) beschriebene, automatische Bildaufnahme deaktiviert.

2.1.2 Flußdiagramm MODEM_COM

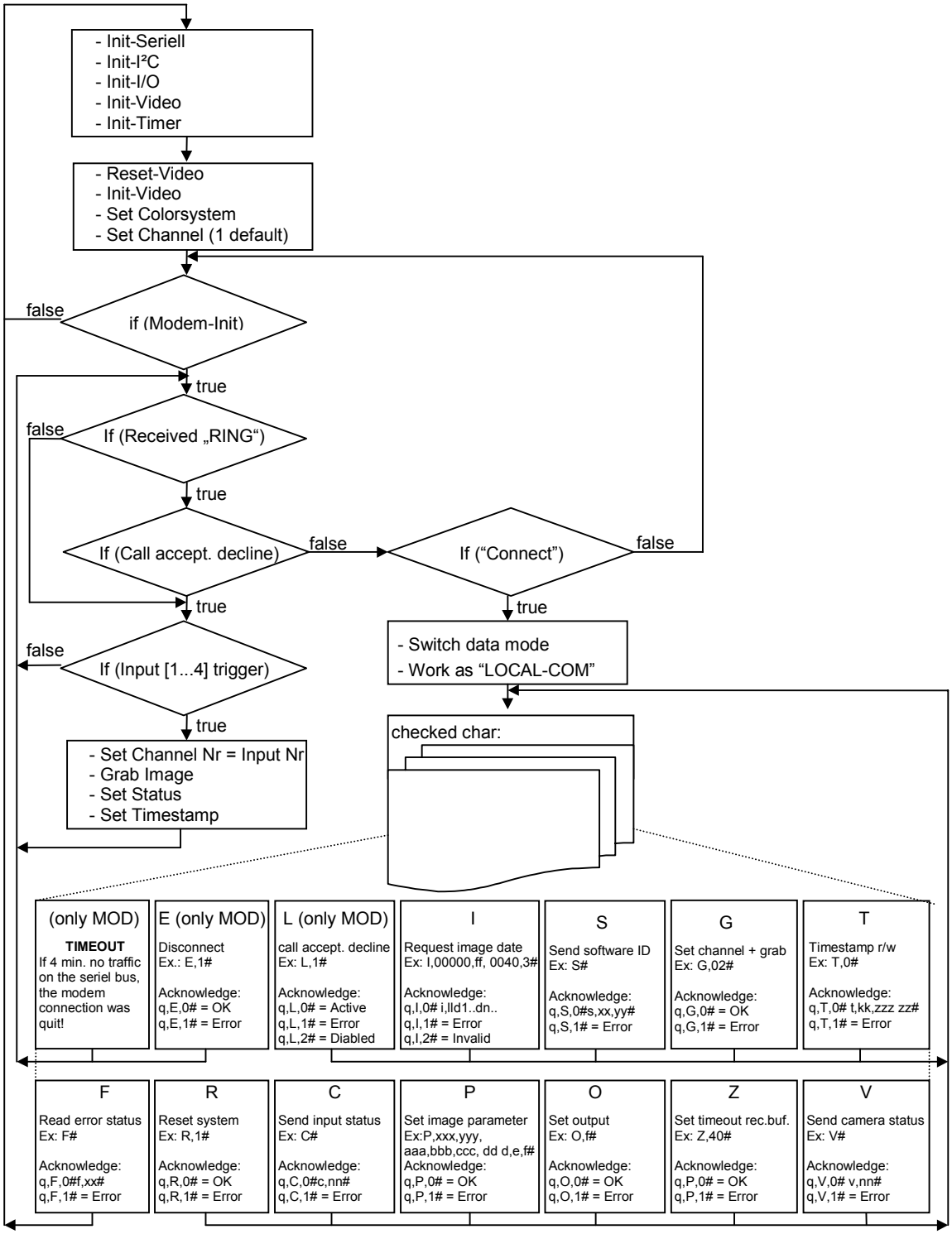


Bild 4: Flußdiagramm MODEM_COM

2.1.3 Übersicht über Befehle und Befehlsaufbau

Der Aufbau und die Funktion der Befehle und Quittungen ist im Handbuch grabbMODUL-4, L-612, Abschnitt „LOCAL_COM“ beschrieben. An dieser Stelle soll nur auf die Erweiterungen für den Modembetrieb eingegangen werden.

Folgende Befehle sind in der Firmware LOCAL_COM umgesetzt:
- **„C“** - Eingangs-Status anfordern
- **„F“** - Fehler-Status anfordern
- **„G“** - Kamerabild von Kanal k grabben
- **„I“** - Bilddaten anfordern
- **„J“** - Reserviert
- **„K“** - Reserviert
- **„O“** - Output-Status setzen
- **„P“** - Bildgröße und –skalierung einstellen
- **„R“** - Einheit zurücksetzen (**Erweitert für Modembetrieb!**)
- **„S“** - Software-ID abfragen
- **„T“** - Zeitstempel lesen / rücksetzen
- **„U“** - Reserviert
- **„V“** - Kamera-Status anfordern
- **„Z“** - Empfangs-Timeout der Schnittstelle setzen
- **0x0D** - Reserviert (Steuerzeichen Modem)
- **0x0A** - Reserviert (Steuerzeichen Modem)

Folgenden Befehle kommen in der Firmware MODEM_COM dazu:
- **„L“** - Rufannahme sperren
- **„E“** - Verbindung beenden

Bitte beachten Sie, daß das grabbMODUL-4 im Ruhezustand (keine Verbindung hergestellt) mit dem Modem über den AT-Befehlssatz kommuniziert. Erst nachdem eine Verbindung, eingeleitet durch ein „RING“, aufgebaut wurde, schaltet das Modem mit dem Modem der Gegenstelle auf „Transparenz“-Modus. Erst dann unterhält sich das Modul quasi via seriell mit dem System (z.B. PC) auf der Gegenstelle.
Sollten in diesem Modus für 4min keine Zeichen ausgetauscht werden dann wird von einer Störung ausgegangen. In diesem Fall versucht das grabbMODUL-4 in einen definierten Zustand zu gehen. Das heißt, es wird der Befehl zum Trennen der Verbindung gesendet und die Anworten vom Modem getestet.

2.1.4 Rufannahme sperren

Befehlscode: L,x#
Parameter: 0 = Rufsperre deaktiviert
1 = Rufsperre aktivieren
Beispiel: L,1#
Quittung: `q,L,0#` = Bestätigung, Rufsperre aktiviert
`q,L,1#` = Parameterfehler, keine Aktion
`q,L,2#` = Bestätigung, Rufsperre wurde deaktiviert

Bemerkung: Die Rufsperre ist erst wirksam wenn die aktuelle Verbindung mittels „E“-Befehl beendet worden ist. Sie ist dann für 2 min. aktiv, unabhängig wie oft und wie viele „RING“ vom Modem gesendet werden.

2.1.5 Verbindung beenden

Befehlscode: E,x#
Parameter: 1 = Verbindung beenden
alle anderen: Befehlsabruch
Beispiel: E,1#
Quittung: `q,E,0#` = Bestätigung, Verbindung wird getrennt
`q,E,1#` = Parameterfehler, Verbindung bleibt bestehen

Bemerkung: Eine bestehende Verbindung sollte immer über den Befehl „E“ vom Host beendet werden. Wird von der Gegenstelle aufgelegt, so kann dies vom Modul nicht erkannt werden. In diesem Fall bemerkt das Modul die Unterbrechung erst nach Ablauf der voreingestellten 4 min und ist erst dann in der Lage einen neuen Anruf entgegen zu nehmen.

Nachdem das Modul die Bestätigung gesendet hat, erfolgt das eigentliche Auflegen nach ca. 1,1s (Guard-Time) von der Modulseite durch das Senden der Escape-Sequenz („+++“) und dem Auflegebefehl („ATH0“). Nachdem das Modul-Modem aufgelegt hat, erkennt dies das Host-Modem und legt ebenfalls auf. Je nach Modemtyp kann die Zeitspanne bis zum Auflegen unterschiedlich sein. Sollten Sie von der Hostseite sofort wieder wählen wollen, empfiehlt sich ein sofortiges auflegen durch einen Ein/Aus-Übergang der DTR-Leitung zu erzwingen.

2.1.6 Einheit zurücksetzen

Befehlscode: Siehe Handbuch grabbMODUL-4, L-612, Abschnitt „Einheit zurücksetzen“.

Bemerkung: Zusätzlich zu der im Handbuch beschriebenen Funktionalität wurde dieser Befehl um folgendes erweitert: Bevor der Software-Reset auf dem Modul ausgeführt wird, sendet das grabbMODUL-4 die Steuerbefehle zum kontrollierten Beenden einer bestehenden Verbindung.

2.2 Testen von MODEM_COM mit einem Terminal-Programm

Um das Programm „MODEM_COM“ zu testen benötigen Sie eine in Bild 5 gezeigte Gerätekonfiguration.

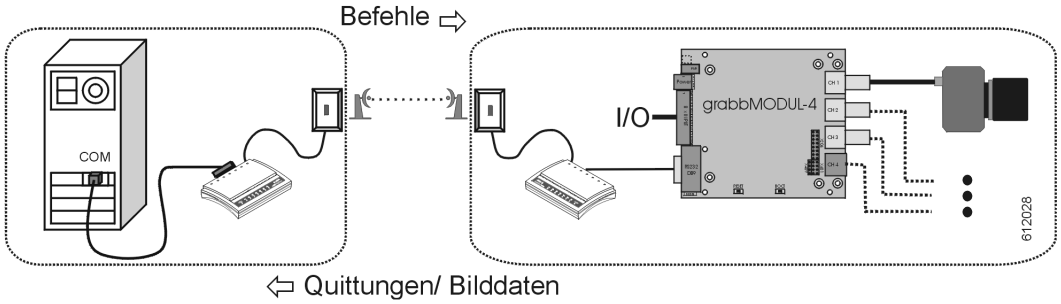


Bild 5: Gerätekonfiguration für die MODEM_COM - Firmware

Der Grundbestandteil dieser Konfiguration ist das grabbMODUL-4 inkl. Kamera und den dazugehörigen Anschlußkabel. Die allgemeine Modulinbetriebnahme ist im Handbuch grabbMODUL-4, L-612, Abschnitt „Rapid-Development-Kit Inbetriebnahme“ ausführlich erläutert.

Ergänzend zu dieser Grundkonfiguration werden folgende Komponenten für eine „MODEM_COM“ Verbindung benötigt:

- Modem, welches direkt an die serielle Schnittstelle des grabbMODUL-4 und an einen freien Telefonanschluß angesteckt wird.
- Ein weiteres Modem, welches direkt an einen PC (seriell, USB oder integriert) und an einen anderen freien Telefonanschluß gesteckt wird (Gegenstelle).

Bevor die Gerätekonfiguration getestet werden kann, muß sichergestellt sein, daß sich die Firmware „MODEM_COM“ im Flash-Speicher des grabbMODUL-4 befindet. Führen Sie hierzu bitte die in Abschnitt „Installation der PHYTEC FlashTools und Download eines Programmcodes“ des Handbuch L-612 durch und laden Sie das File *„Modem_Com_Vxx.h86“* **(SO-670-X1)** in das Modul.

Hinweis:
Sollten Sie das Modem im Betrieb aus/einschalten, müssen Sie zur Neuinitialisierung der Modem-Modul Verbindung auch einen Reset des grabbMODUL-4 durchführen.

Zum Test der Funktionen und des Protokolls kann auf der PC-Seite ein Terminal-Programm verwendet werden. So kann zum Beispiel das bei Windows mitgelieferte Programm „HyperTerminal“ verwendet werden. Es können die Steuerzeichen an das grabbMODUL-4 gesendet und die Anworten bzw. Bildrohdaten (im ASCII-Format) angezeigt werden. Die im folgenden beschriebene Vorgehensweise kann sich bei verschiedenen Betriebssystemen leicht unterscheiden.

Stellen Sie als erstes sicher, daß ein Modem an den PC angeschlossen ist. Des weiteren muß Ihnen der COM-Port, den das MODEM belegt, bekannt sein. Bei internen Modems wird ihnen dies im Gerätemanager angezeigt.

- Starten Sie das Programm *HyperTerminal* im Verzeichnis *Start\Programme\Zubehör* bzw. *Start\Programme\Zubehör\Kommunikation*. (Das Verzeichnis kann je nach Betriebssystem leicht variieren).
- Daraufhin öffnet sich folgendes *HyperTerminal* Fenster:

Bild 6: HyperTerminal: Programm erstellen

- Durch einen Doppelklick auf das *HyperTerminal*“- Symbol wird eine neue HyperTerminal Verbindung gestartet.

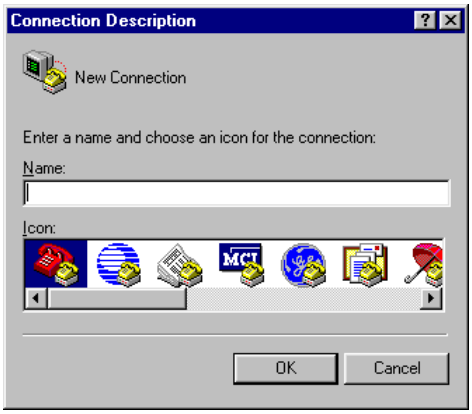


Bild 7: HyperTerminal: Programm benennen

- Es erscheint das Fenster „*Beschreibung der Verbindung*“. Geben Sie „MODEM Verbindung“ im Namensfeld ein.
- Nach dem Bestätigen durch Klicken auf „*OK*“ wird eine neue HyperTerminal Verbindung mit dem Namen „*MODEM Verbindung*“ hergestellt.

- Spezifizieren sie im „*Eigenschaften von MODEM Verbindung“*-Fenster die Eigenschaften der COM-Verbindung.

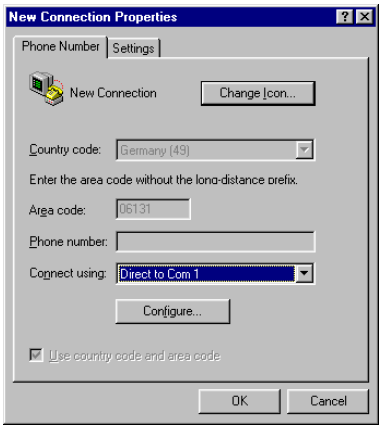
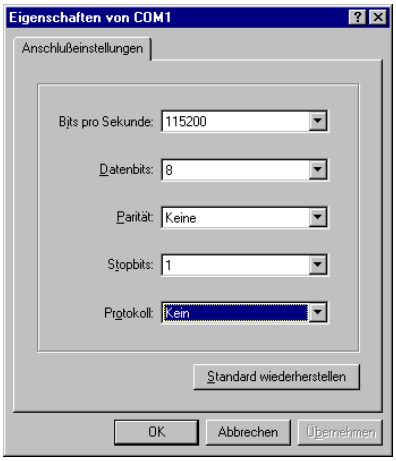


Bild 8: HyperTerminal: Programm Eigenschaften

- Wählen Sie im Auswahlfeld „*Verbindung über:*“ **direkt** den COM-Port, an dem Ihr Modem angeschlossen ist (z.B. COM 1) und bestätigen dies.
- Durch Betätigung der Schaltfläche „Konfigurieren / Anschlußeinstellungen“ öffnet sich das Fenster „*Eigenschaften von COM1“*. Stellen Sie in diesem Fenster folgende COM Parameter ein:



- Bits pro Sekunde = *115200*
- Datenbits = *8*
- Parität = *Keine*
- Stop Bits = *1*
- Protokoll = *Kein*

Bild 9: HyperTerminal: Programm Konfigurieren

- Klicken Sie auf „*OK“* und wechseln Sie zum Monitor-Fenster *“MODEM Verbindung – HyperTerminal”*.

Schließen und speichern Sie HyperTerminal und starten Sie das nun neu erstellte Verbindungssymbol „Modem Verbindung“, damit die durchgeführten Einstellungen übernommen werden!
Je nach Betriebssystem-Variante befindet sich es u.U. in dem separaten Ordner „HyperTerminal“.

- Nach dem Start des „*MODEM Verbindung – HyperTerminal*“ sehen sie das Monitor-Fenster. Weiterhin werden in der unteren Leiste Statusinformationen über die Verbindung eingeblendet.

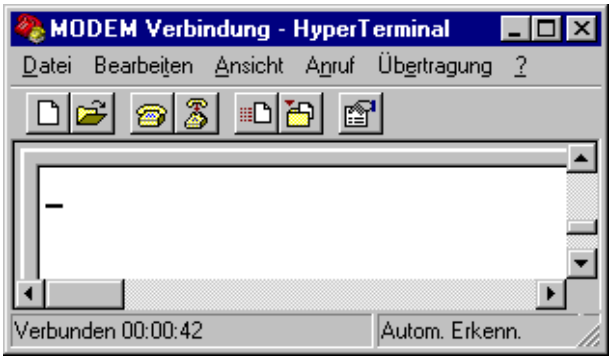


Bild 10: HyperTerminal: Programm starten

- Führen Sie einen Reset des grabbMODUL-4 Boards durch, indem Sie den RESET-Taster auf dem Modul kurz betätigen. Damit wird das „MODEM_COM“ Programm aus dem Flash gestartet.
- Als erstes soll die Kommunikation PC-Modem getestet und synchronisiert werden. Tippen Sie im HyperTerminal-Fenster „*AT<ENTER>*“. Als Anwort sollte Sie vom Modem „*OK<ENTER>*“ erhalten. Sollte dies nicht der Fall sein prüfen Sie bitte die PC-Modem Verbindung.

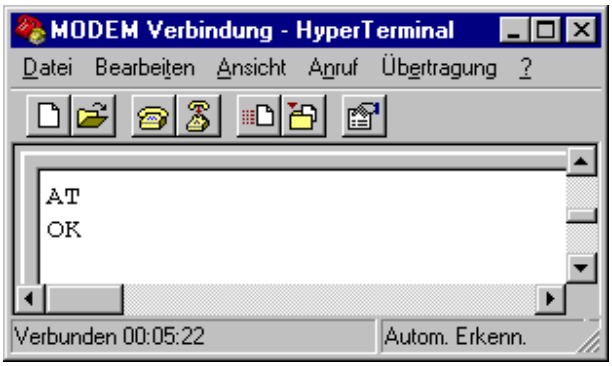


Bild 11: HyperTerminal: Modem Test / Synchronisation

- Im nächsten Schritt soll das Modem am grabbMODUL-4 angewählt und die Verbindung aufgebaut werden. Das Anwählen erfolgt durch „*ATDxxx<ENTER>*“ (**xx steht dabei für die MODUL-Modem Rufnummer!**). Das grabbMODUL-4 auf der Gegenstelle nimmt den Ruf beim ersten „RING“ an und nachdem die Modems untereinander die Verbindung aufgenommen haben sendet Ihnen das PC-Modem eine Antwort, die bei erfolgreicher Verbindung immer mit „*CONNECT ...*“ beginnt (siehe Bild 12).

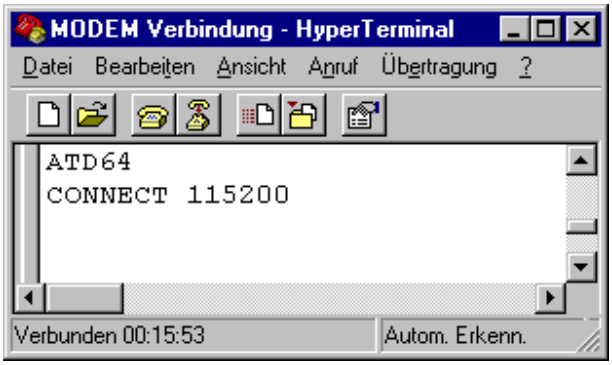


Bild 12: HyperTerminal: Verbindung aufbauen

- Nach der erfolgreichen Verbindungsaufnahme befinden sich beide Modems im DATA-Modus. Das heißt, der PC und das grabbMODUL-4 können über die serielle Verbindung kommunizieren als ob sie direkt miteinander verbunden wären. Als Beispiel soll nun die Funktion „Software ID abfragen“ getestet werden:
  Tippen Sie folgende Zeichen ein: „`S#`“
  Antwort des Moduls: „`q,S,0#s,03,05#`“ (je nach Version)

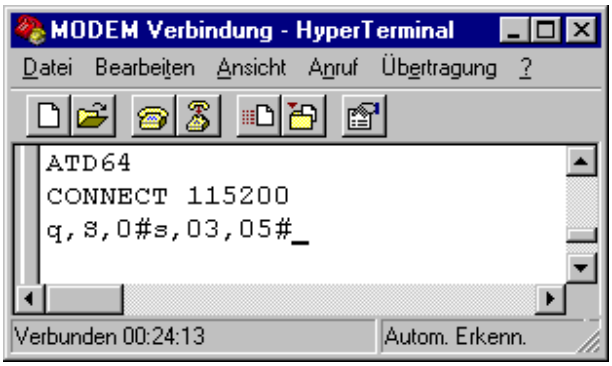


Bild 13: HyperTerminal: Test PC – Modul Kommunikation

- Die Beendung einer Verbindung sollte immer über das vereinbarte Protokoll realisiert werden (siehe 2.1.5). Geben Sie dazu den Befehl „*E,1#*“ ein. Das grabbMODUL-4 auf der Gegenstelle sendet erst die Quittung und dann nach ca. 1,1s (Guard-Time) wird in den Kommando-Modus gewechselt („+++“) und die Verbindung kontrolliert beenden („*ATH0*“ wird nicht immer geechot!). Auf der PC/Modem Seite wird dies nach einer bestimmten Zeit durch ein „*NO CARRIER*“ angezeigt (siehe Bild 14). Die Strecke ist nun bereit für eine erneute Verbindungsaufnahme.
  Nachdem das Modul-Modem aufgelegt hat, erkennt dies das Host-Modem und legt ebenfalls auf. Je nach Modemtyp kann die Zeitspanne bis zum Auflegen unterschiedlich sein. Sollten Sie bei einer langen Zeitspanne mit dem Hyperterminal sofort wieder wählen wollen, empfiehlt sich ein sofortiges Auflegen durch schließen/öffnen des Terminalprogramms zu erzwingen.

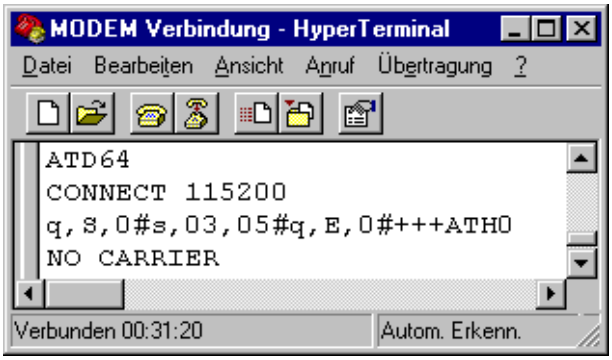


Bild 14: HyperTerminal: Verbindung beenden

Hinweis:
Wird eine Verbindung nicht über den Protokoll Befehl „*E,1#*“ beendet sondern auf der PC Seite aufgelegt, dann kann dies vom Modem nicht erkannt werden. In diesem Fall bemerkt das Modul die Störung nach den voreingestellten 4 min. und ist erst dann in der Lage, einen neuen Anruf entgegen zu nehmen.

- Auf diese Art und Weise können alle Protokollfunktionen getestet werden.

Hinweis:
Beim Test des „`I`“-Kommandos (Bilddaten-Anforderung) müssen Sie nach jedem Datenblock die Quittungsbestätigung „`q,i,0#`“ durch manuelle Eingabe im *HyperTerminal* senden.
Damit diese manuelle Eingabe nicht durch voreingestellte Timeoutzeiten unterbrochen wird, empfiehlt es sich, diese durch den `Z`-Befehl („`Z,30#`“), der in Handbuch grabbMODUL-4, L-612 Abschnitt „*Empfangs-Timeout der Schnittstelle setzen*“ beschrieben ist, zu erhöhen.
Da die Zeichenfolge „0x2B,0x2B,0x2B“ („+++“, kann in Bilddaten zufällig vorkommen) im DATA-Modus als Modem-Befehl interpretiert werden könnte (das Modem würde dann den Daten-Modus verlassen), wird diese Zeichenfolge vom grabbMODUL-4 beim Senden von Bilddaten unterdrückt.

- Zur Beendigung der Verbindung zwischen PC und Modem klicken Sie im Hyperterminal die Schaltfläche „*Verbindung beenden*“ oder schließen Sie das Programm.


“Verbindung beeenden”

Sollten Sie keine Ausgaben im HyperTerminal Fenster sehen, kontrollieren Sie bitte die Spannungsversorgung, die COM-Port - Parameter und die RS-232 Verbindung.

2.3 Hinweise zur Erstellung des PC Programms

An dieser Stelle einige Hinweise zur Erstellung des PC-Programms:

- Bevor Sie ein Programm zur Modem-Übertragung erstellen, sollten Sie zuerst eine lokale Kommunikation mit dem grabbMODUL-4 herstellen und ein Programmgerüst mit den wichtigsten Steuerbefehlen im lokalen Modus erstellen und testen. Das Modul Programm „LOCAL_COM“ findet sich auf der Treiber-CD SO-670. Die Beschreibung finden sie im Handbuch zum grabbMODUL-4 (Dokument L-612).
- Damit das PC-Programm über ein Modem kommunizieren kann, müssen folgende Schritte auf PC-Seite umgesetzt werden:
  - Initialisierung des Modems
  - Wahl einer bestimmten Telefonnummer (Gegenstelle am Modul).
  - Feststellen, ob Verbindung zustande gekommen ist.
  - Austausch der im Protokoll festgelegten Zeichen im seriellen Modus.
  - Möglichkeiten, um die Bildrohdaten als Bild darstellen zu können.
  - Einhalten der im Protokoll definierten Timeout-Zeiten.
  - Kontrollmöglichkeiten der Leitungsverbindung.
  - Kontrollierte Beendung einer Verbindung („*E,1#*“).
- Sie können die ersten Tests auch innerhalb einer internen Telefonanlage durchführen, um bei der Fehlerbeseitigung die Systeme wieder schnell in einen Ausgangszustand zu versetzen.

Hinweis:

- Je nach dem ob Sie gerade im lokalen- oder im Modem-Modus arbeiten, muß auch das entsprechende Gegenstück in den FLASH-Speicher des grabbMODUL-4 geladen werden.
- Die verwendeten Modems sollten den gleichen Übertragungsstandard verwenden (z.B. analog analog oder analog ISDN-Gerät im Analog-Geräte-Modus). Eine gemischte Verbindung z.B. analog – ISDN ist nicht möglich.
- Sollte sich eines der Modems auf der Telefonseite in einem undefinierten Zustand befinden (z.B. durch eine Störung) ,so ist dies meist nur durch ein Power off/on zu beheben. Damit eine Synchronisation zwischen Modul und Modem erfolgt, muß in diesem Fall auch das grabbMODUL-4 aus- und wieder eingeschaltet werden. Sollte dieser Zustand häufiger vorkommen, empfehlen wir, ein Modem mit interner „Watchdog“ Funktion zu verwenden.
- Beachten Sie, daß Modems gegebenenfalls geringfügig unterschiedliche Kommunikationsprotokolle haben können oder in Bezug auf die Ländereigenschaften (Leitungsdefinitionen) von einander abweichen können.

Prüfen Sie vor dem Einsatz eines Modems daher dessen Verwendbarkeit und Zuverlässigkeit genau. Wir empfehlen in diesem Zusammenhang gegebenenfalls die Durchführung eines Feldversuchs.

- Nachdem das Modul-Modem aufgelegt hat, erkennt dies das Host-Modem und legt ebenfalls auf. Je nach Modemtyp kann die Zeitspanne bis zum Auflegen unterschiedlich sein. Sollten Sie von der Hostseite sofort wieder wählen wollen, empfiehlt sich ein sofortiges auflegen durch einen Ein/Aus-Übergang der DTR-Leitung zu erzwingen.

2.4 Modem.lib – Funktionen für den AT-Befehlssatz

Die *Modem.lib* beinhaltet Funktionen zum Initialisieren und zur Kommunikation mit einem Geräte (Modem) über den AT-Befehlssatz.
Die Definition BYTE steht für „unsigned char“ und die Definition von WORD für „unsigned int“.

Wichtig: Zum Verwendung einer Funktion aus der *Modem.lib* muß die *Modem.h* in Ihr Projekt eingebunden (included) werden.

Folgende Funktionen werden in der *Modem.lib* zu Verfügung gestellt:
- Modem_Init()
- Modem_Call()
- Set_Modem()
- Read_Modem()
- Read_Modem_without_0D0A()
- Check_AT()
- Read_Error_Code_Modem()

Ein AT Befehlskommunikation ist üblicherweise wie folgt aufgebaut:

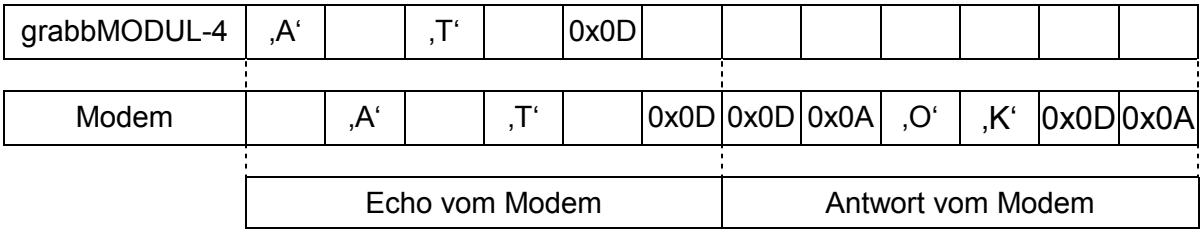


Bild 15: Kommunikation mittels des AT-Befehlssatzes (Beispiel)

Wichtig: Die Funktionen der Modem.lib werten auch Echos und Steuerzeichen aus. Deshalb ist es notwendig, daß sich die verwendeten Modems an dieses Standardprotokoll halten und die ECHO-Funktion des Gerätes nicht ausgeschaltet ist. In diesem Fall wird es bei einer Kommunikation immer zu einem Fehler kommen!

2.4.1 Modem_Init

Funktion: Initialisierung und Synchronisation der Baudrate mit einem Gerät (Modem) mittels AT-Befehl.

BYTE Modem_Init (void)

Rückgabewert: 0 = Gerät konnte erkannt werden, Synchronisation erfolgreich
1 = Es konnte kein Gerät erkannt werden

Die Funktion **Modem_Init()** muß sicherstellen, daß sich das Modem in einem definierten Zustand befindet. Deshalb wird zuerst die Zeichenfolge „+++“ (Wechsel von Data- in Kommando-Modus) gesendet. Als nächstes wird durch „*ATH0*“ sichergestellt, daß eine bestehende Verbindung beendet wird. Diese beiden Befehle werden „blind“ - ohne Auswertung der Antwort - geschickt. Nach 5 s wird die eigentliche Synchronisierung durch den Befehl „*AT*“ und das Testen der Antwort „*OK*“ als Entscheidungsgrundlage für eine erfolgreiche Initialisierung genommen.
Tritt bei der „AT“-Synchronisierung ein Fehler auf, wird in der Fehlervariable „*ERROR_CODE_MODEM*“ das Fehlerbit_0 (Bit-Wert 1)gesetzt.

2.4.2 Modem_Call

Funktion: Erkennt auf der seriellen Geräteverbindung einen Anruf („RING“), sendet den Befehl zur Rufannahme und testet die Ausführung.

Byte Modem_Call (void)

Rückgabewert: 0 = Verbindungsaufnahme erfolgreich
1 = Verbindung ist nicht zustande gekommen

Die **Modem_Call()**-Funktion vergleicht eine vollständig empfangene Zeichenfolge mit dem String „RING“. Bei Gleichheit wird an das Modem der Befehl „ATA“ zum Annehmen des Anrufs gesendet. Danach versucht das Modem mit der Gegenstelle selbständig eine Verbindung herzustellen. Dies kann je nach Leitungsqualität und aktuellen Transfer einige Zeit in Anspruch nehmen. Deshalb wird auf das Ergebnis der Verbindungsherstellung bis zu 30 s gewartet. Ist die Verbindungsaufnahme erfolgreich, wird dies vom Modem durch „CONNECT ...“ gemeldet. Da je nach Modem- und Leitungseigenschaften nach dem „CONNECT“ weitere Informationen folgen können, wird von der Software nur das einleitende „CONNECT“ überprüft und der Rest der Sendung verworfen.

2.4.3 Set_Modem

Funktion: Schreibt den übergebenen String entsprechend des AT-Protokolls an die serielle Schnittstelle.

BYTE Set_Modem (char *at_out_array)

*at_out_array: String mit zu sendender Zeichenfolge

Rückgabewert: 0 = Senden des Strings erfolgreich
1 = falsche bzw. keine Antwort vom Modem, Sendung abgebrochen

Die **Set_Modem()** Funktion sendet den ihr übergebenen String inklusive der Zeichen-Endekennung 0x0D. Nach jedem Zeichen wird erst das Echo überprüft, bevor das nächste Zeichen gesendet wird (siehe Bild 15: Kommunikation mittels des AT-Befehlssatz).

2.4.4 Read_Modem

Funktion: Empfängt eine Zeichenfolge entsprechend des AT-Protokolls von der serielle Schnittstelle.

BYTE Read_Modem (char *at_in_array)

*at_in_array: String in dem die empfangene Zeichenfolge abgelegt wird

Rückgabewert: 0 = Empfang einer AT-Sequenz erfolgreich
1 = Anfang- bzw. Ende-Kennung der Zeichenfolge auf der Schnittstelle falsch, bzw. keine Zeichen

Eine Sendung von einem Gerät mit AT-Befehlssatz wird immer mit 0x0D, 0x0A eingeleitet und mit 0x0D, 0x0A beendet (siehe Bild 15: Kommunikation mittels des AT-Befehlssatz). Die **Read_Modem()** Funktion erwartet von der Schnittstelle eine Zeichenfolge die mit diese Kennung gesendet werden. Die Zeichenfolge (ohne Kennung) wird dann in den „**at_in_array*“-String geschrieben. Sollten das Protokoll: fehlerhaft sein, das Senden von Zeichen ausbleiben oder 50 Zeichen überschreiten, dann kehrt die Funktion mit dem Fehlerparameter = 1 zurück und es wird in der Fehlervariable „*ERROR_CODE_MODEM*“ das Fehlerbit_1 (Bit-Wert 2) gesetzt.

2.4.5 Read_Modem_without_0D0A

Funktion: Empfängt eine Zeichenfolge entsprechend des AT-Protokolls von der serielle Schnittstelle ohne die Anfangskennung 0x0D und 0x0A.

BYTE Read_Modem_without_0D0A (char *at_in_array)

*at_in_array: String in dem die empfangene Zeichenfolge abgelegt wird

Rückgabewert: 0 = Empfang einer AT-Sequenz erfolgreich
1 = Ende-Kennung der Zeichenfolge auf der Schnittstelle falsch, bzw. keine Zeichen

Eine Sendung von einem Gerät mit AT-Befehlssatz wird immer mit 0x0D, 0x0A eingeleitet und mit 0x0D, 0x0A beendet (siehe Bild 15: Kommunikation mittels des AT-Befehlssatz). Die **Read_Modem_without_0D0A()** Funktion erwartet von der Schnittstelle eine Zeichenfolge bei der die Anfangskennung (0x0D, 0x0A) schon gelesen wurde. Die Zeichenfolge (ohne Kennung) wird dann in den „**at_in_array*"-String geschrieben Sollten das Protokoll: fehlerhaft sein, das Senden von Zeichen ausbleiben oder 50 Zeichen überschreiten, dann kehrt die Funktion mit dem Fehlerparameter = 1 zurück und es wird in der Fehlervariable „*ERROR_CODE_MODEM*" das Fehlerbit_1 (Bit-Wert 2) gesetzt.

2.4.6 Check_AT

Funktion: Synchronisation der Baudrate mit einem Gerät (Modem) mittels AT-Befehl.

BYTE Check_AT (void)

Rückgabewert: 0 = Gerät konnte erkannt werden, Synchronisation erfolgreich
1 = Es konnte kein Gerät erkannt werden

Die Funktion **Check_AT()** sendet den Befehl „*AT*" und Testen die Antwort „*OK*" um sicherzustellen daß, das angeschlossenen Gerät (Modem) noch einsatzbereit ist.
Tritt bei der „AT"-Synchronisierung ein Fehler auf, wird in der Fehlervariable „*ERROR_CODE_MODEM*" das Fehlerbit_0 (Bit-Wert 1)gesetzt.
Dieser Befehl sollte nur verwendet werden, wenn Sie sich im Kommando-Modus befinden.

2.4.7 Read_Error_Code_Modem

Funktion: Lesen der Fehlervariable „*ERROR_CODE_MODEM*".

BYTE Read_Error_Code_Modem (void)

Rückgabewert: 0x00 = kein Fehler aufgetreten
0x01 = Initialisierung Modem fehlgeschlagen
0x02 = Antwort des Modems falsch

3. CopyFunc: Library mit Funktionen zum Bildkopieren

3.1 Funktionsweise

Mit den Funktionen in der CopyFunc.lib stellt die PHYTEC Ihnen sehr effektive Assemblerroutinen zum Kopieren von Bilddaten zur Verfügung.
Diese Funktionen sollten immer dann genutzt werden, wenn Bilddaten schnell aus dem Nur-Lese-Speicher des grabbMODUL-4 (VIDEO-RAM) in den Arbeitsspeicher (RAM) des µController transferiert werden sollen.
Der Vorteil gegenüber einer xmemcpy()-Funktionen ist die bis zu **7mal schnellere Ausführung** (siehe Bild 17) und ein speicheroptimiertes Kopieren bei Grauwert-Bildern (siehe Bild 16).

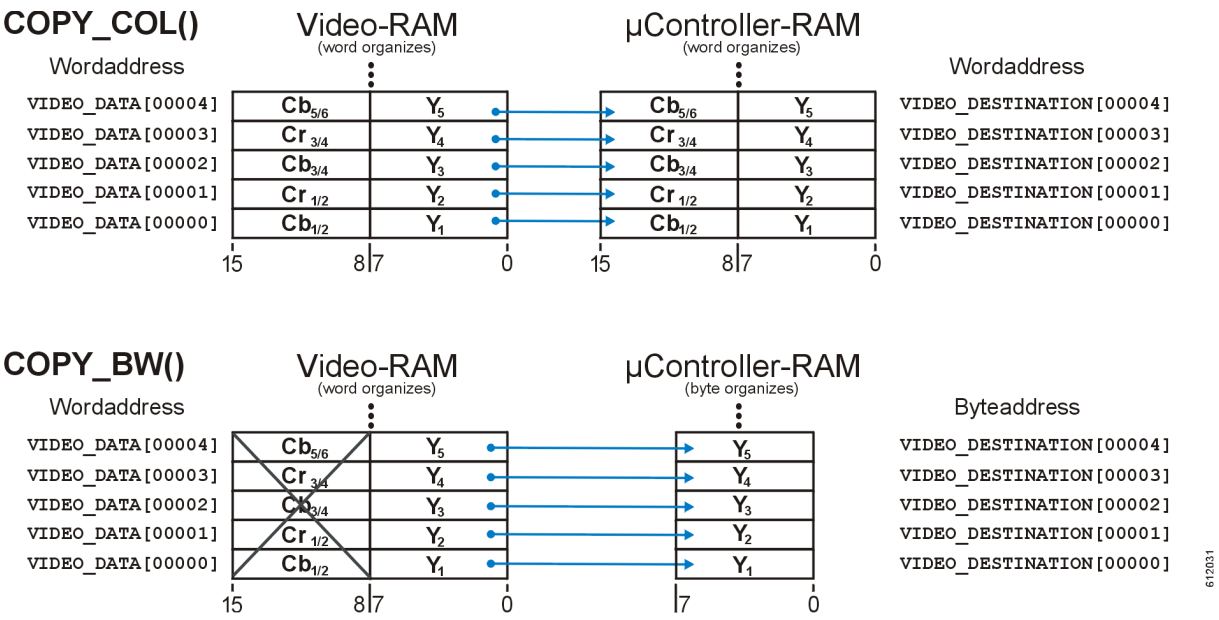


Bild 16: Speicherorganisation der CopyFunc.LIB-Funktionen

Hinweis: Beachten Sie das beim Kopieren der Bilddaten auch genügend Zielspeicher (µC-RAM) zu Verfügung stehen muß. Bei großen Bildern sollte ein Modul mit 1MB µC-RAM (z.B. VM-004-6) verwendet werden.
Die Funktionen der CopyFunc.lib sind für die Verwendung von xHuge Arrays ausgelegt, deshalb müssen Projekte die diese Library verwenden im HLarge Model compiliert werden.

3.2 CopyFunc.lib – Funktionen zum Bildkopieren

Die *CopyFunc.lib* beinhaltet Assembler-Funktionen zum schnellen Kopieren von Bilddaten aus dem „Nur-Lese“-Speicher des grabbMODUL-4 (VIDEO-RAM) in den Arbeitsspeicher (RAM) des µController. Die Definition BYTE steht für „unsigned char“ und die Definition von WORD für „unsigned int“.

Wichtig: Zum Verwendung einer Funktion aus der *CopyFunc.lib* muß die *CopyFunc.h* in Ihr Projekt eingebunden (included) werden.

Folgende Funktionen werden in der *CopyFunc.lib* zu Verfügung gestellt:
- Copy_Col()
- Copy_BW()
- Copy_Byte_Mem()

Die Funktionen der CopyFunc.lib können auch zum kopieren von Arrays ohne Bildinhalt verwendet werden. In diesem Fall sind die Besonderheiten der einzelnen Funktionen zu beachten.

Weitere Zeitoptimierungen der Funktionen sind im Abschnitt „3.2.4 Tips zur Zeitoptimierung der Copy-Funktionen“ beschrieben.

3.2.1 Copy_Col

Funktion: Schnelles Kopieren aller 16-Bit eines 16-Bit organisierten Quell-Arrays in ein 16-Bit organisiertes Ziel-Arrays (siehe Bild 16).

void Copy_Col (WORD xhuge *col_dest_ptr, WORD xhuge *col_source_ptr, unsigned long lenght)

*col_dest_ptr:	Pointer auf ein Array vom Typ WORD (Ziel)
*col_source_ptr:	Pointer auf ein Array vom Typ WORD (Quelle)
lenght:	Anzal der **Bytes** die kopiert werden sollen. Beachte: Anzahl der Bytes muß immer **gerade** sein!

Die Funktion **Copy_Col()** kopiert n Bytes (n = lenght) vom Array mit dem Startpunkt „*col_source_ptr“ in das Array mit dem Startpunkt „*col_dest_ptr“. Das Kopieren erfolgt dabei Wordweise (16Bit).

Beispiel:
Bei den folgendem Beispiel wird das Quell-Array VIDEO_DATA[max. 1MB] verwendet welches in den grabbMODUL-4 Beispielen schon vordefiniert ist.

```
...
unsigned int xhuge VD_BUFFER_col [65536];              // = 128kByte

main
{
Copy_Col(VD_BUFFER_col, VIDEO_DATA, 131072);      // Copy 64kByte Col-Pixel
}
```

Tips zur Beschleunigung der Routine finden Sie im Abschnitt „3.2.4 Tips zur Zeitoptimierung der Copy-Funktionen“.

Wichtig: Die Funktion *Copy_Col()* ist nicht in der Lage Speicherüberschneidungen oder falsch angegebenen Speicherbereiche zu erkennen. Die Anzahl der zu kopierenden Elemente muß auch bei Farbbildern in Bytes angegeben werden und immer gerade sein.

3.2.2 Copy_BW

Funktion: Schnelles Kopieren der jeweils unteren 8-Bit eines 16-Bit organisierten Quell-Arrays in ein 8-Bit organisiertes Ziel-Arrays (siehe Bild 16).

void Copy_BW (BYTE xhuge *bw_dest_ptr, WORD xhuge *bw_source_ptr, unsigned long lenght)

*bw_dest_ptr:	Pointer auf ein Array vom Typ BYTE (Ziel)
*bw_source_ptr:	Pointer auf ein Array vom Typ WORD (Quelle)
lenght:	Anzal der **Bytes** die kopiert werden sollen. Beachte: Anzahl der Bytes muß immer **gerade** sein!

Die Funktion **Copy_BW()** kopiert n Bytes (n = lenght) vom Array mit dem Startpunkt „*bw_source_ptr“ in das Array mit dem Startpunkt „*bw_dest_ptr“. Aus dem Quellarray werden jeweils nur die unteren 8Bit (low Byte) gelesen und in den Zielspeicher übertragen.

Beispiel:
Bei den folgendem Beispiel wird das Quell-Array VIDEO_DATA[max. 1MB] verwendet welches in den grabbMODUL-4 Beispielen schon vordefiniert ist.

```
...
unsigned char xhuge VD_BUFFER_bw [131072];          // = 128kByte

main
{
Copy_BW(VD_BUFFER_bw, VIDEO_DATA, 131072);     // Copy 128kByte BW-Pixel
}
```

Tips zur Beschleunigung der Routine finden Sie im Abschnitt „3.2.4 Tips zur Zeitoptimierung der Copy-Funktionen“.

Wichtig: Die Funktion *Copy_BW()* ist nicht in der Lage Speicherüberschneidungen oder falsch angegebenen Speicherbereiche zu erkennen. Die Anzahl der zu kopierenden Elemente muß in Bytes angegeben werden und immer gerade sein.

3.2.3 Copy_Byte_Mem

Funktion: Schnelles Kopieren eines 8-Bit Quellarray in ein anderes 8-Bit organisierten Ziel-Arrays).

void Copy_Byte_Mem (BYTE xhuge *byte_dest_ptr, BYTE xhuge *byte_source_ptr, unsigned long lenght)

*byte_dest_ptr: Pointer auf ein Array vom Typ BYTE (Ziel)
*byte_source_ptr: Pointer auf ein Array vom Typ BYTE (Quelle)
lenght: Anzal der **Bytes** die kopiert werden sollen.

Die Funktion **Copy_Byte_Mem()** kopiert n Bytes (n = lenght) vom Array mit dem Startpunkt „*byte_source_ptr“ in das Array mit dem Startpunkt „*byte_dest_ptr“. Aus dem Quellarray werden jeweils 8Bit gelesen und in den Zielspeicher übertragen.

Beispiel:

```
...
unsigned char xhuge BUFFER_Quelle [131072];          // = 128kByte
unsigned char xhuge BUFFER_Ziel [131072];            // = 128kByte

main
{
Copy_Byte_Mem(BUFFER_Quelle, BUFFER_Ziel, 131072);   // Copy 128kByte BW-Pixel
}
```

Tips zur Beschleunigung der Routine finden Sie im Abschnitt „3.2.4 Tips zur Zeitoptimierung der Copy-Funktionen“ wobei die Zeiten zur Übertragung eines Bytes mit der Übertragung eines Pixels der Copy_BW-Funktion verglichen werden können.

Wichtig: Die Funktion *Copy_Byte_Mem()* ist nicht in der Lage Speicherüberschneidungen oder falsch angegebenen Speicherbereiche zu erkennen.

3.2.4 Tips zur Zeitoptimierung der Copy-Funktionen

Die in Bild 17: Zeitmessungen der CopyFunc-LIB Funktionen“ dargestellten Zeiten sind für die Standardauflösungen berechnet worden. Mittels der Zeiten pro Pixel können diese auf beliebige Auflösungen umgerechnet werden.
Die einzelnen Spalten werden im folgenden beschrieben.

Controller mit 1 Waitstate beim Zugriff auf RAM-Bereiche (Standart)					
			Spalte 1	**Spalte 2**	**Spalte 3**
	Funktion:	Anzahl Pixel	xmemcpy(), Keil	Copy_Col/BW(), Phytec	Copy_Col/BW(), 16k optimiert
	Zeit pro Pix (16/8 Bit)	1 Pixel	4,49 µs	1,12 µs	0,80 µs
Full	Bild (160x120)	19200 Pixel	86,13 ms	21,53 ms	15,38 ms
	Bild (180x144)	25920 Pixel	116,28 ms	29,07 ms	20,76 ms
CIF	Bild (320x240)	76800 Pixel	344,53 ms	86,13 ms	61,52 ms
	Bild (360x288)	103680 Pixel	465,12 ms	116,28 ms	83,06 ms
QCIF	Bild (640x480)	307200 Pixel	1378,13 ms	344,53 ms	246,09 ms
	Bild (720x576)	414720 Pixel	1860,47 ms	465,12 ms	332,23 ms

Controller mit 0 Waitstate und no R/W Delay beim Zugriff auf RAM-Bereiche (Beachte: Änderung in startup / sartfla)					
	Funktion:	Anzahl Pixel	xmemcpy(), Keil	Copy_Col/BW(), Phytec	Copy_Col/BW(), 16k optimiert
	Zeit pro Pix (16/8 Bit)	1 Pixel	4,31 µs	1,04 µs	0,62 µs
Full	Bild (160x120)	19200 Pixel	82,76 ms	19,92 ms	11,87 ms
	Bild (180x144)	25920 Pixel	111,73 ms	26,89 ms	16,02 ms
CIF	Bild (320x240)	76800 Pixel	331,05 ms	79,69 ms	47,46 ms
	Bild (360x288)	103680 Pixel	446,92 ms	107,58 ms	64,07 ms
QCIF	Bild (640x480)	307200 Pixel	1324,22 ms	318,75 ms	189,84 ms
	Bild (720x576)	414720 Pixel	1787,70 ms	430,31 ms	256,29 ms

Bild 17: Zeitmessungen der CopyFunc-LIB Funktionen

Hinweis: Die Funktionen Copy_Col und Copy_BW benötigen die gleiche Zeit zum Kopieren eines Pixels. Da bei der Copy_Col-Funktion ein Pixel aus 16Bit besteht, wird in der selben Zeit die doppelte Menge an Bytes, als bei der Copy_BW (8Bit), bewegt.
Die Zeitmessungen der memcopy() und xmemcopy()- Funktionen basiert auf die Funktionen des µVison-2 Tools der Firma Keil.

Spalte 1)

Die verwendete Funktionen *xmemcopy()* gehören zum Standardumfang des Entwicklungssystem der Firma Keil für C16x Controller. Die Zeiten wurden mit der Version µVision-2 und der Einstellung „speed optimized“ ermittelt.

Spalte 2)

Zum Kopieren der Bilddaten wurden die Funktionen *Copy_Col()* und *Copy_BW()* der Firma Phytec verwendet. Mit diesen kann eine über 4mal schnellere Abarbeitung erreicht werden.

Spalte 3)

Eine weitere Zeitoptimierung ist möglich wenn man beachtet das der C165 intern mit Datenpage-Pointer arbeitet die für 16k große „Pages“ optimiert sind. Liegen nun Quell- und Zielpointer auf einer gleichen Page-Adresse (die unteren 14Bit haben den selben Wert) dann erkennen dies die *Copy_Col()* und *Copy_BW()* Routinen und arbeiten noch einmal bis zu 40% effektiver.
Voraussetzung ist nun das gezielte Plazieren (Mappen) der verwendetet Arrays. Der Quellspeicher (VIDEO_DATA[]) ist beim grabbMODUL-4 schon fest auf die Adresse 200000H gelegt. Das heißt der Zielspeicher muß auf einen Adressbereich gelegt werden bei dem die unteren 14Bit mit 0 belegt sind.
An dieser Stelle soll eine ihnen eine Möglichkeit gezeigt werden:

Mann definiert ein Array in einer Assemblerroutine und vergibt einen festen Speicherbereich mit dem „AT“-Befehl.
Öffnen Sie die „startfla.a65“ und vergeben einen Namen und einen Speicherplatz. Orientieren Sie sich dabei am Beispiel des VIDEO_DATA Arrays.

Startfla.a65:

```
...
PUBLIC  ?C_STARTUP, VIDEO_DATA, VD_BUFFER_col                    ;(Zeile 554)
...

; ********************************************************************************      (Zeile 1157)
; ***** LOCATE AN BUFFER INTO µC-RAM AT ADDRESS 048000H ******
; ********************************************************************************

VID_BUFFER1 section xdata AT 060000H 'XDATA2'

 VD_BUFFER_col      DSW    10000H        ; 128k from Adress 060000h to 07FFFFh
;VD_BUFFER_bw       DSB    10000H        ;   64k from Adress 060000h to 06FFFFh

VID_BUFFER1 ends
; ____________________________________________________________________________
```

Weiterhin muß dieses Array nun in Ihrem Programm bekannt gemacht werden um es dann zu verwenden.

TestProgramm.C65:

```
...
extern const volatile unsigned int xhuge VD_BUFFER_col[];      // Define in the startup/startfla.a65
...

main
{
Copy_Col(VD_BUFFER_col, VIDEO_DATA, 131072);              // Copy 64kByte Col-Pixel
}
```

Hinweis: Beachten Sie Ihre Speicherorganisation:
- Ist der verwendete Speicher physisch vorhanden?
- Gibt es Überschneidungen mit Ihrem Code-Speicher?

Überprüfen Sie die Einstellungen und die Lage Speicherbereiche in ihren MAP-Files.
Der Geschwindigkeitsvorteil durch feste Speicherbereiche sollte nicht durch unterschiedlichen Offsets bei der Pointerübergabe vergeben werden.

Controller mit 0 Waitstate: Spalte 1, Spalte 2 und Spalte 3)

Eine weitere Zeitoptimierung um ca. 20% ist möglich, wenn man die Zugriffszeiten auf die RAM-Bereiche des Controllers optimiert.
Standardmäßig ist der RAM Zugriff mit einem Waitstate (50ns) in der *sartfla.a65* und der *startup.65* voreingestellt. Beim Einsatz von 55ns RAMs kann ein Zugriff mit null Waitstate gefahren werden, wenn gleichzeitig der Buszugriff mit „no read / write delay“ gefahren wird.

Führen Sie folgende Einstellungen in der *startfla.65* bzw. *startup.a65* durch:

```
_MCTC1   EQU    0       ; Memory wait states is 0 (MCTC1 field = 0FH)
_RWDC1   EQU    1       ; 1 = No Delay Time     0 States

_MCTC2   EQU    0       ; Memory wait states is 0 (MCTC2 field = 0FH)
_RWDC2   EQU    1       ; 1 = No Delay Time     0 States
```

Hinweis: Wenn der Buszugriff verändert wird, müssen die in den Demo-applikationen verwendeten Delay-Funktionen neu kalibriert werden!