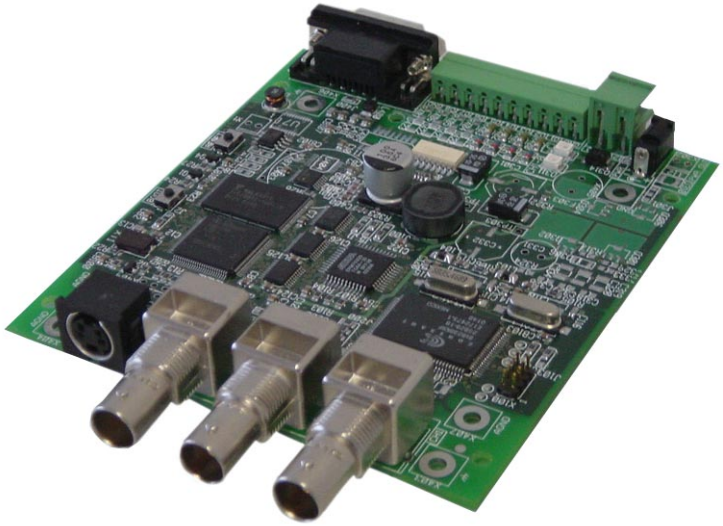
PHYTEC

grabbMODUL-4



Hardware Manual

Ausgabe Januar 2005

Ein Produkt eines Unternehmens der PHYTEC Technologie Holding AG

Im Buch verwendete Bezeichnungen für Erzeugnisse, die zugleich ein eingetragenes Warenzeichen darstellen, wurden nicht besonders gekennzeichnet. Das Fehlen der © Markierung ist demzufolge nicht gleichbedeutend mit der Tatsache, daß die Bezeichnung als freier Warenname gilt. Ebenso wenig kann anhand der verwendeten Bezeichnung auf eventuell vorliegende Patente oder einen Gebrauchsmusterschutz geschlossen werden.

Die Informationen in diesem Handbuch wurden sorgfältig überprüft und können als zutreffend angenommen werden. Dennoch sei ausdrücklich darauf verwiesen, daß die Firma PHYTEC Meßtechnik GmbH weder eine Garantie noch die juristische Verantwortung oder irgendeine Haftung für Folgeschäden übernimmt, die auf den Gebrauch oder den Inhalt dieses Handbuches zurückzuführen sind. Die in diesem Handbuch enthaltenen Angaben können ohne vorherige Ankündigung geändert werden. Die Firma PHYTEC Meßtechnik GmbH geht damit keinerlei Verpflichtungen ein.

Ferner sei ausdrücklich darauf verwiesen, daß PHYTEC Meßtechnik GmbH weder eine Garantie noch die juristische Verantwortung oder irgendeine Haftung für Folgeschäden übernimmt, die auf falschen Gebrauch oder falschen Einsatz der Hard- bzw. Software zurückzuführen sind. Ebenso können ohne vorherige Ankündigung Layout oder Design der Hardware geändert werden. PHYTEC Meßtechnik GmbH geht damit keinerlei Verpflichtungen ein.

Informieren Sie sich:

	EUROPA	NORD AMERIKA
Adresse:	PHYTEC Technologie Holding AG Robert-Koch-Str. 39 D-55129 Mainz GERMANY	PHYTEC America LLC 203 Parfitt Way SW, Suite G100 Bainbridge Island, WA 98110 USA
Angebots Hotline:	+49 (800) 0749832 order@phytec.de	+1 (800) 278-9913 sales@phytec.com
Technische Hotline:	+49 (6131) 9221-31 support@phytec.de	+1 (800) 278-9913 support@phytec.com
Fax:	+49 (6131) 9221-33	+1 (206) 780-9135
Web Seite:	http://www.phytec.de	http://www.phytec.com

1. Auflage Januar 2005

Bildverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Über dieses Handbuch

Dieses Handbuch ist in drei Teile gegliedert:

- **Teil 1 – Inbetriebnahme**

  ‚Dieser Teil beschreibt die Inbetriebnahme des grabbMODUL-4 mit dem Rapid-Development-Kit (Best.Nr. VPK-047).
  Sie werden Schritt für Schritt lernen, wie Sie eine Applikations-Firmware in das grabbMODUL-4 laden, die elektrischen Anschlüsse vornehmen und mittels einer PC-Gegenstellen-Software ein Bild über die serielle Schnittstelle übertragen.
  (Sie können die Inbetriebnahme auch ohne das Rapid-Development-Kit mit einer eigenen Videokamera vornehmen. Manche Schritte können dann von der Beschreibung abweichen.)

- **Teil 2 – Technische Daten**

  In Teil 2 finden Sie Angaben über technische Daten und Spezifikationen sowie die Anschlußmöglichkeiten des grabbMODUL-4.

- **Teil 3 – Programmierhandbuch**

  Im dritten Teil finden Sie eine Beschreibung der Treiber-Bibliothek und eine Einführung in die Programmierung des grabbMODUL-4. Dieser Abschnitt ist Referenz für den Programmierer, der eigene Programme für das grabbMODUL-4 (Firmware) oder ein Programm für einen Host-PC entwickeln möchte. Voraussetzungen dafür sind Kenntnisse der entsprechenden Programmierumgebung (z.B. Keil-Compiler für den C165-Microcontroller) und der Programmiersprache C.

Hinweis:
Weiterführende Informationen finden Sie auf unserer Homepage www.phytec.de unter SUPPORT > FAQs Bildverarbeitung.

1.2 Überblick

Das grabbMODUL-4 ist ein „stand-alone“-Bildverarbeitungsrechner auf Basis des Infineon C165-Microcontrollers.

Es integriert folgende Komponenten auf einer 100 x 110 mm großen Leiterplatte:

- vollwertigen 16-Bit Microcontroller-Rechenkern
- Farb-Framegrabber mit digitalem Videoprozessor
- Schaltspannungsversorgung
- I/O-Schnittstellen

Entsprechend des Einsatzgebiets gibt es verschiedene Möglichkeiten, das grabbMODUL-4 zu betreiben:

- **stand-alone-System**

  Anwendungsbeispiele:

  - Qualitätskontrolle: Füllstandsmessung, Anwesenheitskontrolle
  - Automatisierung: Nachführung von Solaranlagen

  Für die konkrete Anwendung wird ein spezielles Programm erstellt. Dieses Programm wird fest in das grabbMODUL-4 programmiert („Firmware“). Aufgrund dieses Programms bearbeitet der Rechenkern des grabbMODULs die Bilddaten völlig autark (ohne zusätzlichen Rechner).
  Verarbeitungsergebnisse können z.B. über die I/O-Ports auf einen Prozeß wirken oder über eine der Schnittstellen übertragen werden.

  Die Anwendungssoftware wird für die C165-Controller-Umgebung des grabbMODULs entwickelt. Dazu ist ein entsprechender Compiler (z.B. von der Fa. Keil) erforderlich.
  Die speziellen Komponenten des grabbMODUL-4 wie Framegrabber-Teil oder I/O-Ports können Sie auf einfache Weise ansprechen, indem Sie die mitgelieferten Treiber-Bibliotheken verwenden.

  PHYTEC erstellt auf Wunsch für Sie die entsprechende Anwendersoftware für Ihr Projekt.

- **Betrieb mit übergeordneten Host-Rechner**

  Anwendungsbeispiele:

  - Fernüberwachung / Fernwartung von verteilten Systemen
  - Zugangskontrolle, Security-Systeme

  Das grabbMODUL-4 ist mit einem übergeordneten Rechner verbunden (z.B. über die serielle Schnittstelle). Es nimmt vor Ort die Bilddaten auf, führt ggf. eine Vorverarbeitung durch und übermittelt die Daten an den übergeordneten Rechner.
  Dies kann auch über eine Modem- oder Datenfunk-Verbindung zu einem entfernt stehenden Rechner geschehen. Durch Verwendung der seriellen Schnittstelle können auch bestehende Datennetze genutzt werden.

  Hier wird auf dem grabbMODUL ebenfalls eine Firmware benötigt, die entsprechend der Aufgabenstellung mit dem Host-Rechner kommuniziert. PHYTEC liefert mit dem grabbMODUL-4 eine fertige Firmware mit, die eine einfache, unkomprimierte Bildübertragung über die serielle Schnittstelle ermöglicht. *Spezifikationen und Kommandos zu dieser Firmware finden Sie in Teil 3 dieses Handbuchs*.

1.3 Lieferumfang

1.3.1 Lieferumfang grabbMODUL-4 (VM-004)

- grabbMODUL-4 Leiterplatte (VM-004)
- Treiber- und Demo-CD (SO-670)
- PHYTEC Spektrum-CD (SO-376)
- dieses Manual (L-612)

1.3.2 Lieferumfang Rapid-Development-Kit (VPK-047)

- grabbMODUL-4 Leiterplatte (VM-004)
- Treiber- und Demo-CD (SO-670)
- PHYTEC Spektrum-CD (SO-376)
- Farbkamera VCAM-110-1 (AK039-1)
- S-Video-Kameraanschlußkabel (WK051)
- Nullmodem-Kabel (WK041)
- Steckernetzteil für grabbMODUL-4 (SV001)
- Steckernetzteil für Kamera (SV009)
- Video-Objektiv 16mm Brennweite (AO012)
- Kamera-Tischstativ (AZ004)
- dieses Manual (L-612)

1.4 Zubehör

Bei PHYTEC können Sie folgendes Zubehör gesondert erhalten:

- BNC-Kamera-Anschlußkabel, Länge 1 m, Best.Nr. WK057
- BNC-Kamera-Anschlußkabel, Länge 2 m, Best.Nr. WK058
- BNC-Kamera-Anschlußkabel, Länge 10m, Best.Nr. WK039
- S-Video-Kamera-Anschlußkabel, Länge 2 m, Best.Nr. WK051
- Nullmodem-Kabel zur Verbindung mit einem PC (serielle Schnittstelle, DB-9 Buchse), Best.Nr. WK041
- Steckernetzteil mit koaxialem Stromversorgungsstecker, 12 V, 500 mA, Best.Nr. SV001
- Steckernetzteil mit offenen Kabelenden, 12 V, 500 mA, Best.Nr. SV009
- Phoenix-Stecker für I/O-Port, mit geradem Kabelabgang, Best.Nr. GP131
- Phoenix-Stecker für I/O-Port, mit 90° gewinkeltem Kabelabgang, Best.Nr. GP102
- Phoenix-Stecker 2polig für Spannungsversorgungs-Eingang, mit geradem Kabelabgang, Best.Nr. GP028
- USB-RS232 (9 Pin) Konverter-Kabel, Best.Nr. ZUB-007

Teil 1

Inbetriebnahme

2 Rapid-Development-Kit Inbetriebnahme

In diesem Abschnitt zeigen wir Ihnen in einfachen und verständlichen Schritten die Vorgehensweise bei der ersten Inbetriebnahme des grabbMODUL-4. Die Inbetriebnahme wird Ihnen anhand der Komponenten, die in einem RDK grabbMODUL-4 enthalten sind, demonstriert. Parallel dazu können Sie mit dieser Anleitung eine Inbetriebnahme auch mit einem grabbMODUL-4 und eigenen Komponenten realisieren.

Folgende Schritte werden im folgenden erläutert werden:

- Systemanforderungen
  - Welche Hardwareanforderungen gibt es?
- Schnittstellen und deren Verwendung des grabbMODUL-4.
  - Was muß wie an das grabbMODUL-4 angeschlossen werden?
- Installation und Verwendung der PHYTEC FlashTools
  - Womit wird eine Software in das grabbMODUL-4 geladen?
- Download von Programmen im Hex-File-Format vom PC in den externen Flash Speicher des Moduls
  - Wie wird eine Software in das grabbMODUL-4 geladen?
- Testen eines Programms im grabbMODUL-4
  - Wie und womit kann ich das Programm im Modul testen?

Das in der Demo-Software gezeigte Beispiel demonstriert nur einen Teil der Möglichkeiten die Sie mit einem grabbMODUL-4 realisieren können. Um dieses Modul an verschiedenste Anwendungsgebiete anzupassen, muß eine der Aufgabenstellung angepaßte Software erstellt werden. Die Vorgehensweise ist in Abschnitt *6, „Treiber-Software“*- Grundlage erläutert.

2.1 Systemanforderungen

Zur Verwendung des grabbMODUL-4 sind folgende Hardware-Komponenten erforderlich:

1) Bei Verwendung des RDK „grabbMODUL-4“:

- das PHYTEC-RDK „grabbMODUL-4“
- ein IBM-kompatibler PC (486 oder höher, mit einem Windows Betriebssystem Windows9x/NT/2K/ME/XP) mit einer freien seriellen Schnittstelle[1]

2) Bei Verwendung des grabbMODUL-4:

- das PHYTEC grabbMODUL-4
- ein DB-9 serielles Kabel Buchse-Buchse („Nullmodem-Kabel“) zum Programmdownload
- Stromversorgung 8-28 V (2W) Gleichspannung (ungeregelt)
- Analoge Videokamera mit Spannungsversorgung und Objektiv
- Video-Verbindungskabel BNC bzw. S-Video auf Kameraanschluß
- ein IBM-kompatibler PC (486 oder höher, mit einem Windows Betriebssystem Windows9x/NT/2K/ME/XP) mit einer freien seriellen Schnittstelle[1]

Weiter werden für die Inbetriebnahme die mitgelieferten PHYTEC FlashTools und die grabbMODUL-4 Demosoftware benötigt.

Hinweis:
Benutzen Sie nach Möglichkeit ein fertig konfektioniertes Nullmodem-Kabel, um Fehler auszuschließen. Informationen zum Aufbau eines Nullmodem-Kabels finden Sie auf den FAQ-Seiten unserer Homepage.

[1] Optional kann eine freie USB-Schnittstelle verwendet werden. In diesem Fall wird ein USB-RS232/9 Pin Konverter-Kabel benötigt (Best.Nr. ZUB-007)

2.2 Schnittstellen des grabbMODUL-4

Die Anschlüsse des grabbMODUL-4 befinden sich an der rechten und linken Boardseite und sind bei einem Einbau in ein Gehäuse von außen zugänglich.

- Die Abbildung zeigt die Anschlußmöglichkeiten des grabbMODUL-4:

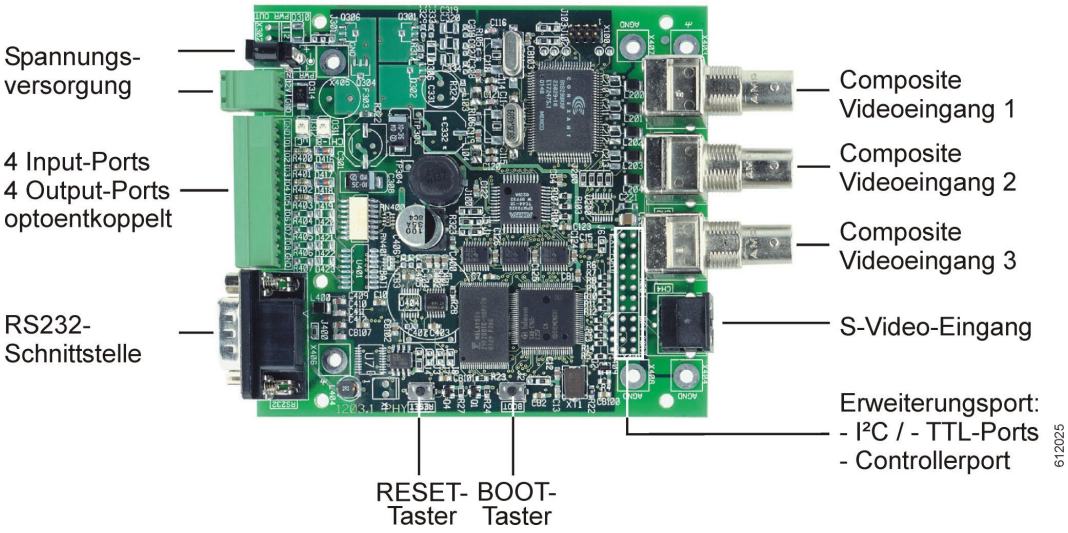


Bild 1: grabbMODUL-4: Übersicht der Anschlußmöglichkeiten

- Folgende Anschlüsse und Bedienelemente werden zum Download eines Hex-Files in den FLASH Speicher des grabbMODUL-4 benötigt:
  - Spannungsversorgung
  - PC (Host mit einer freien serieller Schnittstelle COM) an der RS232 Buchse über ein „Nullmodem-Kabel“ mit der RS232 Schnittstelle auf dem grabbMODUL-4
  - BOOT / RESET Taster
- Für den Betriebseinsatz werden weiterhin folgende Anschlüsse benötigt:
  - Standard Video Signal an einen Composite Videoeingang bzw. am S-Video-Eingang

Im Auslieferungszustand befindet sich bereits die aktuelle PHYTEC-Firmware „LOKAL_COM“ im Flash Speicher des Moduls. Zur Inbetriebnahme können Sie in diesem Fall direkt zu *Abschnitt 2.5, „Testen des Modul-Programms LOCAL_COM*“ gehen.

2.3 Anschlüsse des grabbMODUL-4 zum „Download“

Verbinden sie die RS-232 Schnittstelle (COM) ihres PCs mit der DB-9 (RS232) Buchse auf dem grabbMODUL-4 Board mittels dem seriellen „Nullmodem-Kabel“ (WK041).

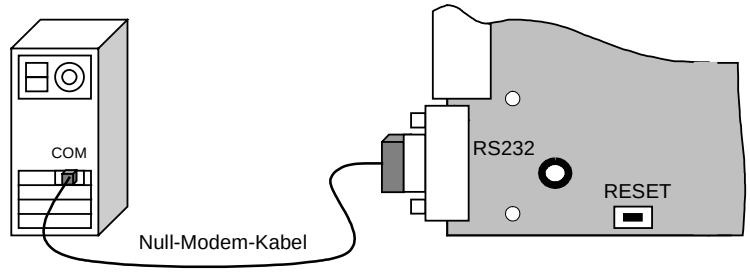


Bild 2: grabbMODUL-4: Verbindung zum PC

Verbinden sie die „PWR IN“ Buchse entweder über den NG-Stecker (Steckernetzteil SV001) oder über den 2-poligen Phoenix-Stecker mit einer 8-28 V (2W) Gleichspannungsquelle. Beachten Sie die in *Abschnitt 3.1.1* beschriebene Polarität der Steckverbinder. Die grüne LED uC leuchtet und zeigt Betriebsbereitschaft an.

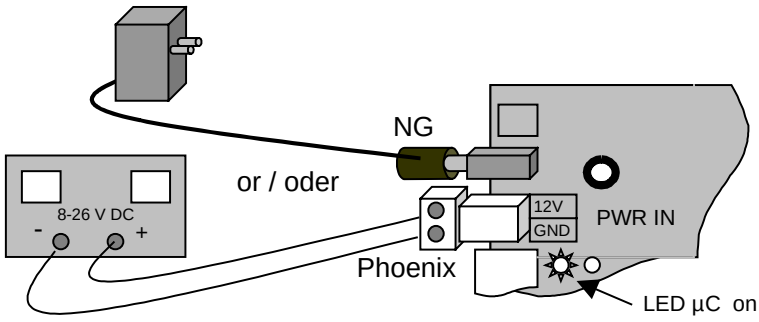


Bild 3: grabbMODUL-4: Power Verbindung

Drücken Sie gleichzeitig den RESET- und den BOOT- Taster *(siehe Bild 4)* auf dem grabbMODUL-4. Lassen Sie zuerst den RESET und dann, nach 2 bis 3 Sekunden, den BOOT Taster los.

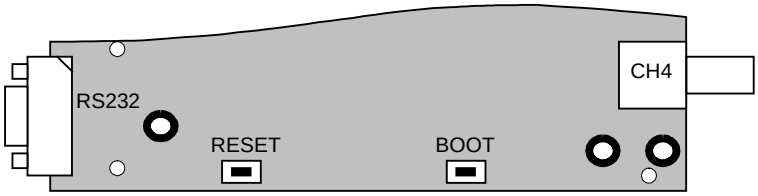


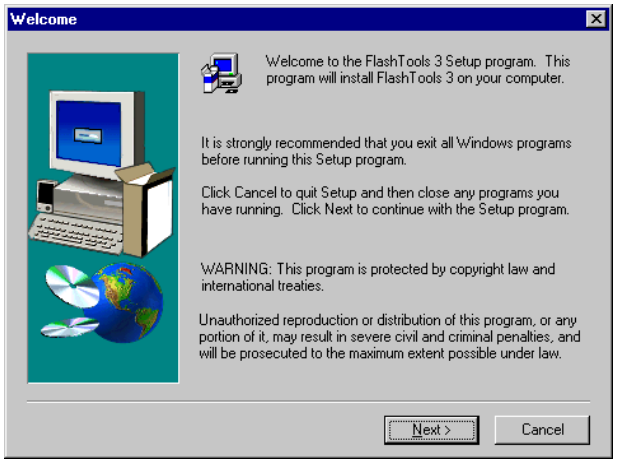
Bild 4: RESET und BOOT Taster

Diese Sequenz des Schließens und Öffnens des RESET- und des BOOT Tasters versetzt das grabbMODUL-4 in den „Bootstrap-mode“. Zur korrekten Arbeitsweise der FlashTools wird dieser Mode benötigt.

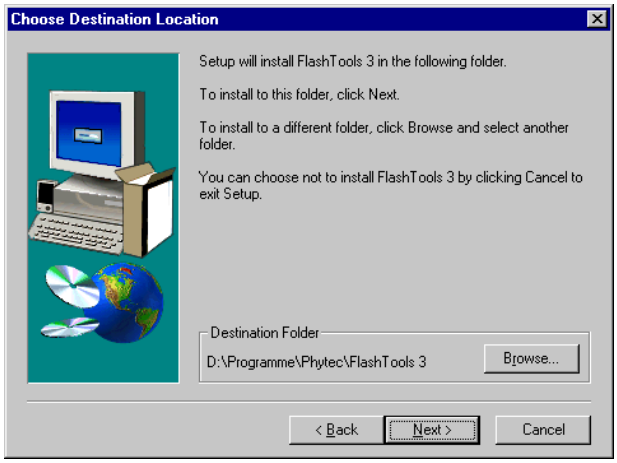
Das grabbMODUL-4 sollte nun mit einem PC über einem „Nullmodem-Kabel“ und mit einer Spannungsversorgung verbunden sein. Weiterhin sollte sich das Modul durch die BOOT/RESET Taster Sequenz im „Bootstrap-mode“ befinden. Das grabbMODUL-4 Board wird im weiteren als „Ziel-Hardware“ bezeichnet.

2.4 Installation der PHYTEC FlashTools und Download eines Programmcodes

Die aktuellen FlashTools befinden sich auf der Phytec-Spektrum-CD oder können auf der Phytec Homepage heruntergeladen werden. Starten Sie das Setup-Programm „setup.exe“ aus dem Verzeichnis ***.../Software/FlashTools3***.



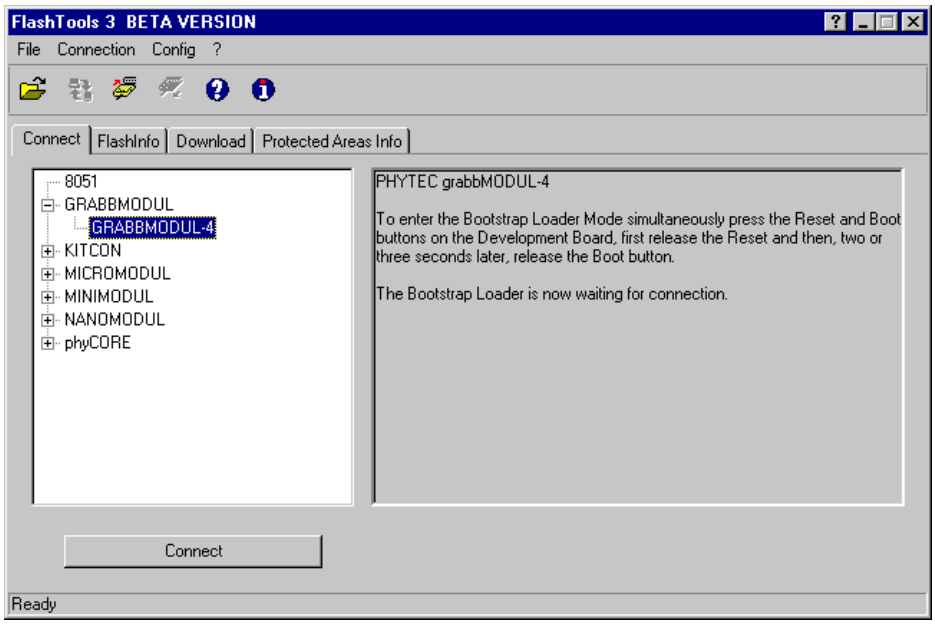
Folgen Sie den Anweisungen zur Vergabe der Pfad- und Applikationsnamen bis zum erfolgreichen Abschluß der Installation.



Nach Abschluß der Installation kann das FlashTools-Programm direkt gestartet werden. Es ist kein Neustart des Rechners erforderlich.

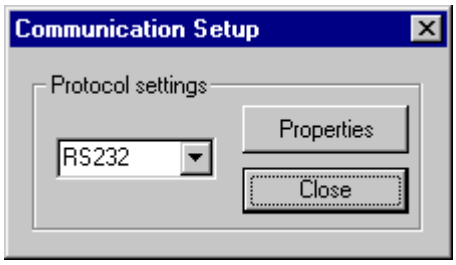
Bei einer Erstinstallation kann eine Abfrage nach einem Phytec „Registration Key“ erscheinen. In diesem Fall kann dieser Schlüssel kostenlos telefonisch/elektronisch bei der Phytec Meßtechnik GmbH angefordert werden.

- Starten Sie die FlashTools für Windows durch eine Doppelklick auf das FlashTools-Symbol oder wählen Sie die FlashTools aus der *Programme/PHYTEC FlashTools* Programmgruppe.
- Es erscheint „***Connect***“ als aktives Registerblatt.

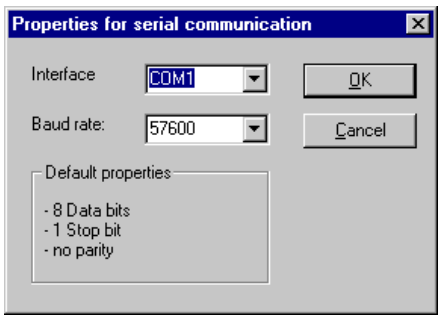


- Wählen Sie das Verzeichnis „*GRABBMODUL*“ und dann den Eintrag „*GRABBMODUL-4*“ aus der Ziel-Hardware-Liste.

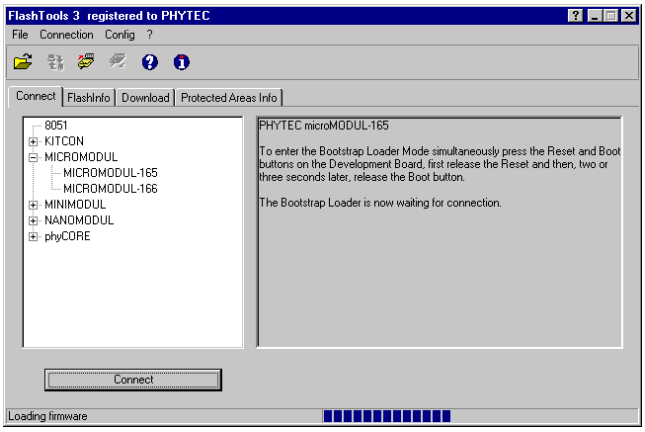
- Zur Auswahl der korrekten Einstellungen ihrer seriellen Schnittstelle wählen Sie im „***Config***“ Menü den Eintrag „*Protocol*“ zum Starten des „*Communication Setup*“.



- Wählen Sie den Eintrag „*RS232*“ und dann „*Properties*“.



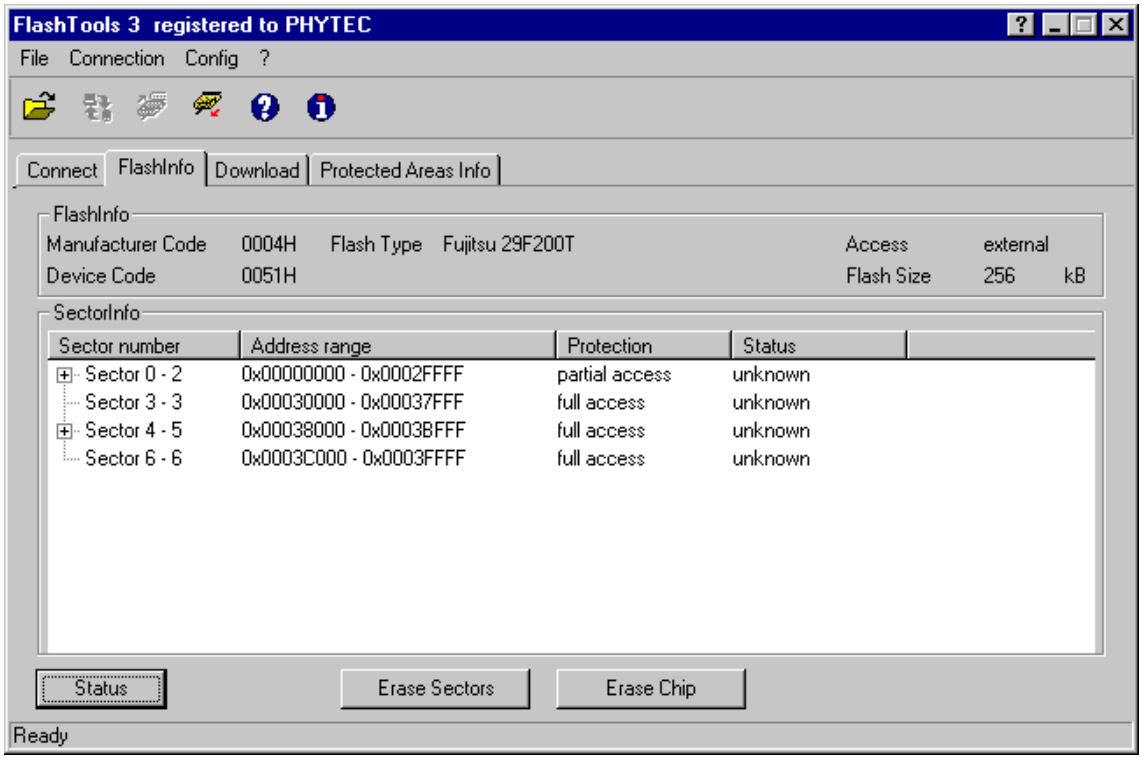
- Wählen Sie die COM-Schnittstellen Nummer (z.B. COM1), die Sie an Ihrem PC verwenden, und wählen Sie die „*Baud Rate*“ *57600*. Bestätigen Sie Ihre Wahl durch Klicken auf „*OK*“. Das „***Communicaton Setup***“ Fenster mit „*Close*“ wieder schließen.



- Im Registerblatt „***Connect***“ drücken Sie nun auf *Connect*, um den microcontroller-basierten Teil der FlashTools in die Ziel-Hardware zu übertragen.

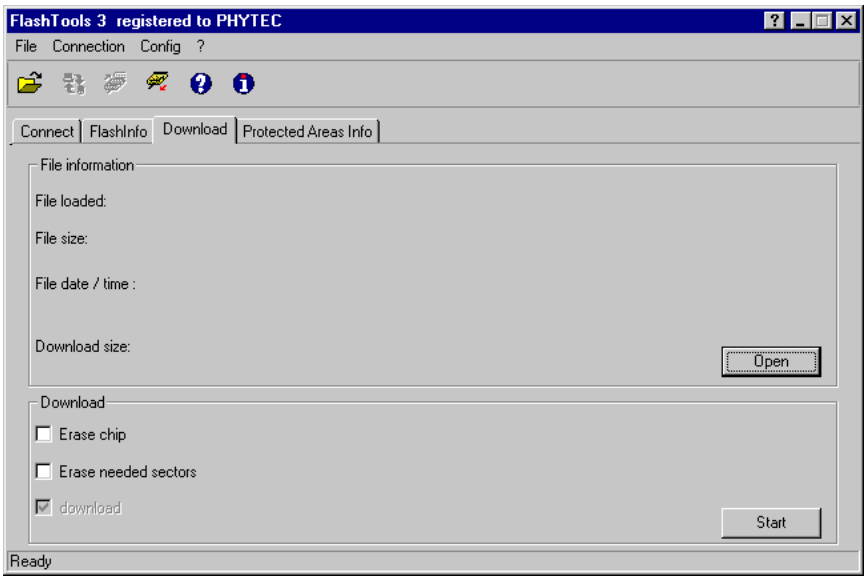
Der Microcontroller ist durch eine automatische Baudratenerkennung in der Lage, mit der vom Programm eingestellten Baudrate zu kommunizieren. Sollte ein Problem mit der eingestellten Baudrate auftreten, wählen Sie bitte eine andere Baudrate. Versetzen Sie das Modul erneut in den „Bootstrap-mode“ zum Testen der neuen Baudrate.

- Nach erfolgreichem Transfer der Daten wird im FlashTools Programm das Registerblatt ***„FlashInfo“*** aktiv. Dieses zeigt die Sektoren und deren Adressbereiche im Flash Speicher:

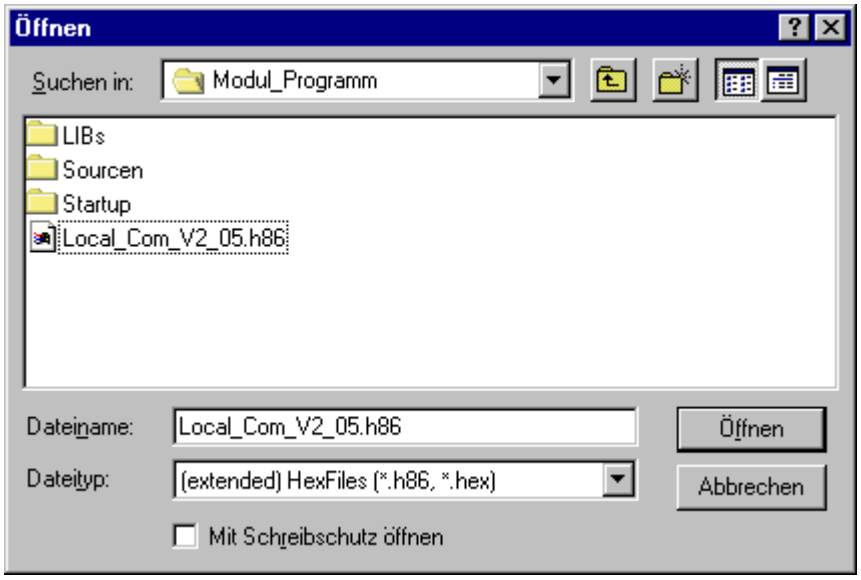


- Wählen klicken Sie auf „*Erase Chip*“, um alle Sektoren des Flash Speichers zu löschen. Danach sollte der Staus aller Sektoren als „blank“ (leer) angezeigt werden.

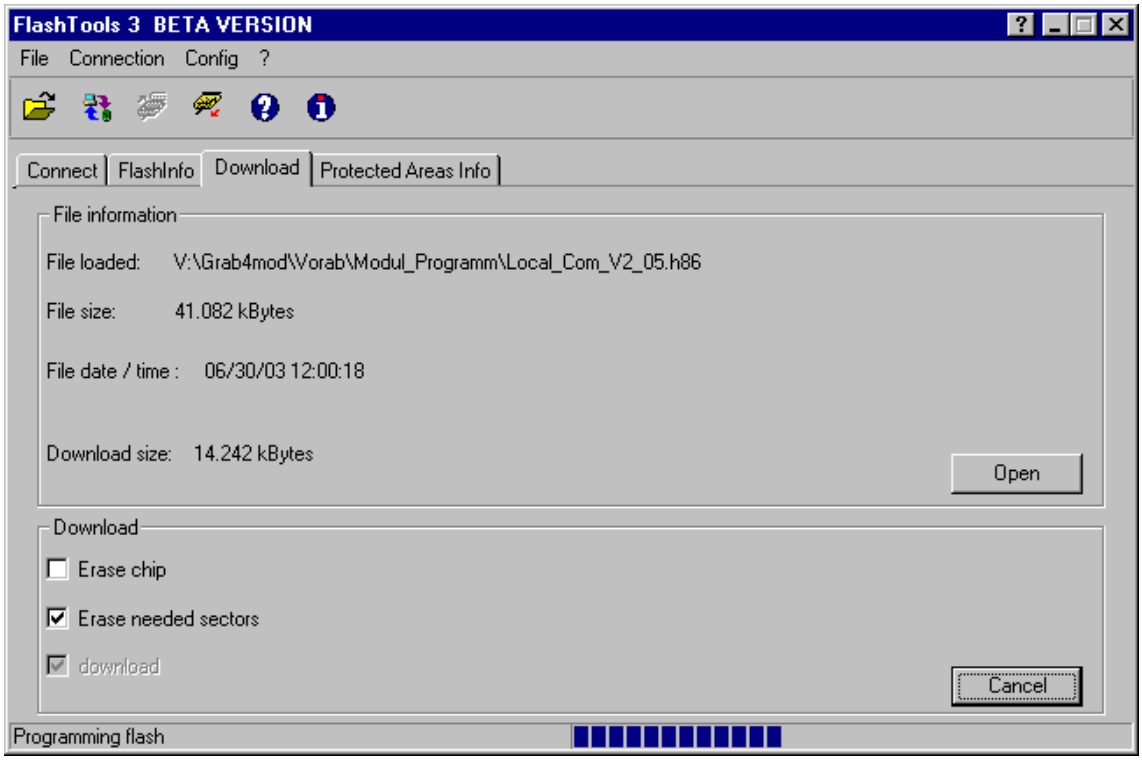
- Zur Auswahl und zum Download eines Hex-Files in den Flash Speicher wählen Sie das Registerblatt „***Download***“.



- Klicken Sie auf „*Open*“, um ein Hex-File zum Download auszuwählen.



- Wählen Sie das Hex-File, welches für das grabbMODUL-4 erstellt wurde, aus. Für den ersten Test wählen Sie die PHYTEC Firmware „*Local_Com_Vx_x.h86*“ (z.B. für Versionsnummer 2.5 = V2_05.h86) von der CD zum Download.
- Klicken Sie auf „*Öffnen*“.



- Klicken Sie nun auf die Schaltfläche „*Start*“, um den Download des Hex-Files zu beginnen. Während des Downloads ändert der „*Start*“-Button sein Aussehen in „*Cancel*“ und dient in dieser Zeit zum Abbruch des Vorgangs. Der Fortschritt des Downloads wird rechts unten als Balken dargestellt.

- Jetzt ist das Programm in den Flash Speicher des grabbMODUL-4 transferiert worden.. Zum Beenden der Verbindung schließen Sie das FlashTools-Programm.

Hinweis:
Das FlashTools Programm muß geschlossen werden, damit der COM Port des PCs freigegeben wird und die COM Schnittstelle von anderen Programmen verwendet werden kann.

2.5 Testen des Modul-Programms LOCAL_COM

Mit der Firmware LOCAL_COM („Local_Com_Vx_x.h86“) stellt PHYTEC Ihnen eine fertige Modulsoftware zur Verfügung, mit der Sie über die serielle Schnittstelle von einem Host-Rechner (z.B. einem PC) das grabbMODUL-4 steuern und Bilddaten vom Modul übertragen können.
Im Rahmen der Inbetriebnahme wird Ihnen im folgenden die Softwaregegenstelle auf dem PC erläutert. Eine detaillierte Beschreibung der Firmware, deren Funktionsweise und Einsatzmöglichkeiten sind im Abschnitt *6.4, „Arbeiten mit Phytec-Firmware*“ beschrieben.
Die Firmware LOCAL_COM kann mit einem Windows-Demoprogramm oder mit einem Terminalprogramm getestet werden.

2.5.1 Spannungs- und Kommunikationsverbindung

Verbinden das Modul mit einem PC und einer Spannungsversorgung wie es im Abschnitt *2.3, „Anschlüsse des grabbMODUL-4 zum „Download“* beschrieben ist.

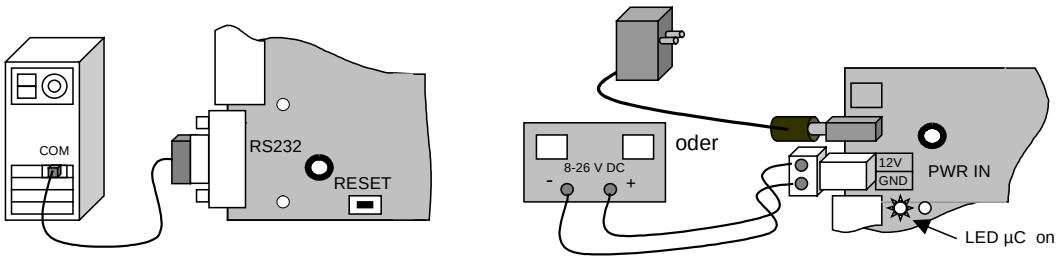


Bild 5: grabbMODUL-4: Power und PC Verbindung

Mit Anlegen der Versorgungsspannung ist das Modul betriebsbereit. Die grüne LED uC leuchtet und zeigt Betriebsbereitschaft an. Wenn Sie zuvor einen Download durchgeführt haben, drücken Sie die Taste RESET, um das Modul neu zu starten.

2.5.2 Kameraverbindung

Um Bilddaten zu digitalisieren, benötigen Sie eine analoge Standard-Video-Kamera mit einem entsprechenden Objektiv. Im Folgenden wird die Kamera des RDKs und deren Anschluß an das grabbMODUL-4 beschrieben. Achten Sie bei Verwendung eigener Kameras auf die richtige Norm des Videosignals.

Die Kamera

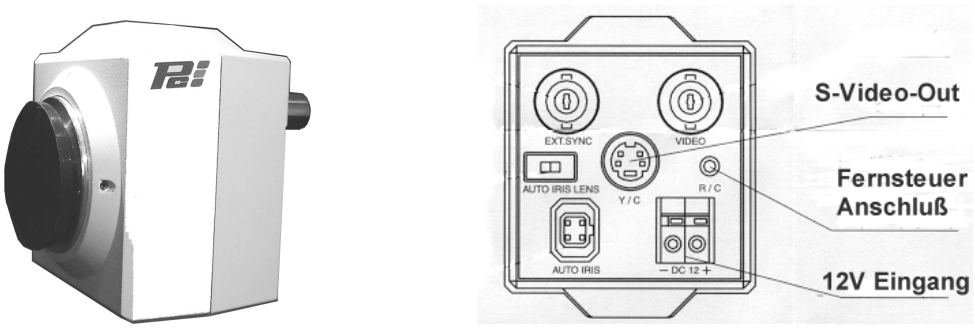


Bild 6: Die Kamera des Kits und deren Anschlußfeld

Hinweis:
Die Kamera wird ohne Objektiv geliefert. Die schwarze Kappe dient zum Schutz des Sensorchips und muß vor dem Anbringen des Objektives entfernt werden.

Die Rückseite (*Bild 6*) beinhaltet das Anschlußfeld, an das die Spannungsversorgung und das Videokabel angeschlossen werden. Die runde schwarze Buchse in der Mitte mit der Beschriftung Y/C ist die S-Video Buchse (sog. Mini-DIN Buchse), mit der die Kamera und das grabbMODUL-4 durch das S-Videokabel verbunden werden.

Verbinden Sie die Kamera und das grabbMODUL-4 mittels dem S-Videokabel (WK051). Verbinden Sie dazu S-Video-Out (*siehe Bild 6*) der Kamera mit dem S-Video-Eingang (*siehe Bild 1*) des grabbMODUL-4.
Bei Kameras mit Composite-Signalausgang verwenden Sie den Composite-Videoeingang 1 (BNC Buchse) am grabbbMODUL-4.

Es muß nun noch die Spannungsversorgung mittels dem Steckernetzteil SV009 angeschlossen werden. Auf dem Anschlußfeld der Kamera befindet sich ein viereckiges Klemmenpaar mit einer roten und einer schwarzen Klemme sowie der Beschriftung – *DC 12 +* (*Bild 6*). Halten Sie die kleinen grauen Tasten in der Mitte der Klemmen gedrückt und führen das **schwarze** Kabelende in die **schwarze** Klemme und das **rot** markierte in die **rote** Klemme.
Lassen Sie die Tasten wieder los; die Kabelenden sollten sicher eingeklemmt sein. Vergewissern Sie sich nochmals, ob die Polung stimmt und durch leichten Zug an den Kabeln, ob sie sicher verriegelt sind.
Die Kamera sollte nun wie im *Bild 7* abgebildet angeschlossen sein.

Bild 7: Kameraanschluß S-Video und Spannungsversorgung

Hinweis:
Beachten Sie die Polarität der Spannungsversorgung.

Das Objektiv und die Fokussierung

Das im Lieferumfang (nur RDK) enthaltene Objektiv (*Bild 8)* mit 16 mm Brennweite ist nach dem C-Mount Standard gefertigt.

Bild 8: Das C-Mount Objektiv

Die Kamera hat ein Gewinde, an das Sie alle Objektive, die ein C- oder CS-Mount Schraubanschluss besitzen, befestigen können. Bei C-Mount Objektiven muß der beim Kamerazubehör mitgelieferten 5 mm Zwischenring (*siehe Bild 9*) verwendet werden. Bei CS-Mount Objektiven müssen Sie den in *Bild 9* gezeigten 5 mm dicken Zwischenring weglassen.

Zusammenbau

Entfernen Sie zuerst die aufgesteckte Staubschutzkappe (*Bild 9*). Berühren Sie bitte **nicht** das Fenster vor dem Sensorchip.

Hinweis:
Bewahren Sie die Kappe gut auf. Sollte die Kamera ohne Objektiv transportiert werden, empfiehlt es sich, die Kappe wieder anzubringen, um den Sensorchip vor Fingerabdrücken, Staub etc. bzw. mechanischer Beschädigung zu schützen.

Nehmen Sie den 5 mm Adapterring und schrauben ihn, wie in *Bild 9* gezeigt, auf die Kamera.

Hinweis:
Der in der Kamera befindliche Gewindering darf dabei nicht verstellt werden. Die Fixierschrauben (Madenschrauben) an den Seiten dürfen nicht gelöst werden.

Schrauben Sie dann das Objektiv auf. Die Fokussierung wird später bei laufendem Betrieb der Kamera und grabbMODUL-4 vorgenommen.

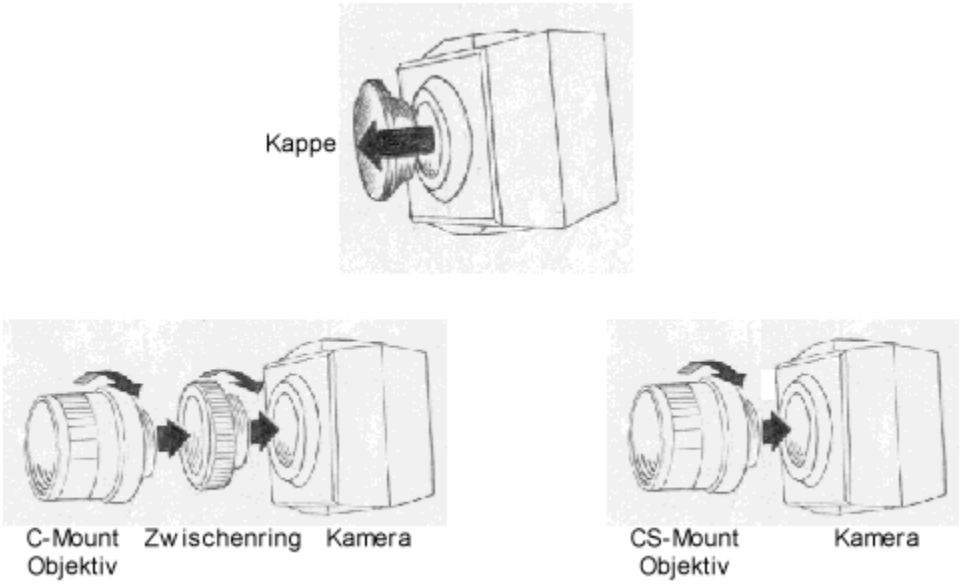
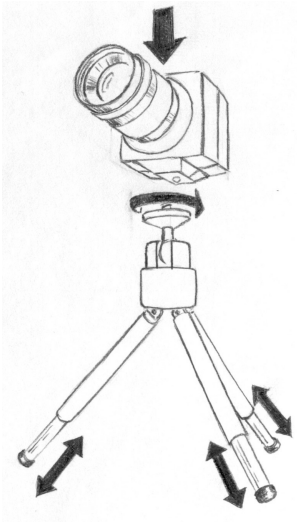


Bild 9: Montage des Objektivs auf die Kamera

Befestigen der Kamera am Stativ

Das mitgelieferte Stativ (nur RDK) dient der sicheren Befestigung und dem stabilen Aufstellen der Kamera.
Das Stativ besitzt einen beweglichen und arretierbaren Kugelkopf, durch den die Kamera in alle möglichen Positionen gebracht werden kann (waagerecht oder auch senkrecht).



Die drei Teleskopbeine lassen sich ausziehen.
Bei großen und schweren Objektiven sollten die Stativbeine herausgezogen werden, um ein Umkippen durch den verschobenen Schwerpunkt zu verhindern.
Lösen Sie die Sicherungsschraube des Stativkopfes und befestigen Sie die Kamera wie in *Bild 10* gezeigt.
Drehen Sie dazu den Stativkopf, um die Kamera zu verschrauben. Die Beschriftung auf dem Anschlußfeld der Kamera muß lesbar sein (bzw. BNC-Buchsen oben), um später ein aufrechtes Bild zu erhalten.

Bild 10: Kamerabefestigung auf Stativ

Hiermit sollten Sie alle Hardwarevoraussetzungen zur Inbetriebnahme geschaffen haben (*siehe Bild 11*).

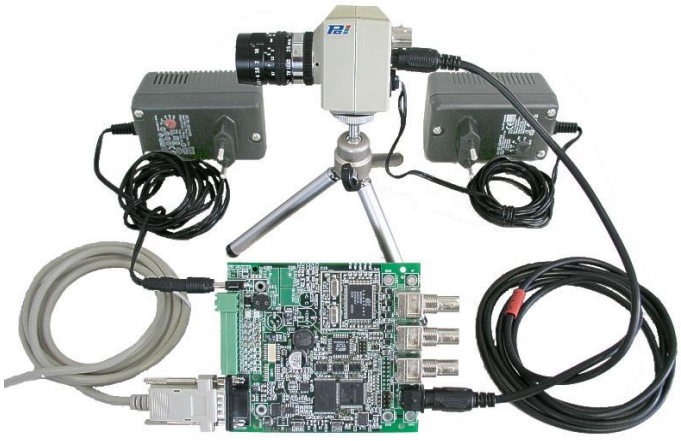


Bild 11: Betriebsfertiges RDK grabbMODUL-4

2.5.3 Testen des Modul-Programms mit dem Windows-Demoprogramm

Mit dem mitgelieferten PC-Demoprogramm „PC_Vxx.exe“ kann der Anwender mit der Firmware „LOCAL_COM“ auf dem grabbMODUL-4 über RS232 kommunizieren. Es können Einstellungen gesetzt und abgefragt werden sowie Bilddaten angefordert und dargestellt werden. Die Software arbeitet mit dem in *Abschnitt 6.4.1* beschriebenen Protokoll.
Die PC-Software ist unter den Windows-Betriebssystemen WIN98, ME, NT, 2000 und XP lauffähig.

Hinweis:
Das Demoprogramm kann nur die in „LOCAL_COM“ festgelegten Funktionen demonstrieren. Der wesentlich größere Funktionsumfang und vor allen die „stand-alone“ Fähigkeiten des grabbMODUL-4 können mit diesem Programm nicht demonstriert werden.

Windows-Demoprogramm starten und konfigurieren

Starten Sie die PC-Software „PC_Vxx.exe“ (xx steht für die aktuelle Versionsnummer). Das Programmfenster „Com Bild“ öffnet sich:

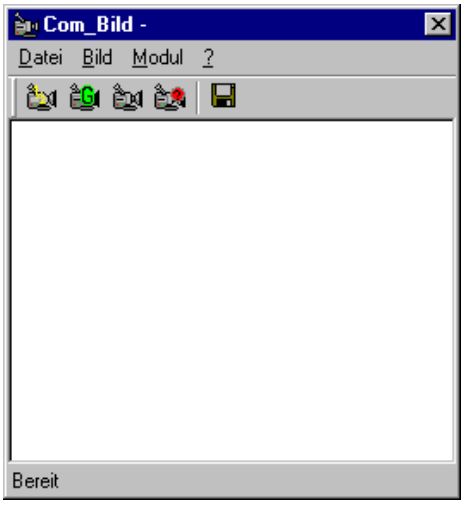


Bild 12: Windows-Demoprogramm für grabbMODUL-4

Die einzelnen Icons des PC Programms haben folgende Funktion:

				
Bild im 4-Bit NibbleMode abholen s/w	Bild auf Modul grabben u. speichern	Bild im 8-Bit Mode ab-holen	Aufruf des Menü Modulstatus	aktuelle Bild im Fenster speichern

Wählen Sie in diesem Programm den Punkt *Modul – Einstellungen*:

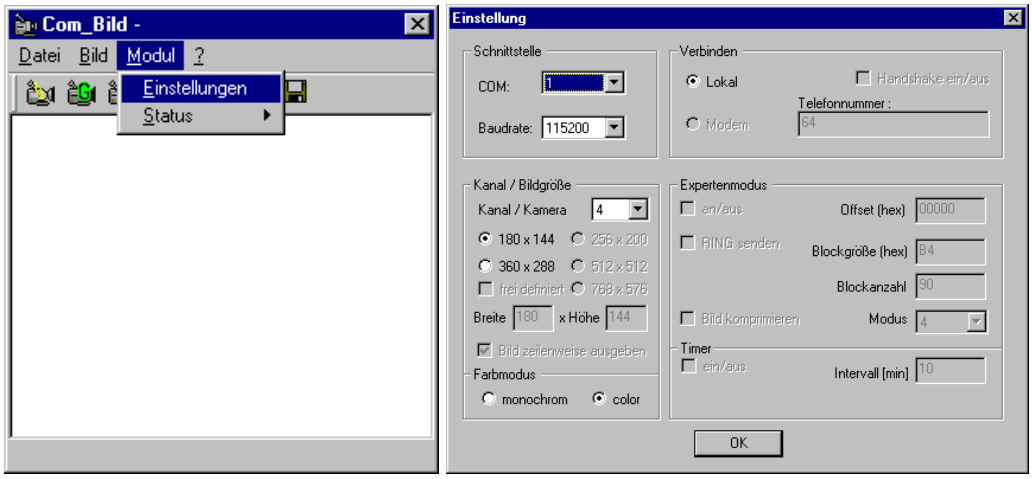


Bild 13: Windows-Demoprogramm: Menü Einstellung

In dem Registerblatt *Einstellung* (*siehe Bild 13*) wählen Sie bitte:

- die an Ihrem PC verwendete Schnittstelle (*COM*) und die *Baudrate 115200* aus
- unter „*Verbinden*“ den Modus *LOKAL* und „*Handshake*: **aus**“.
- unter Kanal/Bildgröße den *Kanal/Kamera „4“* (S-Video) und stellen Sie die Bildgröße zunächst auf *180 x 144* Pixel ein.
- Den gewünschten *Farbmodus „monochrom“* oder *„color“*.

Klicken Sie auf <*OK*>, um das Einstellungsfenster zu schließen.

Hinweis:
Wenn Sie Ihre Kamera an einen anderen Kanal als S-Video angeschlossen haben, müssen Sie diesen Kanal auswählen.
Wenn mit einer anderen Baudrate als 115200 Baud gearbeitet werden soll, so muß erst die Modulsoftware geändert werden. Die Modul-software besitzt keine automatische Baudratenerkennung!

Testen der Kommunikation und des Kamerasignals

Klicken Sie auf das Icon „Modulstatus“ und es erscheint das Registerblatt „Modulstatus“ (*Bild 14*):

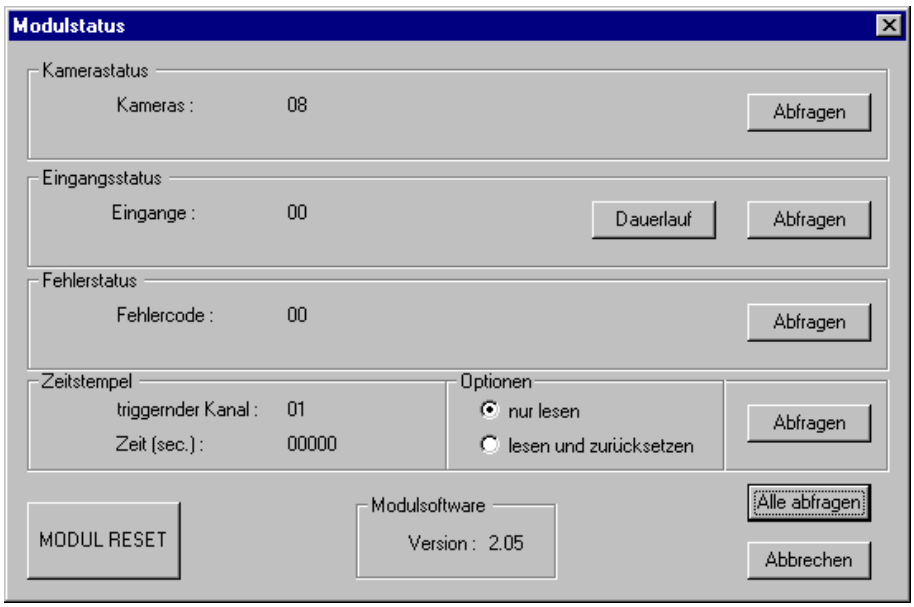


Bild 14: Windowsdemoprogramm: Menü Modulstatus

Klicken Sie bei *Kamerastatus* auf <*Abfragen*>. Es sollte bei einer korrekt angeschlossenen S-Videokamera eine „08“ (= Kanal 4, Beschreibung *siehe 6.4.6, „Kamera-Status anfordern*“) erscheinen.

Mögliche Ausgaben:

- Wert ist = 08: Es kann ein Bild angefordert werden.
- Es erscheint die Meldung „FehlerV“: Es konnte keine Verbindung mit dem grabbMODUL-4 hergestellt werden. Prüfen Sie die Einstellungen, das Verbindungskabel und die Spannungsversorgung.
- Wert ist = 00: Es wurde kein Kamerasignal erkannt. Prüfen Sie die Videoverbindung und Spannungsversorgung der Kamera.
- Wert ist < 08: Sie haben eine Kamera an einem anderen Kanal als S-Video angeschlossen. Es kann ein Bild angefordert werden, wenn im Einstellungsmenü der richtige Kanal ausgewählt wird.
- Wert ist > 08: Sie haben mehr als eine Kamera angeschlossen. Es kann ein Bild angefordert werden, wenn im Einstellungsmenü der richtige Kanal ausgewählt wird

Bilder übertragen

Klicken Sie nun auf das Icon „*Bild neu grabben*“, um ein aktuelles Bild von der Kamera zu grabben.

Dabei geschieht folgendes: Der PC sendet ein Steuerkommando an das Modul. Die Modul-Software („Firmware“) nimmt die entsprechenden Einstellungen des Video-Grabbers vor und startet die Bildaufnahme mit der vorgegebenen Bildgröße und dem gewünschten Farbformat. Das Bild wird im Bildspeicher des Moduls abgelegt.

Klicken Sie auf das Icon „Bild holen“, um das Bild abzuholen und anzuzeigen.

Dabei wird das Bild aus dem Modulspeicher ausgelesen und über die serielle Schnittstelle an den PC übertragen. Die PC-Software rechnet gegebenenfalls das Farbformat zur Anzeige um und stellt das Bild (*siehe Bild 15*) dann zeilenweise im Ausgabefenster dar:

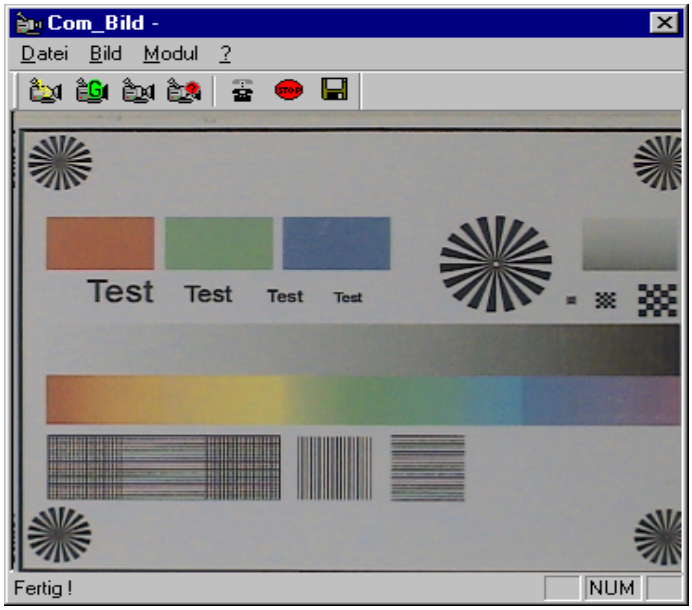


Bild 15: Windows-Demoprogramm: Anzeige des Bildes

Die Darstellungsgeschwindigkeit ist dabei durch die Übertragungsrate der seriellen Schnittstelle begrenzt. Beachten Sie, daß das Bild wesentlich schneller aufgenommen wird und im Modul zur Verfügung steht.

Mögliche Probleme:

1) Es erscheint ein unscharfes Bild:

- Stellen Sie das Objektiv scharf. (Nutzen Sie dazu das kleinste schwarzweiß Bild im Loop Modus (Taste F7). Beenden des Loop-Modus mit der *Esc-Taste*. **Bitte Beachten**: Das Bild wird dann noch komplett ausgelesen)

2) Es wird kein Bild angefordert und es erscheint die Fehlermeldung „Protokollfehler...“:

- Es besteht keine Verbindung zum Modul oder die Schnittstelle wurde falsch ausgewählt.
- Es wird mit einer anderen Baudrate als mit 115200 gearbeitet.
- Es wird kein Nullmodem-Kabel verwendet.

3) Es erscheint ein leicht verzerrtes Bild und am Ende die Fehlermeldung „Protokollfehler...“ oder „Time Out...“:

- Der FIFO-Buffer der Schnittstelle ist vom Betriebssystem zu hoch eingestellt (Änderung in ⇨ Systemsteuerungen ⇨ Anschlüsse (COM-PORT) ⇨ Erweiterte Einstellungen ⇨ FIFO verwenden und Empfangsbuffer auf „NIEDRIG“ stellen).
- Die serielle Verbindung ist gestört.

4) Es erscheint ein schwarzes oder blaues Bild:

- Es ist keine Kamera am ausgewählten Kanal angeschlossen.

5) Es erscheint ein verzerrtes Bild und keine Farbe:

- Es wurde eine NTSC statt einer PAL Kamera angeschlossen

Weitere Funktionen

Mit der Software können weitere Funktionen der Firmware getestet werden. Diese sind im Abschnitt *6.4.1* beschrieben und werden an dieser Stelle nur kurz aufgezählt.

Durch Klicken auf das Icon „*Bild im Nibble ½ holen*“ wird ein schwarz/weiss Bild im Nibble-Mode (4-Bit) abgeholt und angezeigt.

Durch Klicken auf das Icon „*Bild speichern*“ wird das aktuelle Bild im Ausgabefenster gespeichert.

Im Menü Modulstatus können Informationen zu Eingangs-/Ausgangsstatus, Fehlerstatus, Zeitstempel und Versionscode gelesen, sowie ein Modulreset durchgeführt werden.

2.5.4 Testen des Modul-Programms mit einem Terminal-Programm

Anstatt mit dem PC-Demoprogramm kann zum Test der Funktionen und des Protokolls kann auch 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 Antworten bzw. Bildrohdaten angezeigt werden.

Die im Folgenden beschriebene Vorgehensweise kann sich von Betriebssystem zu Betriebssystem leicht unterscheiden.

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

Bild 16: HyperTerminal: Programm erstellen

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

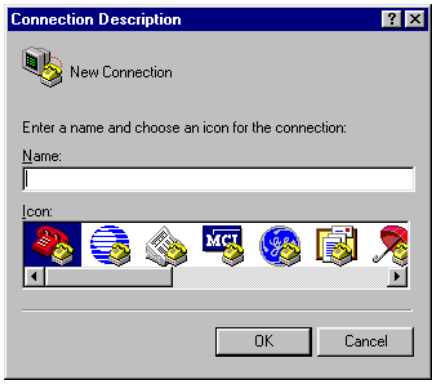


Bild 17: HyperTerminal: Programm benennen

- Es erscheint das *Verbindung Beschreibung-Fenster*. Geben Sie „*COM1 Verbindung*“ im Namensfeld ein (beachten Sie die Verwendung des korrekten COM Port an Ihrem PC).
- Nach dem Bestätigen durch Klicken auf „*OK*“ wird eine neue HyperTerminal Verbindung mit dem Namen „COM1 Verbindung“ hergestellt.
- Spezifizieren Sie im „*Eigenschaften von COM1 Verbindung*“-Fenster durch Drücken der „*Konfigurieren*“-Schaltfläche die Eigenschaften der COM-Verbindung.

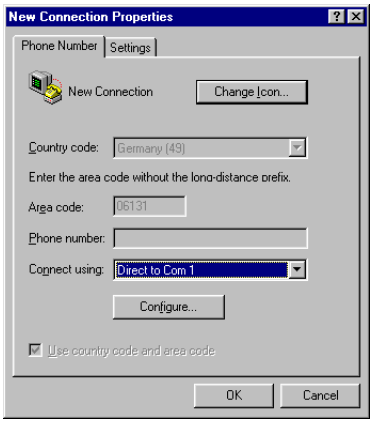
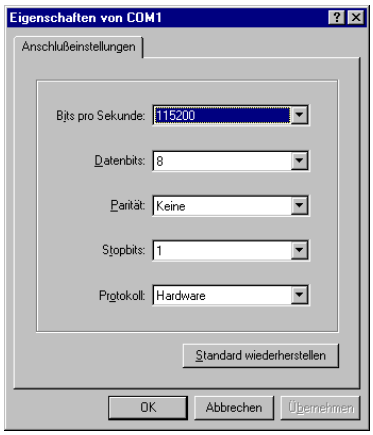


Bild 18: HyperTerminal: Programm Eigenschaften

- Wählen Sie im „*Eigenschaften von COM1*“ Fenster folgende COM Parameter:



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

Bild 19: HyperTerminal: Programm Konfigurieren

- Klicken Sie auf „*OK*“ und wechseln Sie zum Monitor-Fenster *“COM1 Verbindung – HyperTerminal”*.

Schließen Sie HyperTerminal und starten Sie es erneut, damit die durchgeführten Einstellungen übernommen werden!

- Nach dem Start des „*COM1 Verbindung – HyperTerminal*“ sehen Sie das Monitor-Fenster. Weiterhin werden in der unteren Leiste Statusinformationen über die Verbindung eingeblendet.

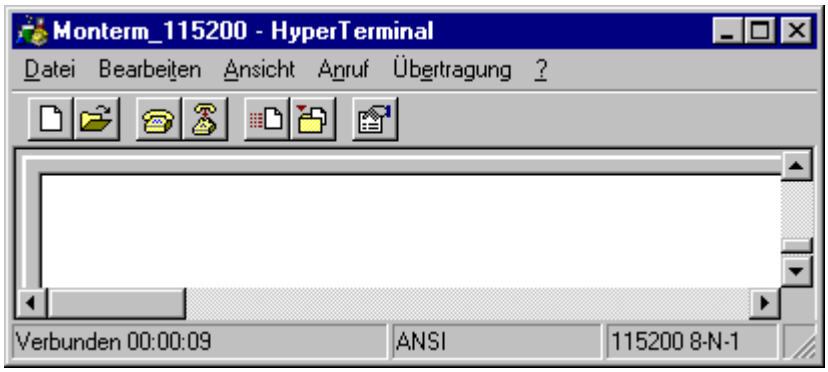


Bild 20: HyperTerminal: Programm starten

- Führen Sie einen Reset des grabbMODUL-4 Boards durch, indem Sie den RESET-Taster auf dem Modul kurz betätigen. Damit wird das Vx_x.h86 Programm aus dem Flash gestartet.
- Als Beispiel soll nun die Funktion „**Software ID abfragen**“ getestet werden:
  Tippen Sie folgende Zeichen ein: `S#`
  Antwort des Moduls: `q,S,0#s,02,05#`

Bild 21: HyperTerminal: Programm Test

- Wünschen Sie ein Echo der eingegeben Werte, dann aktivieren Sie die Echo-Funktion des HyperTerminals.
- 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 Abschnitt *6.4.16, „Empfangs-Timeout der Schnittstelle setzen*“ beschrieben ist, hochzusetzen.

- Zur Beendigung der Verbindung klicken Sie auf die Schaltfläche „*Verbindung beenden*“ in der HyperTerminal-Symbolleiste 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.

Teil 2

Technische Daten Hardwarebeschreibung

3 Technische Daten

Abmessungen: 100 x 110 mm (Leiterplatte)
100 x 140 mm mit Buchsen
(Einbau in Euro-Gehäuse möglich)

Gewicht: 130 g (Standardausbau VM-004)

Temperaturbereich:

Parameter	Condition	Min.	Typ.	Max.
Temperature	operating	0°C	-	70°C
	Storage	-40°C	-	90°C
Humidity (r.F.)	not condensed	-	-	95 %

Versorgung: 8..28 V DC ungeregelt

Parameter	Condition	Min.	Typ.	Max.
Pwr Supply Voltage	-	8 V	-	28 V
	Abs. Maximum	-	-	30 V

Leistungsaufnahme:

Parameter	Condition	Min.	Typ.	Max.
Power Consumption	operating	-	2 W	-
	Framegrabber Sleep-Mode	-	1.2 W	
	Grabber Sleep + CPU Idle	0.9 W		

Video-Eingänge: (Modell VM-004)
3 Composite-Videoeingänge, 75Ω, $1V_{ss}$
1 S-Video-Eingang 75Ω ($0,7\ V_{ss}$ / $0,3V_{ss}$)

Videoformate: PAL (B,G,H,I), NTSC (M)
bzw. entsprechende CCIR-Formate monochrom

Synchronisation:	Composite-Sync. bzw. Sync. auf Y-Signal externe Synchronisation nicht möglich	
Bilddatenformat:	16 Mio. Farben: YCrCb 4:2:2 256 Graustufen: Y8	
Bildauflösung:	max. 768 x 576 Pixel (PAL) bzw. 640 x 480 Pixel (NTSC) Auflösung frei skalierbar in X- und Y-Richtung bis 14:1	
Bildeinzug:	Halbbild: 20 ms Vollbild: 40 ms Bildtransfer in den Controllerspeicher in Echtzeit ohne Nutzung von Controller-Ressourcen	
Bildkorrektur:	Gammakorrektur Helligkeit (+/- 50 %) Kontrast (0%...235 %) Farbsättigung (U: 0...201 %, V: 0...283 %) Farbton (+/- 90°, nur NTSC)	
Bildspeicher:	1 MB SRAM (524.288 Bildpunkte) sequentielle Speicherung von 1. und 2. Halbbild	
Rechenkern:	16-Bit Infineon C165	
Taktfrequenz:	25 MHz	
Speicherausbau:	RAM:	256 kByte, optional bis 1 MB, zuzügl. Bildspeicher
	FLASH:	256 kByte, optional bis 1 MB
	EEPROM:	1024 Byte, seriell
Realtime-Clock:	optional	

Ports: 4 optoentkoppelte Signaleingänge

Parameter	Condition	Min.	Typ.	Max.
Input Low Voltage	-	0 V	-	5.5 V
Input High Voltage	-	6.5 V	-	24 V
Input Voltage	Absolute Max.	-	-	26 V
Input High Current	V_{IH}=26 V	-	-	7 mA
Reverse Voltage	Absolute Max.	-	-	-21.5 V
Reverse Current	V_{IR}=-21.5 V	-	-	7 mA
Switch Freq. (symm. Rectangle)	V_{IH}=6.5 V	40 Hz	-	-
	V_{IH}>7.0 V	-	90 Hz	-

4 optoentkoppelte Signalausgänge open collector, gemeinsame Emitter

Parameter	Condition	Min.	Typ.	Max.
CE-Breakdown Volt.	Ic=0,5mA	80 V	-	-
EC-Breakdown Volt.	I_E=0,1 mA	7 V	-	-
Collector Dark Current	V_{CE}=48V	-	0.01uA	0.1uA
	V_{ce}= 48V, T_a=85°C	-	2 uA	50 uA
Capacitance (C-E)	V=0, f=1MHz	-	10 pF	-
C-E Saturation Volt.	switched ON	-	0.3 V	-
Off-State Collector Current	V_{CE}= 48V	-	-	15 uA
max. Power P_c max	1 x ON	-	-	100mW
Collector Current	Rc=1k, Uc=10V	-	5 mA	-
ΔP_T/°C	Maximum Rating	-	-	-1.7 mW/°C

1 I²C-Schnittstelle (Master, software-emuliert)

4 TTL-I/O Leitungen (Controllerports)

Parameter	Condition	Min.	Typ.	Max.
Input Low Voltage		-0.5 V	-	0.9 V
Input High Voltage		1.9 V	-	5.5 V
Output Low Voltage	I_{OL}=1.6 mA	-	-	0.45 V
Output High Voltage	I_{OH}=-250 uA	4.5 V	-	-
	I_{OH}=-1.6 mA	2.4 V	-	-

3.1 Anschlüsse

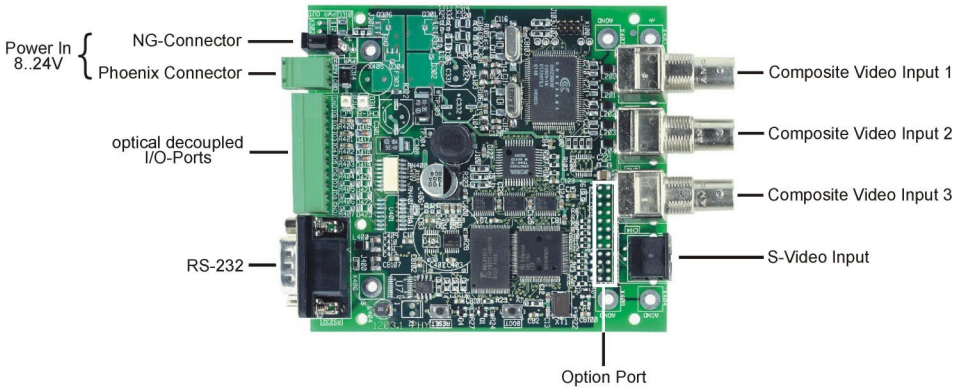


Bild 22: Übersicht über die Anschlüsse

3.1.1 Spannungsversorgung

Die Spannungsversorgung des grabbMODUL-4 kann entweder über den koaxialen Spannungsversorgungsstecker („NG-Stecker“) oder die zweipolige Phoenix-Buchse erfolgen (*siehe Bild 22*).

Der zulässige Eingangsspannungsbereich beträgt 8...28 V Gleichspannung. Als Spannungsquelle genügt ein ungeregeltes Gleichspannungsnetzteil. Die Leistungsaufnahme des grabbMODUL-4 beträgt ca. 2 Watt. Der Spannungsversorgungs-eingang des grabbMODUL-4 ist verpolungssicher ausgeführt.

Passender NG-Steckerdurchmesser: 1,1 mm

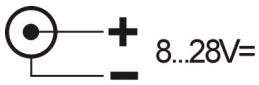


Bild 23: Polarität NG-Buchse (X300)

Der Phoenix-Stecker bietet die Möglichkeit, die Stromversorgung über eine rastbare Schraubklemmenverbindung durchzuführen. Der passende Stecker kann bei PHYTEC unter der Bestell-Nr. GP088 bezogen werden.

Die Anschlußbelegung entspricht dem Aufdruck auf der Leiterplatte:

Spannungsversorgung X301	
Pin	**Funktion**
12V	+8...28 V Power in
GND	Power Ground

Tabelle 1: Belegung Spannungsversorgungsstecker

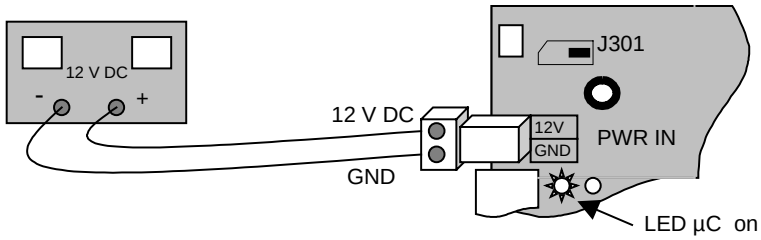


Bild 24: Anschluß der Spannungsversorgung

Bei korrekt anliegender Spannungsversorgung leuchtet die grüne LED uC .

Hinweise:
Die Spannungsversorgung ist durch die Sicherung F302 auf der Unterseite des Moduls mit 1A abgesichert. Sollte die LED uC trotz korrekt angelegter Versorgung nicht leuchten, kann die Sicherung defekt sein. In diesem Fall muß das Modul durch PHYTEC überprüft werden.

Die Stecksicherung F303 ist im normalen Betrieb des grabbMODUL-4 nicht erforderlich und deshalb im Standard-Auslieferungszustand nicht bestückt.

Achtung!
Die Spannungsversorgung ist eingangsseitig mit einer Transil-Diode gegen Überspannungstransienten geschützt.
Überschreitet die Versorgungsspannung den Wert 28 V, so kann dies zu Schäden auf dem Board oder an der Spannungsversorgung führen. Gegebenenfalls ist extern eine entsprechende Sicherung einzuschalten.

Durch Anlegen der Betriebsspannung beginnt das Modul mit der Ausführung der im FLASH programmierten Firmware. Ein zusätzlicher manueller Reset ist nicht erforderlich. Es sollte jedoch darauf geachtet werden, daß die Versorgungsspannung beim Einschalten sauber ansteigt und keine Einbrüche aufweist, damit ein korrekter interner Modul-Reset sichergestellt ist.

Bei Bedarf kann das grabbMODUL-4 durch ein Steuersignal eingeschaltet werden. Dazu dient der *Shutdown*-Anschluß TP304. Durch Verbindung des TP304 mit Masse kann das Modul *aus*geschaltet werden. Um das Modul einzuschalten, wird TP304 *offen* gelassen.

Hinweis:
TP304 darf nicht mit V_{CC} oder einer anderen Spannung verbunden werden. Der Pin darf nur gegen GND geschaltet werden. Gegebenenfalls ist eine Schottky-Diode (z.B. BAT42) so einzuschalten, daß Strom nicht in den Pin hineinfließen kann.

3.1.2 Serielle Schnittstelle

Die Serielle Schnittstelle des Moduls ist an dem 9-poligen Sub-D-Stecker RS232 (P400) herausgeführt. Die Buchsenbelegung entspricht dabei der eines PCs (DTE-Gerät):

Serielle RS-232-Schnittstelle P400	
Pin	**Funktion**
1	N.C.
2	RxD (Received Data)
3	TxD (Transmitted Data)
4	DTR (Data Terminal Ready)
5	GND (Signal Ground)
6	DSR (Data Set Ready)
7	RTS (Request To Send)
8	CTS (Clear To Send)
9	N.C.

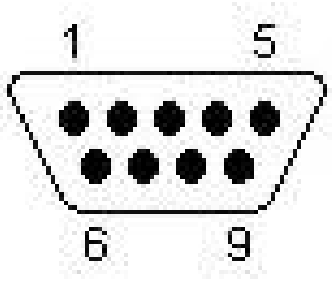


Tabelle 2: Beschaltung der seriellen Schnittstelle

Hinweis:
Die Beschaltung der Schnittstelle entspricht der eines PCs. Das bedeutet, daß Datengeräte wie z.B. ein Modem direkt angeschlossen werden können (über ein normales serielles Kabel).
Geräte wie PCs, die selbst wie DTE-Geräte beschaltet sind, müssen über ein Null-Modemkabel (Best.Nr. WK041) verbunden werden. Ein Nullmodem-Kabel kreuzt u.a. die Leitungen RxD und TxD.
Die Leitungen DSR und DTR sind auf dem Modul miteinander verbunden. Falls dies nicht erwünscht ist, muß der Jumper J400 geöffnet werden.

Erläuterungen zu den übrigen Steuerleitungen

Primär werden nur die Leitungen TxD, RxD und GND für eine serielle Datenübertragung benötigt. Diese „Drei-Draht-Verbindung“ ist für die meisten Anwendungsfälle ausreichend, wenn die Verbindung zusätzlich durch Steuerzeichen synchronisiert wird.
In Anwendungsfällen, bei denen keine Steuerzeichen gesendet werden können (z.B. bei einer binären Modem-Verbindung), erfolgt die Synchronisation über die Handshake-Leitungen der seriellen Schnittstelle.

Hier gibt es zwei Leitungspaare:

DSR/DTR: Diese Leitungen stellen die grundsätzliche Sende- und Empfangsbereitschaft zwischen Sende- und Empfangsgerät sicher. Beim grabbMODUL-4 sind DSR- und DTR-Leitung intern mit einem Jumper verbunden. Auf diese Weise wird immer Sende-/Empfangsbereitschaft signalisiert, was in den allermeisten Anwendungen ausreichend ist.
Falls die Verbindung DSR/DTR nicht gewünscht ist, kann J400 aufgetrennt werden.

RTS/CTS: Mit diesen Leitungen wird während einer Übertragung Sende- bzw. Empfangsbereitschaft signalisiert. Dem angeschlossenen Gerät (z.B. einem Modem) wird vom Modul durch Aktivierung der RTS-Leitung der Beginn einer Datenübertragung angezeigt. Das Gerät zeigt seinerseits die Bereitschaft zur Datenübertragung durch Aktivierung des CTS-Signals an. Während der Übertragung können die Signale von beiden Geräten deaktiviert werden, was zu einer temporären Unterbrechung der Übertragung führt (z.B. wenn der Empfangspuffer voll ist*). Einzelheiten zu diesem Protokoll finden Sie in Beschreibungen zur RS-232-Schnittstelle.*

Hinweis:
Anders als bei UART-Bausteinen in PCs findet beim C165-Controller keine automatische Steuerung der Handshake-Signale statt, da der Controller nicht über entsprechende FIFO-Puffer verfügt. Bei Bedarf muß der Anwender solche Puffer und die zugehörige Schnittstellensteuerung softwaremäßig implementieren (z.B. als Interrupt-Routine).
Eine entsprechende Funktion finden Sie auf der Treiber-CD.

Die Signalleitungen RTS/CTS sind an folgenden Controller-Portpins verfügbar:

Signal	Controller-Port	Direction	Description
RTS	P3.13	OUT	RS-232: Ready To Send
CTS	P3.15	IN	RS-232: Clear To Send

Tabelle 3: Signalzuordnung der RS232-Handshake-Leitungen

3.1.3 Video-Eingänge

An das grabbMODL-4 können zwei Typen von Videoquellen angeschlossen werden:

(a) Composite-Videoquellen

Bei Composite-Signalen werden alle Bildsignale auf einer Leitung geführt. Typische Steckverbindungen sind BNC- oder Cinch-Stecker. An das grabbMODUL-4 können bis zu drei Composite-Videoquellen angeschlossen werden. Dazu dienen die BNC-Buchsen CH1 bis CH3.
Anschlußbelegung:

BNC-Videoeingänge	
Pin	**Funktion**
Shield	GND (Signal Ground)
Pin	Video Input (Composite)

Tabelle 4: Beschaltung der BNC-Videoeingänge

An diese Eingänge können sowohl Farb- als auch Schwarzweiß-Kameras angeschlossen werden.

(b) S-Video-Quellen

Der Vorteil des S-Video-Anschlusses ist die getrennte Führung von Helligkeits- und Farbsignal. Dies verhindert Moiréstörungen an feinen Bildstrukturen und verbessert die tatsächliche Auflösung bei Farbbildern.

Eine S-Video-Quelle können Sie an der 4-poligen Mini-DIN-Buchse CH4 anschließen.

Anschlußbelegung:

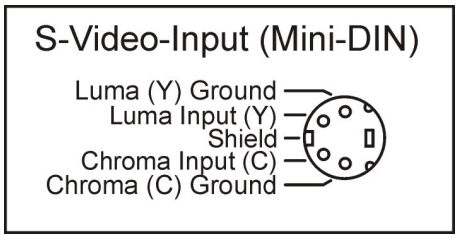
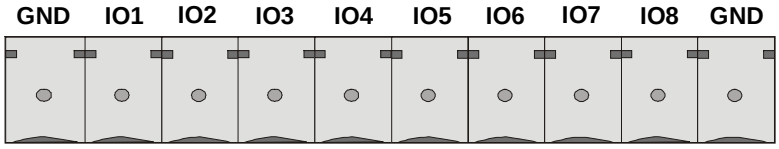


Bild 25: Belegung der S-Video-Buchse

Hinweis:
Die Umschaltung zwischen den Eingangskanälen erfolgt per Software. Beachten Sie, daß der richtige Modus (Composite / S-Video) eingestellt werden muß.

3.1.4 Optoentkoppelte I/O-Ports

Über die optoentkoppelten I/O-Ports können Steuersignale mit Peripheriegeräten ausgetauscht werden. Es stehen vier Eingänge und vier Ausgänge zur Verfügung:



optoentkoppelte I/O (X400)	
Pin	**Funktion**
GND	I/O-Ground (– / Emitter)
IO1	INPUT 0 (+)
IO2	INPUT 1 (+)
IO3	INPUT 2 (+)
IO4	INPUT 3 (+)
IO5	OUTPUT 0 (Collector)
IO6	OUTPUT 1 (Collector)
IO7	OUTPUT 2 (Collector)
IO8	OUTPUT 3 (Collector)
GND	I/O-Ground (– / Emitter)

Bild 26: Anschlußbelegung I/O-Port

Hinweis:
Die Ground-Leitungen aller I/O-Kanäle sind zusammengeschaltet.
Die I/O-Ground-Leitungen sind von der Masse des grabbMODUL-4 galvanisch getrennt.

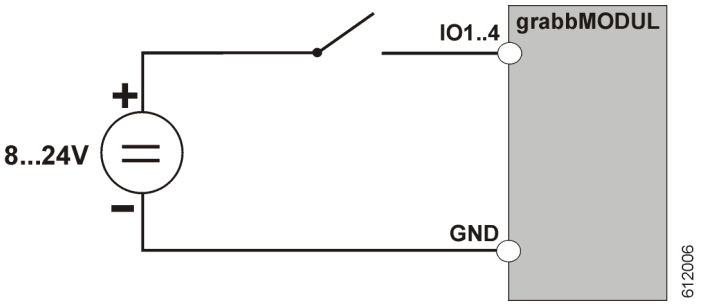


Bild 27: Typische Eingangsbeschaltung des I/O-Ports

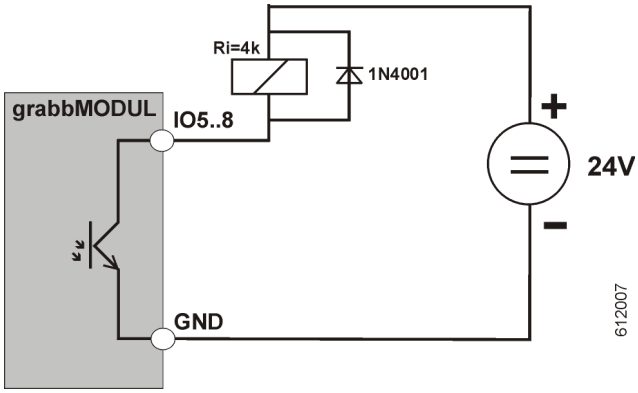


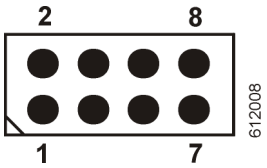
Bild 28: Typische Ausgangsbeschaltung des I/O-Ports

Bild 27 und *Bild 28* zeigen typische Beschaltungen des I/O-Ports.

Achtung!
Achten Sie bei der Beschaltung der Ports auf die richtige Polarität und die Einhaltung der maximal zulässigen Strom - und Spannungswerte (Abschnitt *3*), da das Modul ansonsten beschädigt werden könnte.

3.1.5 Option Port

Der Option-Port OPT-PORT, X401 ist eine 8polige Stiftleiste, an der einige zusätzliche Steuersignale des Controllers zur Verfügung stehen. Dieser Port ist gut dazu geeignet, um eigene Peripherie an das grabbMODUL-4 anzuschließen.



Option-Port (X401)	
Pin	**Funktion**
1	Vcc (+5 V)
2	GND (Signal Ground)
3	SCL (I²C-Bus Clock Line)
4	SDA (I²C-Bus Data Line)
5	P6.4 / CS4 (Controller Port 6.4)
6	P6.5 (Controller Port 6.5)
7	P6.6 (Controller Port 6.6)
8	P6.7 (Controller Port 6.7)

Tabelle 5: Signale des Option-Port-Steckers X401

Bei den Leitungen SCL und SDA handelt es sich um die Clock- und Datenleitung der integrierten I²C-Schnittstelle.
Über die I²C-Schnittstelle können externe Geräte abgefragt oder angesteuert werden, die eine I²C-Schnittstelle im Slave-Modus besitzen. Es können mehrere I²C-Geräte an den Bus angeschlossen werden, sie müssen sich jedoch in ihrer Geräteadresse unterscheiden.

Die I²C-Schnittstelle des grabbMODUL-4 ist ein softwaremäßig implementierter I²C-Busmaster. Sie können also Slave-Geräte an diesem Bus betreiben, der Bus wird vom Master verwaltet.
Zur Ansteuerung der Geräte verwenden Sie die I²C-Routinen der mitgelieferten Treiber-Bibliothek.

Hinweis:
Die I²C-Schnittstelle wird mit TTL-Pegel betrieben. Dadurch ist die maximale Leitungslänge begrenzt. Es sollten keine Leitungen länger als 30 cm verwendet werden.

Achtung!
An der I²C-Schnittstelle sind modul-intern bereits I²C-Geräte angeschlossen (EEPROM und – optional – RTC). Beachten Sie beim Anschluß externer I²C-Geräte, daß kein Adreßkonflikt mit den internen I²C-Geräten entsteht. In diesem Fall werden die Geräte nicht richtig funktionieren.
Gegebenenfalls ist es möglich, daß die Firmware des Moduls nicht richtig startet.

Folgende I²C-Adressbereiche dürfen extern nicht belegt werden:

intern belegte I²C-Adressbereiche		
Gerät	**Option**	**Bereich**
EEPROM (40C08, **default**)	-	$A8_{HEX}$...AF_{HEX}
EEPROM (40C04)	J4 = 1+2	AC_{HEX}...AF_{HEX}
	J4 = 2+3	$A8_{HEX}$...AB_{HEX}
RTC	optional bestückt	$A2_{HEX}$...$A3_{HEX}$
Videoprozessor	-	80_{HEX}...$8C_{HEX}$

Hinweis*:*
Weiterführende Informationen über die I²C-Schnittstelle finden Sie im Internet z.B. auf der Homepage von Philips Semiconductor.

Die Controllerports P6.4...P6.7 stehen dem Anwender frei zur Verfügung. Sie können hier Bausteine anschließen, die TTL-kompatibel sind (nähere Spezifikationen entnehmen Sie bitte dem Abschnitt *3, „Technische Daten"*). Die Controllerports können Sie in Ihrem Programm direkt ansprechen, Sie benötigen keinen speziellen Software-Treiber.
Beachten Sie, daß die Controller-Pins per default als Eingänge definiert sind. Für die Verwendung als Ausgang muß der betreffende Pin entsprechend konfiguriert werden.

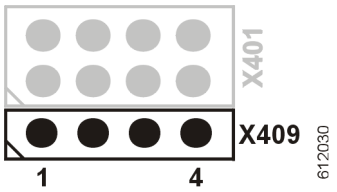
Hinweis:
Controller-Portpins können nicht größere Ströme treiben. Insbesondere Ströme, die aus dem Pin herausfließen (wenn der Pin auf Vcc geschaltet ist), können keine größere Last treiben. Hier bricht die am Pin anliegende Spannung schnell zusammen. *Beachten Sie bitte diesbezüglich die Spezifikationen der Portpins.*

Hinweis:
Die Signale P6.4..P6.7 sind auch an der µC-Schnittstelle X402 herausgeführt. Achten Sie darauf, daß die Leitungen nicht versehentlich mehrfach beschaltet werden. Die Leitung P6.4 kann alternativ auch als Chip-Select Signal /CS4 verwendet werden. Achten Sie darauf, daß es bei Beschaltung von P6.4 nicht zu Konflikten mit einer eventuellen Verwendung der Leitung als Chip-Select Signal kommt.

3.1.6 Extended Option Port (Optional)

Hinweis: Dieser Anschluß ist nur bei besonderen Ausführungen des grabbMODUL-4 verfügbar. In der Standard-Variante ist dieser Anschluß nicht vorhanden.

Bei Varianten des grabbMODUL-4 sind vier TTL-Signale für eine serielle Kommunikation an der Stiftleiste X409 vorhanden.



Extern-Options-Port (X409)	
Pin	**Funktion**
1	RTS_TTL
2	TXD0_TTL
3	RXD0_TTL
4	CTS_TTL

Tabelle 7: Belegung des Extern-Options-Ports (X409)

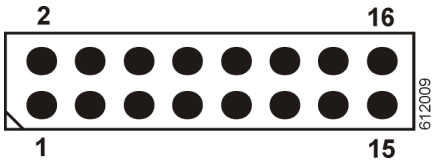
Die Leitungen sind direkt mit entsprechenden Controllerpins verbunden. Die Verwendung als serielle Schnittstelle mit TTL Pegel erfolgt per Software (siehe Treiberbeschreibung).

Wichtig: Alle vier TTL-Signale sind in der Standardausführung mit dem RS232 Treiber auf der Platine verbunden. Eine eigene Verwendung der Signale ist nur möglich wenn dieser Baustein deaktiviert ist.

3.1.7 Extended Video Port (Optional)

Hinweis:
Dieser Anschluß ist nur bei besonderen Ausführungen des grabbMODUL-4 verfügbar. In der Standard-Variante ist dieser Anschluß nicht vorhanden.

Bei Varianten des grabbMODUL-4 sind vier weitere Composite-Videoeingänge an dieser Stiftleiste EXT-PORT X204 anschließbar.



Extern-Video-Port (X204)	
Pin	**Funktion**
1	Vcc (+5 V)
2	PE GND (Shield Ground)
3	V_{OUT} (Kameraversorgung, optional)
4	GND (Signal Ground)
5	GND (Signal Ground)
6	Composite Video Input 5
7	GND (Signal Ground)
8	Composite Video Input 6
9	GND (Signal Ground)
10	Composite Video Input 7
11	GND (Signal Ground)
12	Composite Video Input 8
13	V_{OUT} (Kameraversorgung, optional)
14	GND (Signal Ground)
15	Vcc (+5 V)
16	PE GND (Shield Ground)

Tabelle 6: Belegung des Extern-Video-Ports (X204)

Die Umschaltung der Kanäle erfolgt per Software (*siehe Treiberbeschreibung).*

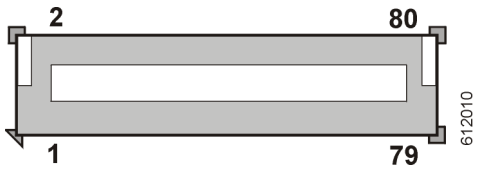
Achtung!
Zum Anschluß der zusätzlichen Eingänge ist eine Zusatzplatine erforderlich (z.B. VM-004-EXT-BNC). Die Videoquellen können **nicht** direkt an der Stiftleiste eingespeist werden.

Der Schirm (PE-GND) ist intern kapazitiv an die Betriebsmasse angekoppelt und zusätzlich galvanisch mit den Befestigungslöchern X403 und X406 verbunden. Hier kann bei Bedarf eine leitende Verbindung zum Gehäuse o.ä. hergestellt werden.

3.1.8 Microcontroller-Port (optional)

Hinweis:
Dieser Anschluß ist nur bei besonderen Ausführungen des grabbMODUL-4 verfügbar. In der Standard-Variante ist dieser Anschluß nicht vorhanden.

Der Microcontrollerport ist eine Erweiterungsschnittstelle an der alle wichtigen Signale des Microcontrollers vorhanden sind. Sie kann zum Anschluß von komplexer Hardware verwendet werden.
Die Schnittstelle ist als Molex-Stecker (2 x 40pol, Rastermaß 0,653) ausgeführt. Passende Verbinder können Sie unter der PHYTEC-Bestellnummer VB091 (Modulabstand ca. 5 mm) oder VB084 (Modulabstand ca. 10 mm) bestellen.



Microcontroller Port (X402)			
Pin	**Dir.**	**Name**	**Funktion**
1	PWR	Vcc	+5 V Supply Output
2		Vcc	
3	-	GND	Signal Ground
4	I/O	/RESET	/Reset (Initiate)
5	I/O	IN7/P2.9	Microcontroller Port P2.9
6		IN6/P2.8	Microcontroller Port P2.8
7	IN	/NMI	Not Maskable Interrupt Input
8	-	GND	
9	I/O	P6.4/CS4	uC-Port P6.4 = /CS4
10	OUT	/CS3	Chip Select 3 (peripheral CS)
11	OUT	ALE	Address Latch Enable *)
12	OUT	/RESOUT	/Reset (Output)
13	-	GND	
14	OUT	/RD	Memory Read
15	OUT	/WRL	Memory Write / Write Low Byte
16	OUT	A0	Microcontroller Address A0 *)
17		A1	Microcontroller Address Bus
18	-	GND	
19	OUT	A2	Microcontroller Address Bus
20		A3	
21		A4	
22		A5	
23	-	GND	
24	OUT	A6	Microcontroller Address Bus
25		A7	
26		A8	
27		A9	
28	-	GND	
29	OUT	A10	Microcontroller Address Bus
30		A11	
31		A12	
32		A13	
33	-	GND	
34	OUT	A14	Microcontroller Address Bus
35		A15	
36	I/O	D0	Microcontroller Data Bus
37		D1	
38	-	GND	
39	I/O	D2	Microcontroller Data Bus
40		D3	
41		D4	
42		D5	

43	-	GND	
44	I/O	D6	Microcontroller Data Bus
45		D7	
46	OUT	A16	Microcontroller Address Bus
47		A17	
48	-	GND	
49	OUT	A18	Microcontroller Address Bus
50		A19	
51		A20	
52		A21	
53	-	GND	
54	OUT	A22	Microcontroller Address Bus
55		A23	
56		D8	Microcontroller Data Bus
57		D9	
58	-	GND	
59	I/O	D10	Microcontroller Data Bus
60		D11	
61		D12	
62		D13	
63	-	GND	
64	I/O	D14	Microcontroller Data Bus
65		D15	
66	OUT	/WRH	Memory Write (High Byte)
67	IN	/RDY	Microcontroller /Memory Ready
68	-	GND	
69	IN	BOOT	Activate Bootstrap Loader [2])
70	I/O	P6.5	Microcontroller Port 6.5
71		P6.6	Microcontroller Port 6.6
72		P6.7	Microcontroller Port 6.7
73	-	GND	
74	OUT	/ACTIVE	Framegrabber Active Status
75	I/O	SDA	Serial I²C Bus: Data Line
76	OUT	SCL	Serial I²C Bus: Clock Line
77	I/O	CTRL1	reserved (P2.14)
78	-	GND	
79	PWR	Vcc	+5V Supply Output
80		Vcc	
*) used in multiplexed bus mode only [2]) must not be connected to GND, might only be tied to VCC			

Tabelle 7: Belegung des Microcontroller-Erweiterungssteckers X402

3.1.9 RAM-Speichersicherung

Hinweis:
Diese Option ist nur bei besonderen Ausführungen des grabbMODUL-4 verfügbar. In der Standard-Variante ist diese Funktion nicht vorhanden.

Auf dem grabbMODUL-4 kann die Möglichkeit, das modulinterne SRAM (Arbeitsspeicher und Video-Bildspeicher) bei Ausfall der Versorgungsspannung mit einer Batterie zu puffern, vorgesehen werden.
Dazu benötigen Sie ein Modul, bei dem der Batterie-Controller U9 bestückt ist.

Hinweis:
Die Möglichkeit zur Speicherpufferung ist abhängig von der Art und der Größe des bestückten RAM-Speichers. Große und schnelle RAMs benötigen auch im Standby-Moduls so viel Strom, daß eine Batterie schnell entleert wäre.
Falls Sie in Ihrer Applikation ein batteriegepuffertes RAM benötigen, erfragen Sie bitte Modul-Varianten mit entsprechend angepaßtem Speicherausbau.
Bis zu 512 Byte (optional: 1 kByte) Anwendungsdaten können Sie auch nichtflüchtig im EEPROM des Moduls speichern.
Die Speichersicherungsbatterie wird auch benötigt, wenn das Modul mit einer Realtime-Clock (RTC) ausgestattet ist.

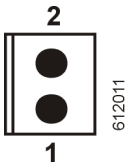
Zur Pufferung des RAM-Speichers gibt es zwei Möglichkeiten:

(a) Anschluß einer Lithium-Batterie

Auf dem Platz BAT1 (Kreissymbol auf der Moduloberseite) kann eine Lithium-Batterie vom Typ CR2032 (PHYTEC-Bestellnr. BL003) bestückt werden.
Diese Batterie wird auch benötigt, wenn die Real-Time-Clock (RTC) bestückt ist.
Die Lebensdauer der Batterie ist abhängig von der Kapazität und dem auf dem Modul vorhandenen Speicherausbau.
Der Ladezustand der Batterie kann mit Hilfe der Batteriespannungsüberwachung überprüft werden.

(b) Anschluß eines externen Pufferkondensators

Eine Netzausfallüberbrückung ist auch mit einem externen Pufferkondensator für eine begrenzte Zeit möglich. Dazu wird am besten ein hochkapazitiver Kondensator (Gold Cap) verwendet.
Dieser Pufferkondensator kann an dem Steckverbinder X1 angeschlossen werden:



Pufferkondensator (X1)	
Pin	**Funktion**
1	VPD (Kondensator +)
2	GND

Tabelle 8: Anschlußbelegung Pufferkondensator

Achtung!
Da der Kondensator über diesen Steckverbinder sowohl geladen als auch entladen wird, ist es wichtig, eine entsprechende Ladeschaltung vorzusehen.
Bei einem direkten Anschluß würde beim Einschalten der Versorgungsspannung ein hoher Strom in den Kondensator fließen, der zu Schäden am Modul führen könnte. Sehen Sie daher eine geeignete Ladeschaltung vor (*Bild 29* zeigt eine einfache Schaltung).

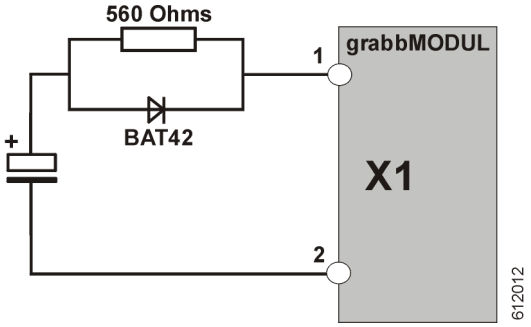


Bild 29: Einfache Außenbeschaltung für Pufferkondensator

Hinweis:
X1 kann auch zum Anschluß einer wiederaufladbaren Batterie verwendet werden. Hierbei ist eine geeignete Ladeschaltung zu verwenden.

Die RTC wird immer aus Batterie und / oder Pufferkondensator versorgt. Zur Aktivierung der RAM-Speicherpufferung muß das Modul folgendermaßen konfiguriert werden:

Pos.	Beschreibung	Einstellung
J1	Battery Backup Enable 1+2 = Fast RAM, Battery Backup Disabled 2+3 = Normal RAM, Battery Backup Enabled	2+3
R20	Battery Backup Jumper 0 R = Fast RAM, Battery Backup Disabled open = Normal RAM, Battery Backup Enabled	open

Tabelle 9: Konfigurierung Battery-Backup

(Die Einstellungen werden sowohl für Batterie- als auch für Kondensatorpufferung benötigt).

4 Blockdiagramm

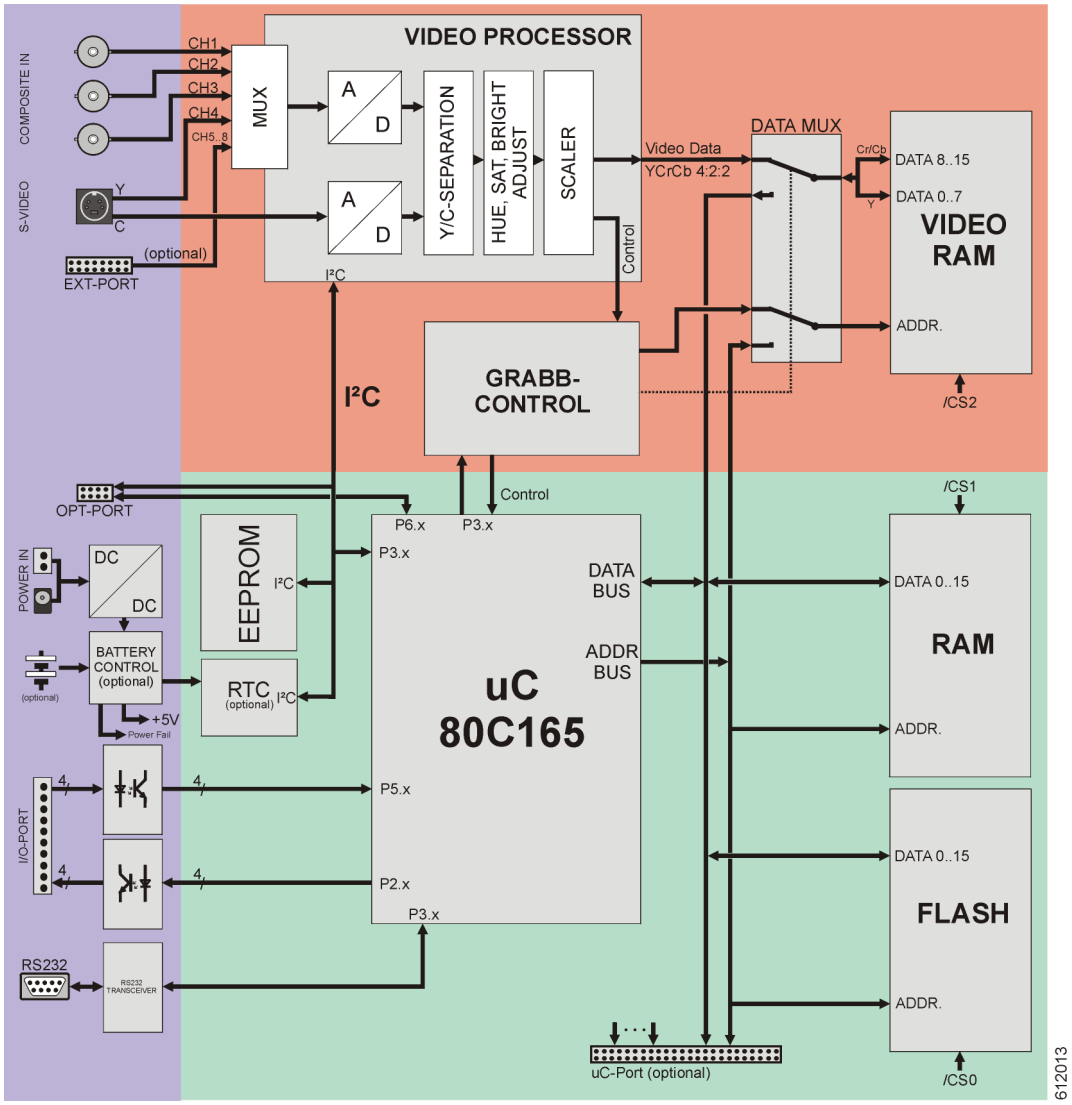


Bild 30: Blockdiagramm VM-004

Rechenkern

Der Rechenkern des grabbMODUL-4 besteht aus einem C165-Microcontroller mit FLASH-Programmspeicher und SRAM-Datenspeicher. Der Speicher arbeitet im 16-Bit non-multiplexed-Mode. Zusätzlich besitzt das grabbMODUL-4 ein EEPROM, in dem der Anwender nichtflüchtige Daten speichern kann.

Optional kann das grabbMODUL-4 mit einer Echtzeituhr bestückt werden, das wie das EEPROM über die I²C-Schnittstelle angesprochen werden kann. Die wesentlichen Signale des Controllers sind an der optional bestückten Stiftleiste „uC-Port“ für Erweiterungsschaltungen herausgeführt.

Framegrabber

Die Anbindung des autark arbeitenden Framegrabber-Teils erfolgt durch das Video-RAM. Über einen Daten-Multiplexer wird dieses RAM entweder mit dem Video-Prozessor oder mit dem Microcontroller verbunden. Die Umschaltung erfolgt automatisch durch die Framegrabber-Steuerung.

Während eines Grabb-Vorgangs (Bildaufnahme) wird das Video-RAM aus dem Controller-Speicherbereich ausgeblendet, so daß der Video-Prozessor die Bilddaten in diesen Speicher schreiben kann. Danach wird das Video-RAM automatisch wieder in den Controller-Speicherbereich eingeblendet.

Die Bildaufnahme wird ohne Belastung von Controller-Ressourcen durch die im Framegrabber-Teil integrierte Ablaufsteuerung (*Grabb-Control*) durchgeführt. Der Controller startet den Vorgang durch ein Steuersignal. Die Aufnahme eines Vollbilds dauert dann längstens 80 ms. In dieser Zeit kann der Controller normal weiterarbeiten, lediglich der Bildspeicher (*Video-RAM*) ist – wie erwähnt – nicht verfügbar.
Über ein Statussignal teilt die Ablaufsteuerung dem Controller das Ende des Aufzeichnungsvorgangs mit. Ab diesem Zeitpunkt kann der Controller wieder auf das Video-RAM zugreifen und die Bilddaten lesen (Schreibzugriffe auf den Bilddatenspeicher sind controllerseitig nicht möglich).

Bis zu drei Composite- und eine S-Videoquelle können an das grabbMODUL-4 angeschlossen werden.
Optional kann das grabbMODUL-4 auch mit einem 8-kanaligen Multiplexer bestückt werden. Es stehen dann acht (Composite-)Video-Eingänge zur Verfügung.

Die Video-Eingänge werden an einen Multiplexer geführt. Per Software kann der gewünschte Eingangskanal ausgewählt werden. Die Umschaltung zwischen zwei Kanälen dauert dabei weniger als 300 ms.

Der Videoprozessor führt die Digitalisierung des Bildsignals durch. Bei S-Video-Signalen erfolgt die Digitalisierung des Farbsignals durch einen eigenen, zweiten A/D-Wandler, wodurch eine besonders gute Farbwiedergabe und Bildqualität erreicht wird.

Die digitalen Bilddaten werden anschließend gefiltert und korrigiert. Hier kann der Anwender Parameter wie Helligkeit, Kontrast oder Farbsättigung beeinflussen. Der *Scaler* legt die Bildauflösung und den gewünschten Bildausschnitt fest. So kann ohne Controllerbelastung genau die Datenmenge aus den Bilddaten herausgegriffen werden, die für die Anwendung benötigt werden.
Die Speicherung der Bilddaten im Video-RAM erfolgt immer im YCrCb-Format. Dieses Format benötigt für volle Farbwiedergabe (16 Mio. Farben) nur 2 Byte Speicherplatz pro Pixel. Dieses Format ist einerseits für Farbbildübertragung günstig, da es Übertragungsbandbreite spart. Zum Anderen bietet es Vorteile bei der Bildanalyse, da Helligkeitsanteil (Y) und Farbanteil (CrCb) anders als im RGB-Format schon getrennt vorliegen.

Peripherie

Auf dem grabbMODUL-4 sind häufig benötigte Peripherie-Komponenten integriert. Das Schaltnetzteil ermöglicht eine Versorgung des Moduls aus einer ungeregelten Gleichspannung mit einem weiten Eingangsspannungsbereich von 8...28 V. Dies deckt typische Versorgungsspannungen auch in Industrie- und Kfz-Anwendungen ab. Optional besteht die Möglichkeit, das RAM aus einer Backup-Batterie zu puffern.
Jeweils 4 Ein- und Ausgangsleitungen stehen optoentkoppelt für beliebige Schaltsignale zur Verfügung.
Zur Kommunikation mit einem Host-Rechner, einem Modem oder einem andern Peripheriegerät kann die voll ausgebaute RS-232-Schnittstelle (mit Hardware-Handshake) verwendet werden. Weitere Zusatzgeräte lassen sich z.B. über den Erweiterungsport mit I²C-Schnittstelle und Controller-Portleitungen anschließen.

5 Jumper und Ressorcenbelegung

5.1 Jumperplan

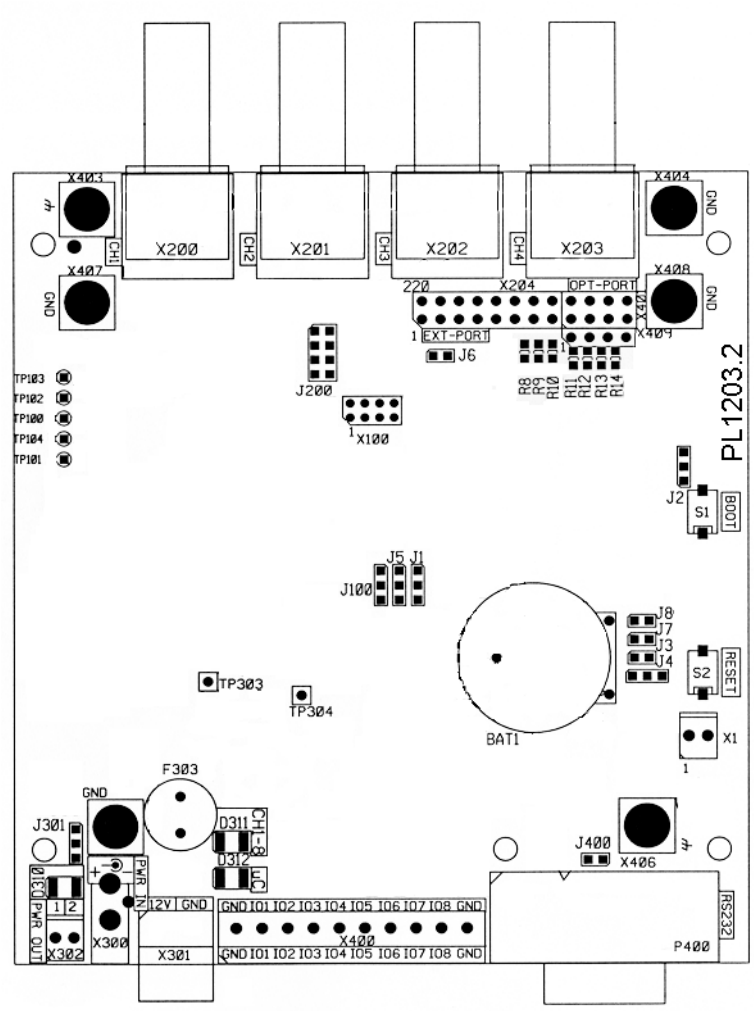


Bild 31: Lage der Jumper (Oberseite)

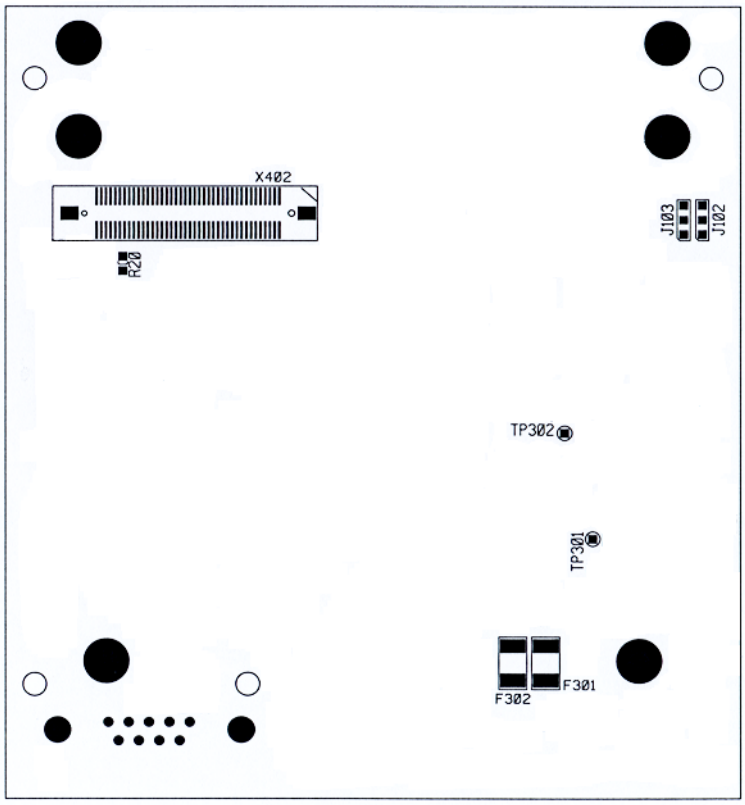


Bild 32: Lage der Jumper (Unterseite)

5.2 Jumper und Optionen

Sicherungen

Pos.	Beschreibung	Wert
F301	nicht bestückt	-
F302	Modulversorgung	1A T
F303	nicht bestückt	-

Netzteil Shutdown

Pos.	Beschreibung	Standard
TP304	open: Modul in Betrieb GND: Shutdown (Netzteil aus)	open

Über den Anschluß TP304 kann das Modul ausgeschaltet werden. Beachten Sie, daß der Pin nicht gegen Vcc oder eine andere Betriebsspannung geschaltet werden darf (*siehe auch Abschnitt 3.1.1*).

RAM-Größe

Pos.	Beschreibung	Default
J5	1+2 = 1 MByte Arbeitsspeicher (2 x 512 kB SRAM) 2+3 = 256 kByte Arbeitsspeicher (2 x 128 kB SRAM)	2+3

Das grabbMODUL-4 kann werksseitig mit unterschiedlich großem Arbeitsspeicher ausgestattet sein. J5 wird werksseitig entsprechend eingestellt. Verändern Sie die Position von J5 nicht.

RAM-Batterie-Backup

Pos.	Beschreibung	Default
J1	Battery Backup Enable 1+2 = Fast RAM, Battery Backup Disabled 2+3 = Normal RAM, Battery Backup Enabled	1+2
R20	Battery Backup Jumper 0 R = Fast RAM, Battery Backup Disabled open = Normal RAM, Battery Backup Enabled	0R

Bei Bedarf kann das grabbMODUL-4 mit batteriegepuffertem RAM ausgestattet werden. Dazu wird eine spezielle Modul-Version benötigt, die für Batteriepufferung vorgesehen ist (*siehe dazu auch Abschnitt 3.1.9).*

EEPROM-Busadresse

Pos.	Beschreibung	Default
J4	40C08: keine Bedeutung (default Bestückung) 40C04: 1+2 = I²C-Busadresse EEPROM = AC_{HEX} 2+3 = I²C-Busadresse EEPROM = $A8_{HEX}$	2+3

Der Jumper legt die Basisadresse des seriellen EEPROMs auf dem I²C-Bus fest.

EEPROM Write-Protect

Pos.	Beschreibung	Default
J3	**24C01 / 24C02** open (no options)	
	24C04 / 24C08 open = kein Schreibschutz close = Schreibschutz	open

Durch Schließen von J3 kann der Inhalt des EEPROMs schreibgeschützt werden. Mit geschlossenem Jumper sind keine Schreibzugriffe auf das EEPROM mehr möglich.
Diese Option besteht nicht bei allen installierbaren EEPROMs. Beim standardmäßig installierten EEPROM kann der Schreibschutz aktiviert werden.

RTC Interrupt

Pos.	Beschreibung	Default
J7	open = keine Interrupt-Möglichkeit der RTC close = RTC Interrupts hardwaremäßig möglich	open

Wenn die RTC auf dem grabbMODUL-4 installiert ist, besteht die Möglichkeit, den Interrupt-Ausgang der RTC auf den Controller-Portpin P3.2/CAPIN zu führen.

Power-Fail – Interrupt

Pos.	Beschreibung	Default
J8	open = kein Power-Fail-Interrupt close = Power-Fail-Interrupt möglich	open

Pos.	Beschreibung	Default
R30 R31 R32	Spannungsüberwachung Sicherungsbatterie: R30 = R31 = 10 MΩ R32 = offen	R32=offen R30=10M R31=10M
	Betriebsspannungsüberwachung: R30 = 100k R31 = offen R32 = $\left(\frac{U\,\min}{1{,}25V}-1\right)\cdot 100k\Omega$	

Bei installiertem Batterie-Controller (Option) kann ein Unterschreiten der Betriebsspannung oder ein Abfall der Spannung der Sicherungsbatterie unter einen bestimmten Wert einen nicht maskierbaren Interrupt (NMI) auslösen.
Wird die Spannung der Sicherungsbatterie überwacht, so löst ein Unterschreiten der Spannungsschwelle von 2,5V einen NMI aus.
Soll die Betriebsspannung überwacht werden, kann der Anwender durch Wahl des Widerstandswerts von R32 die Schaltschwelle selbst festlegen (*siehe Formel*).
Legt man die Schaltschwelle U_{min} ca. 15% unter die Nennbetriebs-spannung, so ergeben sich für typische Betriebsspannungen folgende Werte für R32:

U_B	U_B-15%	R32	U_{MIN}
10V	8,5 V	560k	8,25 V
12V	10,2 V	750k	10,6 V
15V	12,75 V	910k	12,63 V
20V	17 V	1,2M	16,25 V
24V	20,4 V	1,5M	20,0 V

(die abweichenden Spannungswerte bei U_{min} ergeben sich durch Rundung auf Normreihenwerte.)
Ein Widerstandswert von 560 kΩ sollte nicht unterschritten werden.

Controller-Konfiguration

Die Konfiguration des Microcontrollers erfolgt über Konfigurationswiderstände, die an bestimmten Datenleitungen angeschlossen sind. Die meisten Einstellungen stehen in direktem Zusammenhang mit der Hardware-Architektur und dürfen daher nicht vom Anwender verändert werden!

Pos.	Beschreibung				Default
R8 R9	CHIPSEL-Konfiguration				4k7,4k7
	R8	R9	externe Chip-Selects		
	open	open	/CS3,/CS4 aktiv		
	4k7	4k7	/CS3,/CS4 disabled (default)		
R10 R11	Konfiguration Segment-Adreßleitungen				4k7,4k7
	R11	R10	Adreßraum		
	open	open	256 kByte		
	4k7	4k7	1 MB (default)		
	open	4k7	16MB		
R12 R13 R14	Clock Mode				4k7,4k7,4k7
	R14	R13	R12	CPU-Clock (C165)	
	open	don't care	don't care	f_{OSZ} : 2	
	4k7	don't care	don't care	f_{OSZ} (default)	

RS-232 Handshake

Pos.	Beschreibung	Default
J400	open = DTR / DSR nicht verbunden close = DTR / DSR verbunden (Modem-Betrieb)	open

Bei binärer Datenübertragung über die serielle Schnittstelle ist es erforderlich, die Hardware-Handshake-Signale zu benutzen. Dies ist z.B. dann der Fall, wenn Daten über ein Modem übertragen werden.
Die RS-232-Schnittstelle sieht zwei Paare von Handshake-Leitungen vor. Das Paar DTR/DSR signalisiert dabei die grundsätzliche Sendebereitschaft der Geräte. Das grabbMODUL-4 kann diese Signalpaare nicht steuern bzw. auswerten. Daher können diese Signale mit J400 gebrückt werden, wodurch die prinzipielle Sendebereitschaft immer signalisiert wird. Die Kommunikationssteuerung erfolgt über das Signalpaar RTS/CTS, das an P3.13/P3.15 zur Verfügung steht (*siehe auch Abschnitt 3.1.2*).

5.3 Ressourcen

Dieser Abschnitt gibt einen Überblick über die Belegung und Zuordnung der Controller-Ressourcen.

Hinweis:
Signale, bei denen ein Eintrag in der Spalte „Belegung“ vorgenommen ist, sind auf dem Modul fest zugeordnet. Diese Signale sind eventuell auch extern verfügbar. Die Funktion dieser Signale darf jedoch nicht verändert werden (z.B. durch Umprogrammierung der Portfunktion), da dadurch die Funktion des Moduls verändert werden könnte. Achten Sie auf die Datenflußrichtung (IN/OUT) dieser Signale, falls Sie externe Beschaltungen vornehmen möchten.

Signale ohne Eintrag in der Spalte „Belegung“ können vom Anwender frei verwendet werden. Die Spalte „verfügbar“ gibt an, an welchem Steckverbinder das Signal herausgeführt ist (Stecker-Pin).

Signale, die mit „reservierte Steuerleitung“ bezeichnet sind, dürfen vom Anwender nicht verwendet werden.

Änderungen an den vordefinierten Portfunktionen können eine Zerstörung des Moduls zur Folge haben!

Chip-Select-Signale

Signal	Belegung	verfügbar	Beschreibung
/CS0	FLASH	-	Chip-Select Flash Speicher
/CS1	RAM	-	Chip-Select RAM-Speicher
/CS2	VIDEO-RAM	-	Chip-Select Video-RAM-Speicher
/CS3	-	X402-10	Chip-Select 3
/CS4	-	X402-9	Chip-Select 4 (shared with P6.4)

Controllerports

Signal	Belegung	verfügbar	Beschreibung
P0.x	DATA	-	Data Bus
P1.x	ADDRESS	-	Address Bus
P4.0 - P4.3	ADDRESS	-	Address Bus (Bank Select)
P4.4 P4.5 P4.6 P4.7	ADDRESS / PORT	X402-51 X402-52 X402-54 X402-55	Address Bus (Bank Select) oder freier Port Pin (abhängig von der Speicherkonfiguration)
P6.0 P6.1 P6.2	/CS0 /CS1 /CS2	-	interne Chip Select-Signale
P6.3 P6.4	-	X402-10 X402-09 ²	freier /CS3 oder freier Port freier /CS4 oder freier Port
P6.5 P6.6 P6.7	-	X402-70 ² X402-71 ² X402-72 ²	freier Port
P5.x	OPTICAL IN	-	optoentkoppelte Eingänge (X400)
P2.0 P2.1	-	X402-05 X402-06	freier Port
P2.2 - P2.13	OPTICAL OUT	-	optoentkoppelte Ausgänge (X400)
P2.14	CTRL1	X402-77	reservierte Steuerleitung
P2.15	-	-	N.C.
P3.0	/ACTIVE	X402-74	Framegrabber aktiv-Signal (Output)
P3.1	MUX_A0	-	reservierte Steuerleitung
P3.2	RTC-IRQ*	-	Interruptleitung Realtime-Clock
P3.3	SDA	X402-75 ²	I²C-Schnittstelle Datenleitung
P3.4	SCL	X402-76 ²	I²C-Schnittstelle Clockleitung
P3.5	/START	-	reservierte Steuerleitung
P3.6	INIT	-	reservierte Steuerleitung
P3.7	FRAMECAP	-	reservierte Steuerleitung
P3.8	INSTANT	-	reservierte Steuerleitung
P3.9	-	-	N.C.
P3.10	TXD0_TTL	X409 ³	RS232-Schnittstelle TxD, TTL-Pegel
P3.11	RXD0_TTL	X409 ³	RS232-Schnittstelle RxD, TTL-Pegel
P3.12	/WRH	X402-66	Controller Write-High-Signal
P3.13	RTS_TTL	X409 ³	RS232-Schnittstelle RTS, TTL-Pegel
P3.15	CTS_TTL	X409 ³	RS232-Schnittstelle CTS, TTL-Pegel

*) abhängig von Modul-Bestückung
²) auch an X401 verfügbar
³) diese Signale können nur verwendet werden, wenn der RS-232-Treiber U5 nicht bestückt ist

sonstige Controller-Leitungen

Signal	Belegung	verfügbar	Beschreibung
/NMI	POWER FAIL*	X402-07	Not Maskable Interrupt
/RD	/READ	X402-14	Controller Read-Signal
/WRL	/WRL	X402-15	Controller Write-Low-Signal
/READY	-	X402-67	Speicher-Timing-Delayanforderung
ALE	-	X402-11	Address-Latch Enable
/RESET	/RESET	X402-04	Reset Initiate
/RESOUT	/RESOUT	X402-12	Reset Output

*) abhängig von Modul-Bestückung

reservierte I²C-Adreßräume

intern belegte I²C-Adressbereiche		
Gerät	**Option**	**Bereich**
EEPROM		$A8_{HEX}$...AF_{HEX}
RTC	optional bestückt	$A2_{HEX}$...$A3_{HEX}$
Videoprozessor	-	80_{HEX}...$8C_{HEX}$

5.4 Anwendungsgebiete und Sicherheitshinweise

Achten Sie beim Einsatz des grabbMODUL-4 auf die Einhaltung der spezifizierten Betriebsbedingungen. Lesen Sie vor der Inbetriebnahme diese Anleitung sorgfältig.

- Das grabbMODUL-4 dient zur Digitalisierung von Videosignalen von Standard-TV-Kameras und der Verarbeitung dieser Daten. Es können Signale von Composite-Videokameras verarbeitet werden, die den Normen CCIR B,G,H,I und Unternorm CCIR B,G,H,I/PAL entsprechen. Alternativ können Signale nach CCIR M/NTSC eingespeist werden. Die Kamerasignale können auch nach Luma- und Chromaanteil getrennt gemäß der S-Video-Norm zugeführt werden.

- Das grabbMODUL-4 ist als Bestandteil eines Systems ausgelegt. Bei der Konzeption des Systems sind die Technischen Daten des grabbMODUL-4 sowie die für den Anwendungsfall geltenden Normen und Sicherheitsrichtlinien zu beachten.

- Das Gerät ist für den Einsatz in trockener und sauberer Betriebsumgebung konzipiert. In den meisten Anwendungen ist es erforderlich, das grabbMODUL-4 mit einem geeigneten Gehäuse zu versehen, um die Betriebsbedingungen einzuhalten. Es ist zu prüfen, ob zusätzliche Maßnahmen zur Einhaltung von Sicherheits- und Funkstörnormen sowie zur Aufrechterhaltung der Betriebssicherheit erforderlich sind.

- Bei gewerblicher Anwendung sind die Unfallverhütungsvorschriften des Verbands der gewerblichen Berufsgenossenschaften für elektrische Anlagen und Betriebsmittel zu beachten.

- Vor der Inbetriebnahme ist generell zu prüfen, ob das Gerät für den vorgesehenen Anwendungsfall und Einsatzort geeignet ist. Im Zweifelsfall sind unbedingt Rückfragen bei Fachleuten, Sachverständigen oder dem Hersteller erforderlich.

- Das Produkt ist vor starken Erschütterungen und Vibrationen zu schützen. Erforderlichenfalls ist eine Federung oder Polsterung vorzusehen, die jedoch nicht die Belüftung des Geräts behindern darf.

Eine eventuell notwendige Reparatur darf nur vom Fachmann unter Verwendung von Originalbauteilen durchgeführt werden. Beim Anschluß des Geräts, nur zugelassene und geprüfte Anschlußkabel verwenden. Es muß auf korrekte Abschirmung und Entstörung der Kabel geachtet werden.

5.5 Hinweise zur CE-Konformität und Störsicherheit

Anmerkungen zum EMV-Gesetz für das grabbMODUL-4

Beachten Sie bitte beim Aufbau eines Gerätes oder Systems, das ein grabbMODUL-4 enthält, die Regelungen bezüglich CE-Konformität und Störsicherheit.

Da Microcontroller-Systeme mit hohen Taktfrequenzen arbeiten, besteht die Gefahr der Aussendung von elektromagnetischen Feldern oder Störung durch diese. Dem kann durch sorgfältigen Aufbau des Systems entgegengewirkt werden.

Das grabbMODUL-4 wurde in bestimmten Betriebskonfigurationen erfolgreich auf die Einhaltung der CE-Normen DIN EN 55024 und DIN EN 55022 geprüft.
Die Ergebnisse dieser Prüfungen lassen sich grundsätzlich auf ähnliche Modul-Konstellationen übertragen. Dies ersetzt jedoch nicht eine Prüfung des von Ihnen entwickelten Gesamtsystems.

Achtung!
Beachten Sie, daß durch Anschluß von Zusatzkomponenten, Zuleitungen und durch den Einbau des Moduls in Gehäuse, Geräte oder Anlagen die zertifizierungsrelevanten Eigenschaften des Moduls verändert werden können.
Es liegt daher in der Verantwortung des Systemintegrators, die Normenkonformität des Gesamtgerätes bzw. Gesamtsystems sicherzustellen.

Das grabbMODUL-4 bietet unterschiedliche Möglichkeiten der Masseanbindung zur Anpassung an unterschiedliche Betriebssituationen. Ein Teil der Befestigungsbohrungen liegen auf Betriebsmasse (GND). Über metallische Stehbolzen oder Kabelverbindungen kann GND des Moduls mit dem Gehäuse oder PE verbunden werden. Durch Kunststoff-Stehbolzen kann eine Isolierung der Massepotentiale erreicht werden. Die kann wünschenswert sein, um z.B. Masseschleifen zu vermeiden.

Die neben den Video-Buchsen und der RS-232-Buchse befindlichen Shield-GND Anschlüsse X403 und X406 stellen eine Verbindung zu den Schirmanschlüssen der entsprechenden Buchsen her. Sie können bei Bedarf zur Schirmanbindung an das Gehäuse o.ä. benutzt werden.

Bei den von PHYTEC durchgeführten Messungen der Grundkonfiguration VM-004 wurden keine zusätzlichen Schirmmaßnahmen benötigt.

Achtung!
Beachten Sie, daß starke Störspitzen auf den Videosignalen oder auf dem Kabelschirm die Eingangsstufe des grabbMODUL-4 zerstören können.
In stark gestörten Industrieumgebungen und bei Verwendung langer Zuleitungen sind daher unbedingt zusätzliche Entstörmaßnahmen vorzusehen.
Bei Verlegung langer Videoleitungen oder dem Einbau der Bildverarbeitungskomponenten in Maschinen oder Anlagen können Mantel- bzw. Potentialausgleichsströme auftreten, die durch geeignete Vorrichtungen von den Eingängen des grabbMODUL-4 fernzuhalten sind.
PHYTEC kann keine Haftung für Schäden übernehmen, die durch unsachgemäßen Anschluß der Signalquellen entstanden sind.

Teil 3

Programmierhandbuch

6 Treiber-Software

6.1 Grundlagen

Damit das grabbMODUL-4 in der von Ihnen vorgesehenen Applikation eingesetzt werden kann, muß eine passende Anwendungssoftware (*Firmware*) im Modul vorhanden sein.
Diese Firmware können Sie entweder selbst erstellen oder Sie können auf eine bereits vorgefertigte Firmware zurückgreifen, die auf eine bestimmte Aufgabe bzw. ein bestimmtes Aufgabengebiet zugeschnitten ist.
Ein weiterer Weg ist es, sich von uns eine an Ihre Aufgabe angepaßte Firmware erstellen zu lassen.
Je nach Vorgehensweise sind unterschiedliche Schritte und Entwicklungswerkzeuge erforderlich, um die Anwendung zu erstellen:

- **Erstellen einer eigenen Software für das grabbMODUL-4**

  Hier erhalten Sie die größtmögliche Flexibilität bei der Gestaltung der Anwendungssoftware. Sie erstellen ein eigenes Programm für den C165-Rechenkern des grabbMODUL-4.
  Dazu benötigen Sie Kenntnisse in der Programmierung dieses Microcontrollers und eine entsprechende Entwicklungsumgebung (z.B. von Keil). Die Programmierung wird typischerweise in C durchgeführt.
  Das erstellte Firmware-Programm kann mit Hilfe der PHYTEC-FlashTools, die im Lieferumfang des Moduls enthalten sind, in den Flash Speicher des grabbMODUL-4 programmiert werden.

  Falls Ihre Aufgabenstellung die Kommunikation mit einem anderen Gerät vorsieht (z.B. Bildfernübertragung), so muß auch für dieses Gerät ein Programm erstellt werden. Bei Bildübertragung an einen PC wäre das z.B. eine Darstellungs- und Kommunikationssoftware auf dem PC. Diese erstellen Sie mit den bekannten Werkzeugen für die PC-Programmierung.

Variante 1) Erstellen eigener Software für das grabbMODUL-4

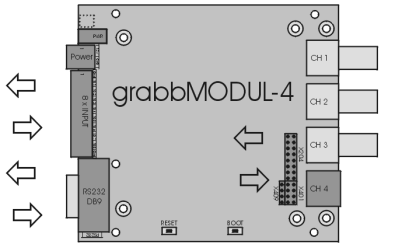
a) Nutzung eines Host zur Kommunikation

Software mit Tools für den eingesetzten Host erstellen.

Software mit Tools für den 80C165 erstellen.

Host

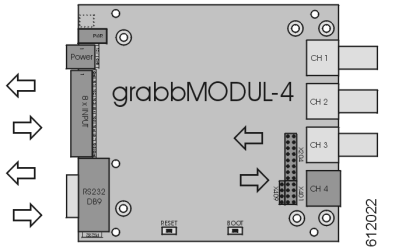
Kommunikation mit einem Host (Embedded Systeme, PC, SPS, ...) direkt über Schnittstellen (seriell, I/O, TTL, I²C,...) oder mit Umsetzer (Telefon, Funk, USB, Ethernet, ...) mittels eigenem Protokoll.



a) Nutzung des Moduls “stand alone”

Software mit Tools für den 80C165 erstellen.

Direkte Steuerung von Applikationen, Schaltungen oder vonBausteinen über vorhandene Schnittstellen (seriell, I/O, I2C, ...).



612022

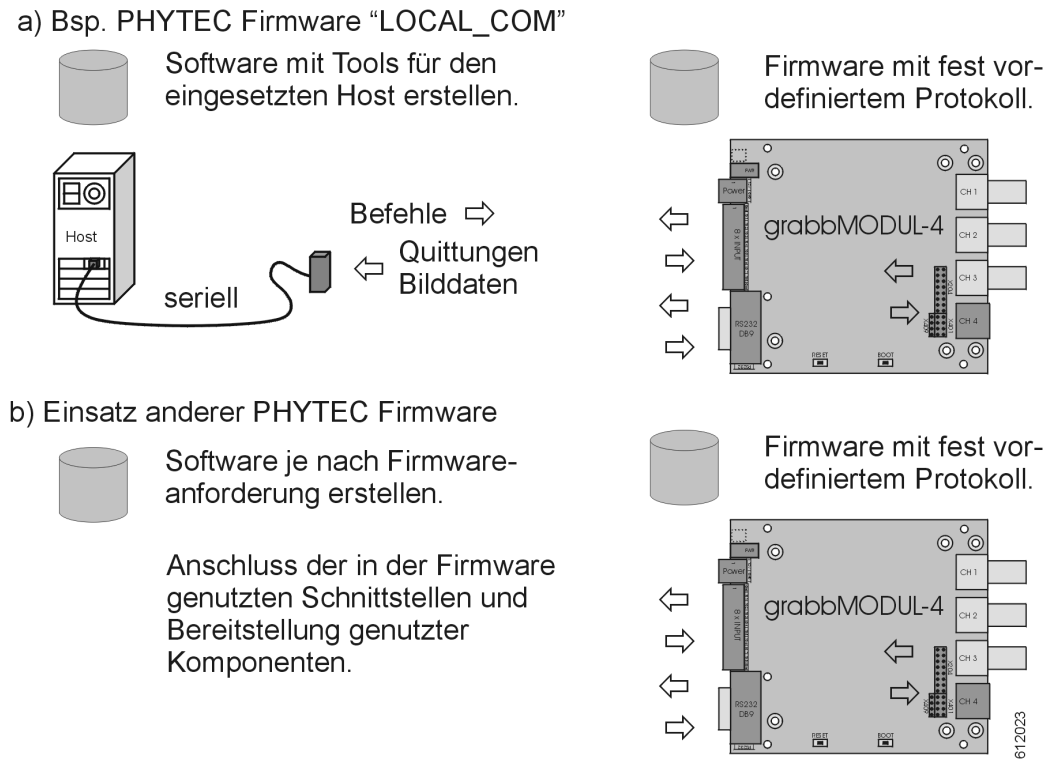
Hinweis:
Zum Verständnis der Programmierung des grabbMODUL-4 und des Aufbaus der Bilddaten empfehlen wir, die *Abschnitte 6.2, „Arbeiten mit Videodaten“ und 6.3.1, „Konfiguration des grabbMODUL-4“ sowie 6.3, „Erstellen eigener Software (Firmware)“* zu lesen.

- **Anwendung einer vorgefertigten PHYTEC Firmware**

  Die Verwendung einer fertigen Firmware hat den Vorteil, daß Sie ohne Kenntnisse der Programmierung des grabbMODUL-4 eine Anwendung erstellen können.
  Allerdings bietet die fertige Firmware lediglich einen festgelegten Funktionsumfang und ist auf die vorprogrammierten Schnittstellen festgelegt. Sie bietet also weniger Flexibilität als eine selbst erstellte Firmware.
  Die fertige Firmware brauchen Sie lediglich mit Hilfe der im Lieferumfang enthaltenen FlashTools in den Programmspeicher des grabbMODUL-4 zu laden.
  Je nach Funktion der Firmware ist gegebenenfalls noch eine Parametrierung erforderlich, die z.B. über die serielle Schnittstelle mit Hilfe eines PCs erfolgt.

  Im Lieferumfang des grabbMODUL-4 ist bereits eine Firmware zur Bildübertragung an einen PC enthalten. Das grabbMODUL-4 wird hierbei über die serielle Schnittstelle mit einem PC verbunden. Über Steuerbefehle kann eine Bildaufnahme gestartet werden. Das Modul überträgt auf Anforderung die Bilddaten an den PC, wo sie ausgewertet oder angezeigt werden können.
  Ein entsprechendes PC-Programm, das die gewünschte Funktion durchführt, können Sie mit den bekannten Werkzeugen für die PC-Programmierung erstellen.
  Eine Beschreibung der Steuerkommandos und des Datenprotokolls auf der seriellen Schnittstelle finden Sie in Abschnitt *6.4.1, „LOCAL_COM: Firmware zur Lokalen Bildübertragung“*.

Variante 2) Nutzung von Phytec Firmware auf dem grabbMODUL-4



Hinweis:
Zum Verständnis des Aufbaus der Bilddaten empfehlen wir, den Abschnitt *6.2, „Arbeiten mit Videodaten“* zu lesen.

6.2 Arbeiten mit Videodaten

In diesem Abschnitt möchten wir einen kurzen Einblick in die Videotechnik und die darin verwendeten Signale und Datenformate geben. Dieses Wissen soll es Ihnen erleichtern, die Abläufe zu verstehen, die zur Bildaufnahme führen. Es ist außerdem Voraussetzung, um die Datenformate der Bilder und das Arbeiten mit ihnen zu verstehen.
(Bei der Beschreibung wird von der europäischen PAL-Videonorm ausgegangen. Die Angaben gelten entsprechend auch für das US-amerikanische NTSC-System, hier unterscheiden sich aber z.B. die Zeilenzahlen.).

6.2.1 Videosignal und Digitalisierungsvorgang

Das vom Grabber verarbeitete analoge Norm-Videosignal besteht aus 625 Zeilen, die in zwei Halbbilder (Fields) aufgeteilt sind und von der Videoquelle nacheinander geliefert werden. Das erste Halbbild (ungerade / odd) beinhaltet die Zeilen 1 bis 313, das zweite (gerade / even) die Zeilen 314 bis 625. Die Halbbilder sind ineinandergeschachtelt, um im TV-Bild den Flimmereffekt zu verringern. Räumlich gesehen folgt daher auf Zeile 1 die Zeile 314. Soll ein Vollbild dargestellt werden, müssen die beiden nacheinander empfangen Halbbilder, entsprechend verschachtelt werden (*Bild 33*).

Neben einigen Vor- und Nachlaufzeilen beinhaltet das Videosignal noch Prüf- und Datenzeilen sowie Zeilen mit Videotext-Information, so daß die effektive Bildfläche aus zwei Halbbildern mit je 288 Zeilen besteht.

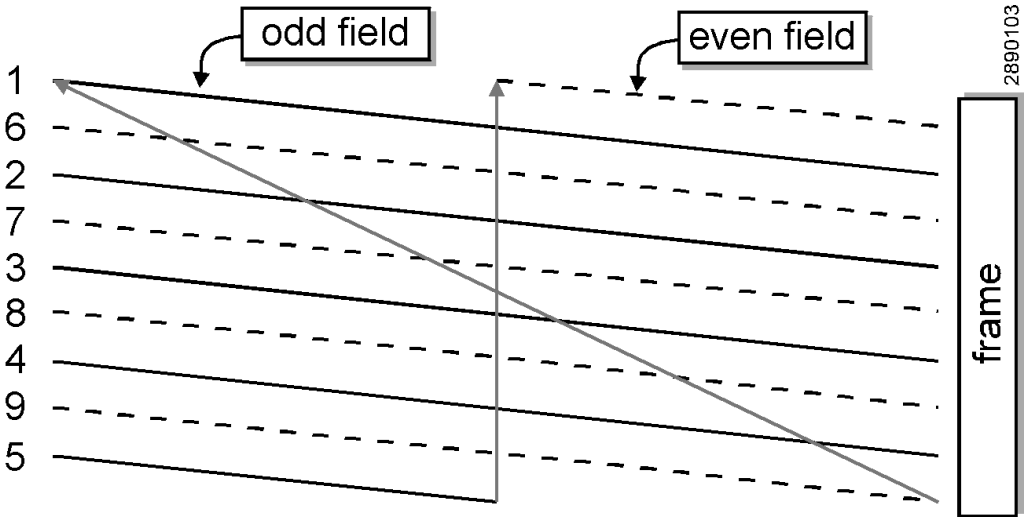


Bild 33: Zeilensprungverfahren (Beispiel mit 9 Zeilen)

Jedes Halbbild wird in 20 ms aufgebaut. In einem Halbbild ist bereits die vollständige Bildfläche zu erkennen, die vertikale Auflösung ist jedoch um die Hälfte reduziert. In vielen Anwendungen ist dies bereits ausreichend, so daß ein vollständiges Digitalisierungsergebnis bereits nach 20 ms vorliegt.

Gegebenenfalls kann die Auflösung in X-Richtung ebenfalls um die Hälfte verringert werden, um ein unverzerrtes Bild zu erhalten.

Eine Auflösungsverminderung in X-Richtung kann die Digitalisierung jedoch nicht weiter beschleunigen, da das Zeitraster des Bildaufbaus vorgegeben ist.

Wird die volle TV-Bildauflösung benötigt, so muß der Einzug beider Halbbilder abgewartet werden (40 ms). Die beiden Halbbilder folgen dabei zeitlich nacheinander.

Um die Verzahnung der Halbbilder zu ermöglichen, ist die letzte Zeile des ungeraden (ersten) Halbbilds auf die Hälfte verkürzt. Demzufolge besteht die erste Zeile des zweiten Halbbilds nur aus der zweiten Zeilenhälfte.

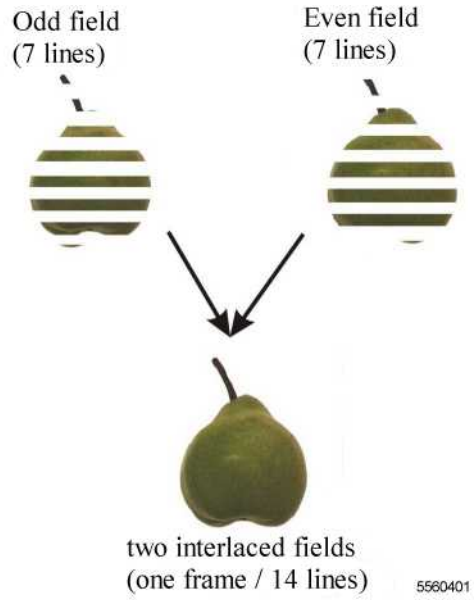


Bild 34: Halb- und Vollbilder

Ein Problem bei der Digitalisierung von TV-Bildern besteht darin, daß ein Objekt bei schneller Bewegung zwischen der Aufnahme des ersten und des zweiten Halbbilds schon ein merkliches Stück vorgerückt ist; die beiden Halbbilder passen also nicht mehr zueinander (Unschärfe / Verzerrungen).

Auch aus diesem Grund benutzt man daher häufig - auf Kosten der vertikalen Auflösung - nur ein Halbbild.

Bild 35: Kammeffekt bei bewegten Objekten im Vollbild-Modus (schematisch)

Wie sieht nun der Zeitablauf eines Digitalisierungsvorgangs aus?

Dies ist abhängig von dem Zeitpunkt, zu dem die Digitalisierung gestartet wurde (relativ zu dem von der Kamera gelieferten Bildsignal) und dem gewählten Bildmodus.

Bild 36 zeigt verschiedene Szenarien:

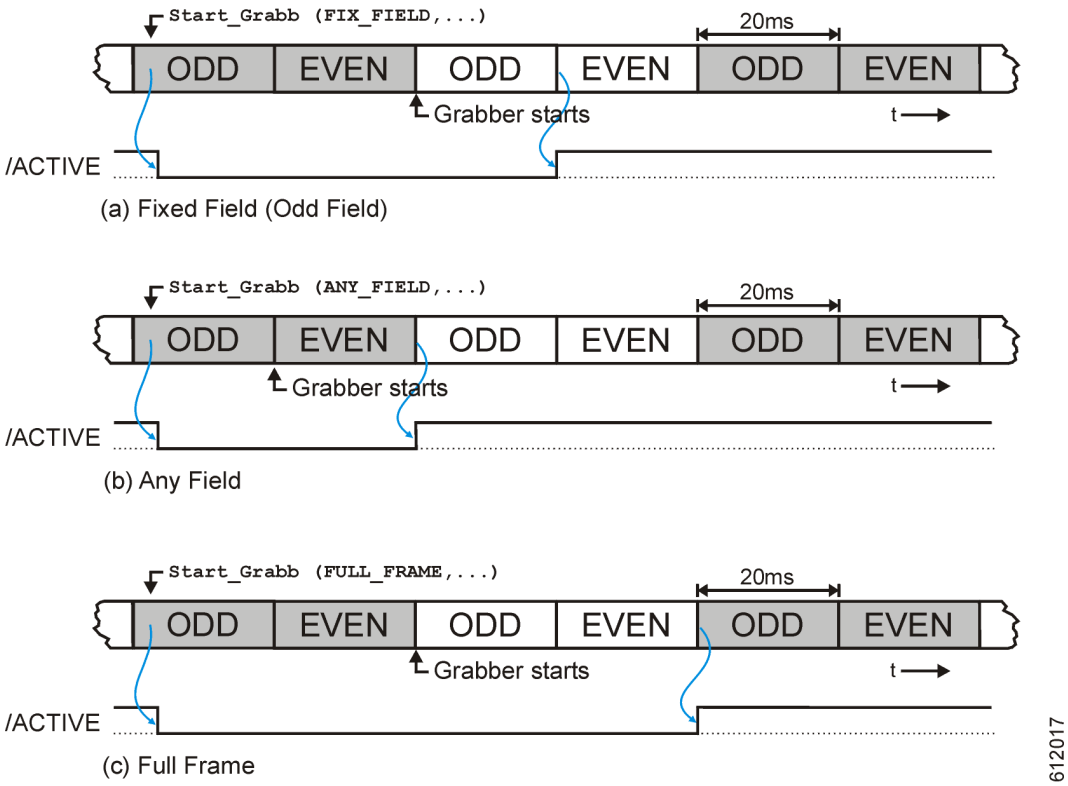


Bild 36: Zeitablauf des Digitalisierungsvorgangs

Das von der Kamera gelieferte Bildsignal enthält abwechselnd das erste Halbbild (ODD) und das zweite Halbbild (EVEN) eines Bilds. Die Dauer eines Halbbilds beträgt 20 ms. Ein vollständiges Bild besteht aus zwei Halbbildern (entsprechend eingefärbt).

Ist eine Bildgröße von maximal 768 x 288 Pixeln ausreichend, wird man nur ein Halbbild zur Digitalisierung anfordern (*Bild 36a*). Je nach Einstellung kann das grabbMODUL-4 in diesem Fall das erste oder das zweite Halbbild eines Vollbilds digitalisieren. Die Default-Einstellung ist „erstes Halbbild“ (ODD); dies ist auch die Grundlage für die Darstellung in Bild (a).
Grundsätzlich ist es egal, ob man das erste oder das zweite Halbbild heranzieht. Es sollte jedoch immer das gleiche Halbbild verwendet werden.

Aufgrund des räumlichen Versatzes der Halbbilder würde ansonsten der Eindruck entstehen, daß das Bild eine Zeile nach oben bzw. unten springt, wenn mehrere Bilder nacheinander dargestellt werden.

Nach Starten des Grabbers mit dem Befehl *Start_Grabb()* wird das Signal */ACTIVE* auf logisch „0“ gesetzt. Es zeigt an, daß ein Digitalisierungsprozeß aktiv ist. Der Framegrabber digitalisiert aber zunächst noch keine Daten, sondern befindet sich im Wartezustand. Die Bildaufnahme startet erst mit dem *Beginn* des nächsten vollständigen Halbbilds der angeforderten Parität (im Beispiel: ODD). Es werden nun die Bilddaten in Echtzeit in das Video-RAM geschrieben. Am Ende des Halbbilds ist die Bildaufnahme abgeschlossen und das Signal */ACTIVE* geht wieder in den Ruhezustand „1“. Jetzt kann vom Microcontroller auf das Video-RAM zugegriffen und die Bilddaten können verarbeitet werden.

Achtung!
Das Signal */ACTIVE* zeigt an, daß der Framegrabber aktiv ist. Während der Zeit, in der */ACTIVE* = „0“ ist, kann nicht auf das Video-RAM zugegriffen werden.
Das Anwenderprogramm muß also nach dem Anstoß einer Bildaufzeichnung das Signal */ACTIVE* prüfen und darf vor Auftreten der nächsten positiven Flanke (Wechsel 0 → 1) nicht auf den Bildspeicher zugreifen.

Lesezugriffe auf das Video-RAM während */ACTIVE* = 0 ist ergeben einen zufälligen Wert.

Da es eine geringe Verzögerungszeit zwischen dem Start-Kommando und */ACTIVE* → 0 gibt, sollte /ACTIVE nicht direkt nach einem Start-Kommando abgefragt werden, um Fehlinterpretationen zu vermeiden. Sinnvoll ist die Einhaltung einer Verzögerungszeit von 15 bis 20 ms.

Hinweis:
Details zur Bildanforderung und zu den Parametern des Befehls *Start_Grabb()* finden Sie im Abschnitt *6.2.6*.

In Bild (a) ist ein zeitlich ungünstiger Fall dargestellt. Das Start-Kommando kommt hier knapp nach Beginn eines ODD-Halbbilds.

Der Framegrabber muß daher fast zwei Halbbilder verstreichen lassen, bis der Beginn eines ODD-Bilds die Bildaufzeichnung starten kann. Zwischen Anforderung und Ende des Digitalisierungsprozesses vergehen daher hier fast 60 ms.

Kommt es auf eine möglichst geringe Verzögerungszeit zwischen Startbefehl und Beginn der Bildaufzeichnung an, kann der Parameter *ANY_FIELD* beim Start des Grabbers verwendet werden (*Bild 36b*). Hier wird das nächste Halbbild digitalisiert, unabhängig davon, ob es sich um ein ODD oder ein EVEN-Halbbild handelt. Nachteilig ist dabei, daß das aufgenommene Bild in der Vertikalen um eine Zeile springen kann (*siehe oben*).
Die Verzögerungszeit zwischen Startbefehl und Digitalisierungsbeginn beträgt hier weniger als 20 ms, der gesamte Vorgang ist nach maximal 40 ms abgeschlossen.

Werden mehr als 288 Zeilen benötigt, muß ein Vollbild angefordert werden (*Bild 36c*). Hierzu wird der Parameter *FULL_FRAME* benutzt. Der Zeitablauf gestaltet sich dabei prinzipiell wie bei der Anforderung eines Halbbilds. Da nun jedoch zwei aufeinanderfolgende Halbbilder aufgezeichnet werden, dauert die Aufnahme 20 ms länger. Die Halbbilder werden im Video-RAM auch räumlich nacheinander abgelegt und müssen zur Darstellung oder Auswertung wieder entsprechend verzahnt werden (*Bild 37*).

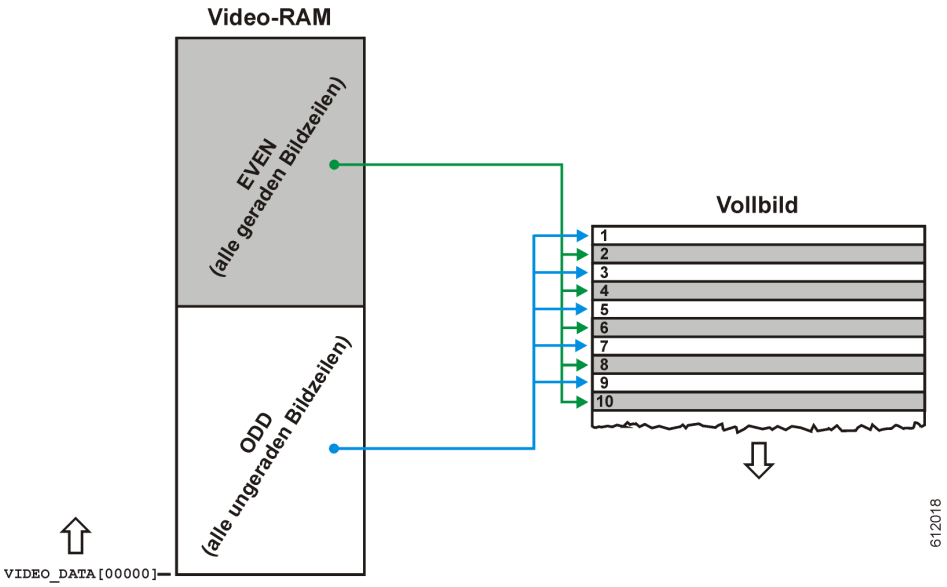


Bild 37: Organisation der Bilddaten bei Vollbildspeicherung

6.2.2 Farbübertragung und Farbspeicherung

Farbinformation und Bildhelligkeit werden zur Bildübertragung von den TV-Systemen prinzipiell getrennt. Übertragen wird das Bildhelligkeitssignal (Luma-Signal, Y-Signal) und das Farbdifferenzsignal (Chroma-Signal). Letzteres charakterisiert die Farbe eines Bildpunkts durch die beiden Parameter Farbton und Farbsättigung.

Die Fernsehsysteme reduzieren die Übertragungsbandbreite des Farbsignals gegenüber dem Helligkeitssignal. Die Farbe wird also „unschärfer“ übertragen als die Helligkeit eines Bildpunkts. Dies entspricht anschaulich gesprochen dem Vorgehen eines Zeichners, der die Objektkonturen zunächst mit einer spitzen Feder zieht und das Bild anschließend mit einem groben Pinsel coloriert.

Die Y-Bandbreite beträgt beim PAL (B,G,H,I) - System 5 MHz, die des Chroma-Signals 1,5 MHz.

Das Chroma-Signal wird bei PAL auch als U/V-Signal, bei NTSC als Q/I-Signal bezeichnet. V- bzw. I-Signal charakterisieren Rottöne während das U- bzw. Q-Signal blauvioletten Farbtönen zugeordnet ist. Zusammenfassend spricht man vom Cr/Cb-Signal für (Chroma Red / Chroma Blue).

Über das Werte-Tripel (Y,Cr,Cb) werden Helligkeit und Farbe eines Bildpunkts vollständig definiert. Diese Werte lassen sich einem Bildverarbeitungssystem zur Farberkennung oder -kontrolle ohne weitere Vorverarbeitung zuführen.
Um ein Bild – z.B. auf dem Bildschirm eines PCs – darzustellen, wird in der Regel das RGB-Format benötigt. Hier besteht die Information über Farbe und Helligkeit in den Farbauszügen Rot, Grün und Blau (R,G,B).

Die Umrechnung ist nach CCIR-Empfehlung für PAL durch folgende Matrix definiert:

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1{,}371 & -191{,}45 \\ 1 & -0{,}338 & -0{,}698 & 116{,}56 \\ 1 & 1{,}732 & 0 & -237{,}75 \end{pmatrix} \cdot \begin{pmatrix} Y \\ U \\ V \\ 1 \end{pmatrix}$$

Die YCrCb Werte die beim grabbMODUL-4 vom Videobaustein geliefert werden, entsprechen im default Zustand dem Wertebereich Y[16...253] und CrCb[2...253]. Vom Hersteller des Videobausteins werden die Formeln zur RGB-Berechnung wie folgt angegeben:

R = 1,64 (Y-16) + 1,596 (Cr-128)
G = 1,64 (Y-16) - 0,813 (Cr-128) - 0,391 (Cb-128)
B = 1,64 (Y-16) + 2,018 (Cb-128)

Y[16...253], Cr/Cb[16...240], RGB[0...255]

Die Speicherung der Bilddaten erfolgt beim grabbMODUL-4 immer im YCrCb-Format. Dies ist günstiger, da es weniger Datenvolumen benötigt. Statt drei Byte pro Pixel (RGB-Format) sind nur zwei Byte (ein Wort) erforderlich. Die unteren acht Bit spezifizieren dabei die Helligkeit, die oberen acht den Farbanteil (Cr/Cb).

Als Farbanteil wird dabei zu jedem Y-Wert abwechselnd Cb und Cr geliefert. Man erhält so zu jedem Pixel nur die „halbe“ Farbinformation, entweder den Rot- oder den Blauanteil. Die jeweils fehlende Information kann durch Übernahme des Werts des Nachbarpixels gewonnen werden. Übertragen und gespeichert wird die Farbe also nur in der halben Auflösung der Helligkeit. Da die Farbinformation jedoch vom Fernsehsystem ohnehin bandbreitenreduziert übertragen wird (s.o.), wird durch diese Vorgehensweise kein Verlust verursacht. Dieses Datenformat wird als YCrCb 4:2:2 bezeichnet.

Mit dem ersten Pixel jeder Zeile wird Y_1,$Cr_{1/2}$ geliefert, mit dem zweiten Y_2,$Cb_{1/2}$ usf. *Bild 38* verdeutlicht, wie die Daten im Speicher abgelegt werden.

Wie greift man nun auf die Bilddaten, die man benötigt, zu?

In *Bild 38* ist dies für ein Farbbild dargestellt. Es werden hier immer Paare von zwei Pixeln (= zwei 16-Bit-Worten) benötigt, um die erforderlichen Informationen vollständig zu erhalten. Für das erste Pixel wird die fehlende Cr-Information aus dem Farbwert des zweiten Pixels übernommen und umgekehrt für das fehlende Cb des zweiten Pixels der Wert des ersten Pixels eingesetzt. Da die Bandbreite des Farbsignals ohnehin beschränkt ist, gehen dabei keine Informationen verloren, obwohl das auf den ersten Blick so scheint.
Die Gewinnung von Graustufendaten ohne Farbinformation ist einfach: Hier muß einfach nur das untere Byte eines 16-Bit-Wortes ausgelesen werden. Es beinhaltet für jedes Pixel den Helligkeitswert.

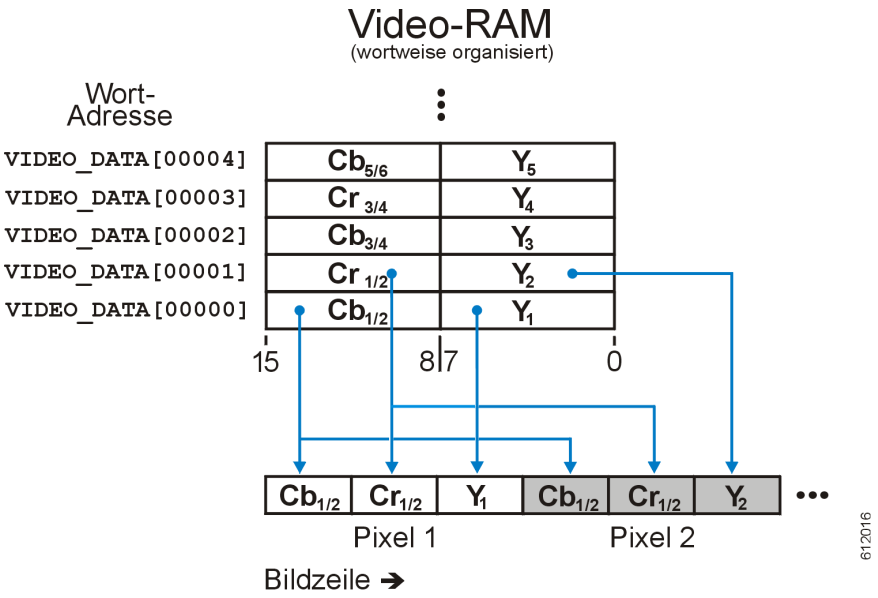


Bild 38: Organisation des Video-RAMs

Hinweis:
Bei Verarbeitung von Farbdaten sollte auf dem Modul am besten immer in YCrCb-Darstellung gearbeitet werden, da diese Darstellung wenig Speicherplatz benötigt und den Vorteil bietet, daß Helligkeit und Farbinformation getrennt sind.
Bei der Übertragung von Farbbildern zu einem PC o.ä. sollte ebenfalls so lange wie möglich das YCrCb-Format beibehalten werden. Erstens reduziert sich dadurch die zu übertragende Datenmenge und zweitens läßt sich die zur Darstellung erforderliche RGB-Umrechnung auf dem PC wesentlich schneller durchführen.

6.3 Erstellen eigener Software (Firmware)

In diesem Abschnitt soll die allgemeine Vorgehensweise bei der Erstellung eigener Software für das grabbMODUL-4 beschrieben werden.

Folgende Software-Komponenten werden benötigt:

- Ein PC-Softwaretool, um Programme für den Infineon-Microcontroller C165 zu erstellen. Hier sollen die Tools (Compiler, Assembler, Linker und Hex-Converter) der Firma Keil, die in der Projektoberfläche µVision zusammengefaßt sind, verwendet werden. Die nachfolgenden Beispiele können auch mit der Light-Version des Keil-Tools umgesetzt werden. *Hinweise zur Installation des Softwareentwicklungstools entnehmen Sie bitte den Herstellerinformationen*.
- Ein PC-Softwaretool, das eine serielle Kommunikation mit dem grabbMODUL-4 herstellt und das generierte Hex-Programm im Flash Speicher des Moduls ablegt. Hier soll das FlashTools-Programm der Firma Phytec Meßtechnik GmbH verwendet werden. Die Handhabung ist im Abschnitt *2.4*, *"Installation der PHYTEC FlashTools und Download eines Programmcodes"* beschrieben.

Vorgehensweise:

- Starten des µVision Tools:
  - Erstellen eines neuen Projekt bzw. Laden eines vorhandenen grabbMODUL-4 Projekts
  - Einbinden der Phytec Hardwaretreiber (entfällt bei vorhandenen grabbMODUL-4 Projekt)
  - Schreiben des Programms
  - Compilieren, Assemblieren, Linken und Hexfile erstellen

- Starten der FlashTools:
  - Download des Hexfile in den Flash Speicher

- Fertig - das Programm läuft selbständig auf dem grabbMODUL-4 nach „PowerOn" bzw. „Reset" ab

6.3.1 Konfiguration des grabbMODUL-4

Auf dem grabbMODUL-4 befinden sich in der Standardausführung 256k RAM, 256k Flash und 1MB Video-RAM. Diese Speicherbausteine sind per Hardware jeweils einem Chip-Select Signal zugeordnet und können so über Software bestimmten Speicherbereichen zugeordnet werden.

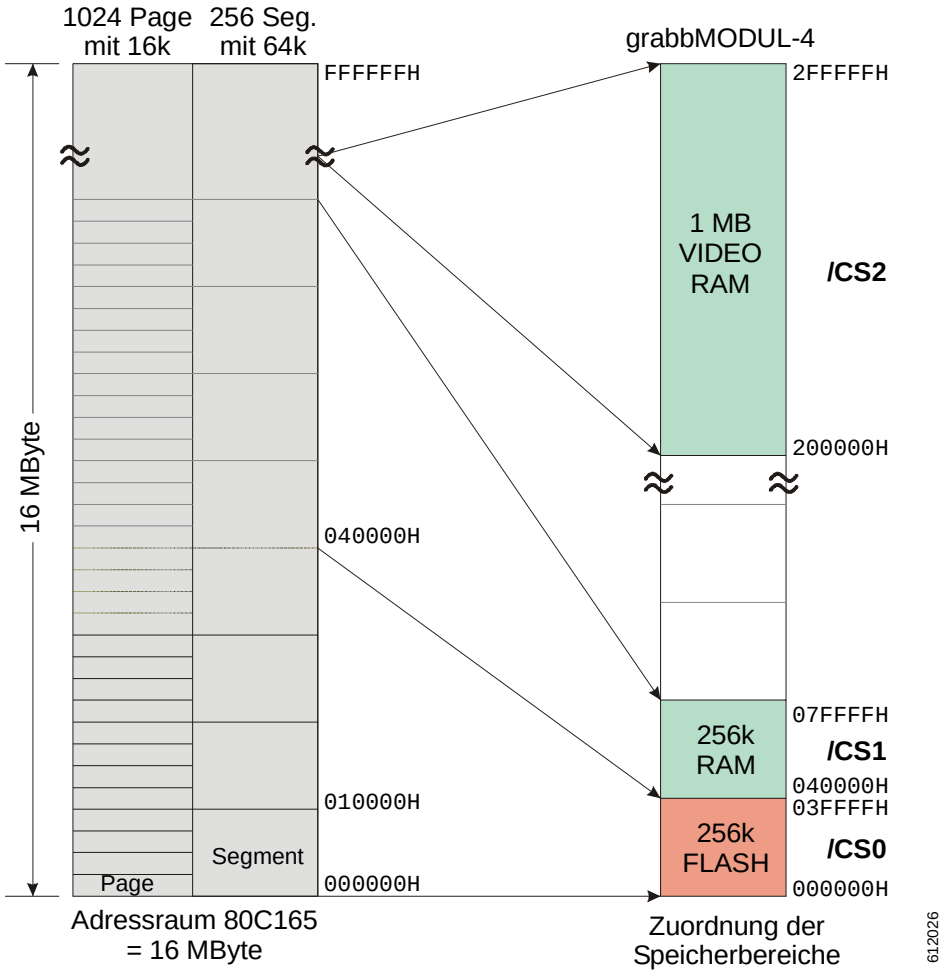


Bild 39: Speicherzuordnung auf dem grabbMODUL-4

Die Zuordnung der Speicherbereiche bei der Grundkonfiguration ist in *Bild 39* dargestellt und wie folgt aufgeteilt:

Speichertyp	Chipselect-Signal	Speicherbereich
256k FLASH	/CS0	000000H – 03FFFFH
256k RAM	/CS1	040000H – 07FFFFH
1MB VIDEO-RAM	/CS2	200000H – 2FFFFFH

Die Zuordnung erfolgt in der Datei „startfla.a65“ und muß ebenfalls in den Projekteinstellungen beachtet werden.
Andere Zuordnungen sind nur dann notwendig, wenn die Speicherbereiche vom Anwender anders gelegt bzw. wenn eine grabbMODUL-4 Variante mit einem anderen Speicherausbau verwendet werden soll.
Als mögliche Bestückungsvarianten kommen in Frage:

	64k	256k	512k	1Mb	2Mb
FLASH		x	x	x	x *
RAM	x	x		x	
VIDEO-RAM	x	x		x	

* Verwendung nur mit Einschränkungen möglich.

6.3.1.1 Adressierung des Speichers

Im Allgemeinen wird die Adressierung des RAM/Flash Speichers in den Projekteinstellungen des Softwaretools verwaltet. Bei einer direkten Adressierung müssen die in *Bild 39* gezeigten Zuordnungen beachtet werden.
Eine direkte Adressierung des VIDEO-RAM Speicherbereiches in C ist durch das in der „startfla.a65“ definierten Arrays „unsigned int VIDEO_DATA“ möglich. Hierbei kann direkt auf ein 16-Bit Pixelwert lesend zugegriffen werden.

```
Pixel = VIDEO_DATA[Pixelposition];
```

oder komponentenweise:

```
PixelHelligkeit = (unsigned char) VIDEO_DATA[Pixelposition];
PixelFarbe = (unsigned char)(VIDEO_DATA[Pixelposition]>>8);
```

Die Organisation der Pixelinformation in diesem 16-Bit-Wert ist in Abschnitt *6.2.2* erläutert und in *Bild 39* dargestellt.

Hinweis:
Auf den VIDEO_DATA[] Bereich kann vom Controller aus nur lesend zugegriffen werden!

6.3.2 Bearbeiten eines vorhandenen Projekts

Aufgabe:
Erweiterung eines aktuellen Projekts wie folgt:

- Abfragen des Videokanals 4, ob ein Videosignal vorhanden ist
- Ausgabe der Information über die serielle Schnittstelle

Realisierung:

Im Folgenden wird die Projektoberfläche µVision der Firma Keil verwendet. Weiterhin beziehen sich die Projekteinstellungen auf einen Speicherausbau 256k RAM und 256K FLASH sowie 1MB Videospeicher auf dem Modul. Das zu generierende Hex-File soll aus dem Flash Speicher des Moduls lauffähig sein.

1) Legen Sie ein neues Verzeichnis auf Ihrer Festplatte an (z.B. ***...\gm4_Test***).

2) Kopieren Sie von der grabbMODUL-CD das gesamte Verzeichnis ***„grabbMODUL4\Modul_Programm\Sourcen\gm4_Test\“*** in dieses angelegte Verzeichnis auf ihrer Festplatte.

3) Starten Sie die µVision Oberfläche (kostenlose Light-Version auf der SO-670 CD) auf Ihrem Rechner und laden Sie das Projekt „`gm4_Test.Uv2`“ aus ihrem Verzeichnis. Bei diesem Projekt sind alle für das grabbMODUL-4 relevanten Projekteinstellungen vordefiniert. Weiterhin sind alle Softwaretreiber in das Projekt eingebunden und die notwendigen Softwareinitialisierungen aufgerufen worden.

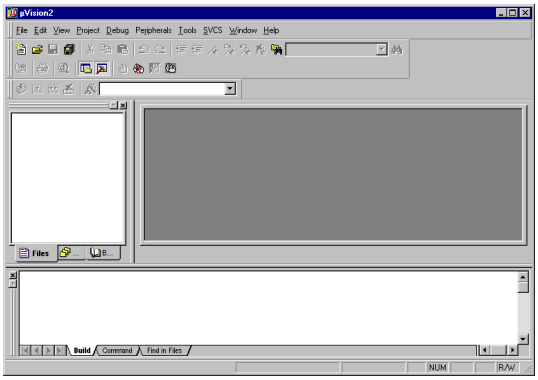


Bild 40: Projektoberfläche µVision

4) Aktivieren Sie das Registerblatt „gm4_Test.c“, um die Software-Source zu bearbeiten.

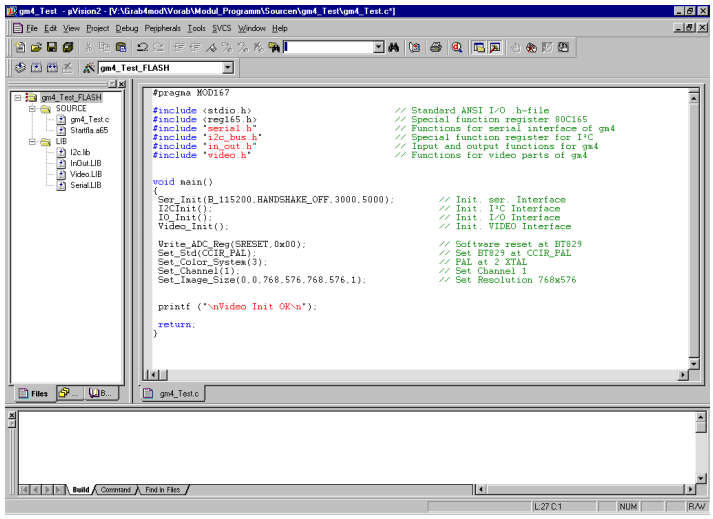


Bild 41: Projekt „gm4_Test.Uv2“

5) Suchen Sie im main() die freie Zeile zwischen printf ("\nVideo Init OK\n"); und return; :

```
printf ("\nVideo Init OK\n");
```

⇦

```
return;
```

Bis zu dieser Stelle sind alle relevanten Initialisierungen für:

- die serielle Schnittstelle (115200 Baud, 8-Bit)
- Grabber-Grundfunktionen
- Videosignale (PAL / CCIR)

durchgeführt worden.

6) Fügen Sie nach der Zeile *printf ("\nVideo Init OK\n");* folgenden Code ein:

```
while(1) {
printf ("\rSignal Kanal 4 = %i",((unsigned int)(Read_ADC_Reg(STATUS)&0x80)>>7));
}
```

Damit wird dauerhaft das „Signal-Present“-Flag im STATUS-Register des Videobausteins abgefragt. *Eine nähere Beschreibung zu den einzelnen Flags im Videobaustein finden Sie im Handbuch zum BT829A.*

7) Erstellen Sie dann aus diesen Sourcen durch <Project ⇨ Rebuild all target files> ein ***gm4_Test.h86*** – Hex-File.

8) Um das ***gm4_Test.h86*** - File in das grabbMODUL-4 zu laden, verwenden Sie bitte die PHYTEC FlashTools. Die Handhabung zum Download eines Files ist in Abschnitt *2.4, "Installation der PHYTEC FlashTools und Download eines Programmcodes"* beschrieben.

9) Da dieses Programm nach einem Modul-Reset ständig das „Signal-Present-Flag" auf der seriellen Schnittstelle ausgibt, benötigen Sie noch eine Möglichkeit, diese Information auch sichtbar zu machen. Zu diesem Zweck nutzen sie ein Terminalprogramm auf einem PC. Die Einrichtung und das Starten des Terminalprogramms sind in Abschnitt *2.5.4, „Testen des Modul-Programms mit einem Terminal-Programm"* beschrieben.

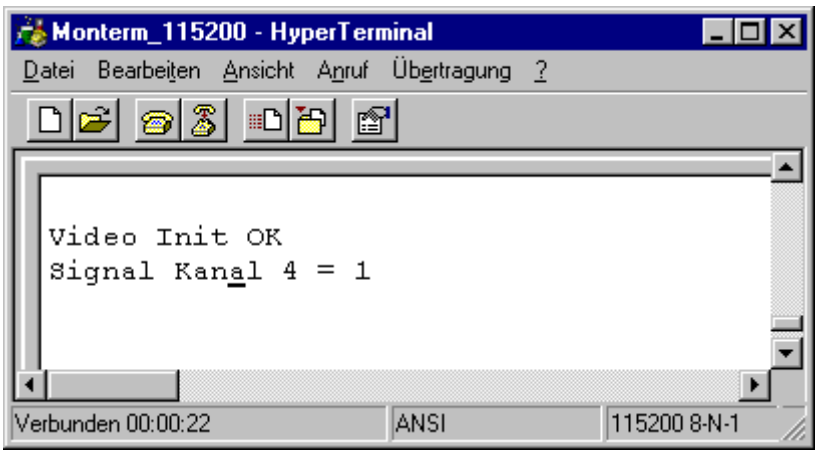


Bild 42: Terminal0programm mit Testausgabe

Das Terminalprogramm sollte eine „1" ausgeben wenn ein Videosignal an Kanal 4 angeschlossen ist. Im Falle einer „0" überprüfen Sie bitte die Videoleitung bzw. das Kamerasignal.

Hinweis:
Bei Verwendung anderer Varianten des grabbMODUL-4 (VM-004-Xx) müssen die speziellen Projektunterverzeichnisse gewählt werden. Die mitgelieferte Firmware „LOCAL_COM" kann bedingt durch die Größe nur mit der Vollversion des Keil-Compilers bearbeitet werden.

6.3.3 Erstellen eines eigenen Projekts

Aufgabe:
Erstellen eines Projekts für das grabbMODUL-4:

- Projekteinstellungen vornehmen
- Libraries und Sourcen ins Projekt einbinden
- Source-File mit folgenden Funktionen erstellen:
  - Initialisierung der seriellen Schnittstelle
  - Grundinitialisierungen der Grabber-Funktionen auf dem Modul
  - Abfragen des Videokanal 4, ob ein Videosignal vorhanden ist
  - Ausgabe der Information an der seriellen Schnittstelle

Realisierung:

Im Folgenden wird die Projektoberfläche µVision der Firma Keil verwendet. Weiterhin beziehen sich die Projekteinstellungen auf einen Speicherausbau 256k RAM und 256K FLASH sowie 1 MB Videospeicher auf dem Modul. Das zu generierende Hex-File soll aus dem Flash Speicher des Moduls lauffähig sein.

1) Legen Sie ein neues Verzeichnis auf Ihrer Festplatte an (z.B. ***...\gm4_Test***)

2) Kopieren Sie folgende Dateien von der CD in Ihr lokales ***...\gm4_Test\...*** Verzeichnis:

- ***...Modul\Programm\LIBs\Video_Lib\Hlarge*** ***VIDEO.LIB***
- ***...Modul\Programm\LIBs\Video_Lib*** ***VIDEO.H***
- ***...Modul\Programm\LIBs\Serial_Lib\Hlarge*** ***SERIAL.LIB***
- ***...Modul\Programm\LIBs\Serial_Lib*** ***SERIAL.H***
- ***...Modul\Programm\LIBs\InOut_Lib\Hlarge*** ***INOUT.LIB***
- ***...Modul\Programm\LIBs\InOut_Lib*** ***INOUT.H***
- ***...Modul\Programm\LIBs\I2C_Lib\Hlarge*** ***I2C.LIB***
- ***...Modul\Programm\LIBs\I2C_Lib*** ***I2C_Bus.H***
- ***...Modul\Programm\LIBs\I2C_Lib*** ***I2C_Hard.H***
- ***...Modul\Programm\LIBs\I2C_Lib*** ***Misc.H***
- ***...Modul\Programm\Startup*** ***StartFLA.A65***

Hinweis:
Das File „***startup.a65***“ wird benötigt, wenn Sie ein Modulprogramm zur Abarbeitung des Codes aus dem RAM erstellen wollen.

3) Starten Sie die µVision Oberfläche auf Ihrem Rechner. Eine kostenlose Light-Version der Keil-Software finden Sie auf der SO-670 CD.

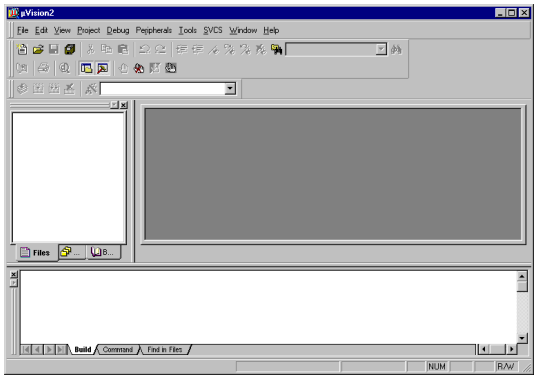


Bild 43: Projektoberfläche µVision

4) Wählen Sie <Project ⇨ New Project> und erstellen Sie ein Projekt mit dem Namen „***gm4_Test***“.

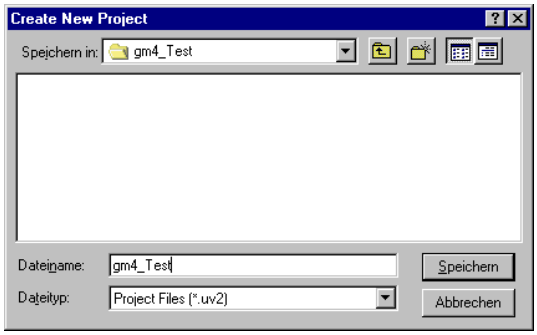


Bild 44: Erstellen eines neuen Projekts

5) Wählen Sie nun <Infineon ⇨ C165> und bestätigen Sie mit OK.

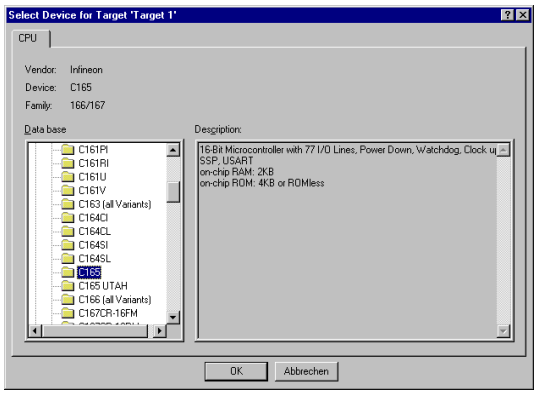


Bild 45: Auswahl des Controllers

6) Bestimmen Sie nun im Projektfenster einen Arbeitsnamen z.B.: *„**gm4_Test_FLASH**“*

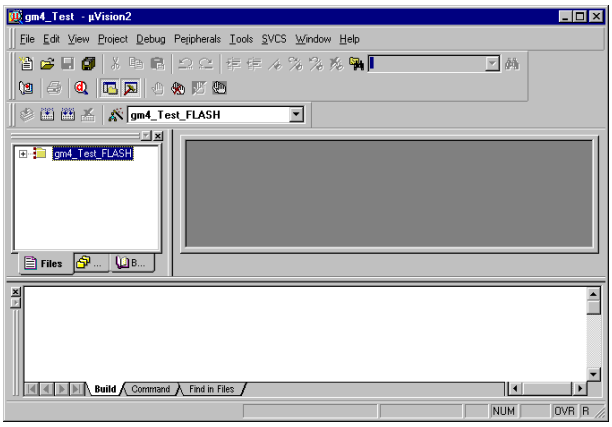


Bild 46: Vergabe eines Arbeitsnamen

7) In nächsten Schritt werden unter <Project ⇨ Options for Target ‚gm4_Test_FLASH‘> die Eigenschaften vergeben. Folgende Einstellungen müssen auf den jeweiligen Registerblättern vorgenommen werden:

Registerblatt: Target

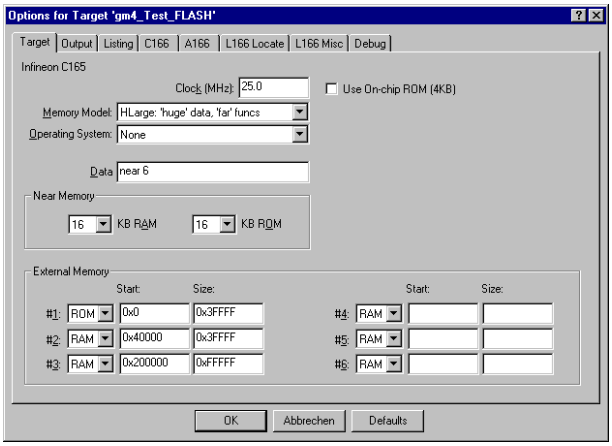


Bild 47: Registerblatt „Target“

Memory Model ⇨ „Hlarge“
External Memory ⇨ #1 ⇨ „ROM“ ⇨ „0x000000“ ⇨ „0x3FFFF“
External Memory ⇨ #2 ⇨ „RAM“ ⇨ „0x040000“ ⇨ „0x3FFFF“
External Memory ⇨ #3 ⇨ „RAM“ ⇨ „0x200000“ ⇨ „0xFFFFF“

Registerblatt: Output

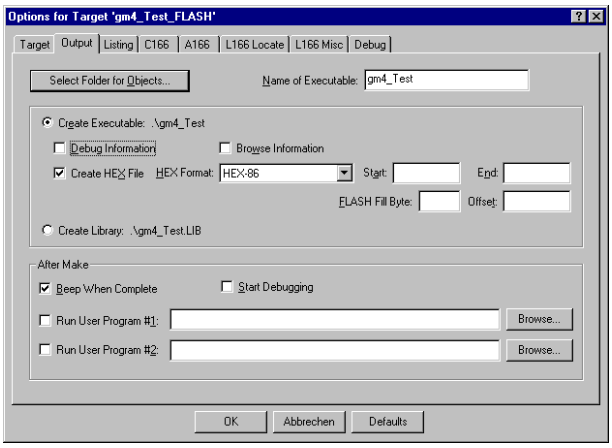


Bild 48: Registerblatt „Output“

Name of Executable: ⇨ „gm4_test“
Create Executable ⇨ Aktivieren Sie: ☑ Create HEX File

8) Es müssen nun die einzelnen Source- und Library-Files in das Projekt eingebunden werden. Zu diesem Zweck werden zwei „Source Groups“ im Projektfenster angelegt. Fügen Sie hierzu die beiden Groups „***SOURCE***“ und „***LIB***“ in <Project ⇨ Targets, Groups, Files, ...> hinzu.

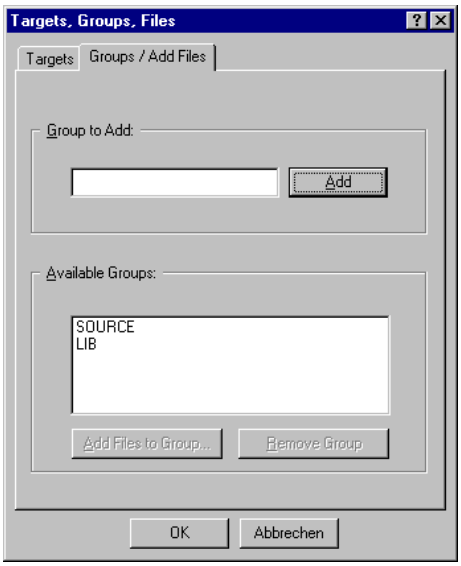


Bild 49: Registerblatt „Targets, Groups, Files, ...“

Zu diesen beiden „Source Groups“ werden durch rechten Mausklick auf die „Source Group“ folgende Dateien hinzugefügt:

- SOURCE ⇨StartFLA.A65
- LIB ⇨ VIDEO.LIB, SERIAL.LIB, INOUT.LIB, I2C.LIB

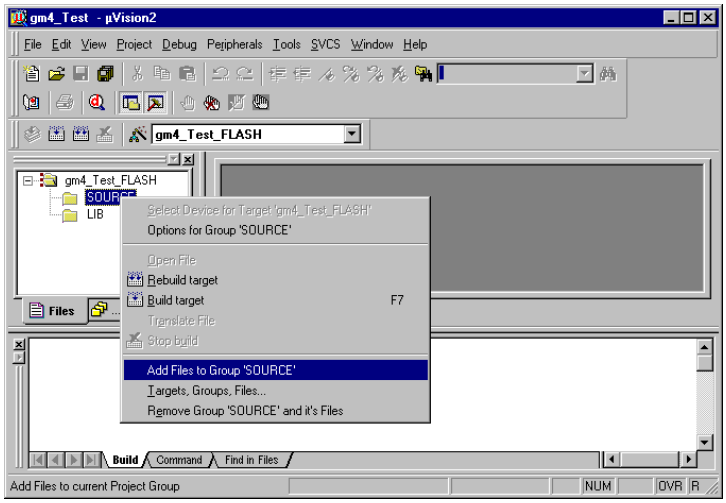


Bild 50: Registerblatt „Add Files to Group ...“

9) Zum Schluß muß noch ein File, in dem der C-Code erstellt werden soll, in das Projekt mit eingebunden werden. Erstellen Sie hierzu ein neues File, speichern es unter dem Namen „***gm4_Test.c***“ und binden es noch in die Source Group „***SOURCE***“ ein. Ihre Projektoberfläche sollte nun fertig eingerichtet sein:

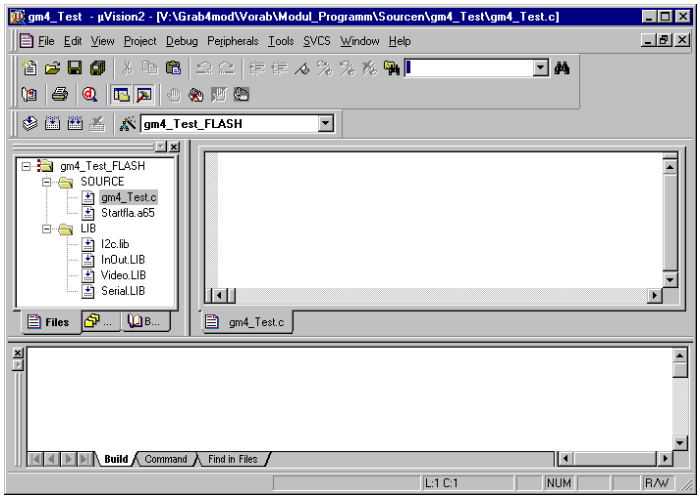


Bild 51: Projekt „gm4-Test“

10) Fügen Sie folgenden Code in die Datei „*gm4_Test.c*“ ein:

```
#pragma MOD167
#include <stdio.h>                                  // Standard ANSI I/O .h-file
#include <reg165.h>                                 // Special function register C165
#include "serial.h"                                 // Functions for serial interface of gm4
#include "i2c_bus.h"                                // Special function register for I²C
#include "in_out.h"                                 // Input and output functions for gm4
#include "video.h"                                  // Functions for video parts of gm4

void main()
{
 Ser_Init(B_115200,HANDSHAKE_OFF,3000,5000);        // Init. ser. Interface
 I2CInit();                                         // Init. I²C Interface
 IO_Init();                                         // Init. I/O Interface
 Video_Init();                                      // Init. VIDEO Interface

 Write_ADC_Reg(SRESET,0x00);                        // Software reset at BT829
 Set_Std(CCIR_PAL);                                 // Set BT829 at CCIR_PAL
 Set_Color_System(3);                               // PAL at 2 XTAL
 Set_Channel(4);                                    // Set Channel 4
 Set_S_Video();                                     // Set S-Video Input
 Set_Image_Size(0,0,768,576,768,576,1);             // Set Resolution 768x576

 printf ("\nVideo Init OK\n");

 while(1) {
   printf ("\rSignal Kanal 4 = %i",((unsigned int)(Read_ADC_Reg(STATUS)&0x80)>>7));
   }

 return;
}
```

Folgender Programmablauf wird umgesetzt:

- Einstellen des MOD167
- Einbinden aller Header-Dateien
- Initialisierung der seriellen Schnittstelle
- Initialisierung der I²C Routinen (Zugriff auf den Videobaustein)
- Initialisierung der I/O Ports auf dem grabbMODUL-4
- Initialisierung des Videobausteins
- Software-Reset des Videobausteins
- Setzen des CCIR/PAL Standards
- Auswahl des CCIR/PAL Quarzes auf dem Modul
- Einstellen des Videoeingangs auf Kanal 4 (S-Video)
- Abfragen des Signal-Present-Flags im Videobaustein (*siehe Handbuch BT829A*)

Hinweis: Die Eigenschaften der Funktionen sind in diesem Handbuch bei der Beschreibung der Libraries erläutert.

11) Erstellen Sie dann aus diesen Sourcen durch <Project ⇨ Rebuild all target files> ein ***gm4_Test.h86*** Hex-File.

12) Um das ***gm4_Test.h86*** File in das grabbMODUL-4 zu laden, verwenden Sie bitte die PHYTEC FlashTools. Die Handhabung zum Download eines Files ist in Abschnitt *2.4, „Installation der PHYTEC FlashTools und Download eines Programmcodes*“Titel beschrieben.

13) Da dieses Programm nach einem Modul-Reset ständig das „Signal-Present-Flag“ auf der seriellen Schnittstelle ausgibt, benötigen Sie noch eine Möglichkeit, diese Information auch sichtbar zu machen. Zu diesem Zweck nutzen sie ein Terminal-Programm auf einem PC. Die Einrichtung und das Starten des Terminal-Programms sind in Abschnitt *2.5.4, „Testen des Modul-Programms mit einem Terminal-Programm“* beschrieben

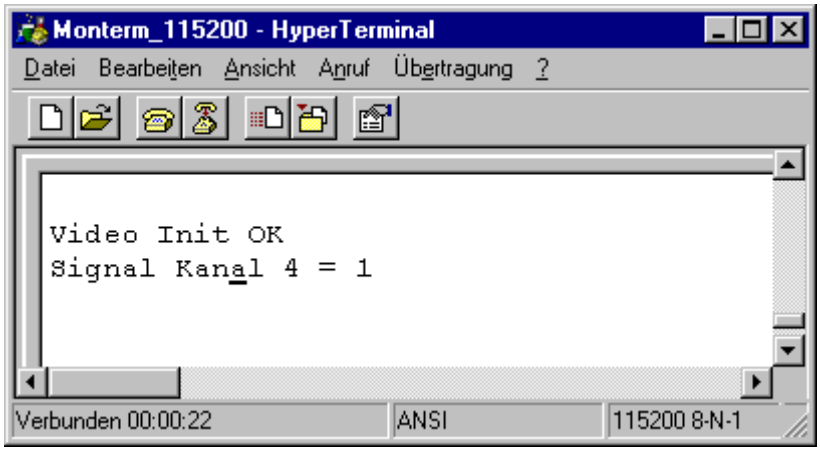


Bild 52: Terminal-Programm mit Testausgabe

Das Terminal-Programm sollte eine „1“ ausgeben, wenn ein Videosignal an Kanal 4 angeschlossen ist. Im Falle einer „0“ überprüfen Sie bitte die Videoleitung bzw. das Kamerasignal.

6.3.4 Prinzipielle Vorgehensweise zur Bildaufnahme

In diesem Abschnitt soll an einigen kleinen Beispielen noch einmal die grundsätzliche Vorgehensweise bei der Initialisierung des Moduls und dem Verarbeiten von Bilddaten erläutert werden.
Da die verwendeten Funktionen und der Aufbau der Bilddaten ausführlich in den anderen Abschnitten erklärt werden, wird hier nur der Ablauf der Aufrufe erläutert.

Achtung!
In diesem Abschnitt werden einige grundlegende Vorgehensweisen beschrieben. Lesen Sie unbedingt diesen Abschnitt, bevor Sie eigene Programme erstellen.

Initialisierung des Microcontrollerkerns:

Diese wird hauptsächlich in der ***„startup.a65“*** (RAM-Variante) bzw. in der ***„startfla.a65“*** (FLASH-Variante) gemacht. Die Änderungen sind in den Kopfzeilen der Dateien erläutert.
Weiterhin müssen bestimmte Speicherzuordnungen in den Projekteinstellungen der Projektoberfläche vorgenommen werden. Nutzen Sie hierzu die von der PHYTEC bereitgestellten Beispielprojekte (*siehe Abschnitte 6.2.2 f.).*

Allgemeine Initialisierungen:

Diese sind nur notwendig, wenn zusätzliche Funktionen genutzt werden sollen.

```
Ser_Init (B_115200, HANDSHAKE_OFF, 3000, 5000);
```

- Initialisiert zum Beispiel die serielle Schnittstelle.

Initialisierung der Videobausteine:

Diese ist notwendig, um Videobilder aufzunehmen und auf die digitalisierten Bilder zugreifen zu können.

I2CInit();
- Initialisiert die I²C Routinen, um auf die Einstellungs-Register des Videobausteins zugreifen zu können.

IO_Init();
- Initialisierung ist nur notwendig, wenn die I/O Ports auf dem grabbMODUL-4 verwendet werden sollen.

Video_Init();
- Initialisierung aller zur Digitalisierung notwendigen Bausteine. Zur Nutzung der grabb-Funktion ist dieser Aufruf unbedingt notwendig!

Einstellungen vor einer Bildaufnahme:

Folgende Funktionen müssen vor einer ersten Bildaufnahme aufgerufen werden:

Write_ADC_Reg (SRESET, 0x00);
- Setzen eines definierten Ausgangszustands im Videobaustein. Diese Funktion sollte nur einmal am Programmanfang verwendet werden!

Set_Std (CCIR_PAL);
- Grundinitialisierung aller Register im Videobaustein BT829. Es werden die Voreinstellungen für eine PAL Videoquelle vorgenommen.

Set_Color_System (3);
- Es wird das Farbsystem und die auf dem Modul realisierte physische Zuordnung zu den Quarzen hergestellt.

Set_Channel (1);
- Einstellung des Kanals an dem die Videoquelle (Kamera) angeschlossen ist.

Set_Image_Size (0 ,0 ,768 ,576 ,768 ,576 ,1);
- Mit dieser Funktion stellen Sie Größe und Form des zu digitalisierenden Bilds ein (wieviel Bildpunkte werden mit welcher Skalierung im Video-Speicher abgelegt).

Bild aufnehmen:

Damit ein Bild digitalisiert und im Videospeicher abgelegt wird, muß der Aufnahmevorgang gestartet werden. Während des Digitalisierungsvorgangs ist kein Zugriff auf Bilddaten möglich. Je nach Bildanforderung kann die Digitalisierung bis zu 80 ms dauern.

```
Start_Grabb (IMAGE_TYPE, RESET);
```

- Starten des Grabbvorgangs, bezogen auf das Videobild, welches als nächstes einläuft und die IMAGE_TYPE-Bedingung erfüllt.

```
while (!noACTIVE);
```

- Warten, bis der Grabbvorgang abgeschlossen ist.

Auf Bilddaten zugreifen:

Der Aufbau des Bildspeichers ist in Abschnitt *6.2.2, „Farbübertragung und Farbspeicherung“* erläutert. Um eine bestimmte Pixelposition zu erreichen, benötigt man die Werte, die in der *Set_image_Size()*- Funktion eingestellt wurden.

Aufgabe 1:

Es soll der Y/C-Wert des Bildpunktes (Pixel) in der Zeile 288 und der Spalte 384 aus einem Vollbild (768 x 576) gelesen werden:

1) Aufruf der Set_Image-Funktion:

   ```
   Set_Image_Size (0 ,0 ,768 ,576 ,768 ,576 ,1);
   ```

2) Daraus ergibt sich ein digitalisiertes Bild im VIDEO_DATA-Speicher des Moduls das wie folgt aufgeteilt ist:

   Halbbild 1 mit den Spalten [1,2,3, ...,768] und den ungeraden Zeilen [1,3,5,...,575] und daran anschließend Halbbild 2 mit den Spalten [1,2,3, ...,768] und den geraden Zeilen [2,4,6,...,576]. Jedes Pixel entspricht einem 16-Bit-Wert, der in Y [Bit7..0] und C [Bit15..8] aufgeteilt ist.

3) Da der VIDEO_DATA - Bereich wortweise organisiert ist, kann der (unsigned int) - Pixelwert direkt gelesen werden:

```
Pixelwert = VIDEO_DATA[Pixelposition];
```
mit
```
Pixelposition = 144L * 768L + 384L
```

Der Pixelwert wird dann noch in Y und C unterschieden:

```
Y_Luma = ((unsigned char)Pixelwert);
C_Chroma = ((unsigned char)(Pixelwert >> 8));
```

Hinweise:

- Zur Berechnung von RGB werden mindestens 2 aufeinanderfolgende Cr/Cb-Werte benötigt, siehe Abschnitt *6.2.2.*, *„Farbübertragung und Farbspeicherung“*.
- Benötigen Sie den Pixelwert in der folgenden Zeile des Vollbilds, so ergibt sich die Pixelposition = 144 * 768 + 384 + Anzahl aller Pixel im ersten Halbbild (768 * 288).
- Auf den VIDEO_DATA[]-Bereich kann vom Controller aus nur lesend zugegriffen werden!

Aufgabe 2:

Es soll der Y/C-Wert des Bildpunkts (Pixels) in der Zeile 144 und der Spalte 192 aus einem Halbbild (384 x 288) gelesen werden.

Realisierung:

1) Aufruf der Set_Image-Funktion:

```
Set_Image_Size (0 ,0 ,384 ,576 ,384 ,576 ,1);
```

2) Daraus ergibt sich ein digitalisiertes Bild im VIDEO_DATA-Speicher des Moduls, das wie folgt aufgeteilt ist:
Halbbild 1 mit den skalierten Spalten [1,2,3, ... ,384] und den skalierten Zeilen [1,2,3, ... ,288]. Jedes Pixel entspricht einem 16-Bit Wert der in Y (Bit7..0) und $C_{R/B}$ (Bit15..8) aufgeteilt ist.

3) Da der VIDEO_DATA Bereich wortweise organisiert ist kann der (unsigned int)-Pixelwert direkt gelesen werden:

```
Pixelwert = VIDEO_DATA[Pixelposition];
```
mit
```
Pixelposition = 144L * 384L + 192L;
```


Der Pixelwert wird dann noch in Y und C unterschieden:

```
Y_Lumina = ((unsigned char)Pixelwert);
C_Chroma = ((unsigned char)(Pixelwert >> 8));
```

Hinweise:

- Zur Berechnung von RGB werden min 2 aufeinanderfolgende Cr/Cb Werte benötigt, siehe Abschnitt *6.2.2, „Farbübertragung und Farbspeicherung.“*.
- Auf den VIDEO_DATA[] Bereich kann vom Controller aus nur lesend zugegriffen werden!

Funktionsbesonderheiten:

An dieser Stelle soll auf die Besonderheit einiger Funktionen und deren Handhabung hingewiesen werden.

1) Folgende Funktionen sollten nur einmal am Anfang des Programms aufgerufen werden:

```
Write_ADC_Reg (SRESET, 0x00);
Set_Std ();
Set_Color_System ();
```

2) Beim Umschalten der Kanäle mit der Set_Channel ()-Funktion kommt es an den analogen Eingängen des Videobausteins zu Pegeländerungen. Diese bewirken, daß das Videosignal erst nach einer gewissen Zeit richtig erkannt werden kann. Deshalb sollte nach einer Kanalumschaltung nicht direkt ein Bild digitalisiert werden. Warten Sie, bis:

a) das Signal korrekt anliegt

```
Set_Channel(1);
Wait_Luma_OK();
```

b) oder eine bestimmte Zeit (ca. 300 ms):

```
Set_Channel(1);
Delay(300);
```

Bei der Verwendung einer S-Videoquelle muß nicht nur der Kanal, sondern auch das S-Videosignal als Signaltyp eingestellt werden:

```
Set_Channel(4);
Set_S_Video();
```

Beim Zurückschalten muß wieder der Signaltyp „FBAS“ eingestellt werden:

```
Set_Channel(1);
Set_FBAS();
```

3) Während des Digitalisierungsvorgangs (max. 80 ms) ist kein Zugriff auf Bilddaten möglich. Der Controller kann jedoch in dieser Zeit voll weiterarbeiten und sich über das noACTIVE-Flag den aktuellen Digitalisierungszustand holen.

noACTIVE = 0 ➔ Grabbvorgang gestartet/aktiv
noACTIVE = 1 ➔ Grabbvorgang beendet/nicht gestartet

Das Warten, bis der Grabbvorgang abgeschlossen ist, ist nur dann notwendig, wenn unmittelbar nach Start_Grabb() mit den Bilddaten gearbeitet werden soll:

```
Start_Grabb();74
while (!noACTIVE);
```

Bei geschickter Aufteilung der Verarbeitung kann der Microcontroller andere Aufgaben ausführen, bei denen nicht auf den Bildspeicher zugegriffen werden muß.

6.3.5 Treiber für den Microcontroller C165-Rechenkern

Das grabbMODUL-4 mit seinem C165 Rechenkern beinhaltet wesentlich mehr Komponenten und Funktionsbausteine als ein einzelnes Rechenkern-Modul. Deshalb hat die Firma Phytec Meßtechnik grundsätzliche und allgemeine Softwareabläufe in Funktionen zusammengefaßt. Diese werden dem Anwender in entsprechenden *Libraries* kostenlos zur Verfügung gestellt.
Die einzelnen *Libraries* werden in den folgenden Abschnitten detailliert vorgestellt und beschrieben. Zur Verwendung der Funktionen aus den *Libraries* müssen diese und die dazugehörigen *Header-Files* in das Anwenderprojekt eingebunden werden.
Neue bzw. erweiterte Libraries werden Ihnen auf unserer Homepage zur Verfügung gestellt.

6.3.6 Video.Lib – Treiber für Framegrabber-Funktionen

Die *Video.Lib* beinhaltet alle Funktionen zum Initialisieren und zum Einstellen der Parameter des Videodigitizers BT829A. Weiterhin wird die Grabber-Steuerung initialisiert.
Damit die *Video.Lib* verwendet werden kann, werden die I²C Routinen benötigt. Deshalb muß in dem zu erstellenden Projekt die *I2C.lib* dazugelinkt und die *i2c_bus.h* eingebunden (included) werden.
Die Definition BYTE steht in diesen Bibliotheken für „unsigned char“ und die Definition von WORD für „unsigned int“.

Achtung!
Vor der ersten Verwendung einer Funktion aus der *Video.Lib* muß die I²C-Schnittstelle initialisiert werden. Dazu muß die Funktion **I2CInit()** einmalig aufgerufen werden. Weiterhin muß die *Video.h* in Ihr Projekt eingebunden (included) werden.

Folgende Funktionen werden in der *Video.Lib* zu Verfügung gestellt:
- Write_ADC_Reg()
- Read_ADC_Reg()
- Video_Init()
- Start_Grabb()
- Set_FIX_Field()
- Set_Std()
- Set_Color_System()
- Set_Brightness()
- Set_Contrast()
- Set_Saturation()
- Set_Hue(int)
- Get_Brightness()
- Get_Contrast()
- Get_Sat_U()
- Get_Sat_V()
- Get_Hue()
- Set_S_Video()
- Set_FBAS()
- Set_BW()
- Set_LDec()
- Set_AGC()
- Set_CKill()
- LumaControl()
- Set_Channel()
- Set_Image_Size()
- Wait_Luma_OK()

Die Funktionen und ihre Parameter werden im folgenden ausführlich beschrieben.

6.3.6.1 Write_ADC_Reg ()

Funktion: Ein Byte in ein Register des Video-Prozessors schreiben

int Write_ADC_Reg(BYTE reg_no, BYTE parameter)

reg_no:	Adresse des Registers im Videoprozessor
parameter:	zu schreibendes Byte

Rückgabewert:	0 = OK -1 = Fehler

Mit dieser Funktion wird ein Byte in ein Register des Video-Prozessors geschrieben. Spezifiziert wird die Adresse des Registers und der zu schreibende Wert. Über den Rückgabewert kann der Erfolg der Operation geprüft werden.

6.3.6.2 Read_ADC_Reg ()

Funktion: Ein Register des Videoprozessors auslesen

BYTE Read_ADC_Reg(BYTE reg_no)

reg_no:	Adresse des zu lesenden Registers

Rückgabewert:	Inhalt des Registers

Diese Routine ermöglicht das Auslesen eines Registers des Videoprozessors.

6.3.6.3 Video_Init()

Funktion: Initialisierung des Video-Framegrabbers

void Video_Init(void)

Diese Funktion initialisiert die für die Steuerung des Video-Grabbers notwendigen I/O Pins des Microcontrollers.

6.3.6.4 Start_Grabb()

Funktion: Bildaufnahme starten

int Start_Grabb (BYTE field, BYTE set_count)

field:	Code für den aufzunehmenden Bildtyp 0 = FIX_FIELD (default ODD) 1 = ANY_FIELD 2 = FULL_FRAME
set_count:	0 = Weiterzählen des V-RAM Zählers 1 = Reset des V-RAM Zählers vor der Aufnahme
Rückgabewert:	0 = OK, -1 = Fehler

Diese Funktion löst die Digitalisierung eines Bildes und das Speichern im Video-RAM-Bereich des grabbMODUL-4 aus. Der Fortschritt des Grabbvorgangs kann über den Status-Pin *noACTIVE* verfolgt werden:

noACTIVE = 0 ➔ Grabbvorgang gestartet/aktiv
noACTIVE = 1 ➔ Grabbvorgang beendet/nicht gestartet

Je nach Anforderung über die Variable *field* werden folgende Bilddaten digitalisiert:

1) **FIX_FIELD:** Der Aufruf der Routine mit diesem Parameter löst die Digitalisierung eines Halbbilds mit der in der Funktion *Set_FIX_Field()* definierten Parität (default: ODD, erstes Halbbild) aus. Das nächste ungerade bzw. gerade Halbbild wird gegrabbt. Die Dauer des Digitalisierungsvorgangs ist abhängig von der eingestellten vertikalen Auflösung und Skalierung. Sie beträgt für ein vollständiges Halbbild 20 ms. Abhängig von dem Startzeitpunkt und der vertikalen Position des Bildausschnitts können bis zu 40 ms bis zum Beginn der Aufzeichnung vergehen (Gesamtzeit max. 60 ms).

2) **ANY_FIELD:** Dieser Parameter löst die Digitalisierung des nächsten verfügbaren Halbbilds aus. Dabei wird nicht darauf geachtet, ob es sich um ein ungerades oder ein gerades Halbbild handelt. Die maximale Verzögerungszeit beträgt hier (wenn der vertikale Ausschnittsbeginn gleich dem Bildbeginn ist) ca. 20 ms (Gesamtdauer max. 40 ms). Der Anwendungsbereich liegt in Applikationen, bei denen möglichst schnell auf ein Triggersignal o.ä. reagiert werden muß.

3) **FULL_FRAME:** Durch den Aufruf mit diesem Parameter wird die Digitalisierung eines Vollbilds, bestehend aus zwei aufeinanderfolgenden Halbbildern angefordert. Die beiden Halbbilder werden sequentiell im Video RAM-Speicher abgelegt. Bei Vollbild-Digitalisierung sollte immer mit dem ODD-Halbbild begonnen werden, damit die Bilder anschließend korrekt verzahnt werden können. Die Dauer des Digitalisierungsvorgangs beträgt 40 ms. Abhängig vom Startzeitpunkt können bis zu 40 ms bis zum Beginn des Aufzeichnungsvorgangs vergehen (Gesamtdauer max. 80 ms).

Mit dem *set_count* Parameter wird festgelegt, an welcher Stelle das angeforderte Bild im Video-Speicher abgelegt wird:

1) ***set_count*=0**: Das nächste zu digitalisierende Bild wird um eine Speicherposition (n+1) versetzt an das vorhergehend gegrabbte Bild gespeichert. Wurde im Vorfeld kein Bild gespeichert, so ist dies die Speicher-Position Null im Video-RAM-Speicher (Startadresse Video-RAM +0x0000). Mit dieser Funktion können mehrere Bilder nacheinander im Video-RAM-Speicher des grabbMODUL-4 abgelegt werden.

2) ***set_count*=1**: Das nächste zu digitalisierende Bild wird ab der Speicher-Position Null im Video-RAM-Speicher abgelegt.

Hinweise:
In den meisten Anwendungsfällen kann mit *set_count* = 1 gearbeitet werden. Das Bild wird dann immer ab dem Beginn des Video-RAMs eingeschrieben.

Wird mit dem Parameter *set_cont* = 0 gearbeitet, so können so viele Bilder hintereinander gespeichert werden, wie Video-RAM vorhanden ist.
Bei einem Speicherausbau von 1 MB (512k = Luma, 512k = Chroma) können z.B. bei einer Auflösung von 128 x 128 Bildpunkten (= 16 kByte pro Bild), 32 Bilder im Video RAM-Speicher des grabbMODUL-4 abgelegt werden. Da Luma und Chroma getrennte Speicherbereiche besitzen, ist es in diesem Rechenbeispiel egal, ob das Bild in schwarz/weiß oder in Farbe angefordert wird.

Wird die obere Grenze des Video-RAMs überschritten, so springt der Schreibzeiger wieder an den Anfang des Bildspeichers zurück und die Bilddaten werden ab dort weiter eingeschrieben (Ringspeicher-Struktur). Daten, die sich an diesen Positionen befinden, werden überschrieben.

Beachten Sie beim Auslesen der Daten, daß bei *set_count* = 0 zwischen den gespeicherten Pixeln jeweils ein „leeres“ Pixel Abstand liegt. Dieses Pixel muß beim Auslesen übersprungen werden.

Durch das Setzen und Rücksetzen des Output-Bits *INIT* kann der Bilddaten-Schreibzeiger per Hand auf die Position Null im Video-RAM gesetzt werden.

6.3.6.5 Set_FIX_Field

Funktion: Halbbild für FIX-Field – Funktion auswählen

int Set_FIX_Field (BYTE odd_even)

odd_even:	Codierung für das aufzunehmende Halbbild 0 = EVEN-Field (2. Halbbild) 1 = ODD-Field (1.Halbbild)
Rückgabewert:	0 = OK, -1 = Fehler

Über diese Funktion kann ausgewählt werden, welches der beiden Halbbilder digitalisiert wird, wenn *Start_Grabb()* mit dem Parameter *FIX_FIELD* aufgerufen wird.

Hinweis:
Die Auswahl muß vor dem Aufruf von *Start_Grabb()* erfolgen und darf bis zum Abschluß des Digitalisierungsvorgangs nicht verändert werden.
Üblicherweise wird das erste Halbbild (ODD) zur Digitalisierung verwendet.

6.3.6.6 Set_Std

Funktion: Videoprozessor mit Standardwerten initialisieren

int Set_Std (BYTE parm_list[])

param_list[ ]:	Zeiger auf Array mit den Inhalten der Prozessor-Register 0x00..0x1A
Rückgabewert:	0 = OK, -1 = Fehler

Mit dieser Funktion können die Register des Videoprozessors auf Standardwerte eingestellt werden. Dazu wird ein Zeiger auf ein Array mit den Inhalten der Register 00_{Hex} bis $1A_{Hex}$ übergeben.

Der Anwender kann für *param_list[]* auf folgende, in der Datei *Video.Lib* vordefinierte Werte zurückgreifen:

1) **CCIR_PAL[]** = Einstellungen für 720x576 CCIR PAL
2) **Square_PAL[]** = Einstellungen für 768x576 Square-Pixel PAL
3) **CCIR_NTSC[]** = Einstellungen für 720x480 CCIR-NTSC
4) **Square_NTSC[]** = Einstellungen für 640x480 Square-Pixel NTSC

Eine Vorinitialisierung der Register des Videobausteins ist zu Beginn des Programms sinnvoll, auch wenn im Programmablauf einzelne Einstellungen (z.B. Auflösung) geändert werden

Der Aufruf

```
#include "Video.h"
...
Set_Std(Square_PAL);
```

stellt z.B. den Videoprozessor des Grabbers für die Bearbeitung von PAL-Bildquellen ein.

Hinweis:
Wir empfehlen dringend, die Initialisierung des Videobausteins mit dieser Routine vorzunehmen, auch wenn Sie später noch einzelne Werte ändern möchten.
Ohne die Initialisierung wird der Baustein nicht richtig arbeiten und die Bildaufzeichnung funktioniert gar nicht oder ist gestört.

6.3.6.7 Set_Color_System

Funktion: Grabber auf verwendetes Farb-System einstellen

int Set_Color_System (BYTE colsys)

colsys:	Code für das Farbsystem 0 = NTSC 1 = PAL 2 = AUTO 3 = PAL_ONLY
Rückgabewert:	0 = OK, -1 = Fehler

Über diese Funktion wird das Farbsystem eingestellt, nach dem der Grabber arbeiten soll. Taktfrequenz und Eingangsregister des Videoprozessors werden entsprechend gesetzt. Der Anwender kann für *colsys* vordefinierte Konstanten einsetzen:

1) **PAL:** Stellt den Grabber für die Verwendung an PAL-Videoquellen ein (wenn nur ein Oszillator bestückt ist).
2) **NTSC:** Stellt den Grabber für die Verwendung an NTSC-Videoquellen ein.
3) **AUTO:** Erkennt das Farbsystem automatisch (nur wenn ein Videosignal anliegt und beide Oszillatoren bestückt sind).
4) **PAL_ONLY:** Stellt den Grabber explizit auf PAL ein (nur wenn beide Oszillatoren bestückt sind).

Hinweise:

- Wird mit dem *colsys* – Parameter ‚AUTO' gearbeitet, kann es zu einer Fehlinterpretation des Farbsystems kommen. In diesem Fall wird das Farbsystem u.U. nicht korrekt eingestellt. Ist eine sichere Erkennung des Systems erforderlich, empfiehlt sich, bei bekannten Videoquellen explizit NTSC oder PAL_ONLY einzustellen.
- In der Standard-Variante VM-004 sind beide Oszillatoren bestückt. Hier dürfen Sie den Parameter ‚PAL' nicht verwenden! Dieser Parameter ist nur für Sonderausführungen mit einem Oszillator bestimmt.

6.3.6.8 Set_Brightness

Funktion: Einstellen der Bildhelligkeit

int Set_Brightness (int bright)

bright: Bildhelligkeit [-128...127], Default = 0 (0%)

Rückgabewert: 0 = OK, -1 = Fehler

Stellt das Register für die Bildhelligkeit im Videoprozessor ein. Der Wert bestimmt eine Konstante, die zu dem Helligkeitswert eines Pixels im Videoprozessor addiert wird. Die Helligkeit läßt sich im Bereich –100 % bis +100 % variieren:

bright = Helligkeit [%] · 1,27 [1/%]

6.3.6.9 Set_Contrast

Funktion: Einstellen des Kontrasts

int Set_Contrast (int contr)

contr: Bildkontrast [0...511], Default = 216 (100%)

Rückgabewert: 0 = OK, -1 = Fehler

Stellt den Kontrast des aufgenommenen Bilds ein. Der Kontrastwert stellt einen konstanten Faktor dar, mit dem der Helligkeitswert der Pixeldaten im Videoprozessor (entsprechend skaliert) multipliziert wird. Die Einstellung ist im Bereich 0 % bis 236,57 % möglich:

contr = Kontrast [%]· 2,16 [1/%]

6.3.6.10 Set_Saturation

Funktion: Einstellen der Farbsättigung

int Set_Saturation (int sat_u, int sat_v)

sat_u:	Sättigung des U-Farbanteils [0...511], Default = 254
sat_v:	Sättigung des V-Farbanteils [0...511], Default = 180

Rückgabewert: 0 = OK, -1 = Fehler

Über diese Funktion wird die Farbsättigung des Bilds getrennt für U- und V-Farbanteil eingestellt. Es handelt sich dabei um eine Veränderung des Verstärkungsfaktors getrennt für die beiden Farbanteile. Normalerweise ist das Verhältnis Werte *sat_u* und *sat_v* gleich. Durch eine Differenz von U- und V-Anteil können Farbstiche (z.B. von nicht abgeglichenen Farbkameras) ausgeglichen werden, also die Farbe des Bilds verändert werden.

sat_u = U-Sättigung [%] * 2,54 [1/%]; Bereich von 0% bis 201,18 %
sat_v = V-Sättigung [%] * 1,8 [1/%]; Bereich von 0% bis 283,8 %

Hinweise:

- Beachten Sie, daß die Nullstellung (100 %) durch die unterschiedlichen Umrechnungsfaktoren für U und V bei verschiedenen
  Registerwerten liegt (für U: 254, für V: 180)
- U und V bilden einen Farbvektor, die Länge des Vektors bezeichnet die Farbsättigung, der Winkel den Farbton. (Die V-Achse charakterisiert Rottöne, die U-Achse Blautöne.)

6.3.6.11 Set_Hue

Funktion: Farbton korrigieren (nur bei NTSC)

int Set_Hue (int hue)

hue:	Farbton, Phasenlage des Farbträgers [-128...127], Default = 0°
Rückgabewert:	0 = OK, -1 = Fehler

Mit dieser Funktion wird bei Digitalisierung von NTSC-Farbbildern der Farbton eingestellt, indem die Phasenlage des Farbträgers variiert wird. Bei PAL hat dieser Wert keine Bedeutung, da hier durch das Farbsystem selbst Phasenfehler automatisch ausgeglichen werden. Die Einstellung ist im Bereich –90° bis +89,3° möglich:

hue = Farbton [°] · 1,422 [1/°]

6.3.6.12 Get_Brightness

Funktion: Helligkeitseinstellung zurücklesen

int Get_Brightness (void)

Rückgabewert:	Inhalt des Helligkeitsregisters des Videoprozessors Bildhelligkeit [-128...127]

Diese Funktion ermöglicht das Auslesen des aktuellen Helligkeitswerts aus dem Videoprozessor.

6.3.6.13 Get_Contrast

Funktion: Kontrasteinstellung zurücklesen

int Get_Contrast (void)

Rückgabewert: Inhalt des Kontrastregisters des Videoprozessors Bildkontrast [0...511]

Die Funktion liest den Inhalt des Kontrast-Registers aus dem Videoprozessor und liefert ihn als Integer-Wert zurück.

6.3.6.14 Get_Sat_U

Funktion: Inhalte des Farbsättigungsregisters U lesen

int Get_Sat_U (void)

Rückgabewert: Wert der aktuellen U-Farbsättigung [0...511]

Der Inhalt des Farbsättigungsregisters U kann durch Aufruf dieser Routine ermittelt werden.

6.3.6.15 Get_Sat_V

Funktion: Inhalte des Farbsättigungsregisters V lesen

int Get_Sat_V (void)

Rückgabewert: Wert der aktuellen V-Farbsättigung [0...511]

Der Inhalt des Farbsättigungsregisters V kann durch Aufruf dieser Routine ermittelt werden.

6.3.6.16 Get_Hue

Funktion: Inhalt des Farbton-Registers zurücklesen

int Get_Hue (void)

Rückgabewert: äquivalent der eingestellten Phasenlage des Farbträgers [-128...127],

Diese Funktion dient dem Rücklesen des eingestellten Hue-Wertes.

6.3.6.17 Set_S_Video

Funktion: Einstellen des ‘S-Video-Modus’

int Set_S_Video (void)

Rückgabewert: 0 = OK, -1 = Fehler

Beim Anschluß von S-Videoquellen muß der zweite ADC des Videoprozessors aktiviert werden. Diese Funktion schaltet den Chroma-ADC ein und deaktiviert die nun überflüssige Farbfalle im Luma-Pfad, wodurch das Bild an Schärfe gewinnt. Außerdem wird der Eingangskanal automatisch auf die S-Video-Buchse eingestellt.

S-Video-Quellen sind Farbkameras, die über einen speziellen Ausgang verfügen, an dem Helligkeits- und Farbsignal getrennt herausgeführt sind.

6.3.6.18 Set_FBAS

Funktion: Einstellen des ‘FBAS-Modus‘ (Composite-Eingänge)

int Set_FBAS (void)

Rückgabewert: 0 = OK, -1 = Fehler

Der Aufruf dieser Routine schaltet den Grabber wieder in den FBAS-Modus zurück: Der Chroma-ADC wird ausgeschaltet und die Farbfalle wieder aktiviert. Dieser Modus muß eingestellt werden, wenn Composite-Signale über die FBAS-Eingänge zugeführt werden. Bei der Standard-Initialisierung wird der FBAS-Mode eingestellt.

Composite-Quellen sind Farbkameras, bei denen Farb- und Helligkeitssignal über eine gemeinsame Leitung übertragen werden (z.B. über eine BNC- oder Cinch-Buchse). Im Vergleich zu S-Video ist die Bildqualität etwas geringer, da an bestimmten Strukturen Störungen (Moiré) auftreten können. Composite-Signale werden im Deutschen entsprechend ihrer Komponenten auch als FBAS-Signale bezeichnet (Farb-Bild-Austast-Synchon-Signal)

6.3.6.19 Set_BW

Funktion: Ein-/Ausschalten der Farbfalle für s/w-Betrieb

int Set_BW (int on)

on: 0 = color-Signal am Eingang (Farbfalle einschalten), Default
1 = s/w-Signal am Eingang (Farbfalle ausschalten)

Rückgabewert: 0 = OK, -1 = Fehler

Wird statt einer Farb- eine Schwarzweißkamera an den Grabber angeschlossen, so ist die Farbfalle, die störendes Farbmoiré aus dem Helligkeitssignal entfernt (*Cross-Color-Effekt*), überflüssig. Mit dieser Funktion kann die Farbfalle ein- und ausgeschaltet werden.

Das Ausschalten der Farbfalle bei s/w-Signalen ist sinnvoll, da durch den Wegfall des Bandfilters die Bildschärfe leicht erhöht wird.

Hinweise:

- Die Farbfalle darf nur bei Anschluß einer schwarzweiß-Kamera abgeschaltet werden. Ansonsten können starke Störstreifen im Bild sichtbar sein.
- Im S-Video-Modus wird die Deaktivierung automatisch vorgenommen.
- Wenn Sie mit Set_FBAS in den Composite-Modus zurückschalten, wird die Farbfalle standardmäßig aktiviert (es wird von Farbvideoquellen ausgegangen).
- Falls nicht sicher ist, welche Bildquelle am Eingang angeschlossen wird, sollte die Farbfalle immer aktiviert bleiben. Der geringe Qualitätsverlust ist in den meisten Anwendungen tolerierbar.

6.3.6.20 Set_LDec

Funktion: Ein-/Ausschalten des Luma-Tiefpassfilters

int Set_LDec (int on, int HFilt)

on:	0 = Luma Decimation ausschalten, Default 1 = Luma Decimation einschalten
HFilt:	0 = Automatische Filterauswahl, Default 1 = CIF Filter (nur bei aktiver Luma Decimation) 2 = QCIF Filter (nur bei aktiver Luma Decimation) 3 = ICON Filter (nur bei aktiver Luma Decimation)
Rückgabewert:	0 = OK -1 = Fehler -2 = unzulässige Kombination (on = 0 und Hfilt = 1,2,3)

Bei kleinen Bildformaten (kleiner 2:1) erzielt man eine bessere Bildwiedergabe, wenn die Auflösung des Eingangssignals reduziert wird (man also die Schärfe des Bilds an die spätere Auflösung anpaßt). Dazu kann mit dieser Routine optional ein Tiefpaßfilter in den Luma-Pfad eingeschaltet werden.

Mit dem Parameter *Hfilt* wird der durch Parameter *on* eingeschaltete Filter der Bildgröße angepaßt.
Automatische Filterwahl macht die Filtereinstellung abhängig von der gesetzten Bildgröße (siehe *Set_Image*). Daneben kann der Filter auf eine der normierten Bildformate *CIF, QCIF* oder *ICON* angepaßt werden.

Hinweis:
Wird der Filtertyp 1, 2 oder 3 mit *Hfilt* ausgewählt, ist dies nur bei eingeschaltetem Tiefpaßfilter *on = 1* möglich.

6.3.6.21 Set_AGC

Funktion: Einstellen der verschiedenen AGC Funktionen

int Set_AGC (int CAGC, int AGC, int Crush)

CAGC:	0 = Nicht-adaptive Chroma AGC, Default 1 = Adaptive Chroma AGC
AGC:	0 = AGC ausgeschaltet 1 = AGC einschaltet, Default
Crush:	0 = Nicht-adaptive AGC, Default 1 = Adaptive AGC
Rückgabewert:	0 = OK, -1 = Fehler

Das grabbMODUL-4 verfügt über zwei AGC-Regelkreise. Die allgemeine AGC überwacht die Signalpegel des Composite- bzw. Y-Signals und regelt die Eingangsamplitude nach. Zusätzlich sorgt die Chroma-AGC für eine Anpassung der Farbträgeramplitude.
Die Composite/Y-AGC kann mit dem Parameter *AGC* ein- bzw. ausgeschaltet werden.
Die Composite/Y-AGC paßt sich bei *Crush*=1 durch adaptives Regelverhalten an das Eingangsvideosignal an.
Die Chroma-AGC paßt sich bei *CAGC*=1 durch adaptives Regelverhalten an die Farbträgeramplitude des Eingangsvideosignals an.

Hinweise:

- Bedingt durch die Hardware-Beschaltung auf dem grabbModul-4 **muß** die *AGC* immer eingeschaltet sein.
- Durch Setzen der Parameter *CAGC* und *Crush* kann ein optimales, kontrastreiches Bild erzeugt werden. Diese Parameter sollten jedoch nicht bei Anwendungen, bei denen mit absoluten Helligkeiten bzw. Farben gearbeitet werden muß, verwendet werden.

6.3.6.22 Set_CKill

Funktion: Ein- bzw. Ausschalten des Farbtöters

int Set_CKill (int CKill)

CKill:	0 = Ausschalten des Farbtöters, Default 1 = Einschalten des Farbtöters
Rückgabewert:	0 = OK, -1 = Fehler

Werden an ein Farbsystem Schwarz/Weiß-Bildquellen angeschlossen, so kann es zu einem leichten Farbrauschen kommen. Der Farbtöter verhindert diesen Effekt, indem er die Anwesenheit des Color-Burst prüft und gegebenenfalls die Farbauswertung automatisch abschaltet.

In speziellen Anwendungen kann es erforderlich sein, ein Farbsignal mit einem sehr schwachen Farbträger auszuwerten. In diesem Fall ist der automatische Farbtöter störend.

6.3.6.23 LumaControl

Funktion: Einstellen des Ausgabeformats des Helligkeitssignals.

int LumaControl (int Range, int Core)

Range:	0 = Wertebereich Luma/Y, 16-253, Default 1 = Wertebereich Luma/Y, 0-255
Core:	0 = alle Helligkeitswerte werden übermittelt, Default 1 = alle Helligkeitswerte <=8 werden auf 0 gesetzt 2 = alle Helligkeitswerte <=16 werden auf 0 gesetzt 3 = alle Helligkeitswerte <=32 werden auf 0 gesetzt
Rückgabewert:	0 = OK, -1 = Fehler

Mit dieser Funktion kann das Ausgabeformat der Helligkeitswerte an die Applikation angepaßt werden.
Range bestimmt den Wertebereich der Helligkeit (mögliche Grauwerte).

- *Range* = 0 entspricht dem normalen Wertebereich, der in CCIR 601 spezifiziert ist. Der Helligkeitsbereich ist damit auf die Werte [16...253] begrenzt, wobei der Wert Y = 16 schwarz entspricht. Der Farbwert entspricht [2...253] mit Cr/Cb = 128 als Null (vorzeichenbehaftete Darstellung).
- *Range* = 1 ermöglicht die Nutzung des vollen Wertebereichs [0...255], wobei der Wert Y = 0 schwarz entspricht. Der Farbwert entspricht [2...253] mit Cr/Cb = 128 als Null (vorzeichenbehaftet).

Eine weitere Möglichkeit, die Helligkeitswerte zu beeinflussen, ist mit dem Einstellen des *Core*-Parameters gegeben. Hierbei werden die durch *Core* (0-3) bestimmten Wertebereiche dem Wert Null zugeordnet (Reduzierung von dunklem Bildrauschen).

6.3.6.24 Set_Channel

Funktion: Einstellen des Eingangskanals

int Set_LDec (BYTE channel)

channel:	Einzustellender Eingangskanal [1..4], optional [1..8] Default = 1
Rückgabewert:	0 = OK, -1 = Fehler

Über diese Funktion kann der Eingangskanal gewählt werden. Es wird der im Digitalisierer befindliche Multiplexer umgeschaltet. Der Kanal 4 dient im S-Video-Betrieb zur Einspeisung der Luma-Komponente (Helligkeit).
Optional kann das grabbMOUDL-4 mit einem zusätzlichen Multiplexer ausgestattet sein. In diesem Fall wird der Kanalbereich auf 1 bis 8 erweitert. Die Umschaltung des zusätzlichen Multiplexes wird dann ebenfalls von der *Set_Channel*-Funktion übernommen.

Hinweise:

- Bei Umschaltung auf den S-Video-Eingang (siehe *Set_S-Video*) wird automatisch der korrespondierende Kanal 4 eingestellt.
- Beachten Sie die Umschaltzeiten, die beim Wechsel von einem Kanal auf einen anderen entstehen (*siehe Abschnitt 6.2.4)*

6.3.6.25 Set_Image_Size

Funktion: Bildgröße und Skalierung einstellen

int Set_FIX_Field (int hpos, int vpos, int hsize, int vsize, int ppl, int lines, int col_system)

hpos, vpos:	Position der linken oberen Ecke des Bildausschnitts im Videobild: (hpos = horizontal, vpos = vertikal)
hsize :	Größe des Bildausschnitts in X-Richtung
vsize :	Größe des Bildausschnitts in Y-Richtung
ppl :	gewünschte Größe des Videobilds in X-Richtung (Pixel Per Line)
lines :	gewünschte Zeilenzahl des Videobilds
col_system:	verwendetes Farbsystem: 0 = NTSC 1 = PAL 2 = automatische Erkennung
Rückgabewert:	0 = OK, -1 = Fehler

Mit der Routine **Set_Image_Size()** werden Größe, Position und Skalierung des vom Grabber gelieferten Bildausschnitts festgelegt.
Über die Parameter *hsize* und *vsize* legt man zunächst ein Fenster mit der Größe des gewünschten Bilds (in Pixeln) fest. *Hsize* gibt also die Anzahl der Pixel in x-Richtung an, die später das aufgenommene Bild hat, *vsize* die Anzahl der Pixel in y-Richtung.
Die Werte *ppl* und *lines* geben hingegen an, wieviel Pixel aus dem ankommenden Videobild generiert werden sollen. *ppl* definiert die Anzahl der Pixel pro Bildzeile, *lines* gibt an, wieviel Zeilen („Pixel in y-Richtung“) erzeugt werden. Mit den beiden Werten kann also die Auflösung des Bilds festgelegt werden. Ein Vollbild im PAL-Format hat 768 Pixel x 576 Zeilen. Um es in höchstmöglicher Auflösung zu digitalisieren, wird *ppl* der Wert 768 zugewiesen und *lines* = 576 gesetzt. Die kleinsten Auflösungen liegen für PAL bei 50 x 40 Pixeln pro Vollbild.

Aufgrund der Tatsache, daß sich die Definition von *ppl* und *lines* auf *Voll*bilder bezieht, die tatsächliche Zeilenzahl der zu digitalisierenden TV-Bilds also doppelt so hoch ist, ist das Breiten- / Höhenverhältnis bei der höchstmöglichen *Halbbild*-Auflösung 720 x 288 Pixel um den Faktor zwei verzerrt. Um dies zu beheben, digitalisiert man entweder zwei Halbbilder (odd und even) und verschachtelt sie zeilenweise ineinander (resultierende Auflösung 768 x 576) oder man reduziert die Auflösung auf *ppl*=384, *lines*=576 und arbeitet mit der proportionsgerechten Auflösung 384 x 288.

Anmerkung:

> Bei Vermessungs- und Automatisierungsanwendungen ist nicht unbedingt eine proportionsgerechte Darstellung nötig, sofern die Verzerrung im Algorithmus berücksichtigt wird. So kann man z.B. die volle Halbbildauflösung 768 x 288 bei Vermessungen nutzen, die so in X-Richtung genauer sein kann als in Y-Richtung.

Das mit *hsize* und *vsize* aufgespannte Fenster ist nun der Ausschnitt des Bildes, den man von dem digitalisierten Bild der Größe *ppl* x *lines* sieht. Ist *hsize* = *ppl* und *vsize* = *lines* so sieht man das ganze digitalisierte Bild; sind sie kleiner, nur einen entsprechenden Ausschnitt. Das Verhältnis von *hsize* und *vsize* verändert nicht die Proportionen des Bildes, da hier keine Skalierung, sondern nur eine Ausschnittsbildung aus dem bereits skalierten Bild vorgenommen wird. *Bild 53* und *Bild 54* illustrieren die Bedeutung der Parameter.

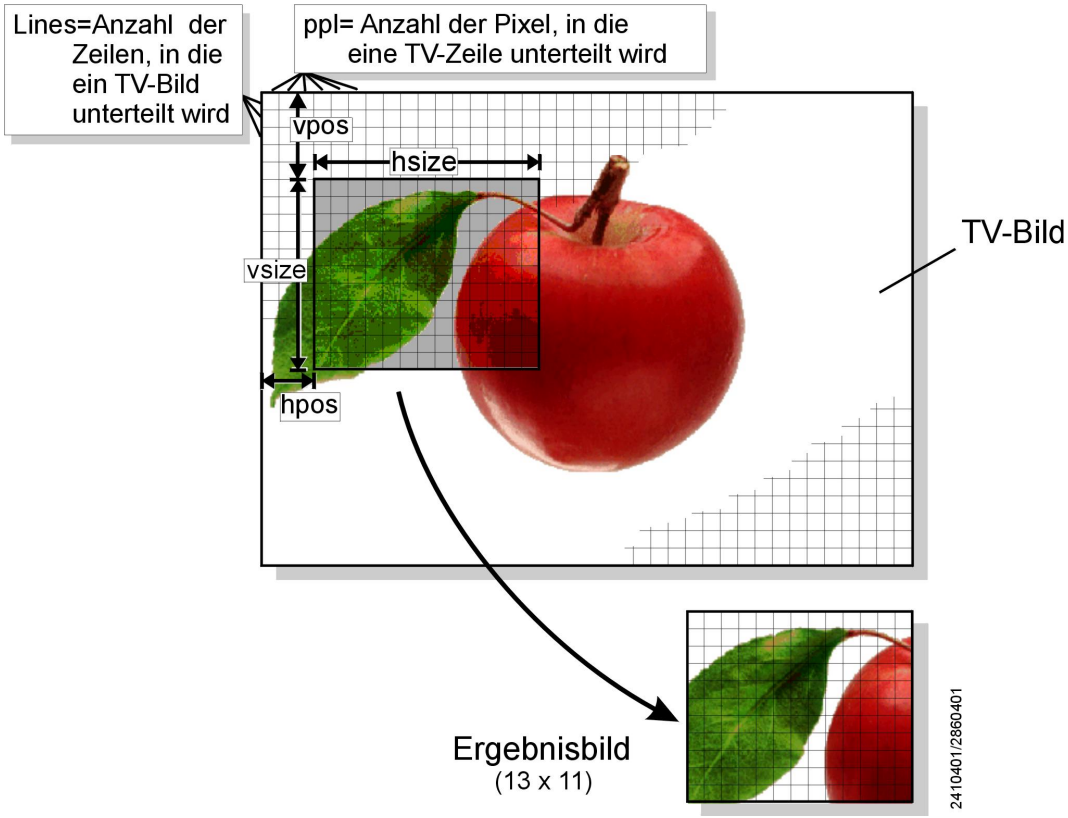


Bild 53: Skalierung und Ausschnittsbildung

Achtung!
vsize und *lines* sind immer in Bezug auf ein Vollbild anzugeben, auch wenn nur ein Halbbild digitalisiert und abgeholt wird.

Ist der mit *hsize* und *vsize* definierte Ausschnitt kleiner als die über *ppl* und *lines* festgelegte Bildgröße, so kann das Fenster mit den Werten von *hpos* und *vpos* in dem digitalisierten Bild verschoben werden. Bei *hpos*=0 und *vpos*=0 liegt es in der linken oberen Ecke des digitalisierten Bilds.

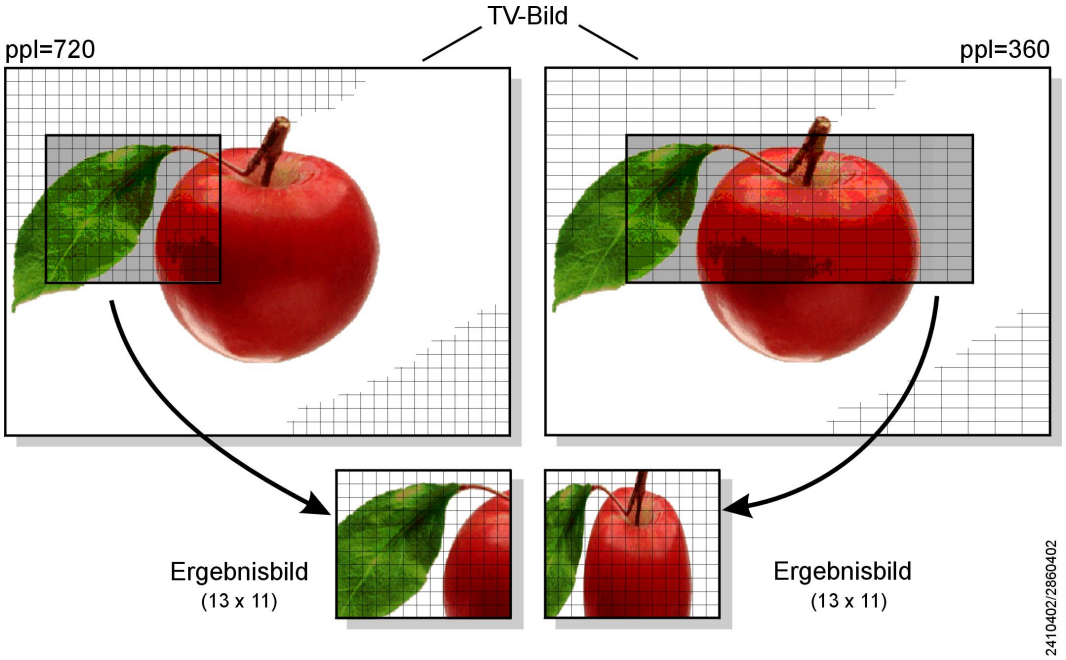


Bild 54: Beispiel zur Skalierung, alle Werte gleich bis auf ppl

Achtung!

- Der über *hsize* und *vsize* in der Größe und *hpos* und *vpos* in der Position definierte Fensterbereich darf an keiner Stelle den durch *ppl* und *lines* definierten Bereich des digitalisierten Bilds verlassen:

hpos=100, hsize=200, ppl=200:	nicht zulässig, da die letzten 100 Pixel nicht definiert sind
hpos=0, hsize=200, ppl=200:	zulässig; alle Pixel liegen im Bildbereich
hpos=100, hsize=100, ppl=200:	zulässig
hpos=100, hsize=200, ppl=300:	zulässig
hpos=300, hsize=300, ppl=800:	nicht zulässig: Bildbereich hat mehr Pixel, als TV-Norm liefert (entsprechend in Y-Richtung)

- Alle Parameter in horizontaler Richtung müssen *gerade* Werte haben! (ppl=123 ist nicht erlaubt, ppl=124 ist zulässig.)
- Alle Parameter in vertikaler Richtung müssen bei Vollbild-Digitalisierung gerade Werte haben !

Beispiele:

- Es soll ein Vollbild 768 x 576 Pixel digitalisiert werden, dessen Auflösung und Proportionen dem TV-Bild entsprechen.
  Set_Image_Size (0,0,768,576,768,576,1)

  Achtung!
  Das digitalisierte Bild besteht aus 2 Halbbildern, die bei der Darstellung von Hand verzahnt werden müssen.

- Es soll ein Halbbild (CIF) 384 x 288 Pixel digitalisiert werden, dessen Auflösung und Proportionen dem TV-Bild entsprechen.
  Set_Image_Size (0,0,384,576,384,576,1)

  Hinweis:
  Das digitalisierte Bild zeigt den selben Bildbereich wie bei der Vollbilddigitalisierung, das Ergebnisbild ist jedoch nur halb so groß (halbe Auflösung).

detailliertes Beispiel:

- Es soll ein quadratisches Bild der Größe 256 x 256 Pixel digitalisiert werden, dessen Auflösung und Proportionen dem TV-Bild entsprechen.

(a) Auflösung / Skalierung

Zur Digitalisierung genügt ein Halbbild mit 288 Zeilen. Damit das Höhen-/Breitenverhältnis stimmt, muß die Auflösung in X-Richtung ebenfalls auf die Hälfte verringert werden (denn: *Halb*bild = halbe Höhe). Also horizontal: 768 : 2 = 384.
Es ergeben sich *ppl*=384
Da in vertikaler Richtung immer der Zeilenwert für Vollbilder eingestellt werden muß, berechnet sich:
lines=288 · 2 = 576

Von diesem Vollbild digitalisieren wir später nur ein Halbbild, also die halbe Zeilenzahl, wodurch sich unser gewünschter Wert von effektiv 288 Zeilen ergibt.

(b) Ausschnittsgröße

Das Bild soll quadratisch 256 x 256 Pixel groß sein. Daraus ergibt sich direkt:
hsize=256, *vsize*=256 · 2 = 512.

(Auch hier gilt, daß die Berechnung in vertikaler Richtung auf Vollbild-Basis beruht.)

(c) Positionierung

Es ist sinnvoll, den Bildausschnitt zu zentrieren. In X-Richtung werden von 384 Pixeln nur 256 im Fenster dargestellt. Es entsteht ein Rand von 384-256=128 Pixeln, der gleichmäßig auf beide Seiten verteilt werden soll, also jeweils 64 Pixel links und rechts. *hpos* ist die Größe des linken Randes, also *hpos*=64.
Entsprechend in Y-Richtung: (576-512):2=32
Bezogen auf ein Vollbild: *vpos*=32 · 2=64

```
Set_Image_Size (62,32,256,512,384,576,1)
```

Bemerkung: Es ist hier falsch, ppl=256 und lines=256 · 2=512 zu setzen. Dadurch würde das Breiten-/Höhenverhältnis verändert (TV-Norm 4:3) werden zu 1:1 und das Bild wäre verzerrt. Es wäre jedoch möglich, lines=256 · 2=512 zu setzen und *ppl* über das Verhältnis von Bildbreite und -höhe zu berechnen. So wäre die Bildhöhe optimal ausgenutzt.

Mit dem Parameter *col_system* wird angegeben, welches TV-Farbsystem benutzt wird. Die Routine verwendet diese Angabe, um die Skalierung korrekt vornehmen zu können. Statt der Zahlenwerte 0, 1, 2 können auch die vordefinierten Konstanten *NTSC*, *PAL*, *AUTO* verwendet werden.

Achtung!
Da bei der Farbbild-Verarbeitung immer Pixelpaare benötigt werden, muß die Anzahl der Pixel pro Zeile immer gerade sein.
Dies ist bei der Einstellung der Bildgröße unbedingt zu berücksichtigen!
Laut Definition beginnt das erste 16-Bit-Wert im Speicher mit dem Cb-Wert, es folgt dann der Cr-Wert.

6.3.6.26 Wait_Luma_OK

Funktion: Wartet, bis angelegtes Videosignal eingeschwungen ist.

int Wait_Luma_OK (void)

Rückgabewert: 0 = Signal OK,
-1 = Timeout abgelaufen

Im Videobaustein befindet sich eine AGC, deren Digitalisierungsfenster sich automatisch an den Pegel des Eingangssignals (Luma-Komponente) anpaßt. Wird ein Videosignal aufgeschaltet, benötigt diese Automatik eine gewisse Zeit, um diese Anpassung vorzunehmen.
Solange das Signal noch nicht eingeschwungen ist, wird ein Luma-ADC-Overflow generiert. Die Wait_Luma_OK Funktion wartet, bis dieses Luma-ADC-Overflow-Flag nicht mehr gesetzt wird.
Diese Funktion sollte verwendet werden, wenn ein Kanal umgestellt wird und möglichst schnell danach ein Bild gegrabbt werden soll.

Hinweis:
Wenn Sie direkt nach einer Kanalumschaltung grabben wollen, rufen Sie erst *„Wait_Luma_OK()“* auf und warten dann noch einige Millisekunden, bis das Signal vollständig stabil anliegt. Ansonsten können störende Helligkeitsschwankungen auftreten.

6.3.7 Serial.lib – Treiber für die serielle Schnittstelle

Die *Serial.lib* beinhaltet Funktionen zum Initialisieren und Arbeiten mit der seriellen Schnittstelle des 80C156 Microcontrollers.
Die Schnittstelle wird auf dem in Abschnitt *3.1.2, „Serielle Schnittstelle“* beschriebenen D-SUB Steckverbinder zur Verfügung gestellt. Die Schnittstelle kann auch im Hardware-Handshake betrieben werden und der Empfang von Zeichen erfolgt über eine Interrupt-Routine.

In dieser Bibliothek steht die Definition ‚BYTE‘ für „unsigned char“ und die Definition ‚WORD‘ für „unsigned int“.

Achtung!
Vor der ersten Verwendung einer Funktion aus der *Serial.Lib* muß die Funktion **Ser_Init()** einmalig aufgerufen werden. Weiterhin muß die *Serial.h* in Ihr Projekt eingebunden (included) werden.

Folgende Funktionen werden in der *Serial.lib* zu Verfügung gestellt:

- Ser_Init ()
- Set_Get_Ser_Timeout()
- Init_Buffer ()
- Put_Buffer ()
- Get_Buffer()
- Buffer_Status()
- Char_Present_TimeOut()
- get_ser()
- read_ser()
- put_ser()
- Read_Error_Code_Serial()

6.3.7.1 Ser_Init

Funktion: Initialisierung der seriellen Schnittstelle mit 8 Datenbits, einem Stoppbit und ohne Parität.

int Ser_Init (WORD baudrate, BYTE handshake,
WORD get_ser_timeout, WORD put_ser_timeout)

baudrate:	gewünschte Baudrate der seriellen Schnittstelle B_9600 = 9600 Baud, B_57600 = 57600 Baud B_115200 = 115200 Baud
handshake:	Einstellen des Hardwarehandshakes (CTS, RTS) HANDSHAKE_ON = Hardwarehandshake aktiv HANDSHAKE_OFF = Hardwarehandshake aus
get_ser_timeout:	Timeout-Zeit [0..65535] in ms beim Empfang eines Zeichens von der seriellen Schnittstelle mit der Funktion get_ser().
put_ser_timeout:	Timeout-Zeit [0..65535] in ms beim Senden eines Zeichen von der seriellen Schnittstelle mit der Funktion put_ser(). **Achtung! Diese Zeit ist nur bei eingeschalteten Hardwarehandshake aktiv**.
Rückgabewert:	0 = OK, -1 = Fehler

Mit der Funktion **Ser_Init()** wird die seriellen Schnittstelle mit der übergebenen Baudrate, dem übergebenen Handshake, 8 Datenbits, einem Stoppbit und ohne Parität initialisiert. Weiterhin werden die Timeout-Zeiten der *get_ser()*- und *put_ser()*- Funktion eingestellt.
Der Empfang eines Zeichens erfolgt mittels einer Interrupt-Routine, die das Zeichen in einem Empfangs-Ringbuffer (*In_Buffer*) ablegt. Dieser Buffer kann 250 8-Bit-Zeichen beinhalten. Der Zugriff erfolgt über die Funktionen *Init_Buffer(), Put_Buffer(), Get_Buffer()* und *Buffer_Status()*.
In der **Ser_init()** wird dieser Ringbuffer aufgebaut und gelöscht und der Empfangs-Interrupt (*S0RIC*) mit Interrupt Level = 1 und Interrupt-Group-Level = 1 initialisiert.

Achtung!
Vor der ersten Verwendung einer Funktion aus der *Serial.lib*, muß die **Ser_Init()** einmalig aufgerufen werden. Die Einstellungen: 8 Datenbits, 1 Stoppbit und keine Parität können mit dieser Funktion nicht verändert werden.

Es wird automatisch ein Eingangs-Ringpuffer mit der Bezeichnung *In_Buffer* erzeugt. Bei Verwendung der nachfolgend beschriebenen erweiterten Ringpuffer-Funktionen in Verbindung mit der seriellen Schnittstelle müssen Sie deshalb für den Übergabeparameter *Queue* die Bezeichnung *In_Buffer* einsetzen.

In den meisten Fällen genügt es, die einfachen Funktionen zur Bedienung der seriellen Schnittstelle zu benutzen:
get_ser(); – Zeichen holen
put_ser(); – Zeichen schreiben
Char_Present_TimeOut(); – auf Zeichen warten
Read_Error_Code_Serial(); – Fehlerbehandlung

Zur Fehlerbehandlung beim Betrieb der seriellen Schnittstelle dient die Fehlervariable ERROR_CODE_SERIAL (Typ unsigned char). Um die serielle Schnittstelle benutzen zu können, müssen Sie diese Variable **global** in Ihrem Hauptprogramm definieren.

6.3.7.2 Set_Get_Ser_Timeout

Funktion: Überschreiben der in der Ser_Init()-Funktion eingestellten Get_Ser_Timeout - Zeit.

void Set_Get_Ser_Timeout (WORD get_ser_timeout)

get_ser_timeout:	Time_Out-Zeit [0..65535] in ms beim Empfang eines Zeichen von der seriellen Schnittstelle mit der Funktion get_ser().

Mit der Funktion **Set_Get_Ser_Timeout()** wird die maximale Zeit festgelegt in der die *get_ser()* – Funktion versucht, ein Zeichen von der seriellen Schnittstelle abzuholen.
Diese Funktion dient zur Änderung der Timeout-Zeit während des Programmlaufs, ohne eine vollständige Initialisierung mit der *Ser_Init()* durchzuführen.

6.3.7.3 Init_Buffer

Funktion: Initialisierung eines Speicherbereiches vom Typ struct Buffer, zum Beispiel des Empfangs-Ringbuffers.

void Init_Buffer (struct Buffer *Queue)

*Queue:	Pointer auf einen Buffer mit folgender Struktur: - unsigned char queue[BUFFER_SIZE]; = FIFO Buffer - unsigned char write; = Position des Schreibzeigers - unsigned char read; = Position des Lesezeigers - unsigned char handle; = Status des FIFO Buffers

Mit der Funktion **Init_Buffer()** werden die Elemente *write, read* und *handle* des übergebenen Buffers auf NULL gesetzt. Das heißt der Buffer wird zurückgesetzt (gelöscht).
Sie können diese und die folgenden Funktionen zur Definition und Verwaltung beliebiger Ringpuffer-Strukturen verwenden.
Die oben beschriebenen Funktionen zur Verwaltung der seriellen Schnittstelle führen die Pufferverwaltung automatisch durch, so daß Sie in den meisten Fällen nicht weiter eingreifen müssen.

6.3.7.4 Put_Buffer

Funktion: Speichern eines Zeichens im übergebenen Buffer.

void Put_Buffer (struct Buffer *Queue, BYTE contents)

*Queue:	Pointer auf eine Struktur vom Typ Buffer (siehe serial.h):
contents:	Zeichen, das in den Ringbuffer abgelegt werden soll.

Mit der Funktion **Put_Buffer()** wird ein Zeichen im Buffer an der Position: *Queue->queue[Queue->write]* abgelegt. Der Buffer wird als Ringbuffer verwaltet und der Füllzustand kann über die Funktion *Buffer_Status()* abgefragt werden.
Der in der *Serial.h* definierte Empfangs-Ringbuffer (*In_Buffer*) wird automatisch von der Interrupt-Routine beschrieben.

Get_Buffer

Funktion: Lesen eines Zeichens vom übergebenen Buffer.

BYTE Get_Buffer (struct Buffer *Queue)

*Queue:	Pointer auf eine Struktur vom Typ Buffer (siehe serial.h).
Rückgabewert:	Zeichen im Buffer an der aktuellen Position des Lesezeigers

Mit der Funktion **Get_Buffer()** wird ein Zeichen aus dem Buffer von der Position: *Queue->queue[Queue->read]* gelesen. Der Buffer wird als Ringbuffer verwaltet und der Füllzustand kann über die Funktion *Buffer_Status()* abgefragt werden.

Achtung!
Die *Get_Buffer()* Funktion sollte nur aufgerufen werden, wenn auch ein Zeichen im Ringbuffer vorliegt. Dies kann mit der *Buffer_Status()*-Funktion überprüft werden. Sollte dies nicht beachtet werden, wird eine 0x00 zurückgegeben.

Hinweis:
Um ein Zeichen aus dem Eingangspuffer der seriellen Schnittstelle zu holen, geben Sie für *Queue* die Bezeichnung *In_Buffer* an:

```
zeichen = GetBuffer(&In_Buffer);
```

Einfacher ist es, die Funktion *get_ser()* zu verwenden, die in den meisten Anwendungsfällen die gleiche Funktionalität besitzt.

6.3.7.5 Buffer_Status

Funktion: Lesen des Zustandes des übergebenen Ring-Buffers.

BYTE Buffer_Status (struct Buffer *Queue)

*Queue:	Pointer auf eine Struktur vom Typ Buffer (siehe serial.h).

Rückgabewert: 0 = Puffer ist leer
1 = Puffer ist gefüllt
2 = Puffer ist mindestens halb voll
3 = Puffer ist voll
4 = Puffer ist übergelaufen

Mit der Funktion **Buffer_Status()** wird ermittelt, ob sich ein Zeichen im übergebenen Puffer befindet. Weiterhin kann der Füllstandsgrad ermittelt werden.

Hinweis:
Um den Status des Eingangspuffers der seriellen Schnittstelle zu prüfen, geben Sie für *Queue* die Bezeichnung *In_Buffer* an:

```
zustand = Buffer_Status(&In_Buffer);
```

6.3.7.6 Char_Present_TimeOut

Funktion: Abfrage, ob ein Zeichen im Empfangs-Ringbuffer (*In_Buffer*) innerhalb einer Timeout-Zeit vorliegt.

BYTE Char_Present_TimeOut (BYTE timeout)

timeout: TimeOut-Zeit [0..255] in s zum Empfang eines Zeichen von der seriellen Schnittstelle.

Rückgabewert: 0 = es liegt min. ein Zeichen im (*In_Buffer*)
1 = es wurde kein Zeichen innerhalb der Timeout-Zeit empfangen

Mit dieser Funktion kann geprüft werden, ob ein Zeichen empfangen und im Ringpuffer der seriellen Schnittstelle abgelegt wurde.
Liegt kein Zeichen im Puffer vor, wartet die Routine die über *timeout* angegebene Zeit auf den Eingang eines Zeichens.
Wird die Timeout-Zeit überschritten, so wird das Bit 1 (0000 0010) in der Fehlervariablen ERROR_CODE_SERIAL gesetzt und der Wert 0x01 zurückgegeben.
Diese Funktion kann dann eingesetzt werden, wenn man Zeichen von der seriellen Schnittstelle in einem vorgegebenen Zeitschema (Protokoll) erwartet.

6.3.7.7 get_ser

Funktion: Abfrage, ob ein Zeichen im Empfangs-Ringbuffer (*In_Buffer*) innerhalb einer Timout-Zeit vorliegt und Abholen des Zeichens.

BYTE get_ser (void)

Rückgabewert:	(0 & ERROR_CODE_SERIAL = 0x02) = Timeout (0...255) = Zeichen aus In_Buffer (ser. Schnittstelle)

Die Funktion **get_ser()** wartet eine vorgegebene Zeit (siehe *get_ser_timeout*) auf ein Zeichen und holt dieses Zeichen ab.
Befindet sich bereits ein Zeichen im Puffer, wird dieses sofort ausgelesen, ansonsten wird die eingestellte Wartezeit auf das Eintreffen eines Zeichens gewartet.
Wird die Timeout-Zeit überschritten, so wird das Bit_1 (0000 0010) in der Fehlervariablen ERROR_CODE_SERIAL gesetzt und der Wert 0x00 zurückgegeben.
Die Timeout-Zeit kann bei der Initialisierung mit der *Ser_init()*- oder später mit der *Set_Get_Ser_Timeout()*- Funktion eingestellt werden.

6.3.7.8 put_ser

Funktion: Sendet ein Zeichen über die serielle Schnittstelle.

void put_ser (BYTE send_char)

send_char:	Zeichen, das über die serielle Schnittstelle gesendet werden soll.

Die Funktion **put_ser()** sendet ein Zeichen über die serielle Schnittstelle mit den in *Ser_Init()* eingestellten Parametern.
Wurde in der *Ser_Init()* der Hardware-Handshake eingeschaltet, so wird das Zeichen nur bei CTS=0 gesendet. Bleibt der Zustand CTS=1 während der gesamten Timeout-Zeit (*put_ser_timeout*) bestehen, so wird das Zeichen nicht gesendet und es wird das Bit 0 (0000 0001) in der Fehlervariablen ERROR_CODE_SERIAL gesetzt.
Die Timeout-Zeit *put_ser_timeout* kann mit der *Ser_init()* - Funktion eingestellt werden.

6.3.7.9 read_ser

Funktion: Lesen eines Zeichens welches im Empfangs-Ringbuffer (*In_Buffer*) vorliegt. Das Zeichens wird nicht abgeholt, das heißt der Zeiger auf aktuelle Position im Ringbuffer wird nicht verändert.

BYTE read_ser (void)

Rückgabewert: (0...255) = Zeichen aus In_Buffer (ser. Schnittstelle)

Die Funktion **read_ser()** liest das Zeichen im Empfangsrinbuffer auf das der Lesezeiger verweist. Der Lesezeiger auf den Buffer wird dabei nicht beeinflusst.
Die Funktion gibt keine Aussage über die Aktualität des Zeichens. Das heißt wenn sich laut *Buffer_Status()* kein Zeichen im Buffer befindet, ist das gelesenen Zeichen undefiniert.

6.3.7.10 Read_Error_Code_Serial

Funktion: Lesen der ERROR_CODE_SERIAL Variablen.

BYTE Read_Error_Code_Serial (void)

Rückgabewert: 0x01 = Senden eines Zeichens fehlgeschlagen (nur bei aktivierten Hardwarehandshake möglich)
0x02 = Empfangen eines Zeichens fehlgeschlagen.

Die in den Funktionen *Char_Present_TimeOut()*, *get_ser()* und *put_ser()* auftretenden Fehler werden in der ERROR_CODE_SERIAL Variablen gespeichert. Diese Information kann mit der Funktion *Read_Error_Code_Serial()* ausgelesen werden.
Die Fehler sind bitweise codiert, d.h. es können auch Kombinationen auftreten.
Durch das Lesen wird der Wert von ERROR_CODE_SERIAL nicht beeinflußt. Ein aufgetretener Fehlercode muß ggf. von Hand zurückgesetzt werden.

6.3.8 InOut.lib – Treiber für die Ein- und Ausgänge

Die *In_Out.lib* beinhaltet alle Funktionen zum Initialisieren und zum Einstellen der Portpins des 80C156 Microcontrollers, die den in Abschnitt *3.1.2, „Serielle Schnittstelle“* beschriebenen optoentkoppelten I/O-Ports zugeordnet sind.
Die Definition BYTE steht für „unsigned char“ und die Definition von WORD für „unsigned int“.

Achtung!
Vor der ersten Verwendung einer Funktion aus der *In_Out.lib* muß die Funktion **IO_Init()** einmalig aufgerufen werden. Weiterhin muß die *In_Out.h* in Ihr Projekt eingebunden (included) werden.

Folgende Funktionen werden in der *In_Out.lib* zu Verfügung gestellt:

- IO_Init()
- Read_IO()
- Write_IO()

Weiterhin werden in der *In_Out.h* Bit-Variablen zur Verfügung gestellt, mit denen man die einzelnen Ports direkt steuern kann. Die Portpins sind den optoentkoppelten I/O-Ports wie folgt zugeordnet:

Variablen-Name	Aufdruck am Steckverbinder	Bezeichnung
IN_0	I/O1	INPUT 0
IN_1	I/O2	INPUT 1
IN_2	I/O3	INPUT 2
IN_3	I/O4	INPUT 3
OUT_0	I/O5	OUTPUT 0
OUT_1	I/O6	OUTPUT 1
OUT_2	I/O7	OUTPUT 2
OUT_3	I/O8	OUTPUT 3

6.3.8.1 IO_Init

Funktion: Initialisierung der optoentkoppelten I/O Ports.

void IO_Init (void)

Diese Funktion initialisiert die für die Steuerung der optoentkoppelten I/O Ports notwendigen Pins des Microcontrollers.

Achtung!
Vor der ersten Verwendung einer Funktion aus der *In_Out.lib*, muß die Funktion **IO_Init()** einmalig aufgerufen werden.

6.3.8.2 Read_IO

Funktion: Lesen der Portpins I/O_1 bis I/O_4.

BYTE Read_IO (void)

Rückgabewert: Zustand der Input-Ports als Hexadezimalwert, binär codiert. Format des Rückgabewertes:

Bitwertigkeit *nn*								
	-	-	-	-	I/O 4	I/O 3	I/O 2	I/O 1
***nn* Binär**	-	-	-	-	Bit 3	Bit 2	Bit 1	Bit 0
***nn* dezimal**	-	-	-	-	8	4	2	1
Bit gesetzt = Pegel an Input *m* vorhanden								

Beispiel: nn = Read_IO();
nn = 06_{HEX} = 06_{DEZ} = 0000 0110_{BIN} = 4 + 2 ⇨ High-Pegel an I/O3 und I/O2 vorhanden!

Mit dieser Funktion kann der momentane Status der optoentkoppelten I/O-Eingänge (siehe Abschnitt *3.1.4, "Optoentkoppelte I/O-Ports"*) abgefragt werden.

Hinweis:
In einer speziellen Bestückungsvariante können I/O_5 bis I/O_8 als Eingang verwendet werden. In diesem Fall enthalten Bit_4 bis Bit_7 des Rückgabewerts die zusätzlichen Eingangsinformationen.

6.3.8.3 Write_IO

Funktion: Steuerung der Portpins I/O_5 bis I/O_8.

void Write_IO (BYTE out_byte)

out_byte: Zustand der Transistoren des Optokopplers an den Ausgängen als Hexadezimalwert, binär codiert. Format des Übergabewertes:

Bitwertigkeit *out_byte*				
	I/O8	I/O 7	I/O 6	I/O 5
***out_byte* binär**	Bit 3	Bit 2	Bit 1	Bit 0
***out_byte* dezimal**	8	4	2	1
Bitwert = 1 = Transistor des Optokopplers gesperrt Bitwert = 0 = Transistor des Optokopplers leitend				

Beispiel: Write_IO(out_byte);
out_byte = E_{HEX} = 14_{DEZ} = 1110_{BIN} = ⇨ Transistor an I/O8, I/O7 und I/O6 gesperrt, Transistor an I/O5 leitend.

Mit dieser Funktion können die optoentkoppelten I/O-Ausgänge I/O_5 bis I/O_8 (*siehe Abschnitt 3.1.4, „Optoentkoppelte I/O-Ports*“) geschaltet werden. Dies kann z.B. zur Steuerung von externen Schaltungen benutzt werden.

Hinweis:
Nach dem Einschalten sind die Transistoren der optoentkoppelten I/O-Ausgänge gesperrt (entspricht *out_byte* = F_{HEX}).
Werden die Transistoren der Ausgänge leitend, so ist der Ausgangspin mit den entkoppelten Masseanschlüssen GND des I/O-Steckers verbunden.

6.3.9 I2C.lib – Treiber für die I2C Bausteine

Die *I2C.lib* beinhaltet Funktionen zum Arbeiten mit der I²C-Peripherie auf dem Board (Videobaustein, EEPROM, RTC), sowie I²C-Bausteinen die über den OPTION-PORT (*siehe Abschnitt 0*) angeschlossen werden.
Das I²C-Protokoll wird per Software mit den Portpins P3.3 (SDA) und P3.4 (SCL) des Microcontrollers emuliert.
Die Definition BYTE und BCD8 steht für „unsigned char“ und die Definition von WORD und BCD16 für „unsigned int“.

Achtung!
Vor der ersten Verwendung einer Funktion aus der *I2C.lib* muß die Funktion **I2C_Init()** einmalig aufgerufen werden. Weiterhin muß die *I2C_Bus.h* in Ihr Projekt eingebunden (included) werden.

Folgende Funktionen werden in der *I2C.lib* zu Verfügung gestellt:

- I2CInit ()	I2C_Bus.h
- I2CWrite ()	I2C_Bus.h
- I2CRead ()	I2C_Bus.h
- BCD2INT ()	Misc.h
- INT2BCD ()	Misc.h
- RTC_Test ()	RTC_8564.h
- RTCSetTime ()	RTC_8564.h
- RTCGetTime ()	RTC_8564.h
- RTCSetAlarm ()	RTC_8564.h
- RTCGetAlarm ()	RTC_8564.h
- RTCGetAlarmStatus ()	RTC_8564.h

Die Funktionen der *I2C.lib* sind logisch verschiedenen Header-Dateien (*I2C_Bus.h, I2C_Hard.h, Misc.h* und *RTC_8564.h*) zugeordnet. Einige Funktionen in der *I2C_Bus.h* und in der *I2C_Hard.h* sind „low-level“-Funktionen des I²C Protokolls und werden im weiteren nicht beschrieben.

6.3.9.1 I2C_Init

Funktion: Initialisierung der I²C-Schnittstelle. (I2C_Bus.h)

void I2C_Init (void)

Mit der Funktion **I2C_Init()** wird die I²C-Schnittstelle für einen Master/Slave-Betrieb initialisiert, wobei der C165 Microcontroller der Master ist und alle an das I²C Port (SCL, SDA) angeschlossenen Geräte als Slave betrachten werden.

Achtung!
Vor der ersten Verwendung einer Funktion aus der *I2C.lib* muß die **I2C_Init()** einmalig aufgerufen werden.
Beachten Sie beim Anschluß externer I²C-Geräte, daß kein Adreßkonflikt mit den internen I²C-Geräten entsteht *(siehe Abschnitt 3.1.5).* In diesem Fall werden die Geräte nicht richtig funktionieren.

6.3.9.2 I2CWrite

Funktion: Schreibt ein oder mehrere Bytes an einen ausgewählten Baustein welcher am I²C Bus angeschlossen ist. (I2C_Bus.h)

BYTE I2CWrite (BYTE *SourcePtr, BYTE DeviceID,
BYTE DestAddr, WORD Size)

*SourcePtr:	Adresse eines Speicherbereichs, der byteweise in das I²C-Gerät geschrieben werden soll.
DeviceID:	8-Bit Device-ID des I²C-Geräts.
DestAddr:	8-Bit Zieladresse im I²C-Gerät.
Size:	Anzahl der Bytes, die in das I²C-Gerät geschrieben werden sollen.
Rückgabewert:	0x00 = Schreiben OK 0xFE = Schreiben nicht erfolgreich 0xFF = Kein I²C Gerät gefunden

Mit der Funktion **I2CWrite()** können 1-Byte bis 65535-Byte große Speicherbereiche vom Microcontroller in das I²C-Device geschrieben werden.

Es werden dabei die in *Size* festgelegte Anzahl von Bytes vom Speicherbereich des **SourcePtr* an den Speicherbereich *DestAddr* des im *DeviceID* spezifizierten I²C-Devices geschrieben.

Achtung!
Niemals Schreibzugriffe direkt nacheinander, sondern mit einem Delay von ca. 3 ms, ausführen.
Sollen Variablen vom Typ char, int oder long an ein I²C-Gerät geschrieben werden, beachten Sie, das die Funktion einen Pointer vom Typ BYTE erwartet.

6.3.9.3 I2CRead

Funktion: Lesen ein oder mehrere Bytes von einem ausgewählten Baustein, welcher am I²C Bus angeschlossen ist. (I2C_Bus.h)

BYTE I2CRead (BYTE *DestPtr, BYTE DeviceID, BYTE SourceAddr, WORD Size)

*DestPtr:	Adresse eines Speicherbereichs, der byteweise mit dem Inhalt des I²C-Gerät beschrieben werden soll
DeviceID:	8-Bit Device-ID des I²C-Geräts
SourceAddr:	8-Bit Leseadresse im I²C-Gerät
Size:	Anzahl der Bytes, die aus dem I²C-Gerät gelesen werden sollen
Rückgabewert:	0x00 = Lesen OK 0xFE = Lesen nicht erfolgreich 0xFF = Kein I²C Gerät gefunden

Mit der Funktion **I2CRead()** können 1-Byte bis 65535-Byte große Speicherbereiche vom I²C-Device in den Microcontrollerspeicher gelesen werden.
Es werden dabei die in *Size* festgelegte Anzahl von Bytes in den Speicherbereich des **DestPtr* vom Speicherbereich *SourceAddr* des im *DeviceID* spezifizierten I²C-Device gelesen.

Achtung!
Sollen Variablen vom Typ char, int oder long von einem I²C-Gerät gelesen werden, beachten Sie, daß die Funktion einen Pointer vom Typ BYTE erwartet.

6.3.9.4 BCD2INT

Funktion: Konvertierung eines BCD-Werts in eine Integer Zahl. (Misc.h)

int BCD2INT(BCD16 BCDValue)

BCDValue: 16-Bit (unsigned int) BCD Wert zur Konvertierung.

Rückgabewert: konvertierter BCD-Wert als Integer Zahl

Mit der Funktion **BCD2INT()** können 16-Bit BCD-Werte in eine Integerzahl umgewandelt werden. Dies wird benötigt, um Werte, die im BCD-Format vorliegen, zum Darstellen oder für mathematische Operationen aufzubereiten.

Beispiel RTC (Realtime-Clock):

BCD Wert in RTC für Sekunden		entspricht Hex-Wert	entspricht Dezimal-Wert	Dezimal-Wert nach Konvertierung BCD2INT
5	5	0x55 hex	85 dez	55 dez
Zehnerstelle	Einerstelle			

Die Werte in der RTC liegen im BCD-Format vor. Somit sind 55 Sekunden als $0x55_{hex}$ bzw. 85_{dez} abgelegt. Wenn man die Sekunden als dezimalen Wert benötigt, muß die $0x55_{hex}$ bzw. 85_{dez} mit der BCD2INT Funktion in eine 55_{dez} umgewandelt werden.

6.3.9.5 INT2BCD

Funktion: Konvertierung einer Integer Zahl in einen BCD-Werte. (Misc.h)

BCD16 INT2BCD(int INTValue)

INTValue: 16-Bit Integer Wert zur Konvertierung.

Rückgabewert: konvertierte Integer Zahl als (unsigned int) BCD-Wert

Mit der Funktion **INT2BCD()** werden Integer-Zahlen in einen 16-Bit BCD-Wert umgewandelt. Dies wird benötigt, da einige Geräte Zahlen im BCD-Format erwarten.

Beispiel RTC:

Dezimalwert in Sekunden	entspricht Hex-Wert	Hex-Wert nach Konvertierung INT2BCD	BCD Wert in RTC für Sekunden	
55 dez	0x37 hex	0x55 hex	5	5
			Zehnerstelle	Einerstelle

Die Werte in der RTC werden im BCD-Format erwartet. Es soll nun der Wert 55_{dez} , der 55 Sekunden entspricht, in der RTC abgelegt werden. Eine 55_{dez} entspricht aber einer $0x37_{hex}$, deshalb muß die 55_{dez} mit der Funktion INT2BCD() in eine $0x55_{hex}$ umgewandelt werden.

6.3.9.6 RTC_Test

Funktion: Testen, ob die RTC (Echtzeituhr) vorhanden und betriebsbereit ist. (RTC_8564.h)

BYTE RTC_Test (BYTE DeviceID)

DeviceID:	8-Bit DeviceID des I²C-Geräts (RTC 8564).
Rückgabewert:	0x00 = Lesen OK 0xFF = Kein I²C Gerät gefunden

Mit der Funktion **RTC_Test()** wird ein Lesezugriff auf das Control_1 Register der RTC durchgeführt. Wenn nach 3 s kein gültiger Lesezugriff durchgeführt werden konnte, wird die Funktion mit einem Fehlercode abgebrochen.

Achtung!

- Da die RTC nach dem Einschalten erst nach etwa 3 s einsatzbereit ist, sollte diese Funktion vor dem ersten Zugriff auf die RTC durchgeführt werden.
- Die Device-ID der Echtzeituhr können Sie aus den Technischen Daten (*Abschnitt 5.3*) entnehmen. Sie können auch das in der Header-Datei vordefinierte Symbol *RTC8564_ID* verwenden.

6.3.9.7 RTCSetTime

Funktion: Setzen von Datum und Uhrzeit in der RTC (RTC_8564.h)

BYTE RTCSetTime (REAL_TIME *RealTime, BYTE DeviceID)

*RealTime:	Zeiger auf Struktur vom Typ REAL_TIME:
	{ BCD8 Second;
	BCD8 Minute;
	BCD8 Hour;
	BCD8 Day;
	BCD8 Weekday; // day of week (sun=0 to sat=6)
	BCD8 Month;
	BCD16 Year; }

DeviceID:	8-Bit DeviceID des I²C-Geräts (RTC 8564).
Rückgabewert:	0x00 = Aktion OK 0xFE = Schreiben nicht erfolgreich

Mit der Funktion **RTCSetTime()** werden die Daten einer Struktur (von der Adresse des Zeigers **RealTime*) für Datum und Uhrzeit in die RTC geschrieben.

Bsp:
```
REAL_TIME Zeit;
Zeit.Hour    = 0x23;
Zeit.Minute  = 0x58;
Zeit.Second  = 0x55;.
Zeit.Weekday = THURSDAY;
Zeit.Day     = 0x31;
Zeit.Month   = 0x12;
Zeit.Year    = 0x2099;
RTCSetTime  (&Zeit, RTC8564_ID);
```
setzt Uhrzeit = 23:58:55 Uhr und Datum = Do. der 31.12.2099.

Achtung!
Die Wochentage sind numerisch sortiert, beginnend mit Sonntag:

SUNDAY	= 0
MONDAY	= 1
TUESDAY	= 2
WEDNESDAY	= 3
THURSDAY	= 4
FRIDAY	= 5
SATURDAY	= 6

6.3.9.8 RTCGetTime

Funktion: Lesen von Datum und Uhrzeit aus der RTC (RTC_8564.h)

BYTE RTCGetTime (REAL_TIME *RealTime, BYTE DeviceID)

*RealTime: Zeiger auf Struktur vom Typ REAL_TIME:

```
{ BCD8  Second;
  BCD8  Minute;
  BCD8  Hour;
  BCD8  Day;
  BCD8  Weekday;        // day of week (sun=0 to sat=6)
  BCD8  Month;
  BCD16 Year; }
```

DeviceID: 8-Bit DeviceID des I²C-Gerätes (RTC 8564)

Rückgabewert: 0x00 = Aktion OK
0xFF = Lesen nicht erfolgreich

Mit der Funktion **RTCGetTime()** werden die aktuellen Daten für Datum und Uhrzeit aus der RTC gelesen und in der Struktur (an der Adresse des Zeigers **RealTime*) abgelegt.

Bsp.:
```
REAL_TIME Zeit;
RTCGetTime  (&Zeit, RTC8564_ID);
```

Wenn in den Variablen folgende Werte stehen:

Zeit.Hour = 0x23,
Zeit.Minute = 0x58,
Zeit.Second = 0x55,.
Zeit.Weekday = THURSDAY,
Zeit.Day = 0x31,
Zeit.Month = 0x12,
Zeit.Year = 0x2099,

entspricht dies einer Uhrzeit von 23:58:55 Uhr und dem Datum – Do. der 31.12.2099.

6.3.9.9 RTCSetAlarm

Funktion: Setzen von Alarm-Tag und Alarm-Uhrzeit der RTC. (RTC_8564.h)

BYTE RTCSetAlarm (ALARM_TIME *AlarmTime,
BYTE EnableExtInt, BYTE DeviceID)

*AlarmTime: Zeiger auf Struktur vom Typ ALARM_TIME:

```
{  BYTE  Minute_Ignore;
   BCD8  Minute;
   BYTE  Hour_Ignore;
   BCD8  Hour;
   BYTE  Day_Ignore;
   BCD8  Day;
   BYTE  Weekday_Ignore;
   BCD8  Weekday; // sun = 0 to sat = 6
}
```

EnableExtInt: Aktivieren bzw. Deaktivieren des externen Interrupts von der RTC 8564 am Portpin 10 (/INT).
0 = deaktiviert, 1= aktiviert

DeviceID: 8-Bit Device-ID des I²C-Gerätes (RTC 8564).

Rückgabewert: 0x00 = Aktion OK
0xFE = Schreiben nicht erfolgreich
0xFF = Lesen nicht erfolgreich

Mit der Funktion **RTCSetAlarm()** werden die Daten einer Struktur (von der Adresse des Zeigers **AlarmTime*) für Alarm-Tag und Alarm-Uhrzeit in die RTC geschrieben. Durch die Variablen Minute_Ignore, Hour_Ignore, Day_Ignore und Weekday_Ignore kann die Gültigkeit der einzelnen Zeitangaben beeinflußt werden.

Beispiel:

- Day_Ignore = 0: Alarm wird am eingestellten Alarm-Tag ausgelöst
- Day_Ignore = 1: Alarm wird unabhängig vom eingestellten Alarm-Tag ausgelöst.

Weiterhin kann über die Variable *EnableExtInt* der Interrupt-Pin 10 (/INT) der RTC aktiviert oder deaktiviert werden. Ist die Funktion aktiviert, so wird beim Setzen des Timer-Flags (TF, *siehe Control2-Register in RTC8564*) oder des Alarm-Flags (AF, *siehe Control2-Register in RTC8564*) ein Interrupt ausgelöst.

Bsp.:
```
ALARM_TIME AlarmZeit;
AlarmZeit.Hour              = 0x23;
AlarmZeit.Hour_Ignore       = 0;
AlarmZeit.Minute            = 0x58;
AlarmZeit.Minute_Ignore     = 0;
AlarmZeit.Second            = 0x55;
AlarmZeit.Second_Ignore     = 0;
AlarmZeit.Weekday           = THURSDAY;
AlarmZeit.Weekday_Ignore    = 0;
AlarmZeit.Day               = 0x31;
AlarmZeit.Day_Ignore        = 0;
RTCSetAlarmTime  (&AlarmZeit,0, RTC8564_ID);
```

setzt die Alarmzeit auf 23:58:55 Uhr am Do. den 31. Alle Alarm-Zeiteinstellungen sind gültig und der externe Interrupt ist nicht aktiviert.

Achtung!
Damit die RTC einen Interrupt auslösen kann, muß der Jumper J7 auf dem grabbMODUL geschlossen werden (*siehe Abschnitt 5.2)*

6.3.9.10 RTCGetAlarm

Funktion: Lesen von Alarm-Tag und Alarm-Uhrzeit aus der RTC. (RTC_8564.h)

BYTE RTCGetAlarm (ALARM_TIME *AlarmTime,
BYTE DeviceID)

*AlarmTime: Zeiger auf Struktur vom Typ ALARM_TIME.

```
{  BYTE  Minute_Ignore;
   BCD8  Minute;
   BYTE  Hour_Ignore;
   BCD8  Hour;
   BYTE  Day_Ignore;
   BCD8  Day;
   BYTE  Weekday_Ignore;
   BCD8  Weekday;                  // sun = 0 to sat = 6
}
```

DeviceID: 8-Bit DeviceID des I²C-Geräts (RTC 8564).

Rückgabewert: 0x00 = Aktion OK
0xFF = Lesen nicht erfolgreich

Mit der Funktion **RTCGetAlarm()** werden die aktuellen Daten für Alarm-Tag und Alarm-Uhrzeit aus der RTC gelesen und in der Struktur (an der Adresse des Zeigers **AlarmTime*) abgelegt.
Die Variablen Minute_Ignore, Hour_Ignore, Day_Ignore und Weekday_Ignore enthalten Informationen zur Gültigkeit der einzelnen Zeitangaben.
Bsp.:

- Day_Ignore = 0, Alarm wird am eingestellten Alarm-Tag ausgelöst
- Day_Ignore = 1, Alarm wird unabhängig vom eingestellten Alarm-Tag ausgelöst.

Bsp.: ALARM_TIME AlarmZeit;
RTCGetAlarmTime (&AlarmZeit, RTC8564_ID);

Wenn in den Variablen folgende Werte stehen:

AlarmZeit.Hour	= 0x23,
AlarmZeit.Hour_Ignore	= 0,
AlarmZeit.Minute	= 0x58,
AlarmZeit.Minute_Ignore	= 0,
AlarmZeit.Second	= 0x55,
AlarmZeit.Second_Ignore	= 0,
AlarmZeit.Weekday	= THURSDAY,
AlarmZeit.Weekday_Ignore	= 0,
AlarmZeit.Day	= 0x31,
AlarmZeit.Day_Ignore	= 0,

wird ein Alarm am Donnerstag dem 31. um 23:58:55 Uhr ausgelöst.

6.3.9.11 RTCGetAlarmStatus

Funktion: Mit dieser Funktion kann der aktuelle Zustand des Alarm-Flags in der RTC gelesen und zurückgesetzt werden. (RTC_8564.h)

BYTE RTCGetAlarmStatus (BYTE ResetAlarm,
BYTE DeviceID)

ResetAlarm:	0 = AF Flag wird nach dem Lesen nicht zurückgesetzt 1 = AF Flag wird nach dem Lesen zurückgesetzt
DeviceID:	8-Bit DeviceID des I²C-Gerätes (RTC 8564).
Rückgabewert:	0x00 = AF-Flag ist nicht gesetzt 0x01 = AF-Flag ist gesetzt 0xFE = Schreiben nicht erfolgreich 0xFF = Lesen nicht erfolgreich

Mit der Funktion **RTCGetAlarmStatus()** wird der Zustand des Alarm-Flag (AF, siehe Control2-Register der RTC8564) gelesen und das Ergebnis zurückgegeben. Weiterhin kann mit der Funktion das Alarm-Flag nach dem Lesen zurückgesetzt werden.

6.4 Arbeiten mit Phytec-Firmware

6.4.1 LOCAL_COM: Firmware zur Lokalen Bildübertragung

Mit der Firmware LOCAL_COM stellt PHYTEC Ihnen eine fertige Modulsoftware zur Verfügung, mit der Sie über die serielle Schnittstelle von einem Host-Rechner (z.B. einem PC) das grabbMODUL-4 steuern und Bilddaten vom Modul übertragen können (*Bild 55*).
Über die optoentkoppelten Eingänge können außerdem Bildaufnahmen durch externe Ereignisse ausgelöst werden (bis zu vier Eingänge können vier Kameras bedienen). Der Zeitpunkt der Auslösung kann durch einen Zeitstempel festgestellt werden.

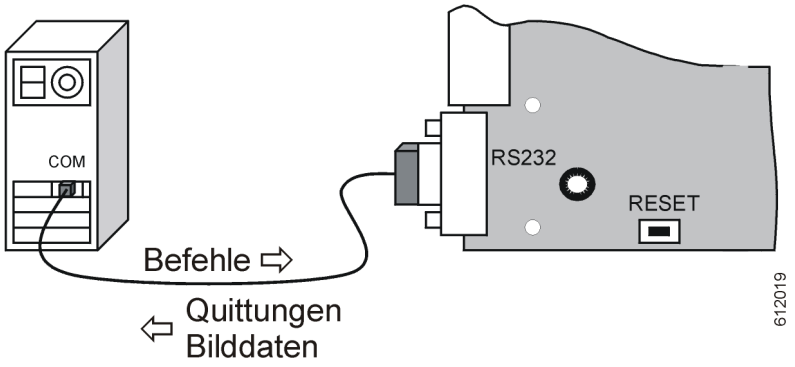


Bild 55: Gerätekonfiguration für LOCAL_COM - Firmware

Bitte beachten Sie:

- Auf dem grabbMODUL-4 muß die Firmware LOCAL_COM installiert sein. Dazu müssen Sie die Datei *local_com_Vxx.h86* mit Hilfe der PHYTEC-FlashTools in den Flash Speicher des Moduls programmieren. (*xx* gibt die Versionsnummer an.). Im Auslieferungszustand befindet sich eine aktuelle Version der Software „LOCAL_COM“ im Flash Speicher des grabbMODUL-4.
- grabbMODUL-4 und Host-Rechner müssen in der Anwendungsumgebung über ein *Nullmodem*-Kabel verbunden werden. Bei einem Nullmodem-Kabel sind die Signalleitungen intern gekreuzt. Benutzen Sie nach Möglichkeit ein fertig konfektioniertes Nullmodem-Kabel, um Fehler auszuschließen. *Informationen zum Aufbau eines Nullmodem-Kabels finden Sie auf den FAQ-Seiten unserer Homepage.*

- Sie müssen für die Host-Seite (i.a. ist das ein PC) eine Anwendungssoftware schreiben, die das grabbModul-4 über die serielle Schnittstelle steuert, die Bilddaten von der Schnittstelle abholt und verarbeitet.
  Diese Software können Sie beliebig gestalten. Im Folgenden finden Sie eine Beschreibung der Kommandos, die Sie über die Schnittstelle zum Modul senden können und das Format der Datenübertragung.
- Die Software kann nach dem Download in das grabbMODUL-4 mittels einem Terminal-Programm (z.B. HyperTerminal) oder einer Windows-Software „PC_Vxx.exe“ getestet werden. Die Vorgehensweise ist im Abschnitt *2.5, „Testen des Modul-Programms LOCAL_COM“* beschrieben.

6.4.2 Hinweise zum Funktionsablauf der Firmware

Die Software auf dem grabbMODUL-4 initialisiert die serielle Schnittstelle mit 115200 Baud, 8 Datenbits, keine Parität, ein Stoppbit und kein Protokoll. Danach wartet die Software auf ein Zeichen von der seriellen Schnittstelle, um dann die entsprechende Funktion auszuführen. Wird vom Host-Rechener eine gültige Protokoll-Sequenz eingeleitet (z.B.: S), erwartet das Modul die folgenden Zeichen. Werden die einzelnen Zeichen nicht innerhalb von jeweils 3 s (voreingestellt) vom Host gesendet, so geht das Modul von einer Störung auf der seriellen Schnittstelle aus. In diesem Fall sendet das Modul eine Fehler-Quittung (z.B.: q,S,1#), setzt das Fehlerbit (0x08 „received character failed“) im Fehler-Status-Register und wartet auf eine neue gültige Protokoll-Sequenz.
Wird eine Protokollsequenz vollständig empfangen, antwortet das grabbMODUL-4 mit einer Verzögerung von ca. 10-500ms. Eine größere Verzögerung kann bei den Befehlen „G“ (bis zu 3 s) und „V“ (bis zu 5 s) auftreten.
Wird während des Wartens auf eine Protokoll-Sequenz ein Flankenwechsel an einem der Input-Pins erkannt, so wird von der Software ein Bild vom dazugehörigen Kanal (Input 1 = Kanal 1, ...) gegrabbt. Die Bildgröße entspricht dabei den mit dem Parameter „P“ eingestellten Werten (default: 180 x 144).

6.4.3 Übersicht über Befehle und Befehlsaufbau

Befehle und Quittungen werden als ASCII-Codes übertragen. Ein Befehl besteht aus einem einleitenden Befehlsbuchstaben, gefolgt von gegebenenfalls erforderlichen Parametern. Als Trennzeichen zwischen Befehlsbuchstaben und Parametern wird ein Kommazeichen „,“ gesendet. Jeder Befehl wird mit einem Nummernzeichen „#“ abgeschlossen.
Quittungen werden mit dem Quittungsbuchstaben Q/q eingeleitet. Als Parameter wird eine Fehlernummer übertragen. Fehlernummer 0 bedeutet „kein Fehler“. Die Quittung wird mit dem Nummernzeichen „#“ abgeschlossen.

Folgende Befehle sind in der Firmware umgesetzt:

- **C** - Eingangs-Status anfordern
- **E** - Reserviert
- **F** - Fehler-Status anfordern
- **G** - Kamerabild von Kanal k grabben
- **I** - Bilddaten anfordern
- **J** - Reserviert
- **L** - Reserviert
- **O** - Output-Status setzen
- **P** - Bildgröße und –skalierung einstellen
- **R** - Einheit zurücksetzen
- **S** - Software-ID abfragen
- **T** - Zeitstempel lesen / rücksetzen
- **V** - Kamera-Status anfordern
- **X** - Reserviert
- **Z** - Empfangs-Timeout der Schnittstelle setzen
- **0x0D** - Reserviert
- **0x0A** - Reserviert

6.4.4 Software-ID abfragen

Befehlscode:	S#	
Parameter:	keine	
Beispiel:	S#	
Quittung:	q,S,0#s,*xx*,*yy*# =	Bestätigung, Versionscode xx.yy z.B. V1.02= 01.02
	q,S,1# = Fehler	

6.4.5 Einheit zurücksetzen

Befehlscode:	R,*x*#
Parameter:	1 = Software-Reset durchführen alle anderen: Befehlsabbruch
Beispiel:	R,1#
Quittung:	q,R,0# = Bestätigung, Grundzustand wird hergestellt q,R,1# = Parameterfehler, kein Rücksetzen

Bemerkung: Durch einen Software-Reset wird das grabbMODUL-4 wieder in den Grundzustand zurückversetzt. Alle eingestellten Werte gehen dabei verloren. Nach dem Rücksetzen müssen diese Werte daher wieder neu eingestellt werden.

6.4.6 Kamera-Status anfordern

Befehlscode: V#
Parameter: keine
Beispiel: V#
Quittung: `q,V,0#v,`*nn*`#` = Bestätigung, Status wird gesendet:
nn = Kamerastatus als Hexadezimalwert, binär codiert
`q,V,1#` = Parameterfehler

Bemerkung: Zwischen `q,V,0#` und `v,nn#` können bis zu 5 sec. Verarbeitungszeit liegen.

Format der Quittung:

Bitwertigkeit *nn*								
	-	-	-	-	I/O 4	I/O 3	I/O 2	I/O 1
nn Binär	-	-	-	-	Bit 3	Bit 2	Bit 1	Bit 0
nn dezimal	-	-	-	-	8	4	2	1
Bit gesetzt = Videosignal an Kanal *m* vorhanden								

Beispiel: Rückgesendete Quittung = `q,V,0#v,0A#`
nn = $0A_{HEX}$ = 10_{DEZ} = 0000 1010_{BIN} = 8 + 2 ⇨ Signal an Kanälen 4 und 2 vorhanden.

6.4.7 Kamerabild von Kanal k grabben

Befehlscode: `G,`*kk#*
Parameter: *kk* = Kanalnummer, dezimal 00...04
Beispiel: `G,02#`
Quittung: `q,G,0#` = Bestätigung
`q,G,1#` = Parameterfehler

Kanalnummer = 0: Es wird das aktuelle Bild vom zuletzt eingestellten Kanal gegrabbt. Da die Kanaleinstellung auch über ein externes Ereignis erfolgt, wird zum Beispiel nach einem Signalwechsel an Input 2 (=Kamera 2) das Bild von Kanal 2 neu gegrabbt.

Kanalnummer = 1..4: Es wird das aktuelle Bild vom übergebenden Kanal gegrabbt.

Bemerkung: Zwischen `q,G,` und `0#` können bis zu 3 sec. Verarbeitungszeit liegen. In dieser Zeit wird das grabbMODUL-4 auf den richtigen Eingangskanal eingestellt und das Bild wird digitalisiert. Die defaultmäßig eingestellte Bildgröße beträgt 180x144 Pixel. Die Bildgröße kann mit dem Befehl `P` verändert werden.

6.4.8 Bilddaten anfordern

Befehlscode: `I,`*ooooo*`,`*ss*`,`*nnnn*`,`*m*`#`
Parameter: *ooooo* = Offset im Bildspeicher, hexadezimal, 5-stellig
ss = Größe eines Datenblocks, hex, 2-stellig
nnnn = Anzahl der Datenblöcke, hex
m = Modus:
-- Graustufen-Bilder --
1 = Low-Nibbles senden
2 = High-Nibbles senden
3 = ganze Bytes senden
-- Farbbilder --
4 = Farbdaten YCrCb 4:2:2 senden
(**Beachte: Im Modus 4 sind nur gradzahlige Blockgrößen möglich!)**
5 = Reserviert (Preview 90x72)
6 = Reserviert (Preview 45x36)
7 = nur Farbauszüge CrCb senden
8 = Reserviert (JPG Bildddaten)

Beispiele:

(a) Bild im Speicher 180 x 144 Pixel, Übertragung als Byte (256 Graustufen),
maximale Blockgröße benutzen:
`I,00000,FF,0066,3#`
von der Modulseite werden dann 101 Blöcke à 255 Pixel gesendet und 1 Block mit 165 Pixeln
(101 x 255 + 165 = 180 x 144)
Bem.: 102 = 0x66

(b) Bild im Speicher 180 x 144 Pixel, Übertragung der oberen 4-Bit jedes Pixels (gepackt), um Datenmenge zu halbieren:
`I,00000,FF,0033,2#`
von der Modulseite werden dann 50 Blöcke à 255 Byte gesendet und 1 Block mit 210 Pixeln
(50 x 255 + 210 = 180 x 144 : 2)
Anschließend kann die Graustufenauflösung ergänzt werden, indem der Befehl
`I,00000,FF,0033,1#`
gesendet wird.

(c) Bild im Speicher 180 x 144 Pixel, Übertragung als vollständiges Farbbild, maximale Blockgröße benutzen:
`I,00000,FE,00CD,4#`
von der Modulseite werden dann 204 Blöcke à 254 Byte gesendet und 1 Block mit 24 Pixeln (204 x 254 + 24 = 180 x 144 x 2)
Bem.: 205 = 0xCD

(d) Bild im Speicher 180 x 144 Pixel, Übertragung des Farbauszugs CrCb, maximale Blockgröße benutzen:
`I,00000,FF,0066,7#`
von der Modulseite werden dann 101 Blöcke à 255 Pixel gesendet und 1 Block mit 165 Pixeln (101 x 255 + 165 = 180 x 144)
Bem.: 102 = 0x66

Quittung:
`q,I,0#` = OK, Sendung folgt
`q,I,1#` = Parameterfehler
`q,I,2#` = ungültiger Wertbereich eines Parameters

Nach der Quittung „q,I,0#“ erfolgt die Sendung der Bilddaten vom grabbMODUL-4 wie sie im Abschnitt *6.4.9, „Bilddatenformat*“ beschrieben ist.
Diese Funktion überträgt die Bilddaten aus dem Video-RAM des Moduls über die serielle Schnittstelle. Da die Datenübertragung einen hohen Zeitbedarf haben kann, gibt es verschiedene Modi, mit denen sich die Datenmenge flexibel handhaben läßt.

Achtung!
In Modus 4 muß die Blockgröße (Parameter *ss*) gerade sein!

Hinweise:
Mit dieser Funktion wird kein Bild aufgenommen, sondern das Bild, das sich im Speicher befindet, übertragen. Zur Aufnahme eines Bilds muß der Befehl G verwendet werden oder ein Alarmsignal an den Alarmeingängen eingehen.

Die Bilddaten werden so übertragen, wie sie im Bildspeicher stehen. Das bedeutet, daß bei Auflösungen größer als 288 Zeilen die beiden Halbbilder getrennt nacheinander übertragen werden (*siehe Abschnitt 6.2.1*). Dies hat den Vorteil, daß bei der Darstellung der Daten auf dem Bildschirm schnell ein Bild mit der halben Zeilenauflösung dargestellt werden kann, indem man zunächst alle empfangenen Zeilen doppelt darstellt. Mit Übertragung des zweiten Halbbilds kann dieses in den geraden Bildzeilen eingefügt werden. Die ursprüngliche Information dieser Zeilen wird dadurch überschrieben.
Für den Betrachter ergibt sich der Eindruck eines progressiv aufgebauten Bildes.
Details zum Format der übertragenen Bilddaten finden Sie in Abschnitt 6.4.9.

Der Standardmodus 3 bewirkt die Übertragung von Graustufen-Bildern („schwarz/weiß“-Bilder). Die Bilddaten werden zeilenweise (von links oben nach rechts unten) übertragen.

Zur Beschleunigung des Bildaufbaus können die Nibble-Modi (Modus 1 und 2) verwendet werden. Das Funktionsprinzip des Nibble-Modes beruht auf einer anderen Übertragung der Helligkeitsinformation. Dazu wird die Helligkeitsinformation jedes Pixels zunächst nur grob übertragen (16 Graustufen). Da dadurch die Informationsmenge auf 4-Bit pro Pixel sinkt, können zwei Pixel in ein Byte gepackt werden und die Bildübertragung ist doppelt so schnell. In einem zweiten Durchlauf kann auf die gleiche Weise die andere Hälfte der Helligkeitsinformation übertragen werden, wodurch die Graustufenauflösung von 8-Bit (256 Stufen) wieder erreicht wird.

Die gesamte Übertragungsdauer entspricht der Dauer der normalen Graustufen-Übertragung (Mode 3). Der Vorteil besteht jedoch darin, daß ein vollständiges Vorschaubild bereits in der Hälfte der Zeit verfügbar ist.

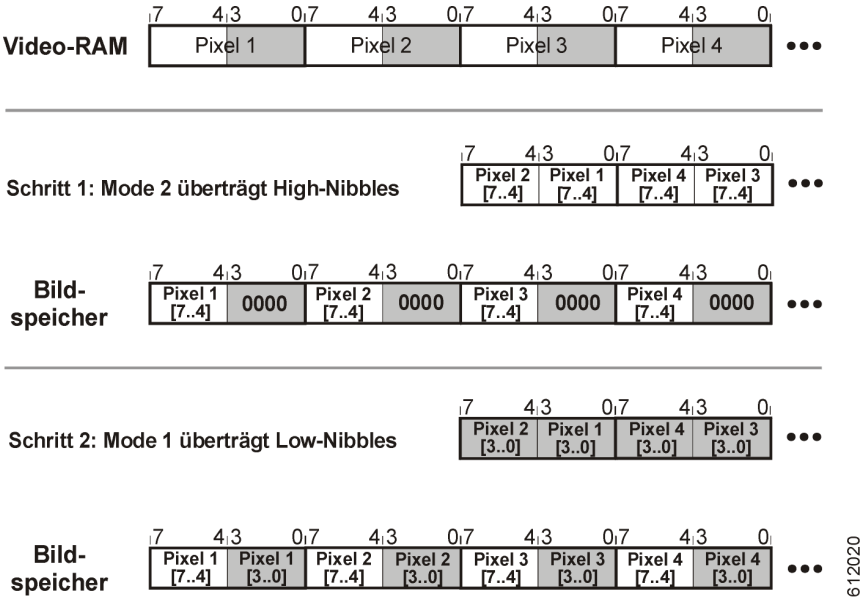


Bild 56: Datenübertragung im Nibble-Mode

Für die Übertragung von Farbbildern sind zwei Modi verfügbar. Modus 4 überträgt ein vollständiges Farbbild im YCrCb-4:2:2-Format. Die Daten werden dabei sequentiell übertragen, d.h. Y_1, $Cb_{1/2}$, Y_2, $Cr_{1/2}$ usw. Zur Darstellung müssen die Daten in der Regel in das RGB-Format überführt werden (*siehe dazu Abschnitt 6.2.2*).
Da hier zwei Bytes logisch zusammenhängen, muß immer eine gerade Anzahl von Bytes pro Block angefordert werden (d.h. der Parameter *ss* muß gerade sein).
Modus 7 überträgt lediglich die Farbauszüge Cr/Cb des Bilds. Die Farbinformation wird wie folgt übertragen: $Cb_{1/2}$, $Cr_{1/2}$, $Cb_{3/4}$, $Cr_{3/4}$ usw.
Auch dieser Modus wurde eingeführt, um eine beschleunigte Vorschau des Bilds zu ermöglichen. Hier kann in einem ersten Schritt mit dem Modus 3 ein Graustufenbild zur Vorschau übertragen werden. Im zweiten Schritt kann mit Modus 7 die Farbinformation geholt und in die schon vorhandene Graubild-Information eingerechnet werden. Das Bild wird quasi in diesem Schritt nachcoloriert.
Denkbar ist natürlich auch eine Kombination der Modi 2,1 und 7.

6.4.9 Bilddatenformat

Bilddaten werden im Binärformat übertragen. Die Übertragung wird mit der Kommandosequenz `i`, eingeleitet. Anschließend werden die Bilddaten in Blöcken gemäß der Anforderung übertragen. Zu Beginn eines jeden Blocks wird die tatsächliche Blockgröße als zweistellige Hex-Zahl im ASCII-Code übertragen. Es folgt dann die spezifizierte Anzahl Datenbytes in binärer Form. Anschließend wird eine Checksumme (binär) über die binären Datenbytes gesendet. Die Checksumme wird auf 00 ergänzt. Direkt anschließend beginnt die Übertragung des nächsten Blocks. Ein kompletter Datenblock ist also folgendermaßen aufgebaut:

i, $ll d_1 d_2 d_3 d_4 d_5 ..... d_{//} c$ [Quittung] $ll d_1 d_2 d_3 d_4 d_5 ..... d_{//} c$ [Quittung] ...

ll = Anzahl tatsächlicher Datenbytes im Block (hex)
d_n = Datenbyte (binär)
c = Checksumme (binär)

Die Checksumme „c“ wird für jeden Block ermittelt und am Ende des Blocks übertragen. Die Checksumme wird durch die Aufsummierung aller Daten ($d_1+d_2+d_3+d_4+d_5+.....d_{//}$) berechnet und es wird dann das low-Byte des Ergebnisses auf 0x0100 ergänzt übertragen.
Beispiel: Es werden 2 Datenbytes d1 = 120 und d2 = 160 in einem Block gesendet. Daraus folgt für c $\Rightarrow$ 120+160 = 280 = 00000001 00011000b $\Rightarrow$ low-Byte = 00011000b = 0x18 =24 $\Rightarrow$ Ergänzt auf 0x0100 $\Rightarrow$ 0x0100 – 0x18 = 0xE8 = 232 = c.

Die Übertragung endet, wenn die spezifizierte Anzahl Blöcke übertragen ist. Werden Daten außerhalb des Bilddatenspeichers angefordert, so wird die Sendeanforderung mit einer Fehlermeldung quittiert.

Nach jedem Block wartet die Einheit auf eine Quittung des Empfängers:

Quittung des Empfängers (Host-Rechner)

`q,i,0#` = OK, das Modul sendet dann den nächsten Datenblock
`q,i,1#` = Fehler / die Sendung wird vom Modul abgebrochen

Wird keine gültige Quittung empfangen, wird die Sendung abgebrochen. Wird nach einer bestimmten Timeout-Zeit (Voreingestellt sind 3 sec.) keine Quittung empfangen, wird die Sendung abgebrochen.

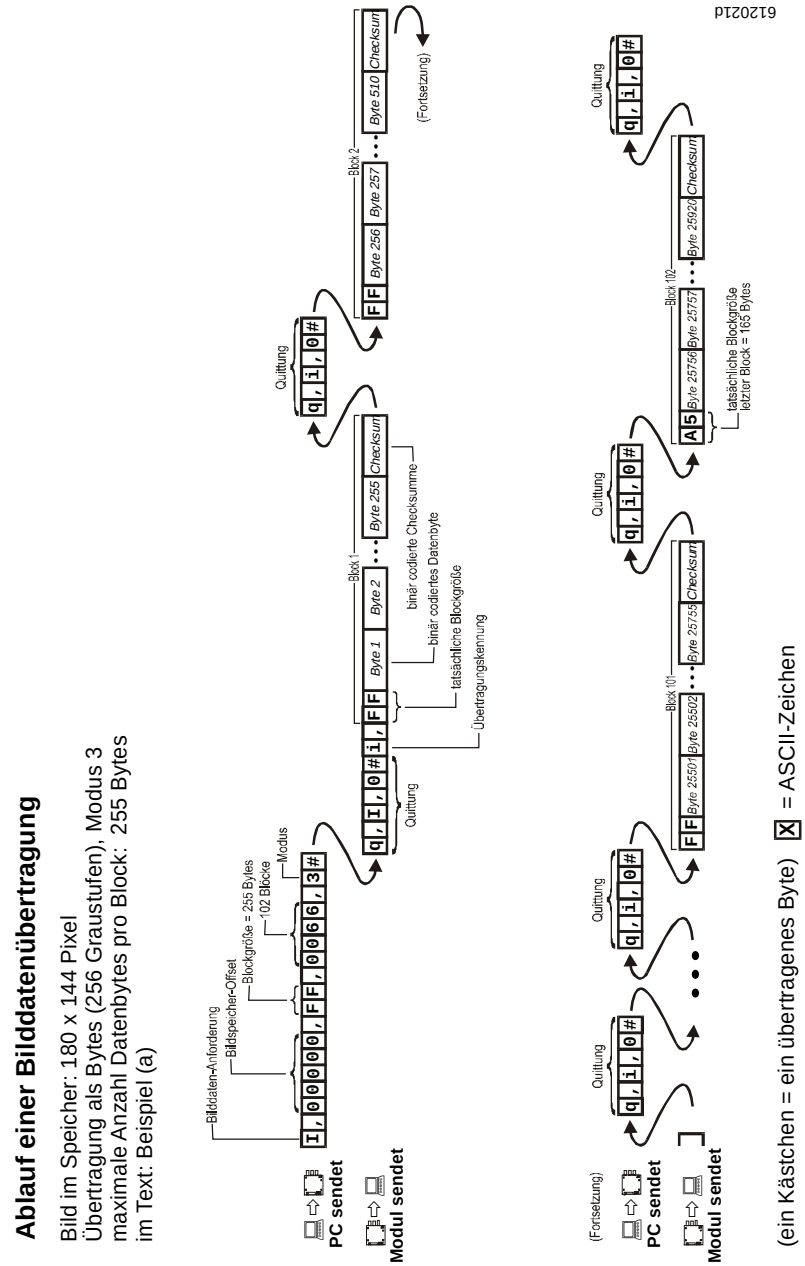


Bild 57: Ablauf einer Bildübertragung

Hinweis:
Da es sich um eine binäre Datenübertragung handelt, darf die Schnittstelle nicht auf „Software-Handshake“ konfiguriert werden (wie z.B. XON/XOFF). Besser ist es, die Puffergröße der Datenblockgröße anzupassen und die Flußkontrolle mittels der Quittungssendungen auf einem höheren Layer zu realisieren.

6.4.10 Bildgröße und –skalierung einstellen

Befehlscode: `P,hhh,vvv,xxx,yyy,ppp,lll,c,i#`

Parameter:
- *hhh* = linke Startposition des Bildausschnittes im Videobild (horizontal), hex, 3-stellig
- *vvv* = obere Startposition des Bildausschnittes im Videobild (vertikal), hex, 3-stellig
- *xxx* = Größe des Bildausschnitts in x-Richtung, hex, 3-stellig
- *yyy* = Größe des Bildausschnitts in y-Richtung, hex, 3-stellig
- *ppp* = gewünschte Größe des Videobilds in x-Richtung (Pixel per lines), hex, 3-stellig
- *lll* = gewünschte Zeilenzahl des Videobilds (y-Richtung), hex, 3-stellig
- *c* = verwendetes Farbsystem, hex, 1-stellig
  - 0 = Reserviert
  - 1 = PAL
  - 2 = Reserviert
- *i* = Voll- oder Halbbildmodus:
  - 0 = Halbbild
  - 1 = Reserviert
  - 2 = Vollbild

Bemerkung: Die mit „P“ übergebenden Parameter entsprechen (bis auf i) denen der „Set_Image_Size“-Funktion. Die Funktion und deren Parameter sind ausführlich im Abschnitt *6.3.6.25, „Set_Image_Size“* beschrieben.
Der Parameter „i“ muß übergeben werden um dem Grabber bekannt zu geben ob ein Halbbild (odd, voreingestellt) oder ein Vollbild (odd und even Halbbild) gegrabbt werden sollen.

Quittung:
- `q,P,0#` = OK
- `q,P,1#` = Parameterfehler
- `q,P,2#` = ungültiger Wertbereich eines Parameters

Beispiele zur Bildgrößen-Einstellung:

(a) Das zu grabbende Bild soll folgende Eigenschaften haben:
- nicht seitenverzerrt
- 1:4 skaliertes
- 180 x 144 Pixel Auflösung
- Halbbildauflösung

daraus ergeben sich folgende Parameter:
Dezimal: 0,0,180,288,192,288,1,0
Hex/Befehl: **`P,000,000,0B4,120,0C0,120,1,0#`**

(b) Das zu grabbende Bild soll folgende Eigenschaften haben:
- nicht seitenverzerrt
- 1:2 skaliertes
- 360 x 288 Pixel Auflösung
- Halbbildauflösung

daraus ergeben sich folgende Parameter:
Dezimal: 0,0,360,576,384,567,1,0
Hex/Befehl: **`P,000,000,168,240,180,240,1,0#`**

(c) Das zu grabbende Bild soll folgende Eigenschaften haben:
- nicht seitenverzerrt
- 1:1 skaliertes
- 768 x 576 Pixel Auflösung (max. Auflösung)
- Vollbildauflösung

daraus ergeben sich folgende Parameter:
Dezimal: 0,0,768,576,768,567,1,2
Hex/Befehl: **`P,000,000,300,240,300,240,1,2#`**

6.4.11 Bildaufnahme durch Alarmeingänge

Eine Bildaufnahme kann nicht nur durch das Kommando G, sondern auch durch ein Ereignis an einem der optoentkoppelten Signaleingänge (*siehe Abschnitt 3.1.4*) ausgelöst werden.
Ein Ereignis ist dabei eine beliebige Flanke, also ein Wechsel des Pegels von log. 0→1 oder von log. 1→0.

Wird Eingang X als aktiv erkannt, so digitalisiert die Einheit das Videosignal von Videoeingang X und legt dieses Bild im Bildspeicher ab. Anschließend wird ein Statusflag gesetzt, das angibt, welcher Videokanal das Bild lieferte. Werden mehrere Triggersignale gleichzeitig aktiv, so wird das Kamerabild vom niederwertigsten Kanal gespeichert.

Hinweis:
Der Bildspeicher enthält bei diesem Verfahren das Bild des Eingangskanals, bei dem als letztes das Triggersignal eine Flanke aufwies.

Das Statusflag kann mit dem Kommando T abgefragt werden. Der Host-Rechner kann dann entscheiden, ob er das gespeicherte Bild abholen möchte oder nicht.
Das Flag kann durch ein Kommandoparameter gelöscht werden.

Hinweis:
Das Statusflag erhält auch dann einen neuen Wert (Kanalnummer), wenn explizit durch den Anwender ein neues Bild von einem anderen Kanal mit dem Kommando „G“ gegrabbt wird.

Mit dem Bild wird ein Zeitstempel erzeugt.
Zur Zeitstempelerzeugung wird ein interner 16-Bit-Zähler ca. im Sekundentakt hochgezählt. Der Zähler beginnt das Hochzählen bei Setzen des Statusflags. Der Zählerstand kann zum Leitstand/PC übertragen werden. Bei Überlauf des maximalen Zählerstands wird der max. Zählerwert zurückgegeben. Dieser Wert ist dann als „Aufzeichnung vor mehr als x Sekunden“ zu interpretieren.
Der Zähler wird bei Rücksetzen des Statusflags durch den Host-Rechner automatisch gestoppt und in den Ruhezustand zurückgesetzt.

6.4.12 Zeitstempel lesen / rücksetzen

Befehlscode:	`T,`*r*`#`
Parameter:	r Rücksetzen: 0 = Zeitstempel nur lesen 1 = Zeitstempel lesen und dann zurücksetzen
Beispiel:	`T,0#`
Quittung:	`q,T,1#` = Parameterfehler `q,T,0#t,`*kk*`,`*zzzzz*`#` *kk* = triggernder Kanal *zzzzz* = Zeit in sec. seit Triggerung, max. 65535 sec.
Beispiel:	`q,T,0#t,03,01800#` =Eingang 3 hat vor 30 min. eine Bildaufnahme getriggert

Eine Bildaufnahme kann (u.a.) durch einen Flankenwechsel an einem Alarmeingang ausgelöst werden. Mit Hilfe des Zeitstempels kann in diesem Fall der Zeitpunkt der Alarmauslösung und der triggernde Kanal bestimmt werden. *Informationen zur Beschaltung der Alarmeingänge finden Sie in siehe Abschnitt 3.1.4, „Optoentkoppelte I/O-Ports*“). Der Zeitstempel gibt die Zeitspanne an, die seit Auslösung der Bildaufnahme verstrichen ist. Die Maximalzeit beträgt ca. 18 Stunden.

Hinweis:
Der triggernde Kanal erhält auch dann einen neuen Wert, wenn explizit durch den Anwender ein neues Bild von einem anderen Kanal mit dem Kommando „G“ gegrabbt wird.
Die Zeitangabe wird durch eine manuell ausgelöste Bildaufnahme nicht verändert.

6.4.13 Eingangs-Status anfordern

Befehlscode: C#
Parameter: keine
Beispiel: C#
Quittung: `q,C,0#c,`*nn*`#` = Bestätigung, Status wird gesendet:
nn = Inputstatus als Hexadezimalwert, binär codiert
`q,C,1#` = Parameterfehler
Bemerkung:

Format der Quittung:

Bitwertigkeit *nn*								
	-	-	-	-	I/O 4	I/O 3	I/O 2	I/O 1
***nn* Binär**	-	-	-	-	Bit 3	Bit 2	Bit 1	Bit 0
***nn* dezimal**	-	-	-	-	8	4	2	1
Bit gesetzt = Pegel an Input *m* vorhanden								

Beispiel:
Rückgesendete Quittung = `q,C,0#c,06#`
nn = 06_{HEX} = 06_{DEZ} = 0000 0110_{BIN} = 4 + 2 ⇨ High-Pegel an I/O3 und I/O2 vorhanden

Mit dieser Funktion kann der momentane Status der optoentkoppelten I/O-Eingänge (*siehe Abschnitt 3.1.4, „Optoentkoppelte I/O-Ports*“) abgefragt werden. Dies kann z.B. zu einer Überprüfung von angeschlossenen, low-aktiven Alarmkreisen benutzt werden.

6.4.14 Output-Status setzen

Befehlscode: O,*s#*
Parameter: s = Bit-Muster für Output Ports, hex, 1-stellig
Beispiel: O,0#
Quittung: q,O,0# = OK
q,O,1# = Parameterfehler

Beispiel: q,O,0# = Bei allen 4 Ausgängen des Moduls wird der Transistor des Optokopplers leitend.

Bemerkung:

Format des Bit Musters „s“:

Bitwertigkeit *s*				
	I/O8	I/O 7	I/O 6	I/O 5
***s* binär**	Bit 3	Bit 2	Bit 1	Bit 0
***s* dezimal**	8	4	2	1
Bitwert = 1 = Transistor des Optokopplers gesperrt Bitwert = 0 = Transistor des Optokopplers leitend				

Beispiel:
Gesendeter Parameter = O,E#
s = E_{HEX} = 14_{DEZ} = 1110_{BIN} = ⇨ Transistor an I/O8, I/O7 und I/O6 gesperrt, Transistor an I/O5 leitend

Mit dieser Funktion können die optoentkoppelten I/O-Ausgänge I/O5-I/O8 (*siehe Abschnitt 3.1.4, „Optoentkoppelte I/O-Ports*“) geschaltet werden. Dies kann z.B. zur Steuerung von externen Schaltungen benutzt werden.

Hinweis:
Nach dem Einschalten sind die Transistoren der optoentkoppelten I/O-Ausgänge gesperrt (s = F_{HEX}).
Werden die Transistoren der Ausgänge leitend, so ist der Ausgangspin mit Masse verbunden.

6.4.15 Fehler-Status anfordern

Befehlscode: F#
Parameter: keine
Beispiel: F#
Quittung: `q,F,0#f,`*nn*`#` = Bestätigung, Status wird gesendet:
nn = Fehlerstatus als Hexadezimalwert, binär codiert
$0x01_{Hex}$ = Initialisierung oder Einstellung von Videoparameter (BT829) fehlgeschlagen
$0x02_{Hex}$ = Starten des Grabb-Vorgangs fehlgeschlagen
$0x04_{Hex}$ = Das Senden eines Zeichens an der seriellen Schnittstelle ist fehlgeschlagen
$0x08_{Hex}$ = Das Empfangen eines Zeichens von der seriellen Schnittstelle ist fehlgeschlagen
$0x10_{Hex}$ = Antwortquittung des Host bei der Bildübertragung fehlerhaft
$0x20_{Hex}$ = Reserviert
$0x40_{Hex}$ = Reserviert
`q,F,1#` = Parameterfehler

Bemerkung:

Format der Quittung:

Bitwertigkeit *nn*								
***nn* Binär**	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
***nn* dezimal**	128	64	32	16	8	4	2	1
Bit gesetzt = Fehler entsprechender Fehlercode vorhanden								

Beispiel:
Rückgesendete Quittung = q,F,0#f,0A#
nn = $0A_{HEX}$ = 10_{DEZ} = 0000 1010_{BIN} = 8 + 2 ⇨ Fehler $0x02_{Hex}$ und $0x08_{Hex}$ sind aufgetreten.

Mit dieser Funktion kann die interne Variable „Fehler-Code“ ausgelesen werden. Mit Hilfe der Fehlernummern kann die Ursache von Fehlfunktionen des Moduls ermittelt werden.

$0x01_{Hex}$ = Initialisierung oder Einstellung von Videoparameter (BT829) fehlgeschlagen. In diesem Fahl ist entweder ein Fehler in der I²C Steuerung auf dem Modul oder ein Defekt im Videobaustein BT829 vorhanden.

$0x02_{Hex}$ = Starten des Grabb-Vorgangs fehlgeschlagen. In diesem Fall ist der Video-Control Baustein auf dem Modul defekt.

$0x04_{Hex}$ = Das Senden eines Zeichens an der seriellen Schnittstelle ist fehlgeschlagen. Dieser Fehler tritt im Handshake-Betrieb auf, wenn das CTS-Signal von der Gegenstelle nicht innerhalb einer bestimmten Zeit (voreingestellt 5 sec.) zurückgenommen wird. Überprüfen Sie das CTS Signal.

$0x08_{Hex}$ = Das Empfangen eines Zeichens von der seriellen Schnittstelle ist fehlgeschlagen. Wenn laut Protokoll Zeichen (Quittungen oder restliche Zeichen einer Anforderung) erwartet werden, wartet das Modul eine bestimmte Zeit (voreingestellt 3 sec.) auf das Zeichen. Wird kein Zeichen innerhalb der eingestellten Empfangs-Timeout-Zeit empfangen wird dieser Fehler gesetzt und eine „0_{dez}“ empfangen. In diesem Fall ist entweder die Leitung der seriellen Verbindung gestört oder Quittungen bzw. die restlichen Zeichen einer Anforderung wurden zu spät gesendet. Stellen Sie mit der Funktion „Z“ (*siehe Abschnitt 6.4.16, „Empfangs-Timeout der Schnittstelle setzen“)* eine andere Timeout-Zeit ein oder überprüfen Sie die serielle Verbindung.

$0x10_{Hex}$ = Antwortquittung des Hosts bei der Bildübertragung fehlerhaft. Dieser Fehler wird gesetzt, wenn bei einer angeforderten Bildübertragung „I, Bilddaten anfordern“ eine fehlerhafte Quittung vom Host gesendet wird *(siehe Abschnitt 6.4.9, „Bilddatenformat“).* Überprüfen Sie die Quittung im Host auf ihre Richtigkeit.

$0x20_{Hex}$ = Reserviert

$0x40_{Hex}$ = Reserviert

6.4.16 Empfangs-Timeout der Schnittstelle setzen

Befehlscode: Z,*tt*#
Parameter: *tt* = Timeoutzeit in Sekunden für ein einzelnes Zeichen von der seriellen Schnittstelle, hex, 2-stellig. Maximaler Wertebereich [0...0x40$_{Hex}$] = [0...64$_{Dez}$] Sekunden.
Beispiel: `Z,08#`
Quittung: `q,Z,0#` = OK
`q,Z,1#` = Parameterfehler

Bemerkung:

Format des Bit Muster „tt“:

Bitwertigkeit *tt*								
***tt* Binär**	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
***tt* dezimal**	-	64	32	16	8	4	2	1
Bit gesetzt = Fehler entsprechend Fehlercode vorhanden								

Beispiel:
Gesendete Parameter = Z,0A#
tt = 0A$_{HEX}$ = 10$_{DEZ}$ = 0000 1010$_{BIN}$ = Timeout Zeit für das Empfangen eines seriellen Zeichens wird auf 10 Sekunden gesetzt.

Das Modul wartet im Grundzustand auf ein Zeichen, welches im Protokoll (Protokoll-Einleitungszeichen) vereinbart ist. Dann folgen laut Protokoll weitere Zeichen bzw. Quittungen. Jedes einzelne dieser Folgezeichen bzw. Quittungszeichen wird in einem bestimmten Zeitraum (Timeout-Zeit) erwartet. Werden diese Zeichen nicht innerhalb der Timeout-Zeit gesendet so geht das Modul wieder in den Grundzustand (Warten auf ein Protokoll-Einleitungszeichen).
Durch das Verändern der Timeout-Zeit können Sie die Wartezeit des Moduls bestimmen und damit festlegen, wie schnell das Modul nach einer Störung im Datenstrom wieder in den Grundzustand springt.

Hinweis:
Ein Hochsetzen der Timeout-Zeit ist auch dann sinnvoll wenn Sie für einen ersten Test mit einem Terminal (z.B. HyperTerminal) arbeiten und die Protokoll-Befehle und Quittungen per Hand eingeben wollen.

Document: grabbMODUL-4
Document number: L-612d_2, Januar 2005

Wie würden Sie dieses Handbuch verbessern?

Haben Sie in diesem Handbuch Fehler entdeckt? Seite

Eingesandt von:
Kundennummer:

Name:

Firma:

Adresse:

Einsenden an:

PHYTEC Technologie Holding AG
Postfach 100403
D-55135 Mainz, Germany
Fax : +49 (6131) 9221-33

Published by

PHYTEC

Ordering No. L-612d_2
Printed in Germany