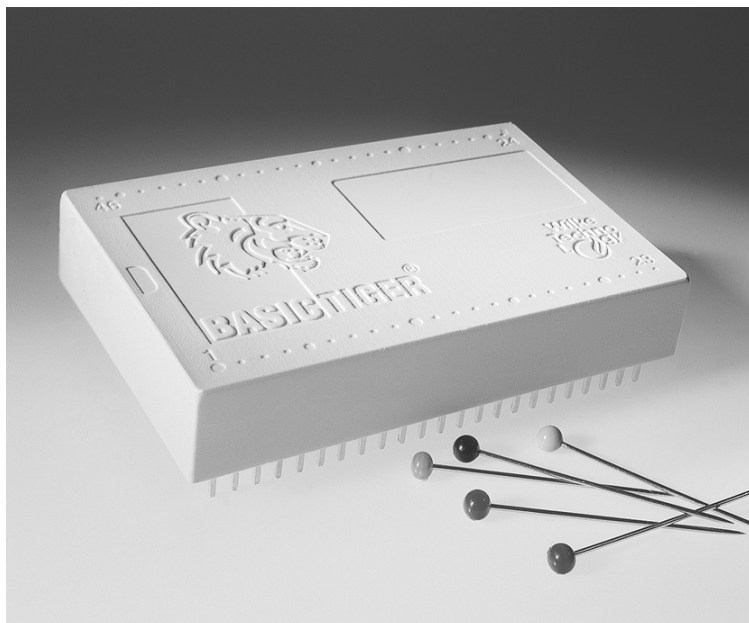




BASIC-TIGER®

Hardware-Handbuch

Installation

Bevor Sie beginnen	1
Installation	2
Entwicklungsumgebung	3
BASIC-Tiger[®]-Steuercomputer	4
Häufig gestellte Fragen - Support	5
Stichwortverzeichnis	6
Anhang	7

Redaktion	Klaus Hiltrop
Abbildungen	Joji
Cover	Werbeagentur Kordoni, Aachen
Druck	Print Production, Aachen
	5. Auflage - Version 5.0
Copyright	© Wilke Technology Heider-HofWeg 23D 52080 Aachen, Germany

Dieses Handbuch, sowie die Hard- und Software, die es beschreibt, ist urheberrechtlich geschützt und darf ohne ausdrückliche schriftliche Genehmigung von Wilke Technology GmbH in keiner Weise vervielfältigt, übersetzt oder in eine andere Darstellungsform gebracht werden.

Warenzeichen BASIC-Tiger[®], TINY-Tiger[®], TigerCube[®] sind eingetragene Warenzeichen von Wilke Technology GmbH.
TouchMemory[®] ist eingetragene Warenzeichen von Dallas Semiconductors.
Windows[™], Windows 95, Windows NT sind eingetragene Warenzeichen der Microsoft Corp.

Diejenigen Bezeichnungen in dieser Publikation von Erzeugnissen und Verfahren, die zugleich Warenzeichen sind, wurden nicht besonders kenntlich gemacht. Solche Namen sind Warenzeichen der jeweiligen Warenzeicheninhaber. Aus dem Fehlen der Markierung[®] kann nicht geschlossen werden, daß diese Bezeichnungen freie Warennamen sind.

Hinweis Herausgeber, Übersetzer und Autoren dieser Publikation haben mit größter Sorgfalt die Texte, Abbildungen und Programme erarbeitet. Dennoch können Fehler nicht völlig ausgeschlossen werden. Wilke Technology übernimmt daher weder eine Garantie noch eine juristische Verantwortung oder Haftung für Folgen, die auf fehlerhafte Angaben zurückgehen. Mitteilungen über eventuelle Fehler werden jederzeit gerne entgegengenommen.

Die Angaben in diesem Handbuch gelten nicht als Zusicherung bestimmter Produkteigenschaften. Änderungen, die dem technischen Fortschritt dienen, bleiben vorbehalten.

Alle Rechte vorbehalten • Printed in Germany
Gedruckt auf chlorfrei gebleichtem Papier

Inhaltsverzeichnis

1	Bevor Sie beginnen	3
	Willkommen	3
	Was ist neu?	4
	Was ist nicht ganz so neu?	6
	Systemvoraussetzungen	7
	Sicherheitshinweis	8
	Typografische Konventionen und Symbole	9
	BASIC-Tiger®	10
	Multitasking	11
	Device-Treiber	11
	Funktionen	12
2	Installation	17
	Installation der Entwicklungsumgebung	17
	Beispiele	23
	Installation der Hardware	27
	Blitzstart / Erste Schritte	30
	Programm „Hello World“	31
	Programm „Lauflicht“	33
	Programm „3Tasks“	35
3	Entwicklungsumgebung	45
	Software-Entwicklungsumgebung	46
	Befehle der Entwicklungsumgebung	47
	Datei-Menü	47
	Bearbeiten-Menü	51
	Suchen-Menü	53
	Anzeige-Menü	56
	Starten-Menü	65
	Debug-Menü	69
	Optionen-Menü	73
	Fenster-Menü	81
	Hilfe-Menü	82
	Editor	84
	Text markieren	84
	Text löschen	84
	Text einfügen	84
	Compiler-Fehlermeldungen	85
	Downloader	95
	Systemvoraussetzungen	95
	Hardware-Entwicklungsumgebung	96

Inhaltsverzeichnis

Plug & Play Lab	99
Spannungsversorgung	101
Batteriepufferung	102
PC-Mode	103
Serielle Anschlüsse	104
Pinbelegung DB9 (SER0)	107
Pinbelegung DB9 (SER1/PC)	108
Relais	110
Darlington-Transistoren (NPN)	112
Beep	114
Mikrofon-Eingang/Klinkenbuchse	116
Analogverstärker (4x)	118
Analog-Eingänge	119
PWM-Verstärker	120
Power-Verstärker	122
Bus-System für LCD, Tastatur und 64 I/O-Pins	125
Belegung J12	128
BASIC Tiger [®] Prototyping-Board	129
Spannungsversorgung	131
Batteriepufferung	132
PC-Mode	133
Serielle Anschlüsse	134
Pinbelegung DB9	135
Pinbelegung DB15	136
Pinbelegung 'Serl 0'-Stiftleiste	137
Darlington transistors (NPN)	138
Beep	140
Analogverstärker (4x)	142
Analog inputs	143
PWM-Filterverstärker	144
Bus-System für LCD, Tastatur und erw. I/O-Pins	146
Pinbelegung des Tastaturanschlusses	150
Pinbelegung des LC-Display-Anschlusses	151
Pinbelegung J23	152
TINY Tiger [®] Prototyping board	153
Spannungsversorgung	155
PC-Mode	156
Serielle Anschlüsse	157
Pinbelegung DB9 'ser 0'	158
Pinbelegung DB9 'ser 1'	159
Treibertransistors (NPN)	160
LED status display	162
Bus-System für LCD, Tastatur und erw. I/O-Pins	164
Pinbelegung des Tastaturanschlusses	168
Pinbelegung des LC-Display-Anschlusses	169

Pinbelegung J2	170
Tiger-Terminal	171
Inbetriebnahme	172
Bus-Steckverbinder für LCD, Tastatur und 8 Outputs	174
Pinbelegung des LC-Display-Anschlusses	175
Erweiterte Outputs	175
Technische Daten des Tiger-Terminals:	176
Programmier-Adapter	177
4 BASIC-Tiger®-Steuercomputer	181
Flash-Speicher	181
Modulreihe A	185
BASIC Tiger® Modul A RESET-In	186
BASIC-Tiger®-Modul A Pinbelegung	188
Pin-Beschreibung	189
TINY Tiger® Modul	193
TINY Tiger® RESET-In	195
TINY Tiger® Pinbelegung	196
Pin-Beschreibung	197
BASIC Tiger® Modul E	201
TINY Tiger® Modul E Pinbelegung	204
BASIC Tiger® Module E Pinbeschreibung	205
TINY Tiger® Modul E RESET-In	207
TCAN – CAN-Tiger	213
BASIC-Tiger®-TCAN-Modul Pinbelegung	215
Pin-Beschreibung TCAN	216
I/O-Erweiterungsmodule	221
Adreßgenerierung	222
EP1-64HDE	223
Pinbelegung EP1-64HDE	223
Pin-Beschreibung EP1-64HDE	224
Adressierung EP1-64HDE	226
EP2-64SDA	233
Pinbelegung EP2-64SDA	233
Pin-Beschreibung EP2-64SDA	234
Adressierung EP2-64SDA	236
EP3-32-32	243
Pinbelegung EP3-32-32	243
Pin-Beschreibung EP3-32-32	244
Adressierung des EP3-32-32	246
EP4-32PDA	253
Pinbelegung EP4-32PDA	253
Pin-Beschreibung EP4-32PDA	254
Adressierung EP4-32PDA	256
Temperatursensor EP4-32PDA	260

Inhaltsverzeichnis

EP5-32GDE	265
Pinbelegung EP5-32GDE	265
Pin-Beschreibung EP5-32GDE	266
Adressierung des EP5-32GDE	268
EP6-UNIVD	275
Pinbelegung EP6-UNIVDE	275
Pin-Beschreibung EP6-UNIVD	276
Adressierung des EP6-UNIVD	278
EP10-16-16	287
Pinbelegung EP10-16-16	287
Pin-Beschreibung EP10-16-16	288
Adressierung des EP10-16-16	290
EP11-8AD, EP12-16AD, EP13-32AD, EP14-64AD	297
Pinbelegung EP11-8AD bis EP14-64AD	297
Pin-Beschreibung EP11-8AD	298
Adressierung der EP11-EP14-AD	300
Erweitertes I/O-System	305
Adreß-Generierung	307
Erweiterung der Ausgänge	310
Erweiterung der Eingänge	313
Ein Beispiel für erweiterte Eingänge und Tastatur	316
Die Softwareseite der erweiterten I/O-Ports	318
Portadresse und Steuerleitungen verlegen	322
Tastatur anschließen	323
Tastatur anpassen	327
5 Häufig gestellte Fragen	331
Rat und Hilfe	334
6 Stichwortverzeichnis	337
7 Anhang	341
ASCII-Tabelle	341
EBCDIC Codes	342
Baudot-Code-Satz	343
ANSI-Steuersequenzen	344
Windows 95/98/NT Shortcuts	346
Short-Cuts Tiger-BASIC® Version 5	348
Kennzeichnung von Widerständen und Kondensatoren	350
Farbcodes	350
Wertkennzeichnung durch Buchstaben	351
Toleranzkennzeichnung durch Buchstaben	352
Mittlere Schrittweite für Widerstands-Wachstum aufeinanderfolgender Werte:	354

Inhaltsverzeichnis

Normreihen für Widerstandswerte	354
BASIC-Tiger [®] Modul A – Nutzung der Pins	361
TINY-Tiger [®] – Nutzung der Pins	365
TINY-Tiger [®] Modul E – Nutzung der Pins	369
BASIC-Tiger [®] -CAN-Modul – Nutzung der Pins	373

Inhaltsverzeichnis

Leere Seite

Bevor Sie beginnen	1
Installation	2
Entwicklungsumgebung	3
BASIC-Tiger [®] -Steuercomputer	4
Häufig gestellte Fragen - Support	5
Stichwortverzeichnis	6
Anhang	7

Leere Seite

1 Bevor Sie beginnen

Willkommen

Willkommen beim BASIC-Tiger[®] Entwicklungssystem.

Dieses Buch macht Sie mit der Hardware des BASIC-Tigers[®] und der BASIC-Tiger[®]-Entwicklungsumgebung vertraut und ermöglicht Ihnen einen schnellen Start.

Lange Zeit galt die Programmierung von Microcontrollern als ein Bereich, der nur Spezialisten vorbehalten war. Die Programmierung fand meistens in Assembler statt, wodurch die Programme zwar schnell, aber schwer verständlich waren. Der Entwicklungsaufwand selbst kleinerer Projekte, hat schnell den Umfang von mehreren Monaten angenommen. Entschied man sich, in BASIC zu programmieren, so wurde die Verarbeitungsgeschwindigkeit relativ langsam und die Möglichkeiten waren eingeschränkt.

Der BASIC-Tiger[®] schließt nun diese Lücke. Durch den innovativen BASIC-Dialekt ist die Programmierung sehr einfach und verkürzt die Einarbeitungszeit. Die Verarbeitungsgeschwindigkeit des BASIC-Tigers[®] liegt sehr hoch und stellt damit so manchen in Assembler oder C programmierten Controller in den Schatten. Multitasking, das ohne komplizierten Overhead auskommt, führt zu besser strukturierten und leichter pflegbaren Programmen. Für immer wiederkehrende Programmieraufgaben steht eine wachsende Bibliothek von Funktionen und Beispiel-Applikationen zur Verfügung. Durch die Unterstützung von Gerätetreibern, die ständig weiterentwickelt werden, kann der BASIC-Tiger[®] mit dem Projekt wachsen und so auch Aufgaben wahrnehmen, an die bei Projektbeginn noch nicht gedacht wurde.

Wenn Sie direkt mit der Programmierung beginnen möchten, führen Sie die Installation durch, und folgen Sie direkt dem Kapitel „Blitzstart/Erste Schritte“.

Eine umfassende Beschreibung der Sprache Tiger-BASIC[®] finden Sie im Programmierhandbuch. Die zu jeder Instruktion oder Funktion abgedruckten Beispiele finden Sie im Verzeichnis **Examples**, die Grafikbeispiele im Verzeichnis

Graphic_Examples.

Für viele Geräte gibt es in Tiger-BASIC[®] sogenannte Device-Treiber. Zu den Geräten zählen externe Geräte wie LCD, Drucker, Encoder-Drehgeber, serielle Geräte. Geräte können sich aber auch im Modul befinden. So gibt es Device-Treiber für die Analogen Eingänge, die analoge Ausgabe mit Pulsweitenmodulation, Frequenzmessung über einen Eingangspin, Pulsausgabe und Pulsweitenmessung, und viel mehr. Die Device-Treiber sind in einem eigenen Handbuch beschrieben. Dieses Handbuch enthält auch einige Applikation, die den Umfang eines Beispiels überschreiten. Nach der Installation finden Sie diese Applikationen im Verzeichnis **Applications** und die Beispiele zu den Device-Treibern im Verzeichnis **DeviceDriver_Examples**.

Bevor Sie beginnen

1

Was ist neu?

Wenn Sie bereits erfahrener Tiger-BASIC[®]-Anwender sind, werden Sie sich sicher für Neuigkeiten interessieren. Sie finden hier eine kurze Übersicht über das Wichtigste, was auf dem Hardware-Sektor und in der Benutzeroberfläche neu ist seit dem letzten Up-to-Date. Unter der nächsten Überschrift 'Was ist nicht ganz so neu?' finden Sie die Neuerungen des Vorigen Up-to-Dates. Neue Instruktionen, Funktionen und Device-Treiber sind in den beiden anderen Handbüchern aufgeführt.

Version 5 kann nun parallel zu Version 4 verwendet werden, da nicht die gleichen Einträge in WIN.INI bzw. der Registry benutzt werden.

Die Verzeichnisnamen und Dateinamen sind länger. Die Namen der Beispielpprogramme zu den Instruktionen und Funktionen sind nun ausgeschrieben. Dies wurde möglich, weil die Unterstützung von WIN 3.1 weggefallen ist. Die Beispiele sind auf mehrere Verzeichnisse nach Themengebiet verteilt.

Der Editor hat einige Neuerungen:

- Es gibt ab der Version 5 die Möglichkeit, Bearbeitungsschritte im Editor rückgängig zu machen. Im Menü Optionen unter Editor wird eingestellt, wieviele Schritte rückgängig gemacht werden können.
- Der Quelltext hat eine farbige Syntax-Hervorhebung (4 verschiedene Stile) und die Hintergrundfarbe ist einstellbar.
- Bei der Anzeige des vom Programm belegten RAMs im Modul wird jetzt auch die Größe des RAMs für das RUN-Time-System berücksichtigt.
- Das Editieren kann blockiert werden, z.B. um sich vor versehentlichem Editieren während einer Debugsitzung zu schützen. Dies läßt sich auch automatisieren: kein Editieren nach jedem Download.
- Bei versehentlichen Änderungen im Quelltext kann nun ein erneutes Compilieren unterbunden werden, da vor dem Compilieren gefragt wird.
- Das Verhalten der Oberfläche beim Downloaden kann nun weitgehend beeinflusst werden. Die Anzahl der Wiederholungsversuche sowie die Wartezeiten lassen sich einstellen. Damit sollten auch Downloads über Modem oder sogar Satellit möglich sein.
- Vergebliche Download-Versuche lassen sich mit ESC vorzeitig abbrechen.
- Die Userbereiche des Flash-Speichers können von der Oberfläche aus eingesehen und in verschiedenen Formaten abgespeichert werden.
- Strings sind ebenfalls in verschiedenen Formaten einsehbar und können abgespeichert werden, z.B. um eine LCD-Grafik im Handbuch abdrucken zu können.
- Die Anzeige der überwachten Ausdrücke ist erweitert worden.
- Im Menü Hilfe kann eine weitere Hilfedatei aufgerufen werden, die letzte Änderungen enthält. Über diese Hilfe sind auch Erweiterungen besser dokumentierbar, die nach der Version 5 erscheinen.
- Automatisches Back-up vor dem Compilieren und zeitgesteuert sind einstellbar.

Bevor Sie beginnen

Es gibt ein neues BASIC-Tiger[®]-Modul mit **CAN-Schnittstelle**. Die Pinbelegung des Moduls finden Sie in diesem Handbuch, die Beschreibung des Device-Treibers zu CAN im Device-Treiber-Handbuch.

Für Anwendungen mit Grafik-LCDs steht ein spezielles Entwicklungkit zur Verfügung (**Graphic-Toolkit**).

1

Bevor Sie beginnen

1

Was ist nicht ganz so neu?

Die hier aufgeführten Neuerungen gab es beim letzten Up-To-Date.

Der Editor hat einige Neuerungen:

- Die Verknüpfung der Dateien mit der Namenserverweiterung 'TIG' mit der Entwicklungsoberfläche (TGBAS32) ist nun möglich. Wenn im Explorer eine Datei *.TIG doppelt angeklickt wird, dann wird Tiger-BASIC[®] aufgerufen und die Datei geladen. Um die Verknüpfung herzustellen, halten Sie die SHIFT-Taste fest, während Sie die rechte Maustaste über einer ausgewählten 'TIG'-Datei klicken. Im Kontextmenü wählen Sie **Öffnen mit**. Im folgenden Dialog kreuzen Sie 'Dateien dieses Typs immer mit diesem Programm Öffnen' an und suchen die Anwendung TGBAS32.EXE (normalerweise im Verzeichnis \TIGERBAS\BIN). Damit ist die Verknüpfung hergestellt.
- Bei der Anzeige des vom Programm belegten RAMs im Modul wird jetzt auch die Größe des RAMs für das RUN-Time-System berücksichtigt.
- Informationen über Device-Treiber (Flash- und RAM-Nutzung, Version) können nun per Mausklick mit dem Kommando 'Gerätetreiberliste' im Menü Ansicht abgefragt werden.
- Die Anzeige der überwachten Ausdrücke ist verbessert worden.
- Der Editor nimmt nun auf Kommando die Einrückungen von Schleifen und IF-Bereichen vor.

Es gibt neue analoge Erweiterungsmodule zur Realisierung erweiterter I/O-Ports. Im Kapitel 0 unter 'I/O-Erweiterungsmodule' ab Seite 297 sind die neuen Module beschrieben. Beachten Sie bitte auch die laufende Werbung sowie unsere Internetseiten (www.wilke.de).

Modul	Kanäle	Auflösung	Temp.
EP11-8AD	8	12 Bit	0°C-70°C
EP12-16AD	16	12 Bit	0°C-70°C
EP13-32AD	32	12 Bit	0°C-70°C
EP14-64AD	64	12 Bit	0°C-70°C

Systemvoraussetzungen

1

Um mit der Entwicklungsumgebung des BASIC-Tigers[®] arbeiten zu können, benötigen Sie einen IBM-kompatiblen Rechner. Wir empfehlen folgende Mindestausstattung:

- Pentium-II-Prozessor mit 233MHz
- Festplatte mit mind. 100 MByte freiem Speicher
- VGA-Grafikkarte
- 32 MB Arbeitsspeicher
- Maus
- Windows 95/98/2000/NT
- ein freier COM-Port
- CD-ROM-Laufwerk

Sicherheitshinweis

Die in diesem Entwicklungspaket enthaltenen Komponenten wie Plug & Play Lab, Prototyping-Boards, Toolkits, Adapter, Kabel etc. dienen nur als Arbeitshilfe bei der Entwicklung und Erprobung von Computer-Software und Hardware-Schaltungen. Ihr Zweck ist, ausschließlich dem sachkundigen Entwickler bei der Anfertigung von Laboraufbauten Zeit zu ersparen und Anregungen für eigene Designs zu geben.

In keinem Fall dürfen diese Komponenten von Nicht-Fachleuten installiert oder betrieben werden. Ebenso ist es nicht vorgesehen, daß derartige Laboraufbauten zur Steuerung von Anlagen und Geräten eingesetzt werden, insbesondere, wenn deren Fehlfunktion zu Schäden oder Gefahren führen können. In keinem Fall dürfen mit diesen Laboraufbauten hohe Spannungen oder Ströme gehandhabt werden.

Bei allen Änderungen an Schaltungen sowie beim Öffnen von Gehäusen, sind die Geräte bzw. Versuchsaufbauten vollständig von der Stromversorgung zu trennen.

Empfindliche Bauteile wie MOS Schaltkreise sind zur Vermeidung von Beschädigung nur in einer antistatisch ausgestatteten Laborumgebung zu handhaben.

Typografische Konventionen und Symbole

Die folgenden Schriften und Symbole wurden verwendet, damit Sie wichtige Informationen schnell erkennen können:

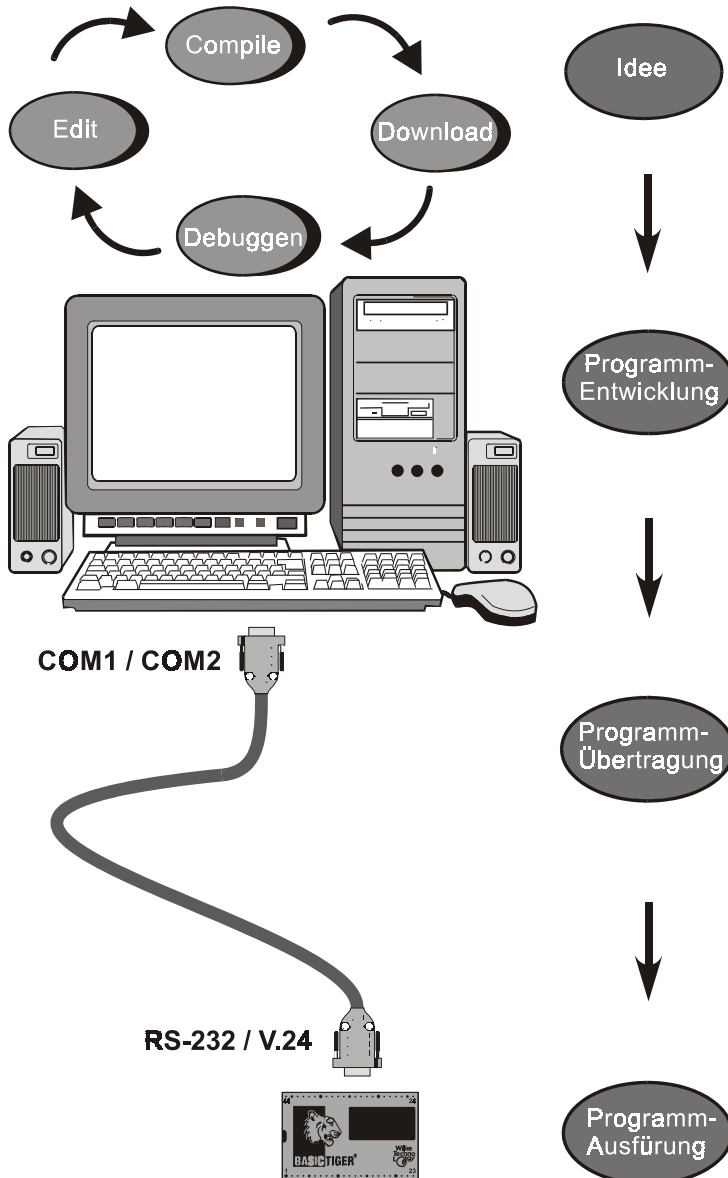
Element	Bedeutung
TASTE	Tastenbezeichnungen, z.B. RETURN
<i>Programmlisting</i>	Tiger-BASIC-Programmlisting
Instruktion	Tiger-BASIC®-Instruktion
<i>Variable</i>	Platzhalter für Elemente, die Sie Ihrer Anwendung gemäß eingeben müssen.
[]	Elemente, die optional eingegeben werden können.
!	Wichtige Anmerkung: bitte beachten!
Tip	Tips und Hinweise, die Ihnen die Arbeit erleichtern.

Bevor Sie beginnen

1

BASIC-Tiger[®]

In diesem Kapitel erfahren Sie etwas über die Besonderheiten des BASIC-Tigers[®].



Multitasking

Die auffälligste Besonderheit des BASIC-Tigers[®] ist seine Multitasking-Fähigkeit. Obwohl BASIC-Tiger[®]-Module nicht viel größer sind als ein CPU Chip, ist ein kompletter Multitasking Steuercomputer enthalten, der über eigenen Programmspeicher (FLASH), Arbeitsspeicher (SRAM + FLASH) sowie eine Reihe von Standard-I/Os verfügt. Mehrere Tiger-BASIC[®]-Programme (Tasks) können in den Programmspeicher der Tigers geladen werden und bleiben dort, ähnlich der Festplatte eines PC, dauerhaft gespeichert, bis sie durch neue Programme überschrieben werden. Der FLASH-Speicher läßt sich ferner auch als dauerhafter Speicher für Daten verwenden, der von BASIC-Programmen aus beschrieben, gelesen und gelöscht werden kann. Der Arbeitsspeicher kann bis zu mehreren MBytes SRAM betragen und läßt sich ebenfalls gegen Stromausfall sichern.

Der Vorteil von Multitasking wird sofort offenkundig, wenn man sich reale Aufgabenstellungen für einen Steuercomputer ansieht. Selten besteht eine Anwendung nur aus einer monolithischen, einzelnen Aufgabe, die linear in einer großen Schleife abgearbeitet wird. Schon bei kleinen Anwendungen gibt es meist 3, 4, 5 oder mehr eigenständige Aufgaben zu erledigen, die weitgehend unabhängig voneinander abzuarbeiten sind. Man denke nur an Ausgaben auf einen Drucker, Eingaben von Keyboards oder seriellen Eingängen etc., die Anwendungen oft „hängen“ lassen können. Um solche Zustände zu vermeiden ist oft einiger zusätzlicher Programmier- und Testaufwand erforderlich. Entsprechend schwerer verstehbar und pflegbar werden solche Programme.

Programmiert in Multitasking, bleibt nichts so leicht hängen. Eingaben, Ausgaben, abgeschlossene Steuerungen oder Auswertungen werden in eigenständigen Tasks abgearbeitet. Ist z.B. eine rechenintensive Auswertung noch nicht abgeschlossen, so werden dennoch erforderliche Steuersignale generiert, der Dialog mit einer Schnittstelle fortgeführt, Informationen auf Displays aufgefrischt und Bedienungstasten überwacht. Derartige Multitasking-Programme laufen nicht nur schneller und zuverlässiger ab, sie sind auch leichter zu pflegen und bieten mehr Übersicht. Zusätzliche Aufgaben können auch später noch leicht als neue Tasks hinzugefügt werden. Der Leistungsbedarf der einzelnen Aufgaben läßt sich durch Priorisierung der Tasks fein austarieren, Kontroll-Tasks können wichtige Funktionen im Auge behalten und ggf. Notprogramme starten und Alarme auslösen.

Die Programmierung in Multi-Tasking ist beim BASIC-Tiger[®] denkbar einfach und bereits mit wenigen BASIC-Zeilen zu realisieren. Ein einfaches Beispiel finden Sie auf Seite 31 unter der Überschrift **Programm „Hello World“**.

Device-Treiber

Durch die Verwendung von Device-Treibern, die die gerätespezifischen Eigenschaften von Peripheriegeräten berücksichtigen, erreicht der BASIC-Tiger[®] ein hohes Maß an Flexibilität und Leistungsfähigkeit bei einfacher Handhabung. Egal wie unterschiedlich verschiedene I/O-Geräte auch sein mögen, stets werden I/O-Kanäle, die mit Device-Treibern arbeiten, über die 6 standardmäßigen BASIC-Instruktionen

Bevor Sie beginnen

1

PRINT, PRINT USING, PUT, INPUT, INPUT LINE und **GET** angesprochen. Die systematische Ansteuerung mit einer Device-Nr. (Device=Gerät) sowie optionaler Sekundär-Adresse und Funktions-Code gestattet einen systematischen, klar verstehbaren Programmaufbau. Die Umleitung von I/O auf alternative Kanäle, das Hinzufügen weiterer I/O-Kanäle und der Übergang zu anderer Hardware wird denkbar einfach. Eine I/O-Instruktion wie: „**PUT #PUMPE, LEISTUNG4**“ leitet den unformatierten Output der Variablen „**LEISTUNG4**“ auf das Gerät „**PUMPE**“, der in der physikalischen Realität aus ganz unterschiedlichen Kanälen bestehen kann: z.B. aus einem asynchronen seriellen Kanal, aus einem PWM-Ausgang, einer parallelen Schnittstelle oder einem ganz anderen Typ.

Die physikalischen Eigenschaften eines I/O-Devices werden weitgehend im Device-Treiber definiert und durch die Instruktion **INSTALL_DEVICE #Nr, Name** dem BASIC-Programm verfügbar gemacht. Um eine Ein- oder Ausgabe auf ein physikalisch anderes Gerät zu leiten, ist lediglich ein anderer Device-Treiber zu wählen oder ein Parameter in der **INSTALL_DEVICE**-Instruktion zu ändern.

Es ist die Aufgabe der Device-Treiber dem Programmierer das Leben zu erleichtern. An Stelle umfangreicher eigener Programmierungen zur Ansteuerung von I/O-Geräten kann sich der Programmierung auf die Bedienung des jeweiligen Gerätes konzentrieren. Übertragungsspezifische Details wie Puffer verwalten, Strobe-Signale erzeugen und auswerten, physikalische Adressen und Zeitverhalten handhaben, werden vom Device-Treiber übernommen. Der augenblickliche Satz von Device-Treibern wird laufend erweitert. Für spezielle Anforderungen können spezielle Treiber entwickelt werden.

Als ein Beispiel für die Vereinfachung durch Device-Treiber kann man sich den Treiber 'LCD1.TDD' ansehen, der die Ansteuerung eines LCD-Displays und einer Tastatur-Matrix mit bis zu 128 Tasten, Shift-LED und Piepser übernimmt. Der Treiber verwaltet nicht nur die normale Ansteuerung dieser Geräte einschließlich gepufferter Ein- und Ausgabe, es stehen auch eine Reihe leistungsfähiger ESC-Sequenzen zur Verfügung, mit denen man z.B. Tasten-Codes, Wiederholrate, Tasten-Klick, Attribute, Sonderzeichen etc. individuell einstellen kann. In seiner Gesamtheit ersetzt dieser Treiber mehrere 1000 Zeilen BASIC-Code.

Der Einsatz dieses Treibers wird u.a. in den Applikations-Programmen 'ANA1_DEM.TIG' und 'LCD_SPC2.TIG' veranschaulicht. (siehe 'Applikationen' im Device-Treiber-Handbuch).

Funktionen

Die ständig wachsende Bibliothek von Funktionen bildet ein mächtiges Tool, um Programmieraufgaben effektiv und mit wenigen Instruktionen zu realisieren. BASIC-Tiger[®]-Funktionen decken die Bereiche

- Integer-Arithmetik
- Floating-Point-Arithmetik
- String-Operationen
- Spezial-Funktionen

ab. Immer wiederkehrende Programmieraufgaben können auf fertige Funktionen zurückgreifen, die hohe Verarbeitungsgeschwindigkeit und kompakten Code erreichen.

Die 32-Bit-Integer-Arithmetik des BASIC-Tigers zeichnet sich durch hohe Geschwindigkeit und eine für viele Anwendungen bei weitem ausreichende Genauigkeit aus. Bei entsprechender Abbildung kann mit dem verfügbaren Wertebereich von -2.147.483.648 ... +2.147.483.647 z.B. eine Skala von -20.000 ... +20.000 mit einer Auflösung von 0,000 01 (= 1/100.000) aufgebaut werden! Dies können Meßwerte wie Temperatur, Druck, Weg, Geschwindigkeit und vieles mehr sein, wie sie in Forschungsprojekten und Steuerungsaufgaben immer wieder vorkommen. Zu den Funktionen im Bereich Integer Arithmetik zählen u.a. EXP, LD, MOD, SGN, ABS, RND, BIT, MASK, IMASK, LREAL, HREAL, LLTOR, LEN, LEN_FIFO, FREE_FIFO, READ_FIFO, etc.

Floating-Point-Arithmetik arbeitet im BASIC Tiger mit doppelter Genauigkeit (15-16 relevante Stellen), was auch hohe, wissenschaftliche Ansprüche abdeckt. Zudem stehen eine Reihe wichtiger Funktionen für komplexe Berechnungen zur Verfügung, u.a. SIN, COS, TAN, COT, ASIN, ACOS, ATAN, ACOT, SINH, COSH, TANH, COTH, LOG, LN, EXP, EXPE, SQRT, etc.

Bei den String-Funktionen wurde zu dem üblichen Funktionsumfang wie CHR\$, LEFT\$, RIGHT\$, MID\$, etc. eine Reihe starker Funktionen hinzugefügt, die eine sehr elegante und schnelle Programmierung anspruchsvoller Aufgaben erlauben. Hierdurch ist es möglich, Variablen vom Typ String in einem weit allgemeineren Zusammenhang einzusetzen, als dies traditionellerweise der Fall wäre. Insbesondere das Suchen, Selektieren, Ersetzen, Füllen, Fragmentieren und Konvertieren ist mit den neuen String-Funktionen ebenso schnell programmiert, wie diese Programme durch hohe Ablaufgeschwindigkeiten beeindrucken. Zu den neuen String-Funktionen zählen u.a. UPPER\$, CONVERT\$, NTO\$, RTOS\$, STOS\$, NFROM\$, RFROM\$, SELECT\$, INDEX, REMOVE\$, REMDOUBLES\$, STRI\$, etc.

Und schließlich gibt es spezielle Funktionen aus dem systemnahen Bereich die diverse Status-Informationen, Prozeßzeit, Versions-Nr, Fehler-Informationen etc. liefern.

Bevor Sie beginnen

1

Leere Seite

Bevor Sie beginnen	1
Installation	2
Entwicklungsumgebung	3
BASIC-Tiger [®] -Steuercomputer	4
Häufig gestellte Fragen - Support	5
Stichwortverzeichnis	6
Anhang	7

Leere Seite

2 Installation

Bevor Sie mit der Installation beginnen, lesen Sie bitte in der LIESMICH-Datei auf der Installations-CD nach, welche aktuellen Informationen oder Änderungen es seit Drucklegung dieses Handbuchs gibt.

Zum Lieferumfang des BASIC-Tiger[®] Entwicklungssystem gehört ein umfassendes Softwarepaket, das unter MS Windows[™] läuft. Dieses Softwarepaket enthält eine Entwicklungsumgebung, in der vom Editor aus der Übersetzungsvorgang (Compilieren), das Programmieren (herunterladen/downloaden) sowie das Debuggen gesteuert wird.

Installation der Entwicklungsumgebung

Beenden Sie zunächst alle Windows-Anwendungen, die zur Zeit geöffnet sind. Legen Sie dann die Installations-CD in das CD-ROM-Laufwerk und warten Sie, bis das Setup-Programm automatisch gestartet wurde. Wenn Setup nicht automatisch startet, dann wählen Sie in der Start-Leiste den Punkt „Ausführen“. Tragen Sie dort als auszuführende Datei „D:SETUP.EXE“ ein, wobei anstelle von ‚D‘ der Laufwerksbuchstabe Ihres CD-ROM-Laufwerks steht. Das Setup-Programm meldet sich zunächst mit dem Willkommensfenster.

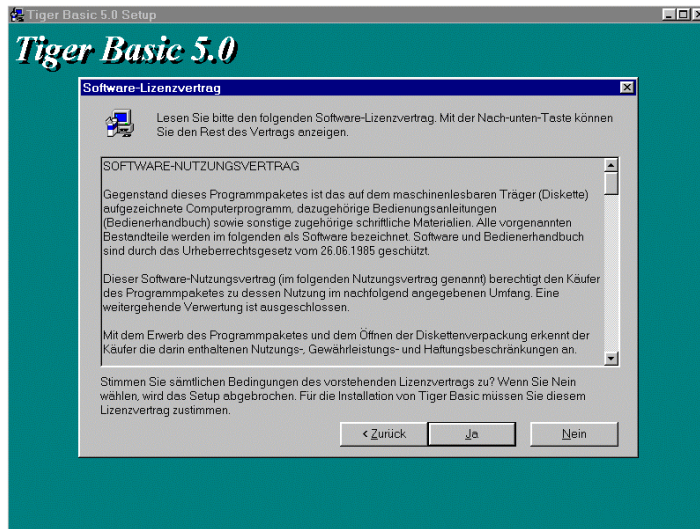


Um mit der Installation fortzufahren, klicken Sie auf „Weiter“.

Installation

2

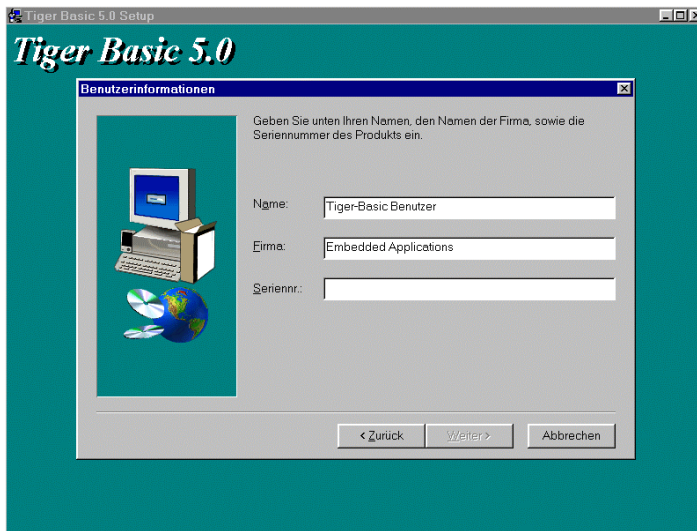
Es erscheint der Lizenzvertrag, dem Sie zur Installation des Programmes zustimmen müssen.



Im folgenden Dialog befinden sich drei Eingabefelder für die Benutzer-Informationen und die Serien-Nummer. Die Benutzer-Informationen sind eventuell bereits mit Angaben aus der Registry Ihres Rechners ausgefüllt. Die Serien-Nummer finden Sie auf der Innenseite oder Rückseite der CD-Hülle.

Haben Sie die Version 5 als Update erhalten, dann verwenden Sie Ihre bisherige Serien-Nummer. Diese kann unter anderem mit dem Befehl „Info über Tiger“ in dem Menü „Hilfe“ nachgesehen werden.

Achten Sie bei der Eingabe der Serien-Nummer auf den Unterschied zwischen „Null“ und „Oh“ sowie „Eins“ und „eL“ oder großem „ih“.



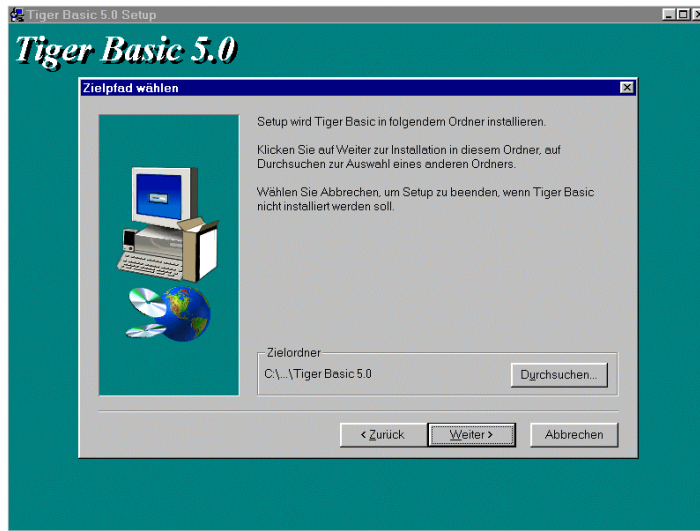
Klicken Sie auf „Weiter“, wenn Sie die Felder ausgefüllt haben.

Installation

2

Sie müssen nun das Unterverzeichnis festlegen, wo Tiger-BASIC® Version 5 installiert werden soll. Da Sie eine eventuell vorhandene Version 4 behalten wollen, wählen Sie ein neues Verzeichnis für die Version 5. In den meisten Fällen ist das vorgeschlagene Verzeichnis „C:\PROGRAMME\Tiger Basic 5.0“ die beste Wahl.

Unter dem ausgewählten Verzeichnis werden weitere Unterverzeichnisse automatisch eingerichtet, in denen sich neben den Compilerdateien auch hunderte von Beispielen befinden.



Klicken Sie anschließend auf „Weiter“.

Für das neue Programm werden neue Symbole (Icons) eingerichtet. Sie können den Ordernamen für die Ablage der Symbole akzeptieren, einen neuen Namen eingeben oder einen aus der Liste wählen.



Klicken Sie anschließend wieder auf „Weiter“.

Installation

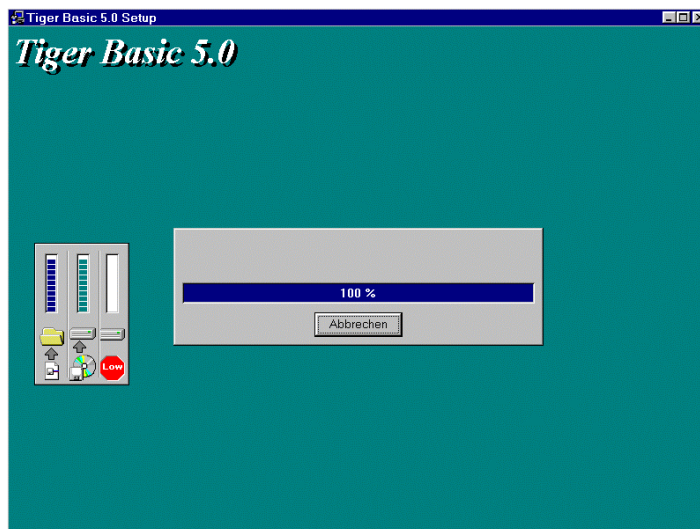
2

Bevor der Kopiervorgang beginnt, zeigt Setup in einer Übersicht alle Angaben, die Sie im Laufe der Installation gemacht haben.



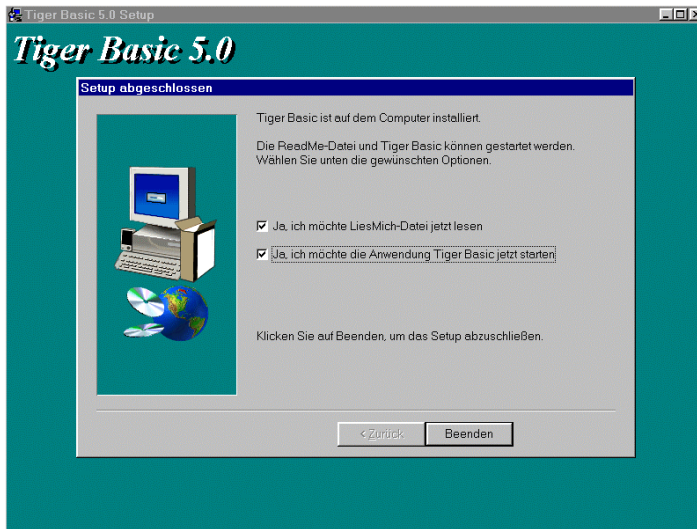
Wenn Sie diese Angaben verändern wollen, klicken Sie auf „zurück“. Wenn Sie mit den Einstellungen zufrieden sind, klicken Sie auf „Weiter“.

Setup kopiert nun alle benötigten Dateien in die Zielverzeichnisse.



Wurde die Installation erfolgreich durchgeführt, erscheint ein Hinweisbildschirm, in dem Sie zwei Optionen auswählen können:

- Sie möchten die Liesmich-Datei lesen. Das sollten Sie tun, denn dort können Änderungen und Korrekturen dokumentiert sein, die nach der Drucklegung der Handbücher stattgefunden haben.
- Sie möchten Tiger-BASIC® sofort starten.



Verlassen Sie mit „Beenden“ diesen letzten Dialog von Setup.

Um Tiger-BASIC® zu starten, verwenden Sie das neu eingerichtete Symbol unter „Start/Programme/Wilke Technology“.

Beispiele

Bei der Software-Installation werden die Programmdateien in ein Unterverzeichnis kopiert und dabei entpackt. Außerdem werden mehrere Verzeichnisse mit Programm-Beispielen eingerichtet, die für den schnellen Einstieg in Tiger-BASIC® nützlich sind:

Examples

im Verzeichnis „Examples“ finden Sie kurze Beispiele zu jeder Instruktion und Funktion. Diese Beispiele sind größtenteils mit den im Handbuch abgedruckten identisch. Sie dienen dem Verständnis der jeweiligen Instruktion oder Funktion und sind in der Regel trotz ihres minimalen Umfangs bereits komplette Programme, die direkt kompiliert und ausgeführt werden können. Wenn Sie rasch verstehen wollen, wie eine Instruktion oder Funktion genau funktioniert, probieren Sie einfach das

mitgelieferte Beispiel und nehmen versuchsweise Veränderungen vor.

Die Dateinamen der Beispiele in diesem Verzeichnis sind jeweils identisch mit der Instruktion/Funktion. Sie finden die Dateinamen auch hier im Handbuch jeweils gleich zu Beginn des abgedruckten Beispiels.

Graphic_Examples

Dem Thema Grafik ist im Handbuch ein eigenes Kapitel gewidmet. Die Grafikbeispiele dieses Kapitels befinden sich im Verzeichnis „Graphic_Examples“. Siehe auch: LCD-Kit.

DeviceDriver_Examples

Im Verzeichnis „DeviceDriver_Examples“ befinden sich die meisten Beispiele des Device-Treiber-Handbuches. Hier finden Sie Beispiele, wie in Tiger-BASIC® Geräte angesprochen werden. Geräte sind ebenfalls die internen Elemente wie serielle Schnittstellen, pulsweiten-modulierte Ausgabe, analoge Eingänge sowie spezielle Funktionen an einfachen digitalen I/O-Pins wie z.B. Pulsausgabe, Pulslängenmessung, Frequenzmessung, Drehgeber-Eingang, etc.

CAN_Examples

CAN als Industrie-Bus und Automatisierungs-Bus benötigt ein spezielles Tiger-BASIC®-Modul. Die Beispiele des CAN-Device-Treibers finden Sie im Verzeichnis „CAN_Examples“ zusammengefaßt.

Applications

Im Verzeichnis „Applications“ werden Anwendungsprogramme gesammelt, die bereits um einiges umfangreicher ausfallen. Sie demonstrieren jeweils bestimmte Programmiertechniken, die Verwendung von Device-Treibern oder andere Anwendungen in einer beispielhaften Art. Applikationen sind ein guter Einstieg, um eigene Entwicklungen zu beginnen und schon frühzeitig lauffähige Zwischenergebnisse zu erzielen. Die Art und Anzahl von Applikationen wird laufend erweitert, insbesondere auch Applikationen, die als ‘Black Box’ eingesetzt werden können, um bestimmte Aufgaben zu lösen. Oft reichen schon kleinere Änderungen an solchen Applikationen aus, um ganz speziell zugeschnittene Lösungen für ein Projekt zu erhalten. Die Applikationen sind nur teilweise im Handbuch abgedruckt und beschrieben. Schauen Sie in den kommentierten Source-Code einer Applikation, um sich einen Überblick zu verschaffen.

LCD-Kit

LCD-Kit ist ein separat erhältliches Zusatzpaket mit einer Hardware zur Grafik-Anwendungsentwicklung. Im Verzeichnis „LCD-Kit“ befinden sich die etwas umfangreicheren Grafik-Beispiele des Grafik-Kits.

Include

Alle Anwendungen greifen auf symbolische Definitionen zurück, um den Quelltext etwas verständlicher zu gestalten. So erhalten Device-Treiber Befehle in Form einer Zahl, in Tiger-BASIC® User-Function-Code genannt. Wenn anstelle der Zahl eine symbolischer Ausdruck geschrieben wird, ist im Quelltext sofort verständlich, was für ein Kommando an den Treiber gegeben wird. Solche symbolischen Definitionen sind in den Include-Dateien zusammengefaßt und diese werden zur Compilationszeit mit eingebunden.

Es gibt Include-Dateien zu verschiedenen Geräten. Wenn ein Gerät in einer Anwendung vorhanden ist, dann wird die zugehörige Include-Datei mit eingebunden und I/O-Adressen, Kommandos, Größen sind ab sofort symbolisch zu bezeichnen. Include-Dateien können auch immer wieder verwendete Unterprogramme enthalten, wie z.B. das Unterprogramm zum Initialisieren des Device-treibers für die Tastatur des Plug & Play Labs, der Hardware-Entwicklungs-Plattform des BASIC-Tiger®. In der folgenden Tabelle einige der wichtigsten Include-dateien und deren Inhalt:

Include-Datei	Zweck
DEFINE_A	allgemeine Definitionen: Gerätenummern der Device-Treiber Baudraten, Konstanten, Fehlernummern.
UFUNC3	User-Function-Codes = Funktionsaufrufe an die Device-Treiber symbolische Bezeichnungen für Parameter.
KEYB_PP	Unterprogramm zur Anpassung des LCD1- Treibers an die Tastatur des Plug & Play Labs.
GR_TK1	Graphic-Toolkit: Port und Pin-Adressen des Toolkits, Unterprogramm zur Initialisierung der LCD-Steuerpins.
LCD_x	symbolische Größenangaben und Steuerse- quenzen der verschieden Grafik-LCDs x steht für die Typnummer (siehe Device- Treiber LCD-6963).
CAN	symbolische Namen für CAN-spezifische Angaben
CAN-SLIO	Adressen und Register-Bitmasken des CAN- SLIO-Chips.
MF2_xxx	umfangreiche Definitionen und Unterpro- gramme zur Anpassung der MF-II-Tastatur.

BIN

Das Verzeichnis, in dem sich die Compiler-Dateien, die Device-Treiber und die Hilfe befinden. Diese Dateien werden vom Anwender nur indirekt verwendet.

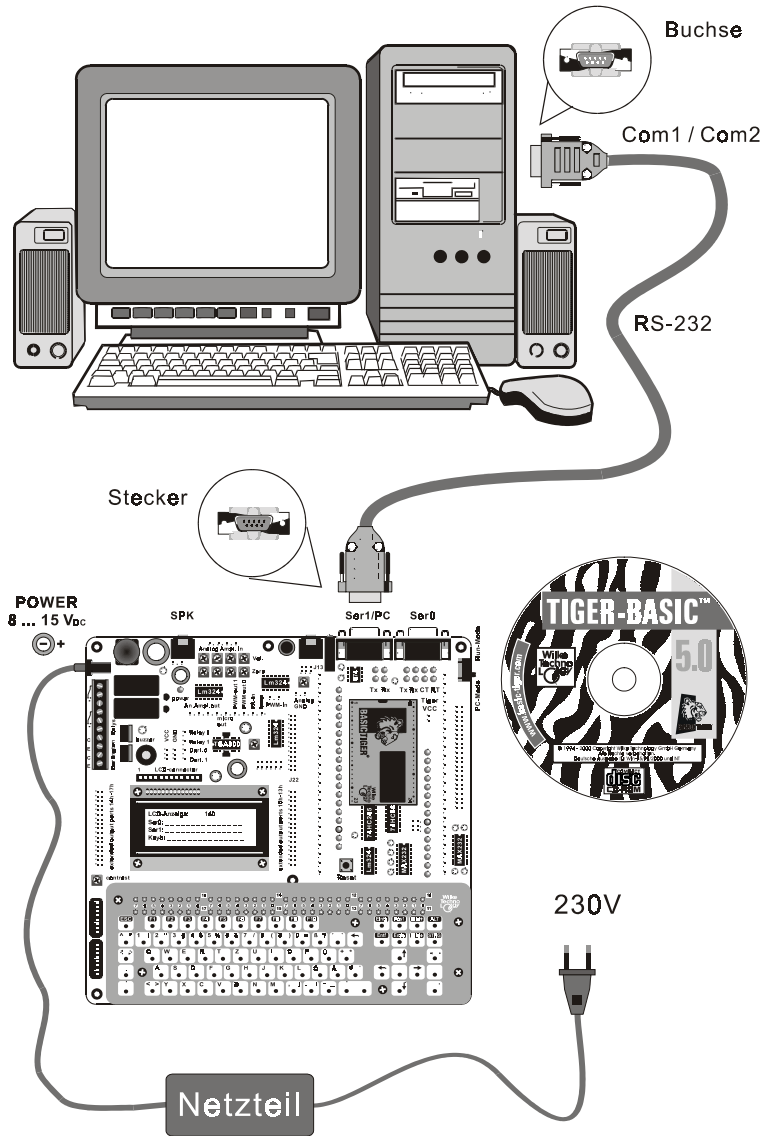
Installation der Hardware

Die Installation der Hardware ist mit wenigen Schritten ausgeführt. Im Laufe der Zeit wird es verschiedene Hardware-Umgebungen zum BASIC-Tiger[®] geben. In der Abbildung ist der Aufbau mit dem Plug & Play Lab dargestellt.

2

Installation

2



- Stellen Sie sicher, daß das Plug & Play Lab noch nicht an das Steckernetzteil angeschlossen ist.
- Stecken Sie das BASIC-Tiger[®]-Modul in den Sockel des Plug & Play Labs (immer bei ausgeschaltetem Plug & Play Lab). Der Pin 1 liegt in der linken oberen Ecke.
- Verbinden Sie das Plug & Play Lab über das serielle Kabel mit einem COM-Port am PC.
- Stellen Sie sicher, daß das Steckernetzteil auf 9 Volt eingestellt ist und die Polarität am Stecker 'außen Plus' ist. Bei falscher Polarität funktioniert das Plug & Play Lab nicht, es wird jedoch nichts zerstört.
- Schließen Sie das Plug & Play Lab an das Steckernetzteil an.
- Starten Sie das Tiger-BASIC[®]-Entwicklungssystem auf Ihrem PC. Sollte die Fehlermeldung „Schnittstelle konnte nicht geöffnet werden“ erscheinen, ignorieren Sie zunächst diese Fehlermeldung und bestätigen mit OK.
- Wählen Sie aus dem Menü **Optionen** den Befehl **Übertragung** und stellen Sie in dem Dialogfenster den COM-Port ein, an dem das Plug & Play Lab angeschlossen ist.
- Wählen Sie aus dem Menü **Optionen** den Befehl **Compiler** und stellen Sie in dem Dialogfenster die Flash- und RAM-Größe Ihres BASIC-Tiger[®]-Moduls ein. (Diese Einstellung ist nur notwendig, wenn Sie ohne angeschlossenes Modul Compilieren wollen.)

Damit ist die Installation der Hardware beendet.

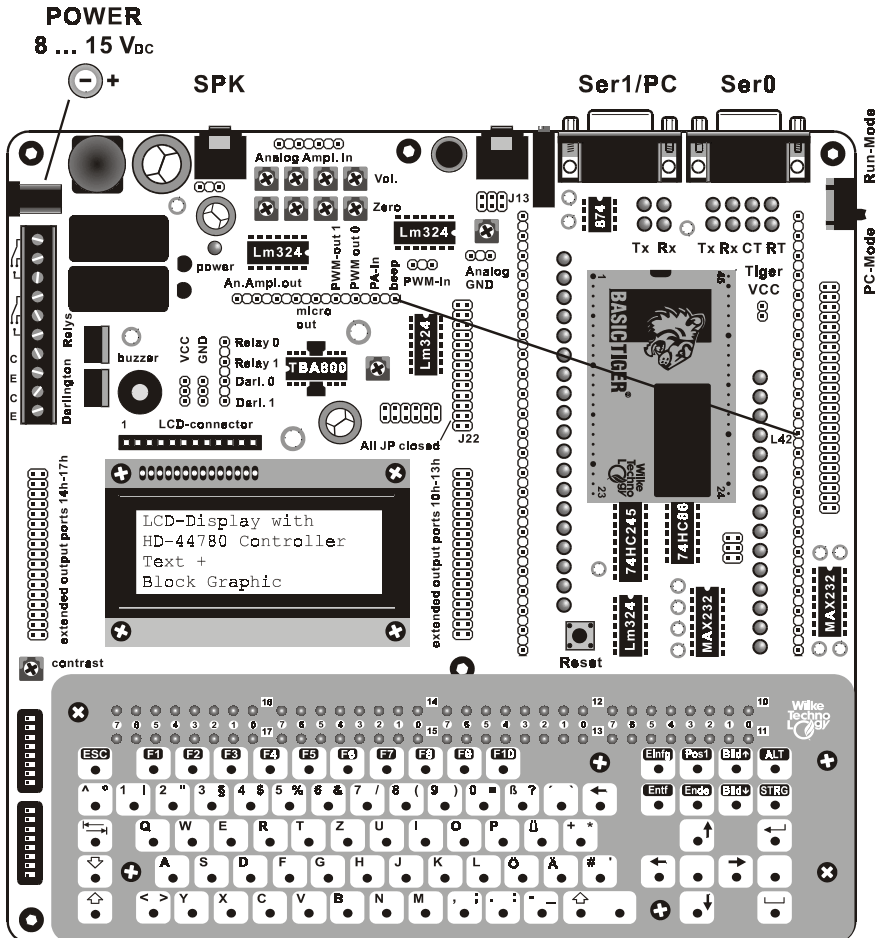
Blitzstart / Erste Schritte

Direkt nach der Installation der Hardware und der Software können Sie mitgelieferte Applikationen ausprobieren.

Die folgenden Beispiele erleichtern Ihnen den Einstieg in die Programmierung des BASIC-Tigers[®].

Im Abschnitt 'Applikationen' finden Sie weitere Beispiele, die Ihnen die Möglichkeit bieten, durch Abwandlung und Zusammenfügen verschiedener Programmabschnitte schnell zu eigenen Anwendungen zu kommen.

Programm „Hello World“



Die 13 Jumper stellen die Verbindungen zum LC-Display und der Tastatur her, die Kabel-Steckbrücke schließt den Summer (beep) an Portpin L42 an.

Installation

2

Dieses Programm schreibt die Begrüßungsmeldung. „Hello World“ auf ein LC-Display. Auf dem Plug & Play Lab müssen die Jumper auf J22 stecken. Auf dem Prototyping-Board stecken sie auf der 2-reihigen Stiftleiste, welche mit 'keybdisplay-I/O' beschriftet ist. Weitere Hinweise für den eigenen Aufbau finden Sie unter der Überschrift 'LC-Display anschließen' im Device-Treiber-Handbuch.

Laden Sie die Applikation mit dem Befehl **Öffnen** aus dem Menü **Datei**. Sie finden das Programm unter dem Namen **HELLO.TIG** im Unterverzeichnis **Examples**, welches sich im Installationsverzeichnis befindet.

Sorgen Sie dafür, daß sich das Modul im PC-Mode befindet:

- Schiebeschalter am Plug & Play Lab in Stellung PC-Mode.
- Modul hat einen Reset erhalten.

Starten Sie das Programm mit dem Befehl **Ausführen** aus dem Menü **Starten**. Nachdem das Programm compiliert worden ist, klicken Sie auf **OK**. Das compilierte Tiger-BASIC®-Programm wird daraufhin in das Modul geladen und automatisch gestartet.

Programmbeispiel:

```
'-----  
' Name: HELLO.TIG  
'-----  
TASK MAIN                                ' Beginn Task MAIN  
' LCD-Treiber installieren (BASIC-Tiger)  
  INSTALL DEVICE #1, "LCD1.TDD"  
' LCD-Treiber installieren (TINY-Tiger)  
'  INSTALL DEVICE #1, "LCD1.TDD", 0, 0, 0, 0, 0, 0, 80h, 8  
  PRINT #1, "Hello World"                ' Auf LC-Display ausgeben  
END                                       ' Ende Task MAIN
```


Programm „Lauflicht“

Dieses Programm erzeugt ein Lauflicht auf dem Plug & Play Lab. Es leuchten nacheinander die LEDs der I/O-Pins von Port 6, 7 und 8. Eine spezielle Jumperung ist nicht erforderlich.

Laden Sie die Applikation mit dem Befehl **Öffnen** aus dem Menü **Datei**. Sie finden das Programm unter dem Namen **RUN_LED.TIG** im Unterverzeichnis **Applicat**, welches sich im Installationsverzeichnis befindet.

Sorgen Sie dafür, daß sich das Modul im PC-Mode befindet:

- Schiebeschalter am Plug & Play Lab in Stellung PC-Mode.
- Modul hat einen Reset erhalten.

Starten Sie das Programm mit dem Befehl **Ausführen** aus dem Menü **Starten**. Klicken Sie auf **OK**, nachdem das Programm compiliert worden ist. Das compilierte Tiger-BASIC®-Programm wird daraufhin in das Modul geladen und automatisch gestartet.

Installation

2

Programmbeispiel:

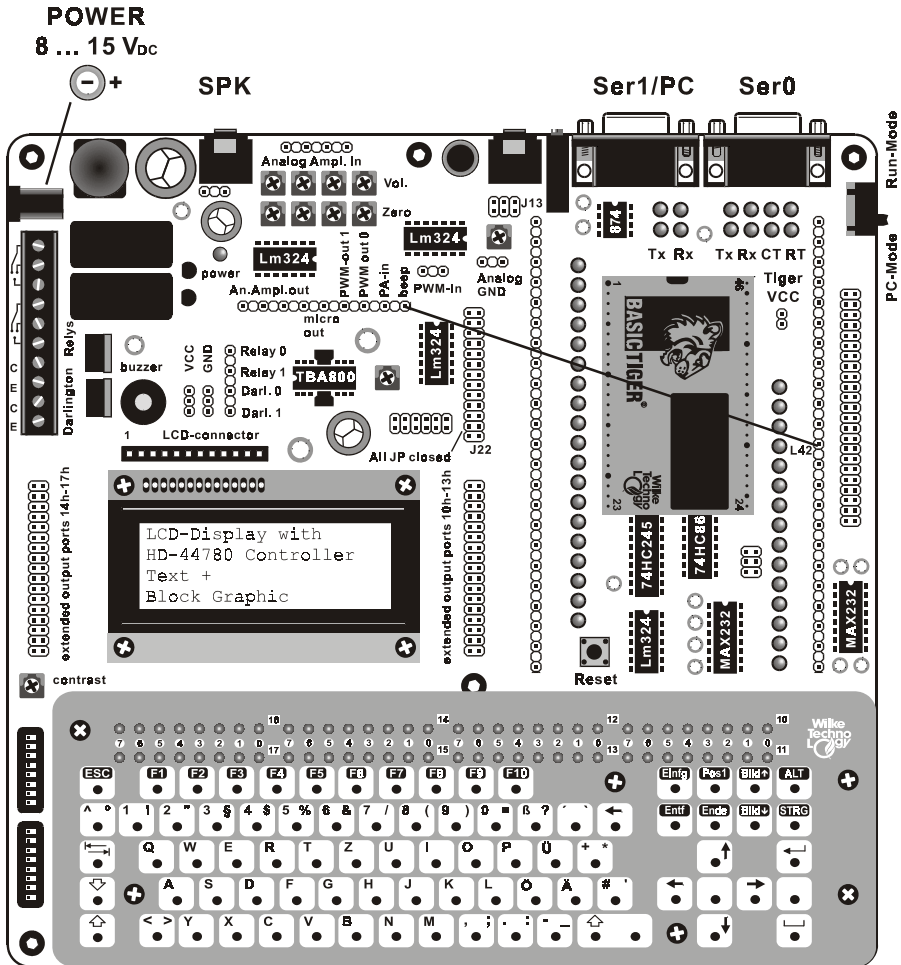
```
'-----
' Name: RUN_LED.TIG
'-----
#include DEFINE_A.INC           ' externe Definitionen
USER_EPOR  ACT, NOACTIVE       ' keine Ext.-Port Aktivitaet

TASK MAIN                       ' Beginn Task MAIN
  BYTE I,J                     ' BYTE-Variablen deklarieren

  DIR_PORT 6,0                 ' Port 6 als Ausgang
  DIR_PORT 7,0                 ' Port 7 als Ausgang
  DIR_PORT 8,0                 ' Port 8 als Ausgang
  FOR I = 6 TO 9               '
    OUT I,255,0                ' Alle LEDs aus
  NEXT                          '
  WHILE 1=1                    ' Endlosschleife
    FOR I = 6 TO 8              ' Ports 6 bis 8
      FOR J = 0 TO 7            ' LEDs 0 bis 7
        IF I <> 7 OR J < 4 THEN  ' Port 7 hat nur 4 Bit
          K = EXP (2,J)         ' LED berechnen
          OUT I, 255, K         ' LED einschalten
          WAIT_DURATION 50      ' 50 ms warten
          OUT I,255,0           ' LED aus
          WAIT_DURATION 50      ' 50 ms warten
        ENDIF                  '
      NEXT                      ' naechste LED
    NEXT                        ' naechster Port
  ENDWHILE                     ' neuer Durchlauf
END                             ' Ende Task MAIN
```

Programm „3Tasks“

2



Die 13 Jumper stellen die Verbindungen zum LC-Display und der Tastatur her, die Kabel-Steckbrücke schließt den Summer (beep) an Portpin L42 an.

Installation

2

Dieses Programm erzeugt auf dem Plug & Play Lab in einer Task einen Klick und läßt in einer anderen Task Leuchtdioden des Ports 8 blinken.

Um den Klick zu hören, muß eine Brücke von Port L42 auf den Pin 'beep' stecken.

Laden Sie die Applikation mit dem Befehl **Öffnen** aus dem Menü **Datei**. Sie finden das Programm unter dem Namen **3TASKS.TIG** im Unterverzeichnis **Examples**, welches sich im Installationsverzeichnis befindet.

Sorgen Sie dafür, daß sich das Modul im PC-Mode befindet:

- Schiebeschalter am Plug & Play Lab in Stellung PC-Mode.
- Modul hat einen Reset erhalten.

Starten Sie das Programm mit dem Befehl **Ausführen** aus dem Menü **Starten**. Klicken Sie auf **OK**, nachdem das Programm compiliert worden ist. Das compilierte Tiger-BASIC[®]-Programm wird daraufhin in das Modul geladen und automatisch gestartet.

Programmbeispiel:

```

'-----
' Name: 3TASKS.TIG
'-----

TASK MAIN                                ' Beginn Task MAIN
' LCD-Treiber installieren (BASIC-Tiger)
INSTALL DEVICE #1, "LCD1.TDD"
' LCD-Treiber installieren (TINY-Tiger)
INSTALL DEVICE #1, "LCD1.TDD", 0, 0, 0, 0, 0, 0, 80h, 8
DIR_PORT 8,0                            ' Port 8 als Ausgang
RUN TASK T2                             ' Task T2 starten
RUN TASK T3                             ' Task T3 starten
LOOP 9999999                            ' sehr viele Loops
    PRINT #1, "<0>";                      ' Ton-Ausgabe: Klick
    WAIT DURATION 2000                  ' 2 Sek warten
ENDLOOP
END                                      ' Ende Task MAIN

'-----
' Task T2: Zeigt 2x pro Sekunde eine Laufvariable an
'-----

TASK T2                                ' Beginn Task T2
LONG L                                ' LONG-Variable deklarieren
FOR L=0 TO 9999999                    ' Zaehlschleife fuer L
    PRINT #1, "<1>L ="; L                ' Ausgabe auf LC-Display
    WAIT DURATION 500                  ' 500 ms warten
NEXT                                  ' naechster Wert
END                                    ' Ende Task T2

'-----
' Task T3: Schaltet alle 200ms die LEDs von Port 8 an oder aus
'-----

TASK T3                                ' Beginn Task T3
LOOP 9999999                          ' sehr viele Loops
    OUT 8,255,0                        ' Ext. I/O Port 8 loeschen
    WAIT DURATION 200                  ' 200ms warten
    OUT 8,255,255                      ' Ext. I/O Port 8 setzen
    WAIT DURATION 200                  ' 200ms warten
ENDLOOP
END                                    ' Ende Task T3

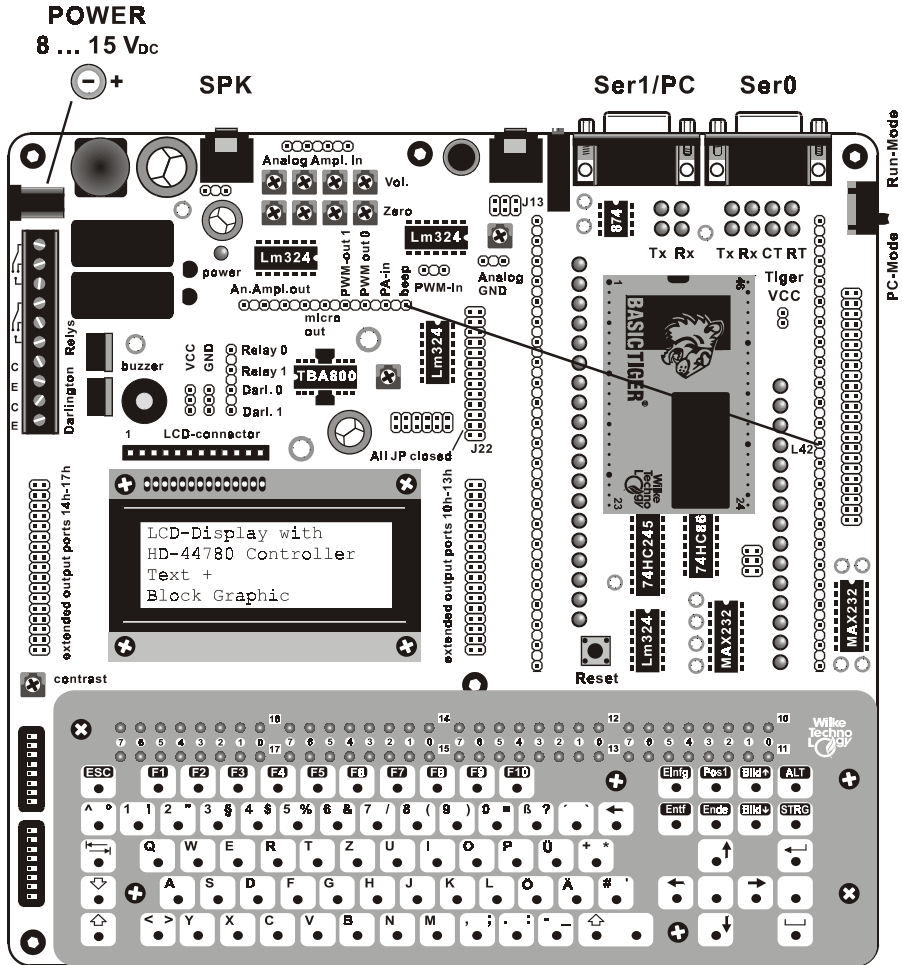
```

Installation

2

Leere Seite

Programm „Taskprio“



Die Jumper stellen die Verbindungen zum LC-Display her, die Kabel-Steckbrücke schließt den Summer (beep) an Portpin L42 an.

Installation

2

Dieses Programm erzeugt zwei Lauflichter auf den Leuchtdioden der erweiterten Ausgänge des Plug & Play Labs, also auf den LED oberhalb der Tastatur.

Eine Task steuert die obere Reihe, die andere Task die untere Reihe LEDs an.

Die Prioritäten der Tasks sind unterschiedlich. Die Lauflichter sind daher unterschiedlich schnell. Die Zahl der Durchläufe wird auf dem LC-Display angezeigt.

Alle 13 Jumper müssen auf J22 stecken.

Laden Sie die Applikation mit dem Befehl **Öffnen** aus dem Menü **Datei**. Sie finden das Programm unter dem Namen **TASKPRIO.TIG** im Unterverzeichnis **Applicat**, welches sich im Installationsverzeichnis befindet.

Sorgen Sie dafür, daß sich das Modul im PC-Mode befindet:

- Schiebeschalter am Plug & Play Lab in Stellung PC-Mode.
- Modul hat einen Reset erhalten.

Starten Sie das Programm mit dem Befehl **Ausführen** aus dem Menü **Starten**. Klicken Sie auf **OK**, nachdem das Programm compiliert worden ist. Das compilierte Tiger-BASIC[®]-Programm wird daraufhin in das Modul geladen und automatisch gestartet.

Programmbeispiel:

```

-----
' Name: TASKPRIO.TIG
-----
WORD ROUNDS_T1, ROUNDS_T2          ' Globale Variablen

TASK MAIN                          ' Beginn Task MAIN
' LCD-Treiber installieren (BASIC-Tiger)
INSTALL DEVICE #1, "LCD1.TDD"
' LCD-Treiber installieren (TINY-Tiger)
' INSTALL DEVICE #1, "LCD1.TDD", 0, 0, 0, 0, 0, 0, 80h, 8
BYTE K                              ' BYTE-Variable deklarieren

ROUNDS_T1 = 0                        ' Variablen initialisieren
ROUNDS_T2 = 0
FOR K = 16 to 23
    OUT K, 255,0                    ' Alle LEDs aus!
NEXT
PRINT #1, "<1bh>c<00h><f0h>";      ' Display-Cursor aus
SET_TASK_PRIO T1,42                 ' Prioritaet der TASK T1
SET_TASK_PRIO T2,123               ' Prioritaet der TASK T2
RUN TASK T1                         ' Task T1 starten
RUN TASK T2                         ' Task T2 starten
WHILE 1=1                           ' Endlosschleife
    PRINT #1, "<1>T1 ="; ROUNDS_T1  ' Durchlaeufer ausgegeben
    PRINT #1, "T2 ="; ROUNDS_T2
    WAIT DURATION 500               ' 500 ms warten
ENDWHILE                            ' Ende Endlosschleife
END                                  ' Ende Task MAIN

-----
' Task T1: Lauflicht auf Ports 10, 12, 14 und 16
-----
TASK T1                            ' Beginn Task T1
LONG I, K                          ' Variablen deklarieren
BYTE Address, VALUE
I = -1                              ' Variablen initialisieren
Address = 16
WHILE 1=1                           ' Endlosschleife
    ' WAIT_DURATION als Verzoeigerung wuerde in diesem Fall
    ' beide Tasks synchronisieren, da WAIT_DURATION CPU-Zeit
    ' freigibt. Daher wird eine FOR-NEXT Schleife verwendet.
    FOR K = 0 TO 1000               ' verbrauche CPU-Zeit
    NEXT
    I = I + 1                       ' LED-Nummer
    IF I > 7 THEN
        OUT Address, 255,0          ' vorherige LED ausschalten
        I = 0                      ' neue LED-Nummer
        Address = Address + 2       ' neue Adresse
        IF Address > 22 THEN
            ROUNDS_T1 = ROUNDS_T1 + 1 ' letzte Adresse?
            Runden zaehlen
            Address = 16            ' Adresse von vorn
        ENDIF
    ENDIF
    VALUE = EXP (2,I)               ' Wert fuer LED berechnen
    OUT Address, 255, VALUE         ' und LED einschalten
ENDWHILE                           ' Ende Endlosschleife

```

Installation

2

```
END                                     ' Ende Task T1

' -----
' Task T2: Lauflicht auf Ports 11, 13, 15 und 17
' -----
TASK T2                                ' Beginn Task T2
LONG I, K                              ' Variablen deklarieren
BYTE Address, VALUE                    '
I = -1                                 ' Variablen initialisieren
Address = 17                           '
WHILE 1=1                               ' Endlosschleife
' -----
' WAIT_DURATION als Verzögerung wurde in diesem Fall
' beide Tasks synchronisieren, da WAIT_DURATION CPU-Zeit
' freigibt. Daher wird eine FOR-NEXT Schleife verwendet.
FOR K = 0 to 1000                       ' verbrauche CPU-Zeit
NEXT                                     '
I = I + 1                               ' LED-Nummer
IF I > 7 THEN                            '
    OUT Address, 255,0                  ' vorherige LED ausschalten
    I = 0                              ' neue LED-Nummer
    Address = Address + 2               ' neue Adresse
    IF Address > 23 THEN                 ' letzte Adresse?
        ROUNDS_T2 = ROUNDS_T2 + 1      ' Runden zählen
        Address = 17                   ' Adresse von vorn
    ENDIF                               '
ENDIF                                   '
VALUE = EXP (2,I)                       ' Wert fuer LED berechnen
OUT Address, 255, VALUE                 ' und LED einschalten
ENDWHILE                                ' Ende Endlosschleife
END                                     ' Ende Task T2
```

Bevor Sie beginnen	1
Installation	2
Entwicklungsumgebung	3
BASIC-Tiger [®] -Steuercomputer	4
Häufig gestellte Fragen - Support	5
Stichwortverzeichnis	6
Anhang	7

Leere Seite

3 Entwicklungsumgebung

Die Entwicklungsumgebung des BASIC-Tiger[®]-Systems besteht aus einer Software-Umgebung, die auf Ihrem PC läuft, und einer Hardware-Entwicklungsumgebung, die eine Plattform mit serieller Schnittstelle darstellt, auf der Ihr BASIC-Tiger[®]-Modul steckt. Meist enthält die Hardware-Plattform allerlei nützliche Elemente, die Sie bei der Entwicklung unterstützen, ohne daß Sie erst eine eigene Platine zusammenlöten müssen. So enthält das Plug & Play Lab bereits ein LCD, eine Tastatur, Relais, Treibertransistoren, Analog-Referenzspannung, erweiterte Ausgänge, und vieles mehr (siehe Hardware-Entwicklungsumgebung).

3

Hinweis: die Entwicklungsplattform ist weitestgehend eine passive Schaltung. Die Plattform kann sich weder ‚melden‘ noch bestimmt die Entwicklungsplattform, welche LEDs nach dem Einschalten leuchten. Allein der BASIC-Tiger[®] meldet sich und steuert die LEDs an.

Wenn Ihre Entwicklung beendet ist, haben Sie in der Regel ein fertiges Produkt, welches an Ihre Kunden ausgeliefert wird. Nun besteht in einigen Fällen der Wunsch, daß Sie nachträglich Firmware-Updates an einen Kunden senden. Ihr Kunde benötigt dann ein Programm (**Downloader**), damit er das Update in das Modul laden kann, denn er besitzt ja in der Regel nicht den BASIC-Tiger[®]-Compiler, und schon gar nicht Ihren Source-Code. Näheres über den Downloader erfahren Sie unter der Überschrift Downloader ab Seite 95. Ihre Hardware braucht für die Möglichkeit des nachträglichen Updates

- die serielle Schnittstelle 1 mit RS-232-Pegel für die Verbindung zu einem PC oder Laptop
- eine Umschaltmöglichkeit des PC-Mode-Pins, um das Modul in den PC-Modus zu versetzen. Nützlich ist ein Reset-Knopf, jedoch kann auch aus- und eingeschaltet werden. In Sonderfällen kann die im Modul befindliche Anwendersoftware das Programm selbst löschen, wodurch das Modul automatisch in den PC-Modus geht, ohne den PC-Mode-Pin zu bedienen.

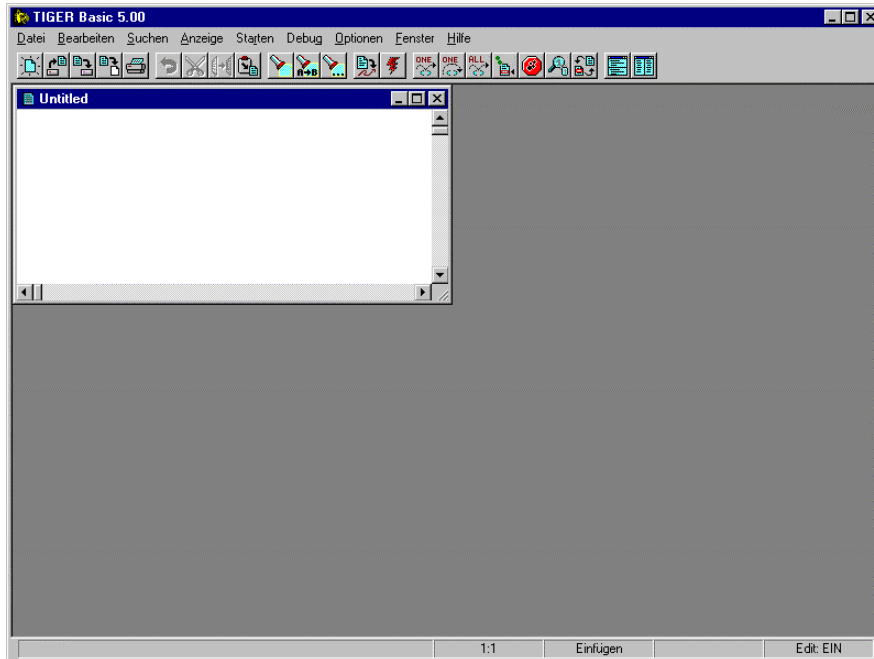
Entwicklungsumgebung

Software-Entwicklungsumgebung

Starten Sie die Entwicklungsumgebung durch einen Doppelklick auf das Icon „Tiger BASIC 5.0“ unter „Start/Programme/Wilke Technology“.

Danach erscheint die Entwicklungsumgebung, in der bei späteren Starts automatisch alle Dateien wieder geöffnet werden, die beim Ende der vorherigen Sitzung geöffnet waren.

3



Im folgenden Teil dieses Abschnitts werden alle Funktionen der Entwicklungsumgebung, nach Menüs geordnet, sowie die Bedienung des Editors erläutert.

Befehle der Entwicklungsumgebung

Zur Befehlseingabe steht Ihnen neben den Menüs sowohl die Bedienung per Tastatur (Steuer- und Funktionstasten) als auch die Bedienung per Maus (Buttons) zur Verfügung. Bewegen Sie einfach den Mauszeiger auf einen Button und lesen die Bedeutung im Quickinfo.

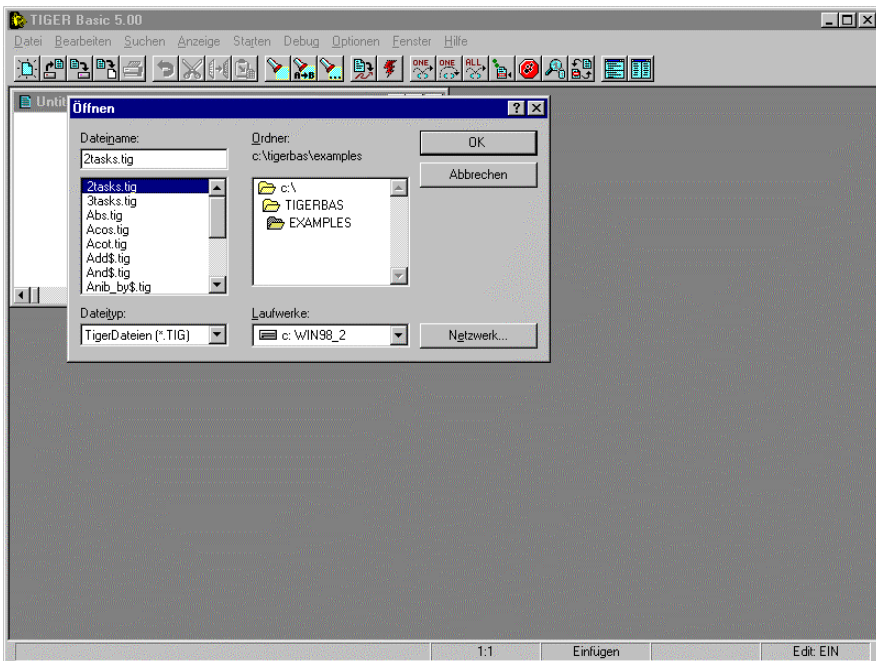
Datei-Menü

Neu

Die Entwicklungsumgebung erzeugt ein neues Fenster zum Eingeben eines neuen Quelltextes. Dieser neue Text hat vorläufig noch keinen Namen (unbenannt)

Öffnen

Im Dialogfenster wählen Sie das Laufwerk, das Unterverzeichnis sowie den Namen der Datei aus und klicken dann auf die Schaltfläche „OK“.



Schließen

Die aktuell bearbeitete Datei wird geschlossen. Sollte der aktuelle Stand noch nicht gesichert sein, findet eine Sicherheitsabfrage statt, in der Sie dies nachholen können.

Entwicklungsumgebung

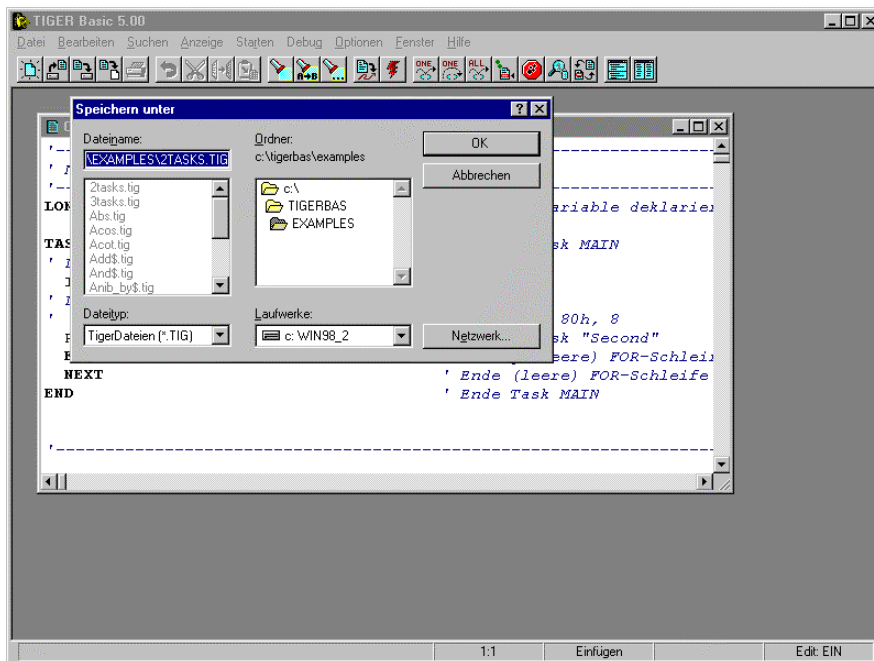
Speichern

Die aktuell bearbeitete Datei wird gespeichert. Ist die Datei bislang noch unbenannt, öffnet sich ein Dialogfenster (siehe „Speichern unter...“).

Speichern unter...

Die aktuell bearbeitete Datei wird unter einem neuen Namen abgespeichert. Im Dialogfenster geben Sie Laufwerk, Verzeichnis und Dateinamen an, unter welchem Ihr Programm gespeichert werden soll. Klicken Sie dann auf die Schaltfläche „OK“.

3



Alles speichern

Wie „Speichern“, allerdings werden alle geöffneten Dateien gespeichert.

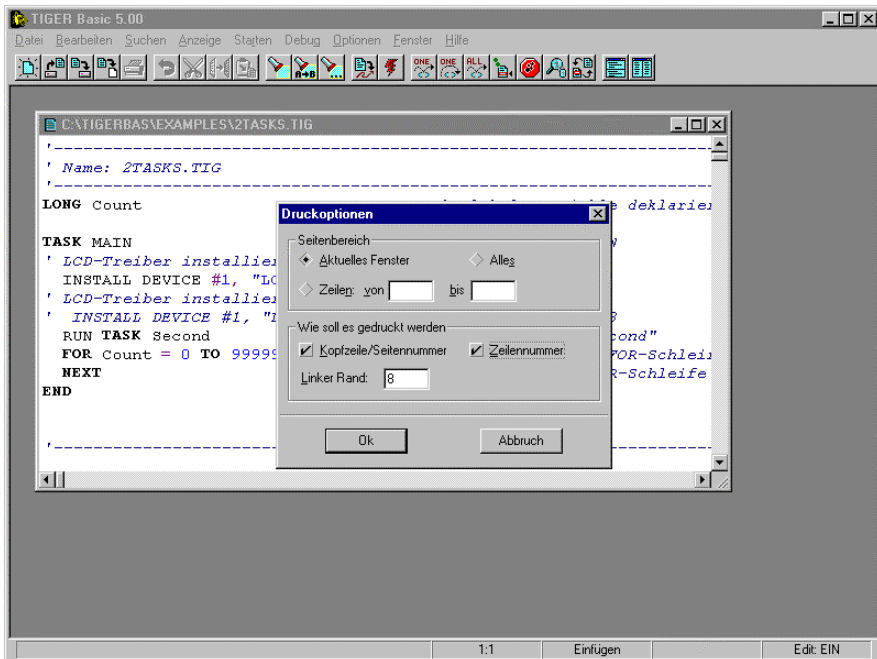
Die Tiger-BASIC®-Entwicklungsumgebung kann beim Speichern von Dateien eine Sicherungsdatei automatisch erstellen. Ob die Sicherungsdatei erstellt wird oder nicht, bestimmen Sie mit dem Befehl **Editor** im Menü **Optionen**. Im Dialogfeld für die Editoroptionen tragen Sie die gewünschte Option ein.

Drucken

Die aktuelle bearbeitete Datei wird komplett oder teilweise ausgedruckt. Die Einstellung „Aktuelles Fenster“ druckt nur den im Editor-Fenster sichtbaren Teil der Datei (wie ein Screen-Shot), die Einstellung „Alles“ druckt die gesamte Datei.

Ist der Schalter „Kopfzeile/Seitennummer“ aktiviert, wird in die erste Zeile jeder Seite der Dateiname und die fortlaufende Seitennummer gedruckt. Mit dem Schalter „Zeilennummer“ kann eine fortlaufende Zeilennummerierung für die gesamte Datei aktiviert werden. Die Einstellung für den linken Rand kann dazu verwendet werden, einen Heftrand beim Ausdruck frei zu lassen.

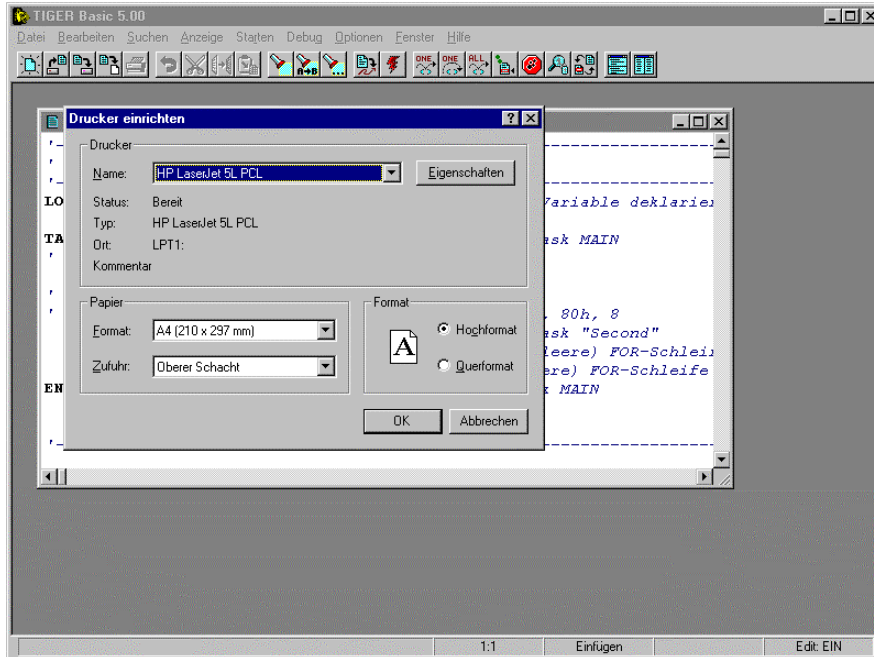
3



Entwicklungsumgebung

Druckereinrichtung

Hier können Sie den Drucker auswählen, auf dem der Ausdruck erfolgen soll, sowie dessen Einstellungen (Papiergröße, Papierzufuhr, Auflösung, Format etc.)



Beenden

Hiermit wird die Arbeit mit der Entwicklungsumgebung beendet. Sollten sich noch ungesicherte Dateien im Editor befinden, können diese nach einer Sicherheitsabfrage gespeichert werden.

Bearbeiten-Menü

Rückgängig machen

Der zuletzt ausgeführte Bearbeitungsschritt wird Rückgängig gemacht. Die Anzahl der Schritte, die rückgängig gemacht werden können, wird im Menü „Optionen“ bei den Editor-Optionen eingestellt. Zu beachten ist, daß Ersetzen von Text eventuell zu sehr vielen Aktionen führt, die einzeln rückgängig gemacht werden. Der Befehl „Verschachtelung hervorheben“ aus dem Menü Bearbeiten kann nicht rückgängig gemacht werden.

Ausschneiden

Ein markierter Text wird aus der bestehenden Position ausgeschnitten und in die Zwischenablage kopiert.

Kopieren

Ein markierter Text wird in die Zwischenablage kopiert.

Einfügen

Der in der Zwischenablage befindliche Text wird an der aktuellen Cursor-Position eingefügt.

Alles markieren

Der gesamte Text im aktuellen aktuellen Fenster wird markiert.

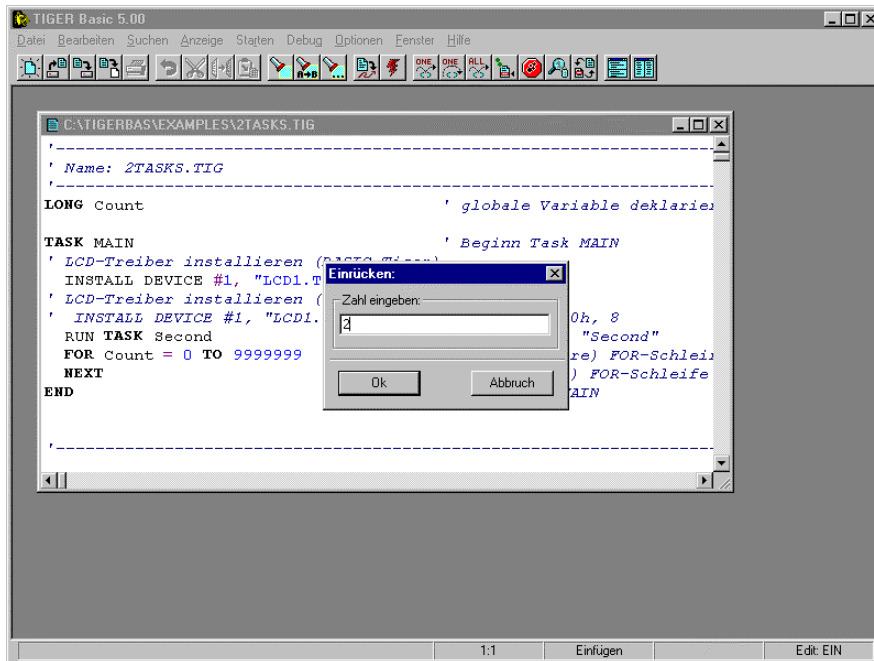
Verschachtelung hervorheben

Hiermit kann eine Quelldatei für bessere Übersichtlichkeit mit von der Verschachtelungstiefe abhängigen Einrückungen versehen werden. Geben Sie die gewünschte Einrückung für die Verschachtelung ein und klicken Sie „OK“. Bei einer Einrückung von 4 wird jede Zeile mit einer neuen Verschachtelung 4 Zeichen weiter nach rechts gerückt: Verschachtelungstiefe 1 wird 4 Zeichen eingerückt, während Verschachtelungstiefe 3 bereits um 12 Zeichen eingerückt wird.

Dieses Kommando verwendet TAB-Zeichen, um die Verschachtelung herbeizuführen. Mit Leerzeichen erzeugte Verschachtelung wird dabei umgewandelt. Dieser Vorgang kann nicht mit dem Kommando „Rückgängig“ wieder aufgehoben werden.

Entwicklungsumgebung

3

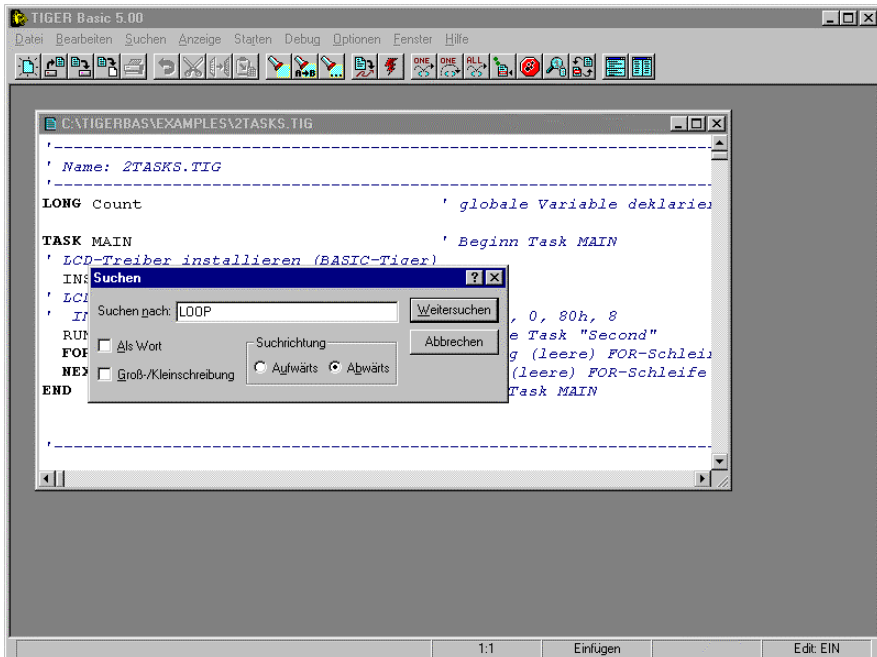


Suchen-Menü

Suchen nach...

Der angegebene Begriff wird ab der aktuellen Cursorposition gesucht. Die Suchrichtung kann aufwärts oder abwärts sein. Zusätzlich können folgende Bedingungen zugeschaltet werden: „Als Wort“ findet den Suchbegriff nur als freistehendes Wort, nicht aber als Teil eines Wortes (z.B. wird der Suchbegriff „und“ nicht innerhalb des Wortes „rund“ gefunden). Außerdem kann eingestellt werden, daß die Groß-/Kleinschreibung beachtet wird („und“ wird gefunden, „UND“ hingegen nicht“).

3

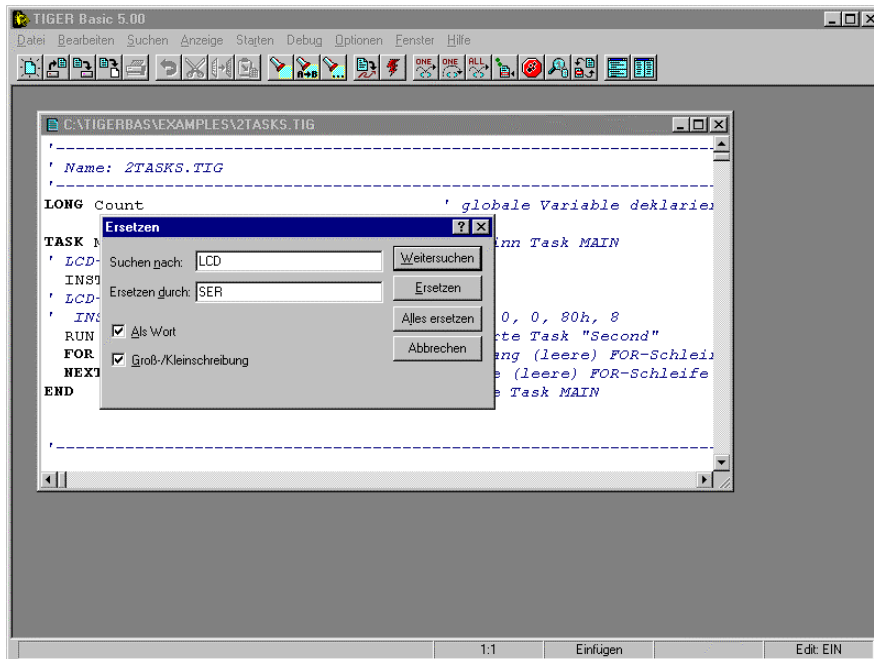


Entwicklungsumgebung

Ersetzen

Der Suchbegriff wird durch einen anderen Begriff ersetzt. Die Schalter „Als Wort“ und „Groß-/Kleinschreibung“ entsprechen denen bei der „Suchen nach...“-Funktion. Durch Klicken auf „Suchen“ wird die Suche aktiviert, die von der aktuellen Cursorposition aus abwärts läuft. Wird der Suchbegriff gefunden, wird er im Editor-Fenster hervorgehoben. Durch Klicken auf „Ersetzen“ wird der Begriff ersetzt und das nächste Vorkommen des Suchbegriffs gesucht. Klicken auf „Suchen“ sucht weiter, ohne das Ersetzen durchzuführen. Der Knopf „Alle ersetzen“ führt das Ersetzen ohne Nachfrage im gesamten Text durch.

3



Suche wiederholen

Wiederholt die letzte durchgeführte Suche. So können nach dem ersten erfolgreichen Suchvorgang alle weiteren Vorkommen des Suchbegriffs bequem aufgespürt werden.

Vorherige Meldung

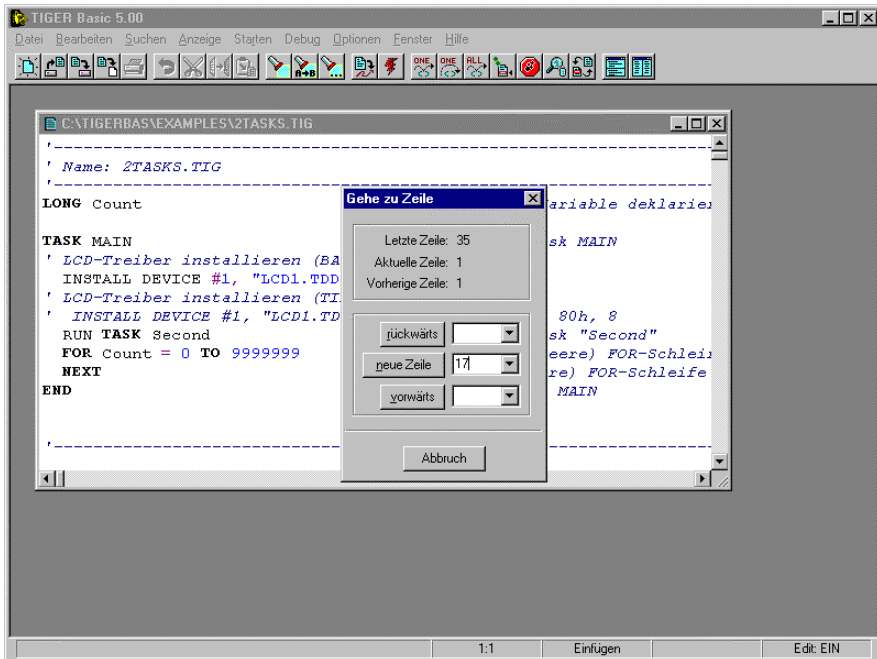
Springt im Meldungsfenster auf die vorherige Meldung und im Editor auf die Zeile, die diese vorherige Fehlermeldung verursacht hat.

Nächste Meldung

Springt im Meldungsfenster auf die nächste Meldung und im Editor auf die Zeile, die diese nächste Fehlermeldung verursacht hat.

Gehe zu Zeile

Durch die Eingabe einer Zeilennummer und Klicken auf „OK“ springt der Editor sofort auf diese Zeile.



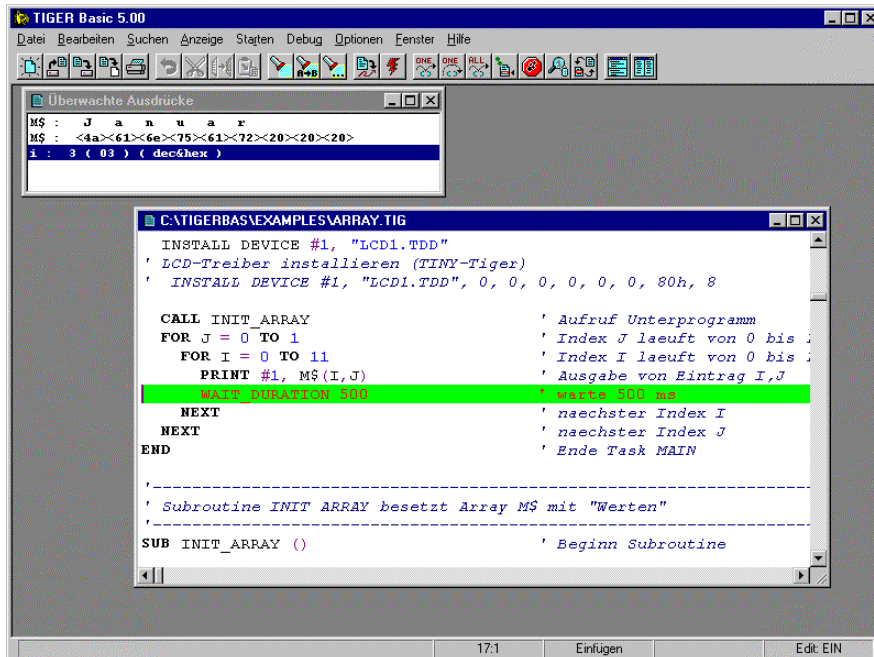
3

Anzeige-Menü

Meldungen

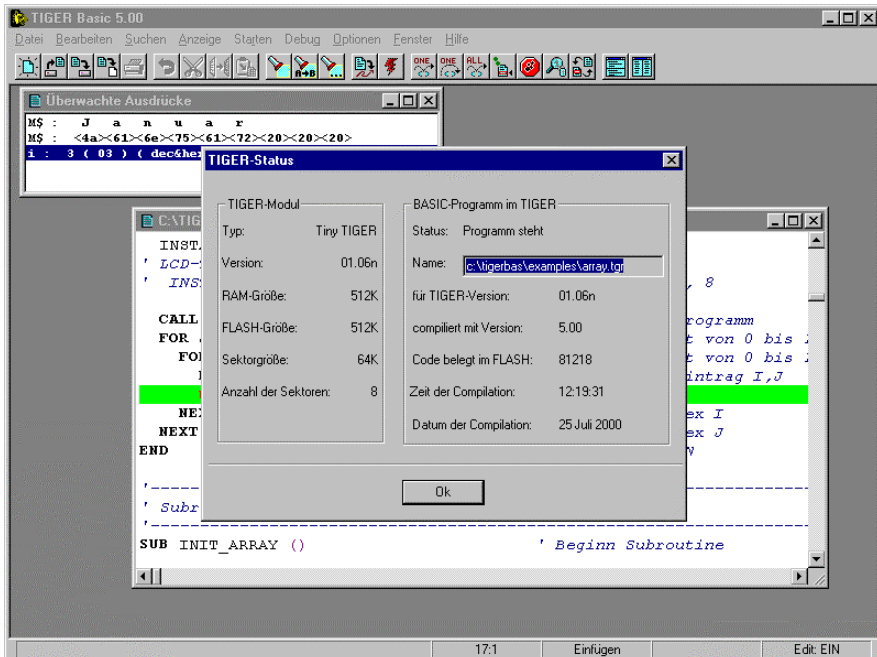
Öffnet das Meldungsfenster. In diesem Fenster, das bei einem Fehler automatisch geöffnet wird, sind alle beim Compilieren aufgetretenen Fehler sowie Warnungen aufgelistet. Die Anzeige besteht zunächst aus dem Wort „Fehler“ bzw. „Warnung“, gefolgt von der Zeile, in der das Problem aufgetreten ist. Dahinter steht in eckigen Klammern die Nummer des Fehlers, gefolgt von der ausführlichen Bezeichnung des Fehlers.

3



Tiger-Status

Hiermit kann eine Statusmeldung ausgegeben werden, die den Angaben über Typ und Inhalt des angeschlossenen Tiger-Moduls macht. So werden z.B. die Version und Speichergößen und Flash-Sektorgrößen des Moduls angezeigt, der Name des im Modul befindlichen Programms, wann es für welche Modulversion mit welcher Compiler-Version compiliert wurde und wieviel Platz das Programm im FLASH und RAM benötigt.

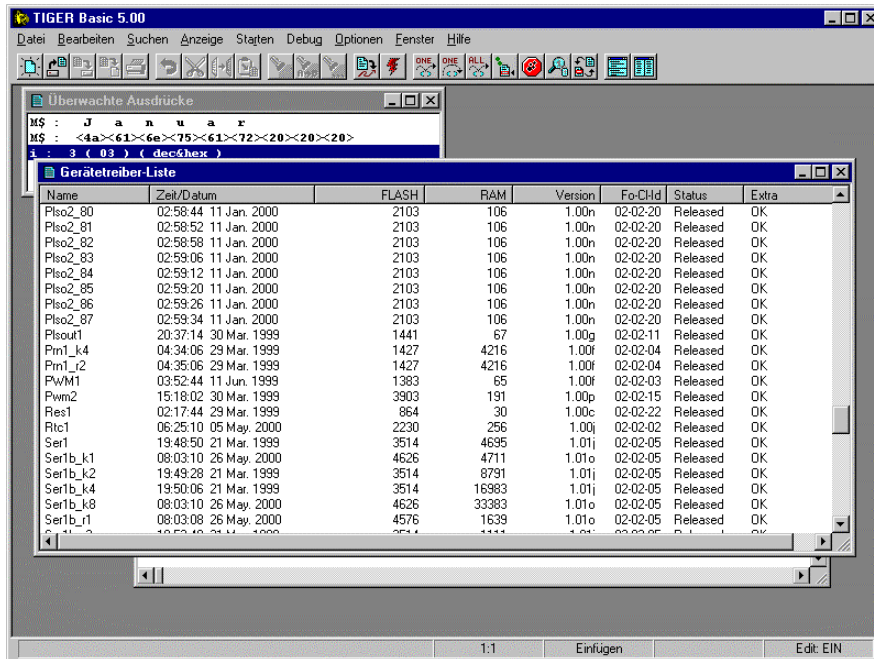


3

Gerätetreiberliste

Entwicklungsumgebung

Hier werden alle zur Verfügung stehenden Gerätetreiber aufgelistet. Neben dem Erstellungsdatum, der Versionsnummer und weiteren Informationen gibt diese Liste auch Aufschluß darüber, wieviel Platz ein Gerätetreiber im FLASH bzw. im RAM beansprucht.



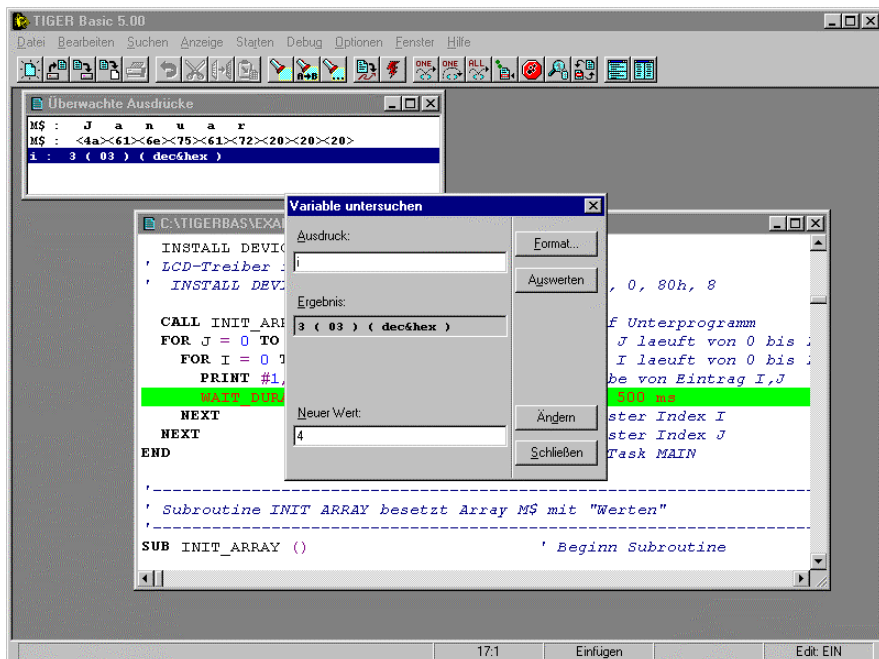
Ausdruck auswerten

Mit diesem Befehl gelangen Sie in ein Dialogfenster, in dem Sie den Wert einer Variablen erfragen und ändern können. Wenn der Cursor beim Aufruf des Dialogs auf einer Variablen steht, dann ist diese direkt im Feld „Ausdruck“ eingetragen. Ein Klick auf den Button „Auswerten“ ist erforderlich, um den Wert der Variablen aus dem Modul auszulesen.

Der Button „Format“ ruft einen Dialog auf, in dem das Darstellungsformat der Variablen eingestellt wird. Die Einstellung wird erst wirksam, wenn erneut der Button „Auswerten“ gedrückt wird.

Numerische Variable: Um den Wert der angezeigten Variablen zu ändern, wird im Eingabefeld „Neuer Wert“ der Wert eingetragen, den die Variable erhalten soll, und durch Klicken auf „Ändern“ übernommen.

3



Überwachte Ausdrücke

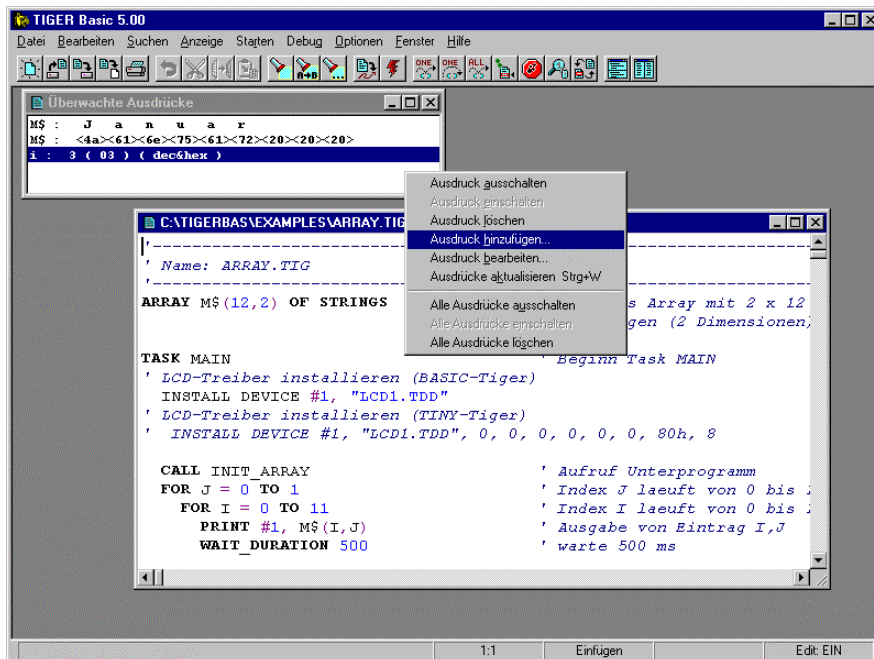
Um Ausdrücke/Variablen ständig zu beobachten, öffnen Sie hiermit das Fenster der überwachten Ausdrücke.

In diesem Fenster erhalten Sie durch einen Klick auf die rechte Maustaste ein Kontextmenü, in dem Sie die Befehle zum Bearbeiten, Hinzufügen, Löschen, Ein- und Ausschalten sowie aktualisieren der zu überwachenden Ausdrücke finden.

Entwicklungsumgebung

Die Befehle des Kontextmenüs der überwachten Ausdrücke im Einzelnen:

Befehl	Beschreibung
Ausdruck ausschalten	Der markierte Ausdruck wird ausgeschaltet und nicht mehr aktualisiert.
Ausdruck einschalten	Der markierte Ausdruck wird wieder eingeschaltet.
Ausdruck löschen	Der markierte Ausdruck wird gelöscht.
Ausdruck hinzufügen	Ein neuer Ausdruck wird der Liste hinzugefügt.
Ausdruck bearbeiten	Ein bestehender Ausdruck wird durch einen anderen ersetzt oder in seiner Erscheinungsform modifiziert.
Ausdruck aktualisieren	Die Liste der Ausdrücke wird zur Laufzeit aktualisiert.
Alle Ausdrücke ausschalten	Alle in der Liste befindlichen Ausdrücke werden ausgeschaltet und nicht mehr aktualisiert.
Alle Ausdrücke einschalten	Alle in der Liste befindlichen Ausdrücke werden wieder eingeschaltet.
Alle Ausdrücke löschen	Alle in der Liste befindlichen Ausdrücke werden gelöscht.



Die Befehle „Ausdruck hinzufügen“ und „Ausdruck bearbeiten“ öffnen ein weiteres Eingabefenster, in dem der Name des zu überwachenden Ausdrucks eingegeben wird. Durch Klicken auf „OK“ wird der Ausdruck in die Liste übernommen, durch Klicken auf „Format“ kann die Darstellung des Ausdrucks individuell angepaßt werden.

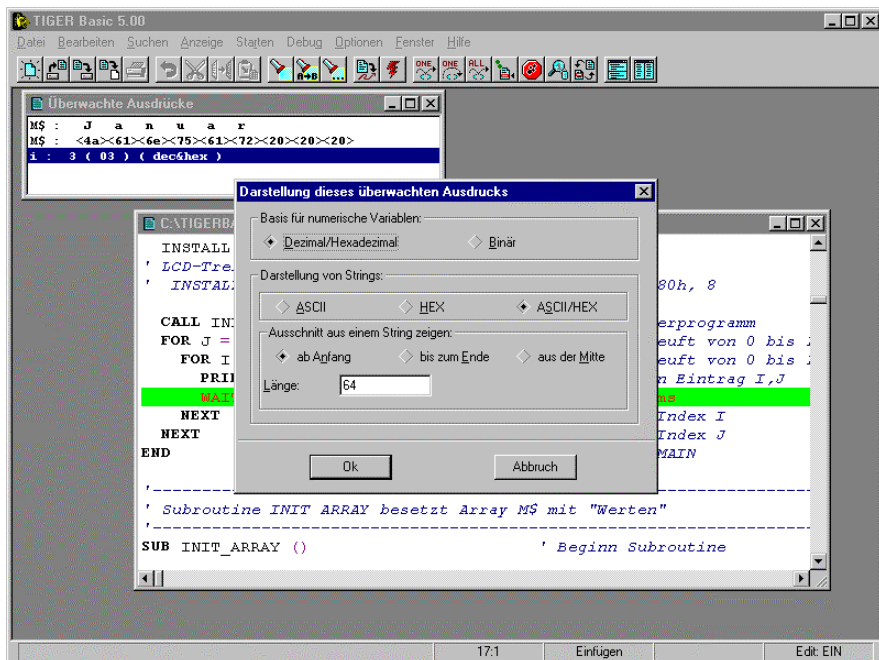
Dabei gibt es folgende Möglichkeiten:

Numerische Variablen/Ausdrücke können „**Dezimal/Hexadezimal**“ oder „**Binär**“ dargestellt werden.

String-Variablen können in „**ASCII**“-Darstellung, in „**HEX**“-Darstellung oder als Kombination „**ASCII/HEX**“ angezeigt werden.

Bei String-Variablen gibt es noch weitere Spezifikationen: Sollen die bei „**der Länge**“ angegebenen Anzahl Zeichen des Strings „**ab Anfang**“ des Strings, „**bis zum Ende**“ des Strings oder „**aus der Mitte**“ des Strings angezeigt werden?

3



Überwachte Ausdrücke aktualisieren

Mit diesem Befehl werden alle Variablen im Fenster „Überwachte Ausdrücke“ manuell aktualisiert.

Im Dialog **Debugger-Einstellungen** aus dem Menü **Optionen** können Sie auch automatisches Aktualisieren nach jedem Einzelschritt einstellen.

Entwicklungsumgebung

FLASH-Daten untersuchen

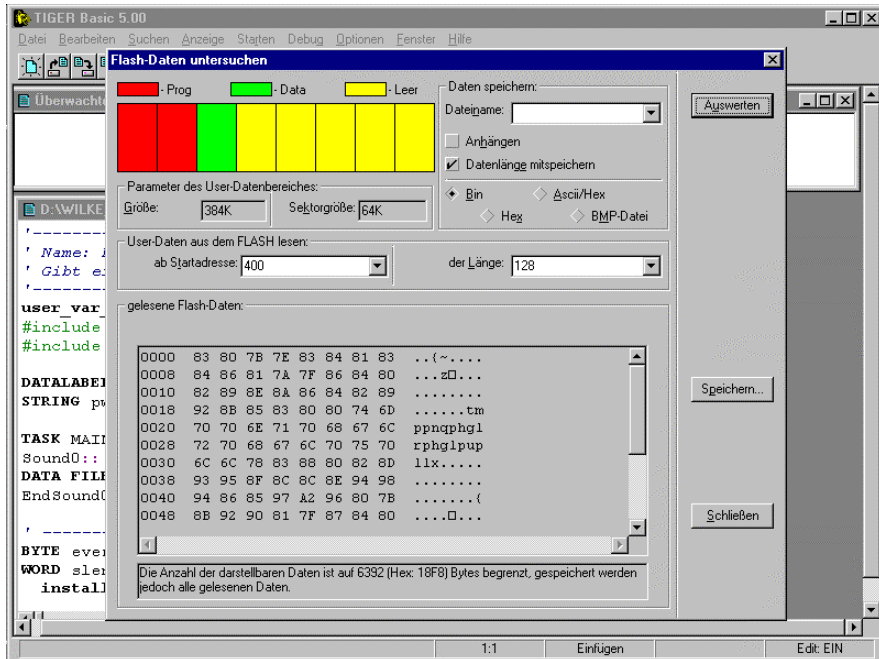
Der Dialog ermöglicht:

- Daten aus dem User-Bereich des Flash-Speichers anzusehen
- Angabe der Anfangsadresse und einer Länge
- Informiert über die Sektorbelegung des Flash-Speichers
Rot = vom Programm belegt
Grün = belegt mit User-Daten
Gelb = frei
- Abspeichern der Daten in verschiedenen Ansichten:
binär, also die Daten so, wie sie sind
HEX, als HEX-Dump ohne ASCII
HEX/ASCII, als HEX-Dump mit ASCII
BMP, als Bitmap-Datei
- Anhängen der Daten an eine vorhandene Datei

Um den Flash-Speicher auszuwerten, muß das Modul im PC-Modus erreichbar sein. Die Oberfläche muß ein Programmfenster mit dem Programm im Modul assoziieren können.

Sie geben eine Startadresse und die Anzahl der Bytes an. Nach Anklicken des Buttons „Auswerten“ werden die Daten aus dem User-FLASH des Tiger-Moduls ausgelesen und angezeigt. Dies kann beim Debuggen hilfreich sein, wenn geklärt werden soll, was an einer Stelle im Flash steht. Voraussetzung ist, daß im aktiven Editorfenster das gleiche Programm ist wie im Modul und das Sie Sich im Debug-Modus befinden.

Die ausgelesenen Daten können in einer Datei gespeichert werden. Dazu ist die Angabe eines Dateinamens sowie die Auswahl des Formats nötig. Die Abspeicherung kann als Datenstrom oder als HEX-Dump erfolgen. Durch Anklicken von „Speichern“ wird die Übertragung gestartet.



3

String untersuchen

Der Dialog ermöglicht:

- Daten aus String anzusehen
- Angabe der Anfangsadresse und einer Länge
- Informiert über die maximale und die aktuelle Länge des Strings
- Abspeichern der Daten in verschiedenen Ansichten:
 - binär, also die Daten so, wie sie sind
 - HEX, als HEX-Dump ohne ASCII
 - HEX/ASCII, als HEX-Dump mit ASCII
 - BMP, als Bitmap-Datei
- Stringlänge mit abspeichern
- Anhängen der Daten an eine vorhandene Datei

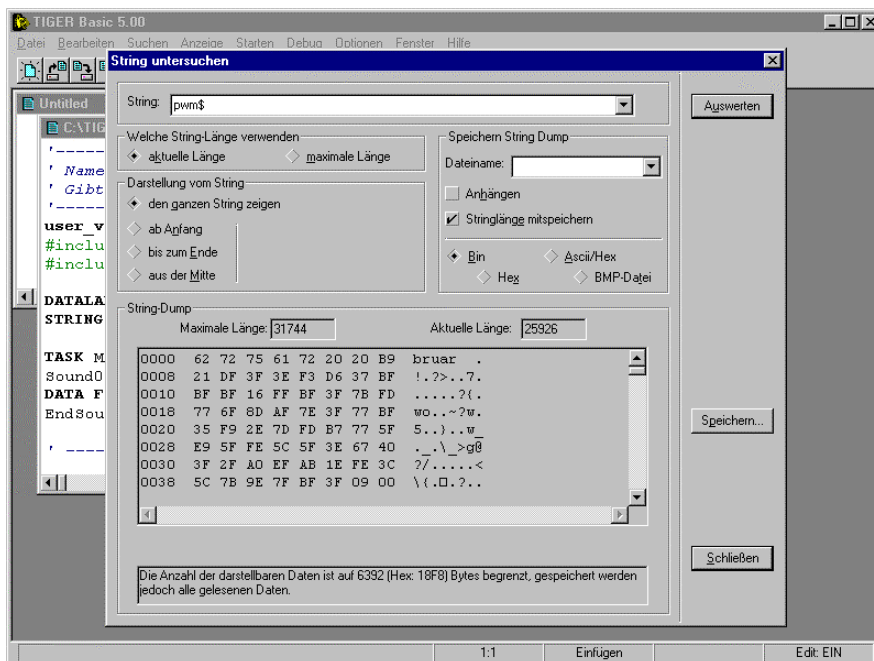
Um einen String auszuwerten, muß das Modul im PC-Modus erreichbar sein. Die Oberfläche muß ein Programmfenster mit dem Programm im Modul assoziieren können.

Entwicklungsumgebung

Sie geben an, ob Sie die aktuelle Länge oder die maximale Länge untersuchen wollen. Die maximale Länge ist im Speicher für den String reserviert. Welche Bytes jedoch initialisiert sind, hängt von der Vorgeschichte ab. Nach Anklicken des Buttons „Auswerten“ werden die Daten aus dem Tiger-Modul ausgelesen und angezeigt. Es kann beim Debuggen hilfreich sein, zu wissen, was an einer Stelle im String steht, die jenseits der aktuellen Länge liegt. So können z.B. die Funktionen STOSS\$, NFROMS, etc. darauf zugreifen

3

Die ausgelesenen Daten können in einer Datei gespeichert werden. Dazu ist die Angabe eines Dateinamens sowie die Auswahl des Formats nötig. Die Abspeicherung kann als Datenstrom (binär), als HEX-Dump mit oder ohne ASCII oder als Bitmap-Datei (z.B. LCD-Grafik-Strings) erfolgen. Durch Anklicken von „Speichern“ wird die Übertragung gestartet.



Starten-Menü

Compilieren

Das im aktiven Fenster befindliche Programm wird compiliert, falls es seit der letzten Compilation verändert wurde. Andernfalls wird der Anwender darüber informiert, daß das Programm bereits compiliert ist, und gefragt, ob erneut compiliert werden soll.. Nach der Compilation wird angezeigt, wieviel Speicherplatz das Programm beansprucht.

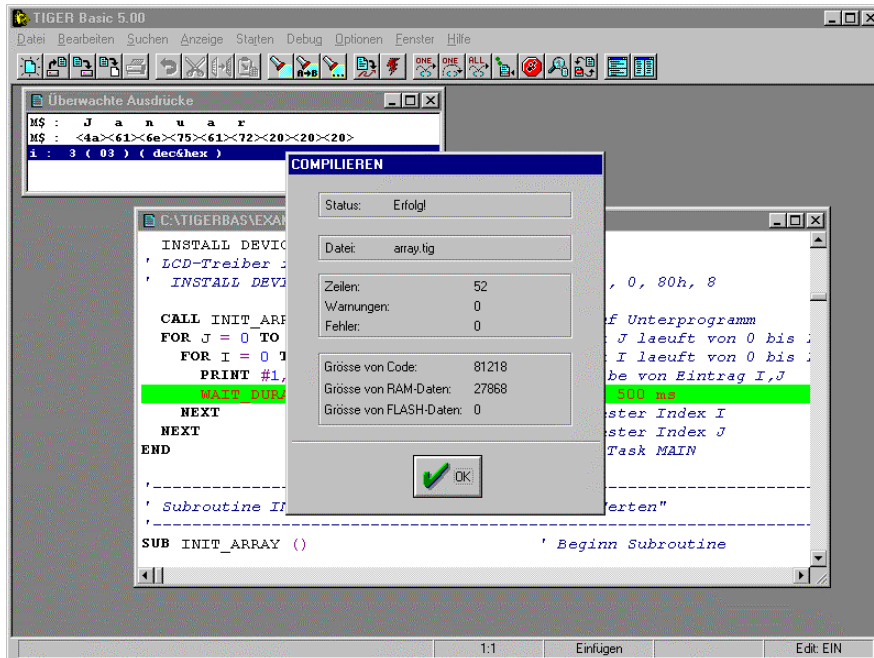
Wird ein BASIC-Tiger[®]-Modul an der Schnittstelle erkannt, informiert sich der Compiler über Größe von RAM und Flash des Moduls. Wird kein Modul gefunden (Time-Out: siehe Optionen), dann werden die Angaben zugrundegelegt, die unter „**Optionen / Compiler**“ eingetragen sind. Dies muß bestätigt werden. Das Warten auf das Time-Out kann mit der ESC-Taste abgebrochen werden.

Zunächst bearbeitet der Präprozessor Ihren Text und fügt Dateien dort ein, wo eine ‘#INCLUDE’-Anweisung gefunden wird, und führt die Ersetzungen im Text durch (siehe ‘#DEFINE’-Anweisung). Fehlermeldungen können bereits vom Präprozessor kommen.

Dann wird der Text übersetzt und dabei auf syntaktische Fehler, Übereinstimmung zwischen Deklaration und Anwendung von Variablen, passende Parameterübergabe bei Unterprogrammen untersucht. Dies kann wiederum zu Fehlermeldungen führen.

Entwicklungsumgebung

Ein lauffähiges BASIC-Tiger®-Programm wird nur erzeugt, wenn der Compiler keine Fehler meldet. Warnungen sind erlaubt.



Unbedingt compilieren

Wie „Compilieren“, allerdings wird die Compilation in jedem Fall ohne Nachfrage durchgeführt.

Ausführen

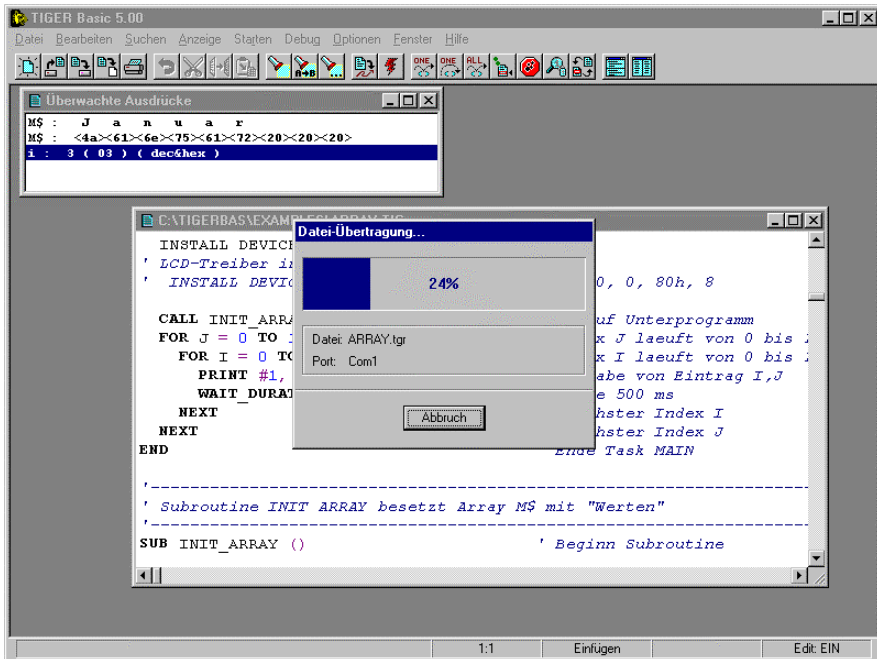
Startet das im aktiven Fenster befindliche Programm. Es findet eine Überprüfung statt, ob das im Modul geladene Programm der neuesten compilierten Version entspricht. Ist dies nicht der Fall, findet zuerst ein Download statt. Ist das Programm in der aktuellen Fassung auch noch nicht compiliert, wird dies ebenfalls nachgeholt. Sollte der Compiler Fehler melden, wird der Vorgang abgebrochen, es findet kein Download statt.

Im Debug-Mode hält das Programm an, wenn es auf einen Haltepunkt läuft.

Beenden Sie nach dem Download den PC-Mode mit Hilfe des Schiebeschalters, dann läuft Ihr Programm nach dem nächsten Reset oder Neustart im Run-Mode.

Programm laden

Das im aktiven Fenster befindliche Programm wird in das Tiger-Modul geladen.. Ist das Programm noch nicht compiliert, wird dies jetzt nachgeholt. Sollte der Compiler Fehler melden, wird der Vorgang abgebrochen, es findet kein Download statt.



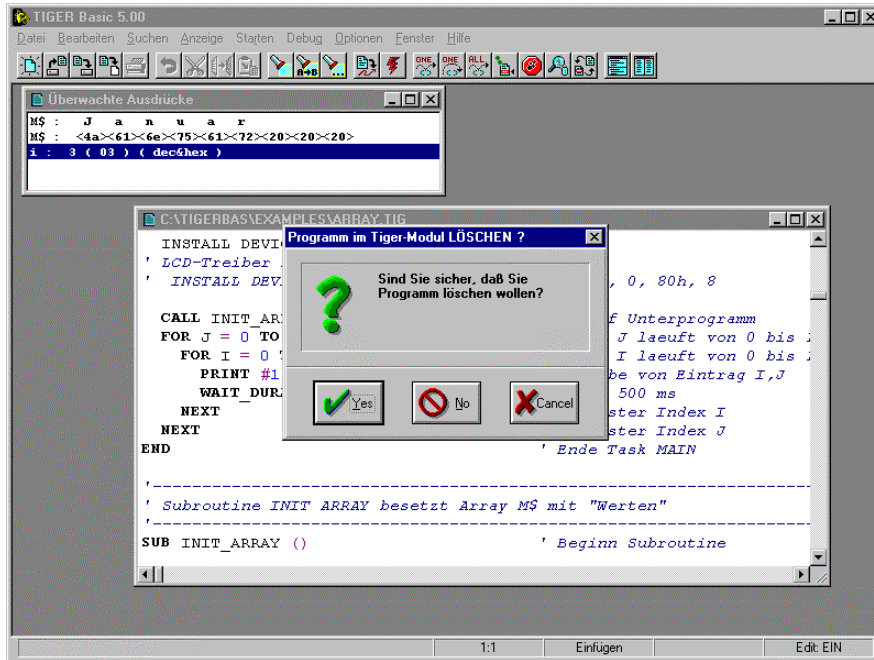
3

Entwicklungsumgebung

Programm löschen

Das im Tiger-Modul befindliche Programm wird gelöscht.

3



Debug-Menü

Ein Programm, welches den Präprozessor und den Compiler ohne Fehlermeldung passiert hat, kann weitere Fehler enthalten, die erst zur Laufzeit auftreten, also im BASIC-Tiger[®]-Modul während der Programmausführung.

Normalerweise hat das Modul keine Möglichkeiten, Fehler zu melden. Es verhält sich deswegen so sinnvoll wie möglich, indem der Fehler entweder ignoriert wird oder die fehlerhafte Task beendet wird. Tritt ein fataler Fehler in der Task MAIN auf, wird sie neu gestartet. Näheres zur Fehlerbehandlung während der Laufzeit erfahren Sie unter 'Fehlerbehandlung während der Laufzeit' im Programmierhandbuch.

Während der Entwicklungsphase sollen Fehler beseitigt werden, um zur Laufzeit ein möglichst kontrolliertes Verhalten des Moduls zu erreichen.

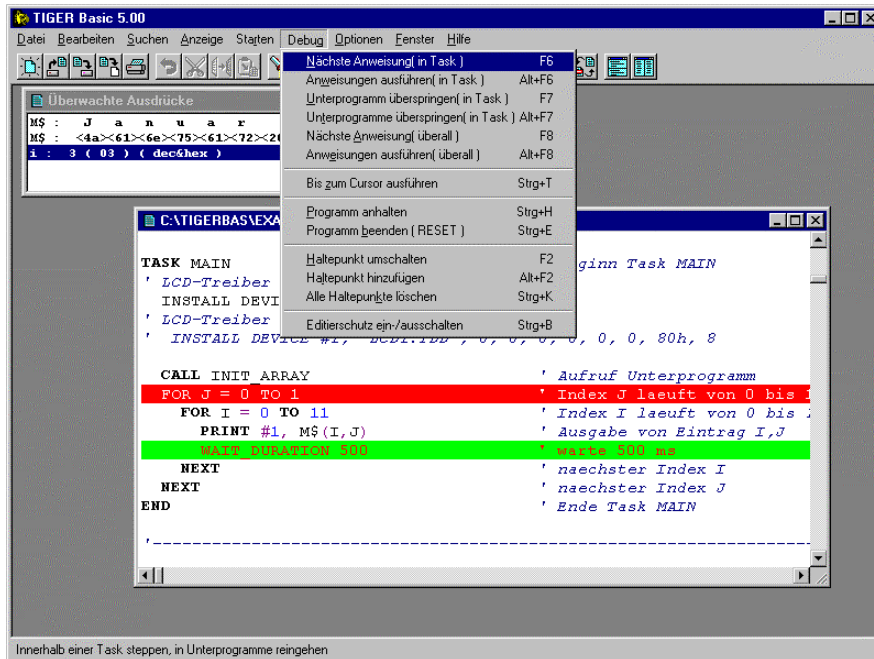
Wenn der Schiebeschalter auf dem Plug & Play Lab in der Stellung PC-Mode steht, befinden sich das Modul im Debug-Mode und es ist möglich, das Programm Schrittwise auszuführen, Haltepunkte zu setzen oder ein laufendes Programm anzuhalten. Im Debug-Mode ist ein laufendes Programm etwas langsamer als im Run-Mode. Bei kurzen Programmen macht sich der Verwaltungsanteil des PC-Modes stärker bemerkbar als bei langen Programmen.

Im Debug-Modus ist das BASIC-Tiger[®]-Modul noch mit dem PC verbunden, und Laufzeitfehler können zum PC zurückgemeldet und gezielt Haltepunkte gesetzt werden, um Fehler aufzuspüren.

Entwicklungsumgebung

Beim Ausführen von Einzelschritten oder an Haltepunkten ist die nächste auszuführende Zeile immer mit Grün unterlegt. Zeilen mit gesetzten, aber noch nicht erreichten Haltepunkten sind mit Rot unterlegt. Hat das Programm auf Grund eines Laufzeitfehlers aus dem Laufen heraus angehalten, dann steht der grüne Balken **hinter** der Anweisung, die den Fehler verursacht hat. Bei Programmen mit mehreren Tasks ist dies in der Regel in der **nächsten Task**, so daß die Fehlermeldung meist nicht zur markierten Stelle im Quell-Code paßt. In diesem Fall steppen Sie mit F8 (Nächste Anweisung (überall)) solange weiter, bis Sie wieder in der fehler-verursachenden Task hinter der fehlerhaften Anweisung angelangt sind.

3



Nächste Anweisung (in Task)

Mit diesem Befehl führen Sie einen einzelnen Programmschritt in der aktuellen Task aus.

Alle anderen Tasks laufen entsprechend ihrer Priorität weiter. Nähere Informationen zur Task-Priorisierung finden Sie unter 'SET_TASK_PRIO' im Programmierhandbuch.

Mit diesem Befehl durchlaufen Sie auch Unterprogramme.

Anweisungen ausführen (in Task)

Hiermit können Sie mehrere Schritte auf einmal ausführen. Die Anzahl der auszuführenden Programmschritte wird in einem Dialogfenster abgefragt.

Unterprogramm überspringen(in Task)

Mit diesem Befehl führen Sie einen einzelnen Programmschritt in der aktuellen Task aus, jedoch werden Unterprogramme mit einem Schritt ausgeführt.

Unterprogramme überspringen(in Task)

Hiermit können Sie mehrere Schritte auf einmal ausführen. Die Anzahl der auszuführenden Schritte wird in einem Dialogfenster abgefragt. Unterprogramme werden dabei als ein Schritt angesehen.

Nächste Anweisung (überall)

Mit diesem Befehl führen Sie einen einzelnen Programmschritt in der Task aus, die gerade entsprechend der Prioritätsverteilung an der Reihe ist. Damit läßt sich das Task-Switching sichtbar nachvollziehen. Anmerkung: im RUN-Modus ergeben sich durch andere zeitliche Gegebenheiten nicht die gleichen Task-Wechsel.

Anweisungen ausführen (überall)

Hiermit können Sie mehrere Schritte ausführen. Die Anzahl der auszuführenden Schritte wird in einem Dialogfenster abgefragt.

Bis zum Cursor ausführen

Das Programm läuft ab seiner aktuellen Position bis zu der Programmzeile, in der sich im Editorfenster der Cursor befindet. Dort wird der Programmablauf unterbrochen.

Programm anhalten

Der Programmlauf wird unmittelbar gestoppt, die als nächstes auszuführende Programmzeile wird grün hinterlegt. Nun können mit den Debug-Befehlen einzelne Anweisungen ausgeführt oder das Programm mit „Ausführen“ aus dem Starten-Menü fortgesetzt werden.

Programm beenden (RESET)

Das laufende Programm wird beendet und in seinen Ausgangszustand gebracht. Die Funktionsweise entspricht ungefähr einem Hardware-Reset. Das Programm kann danach komplett neu gestartet werden.

Haltepunkt umschalten

Mit diesem Befehl setzen Sie einen Haltepunkt in der Zeile, in der sich der Cursor befindet. Steht Ihr Cursor bereits auf einem bestehenden Haltepunkt, wird dieser entfernt.

Haltepunkt hinzufügen

Dieser Befehl setzt einen Haltepunkt in der Zeile, in der sich der Cursor befindet. Zusätzlich werden Sie jedoch nach der **Anzahl der Durchläufe** gefragt, die das

Entwicklungsumgebung

Programm den Haltepunkt passieren soll, bevor das Programm wirklich angehalten wird.

Alle Haltepunkte löschen

Alle aktiven Haltepunkte werden gelöscht.

3

Wenn sich das Programm gerade auf einem Haltepunkt befindet, wird der Haltepunkt durch den grünen Balken überdeckt.

Wenn sich eine Fehlermeldung ergibt, dann ist es oft wünschenswert, den Fehler zunächst zu ignorieren, um weitere Fehler zu lokalisieren, bevor man sich wieder an eine Editierphase begibt. Befindet sich jedoch der Fehler in einer Schleife, so stößt man bei jedem Schleifendurchlauf wieder auf denselben Fehler und der Programmauslauf wird unterbrochen. Im Dialog **Debugger-Einstellungen** aus dem Menü **Optionen** können Sie das Anhalten des Programms bei Run-Time-Fehler unterbinden. Sie sollten jedoch nicht vergessen, es danach wieder einzuschalten.

Hat das Programm auf Grund eines Laufzeitfehlers aus dem Laufen heraus angehalten, dann steht der grüne Balken **hinter** der Anweisung, die den Fehler verursacht hat. Bei Programmen mit mehreren Tasks ist dies in der Regel in der **nächsten Task**, so daß die Fehlermeldung meist nicht zur markierten Stelle im Quell-Code paßt. In diesem Fall steppen Sie mit **F8** (Nächste Anweisung (überall)) solange weiter, bis Sie wieder in der fehler-verursachenden Task hinter der fehlerhaften Anweisung angelangt sind.

Optionen-Menü

Die Arbeitsumgebung

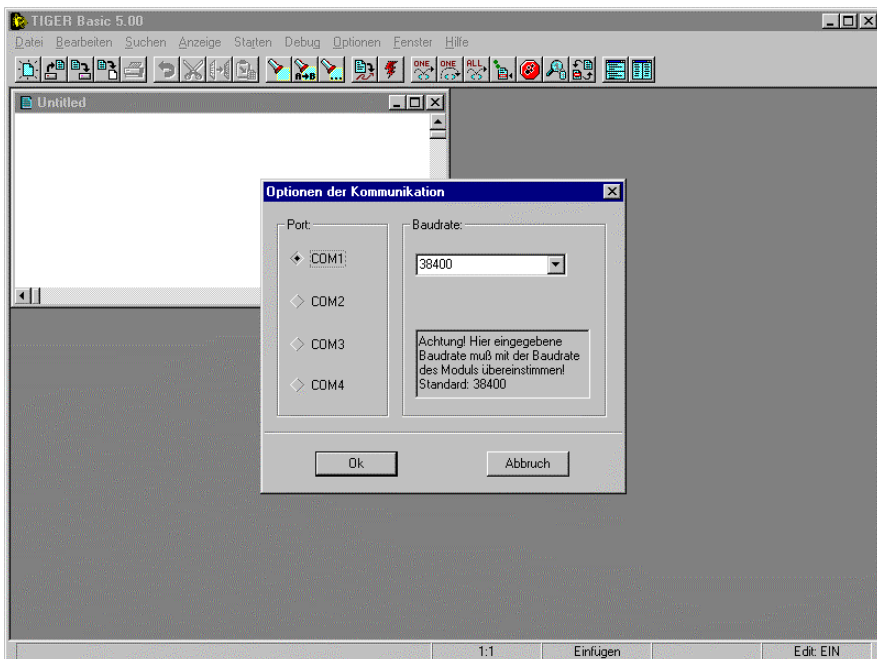
- benutzt einen COM-Port Ihres Rechners zum Programmieren der Module sowie für den Debug-Modus
- geht von einer bestimmten Flash- und RAM-Größe im Tiger-BASIC®-Modul aus
- legt Arbeitsdateien, Auto-Backups und temporäre Dateien in Verzeichnissen an
- erwartet Tiger-BASIC®-Treiber in bestimmten Verzeichnissen

3

Diese und weitere Optionen sind in verschiedenen Untermenüs einstellbar.

Übertragung

Hier legen Sie den für die Datenübertragung zum Tiger-Modul verwendeten seriellen Port und die Übertragungsgeschwindigkeit fest (muß z.Zt. 38400 Baud betragen). Die Übertragungsparameter der Schnittstelle werden automatisch eingestellt.

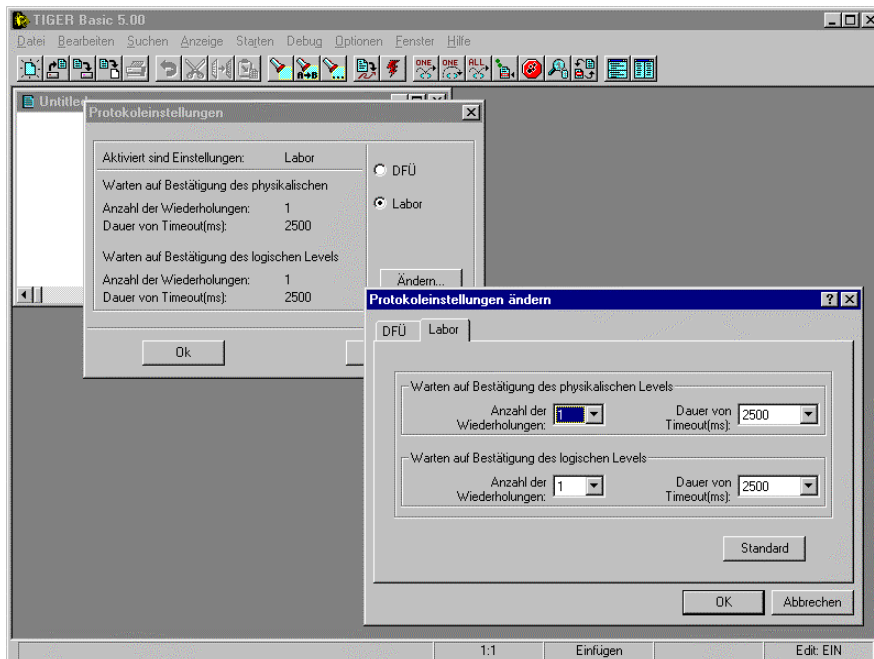


Protokoll

Wenn die Benutzeroberfläche auf dem PC ein Tiger-Modul kontaktiert, dann wird eine bestimmte, voreingestellte Zeit auf die Antwort gewartet. Antwortet das Modul nicht, dann wird der Vorgang wiederholt. Die Einstellungen der Anzahl der Wiederholungen und des Time-Outs wird in diesem Dialog vorgenommen. Es werden zwei Einstellungssätze unterschieden:

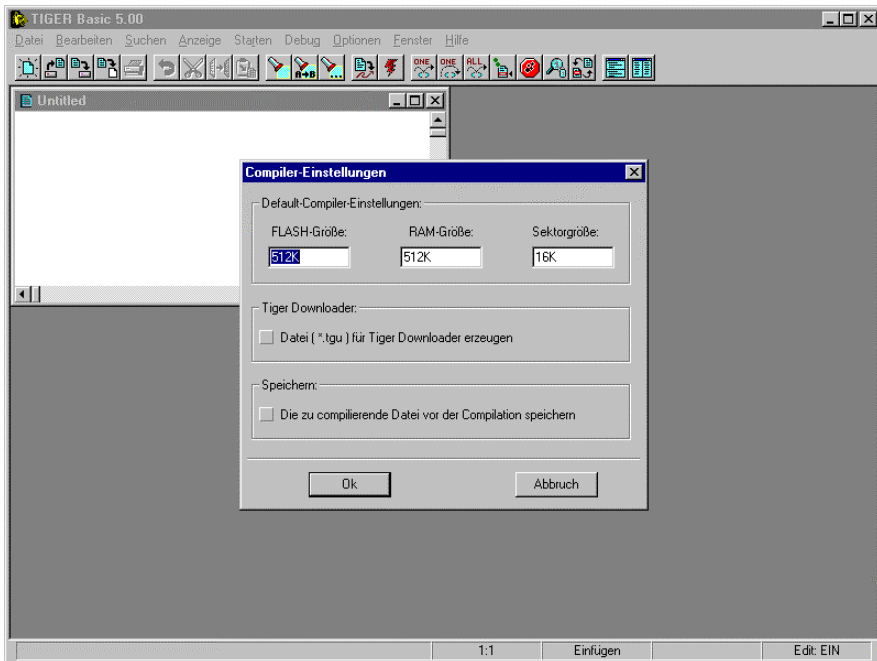
- Labor: PC und Modul sind in der Regel über ein kurzes Kabel direkt verbunden. Verzögerungen entstehen durch Pufferungen der Daten und die Windows-Oberfläche. Übertragungsstörungen sind nicht vorhanden.
- DFÜ: stellvertretend für alle Verbindungen zum Modul, die mit längeren Verzögerungen und eventuellen Störungen behaftet sind (Modem, Funk).

Es kann schnell zwischen den Einstellungen gewechselt werden. Über den Button „Ändern“ sind die Time-Outs und die Anzahl der Wiederholungen einstellbar.



Compiler

Mit dem Befehl teilen Sie dem Compiler mit, wieviel Flash- und RAM-Speicher Ihr BASIC-Tiger®-Modul hat. Diese Einträge werden jedoch nur verwendet, wenn Sie ohne angeschlossenes BASIC-Tiger®-Modul compilieren und der Compiler die tatsächlichen Parameter des Moduls nicht abfragen kann. Außerdem können Sie hier festlegen, ob eine für den Tiger-Downloader nutzbare Datei erzeugt werden soll.



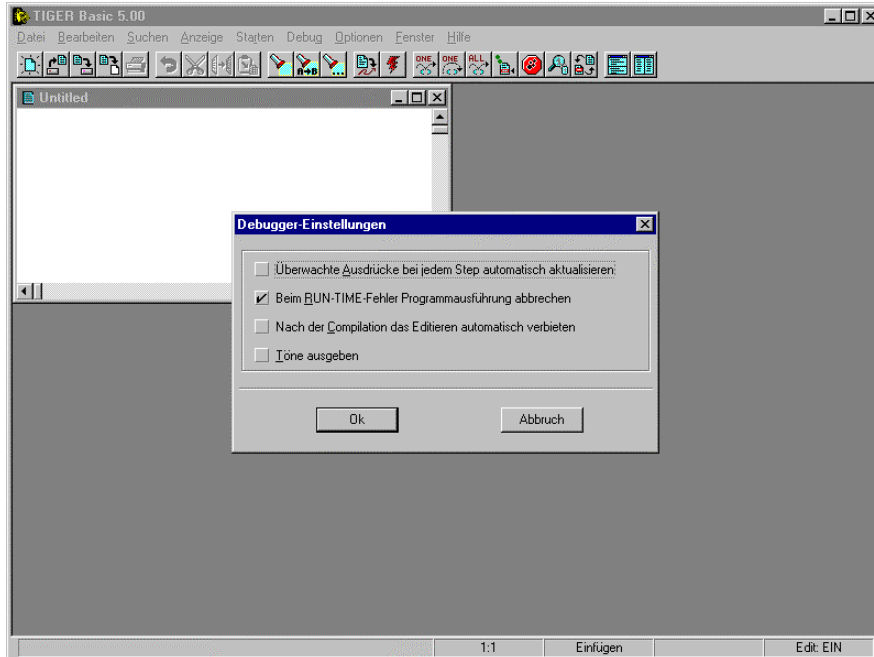
3

Entwicklungsumgebung

Debugger

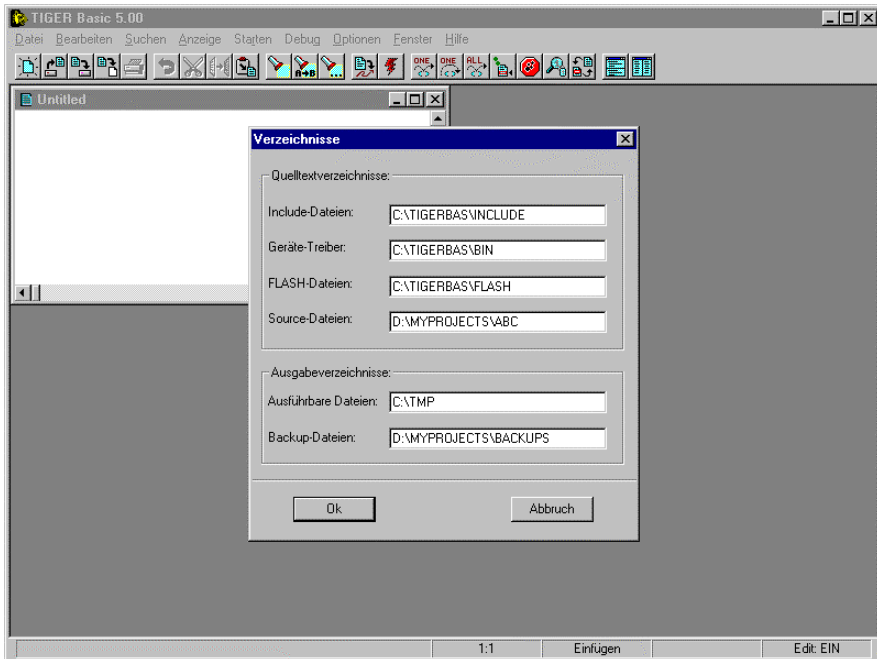
Hier werden für den Debugger relevante Einstellungen vorgenommen. Sollen überwachte Ausdrücke nach jedem im Debug-Modus vorgenommenen Schritt automatisch aktualisiert werden, ohne CTRL-W drücken zu müssen, muß hier der entsprechende Schalter aktiviert werden. Außerdem wird hier eingestellt, ob die Programmausführung im Debug-Modus bei einem auftretenden RUN-TIME-Fehler abgebrochen oder fortgesetzt werden soll. Töne können bei Fehlern ausgegeben werden.

3



Verzeichnisse

Unter diesem Menüpunkt geben Sie ein, wo sich Ihre Include-Dateien, die Geräte-Treiber, die FLASH-Dateien, Ihre Quelldateien, die ausführbaren Tiger-BASIC®-Dateien sowie die Auto-Backup-Dateien befinden.



3

Entwicklungsumgebung

Editor

Schriftart und Schriftgröße

Hier wird sowohl die Schriftart als auch die Schriftgröße eingestellt, die der Editor zur Anzeige des Quellcodes verwendet.

Leerzeichen beibehalten

Wenn nicht angekreuzt, entfernt der Editor anhängende Leerzeichen, wenn die Zeile verlassen wird. (TAB-Zeichen werden nicht entfernt). Das Kommando „Verschachtelung hervorheben“ wird dadurch nicht beeinflusst, daß heißt dieses Kommando ersetzt zur Verschachtelung Leerzeichen durch TAB-Zeichen).

Editormodus

Cursor beachtet Zeilengrenzen und Tabs: Sie können einen Editormodus einstellen, in dem Sie frei auf dem Bildschirm herumlaufen können, unabhängig vom eingegebenen Text, oder alternativ einen Modus, in dem die Zeilenenden bei Cursorbewegungen beachtet werden. Probieren Sie beide Modi aus und entscheiden Sie selbst, welcher Modus für Sie der bessere ist.

Automatisches Einrücken

Kreuzen Sie diese Option an, wenn der Editor automatisch die Einrückung kontrollieren soll.

Editor-Tabulatoren

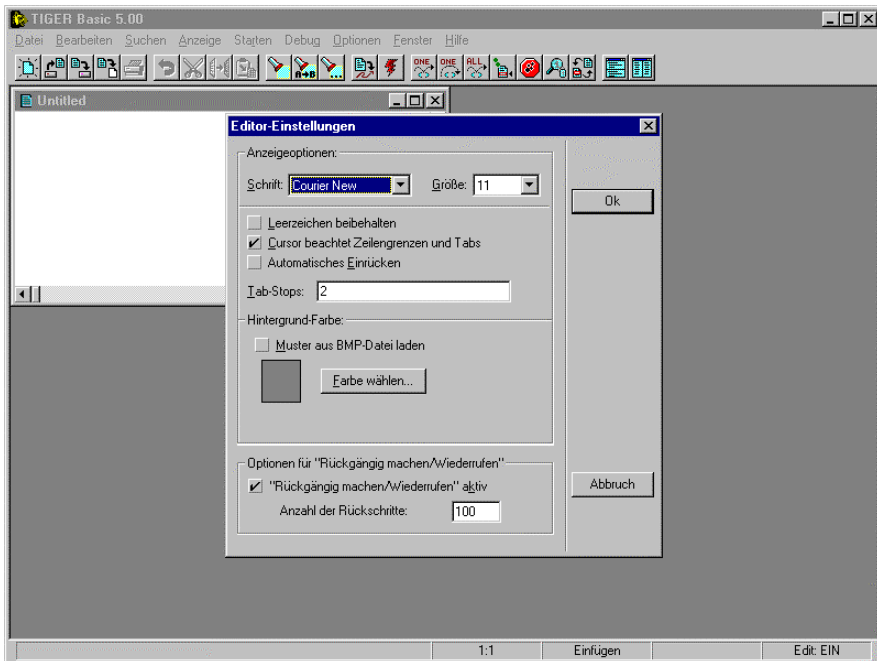
Stellen Sie ein, wieviele Zeichen ein Tabulatorschritt umfaßt.

Editor-Tabulatoren

Stellen Sie ein, wieviele Zeichen ein Tabulatorschritt umfaßt.

Editor-Tabulatoren

Stellen Sie ein, wieviele Zeichen ein Tabulatorschritt umfaßt.



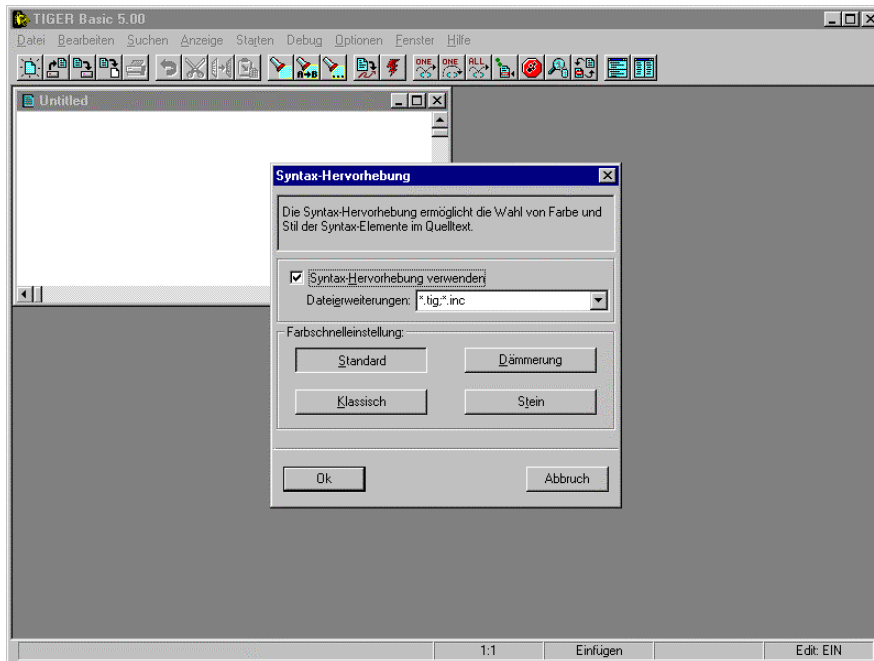
Speichern

Sicherheitskopie

Sie können festlegen, ob beim Abspeichern einer neuen Fassung die vorherige Fassung des Programms als Sicherungskopie erhalten bleibt und in welchen Zeitabständen eine automatische Sicherung durchgeführt werden soll.

Syntax-Hervorhebung

Die Syntax-Hervorhebung wird in diesem Dialog aus- oder eingeschaltet. Die Hervorhebung wirkt die auf Datentypen mit den angegebenen Endungen. Es stehen 4 verschiedene Muster zur Verfügung. Mit Hilfe der Syntaxhervorhebung können reservierte Worte erkannt oder Schreibweisen ausprobiert werden. Schreiben Sie zum Test einmal „inputline“ und fügen den Unterstrich nachträglich ein: „input_line“, und beobachten Sie die Syntaxhervorhebung.



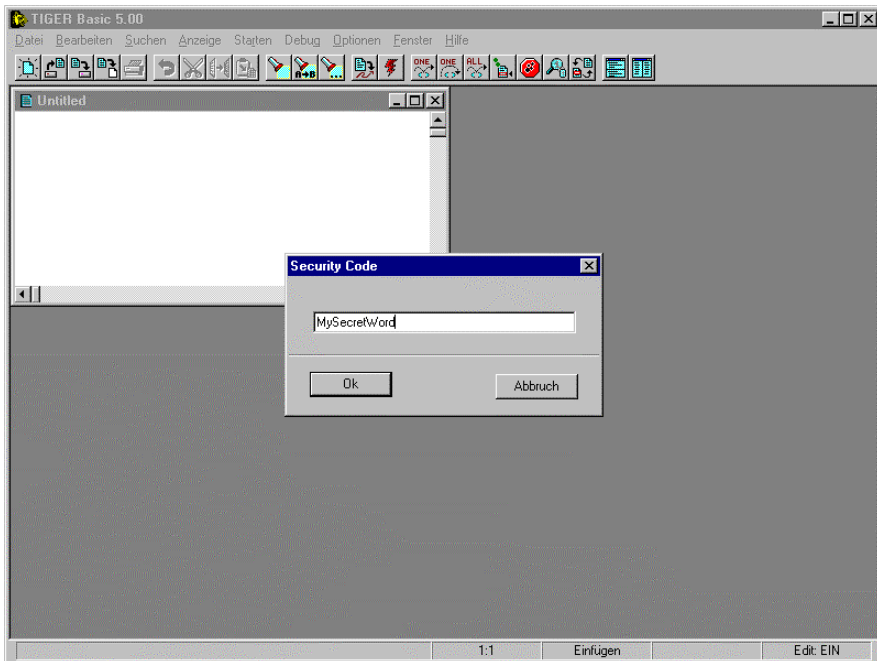
Security-Code

Um Ihre Software im BASIC Tiger® - oder TINY Tiger® -Modul zu schützen, fügen Sie folgende Zeile in Ihren Quelltext ein:

USER_SECURITY "Codewort"

"Codewort" besteht aus 3 bis 31 Zeichen. Das Codewort ist eine Kombination von Buchstaben und Ziffern und beginnt mit einem Buchstaben. Groß- und Kleinschreibung wird unterschieden.

Wenn ein Programm, welches eine USER_SECURITY-Zeile enthält, in das Modul geladen wurde, ist Debugging oder Programm löschen nur möglich, wenn das gleiche Codewort im Dialog **Security Code** des Menüs **Option** eingetragen wurde. Stellen Sie sicher, daß Sie das Codewort nicht vergessen, da es keine Möglichkeit gibt, das Programm im Modul ohne Codewort zu ändern.



3

Fenster-Menü

Wie unter Windows üblich, lassen sich verschiedene Fenster gleichzeitig öffnen. Sie können sich gleichzeitig den Quelltext und Include-Dateien anzeigen lassen. Nach dem Compilieren stehen Meldungen in einem separaten Fenster.

Unter dem Fenster-Menü können Sie die Anordnung der Fenster einstellen (überlappend/nebeneinander), Fenster-Symbole anordnen, alle Fenster gleichzeitig schließen oder ein Fenster gezielt aktivieren.

Hilfe-Menü

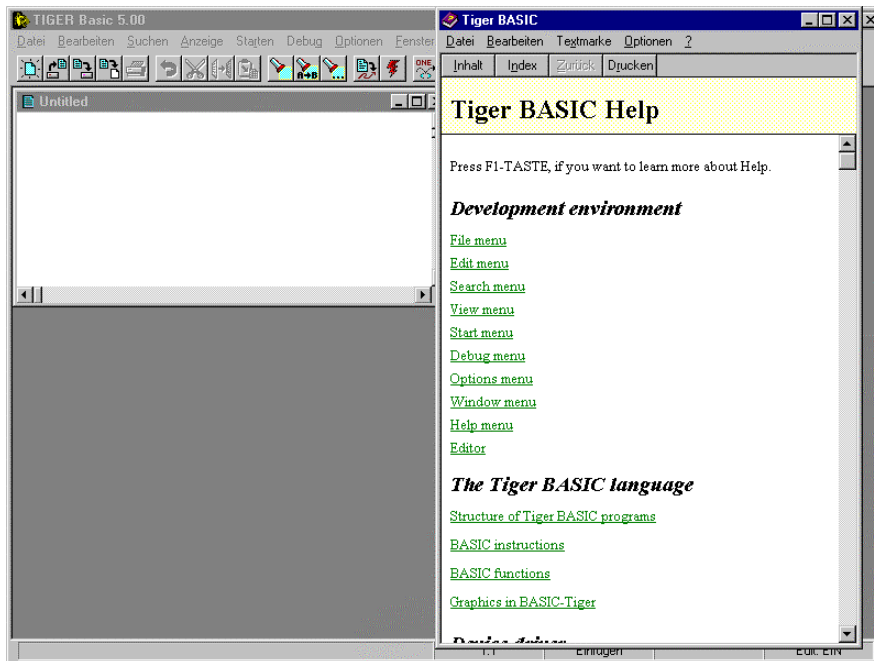
Über den Menüpunkt Hilfe können Sie die Hilfe zu Tiger-BASIC[®] aufrufen. In vielen Fällen ist das kontextsensitive Hilfesystem schneller zur Hand, indem Sie die Instruktion oder Funktion, zu der Sie Hilfe benötigen, in den Quelltext schreiben und F1 drücken, während sich der Cursor auf oder hinter dem Wort befindet.

Die meisten Hilfeseiten beinhalten die Möglichkeit, ein Beispielprogramm aufzurufen. Aus diesem Beispielprogramm lassen sich Teile markieren und mit den Editiertastenkombination STRG-C herauskopieren.

3

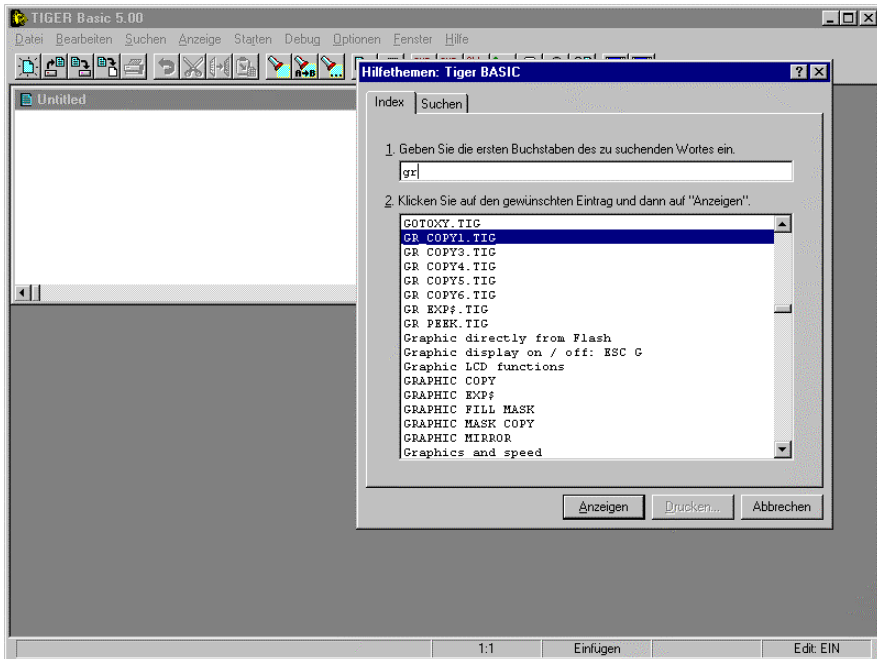
Inhalt

Dieser Menüpunkt bringt Sie zu einer inhaltlich gegliederten Hilfe,



Suche über Schlüsselwort

Hier können Sie gezielt nach einem bestimmten Begriff, wie dem Namen eines Befehls oder einer Funktion, suchen.



3

Editor

Zum Editieren Ihres Programmtextes stehen die üblichen Hilfsmittel des Menüs Bearbeiten zur Verfügung.

Grundsätzlich gilt: bevor Sie Änderungen an Ihrem Text vornehmen können, müssen Sie ihn markieren. Es kann immer nur eine Stelle markiert sein. Die Markierung entspricht einem vergrößerten Cursor.

3

Text markieren

Markieren mit der Maus: beginnen Sie an der Stelle, an der Sie die Markierung beginnen wollen, und fahren Sie mit gedrückter linker Maustaste über den zu markierenden Text.

Markieren mit der Tastatur: gehen Sie an die Stelle, an der Sie die Markierung beginnen wollen, und bewegen Sie mit gedrückter Shifttaste den Cursor über den zu markierenden Text. Sie können jede Taste benutzen, die auch sonst den Cursor bewegt, also die Pfeile, die 'Bild'-Tasten zum Blättern, etc.

Sobald sie den Cursor ohne gedrückte Shifttaste bewegen, verschwindet die Markierung aus dem Text. Der Cursor nimmt wieder seine normale Größe an.

Text löschen

Einzelne Zeichen löschen Sie mit der Rücktaste, der Entfernen-Taste oder durch Überschreiben mit anderen Zeichen.

Markieren Sie den Text und drücken Sie die Rücktaste, die Entfernen-Taste oder überschreiben Sie den markierten Text durch Eingabe eines Zeichens (außer RETURN).

Text einfügen

Stellen Sie sicher, daß der Modus Einfügen eingeschaltet ist (siehe Statuszeile ganz unten).

Schreiben Sie einfach an der Stelle, an der Sie Text einfügen wollen.

Compiler-Fehlermeldungen

	System	
1	Datei kann nicht geöffnet werden	Zu compilierende Datei oder eine der eingebundenen Dateien kann nicht geöffnet werden.
2	Geräte-Treiberdatei kann nicht geöffnet werden	Geräte-Treiberdatei kann von Compiler nicht gefunden oder geöffnet werden. Prüfen Sie den in Ihrem Programm benutzten Namen der Treiberdatei oder den in der Entwicklungsumgebung angegebenen Pfad für Treiberdateien (Siehe: "Optionen-Verzeichnisse").
3	Tiger-Systemdatei kann nicht geöffnet werden	Prüfen Sie, ob das BASIC-Tiger Entwicklungssystem richtig installiert worden ist.
4	Version ist ungültig bei der Tiger-Systemdatei	
5	Tiger-Systemdatei ist verfälscht	
Präprozessor-Fehler:		
6	Rekursive Einbindung einer Datei unzulässig	Eine der mit "#include" eingebundenen Dateien versucht, sich selbst direkt oder indirekt einzubinden.
7	Unzulaessiger Parameter ist bei der Makroersetzung angegeben	hinter '#' wird ein unbekanntes Wort verwendet oder in '#define' fängt das zu ersetzende Wort unmittelbar nach "#define" mit einem unzulässigen Zeichen an; zulässig sind an erster Position "_", eine Ziffer oder ein Buchstabe.
8	Maximale Verschachtelungstiefe für Include-Dateien ist überschritten	Einzubindende Datei war zu tief verschachtelt. Die Tiefe der Verschachtelung ist auf 30 begrenzt.
9	Rekursion in "\"#define\""-Angaben. Siehe Symbol:	Direkte und indirekte Rekursion der Definitionen in '#define' ist unzulässig.

	Compiler-Fehler:	
10	Parameter fehlt wahrscheinlich in dieser Instruktion	Es werden die Sprachkonstruktionen nicht richtig benutzt oder die Schlüsselwörter falsch geschrieben.
11	Syntax-Fehler	
12	Undefinierte Variable	Tiger-BASIC [®] verlangt per Default keine Deklaration der Variablen vor deren Benutzung. Mit Hilfe der Pseudoinstruktion 'USER_VAR_STRICT' kann jedoch die Kontrolle aktiviert werden, die besonders für große Programme zu empfehlen ist. Bei aktivierter Kontrolle erscheint diese Fehlermeldung, wenn eine Variable nicht deklariert wurde.
13	Ungültiges Schlüsselwort	Ein zusammengesetztes Schlüsselwort enthält Schreibfehler im zweiten oder im dritten Teil, wenn das Schlüsselwort in mehreren Wörtern geschrieben wird. Zur besseren Lesbarkeit wird die Verwendung der Unterstriche (nicht Leerzeichen) als Trenner in zusammengesetzten Schlüsselwörtern empfohlen, z.B.: 'SET_TASK_PRIO ...' statt 'SET TASK PRIO ...'.
14	Name doppelt definiert	Der Name in der Definition einer Variablen wurde schon einmal im Programm für die Definition einer anderen Variablen benutzt. Siehe 'Gültigkeitsbereich von Variablen' im Programmierhandbuch.
15	Task/Unterprogramm doppelt definiert	Der Name wurde vorher im Programm für eine andere Task oder ein anderes Unterprogramm benutzt.
16	Instruktion außerhalb Task/Unterprogramm	Eine ausführbare Instruktion (nicht Pseudo-Instruktion) muß innerhalb einer Task oder eines Unterprogramms erscheinen, d.h. nur zwischen den Instruktionen 'TASK ...' und 'END' oder zwischen 'SUB ...' und 'END'.

	Compiler-Fehler:	
17	[17] Unzulässige Verschachtelung. Möglich: Kein END in Task/Unterprogramm	Tasks und Unterprogramme können nicht ineinander verschachtelt werden. Der Fehler tritt auch dann auf, wenn eine Task oder ein Unterprogramm nicht mit der Instruktion 'TASK ...'/'SUB ...' anzufängt oder die Instruktion 'END' vergessen wurde.
18	KByte-Konstante kann nur aus dezimalen Ziffern bestehen	Syntax-Fehler, selbsterklärend.
19	Oktale Zahl ([0-7]) kann keine Buchstaben enthalten	
20	Oktale Zahl kann nur durch die Ziffern 0..7 dargestellt werden	
21	Binäre Zahl ([0-1]) kann keine Buchstaben enthalten	
22	Binäre Zahl kann nur durch die Ziffern 0 und 1 dargestellt werden	
23	Unzulässige Zeichen in numerischer Konstante	
24	Darstellung von RAL-Zahlen nur mit dezimalen Ziffern	
25	Unzulässige Zeichen in der REAL-Konstante	
26	Nur dezimale oder hexadezimale Ziffern sind in spitzen Klammern erlaubt	
27	%-Konstante darf nur aus hexadezimalen Zahlen bestehen	

	Compiler-Fehler:	
28	Variable darf nicht an dieser Stelle erscheinen	In Tiger-BASIC® gibt es Instruktionen, die als Parameter nur Konstante oder nur Variable von einem bestimmten Typ zulassen.
29	Einfache numerische Konstante als Parameter erwartet	
30	Numerische Variable wird hier erwartet	
31	String-Konstante als Parameter erwartet	
32	String-Variable wird hier erwartet	
33	REAL-Konstante an dieser Stelle unzulässig	
34	REAL an dieser Stelle unzulässig	
35	String-Name muß mit '\$'-Zeichen enden	Name einer String-Variablen muss mit dem '\$'-Zeichen enden. String-Operationen oder String-Funktionen können sonst nicht ausgeführt werden.
36	String ist zu lang. Maximale Länge von String:	In der Definition einer STRING-Variablen ist eine zu große Länge angegeben. Maximale Länge ist 65533 Zeichen.
37	Operator für Berechnungen mit REAL unzulässig	Für Floating Point Konstanten und Variablen sind folgende arithmetischen Operatoren zugelassen: '+', '-', '*', '/'
38	Unzulässige Verwendung der arithmetischen Operatoren mit REAL	
39	Linke Klammer fehlt	Öffnende oder schliessende Klammer fehlt in der Deklaration von ARRAY, STRING, FIFO oder beim Zugriff auf die Elemente von ARRAYs.
40	Rechte Klammer fehlt	

	Compiler-Fehler:	
41	Taskname nicht gefunden	In einer Instruktion, die mit der Taskverwaltung zu tun hat, ist der Name einer in diesem Programm nicht existierenden Task angegeben.
42	Kein END in Task/Unterprogramm	In einer Task oder einem Unterprogramm fehlt die END-Instruktion.
43	Maximale Anzahl der Tasks im Programm ist	Maximale Anzahl der Tasks im Programm wurde überschritten. In einem BASIC-Tiger [®] -Programm können maximal 32 Tasks definiert werden.
44	Taskname soll einen Buchstaben als erstes Zeichen enthalten	Name einer Task muß mit einem Buchstaben beginnen.
45	Haupttask mit dem Namen MAIN fehlt	Jeses BASIC-Tiger [®] -Programm muß eine Task mit dem Namen 'MAIN' enthalten.
46	"MAIN"-Task kann nicht verlassen oder gestoppt werden	Die MAIN-Task spielt in Tiger-BASIC [®] eine besondere Rolle. Die Instruktionen EXIT_TASK, STOP_TASK, RUN_TASK, etc. Können nicht auf die Task 'MAIN' angewendet werden.
47	"MAIN"-Task kann nicht gestartet werden	
48	Aufruf einer nicht definierten Funktion	Der angegebene Funktionsname ist entspricht nicht einer Tiger-BASIC [®] -Funktion.
49	Aufruf Unterprogramm /Funktion: falscher Datentyp beim Parameter	Der Datentyp im Aufruf eines Unterprogramms oder einer Funktion stimmt nicht überein mit dem Datentyp in der Definition dieses Unterprogramms bzw. den diese Funktion erfordert. Hier wird auch angegeben, bei welchem Parameter der Datentyp nicht richtig benutzt wird (erster Parameter besitzt die Nummer 1), und wie die Abweichung vom erwarteten Zustand aussieht: 'Typ im Aufruf' / 'Typ in der Definition'.

Entwicklungsumgebung

3

	Compiler-Fehler:	
50	Aufruf Unterprogramm /Funktion: falsche Parameter-Anzahl	Die Anzahl der Parameter im Aufruf eines Unterprogramms oder einer Funktion stimmt nicht mit der Parameter-Anzahl in der Definition dieses Unterprogramms bzw. dieser Funktion überein.
51	Kollision: Konstante im Aufruf wird in der Definition als VAR bezeichnet beim Parameter	Benutzung des Modifizierers 'VAR' in der Deklaration eines Unterprogramms erzwingt die Übergabe einer Variablen. Im Aufruf dieses Unterprogramms im entsprechenden Parameter sind weder Konstante noch Ausdrücke veränderbar.
52	Ausdrücke als Parameter im Aufruf eines Unterprogramms sind unzulässig	Verwendung von Ausdrücken, FIFO-Variablen oder Namen anderer Funktionen sind als Parameter eines Unterprogramms oder einer Funktion unzulässig.
53	FIFO kann nicht als Parameter an ein Unterprogramm übergeben werden	
54	Name der eingebauten Funktion ist als Parametername verwendet beim	
55	RETURN ist nur zum Verlassen des Unterprogramms vorgesehen	RETURN-Instruktion darf nur in einem Unterprogramm erscheinen, nicht in der Task
56	Ungültige Bedingung	Die Bedingung in der 'WHILE'- oder 'IF'-Instruktion enthält Syntax-Fehler.
57	Logische Verknuepfung kann nur innerhalb der Bedingung benutzt werden	Es wird eine logische Verknüpfung außerhalb einer Bedingung verwendet.
58	Vergleichsoperator kann nur innerhalb der Bedingung benutzt werden	Logische Verknüpfungen und Vergleichsoperatoren können nur innerhalb der Bedingung verwendet werden. Konstruktionen wie $a = (b < c)$ sind unzulässig.

	Compiler-Fehler:	
59	String in der Bedingung ohne Vergleichsoperator	Wenn innerhalb der Bedingung eine nicht STRING-Variable ohne Vergleichsoperator auftaucht, kann sie nach den Regeln 'gleich 0 ist FALSE, ungleich 0 ist TRUE' ausgewertet werden. Eine STRING-Variable kann jedoch nicht ausgewertet werden.
60	FOR ohne TO	Typische Syntax-Fehler bei der Verwendung der 'FOR ... NEXT' Schleife
61	FOR ohne NEXT	
62	IF ohne THEN oder falsche Position IF ohne ENDIF	Syntax-Fehler bei der Verwendung der 'IF ... THEN ... ENDIF' Konstruktion.
64	WHILE ohne ENDWHILE	'ENDWHILE'-Instruktion nach einer 'WHILE'-Instruktion fehlt.
65	SWITCH ohne ENDSWITCH	'ENDSWITCH'-Instruktion nach einer 'SWITCH/SWITCHI'-Instruktion fehlt.
66	LOOP ohne ENDLOOP	'ENDLOOP'-Instruktion nach einer 'LOOP'-Instruktion fehlt.
67	Maximale Verschachtelungstiefe für FOR überschritten	Die Verschachtelungstiefe folgender Konstruktionen ineinander ist begrenzt: 'FOR...NEXT', 'LOOP...ENDLOOP', 'IF...THEN...ELSE...ENDIF', 'SWITCH...CASE...ENDSWITCH', 'SWITCHI...CASE...ENDSWITCH'
68	Maximale Verschachtelungstiefe für IF überschritten	
69	Maximale Verschachtelungstiefe für SWITCH überschritten	
70	CASE-Konst. soll Typ von SWITCH-Var. besitzen (hier: numerisch)	In 'SWITCH...CASE...ENDSWITCH' Konstruktion sind nur Vergleiche von Werten mit gleichen Typenklassen erlaubt.
71	CASE-Konst. soll Typ von SWITCH-Var. besitzen (hier: String)	

	Compiler-Fehler:	
72	Maximale Länge für CASE-String-Konstante:	Maximale Länge für CASE-String-Konstante ist auf 250 Zeichen begrenzt.
73	Nur numerischer Ausdruck als Testausdruck in "SWITCHI" ist zulässig	‘SWITCHI’-Konstruktion sind für schnelle Bearbeitung der sequentiellen Fälle als Vereinfachung der universelleren ‘SWITCH’-Konstruktion gedacht. Als Testausdruck ist hier nur numerischen Typ (BYTE, WORD, LONG) und als Wahlkonstanten nur positive ganzen Zahlen vorgesehen.
74	Negative Konstanten in "SWITCHI" sind unzulässig	
75	SWITCH-Konstruktion ist zu gross	SWITCH-Konstruktion enthält zu viele Verzweigungen.
76	Unzulässiger GOTO-Sprung	Ein GOTO-Sprung ist nur unter folgenden Bedingungen zulässig: 1. GOTO-Sprung kann nur innerhalb derselben Task bzw. desselben Unterprogramms ausgeführt werden. 2. Einsprung in eine Schleife ist nicht möglich.
77	Einsprung in Schleife unzulässig	
78	Kein Label in Task/Unterprogramm für GOTO-Sprung:	In der Task oder dem Unterprogramm fehlt die Sprungmarke, die in einer GOTO-Instruktion als Ziel des Sprunges genannt wird.
79	Label doppelt definiert	Label kann nur an einer Stelle stehen.
80	Label kann nur allein in Zeile erscheinen	In einer Zeile, die ein Label enthält, stehen keine weiteren Instruktionen.
81	Array nicht definiert	Array muß vor der ersten Benutzung deklariert werden.
82	String-Länge oder Anzahl Elemente im Array negativ	Länge und Anzahl sind per Definitionem positive Werte.
83	Anzahl Dimensionen im Aufruf ist nicht gleich deren Anzahl in der Deklaration	Die Anzahl der Dimensionen beim Zugriff auf ein Element des Arrays muß genau mit der Anzahl der Dimensionen in der Deklaration dieses Arrays übereinstimmen.

	Compiler-Fehler:	
84	Variable nicht vom Typ ARRAY	Bei der Deklaration einer einfachen Variablen kann keine Dimension angegeben werden. Verwenden Sie ein ARRAY.
85	Maximale Größe für FIFO überschritten	FIFO-Puffer kann maximal 65535 Bytes enthalten. Um Größe des FIFO-Puffers zu berechnen, wird die Anzahl der Elemente mit Größe eines Elements multipliziert.
86	Maximale Gerätenummer überschritten:	Die Devicenummer muß kleiner als 64 sein.
87	Datei ist keine Geräte-Treiberdatei	In der Instruktion 'INSTALL_DEVICE' angegebener Dateiname entspricht nicht einer Tiger-BASIC®-Treiberdatei.
88	Geräte-Treiber-Installation kann nur in Task "MAIN" stattfinden	Alle Geräte-Treiber werden in der MAIN-Task installiert.
89	In der Instruktion fehlt Gerätenummer	In eine Instruktion, in der ein Gerät angesprochen wird, fehlt die Gerätenummer an erster Stelle unter den Parametern. Eine Gerätenummer fängt immer mit dem '#'-Zeichen an.
90	Minimale Stack-Größe ist:	In USER_STACK_SIZE wird eine nicht erlaubte Stackgröße angegeben. Für jede Task wird im RAM STACK-Bereich für Aufrufe von Unterprogrammen reserviert. Default-Größe dieses STACK-Bereiches ist 2048. Die Größe kann mit Hilfe der Pseudo-Instruktion 'USER_STACK_SIZE' im Rahmen von 512 bis 32767 geändert werden.
91	Maximale Stack-Größe ist:	
92	Der Security-Schlüssel ist zu kurz; minimale Länge des Schlüssels ist	Die Länge vom Security-Schlüssel für die Pseudo-Instruktion 'USER_SECURITY' soll zwischen 3 und 32 liegen.
93	Der Security-Schlüssel ist zu lang; maximale Länge des Schlüssels ist	

	Compiler-Fehler:	
94	Funktion mit solcher Nummer für diese Pseudoinstruktion existiert nicht	Für diese Pseudoinstruktion können nur ganz bestimmte Funktionsnummern benutzt werden.
95	Diese logische Port-Nummer ist unzulässig	Unzulässige logische Port-Adresse.
96	Programm beansprucht zu großen RAM-Bereich	Der RAM-Bereich, den das Programm für Daten braucht, ist grösser als der vorhandene RAM-Speicher des im Augenblick angeschlossenen Moduls. 1. RAM wird vor allem von Variablen belegt. ARRAYS und STRINGS belegen in der Regel den meisten Speicher. Beachten Sie in diesem Zusammenhang: Alle STRING-Variablen deren Größe bei der Deklaration nicht angegeben wurde, haben Default-Größe (64 Byte). Diese Default-Größe kann mit Hilfe der Pseudo-Instruktion 'USER_STRING_SIZE' beliebig geändert werden. 2. Für jede Task wird im RAM STACK-Bereich für Aufrufe von Unterprogrammen reserviert. Default-Größe dieses STACK-Bereiches ist 2048. Diese Default-Größe kann mit Hilfe der Pseudo-Instruktion 'USER_STACK_SIZE' im Rahmen von 512 bis 32767 beliebig geändert werden.
97	Programm beansprucht zu großen ROM-Bereich	ROM-Bereich, den das Programm für Code braucht, ist grösser als der FLASH-Speicher des im Augenblick angeschlossenen Moduls.
98	Programm ist zu groß. Maximale Anzahl Zeilen ist begrenzt auf	Für limitierte Versionen des Entwicklungssystems ist die Anzahl Zeilen im Programm ist begrenzt.

Downloader

Wenn Sie Ihren Kunden, die bei Ihnen ein Produkt mit BASIC-Tiger[®] gekauft haben, Firmware-Update schicken wollen, ohne daß Sie Ihren Source-Code weitergeben, dann müssen Sie Ihre Kunden in die Lage versetzen, ohne Tiger-BASIC[®]-Entwicklungsumgebung ein bereits compiliertes Programm in das Modul zu laden. Aus diesem Grund geben Sie einmal jedem Kunden den BASIC-Tiger[®]-Downloader entweder zusammen mit Ihrem Produkt oder zusammen mit dem Update.

Die Datei, die vom Downloader in ein BASIC-Tiger[®]-Modul geladen werden kann, wird in einem speziellen Format vom Compiler erzeugt (siehe **Compiler** im Menü **Optionen** in der Entwicklungs-Umgebung Tiger-BASIC[®]). Die Namenserweiterung dieser Datei lautet standardmäßig „TGU“.

BASIC-Tiger[®]-Downloader wird über Pull-Down-Menüs bedient.

Systemvoraussetzungen

Um den Downloader des BASIC-Tigers[®] betreiben zu können, benötigen Sie (Ihr Kunde) einen IBM-kompatiblen Rechner mit folgender Mindestausstattung:

- Pentium-II-Prozessor mit 233MHz
- Festplatte mit mind. 10 MByte freiem Speicher
- VGA-Grafikkarte
- 16 MB Arbeitsspeicher
- Maus
- Windows 95/98/2000/NT
- ein freier COM-Port
- CD-ROM-Laufwerk oder 3,5“ Diskettenlaufwerk

Der Downloader steht zum Download zur Verfügung im Internet unter www.wilke.de.

Hardware-Entwicklungsumgebung

Eine Beschreibung des BASIC-Tiger[®]-Moduls sowie des Tiny-Tiger[®]-Moduls und weiterer Varianten finden Sie im Kapitel ‚BASIC-Tiger[®]-Steuercomputer‘.

In diesem Kapitel werden drei Boards beschrieben, die als Hardware-Entwicklungsumgebung dienen:

3

Plug & Play Lab ist das umfangreichste der drei Boards. Es enthält eine Tastatur und ein LC-Display sowie diverse Schaltungen zur Analogverarbeitung bis hin zu einem Verstärker mit Lautsprecher. Verbindungen werden per Jumperkabel statt durch Löten hergestellt.

BASIC-Tiger[®] Prototyping Board

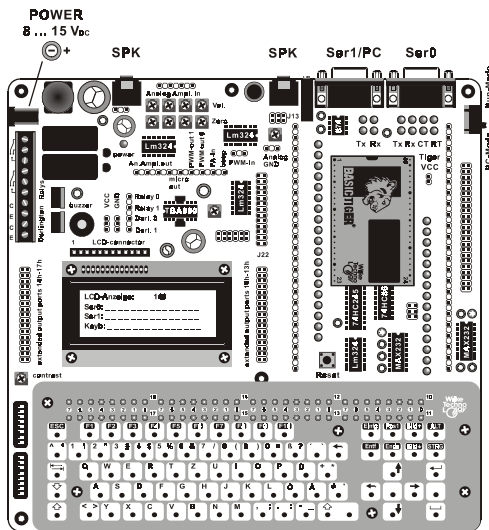
ist ein Euro-Board, welches alles enthält, was zum Arbeiten mit BASIC-Tiger[®]-Modulen mit oder ohne On-board-RS-232-Treibern notwendig ist. Ein Anschluß für eine Tastatur mit bis zu 64 Tasten sowie ein LC-Display-Anschluß sind vorhanden.

Tiny-Tiger[®] Prototyping Board

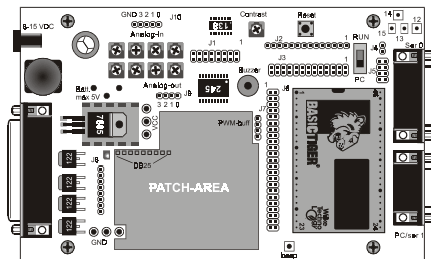
dient als Hardware-Entwicklungsumgebung für Tiny-Tiger[®]. Ein Anschluß für eine Tastatur mit bis zu 64 Tasten sowie ein LC-Display-Anschluß sind vorhanden. Das Board enthält 7 Treibertransistoren und eine 8-LED Statusanzeige.

Bitte stellen Sie sicher, daß Sie sich im richtigen Abschnitt befinden, wenn Sie über Ihr Board nachlesen. Teilweise unterscheiden sich die Boards nur in Details.

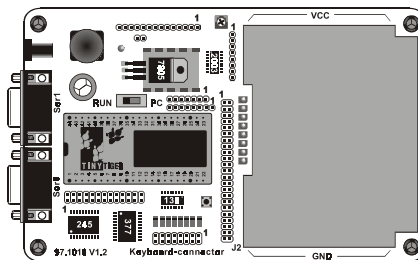
Anmerkung: Technische Änderungen ohne Vorankündigung vorbehalten.



Plug & Play Lab



BASIC-Tiger® Prototyping-Board



Tiny-Tiger®-Prototyping-Board

Entwicklungsumgebung

Leere Seite

3

Plug & Play Lab

Auf dem Plug & Play Lab laufen sofort fertige Anwendungen, ohne daß umfangreiche Versuchsaufbauten während der Entwicklungsphase aufgebaut werden müssen.



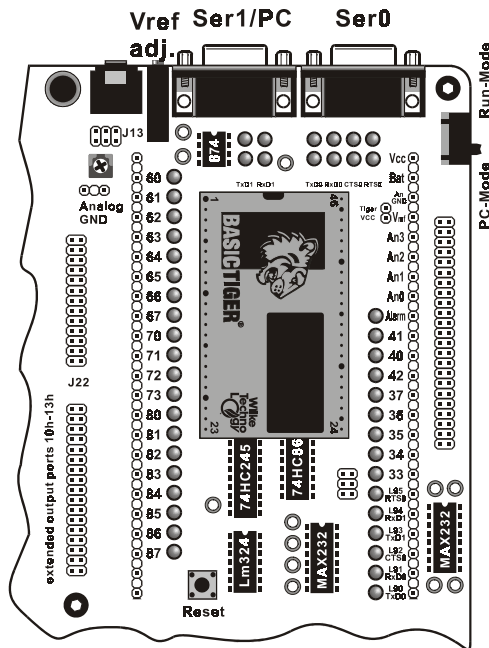
Das Plug & Play Lab versteht sich als Versuchsaufbau (den Sie nicht selbst aufbauen mußten) und darf nur von fachlich geschultem Personal in Betrieb genommen werden.

Auf dem Plug & Play Lab befindet sich:

- direkter PC-Anschluß
- Status-LEDs zur Anzeige des Portstatus
- 2 Serielle RS-232-Ports
- 64 Extended I/O-Pins
- Tastatur mit 76 Tasten
- Reset-Taster
- Summer zur Tonausgabe
- Mikrofonverstärker mit Mikrofon
- Endverstärker mit Lautsprecheranschluß
- LC-Display 4 x 20 Zeichen
- 2 x 8 Dipschalter
- 2 Relais
- 2 Powertransistoren
- Referenzspannung für AD-Umsetzer
- Erweiterungsanschluß
- PWM Ausgangsverstärker mit Filter

Entwicklungsumgebung

Eine kurze Übersicht soll Ihnen zunächst das Plug & Play Lab näher bringen.



Das Bild zeigt den Steckplatz für das BASIC-Tiger[®] Modul A, den Anschluß für den PC (serielle Schnittstelle 1) sowie den PC-Mode-Schalter als wichtigste Elemente für den Einstieg.

Auf den Steckpins rechts und links des Tigermoduls sind alle Anschlußpins des Moduls zugänglich.

Generell ist Vorsicht geboten, wenn per Stekkabel Verbindungen hergestellt werden: Pins, die Outputs sind, dürfen nicht mit VCC oder GND kurzgeschlossen werden oder gar zu hohe Spannung erhalten.

An einigen Pins ist bereits auf dem Plug & Play Lab etwas angeschlossen:

GND ist mit der Masse des Plug & Play Labs verbunden.

AGND ist mit der analogen Masse des Plug & Play Labs verbunden.

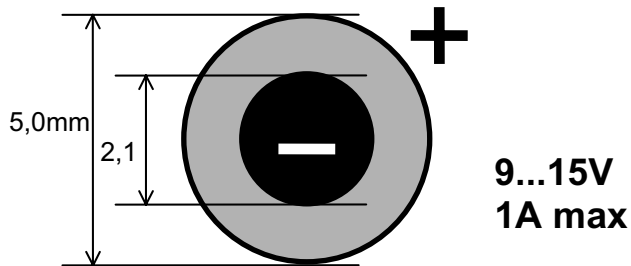
VCC ist über den „Tiger-VCC“-Jumper mit VCC des Plug & Play Labs verbunden.

Reset kann über den Taster auf GND getastet werden.

L90...L95 siehe 'serielle Anschlüsse weiter unten im Text.

Spannungsversorgung

Das Plug & Play Lab wird mit einem 10W-Netzteil mit 9...15V DC versorgt. Minus liegt auf dem Stift der Stromversorgungsbuchse. Die Stromaufnahme liegt je nach Belastung bei maximal 1A. Im Normalbetrieb werden ca. 600mA gezogen, wenn alle LED an sind (300mA ohne LEDs).



3

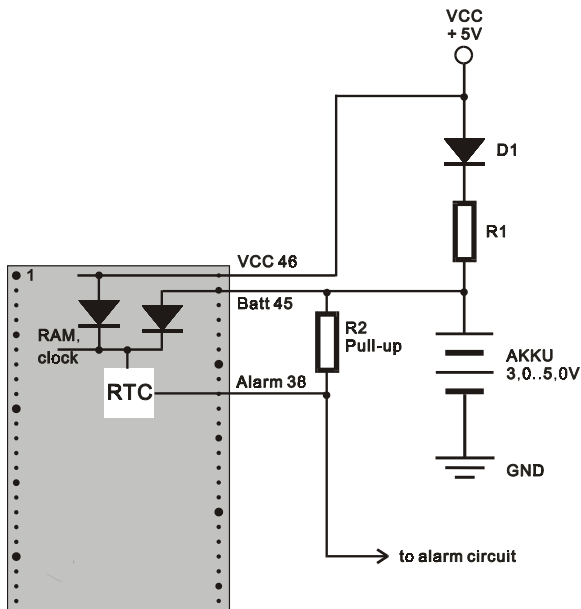
Batteriepufferung

Auf dem Plug & Play Lab befinden sich Zwei Löt pads, an denen ein Akku oder eine Batterie zum Puffern der Versorgung für RAM und Uhr angeschlossen werden kann.

Die Pads befinden sich neben dem Schalter für PC-Mode.

Die Akkuspannung sollte 3,6V betragen und darf 5V nicht überschreiten.

Beispiel für Akkupufferung mit Ladewiderstand für den Akku:

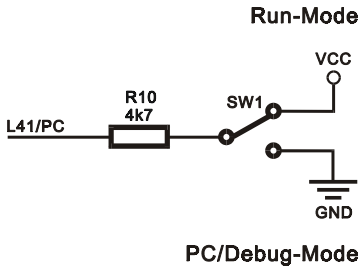


Die Belastung am Alarm-Ausgang bestimmt maßgeblich den Stromverbrauch bei Akku-Pufferung. Auf dem Plug & Play Lab ist hier ein niederohmiger Eingang des Treibers für die LED-Anzeige angeschlossen.

Sie können auf die Anzeige verzichten und den entsprechenden Pin aus der Fassung nehmen, oder für die Entwicklungsphase einen erhöhten Akku-Stromverbrauch in Kauf nehmen.

PC-Mode

Der **PC-Mode-Pin** ist auf dem Plug & Play Lab über 4k7 an den PC-Mode-Schalter gelegt.



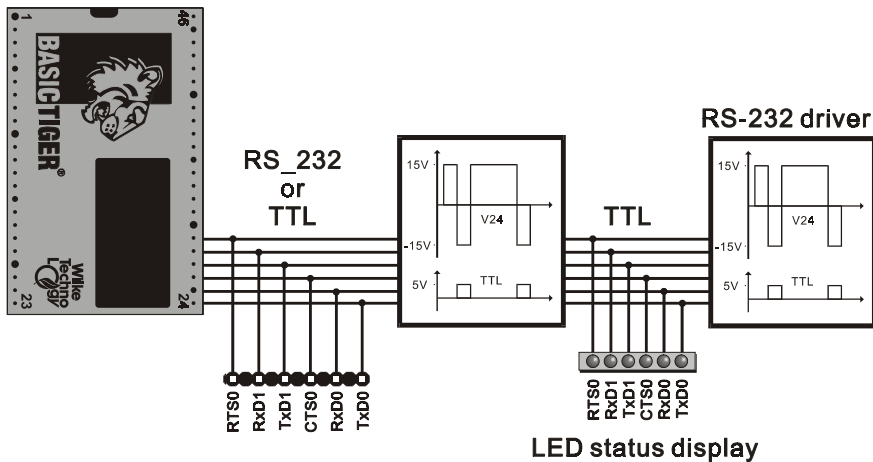
3

Nach einem Reset oder Power-down testet das BASIC-Tiger[®]-Modul, ob der PC-Mode-Pin auf 'low' liegt. Ist dies der Fall, geht das Modul in den PC-Mode/Debug-Mode. Liegt der Pin auf 'high', geht das Modul in den RUN-Mode. Die Zeit zwischen Power-on und erster Aktivität an den I/O-Pins beträgt ca. 220...230msec.



Um das BASIC-Tiger[®]-Modul in den PC-Mode zu versetzen, ist ein Reset oder Power-down erforderlich. Es reicht nicht, während des RUN-Mode den Schiebeschalter in die Stellung PC-Mode zu bringen.

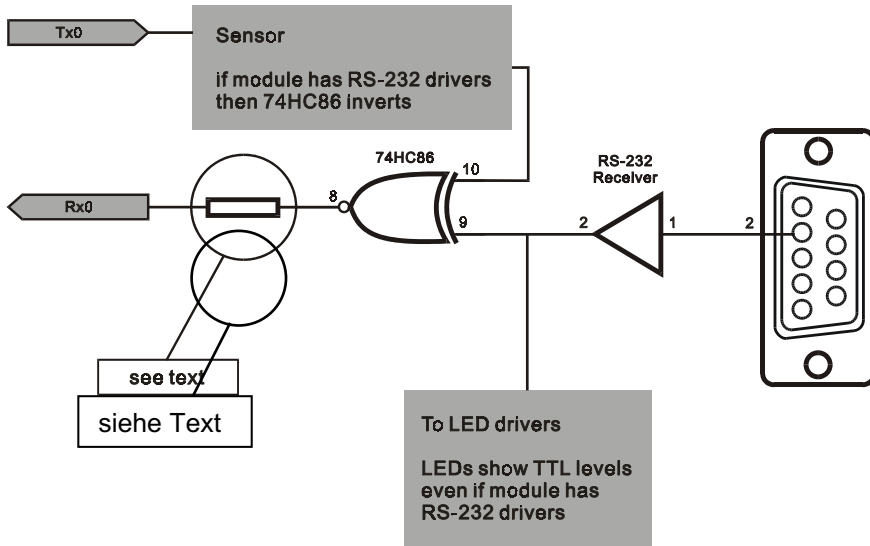
Das Plug & Play Lab ersetzt die RS-232-Treiber, wenn das gesteckte BASIC-Tiger®-Modul keine Treiber besitzt, und akzeptiert auch Tigermodule mit RS-232-Treibern. Bitte beachten Sie die Besonderheiten dieser Pins.



Die **RS-232-Pins L90...L95** werden zunächst über eine interne Logik geführt, die BASIC-Tiger[®]-Module mit oder ohne RS-232-Treiber akzeptiert. Die Signale erscheinen dann an den beiden DB9-Buchsen mit RS-232-Signalpegel. Da RS-232-Treiber das Signal invertieren, muß eine bereits erfolgte Invertierung wieder rückgängig gemacht werden, wenn das gesteckte Modul ebenfalls RS-232-Treiber enthält.

Die **LED** zeigen in jedem Fall die TTL-Pegel der Signale an, auch wenn das BASIC-Tiger®-Modul bereits RS-232-Treiber 'on-board' hat.

Auf der **Stiftleiste** sind die Pins des BASIC-Tiger®-Moduls zugänglich. Auf den Pins RxD0, CTS0 und RxD1 liegen jedoch auch die Ausgänge des 74HC86, der die seriellen Eingänge treibt. Dies ist zu beachten, wenn Sie dort normale Outputs planen. Folgendes Bild verdeutlicht noch einmal die Situation auf den 3 RS-232 Eingangspins.



3

Die Verbindung von dem Treiber 74HC86 zu dem Tigermodul ist je nach Version des Plug & Play Labs verschieden ausgeführt. Die Versionsnummer befindet sich rechts unten auf der Platine.

Plug & Play Labs V1.0 und V1.1: die Eingangs-TTL-Treiber HC86 sind direkt mit dem Modul verbunden. Um die Verbindung zu trennen, müssen IC-Beinchen aus dem Sockel gebogen werden.

Plug & Play Labs V1.2: haben 3 Jumper direkt unterhalb des Moduls, um die Eingangs-TTL-Treiber vom Modul zu trennen (Darstellung entspricht der Anordnung der Jumperpins):

Leitung	Jumperpins	
CT0	vom 74HC86	zum Modul
Rx1	vom 74HC86	zum Modul
Rx0	vom 74HC86	zum Modul

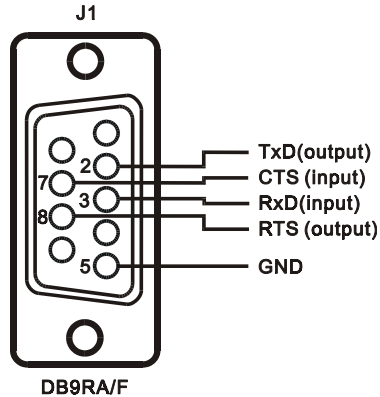
Plug & Play Labs V1.3: haben je einen Widerstand 1k5 in der Leitung zum Modul, so daß extern eingespeiste Signale auf der Stiftleiste lediglich stark genug sein müs-

Entwicklungsumgebung

sen, das Signal vom 74HC86 zu überschreiben. Eine Trennung der Leitung ist nicht erforderlich.

Pinbelegung DB9 (SER0)

Die DB9-Buchse 'ser0' ist hinten rechts angeordnet und besitzt Handshake-Leitungen. Die Buchse ist über die RS-232-Treiber und der oben beschriebenen Zusatzschaltung des Plug & Play Labs mit der Schnittstelle 'Ser0' des Moduls verbunden.

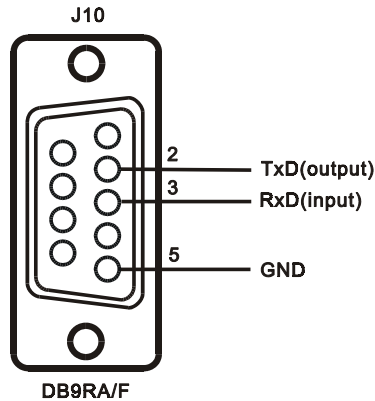


3

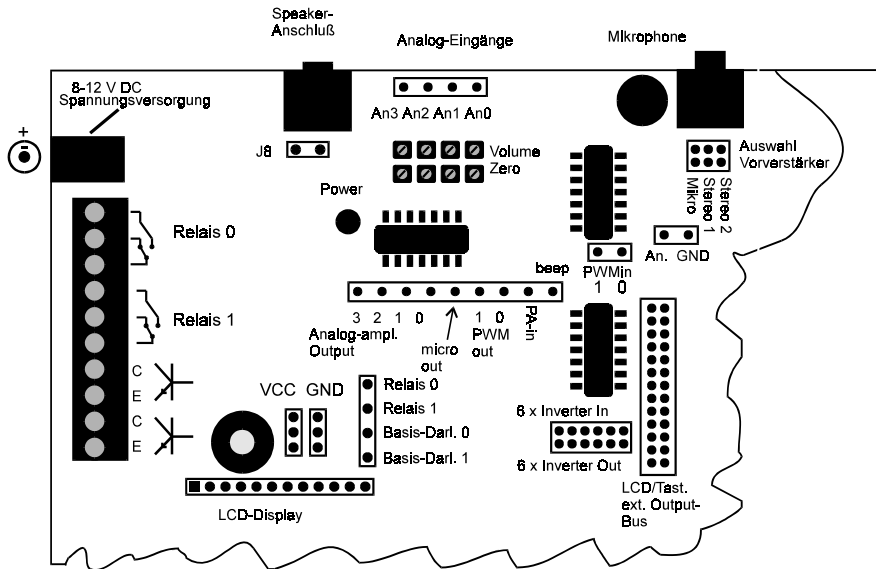
Pinbelegung DB9 (SER1/PC)

Die DB9-Buchse 'PC/ser1' ist über die RS-232-Treiber des Plug & Play Labs mit der Schnittstelle 'Ser1' des Moduls verbunden. Über diesen Anschluß wird das Modul programmiert und die Verständigung zwischen PC und Modul während des Debuggens findet hier statt. Im RUN-Modus steht der Anschluß Ihrer Anwendung zur Verfügung.

3



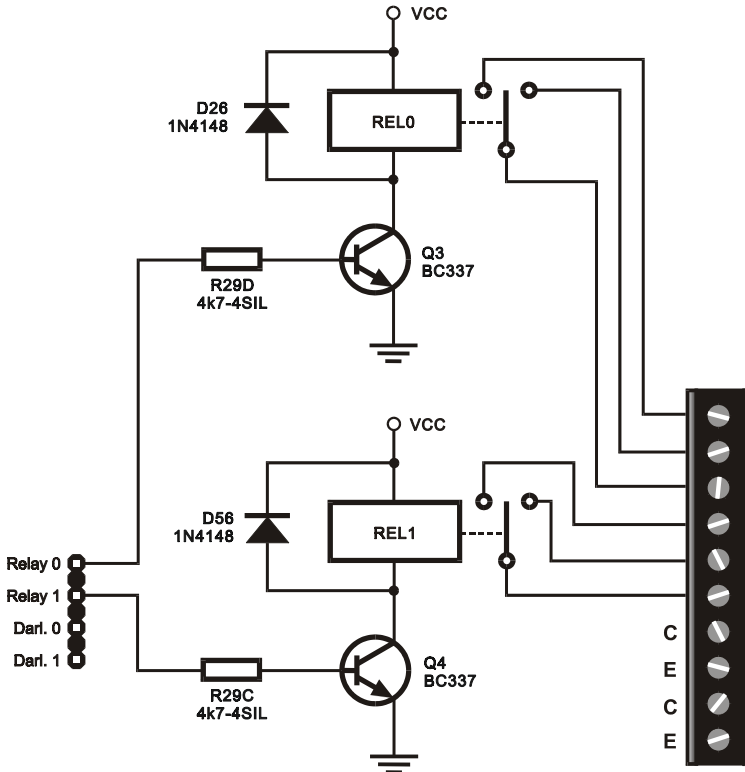
Die Abbildung zeigt die Anordnung der Analogen Elemente des Plug & Play Lab im Überblick. Die Stiftleisten des Analogeiles sind nicht mit irgendwelchen I/O-Pins des BASIC-Tiger®-Moduls verbunden. Die Versorgungsleitungen analog-VCC und analog-GND sind über Ferrite mit digital-VCC bzw. digital-GND verbunden.



110 Wilke Technology GmbH • web: www.wilke.de • eMail: info@wilke.de

Die Relais dürfen nur Spannungen unter 65V schalten. Der geschaltete Strom sollte unter 3A liegen.

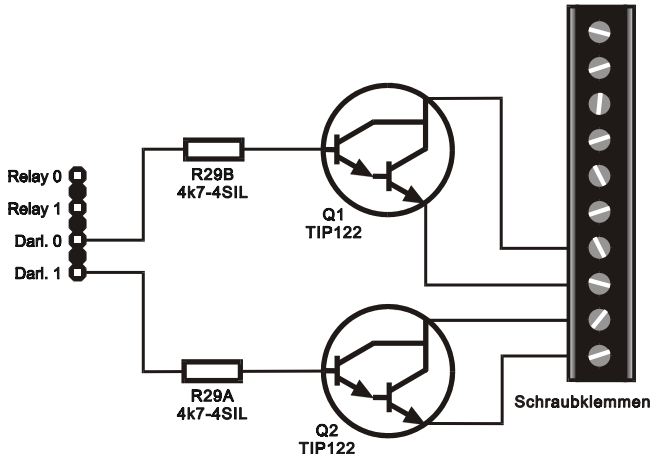
Ein beliebiger Ausgangspin des BASIC-Tiger[®]-Moduls wird auf den Pin gejumpert, der mit Relais0 oder Relais1 beschriftet ist. Ein 'high'-Pegel schaltet das Relais. Bei 'low' befindet es sich im Ruhezustand. Der Aufdruck an den Schraubklemmen beschreibt den Ruhezustand der Kontakte.



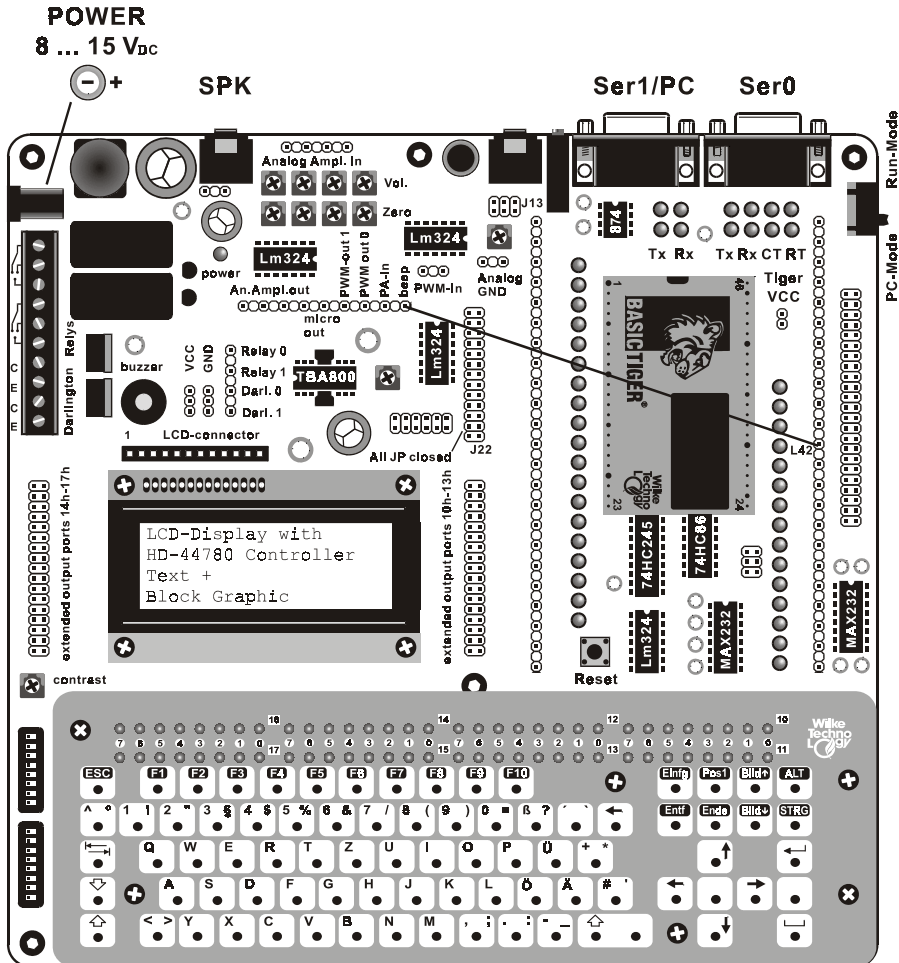
112

Die beiden NPN-Darlington-Transistoren vom Typ TIP122 dürfen bis zu ca. 2A schalten und haben eine interne Schutzdiode.

Ein beliebiger Ausgangspin des BASIC-Tiger[®]-Moduls wird auf den Pin gejumpert, der mit Darl.0 oder Darl.1 beschriftet ist. Ein 'high'-Pegel schaltet den Transistor durch. Beachten Sie, daß der Masse-Anschluß zu der externen Schaltung hergestellt werden muß.



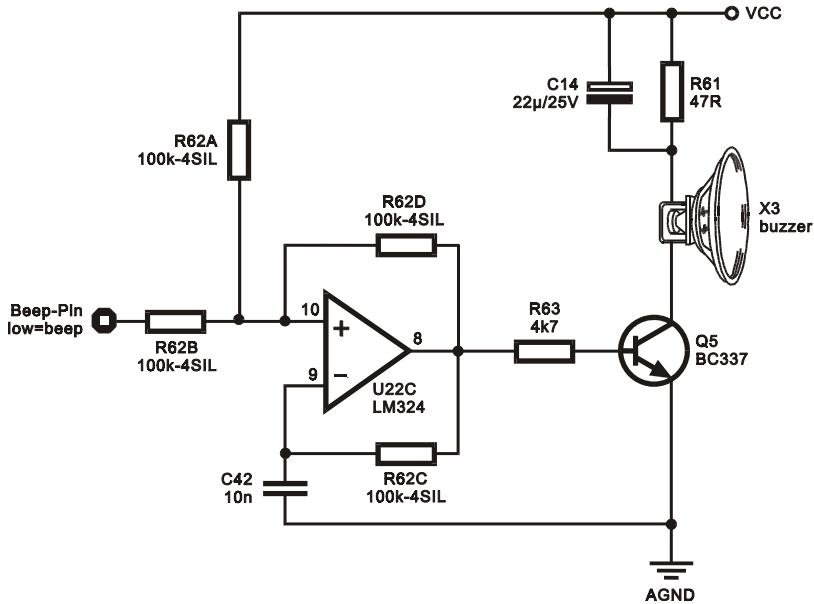
Beep



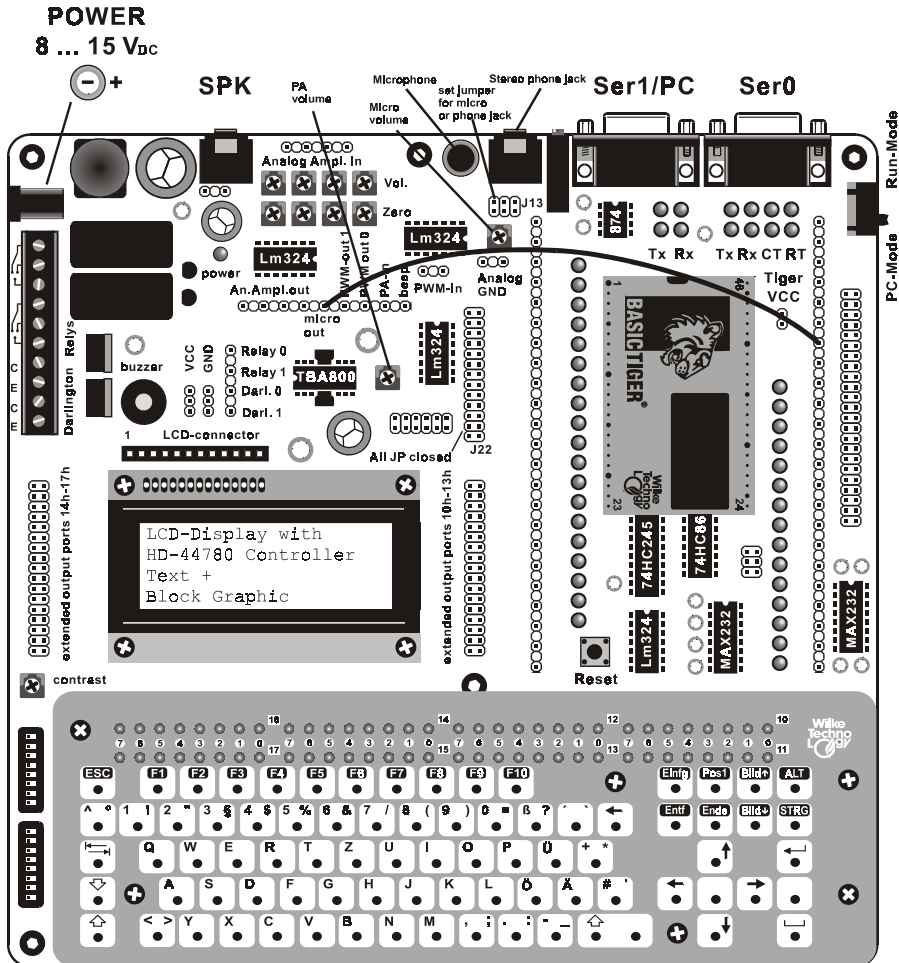
Die Steck-Kabelbrücke verbindet den BASIC-Tiger®-Ausgang L42 mit dem Eingang des Summers (beep).

Für einfache Tonerzeugung (Summer oder Beep) steht ein selbstschwingender Mini-lautsprecher zur Verfügung.

Ein beliebiger Ausgangspin des BASIC-Tiger[®]-Moduls wird auf den Pin gejumpert, der mit 'beep' beschriftet ist (ganz rechts auf der 9-poligen Stiftleiste). Wenn der Pin auf low liegt, wird der Ton erzeugt. Der LCD1-Device-Treiber benutzt für Ton und Tastenklick den Pin L42 (Pin 35).



Mikrofon-Eingang/Klinkenbuchse

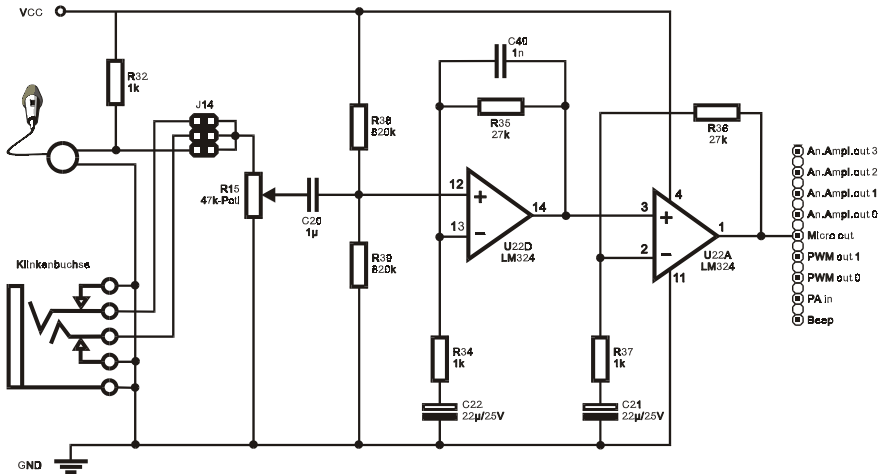


Vom Mikrofon geht es über den Jumper auf J13 zu der Lautstärke-Einstellung des Mikrofonverstärkers. Der Ausgang des Mikrofonverstärkers ist zum Mithören mit dem Eingang des Leistungsverstärkers (PA-in) und zum Messen mit einem Analog-Eingang des BASIC-Tiger®-Moduls, z.B. An0 verbunden. Die Lautstärke des Endverstärkers ist nochmal getrennt einstellbar. Analog-GND ist auf dem Plug & Play Lab intern verbunden.

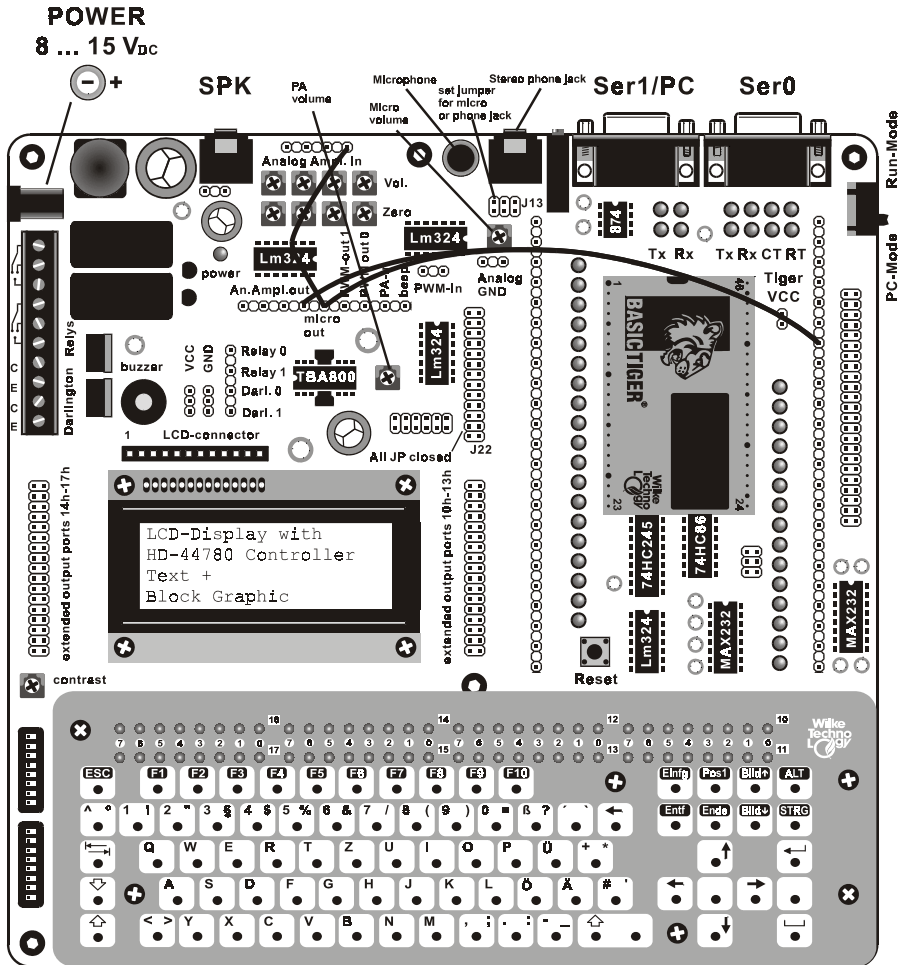
Wahlweise kann das Mikrofon oder einer der beiden Stereo-Klinken-Eingänge auf den Eingangsverstärker geschaltet werden.

Dazu wird ein Jumper auf die Stiftleiste gesteckt, die sich direkt unterhalb der Klinkenbuchse befindet (J13). Die Steckposition ganz links verbindet das Mikrofon mit dem Eingang des Verstärkers.

Mit dem Poti unterhalb des Jumperblocks J13 wird die Lautstärke geregelt. Der Ausgang des Verstärkers liegt auf der langen Stiftleiste und ist mit 'micro' beschriftet. Die Ausgangspegel liegen zwischen 0 und 5 Volt.

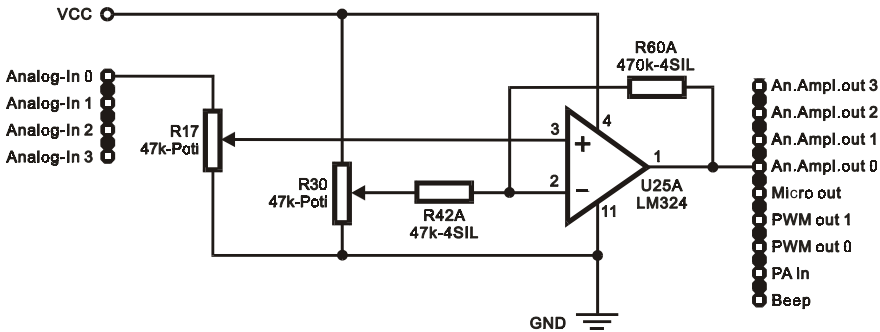


Analogverstärker (4x)



An den Eingang des ersten Analogverstärkers 'An0' ist der Ausgang des Mikrofonverstärkers angeschlossen. Der Ausgang des Analogverstärkers ist mit einem Analog-Eingang des BASIC-Tiger®-Moduls, z.B. An0 verbunden. Verstärkung und Null-Punkt werden am Besten mit Hilfe eines Oszilloskops eingestellt. Analog-GND ist auf dem Plug & Play Lab intern verbunden.

4 identische Verstärker mit verstellbarem Null-Punkt und Eingangspoti sind vorhanden. Signale, die auf den Eingang gesteckt werden, sollten am Ausgang einen DC-Pegel von ca. 2,5V aufweisen (wie z.B. der Mikrofonvorverstärker).



3

Die Eingänge befinden sich auf der Stiftleiste, die mit 'analog-in' beschriftet ist. Darunter befinden sich die Potis für Lautstärke (Vol.) und Nullpunkt (zero). Beachten Sie, daß beide Potis sich gegenseitig beeinflussen.

Die Ausgänge befinden sich auf der 9-poligen Stiftleiste und sind mit 'ampl.-out' beschriftet (Kanäle 3-2-1-0). Die Ausgangspegel liegen zwischen 0 und 4 Volt.

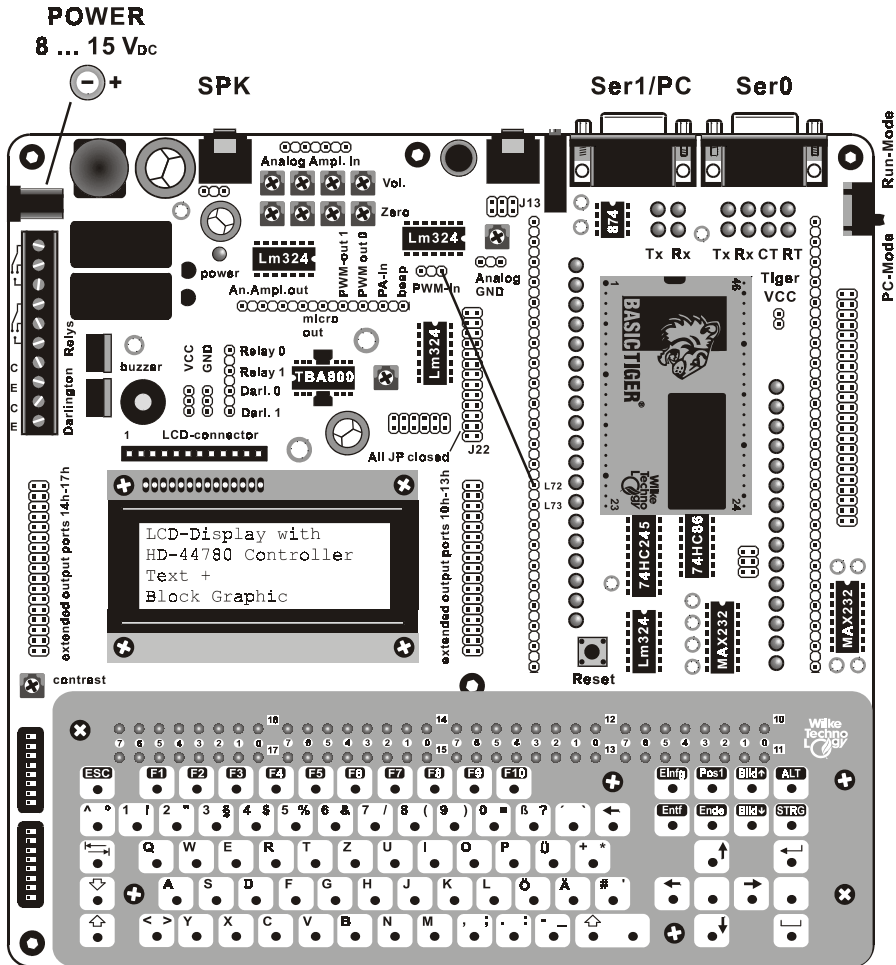
Analog-Eingänge

Die analogen Inputpins des BASIC-Tiger[®]-Moduls **An0...An3** haben einen Eingangswiderstand von 1M und eine Hardware-Auflösung von 10 Bit. Die Auflösung kann durch Berechnung noch gesteigert werden.

Die LEDs zeigen über CMOS-Treiber High und Low an. Die Analog-Pins auf dem BASIC-Tiger[®]-Modul A sind immer Inputs. Beachten Sie, daß die Eingänge der LED-Treiber auch an den Pins An0...An3 liegen, wodurch der Eingangswiderstand herabgesetzt ist.

Die Referenzspannung 'Vref' wird am Spindelpoti neben den DB9-Steckverbindern eingestellt. Die Spannung darf zwischen 3,5V und VCC (5V) liegen. Der Meßbereich wird zwischen 0 und der Referenzspannung aufgespannt. Eingangsspannungen, die gleich der Referenzspannung sind, ergeben den maximalen Meßwert.

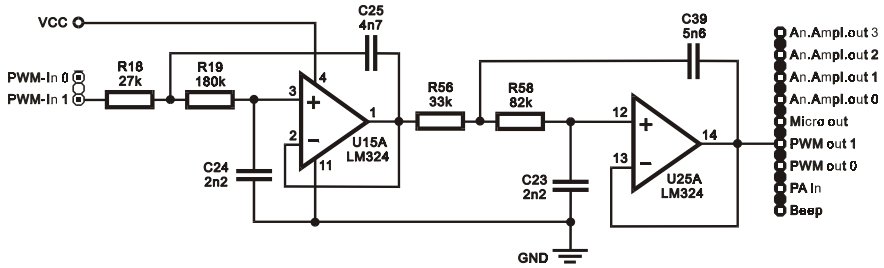
PWM-Verstärker



Ein PWM-Ausgang des Tiger-BASIC®-Moduls ist über einen Spannungsteiler mit dem Eingang des ersten PWM-Verstärkers verbunden. Ohne Spannungsteiler würde sich eine Verzerrung ergeben, da die Opamps auf dem Board nur mit 5V versorgt werden. Der Ausgang des PWM-Verstärkers wird über einen Widerstand mit dem Eingang des Endverstärkers verbunden. Der Widerstand (z.B. 470k) reduziert die Eingangs-Empfindlichkeit des Endverstärkers, wodurch die Lautstärke besser einstellbar ist.

2 PWM-Verstärker sind vorhanden. Die 2-polige Stiftleiste für Kanal 1-0-Input befindet sich etwas nach rechts-oben versetzt zu der 9-poligen Stiftleiste.

Die Ausgänge erscheinen auf der 9-poligen Stiftleiste (PWM buff., Kanäle 1-0).



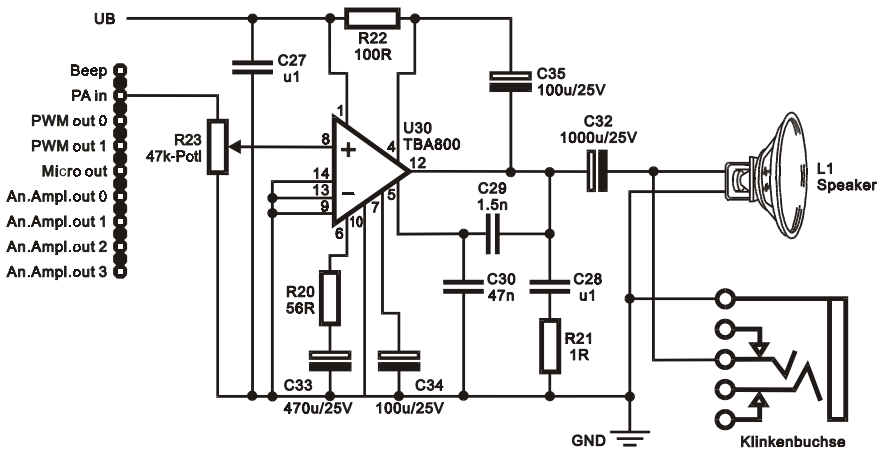
Power-Verstärker

Ein kleiner Leistungsverstärker hat seinen Input auf dem zweiten Pin von rechts auf der 9-poligen Stiftleiste, beschriftet mit 'PA-in'.

Der Ausgang liegt sowohl auf der Klinkenbuchse für den Lautsprecher als auch auf der direkt benachbarten 2-poligen Stiftleiste J8.

Mit einem Vorwiderstand vor 'PA-in' kann die Empfindlichkeit des Eingangspotis reduziert werden.

3

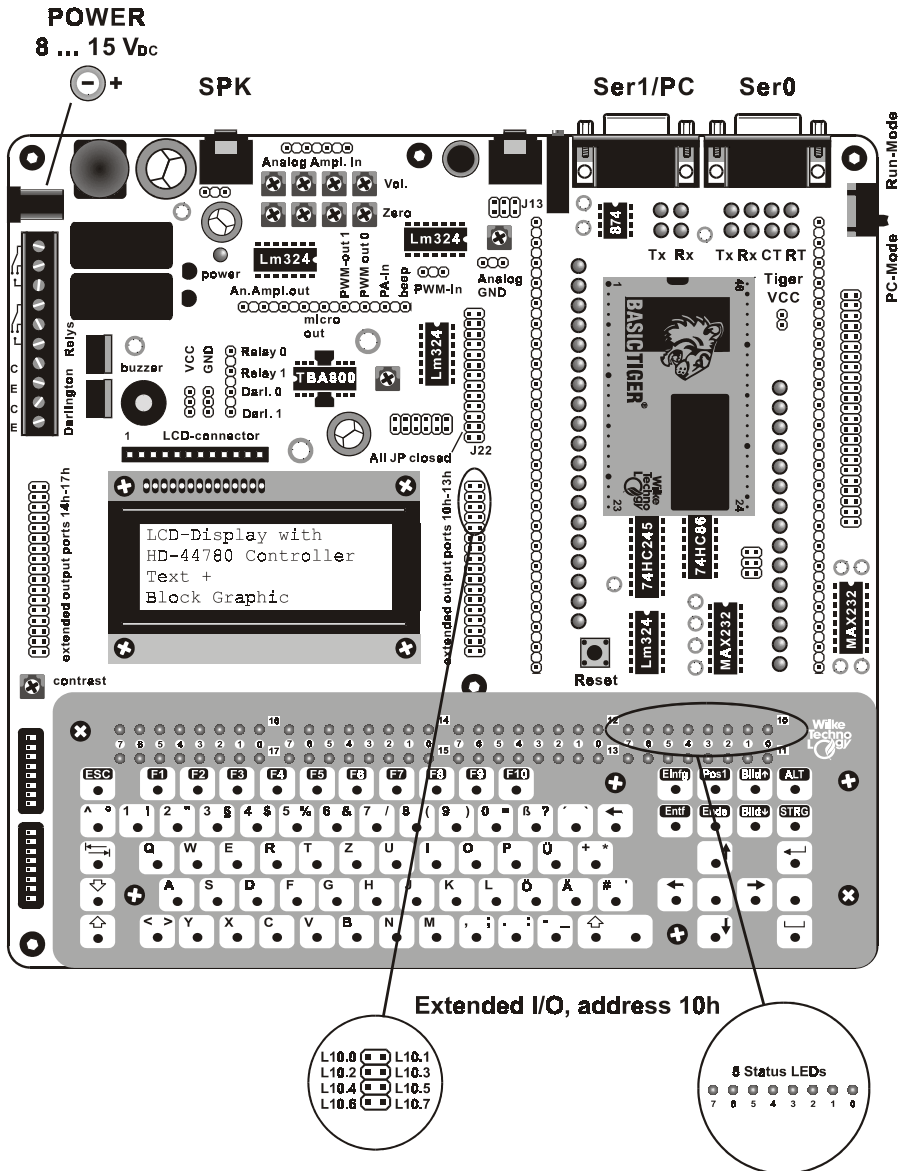


Leere Seite

3

Entwicklungsumgebung

3



Bus-System für LCD, Tastatur und 64 I/O-Pins

Das **LC-Display**, die **Tastatur** sowie die **zusätzlichen Output-Pins** benutzen gemeinsam einen Datenbus und haben jeweils eigene Steuerleitungen zur Aktivierung der verschiedenen Geräte. Alle Leitungen (bis auf 'beep') sind über 13 Jumper auf die passenden Anschlüsse des BASIC-Tiger[®] steckbar. Die langen Steckkabel brauchen somit nicht verwendet zu werden. Weitere Device-Treiber werden ebenfalls diesen Datenbus verwenden.

Wenn die 13 Jumper auf J22 stecken, dann bestehen folgende Verbindungen zum BASIC-Tiger[®]-Modul:

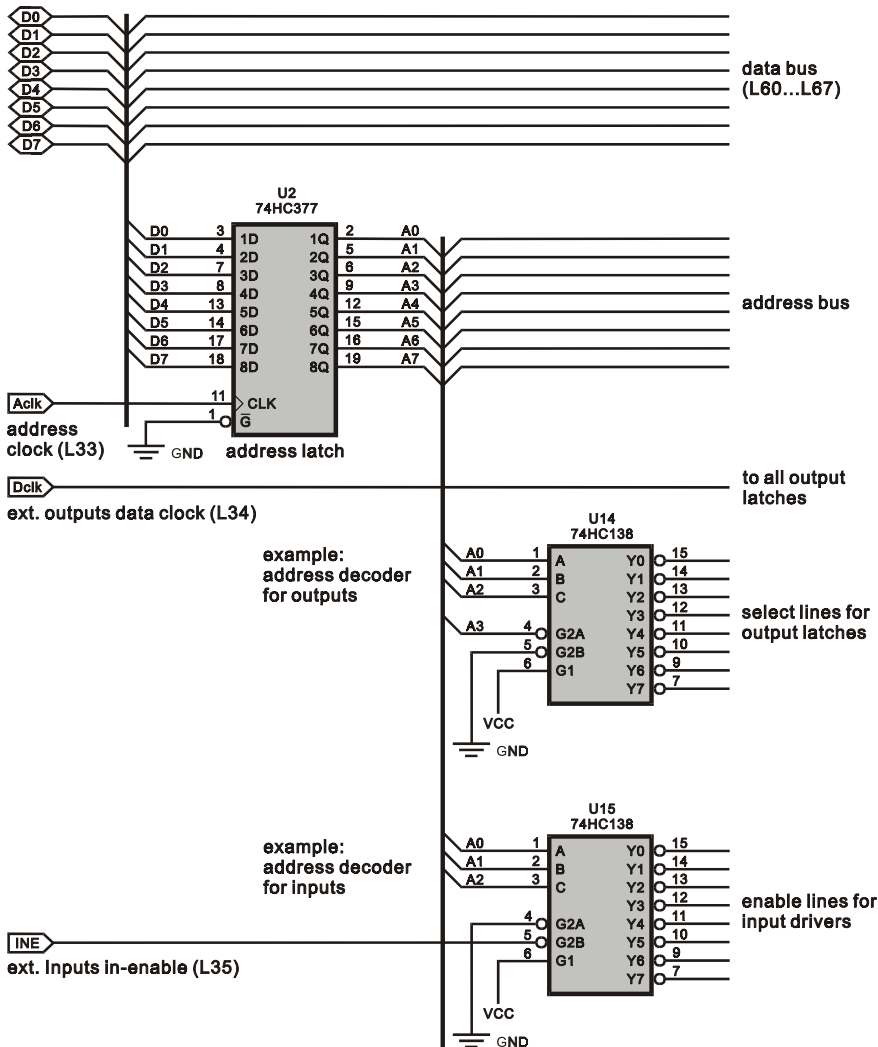
Bus-Bezeichnung	BASIC-Tiger [®] -Pins	Modultyp-A Pin-Nr.	Tiny-Tiger Pin-Nr.
D0...D7	L60...L67	2...9	1...8
Aclk (Adressclock)	L33	30	29
Dclk (Datenclock)	L34	31	30
keyb (Tastatur enable)	L35	32	31
E (LCD: enable)	L36	33	32
RS (LCD: Reg.select)	L37	34	33
beep (nicht auf J22)	L42	35	*
busy-in (PRN1)	L70	10	9
strobe-out (PRN1)	L71	11	10
busy-out (PIN1)	L80	14	13
strobe-in (PIN1)	L81	15	14

* Der Standard-Beep-Pin L42 ist bei Tiny-Tiger[®] nicht vorhanden und wird bei der Installation von LCD1.TDD auf einen anderen Pin verlegt.

Alle Leitungen werden vom Device-Treiber „LCD1.TDD“ automatisch richtig bedient. Zur Benutzung der erweiterten I/O-Pins kein Device-Treiber erforderlich.

Entwicklungsumgebung

Die Abbildung zeigt die Grundlage der erweiterten I/Os, LCD- und Drucker-Anschluß und Tastatur. Die Adreßbus geht von einem 8-Bit-Speicher aus, in den die Adresse mit dem 'Aclk'-Signal eingeschrieben wird.



Weitere Informationen zum Anschluß und zur Benutzung der Komponenten finden Sie an anderer Stelle des Handbuches. Dort wird auch der Aufbau des Plug & Play Labs näher erläutert.

Thema	Kapitel	Seite(n)
Erweiterte I/O allgemein	Erweitertes I/O-System	305 ff
Tastatur	Tastatur anschließen	323
Erweiterte Outputs	Erweiterung der Ausgänge	310
Erweiterte Inputs	Erweiterung der Eingänge	313

Belegung J12

Die 2-reihige Pinleiste J12 am rechten Rand des Plug & Play Labs ist 1:1 mit den Pins des Tigermoduls verbunden. Hier können z.B. eigene Aufbauten auf Lochraster oder ein selbst angefertigtes Centronics-Kabel aufgesteckt werden.

reserved	1	2	VCC
L60	3	4	Battery
L61	5	6	AGND
L62	7	8	Vref
L63	9	10	Analog-3
L64	11	12	Analog-2
L65	13	14	Analog-1
L66	15	16	Analog-0
L67	17	18	Alarm
L70	19	20	L41 / PC
L71	21	22	L40
L72	23	24	L42
L73	25	26	L37
L80	27	28	L36
L81	29	30	L35
L82	31	32	L34
L83	33	34	L33
L84	35	36	L95 / RTS0
L85	37	38	L94 / RxD1
L86	39	40	L93 / TxD1
L87	41	42	L92 / CTS0
Reset	43	44	L91 / RxD0
GND	45	46	L90 / TxD0

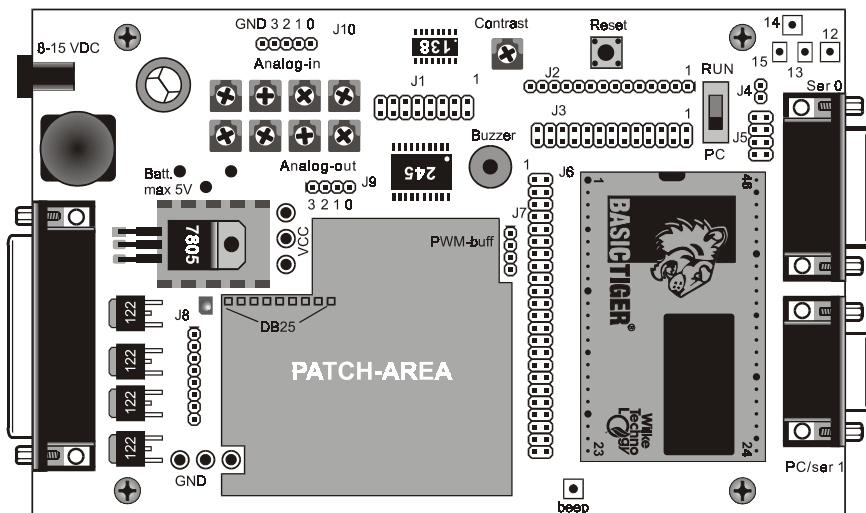
BASIC Tiger® Prototyping-Board

Viele Anwendungen können ohne Erstellung einer eigenen Platine direkt mit dem BASIC Tiger®-Prototyping-Board verwirklicht werden. Das Board hat die Standardgröße 100x160mm und paßt in Euro-Gehäuse oder 19"-Racks. Ebenso kann das Board einfach offen als Entwicklungsboard verwendet werden. Die 4-Lagen-Platine mit ihren Power- und GND-Lagen sorgt für niedrige Emissionswerte und erleichtert das Bestehen eines CE-Tests.

Das BASIC Tiger®-Prototyping-Board enthält:

- direkte PC-Verbindung über einen DB9-RS-232-Port.
- zweite serielle RS-232-Schnittstelle steht auf einer Stiftleiste zur Verfügung.
- 16-Pin Tastatur-Anschluß für Tastaturen mit bis zu 64 Tasten oder DIP-Schaltern
- Reset-Knopf
- Summer für Tonausgabe
- 14-pin Steckverbinder für LC-Display
- 8 Powertransistoren verbunden mit DB25-Stecker
- 4 Analogverstärker (LM324)
- DB15-Stecker
- 2-Kanal-PWM-Verstärker mit Filter

Eine kurze Übersicht soll Ihnen zunächst das BASIC Tiger®-Prototyping-Board näher bringen:



Entwicklungsumgebung

Das Bild zeigt den Steckplatz für das BASIC-Tiger[®] Modul A, den Anschluß für den PC (serielle Schnittstelle 1) sowie den PC-Mode-Schalter als wichtigste Elemente für den Einstieg. Auf den Steckpins links des Tigermoduls sind alle Anschlußpins des Moduls zugänglich.

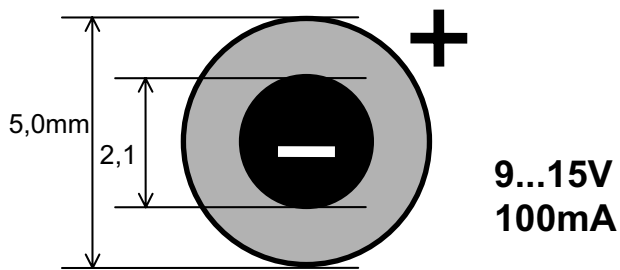
Generell ist Vorsicht geboten, wenn per Steckkabel Verbindungen hergestellt werden: Pins, die Outputs sind, dürfen nicht mit VCC oder GND kurzgeschlossen werden oder gar zu hohe Spannung erhalten.

An einigen Pins ist bereits auf dem BASIC-Tiger[®] Prototyping-Board etwas angeschlossen:

- **GND** ist mit der Masse des BASIC-Tiger[®] Prototyping-Boards verbunden.
- **AGND** ist mit der analogen Masse des BASIC-Tiger[®] Prototyping-Boards verbunden.
- **VCC** ist über den „Tiger-VCC“-Jumper mit VCC des BASIC-Tiger[®] Prototyping-Boards verbunden.
- **Reset** kann über den Taster auf GND getastet werden.
- **L90...L95** siehe ‘serielle Anschlüsse weiter unten im Text.

Spannungsversorgung

Das BASIC-Tiger® Prototyping-Board wird mit einem 5W-Netzteil mit 9...15V DC versorgt. Minus liegt auf dem Stift der Stromversorgungsbuchse. Die Stromaufnahme liegt je nach Belastung bei ca. 100mA.



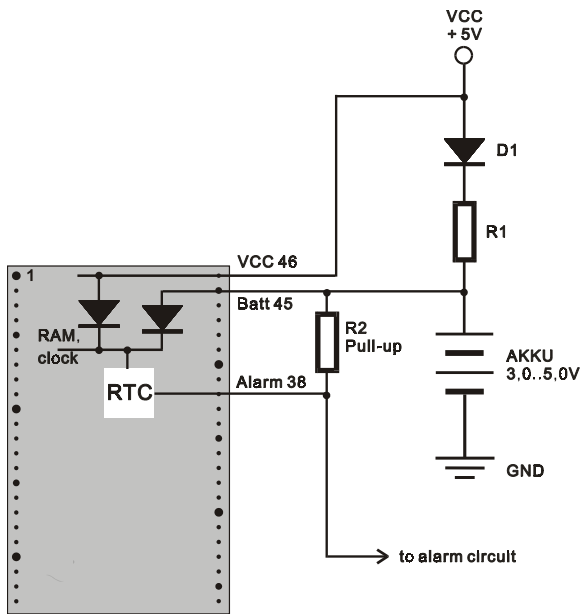
Batteriepufferung

Auf dem BASIC-Tiger[®] Prototyping-Board befinden sich Löt pads, an denen ein Akku oder eine Batterie zum Puffern der Versorgung für RAM und Uhr angeschlossen werden kann.

Die Pads befinden sich zwischen Kühlblech und Elko. Für 'Plus' gibt es 2 Löt pads, das versetzte Pad in der Mitte ist der Minus-Anschluß.

Die Akkuspannung sollte 3,6V betragen und darf 5V nicht überschreiten.

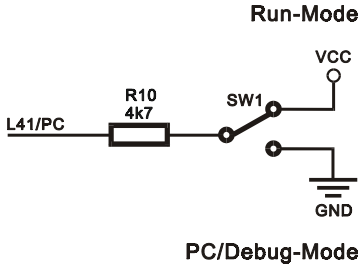
Beispiel für Akkupufferung mit Ladewiderstand für den Akku:



Die Belastung am Alarm-Pin bestimmt maßgeblich den Stromverbrauch im Zustand der Akkupufferung.

PC-Mode

Der **PC-Mode-Pin** ist auf dem BASIC-Tiger® Prototyping-Board über 4k7 an den PC-Mode-Schalter gelegt.



3

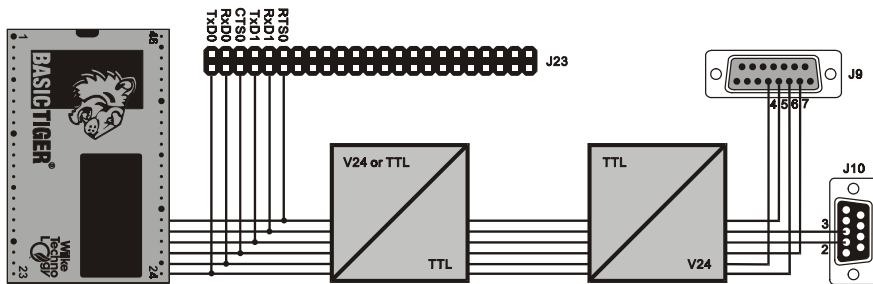
Nach einem Reset oder Power-down testet das BASIC-Tiger®-Modul, ob der PC-Mode-Pin auf 'low' liegt. Ist dies der Fall, geht das Modul in den PC-Mode/Debug-Mode. Liegt der Pin auf 'high', geht das Modul in den RUN-Mode. Die Zeit zwischen Power-on und erster Aktivität an den I/O-Pins beträgt ca. 220...230msec.



Um das BASIC-Tiger®-Modul in den PC-Mode zu versetzen, ist ein Reset oder Power-down erforderlich. Es reicht nicht, während des RUN-Mode den Schiebeschalter in die Stellung PC-Mode zu bringen.

Serielle Anschlüsse

Das BASIC-Tiger[®] Prototyping-Board ersetzt die RS-232-Treiber, wenn das gesteckte BASIC-Tiger[®]-Modul keine Treiber besitzt, und akzeptiert auch Tigermodule mit RS-232-Treibern. Bitte beachten Sie die Besonderheiten dieser Pins.



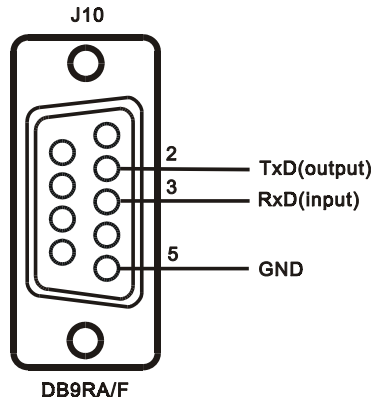
Die **RS-232-Pins L90...L95** werden zunächst über eine interne Logik geführt, die BASIC-Tiger[®]-Module mit oder ohne RS-232-Treiber akzeptiert. Die Signale erscheinen dann an der DB9-Buchse für 'Ser1' und der Stiftleiste für 'Ser0' mit RS-232-Signalpegel.. Da RS-232-Treiber das Signal invertieren, muß eine bereits erfolgte Invertierung wieder rückgängig gemacht werden, wenn das gesteckte Modul ebenfalls RS-232-Treiber enthält.

Auf der **Stiftleiste** sind die Pins des BASIC-Tiger[®]-Moduls zugänglich. Auf den Pins RxD0, CTS0 und RxD1 liegen jedoch auch die Ausgänge des 74HC86, der die seriellen Eingänge treibt. Dies ist zu beachten, wenn Sie dort normale Outputs planen. Dann muß eventuell ein 74HC86 ausgelötet werden oder durch herausbiegen von Anschlußpins teilweise außer Betrieb gesetzt werden.

Bitte ziehen Sie das mitgelieferte Schaltbild hinzu, wenn Unklarheiten über die Anschlußgegebenheiten bestehen.

Pinbelegung DB9

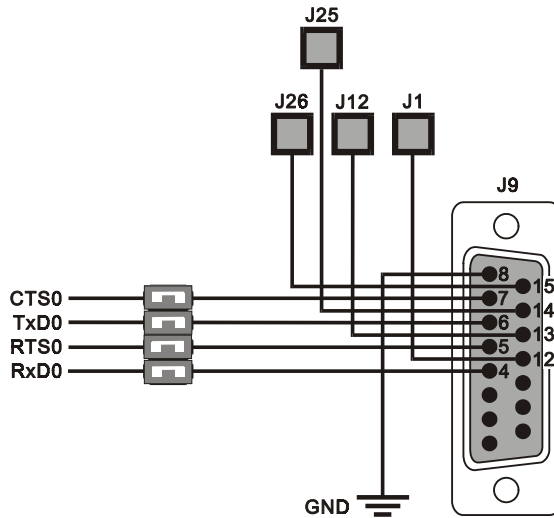
Die DB9-Buchse 'PC/ser1' ist über die RS-232-Treiber des BASIC-Tiger® Prototyping-Boards mit der Schnittstelle 'Ser1' des Moduls verbunden. Über diesen Anschluß wird das Modul programmiert und die Verständigung zwischen PC und Modul während des Debuggens findet hier statt. Im RUN-Modus steht der Anschluß Ihrer Anwendung zur Verfügung.



3

Pinbelegung DB15

Der DB15-Stecker J9 kann über Jumper mit der seriellen Schnittstelle 0 des Tiger-Moduls verbunden werden. 4 weitere Anschlüsse sind frei und können z.B. für Ihre eigenen RS-422 oder RS485 Treiber verwendet werden.



Die folgende Tabelle zeigt die Pinbelegung am DB15-Stecker, wie sie entsteht, wenn die Stiftleiste mit Jumpers bestückt wird. Die übrigen Spalten zeigen, wie eine 9-polige bzw. 25-polige PC-Schnittstelle verbunden wird.

Signal	DB15	DB9 (PC)	DB25 (PC)
Tx0 (output)	6	2	3
Rx0 (input)	4	3	2
RT0 (output)	5	8	5
CT0 (input)	7	7	4
GND	8	5	7

Pinbelegung 'Serl 0'-Stiftleiste

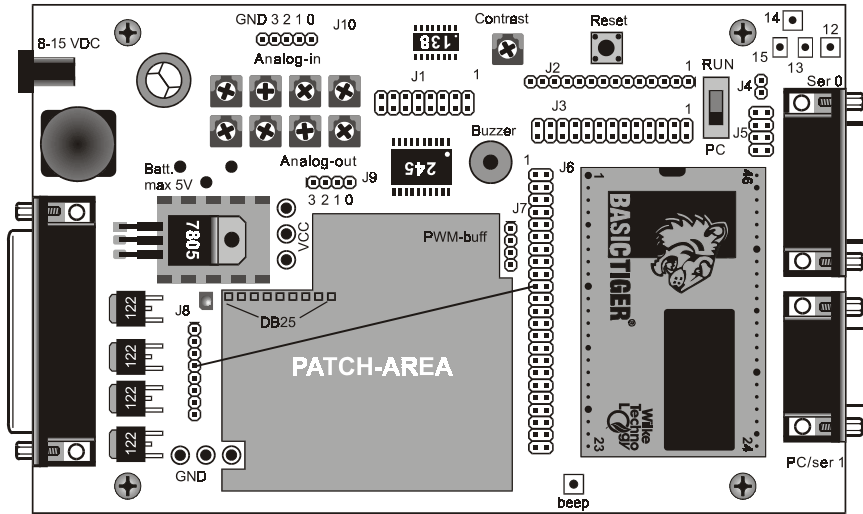
Auf der 2-reihigen Stiftleiste J8/13/14/15 können die Leitungen der Schnittstelle 'Ser 0' direkt mit dem DB15-Stecker oder über freie Verdrahtung mit einem eigenen Steckverbinder verbunden werden .

CTS		Pin 7 auf DB15-Stecker
TxD		Pin 6 auf DB15-Stecker
RTS		Pin 5 auf DB15-Stecker
RxD		Pin 4 auf DB15-Stecker

Ein GND-Anschluß befindet sich z.B. auf 2 Pads des unbestückten Tasters.

Darlington transistors (NPN)

3

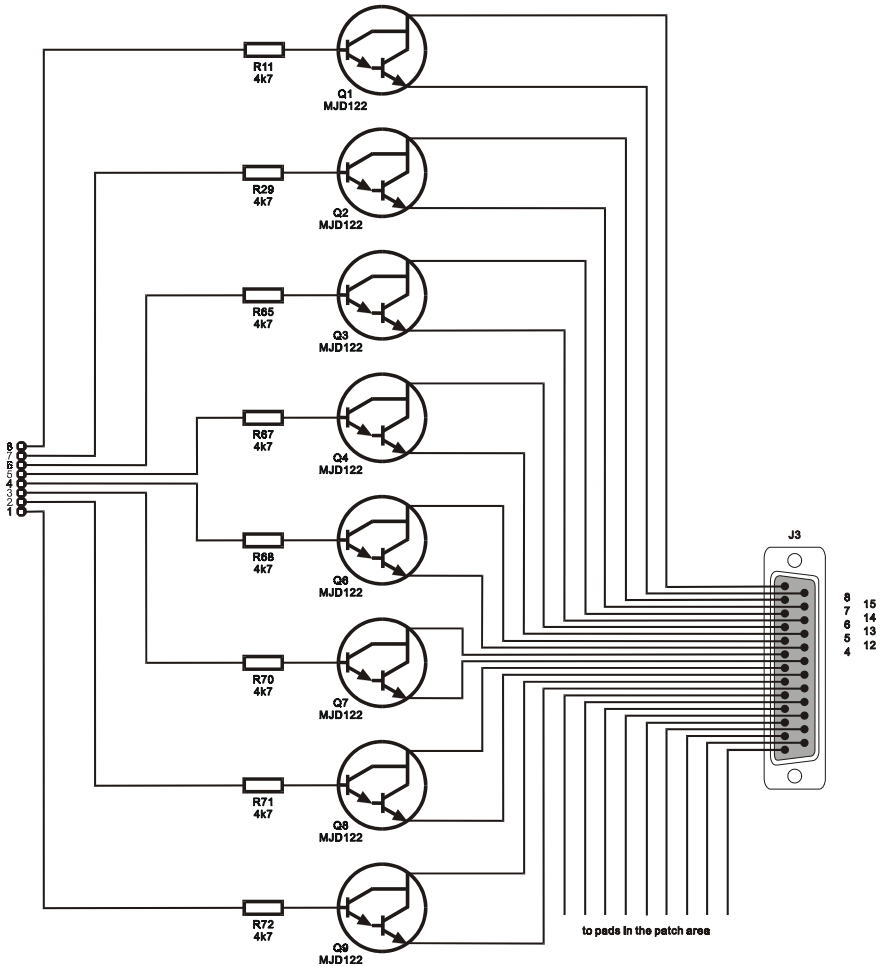


Die Steck-Kabelbrücke verbindet den BASIC-Tiger®-Ausgang L70 mit einem Eingang der Darlington-Transistoren.

BASIC-Tiger® Prototyping-Board

Die 8 NPN-Darlington-Transistoren vom Typ TIP122 dürfen bis zu ca. 2A schalten und haben eine interne Schutzdiode.

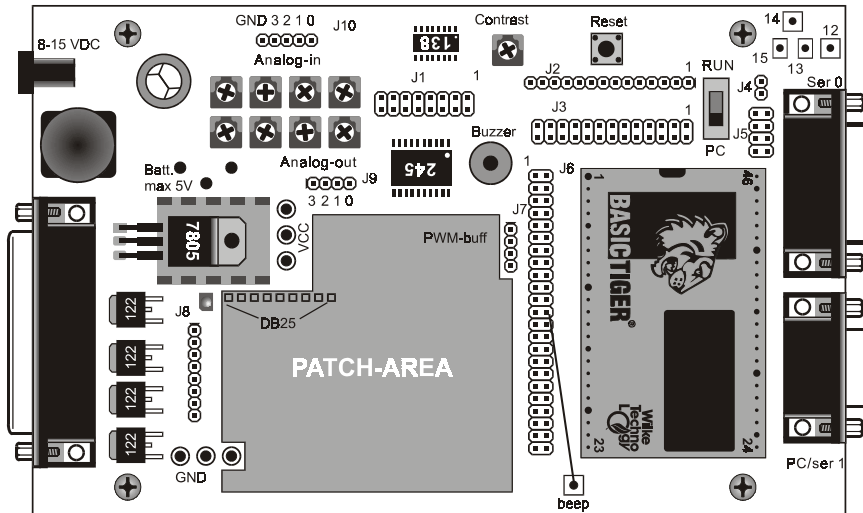
Ein beliebiger Ausgangspin des BASIC-Tiger®-Moduls wird mit einem Lötpad eines Darlingtons verbunden. Die Beschriftung kennzeichnet die Ausgangs-Pin-Nummer am DB25. Ein 'high'-Pegel schaltet den Transistor durch. Beachten Sie, daß der Masse-Anschluß zu der externen Schaltung hergestellt werden muß.



3

Beep

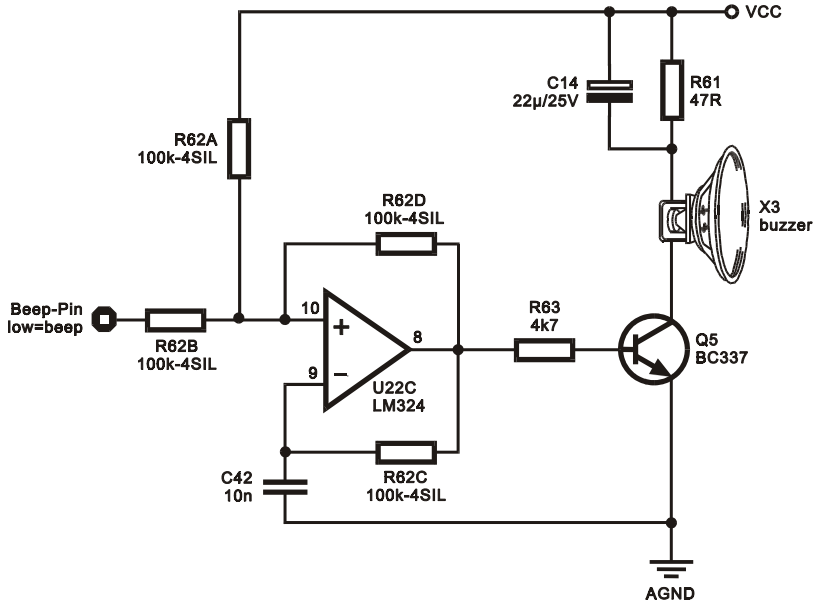
3



Die Steck-Kabelbrücke verbindet den BASIC-Tiger®-Ausgang L42 mit dem Eingang des Summers (beep). Für einfache Tonerzeugung (Summer oder Beep) steht ein selbstschwingender Minilautsprecher zur Verfügung.

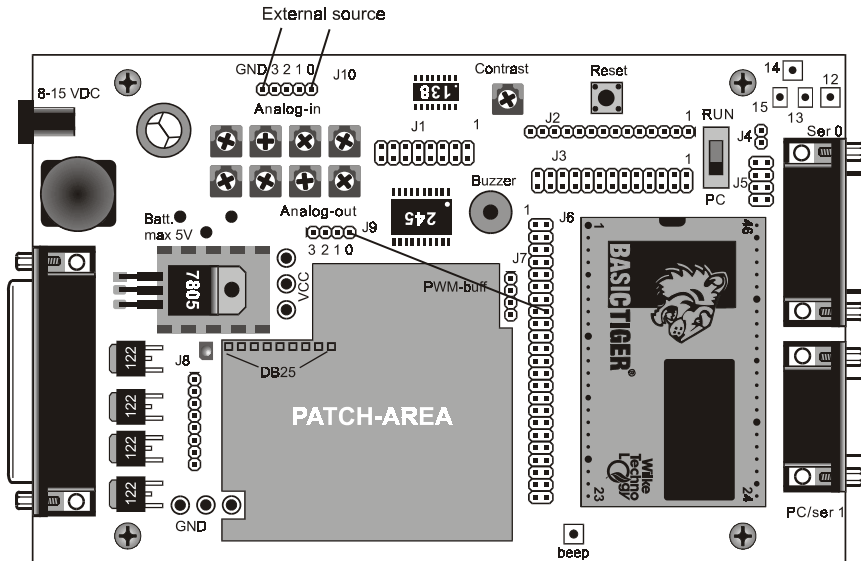
BASIC-Tiger® Prototyping-Board

Ein beliebiger Ausgangspin des BASIC-Tiger®-Moduls wird mit dem Pad verbunden, welches mit 'beep' beschriftet ist (unterhalb des Moduls). Wenn der Pin auf low liegt, wird der Ton erzeugt. Der LCD1-Device-Treiber benutzt für Ton und Tastenklick den Pin L42 (Pin 35).



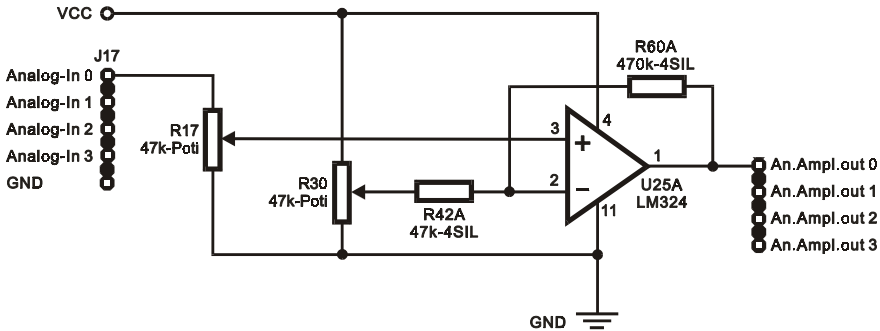
3

Analogverstärker (4x)



Eine externe Wechselspannungsquelle mit einem Offset von 2.5V DC ist an den Eingang des ersten Analogverstärkers 'An0' angeschlossen. Der Analogverstärker-Ausgang ist mit dem analogen Eingang des BASIC Tiger®-Moduls, z.B. An0 verbunden. Verstärkung und Nullpunkt werden am besten mit Hilfe eines Oszilloskops eingestellt. Analog-GND ist intern auf dem BASIC-Tiger® Prototyping-Board verbunden.

4 identische Verstärker mit verstellbarem Null-Punkt und Eingangspoti sind vorhanden. Signale, die auf den Eingang gesteckt werden, sollten am Ausgang einen DC-Pegel von ca. 2,5V aufweisen (wie z.B. der Mikrofonvorverstärker).



3

Die Eingänge befinden sich auf der Stiftleiste, die mit 'analog-in' beschriftet ist. Darunter befinden sich die Potis für Lautstärke (Vol.) und Nullpunkt (zero). Beachten Sie, daß beide Potis sich gegenseitig beeinflussen.

Die Ausgänge befinden sich auf der 4-poligen Stiftleiste und sind mit 'analog-out' beschriftet (Kanäle 3-2-1-0). Die Ausgangspegel liegen zwischen 0 und 4 Volt.

Analog inputs

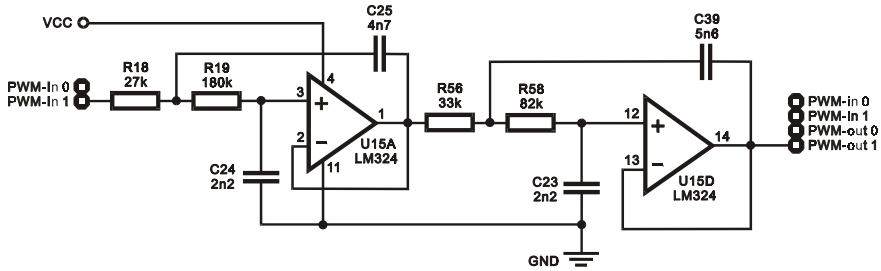
Die analogen Inputpins des BASIC-Tiger®-Moduls **An0...An3** haben einen Eingangswiderstand von 1M und eine Hardware-Auflösung von 10 Bit. Die Auflösung kann durch Berechnung noch gesteigert werden.

Eine externe Referenzspannung wird am BASIC-Tiger®-Pin 'Vref' eingespeist. Die Spannung darf zwischen 3,5V und VCC (5V) liegen. Der Meßbereich wird zwischen 0 und der Referenzspannung aufgespannt. Eingangsspannungen, die gleich der Referenzspannung sind, ergeben den maximalen Meßwert.

144 Wilke Technology GmbH • web: www.wilke.de • eMail: info@wilke.de

2 PWM-Verstärker sind vorhanden.

Die 4-polige Stiftleiste für Kanal 1-0-Input befindet sich links neben dem Modul unterhalb des Summers.



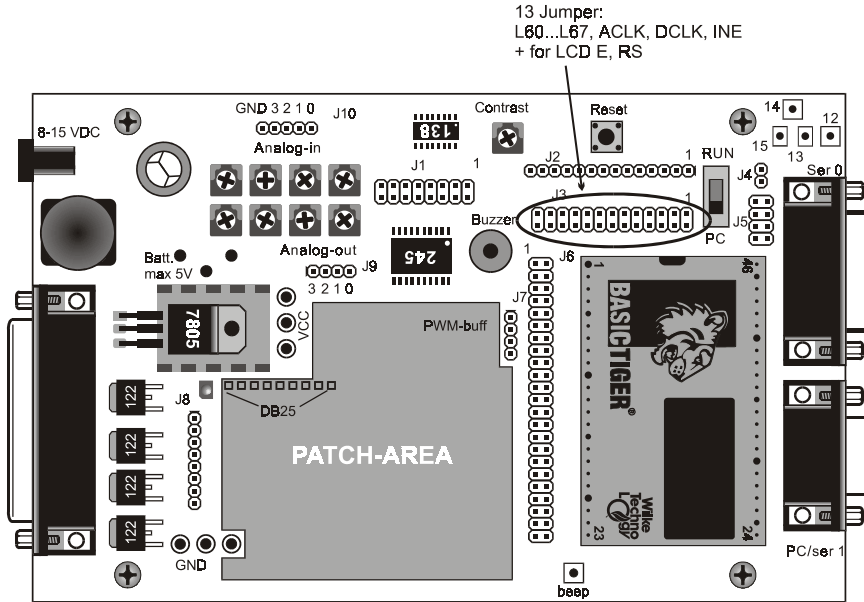
3

PWM-Verstärker-Stiftleiste:

PWM-in 0	1
PWM-in 1	2
PWM-out 0	3
PWM-out 1	4

Bus-System für LCD, Tastatur und erw. I/O-Pins

3



BASIC-Tiger® Prototyping-Board

Das **LC-Display** und die **Tastatur** benutzen gemeinsam einen Datenbus und haben jeweils eigene Steuerleitungen zur Aktivierung. Alle Leitungen (bis auf 'beep') sind über 13 Jumper auf die passenden Anschlüsse des BASIC-Tiger® steckbar.

Wenn die 13 Jumper auf J22 stecken, dann bestehen folgende Verbindungen zum BASIC-Tiger®-Modul:

Bus-Bezeichnung	BASIC-Tiger®-Pins	Modultyp-A Pin-Nr.	Tiny-Tiger Pin-Nr.
D0...D7	L60...L67	2...9	1...8
Aclk (Adressclock)	L33	30	29
Dclk (Datenclock)	L34	31	30
keyb (Tastatur enable an INE)	L35	32	31
E (LCD: enable)	L36	33	32
RS (LCD: Reg.select)	L37	34	33

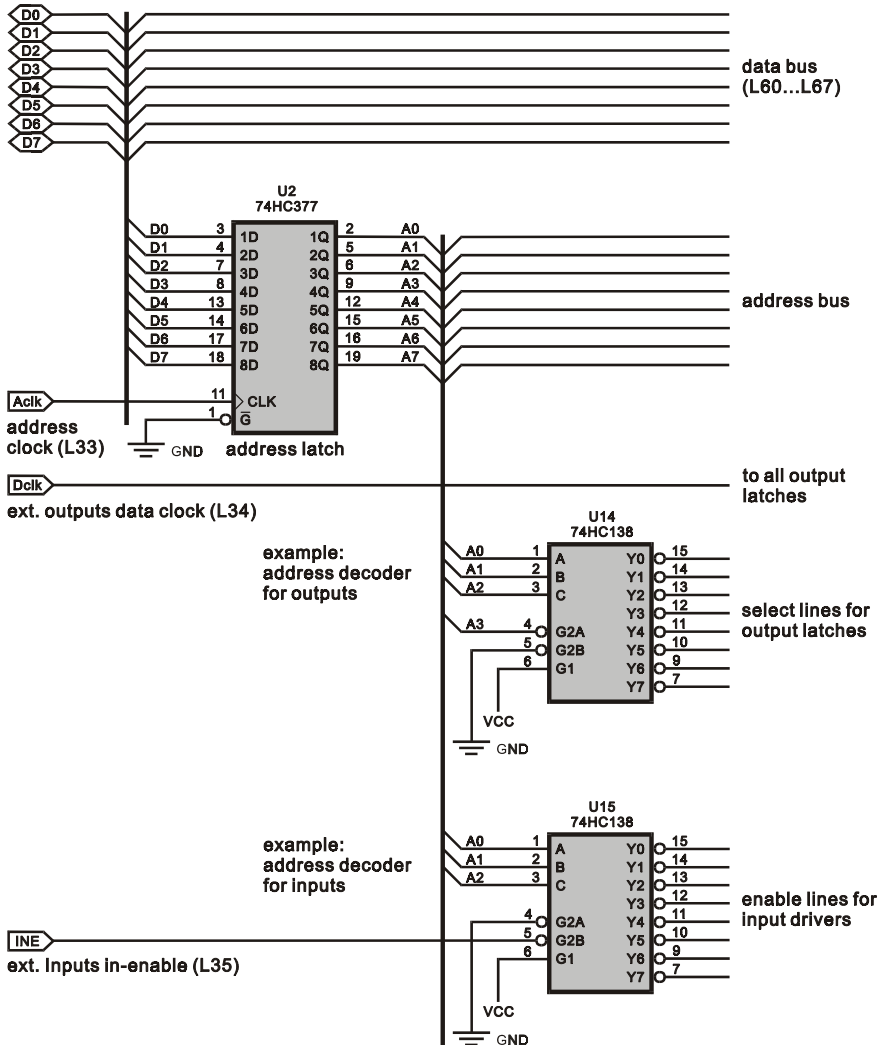
* Da TINY Tiger® den Pin L42 nicht hat, dieser aber vom Device-Treiber LCD1 zur Tonsteuerung verwendet wird, muß die Installation des LCD1-Device-Treibers mit einigen Extra-Parametern durchgeführt werden, um die Ton-Steuerung auf einen anderen Pin zu verlegen.

Alle Leitungen werden von den Device-Treibern automatisch richtig bedient. Zur Benutzung der erweiterten I/O-Pins ist kein Device-Treiber erforderlich.

Achtung: wenn Sie eigene erweiterte I/O-Ports auf dem Patchfeld realisieren, müssen Sie die erweiterten Inputs des Tiny-Tiger®-Prototyping-Boards (Tastatur) deaktivieren. Legen Sie dazu high-Pegel auf den Pin ,keyb' auf J13.

Entwicklungsumgebung

Die Abbildung zeigt die Grundlage der erweiterten I/Os, LCD- und Drucker-Anschluß und Tastatur. Die Adreßbus geht von einem 8-Bit-Speicher aus, in den die Adresse mit dem 'Aclk'-Signal eingeschrieben wird.



BASIC-Tiger® Prototyping-Board

Weitere Informationen zum Anschluß und zur Benutzung der Komponenten finden Sie an anderer Stelle des Handbuches. Dort wird auch der Aufbau des Prototyping-Boards näher erläutert.

Thema	Kapitel	Seite(n)
Erweiterte I/O allgemein	Erweitertes I/O-System	305 ff
Tastatur	Tastatur anschließen	323
Erweiterte Outputs	Erweiterung der Ausgänge	310
Erweiterte Inputs	Erweiterung der Eingänge	313

3

Pinbelegung des Tastaturanschlusses

An der 2-reihigen Stiftleiste J2 für die Tastatur können Sie Ihre eigene Tastaturmatrix mit bis zu 8 Reihen und 8 Spalten anschließen. Pin 1 ist neben dem Kontrast-Poti.

Spalte		Reihe	
0		0	
1		1	
2		2	
3		3	
4		4	
5		5	
6		6	
7		7	

Für die Funktion der Tastatur sind die Jumper auf J22 erforderlich.

Pinbelegung des LC-Display-Anschlusses

Zum Anschluß eines LC-Displays steht eine 1-reihige 14-polige Stiftleiste zur Verfügung. Pin 1 ist neben dem PC-Mode-Schalter.

	GND
	Vcc
	Kontrast
	RS
	R/W (GND)
	E
	D0
	D1
	D2
	D3
	D4
	D5
	D6
	D7

3

Für die Funktion des LC-Displays sind die Jumper auf J22 erforderlich.

Pinbelegung J23

Der 2-reihige Steckverbinder J23 links neben dem Modul-Sockel ist 1:1 mit den Pins des Tiger-Moduls verbunden. Hier können z.B. eigene Aufbauten auf Lochraster oder ein selbst angefertigtes Centronics-Kabel aufgesteckt werden.

reserved	1	2	VCC
L60	3	4	Battery
L61	5	6	AGND
L62	7	8	Vref
L63	9	10	Analog-3
L64	11	12	Analog-2
L65	13	14	Analog-1
L66	15	16	Analog-0
L67	17	18	Alarm
L70	19	20	L41 / PC
L71	21	22	L40
L72	23	24	L42
L73	25	26	L37
L80	27	28	L36
L81	29	30	L35
L82	31	32	L34
L83	33	34	L33
L84	35	36	L95 / RTS0
L85	37	38	L94 / RxD1
L86	39	40	L93 / TxD1
L87	41	42	L92 / CTS0
Reset	43	44	L91 / RxD0
GND	45	46	L90 / TxD0

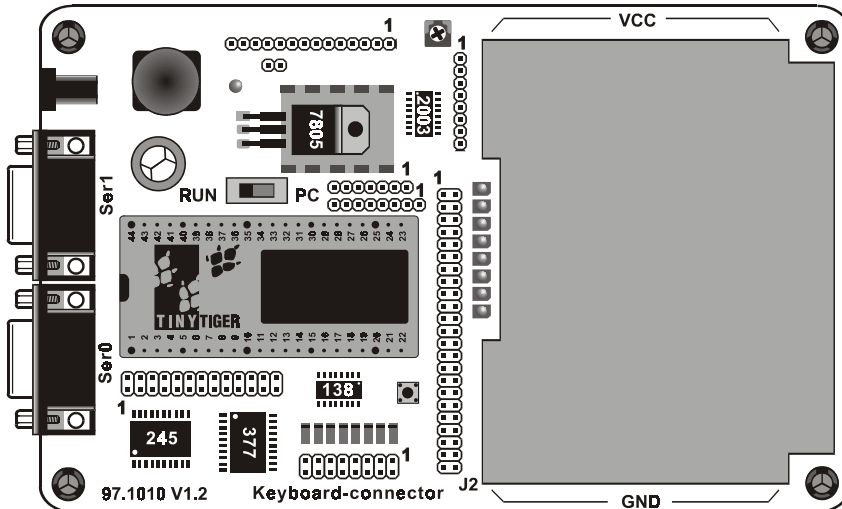
TINY Tiger® Prototyping board

Viele Anwendungen können ohne Erstellung einer eigenen Platine direkt mit dem TINY Tiger®-Prototyping-Board verwirklicht werden. Das Board hat die Standardgröße 100x160mm und paßt in Euro-Gehäuse oder 19“-Racks. Ebenso kann das Board einfach offen als Entwicklungsboard verwendet werden. Das Tiny-Tiger®-Prototyping-Board enthält:

- direkte PC-Verbindung über einen DB9-RS-232-Port.
- zweite serielle RS-232-Schnittstelle steht an einem DB9 zur Verfügung.
- 16-Pin Tastatur-Anschluß für Tastaturen mit bis zu 64 Tasten oder DIP-Schaltern
- Reset-Knopf
- 14-pin Steckverbinder für LC-Display
- 7 Treibertransistoren bis 0.5A
- 8 Allzweck LEDs, mit Treiber-IC
- großes Patchfeld
- Erweiterungsverbinder

3

Eine kurze Übersicht soll Ihnen zunächst das TINY Tiger®-Prototyping-Board näher bringen:



Das Bild zeigt den Steckplatz für das TINY-Tiger® Modul A, den Anschluß für den PC (serielle Schnittstelle 1) sowie den PC-Mode-Schalter als wichtigste Elemente für den Einstieg.

Entwicklungsumgebung

Auf den Steckpins rechts des Tigermoduls sind alle Anschlußpins des Moduls zugänglich.

Generell ist Vorsicht geboten, wenn Verbindungen zu I/O-Pins hergestellt werden: Pins, die Outputs sind, dürfen nicht mit VCC oder GND kurzgeschlossen werden oder gar zu hohe Spannung erhalten.

An einigen Pins ist bereits auf dem TINY-Tiger[®] Prototyping-Board etwas angeschlossen:

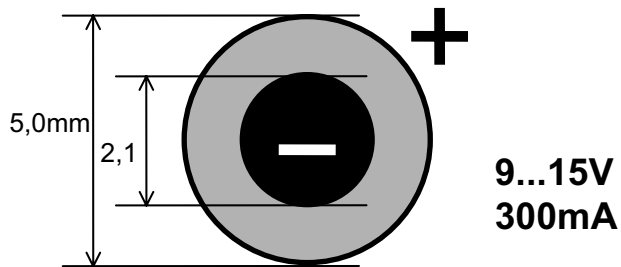
3

- **GND** ist mit der Masse des TINY-Tiger[®] Prototyping-Boards verbunden.
- **AGND** ist mit der analogen Masse des TINY-Tiger[®] Prototyping-Boards verbunden.
- **VCC** ist über den „Tiger-VCC“-Jumper mit VCC des TINY-Tiger[®] Prototyping-Boards verbunden.
- **Reset** kann über den Taster auf GND getastet werden.
- **L90...L95** siehe ‘serielle Anschlüsse weiter unten im Text.

Spannungsversorgung

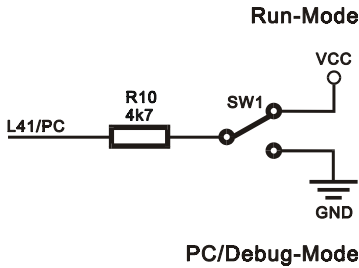
Das TINY-Tiger[®] Prototyping-Board wird mit einem 5W-Netzteil mit 9...15V DC versorgt. Minus liegt auf dem Stift der Stromversorgungsbuchse. Die Stromaufnahme liegt je nach Belastung bei ca. 0,3A.

3



PC-Mode

Der **PC-Mode-Pin** ist auf dem TINY-Tiger[®] Prototyping-Board über 4k7 an den PC-Mode-Schalter gelegt.

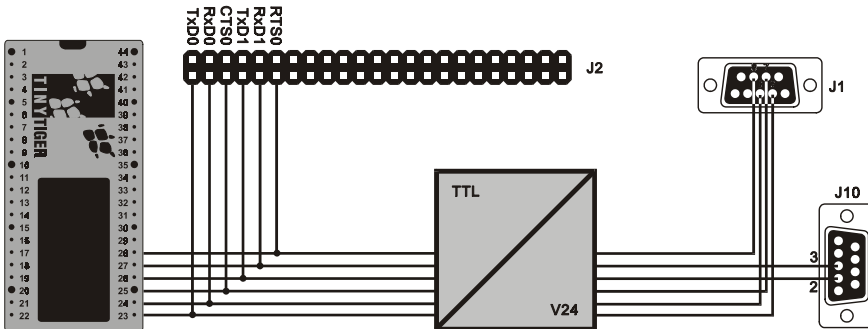


Nach einem Reset oder Power-down testet das TINY-Tiger[®]-Modul, ob der PC-Mode-Pin auf 'low' liegt. Ist dies der Fall, geht das Modul in den PC-Mode/Debug-Mode. Liegt der Pin auf 'high', geht das Modul in den RUN-Mode. Die Zeit zwischen Power-on und erster Aktivität an den I/O-Pins beträgt ca. 220...230msec.

Um das TINY-Tiger[®]-Modul in den PC-Mode zu versetzen, ist ein Reset oder Power-down erforderlich. Es reicht nicht, während des RUN-Mode den Schiebeschalter in die Stellung PC-Mode zu bringen. !

Serielle Anschlüsse

Das TINY-Tiger® Prototyping-Board enthält die RS-232-Treiber. Bitte beachten Sie die Besonderheiten dieser Pins.



3

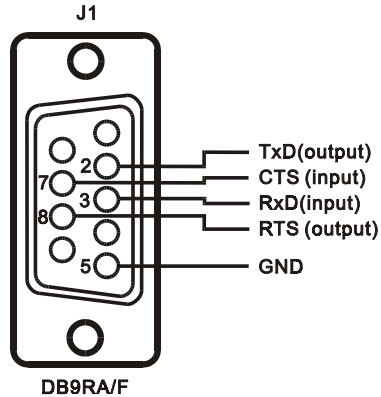
Die **RS-232-Pins L90...L95** werden auf dem Tiny-Tiger®-Prototyping-Board über RS-232-Treiber mit den beiden DB9-Buchsen für 'Ser1' und 'Ser0' verbunden.

Auf der **Stiftleiste J2** sind die Pins des TINY-Tiger®-Moduls zugänglich. Auf den Pins RxD0, CTS0 und RxD1 liegen jedoch auch die Ausgänge der Treiber für die seriellen Eingänge. Dies ist zu beachten, wenn Sie dort normale Outputs planen.

Bitte ziehen Sie das mitgelieferte Schaltbild hinzu, wenn Unklarheiten über die Anschlußgegebenheiten bestehen.

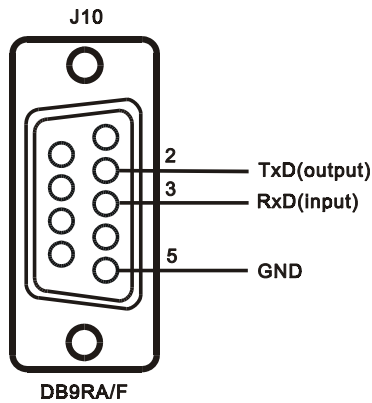
Pinbelegung DB9 'ser 0'

Die DB9-Buchse 'ser0' ist direkt an die Treiber des seriellen Kanals 0 des TI-NY Tiger[®]-Moduls angeschlossen.

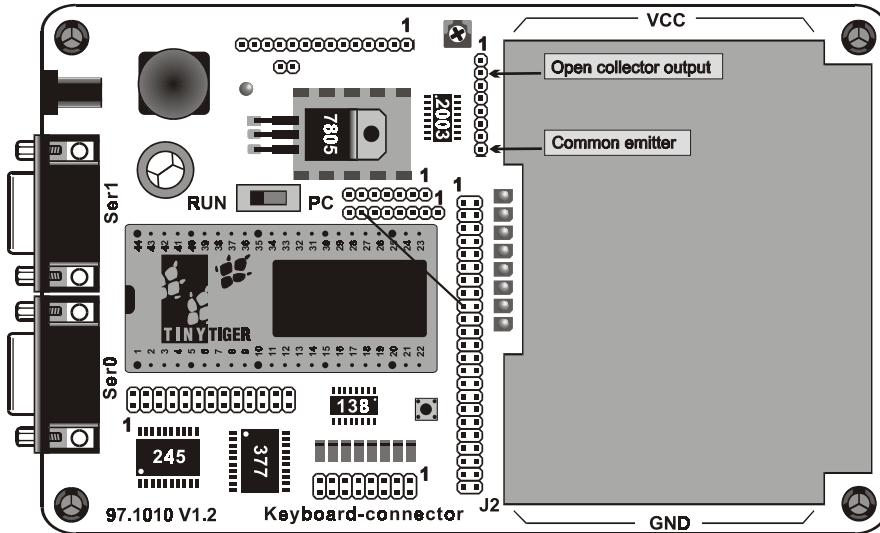


Pinbelegung DB9 'ser 1'

Die DB9-Buchse 'PC/ser1' ist direkt an die Treiber des seriellen Kanals 1 des TINY-Tiger®-Moduls angeschlossen. Dieser Anschluß wird zum Downloaden und Debuggen mit dem PC verbunden.



Treibertransistors (NPN)

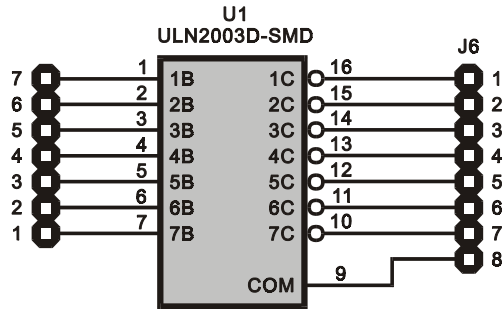


Die Kabelbrücke verbindet den TINY-Tiger®-Ausgang L70 mit einem Eingang der Treibertransistoren.

TINY-Tiger® Prototyping-Board

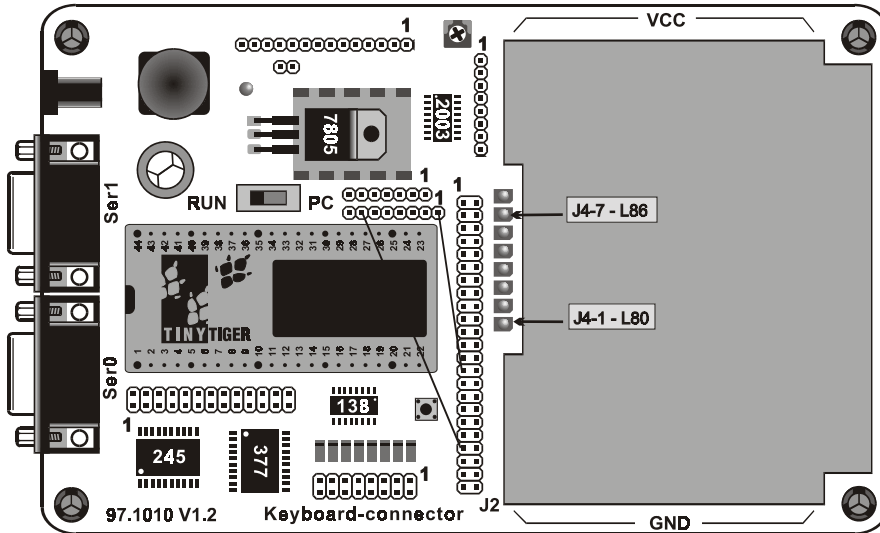
Die 7 NPN Treibertransistoren im Array IC ULN2003 schalten bis zu 0.5 A und haben eine interne Schutzdiode.

Ein beliebiger Ausgangspin des TINY-Tiger®-Moduls wird mit einem Steckpin eines Treibertransistoreingangs verbunden. Ein 'high'-Pegel schaltet den Transistor durch. Beachten Sie, daß der Masse-Anschluß zu der externen Schaltung hergestellt werden muß.



3

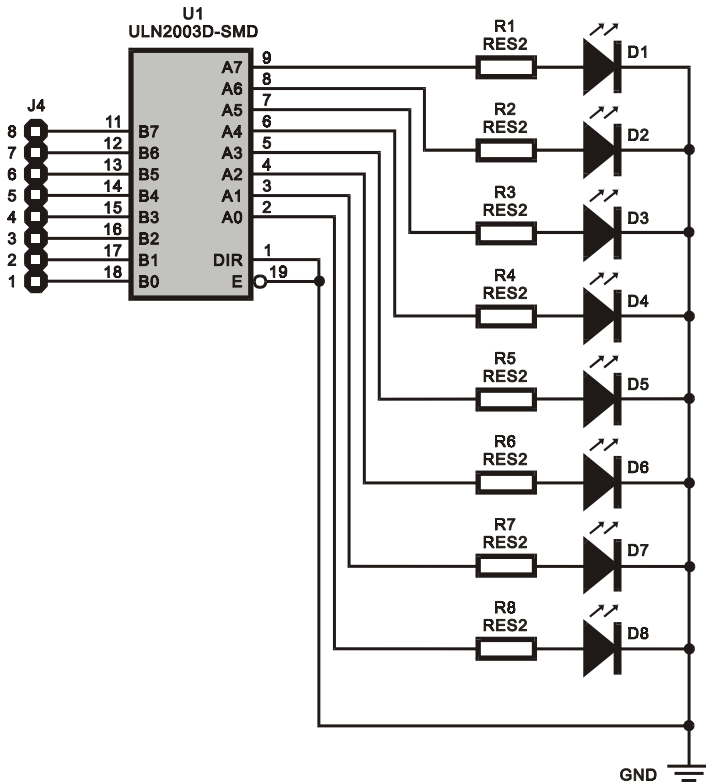
LED status display



Die Kabelbrücke verbindet die Ausgänge L80 und L86 des TINY Tiger[®]-Moduls mit zwei Eingängen der LED-Anzeige.

Die LED-Statusanzeige treibt 8 LED. Wenn der Treiber-Eingang TTL high ist, leuchtet die LED.

Ein beliebiger Ausgangspin des TINY Tiger®-Moduls ist mit der Stiftleiste (J4) der LED-Treiber verbunden.



13 Jumper:
L60...L67, ACLK, DCLK, INE
+ for LCD E, RS

Die Jumper auf J22 verbinden die TINY Tiger[®]-Portpins mit dem Bus-System für LC-Display, und Tastaturanschluß des Tiny-Tiger[®]-Prototyping-Boards.

Das **LC-Display** und die **Tastatur** benutzen gemeinsam einen Datenbus und haben jeweils eigene Steuerleitungen zur Aktivierung. Alle Leitungen sind über 13 Jumper auf die passenden Anschlüsse des TINY-Tiger® steckbar. Die **zusätzlichen Output-Pins** sind in Form eines Treibers realisiert, dessen Eingänge nach Bedarf gesteckt werden (nicht erweiterter I/O-Port).

Wenn die 13 Jumper auf J22 stecken, dann bestehen folgende Verbindungen zum TINY-Tiger®-Modul:

Bus-Bezeichnung	BASIC-Tiger®-Pins	Tiny-Tiger Pin-Nr.
D0...D7	L60...L67	1...8
Aclk (Adressclock)	L33	29
keyb (Tastatur enable an INE)	L35	31
E (LCD: enable)	L36	32
RS (LCD: Reg.select)	L37	33

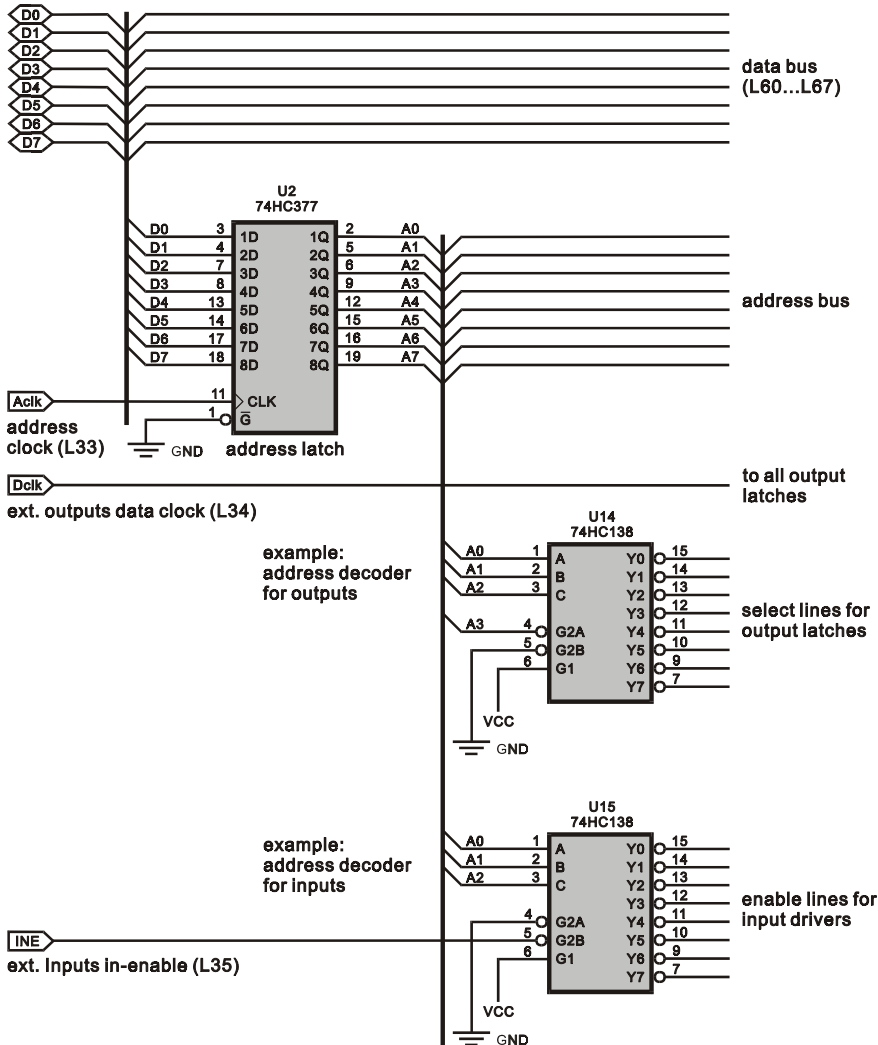
3

Alle Leitungen werden von den Device-Treibern automatisch richtig bedient. Zur Benutzung der erweiterten I/O-Pins ist kein Device-Treiber erforderlich.

Achtung: wenn Sie eigene erweiterte I/O-Ports auf dem Patchfeld realisieren, müssen Sie die erweiterten Inputs des Tiny-Tiger®-Prototyping-Boards (Tastatur) deaktivieren. Legen Sie dazu high-Pegel auf den Pin ‚keyb‘ auf J22.

Entwicklungsumgebung

Die Abbildung zeigt die Grundlage der erweiterten I/Os, LCD- und Drucker-Anschluß und Tastatur. Die Adreßbus geht von einem 8-Bit-Speicher aus, in den die Adresse mit dem 'Aclk'-Signal eingeschrieben wird.



TINY-Tiger® Prototyping-Board

Weitere Informationen zum Anschluß und zur Benutzung der Komponenten finden Sie an anderer Stelle des Handbuches. Dort wird auch der Aufbau des Prototyping-Boards näher erläutert.

Thema	Kapitel	Seite(n)
Erweiterte I/O allgemein	Erweitertes I/O-System	305 ff
Tastatur	Tastatur anschließen	323
Erweiterte Outputs	Erweiterung der Ausgänge	310
Erweiterte Inputs	Erweiterung der Eingänge	313

3

Pinbelegung des Tastaturanschlusses

An der 2-reihigen Stiftleiste J1 für die Tastatur können Sie Ihre eigene Tastaturmatrix mit bis zu 8 Reihen und 8 Spalten anschließen. Pin 1 ist neben dem Kontrast-Poti.

Spalte		Reihe
0		0
1		1
2		2
3		3
4		4
5		5
6		6
7		7

Für die Funktion der Tastatur sind die Jumper auf J22 erforderlich.

Pinbelegung des LC-Display-Anschlusses

Zum Anschluß eines LC-Displays steht eine 1-reihige 14-polige Stiftleiste (J8) zur Verfügung. Pin 1 ist neben dem Kontrast-Poti.

	GND
	Vcc
	Kontrast
	RS
	R/W (GND)
	E
	D0
	D1
	D2
	D3
	D4
	D5
	D6
	D7

3

Für die Funktion des LC-Displays sind die Jumper auf J22 erforderlich.

Pinbelegung J2

Der 2-reihige Steckverbinder J2 an dem Patch-Feld ist mit den Pins des Tiny-Tiger[®]-Moduls verbunden. Hier können z.B. eigene Aufbauten auf Lochraster oder ein selbst angefertigtes Centronics-Kabel aufgesteckt werden. Da der Steckverbinder die gleiche Belegung, wie der entsprechende Steckverbinder J23 des Plug & Play Labs hat, ist eine 1:1-Kopplung mit diesem möglich ist. Es kann jedoch nur ein Modul in einem der beiden Boards stecken.

3

reserved	1		2	VCC
L60	3		4	Battery
L61	5		6	AGND
L62	7		8	Vref
L63	9		10	Analog-3
L64	11		12	Analog-2
L65	13		14	Analog-1
L66	15		16	Analog-0
L67	17		18	Alarm
L70	19		20	L41 / PC
L71	21		22	
L72	23		24	
L73	25		26	L37
L80	27		28	L36
L81	29		30	L35
L82	31		32	L34
L83	33		34	L33
L84	35		36	L95 / RTS0
L85	37		38	L94 / RxD1
L86	39		40	L93 / TxD1
L87	41		42	L92 / CTS0
Reset	43		44	L91 / RxD0
GND	45		46	L90 / TxD0

Tiger-Terminal

Das Tiger-Terminal erweitert das BASIC-Tiger®-Prototyping-Board, das Tiny-Tiger®-Prototyping-Board oder eigene Aufbauten um Tastatur, LC-Display sowie 8 erweiterten Outputs.

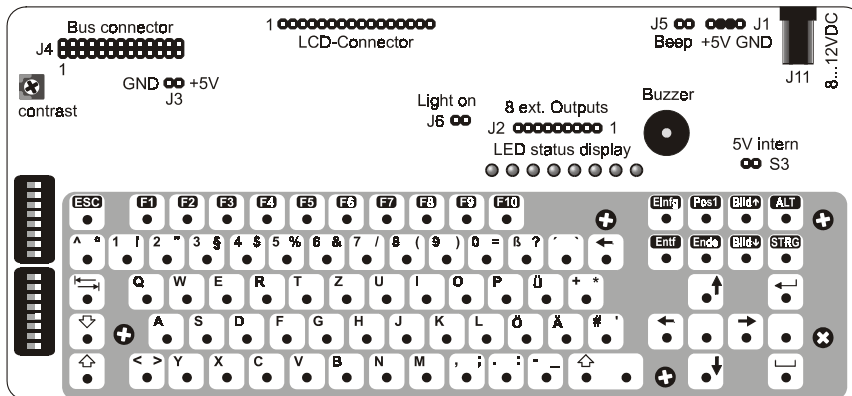


Das Tiger-Terminal versteht sich als Versuchsaufbau (den Sie nicht selbst aufbauen mußten) und darf nur von fachlich geschultem Personal in Betrieb genommen werden.

3

Auf dem Tiger-Terminal befindet sich:

- Tastatur mit 81 Tasten
- 2 x 8 Dipschalter
- LC-Display 4 x 20 Zeichen
- Summer zur Tonausgabe
- 8 Extended Output-Pins
- Statusanzeige der 8 Output-Pins
- Anschluß für Tiger-Datenbus und Steuerleitungen
- Netzteil zur eigenen Stromversorgung mit 8...12V DC



Inbetriebnahme

Das Tiger-Terminal ist eine Zusatz-Hardware zu BASIC-Tiger[®] - oder Tiny-Tiger[®] - Modulen. Es wird vorausgesetzt, daß Sie mit BASIC-Tiger[®] vertraut sind und im Besitz des Original-Handbuches der Entwicklungsumgebung sind.

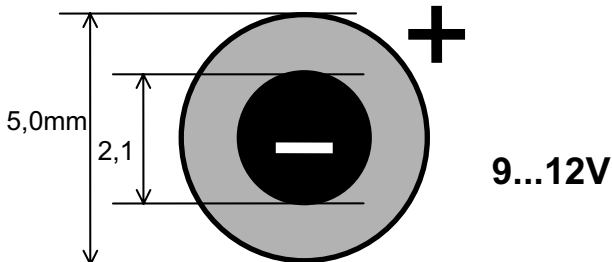
Im Wesentlichen besteht das Tiger-Terminal aus den Bestandteilen Tastatur, LC-Display und einem erweiterten Output-Port, so wie sie im Handbuch der Entwicklungsumgebung unter 'Plug & Play Lab' beschrieben sind.

Die erweiterten Ports sind detailliert im Kapitel 5 unter 'Erweitertes I/O-System' beschrieben.

Die Stromversorgung des Tiger-Terminals kann erfolgen

- unstabilisiert mit 8...12V DC an dem Anschluß für das Steckernetzteil bei gestecktem Jumper auf S3
- stabilisiert mit 5V an dem Anschluß J1 bei offenem Jumper S3

Die Belegung und Maße des Steckers zur Stromversorgung:



Der Anschluß J4 des Tiger-Terminals wird über ein Flachkabel direkt mit dem entsprechenden Bus-Anschluß des BASIC-Tiger[®] - oder Tiny-Tiger[®]-Prototyping-Boards verbunden.

An J5 wird der Ton-Steuerungspin des Moduls angeschlossen. Der Device-Treiber LCD1.TDD verwendet zum Beispiel auf BASIC-Tiger[®]-Modulen den Pin L42 als Ton-Pin für Tastenklick, Fehler-Beep, etc.

Bei gestecktem Jumper J6 ist die Hintergrundbeluchtung des LC-Displays eingeschaltet (sofern vorhanden).

Die Anpassung des Device-Treibers LCD1.TDD an das Terminal geschieht in gleicher Weise, wie dies im Handbuch der Entwicklungsumgebung im Kapitel 8 (Applikationen) unter 'Plug & Play Lab Tastaturanpassung' beschrieben ist.

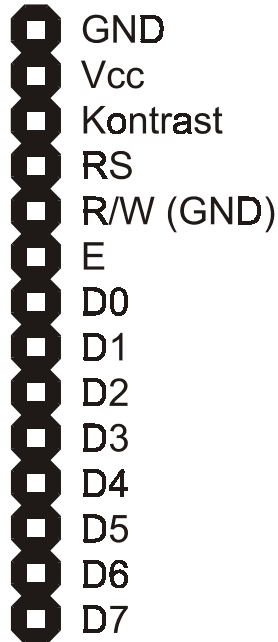
Bus-Steckverbinder für LCD, Tastatur und 8 Outputs

Über ein Flachbandkabel wird das Tigerterminal mit dem Plug & Play Lab oder dem Tiny-Tiger[®]-Prototyping-Board verbunden.

Pin	Pin	Bus-Bezeichnung	BASIC-Tiger [®] -Pins	Modultyp-A Pin-Nr.	Tiny-Tiger Pin-Nr.
1	2	D0	L60	2	1
3	4	D1	L61	3	2
5	6	D2	L62	4	3
7	8	D3	L63	5	4
9	10	D4	L64	6	5
11	12	D5	L65	7	6
13	14	D6	L66	8	7
15	16	D7	L67	9	8
17	18	Aclk (Adressclock)	L33	30	29
19	20	Dclk (Datenclock)	L34	31	30
21	22	ine	L35	32	31
23	24	E (LCD: enable)	L36	33	32
25	26	RS (LCD: Reg.select)	L37	34	33

Pinbelegung des LC-Display-Anschlusses

Das LC-Display ist an einer 1-reihigen 14-poligen Stiftleiste angeschlossen.



3

Der Kontrastregler des LC-Displays befindet sich ganz links in der Nähe des Bus-Steckverbinders.

Erweiterte Outputs

Die 8 erweiterten Outputs werden standardmäßig über die logische Adresse 10h angesprochen. Bit 0 befindet sich rechts auf der Stiftleiste. Die Shift-LED der Tastatur ist an Bit 0 der erweiterten Ports angeschlossen.

Technische Daten des Tiger-Terminals:

Außenmaße / Gewicht:	PCB: 220 x 101 mm mit LCD: ca. 220 x 154 x 64,5 mm / ca. 310g
Stromversorgung	8V...12V DC, ca. 100mA oder 4,7V...5,5V, ca. 100mA mit LC-Display-Backlight: 150mA
Belastbarkeit der erw. I/O-Pins	alle Pins 5mA
Temperaturbereich	0...50°C
Anzahl Tasten	79
Anzahl DIP-Schalter	2 x 8
LC-Display	4 Zeilen à 20 Zeichen (andere Typen sind anschließbar)

Programmier-Adapter

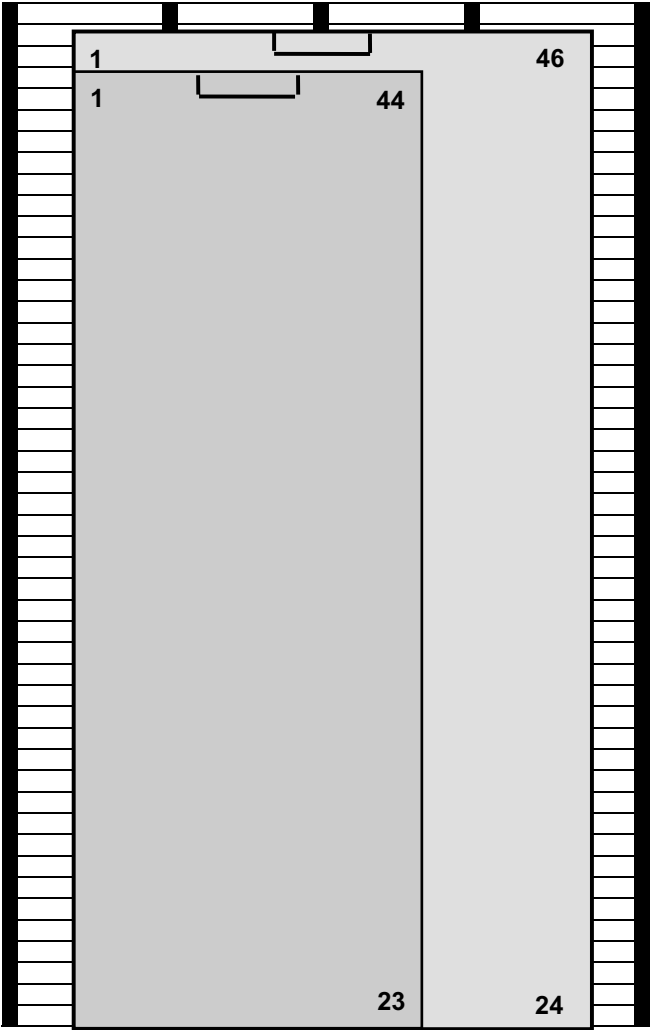
Verwenden Sie den Tiger-ProgrammierAdapter zum Programmieren von BASIC Tiger[®] - oder TINY Tiger[®] -Modulen in Serie.

- Verbinden Sie den Programmier-Adapter über das serielle Kabel mit einem COM-Port am PC.
- Starten Sie das Tiger-BASIC[®]-Entwicklungssystem auf Ihrem PC. Sollte die Fehlermeldung „Schnittstelle konnte nicht geöffnet werden“ erscheinen, ignorieren Sie zunächst diese Fehlermeldung und bestätigen mit OK.
- Wählen Sie aus dem Menü **Optionen** den Befehl **Übertragung** und stellen Sie in dem Dialogfenster den COM-Port ein, an dem der Programmier-Adapter angeschlossen ist.
- Stecken Sie das BASIC-Tiger[®] - oder TINY Tiger[®]-Modul in den Sockel des Programmier-Adapters (immer bei ausgeschaltetem Programmier-Adapter). Der Pin 1 liegt links oben. Stecken Sie das BASIC Tiger[®] - oder TINY Tiger[®]-Modul soweit wie möglich in Richtung zu den Feststellhebeln des ZIF-Sockels in den ZIF-Sockel des Programmier-Adapters.
- Stellen Sie sicher, daß das Steckernetzteil auf 9 Volt eingestellt ist und die Polarität am Stecker 'außen Plus' ist. Bei falscher Polarität funktioniert der Programmier-Adapter nicht, es wird jedoch nichts zerstört.
- Schließen Sie den Programmier-Adapter an das Steckernetzteil an.
- Wählen Sie aus dem Menü **Optionen** den Befehl **Übertragung** und stellen Sie in dem Dialogfenster den COM-Port ein, an dem der Programmier-Adapter angeschlossen ist. Wählen Sie den Befehl **Programm Laden** aus dem Menü **Start**, um das Modul zu programmieren.
- Wenn das Modul nun nicht programmiert werden kann, drücken Sie mit Hilfe eines spitzen Gegenstandes den Reset-Knopf. Normalerweise können die Module jedoch programmiert werden, ohne den Reset-Knopf zu drücken.

Technische Daten:

Maße / Gewicht:	ca. 113 x 89 x 51 mm / ca. 350g
Stromversorgung:	8V...15V / ca. 20mA (ohne Modul)
Anzeige:	Power-LED
Steckverbinder:	DB9-Buchse für PC-COM-Port (1:1)

3



Bevor Sie beginnen	1
Installation	2
Entwicklungsumgebung	3
BASIC-Tiger®-Steuercomputer	4
Häufig gestellte Fragen - Support	5
Stichwortverzeichnis	6
Anhang	7

Leere Seite

4 BASIC-Tiger®-Steuercomputer

BASIC-Tiger® Steuercomputer sind als Modul-Typen unterschiedlicher Leistung und Speichergröße verfügbar. Die Modulreihe A ist z.Zt. mit bis zu 38 eingebauten I/O Leitungen und bis zu 1,5 Mbyte Speicher verfügbar. Module mit mehr Speicher, mehr internen I/O-Pins und zusätzlichen Leistungsmerkmalen sind in Vorbereitung. Programme für BASIC-Tiger® Steuercomputer können auf den unterschiedlichsten Tiger Modul-Typen ablaufen.

Das BASIC-Tiger®-System soll dem Anwender einfach zu programmierende und universell einsetzbare Modul-Computer für ein weites Anwendungsspektrum bieten. Tiger-BASIC® verfügt über leistungsfähige Mechanismen für modulare, strukturierte Programmierung. Ebenso überzeugen Tiger-BASIC Programme durch hohe Ausführungsgeschwindigkeit.

Software-Änderungen können In-Circuit und ohne UV-Löschvorgang sowohl während der Entwicklungsphase als auch später noch vorgenommen werden. Dies kann z.B. über direkte RS-232 Verbindung zu einem PC / Notebook geschehen oder kontaktlos durch IR-Übertragung oder per DFÜ.

Flash-Speicher

Dem Flash-Speicher kommt eine zentrale Bedeutung zu. Im Flash-Speicher werden Tiger-BASIC®-Programme gespeichert und BASIC-Programme können freien Flash als dauerhaften Datenspeicher verwenden. Sie finden hier einige Informationen, die zum näheren Verständnis mancher Instruktionen und Zusammenhänge hilfreich sind.

Im FLASH-Speicher bleiben einmal abgelegte Daten auch dann erhalten, wenn die Versorgungsspannung abgeschaltet wird. Mindestens 10 Jahre. Der FLASH-Speicher kann ohne UV-Licht elektrisch gelöscht und wieder neu programmiert werden. Es sind mindestens 100.000 Löscho-Zyklen möglich.

Der Flash-Speicher kann byteweise beschrieben und gelesen werden. Das Löschen erfolgt Sektorweise. Die Anzahl und Größe der Sektoren hängt vom Modultyp ab. Üblich sind z.Zt. Sektor-Größen von 16 KByte und 64 Kbyte. Über Sektorgröße und Anzahl der freien Sektoren im Modul gibt die Funktion SYSVARN Auskunft.

Sowohl das Schreiben als auch das Löschen nehmen mehr Zeit in Anspruch, als zum Beispiel das Beschreiben des RAM-Speichers. Die Zugriffs-, Lese- und Schreibgeschwindigkeit ist jedoch hoch im Vergleich zu mechanisch bewegten Speichermedien (Festplatte, Floppydisk, ...). Insbesondere ist auch die Zuverlässigkeit und Robustheit des FLASH Speichers höher, was ein wichtiger Faktor in Steuerungen ist.

Will man neue Daten in einen FLASH-Speicher an einen Ort schreiben, wo bereits Daten geschrieben sind, so ist vorher der betreffende Sektor zu löschen. Ein typischer Zyklus eines Flash-Speicher Sektors könnte wie folgt aussehen:

- Sektor Löschen
- Daten schreiben + lesen
- Sektor Löschen

Ein gelöschter FLASH-Sektor enthält nur Bytes mit den Wert 0FFh (255). Beim Schreiben werden nur Bits mit dem Wert „0“ geschrieben. Bits mit dem Wert „1“ bleiben unverändert. In einem Flash-Speicher können niemals Bits vom Wert „1“ geschrieben werden an Positionen in denen bereits ein „0“-Bit im Speicher vorliegt. Dazu müßte der betreffende Sektor zuerst einmal gelöscht werden (alle Bits = „1“ setzen). Erlaubt beim BASIC-Tiger® hingegen ist es „0“-Bits in Speicherstellen mit „1“-Bits zu schreiben. Beispiel:

4

Eine gelöschte FLASH-Speicherzelle (Inhalt: FFh) kann der Reihe nach bitweise mit zusätzlichen 0-Bits beschrieben werden ohne, daß der zugehörige Sektor zwischen- durch gelöscht werden müßte, z.B.:

FFh, 7Fh, 3Fh, 1Fh, Fh, 7h, 3H, 1h, 0h

Dies kann z.B.: in Anwendungen mit Betriebsstundenzähler oder Gebührenzähler verwendet werden. Wenn z.B. alle 10 Sekunden 1 Bit auf 0 gesetzt wird in einem Sektor von 16 Kbyte, so können in diesem Sektor 1 310 720 Sekunden = ca. 364h = ca. 15 Tage gezählt werden. Ist der Sektor voll, kann z.B. ein Übertrag in den nächsten Sektor gebildet werden und der Low-Order-Sektor wird wieder gelöscht.

Der Versuch, ein Bit, das im Flash den Wert „0“ hat mit einer „1“ zu beschreiben, schlägt fehl.

Diese Eigenschaft des FLASH-Speichers gibt auch die Strategie vor, wenn man den FLASH-Speicher als „Pseudo“-RAM verwenden will - z.B. als Ersatz für einen Disk-Speicher mit Filesystem oder als Speicher für gelegentlich zu ändernde Informationen.

Im einfachsten Fall hat man so viele Sektoren frei, wie man Tabellen oder Files speichern möchte, und die Zahl der Schreibvorgänge ist nicht sehr hoch. Dann kann man ganz einfach in den gewünschten FLASH-Sektor schreiben und ihn löschen, wenn Daten durch neue Daten ersetzt werden sollen. Die Anzahl von bis zu 100.000 Löscho-zyklen pro Sektor stellt eine natürliche Grenze dar. Beispiel: bei max. 10 Löscho-zyklen täglich, wird diese Grenze erst nach ca. 30 Jahren erreicht.

Wenn eine Anwendung wesentlich öfter alte Daten durch neue Daten ersetzen will, so macht man von der Eigenschaft Gebrauch, daß sich jedes beliebige Bitmuster (Byte) im FLASH durch den Wert 00h überschreiben läßt. So können Flags und Pointer die mit Daten-Records verknüpft sind, mit einer „Ungültig“ Kennung überschrieben werden. Wenn man also in einem Sektor Platz hat den betreffenden Datensatz 100-mal zu schreiben und auch noch eine Flag-Tabelle darin unterbringen kann,

so kann der Datensatz genau 100 mal neu in den FLASH-Sektor geschrieben werden, bevor der Sektor gelöscht werden muß. Man hat also die Zahl der möglichen Schreib-Lösch Zyklen bereits verhundertfacht.

Will man den Schreib- und Löschvorgang auch noch sichern gegen plötzlichen Power-Down, der zu einer verstümmelten Information führen kann, so wird z.B. mit 2 Sektoren gearbeitet, die in mehreren Stufen nacheinander beschrieben bzw. gelöscht werden.



Während des Löschens mit der Instruktion ERASE_FLASH steht der Flash-Speicher vorübergehend nicht zur Verfügung. In Modulen, deren Datensektoren im gleichen Flash-Chip liegen wie das Programm, führt dies dazu, daß kurzzeitig alle Tasks im Modul zum Stillstand kommen (ca. 0.5 - mehrere Sekunden):

alle Tasks stehen

die I/O-Aktivität frieren ein

die System-Ticks zählen nicht weiter

das stellen der Uhr wird gestört (das-Stellen der Uhr dauert einige Sekunden)

und die Device-Treiber stellen Ihre Aktivitäten ein

Alle zeitkritischen Vorgänge müssen also vor Löschen eines oder mehrerer Flashsektoren in einen kontrollierten Zustand gebracht werden.

Wenn es FLASH-Sektoren gibt, die nicht vom Programm-Code belegt werden, stehen diese einem Tiger-BASIC® Programm als Daten-Speicher frei zur Verfügung. Der benutzbare Flash-Speicher beginnt für den Anwender immer bei Adresse 0. Es stehen immer komplette Sektoren zur Verfügung, die byteweise beschrieben und gelesen werden können. Das Löschen erfolgt stets Sektorweise.

Wichtige Informationen über den Flash-Speicher des Moduls lassen sich in Form von Systemparametern abfragen. (siehe Funktion SYSVARN).

BASIC-Tiger®-Steuercomputer

Leere Seite

4

Modulreihe A

Die Modelle der Modulreihe A unterscheiden sich durch die Größe des Flash- und des RAM-Speichers sowie durch das eventuelle Vorhandensein der RS-232-Treiber und der Echtzeit-Uhr.

Folgende technische Merkmale treffen auf alle Modelle der Modulreihe A zu:

Außenmaße / Gewicht:	ca. 63 x 41 x 12 mm / ca. 50g
Stromversorgung	4,7V...5,5V / ca. 45 mA
Belastbarkeit der I/O-Pins	alle Pins 1,6mA, bzw. max. 8x 3,5mA
Reset	Power-ON-Reset auf dem Modul sowie durch externen Eingang
Batterie f. Uhr + RAM-Pufferung	Anschluß für externe Pufferbatterie (Pin 45) U-batt 3.0...5.0V, I-batt 50µA...150µA
A/D-Eingänge	4 Kanäle (Pin 39 bis Pin 42) Auflösung 8-, 10-Bit Eingangsspannung 0V...Vref (Vref=3.5V...5V, Iref-max=1.5mA) Eingangswiderstand: 1M bei ausgeschaltetem Modul sehr niederohmig!
PWM	2 PWM-Kanäle (Pin 12 und Pin 13) Auflösung 6-, 7-, 8-Bit Wiederholfrequenz 1.2kHz...80kHz

BASIC-Tiger®-Steuercomputer

Technische Daten, die je nach Modell variieren:

RAM	128KB, 256KB, 512KB, 1MB, 2MB
Flash-ROM	128KB, 512KB, 2MB, 4MB
Uhr (optional)	mit Kalender + Weckfunktion (Alarm aktiv low, Pin 38)
2 serielle Kanäle	alternativ TTL-Pegel oder RS-232-Treiber on-board (Pin 24...Pin 29) Pins 24...29 können auch durch Einstellen per Software als digitale I/O verwendet werden, wenn die RS-232-Treiber nicht bestückt sind.

4

EMV-Hinweise: Obwohl bereits in dem Modul einiges für die ‘elektromagnetische Verträglichkeit’ und gegen Störausstrahlung getan wurde (Multi-Layerplatine mit VCC- und GND-Layer, durch Innen-Bedampfung abgeschirmtes Gehäuse, VCC-Ferrit), müssen weitere Maßnahmen getroffen werden, wenn ein CE-Zeichen für das Endgerät angestrebt wird. Das Modul betreffend werden folgende Maßnahmen vorgeschlagen:

- möglichst dicht am VCC-Pin des Moduls einen Tantal- sowie einen keramischen Kondensator vorsehen (z.B. 47µF + 100nF).
- VCC über eine Drossel mit VCC verbinden (z.B. 68nH).
- Alle Ein- und Ausgänge nach Möglichkeit über 1nF oder entsprechende Varistoren mit ähnlicher Kapazität mit Masse verbinden.
- Nicht verwendete I/O-Pins per Software als Eingang initialisieren.

Es gibt kein Patentrezept, wie ein Projekt störfest und abstrahlungsarm wird. Neben den obigen Vorschlägen spielt immer auch die zusätzlich vorhandene Peripherie und Beschaltung eine Rolle.

BASIC Tiger® Modul A RESET-In

BASIC-Tiger® Modul A verfügt über einen RESET-Eingang sowie einen internen RESET-Generator. Der RESET wird üblicherweise für verschiedene Aufgaben eingesetzt:

- definierte Startbedingungen des Computers beim Power-Up herstellen
- Synchronisation mehrerer Computer und logischer Einheiten beim Hochfahren
- Manueller Abbruch, „Paniktaste“ (Warmstart)
- Zusätzliche Sicherheitsfunktion mit zusätzlichem externen Watchdog.

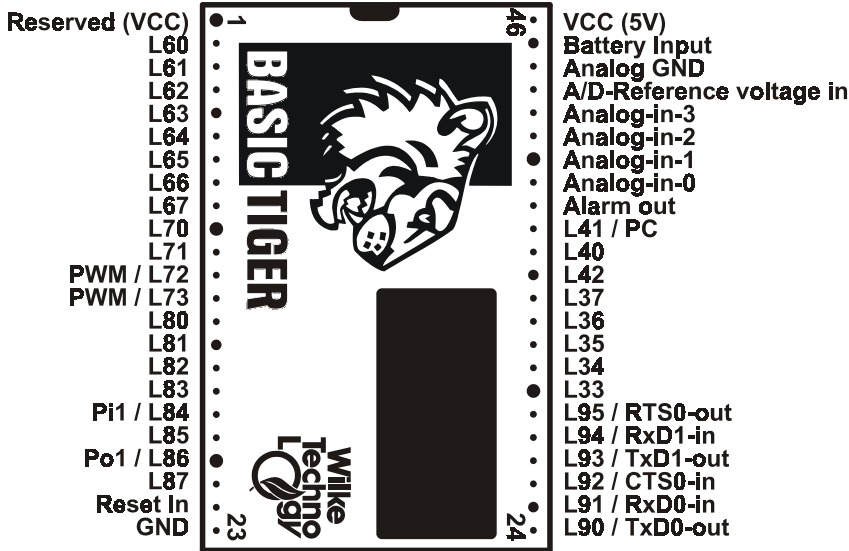
Der Reset-Eingang des BASIC-Tiger® A ist ein digitaler Eingang, der mit einem internen Pull-up von 10k versehen ist und kann offen gelassen werden, wenn das

Design keinen Reset-Eingang benötigt. Um einen Reset auszulösen, braucht der Eingang lediglich mindestens 0,4µsec gegen GND gezogen zu werden. Dieser Eingang ist nicht identisch mit dem modul-internen Reset-Netz, vielmehr wird der Reset dadurch lediglich ausgelöst, intern jedoch mindestens auf eine notwendige Zeitlänge gestreckt. Ein Reset wird ebenso ausgelöst, sobald VCC die Schwellenspannung 4,6V unterschreitet. Auch in diesem Fall ist die Länge des Reset-Impulses durch den internen Kondensator bestimmt und reicht auch bei kurzen Spannungseinbrüchen aus. Es werden Spannungseinbrüche erfasst, die 2µsec oder länger sind.

In größeren Systemen, die aus mehreren Computern und Logikeinheiten bestehen, sollte man stets einen zentralen RESET generieren und an alle Einheiten weiterleiten. Stromversorgungen geben zu diesem Zweck ein „Power-good“ Signal ab, das erst dann aktiv wird (high), wenn sich alle Ausgangsspannungen stabil aufgebaut haben.

BASIC-Tiger®-Steuercomputer

BASIC-Tiger®-Modul A Pinbelegung



Die Pins PWM, Pi1, Po1 sowie die seriellen Pins sind digitale I/O-Pins, wenn der zugehörige Device-Treiber in der Anwendung nicht eingebunden ist.

Pin-Beschreibung

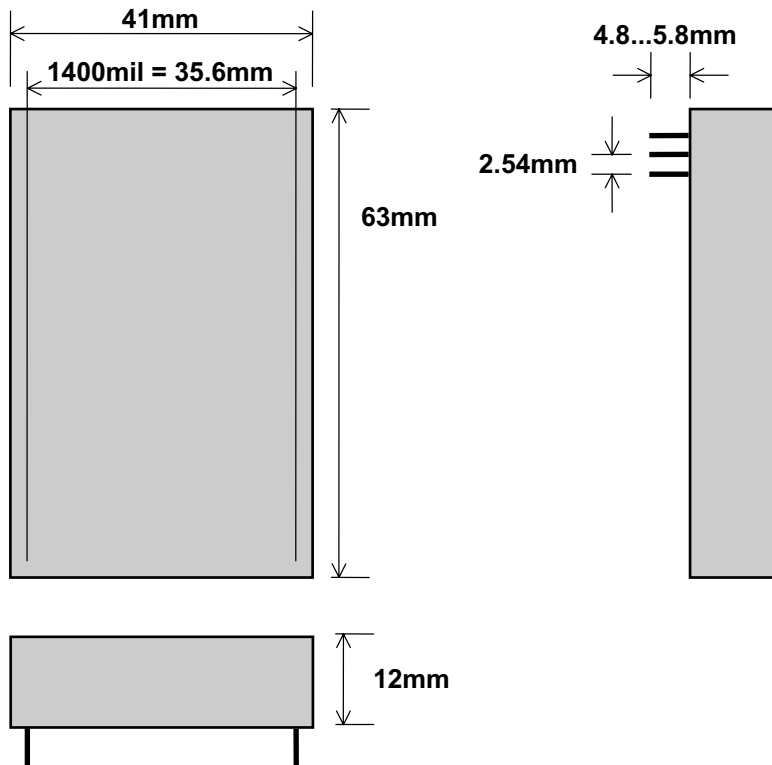
Pin-Nr.	Bezeichnung	Bedeutung
1		reserviert
2	L60	Port 6 - Pin 0
3	L61	Port 6 - Pin 1
4	L62	Port 6 - Pin 2
5	L63	Port 6 - Pin 3
6	L64	Port 6 - Pin 4
7	L65	Port 6 - Pin 5
8	L66	Port 6 - Pin 6
9	L67	Port 6 - Pin 7
10	L70	Port 7 - Pin 0
11	L71	Port 7 - Pin 1
12	L72 / PWM	Port 7 - Pin 2 oder PWM-Ausgang
13	L73 / PWM	Port 7 - Pin 3 oder PWM-Ausgang
14	L80	Port 8 - Pin 0
15	L81	Port 8 - Pin 1
16	L82	Port 8 - Pin 2
17	L83	Port 8 - Pin 3
18	L84	Port 8 - Pin 4
19	L85	Port 8 - Pin 5
20	L86	Port 8 - Pin 6
21	L87	Port 8 - Pin 7
22	Reset in	Reset-Eingang (Taster gegen GND)
23	GND	Masse

BASIC-Tiger®-Steuercomputer

4

Pin-Nr.	Bezeichnung	Bedeutung
24	L90 / TxD0	Port 9 - Pin 0 oder Sendeleitung serieller Port 0
25	L91 / RxD0	Port 9 - Pin 1 oder Empfangsleitung serieller Port 0
26	L92 / CTS0	Port 9 - Pin 2 oder CTS0 (Eingang)
27	L93 TxD1	Port 9 - Pin 3 oder Sendeleitung serieller Port 1
28	L94 RxD1	Port 9 - Pin 4 oder Empfangsleitung serieller Port 1
29	L95 RTS0	Port 9 - Pin 5 oder RTS0 (Ausgang)
30	L33	Port 3 - Pin 3
31	L34	Port 3 - Pin 4
32	L35	Port 3 - Pin 5
33	L36	Port 3 - Pin 6
34	L37	Port 3 - Pin 7
35	L42	Port 4 - Pin 2
36	L40	Port 4 - Pin 0
37	L41 / PC	PC-Mode-Pin
38	Alarm out	Alarm Ausgang (mit Uhr, Open-Collector; sonst NC)
39	Analog In 0	Analogeingang 0 (0V...VCC)
40	Analog In 1	Analogeingang 1 (0V...VCC)
41	Analog In 2	Analogeingang 2 (0V...VCC)
42	Analog In 3	Analogeingang 3 (0V...VCC)
43	A/D-Ref-in	A/D-Referenzspannung.: 3,5...5,0V, 1,5mA max.
44	Analog GND	Analog Masse (intern über Ferrit mit GND verbunden)
45	Batterie in	Anschluß für externe Batterie 3,0...5,0V
46	VCC (5V)	Spannungsversorgung

Abmessungen:



4

BASIC-Tiger®-Steuercomputer

Leere Seite

4

TINY Tiger® Modul

TINY Tiger®-Module sind die kleinsten in TINY BASIC® oder Tiger BASIC® programmierbaren Module.

Module der TINY Tiger®-Serie sind mit 32kByte oder 128kByte RAM sowie mit oder ohne Echtzeituhr erhältlich.

Die Beispiele und Applikationen in diesem Handbuch sind in erster Linie für BASIC-Tiger®-Module erstellt und laufen nicht unbedingt alle auf den Tiny-Tiger®-Modulen mit 32kByte RAM. In der Installationszeile für den Device-Treiber LCD1.TDD müssen zusätzliche Parameter angegeben werden, um den Pin für die Ton-Steuerung auf TINY Tiger®-Modulen zur Verfügung zu stellen.

Der freie RAM-Speicher in einem 32k-TINY Tiger®-Modul beträgt bei geladenem Programm nicht viel mehr als 10kByte. Die folgenden Hinweise sollen Ihnen helfen, ein Maximum an Leistung trotz wenig RAM zu erzielen:

- Die Standard-Stackgröße ist für jede Task 2kByte. Schreiben Sie in Ihren Quellcode die Zeile 'USER_STACK_SIZE nnn', um den Stack jeder Task auf bis zu 512Byte zu verkleinern.
- Die Standard-Stringgröße ist 64 Bytes. Sie sollten die Stringgröße angeben und auf die wirklich benötigte Größe begrenzen. Diese Regel gilt speziell für ARRAYs of STRINGs.
- Binden Sie nur Device-Treiber ein, die Ihre Anwendung wirklich benötigt. Speziell gepufferte Treiber wie LCD1.TDD, SER1.TDD, PRN1.TDD brauchen RAM für ihre Puffer.
- Bei der Erklärung der CALL-Instruktion finden Sie Informationen über die Benutzung des Stacks. Parameter, die an ein Unterprogramm übergeben werden sowie lokale Variable benötigen Stack entsprechend ihrer Größe.
- Während der Entwicklungsphase Ihres Projektes können Sie die Stackauslastung während der Laufzeit mit der Funktion SYSVARn abfragen (Funktionsnummern DSTACK_FILL und DSTACK_FREE).

BASIC-Tiger[®]-Steuercomputer

Die folgenden technischen Daten gelten für alle TINY Tiger[®]-Module:

Maße / Gewicht	ca. 60 x 28 x 10,5 mm / ca. 28g
Stromversorgung	4.7V...5.5V / ca. 45 mA
Belastbarkeit der I/O-Pins	alle Pins 1.6mA, bzw. max. 8x 3.5mA
Reset	Power-ON-Reset auf dem Modul sowie durch externen Eingang
Batterie f. Uhr + RAM-Pufferung	Anschluß für externe Pufferbatterie (Pin 43)
2 serial channels	TTL-level (Pin 23...Pin 28) Pins 23...28 können auch durch Einstellen per Software als digitale I/O verwendet werden.
A/D-Eingänge	4 Kanäle (Pin 39 bis Pin 42) Auflösung 8-, 10-Bit Eingangsspannung 0V...Vref (Vref=3.5V...5V, Iref-max=1.5mA) Eingangswiderstand: 1M bei ausgeschaltetem Modul sehr niederohmig!
PWM	2 PWM-Kanäle (Pin 11 und Pin 12) Auflösung 6, 7, 8-Bit Wiederholfrequenz 1.2kHz...80kHz

Technische Daten, die vom Modultyp abhängen:

Flash-ROM	128KB, 512KB, 2MB
RAM	32kB, 128KB, 512KB
Uhr (optional)	mit Kalender + Weckfunktion (Alarm aktiv low, Pin 34). U-batt 3.0...5.0V, I-batt 50µA...150µA

EMV-Hinweise: siehe unter BASIC-Tiger[®]. Einziger Unterschied: bei Tiny-Tiger[®] ist das Kunststoffgehäuse nicht von innen zur Abschirmung bedampft.

TINY Tiger® RESET-In

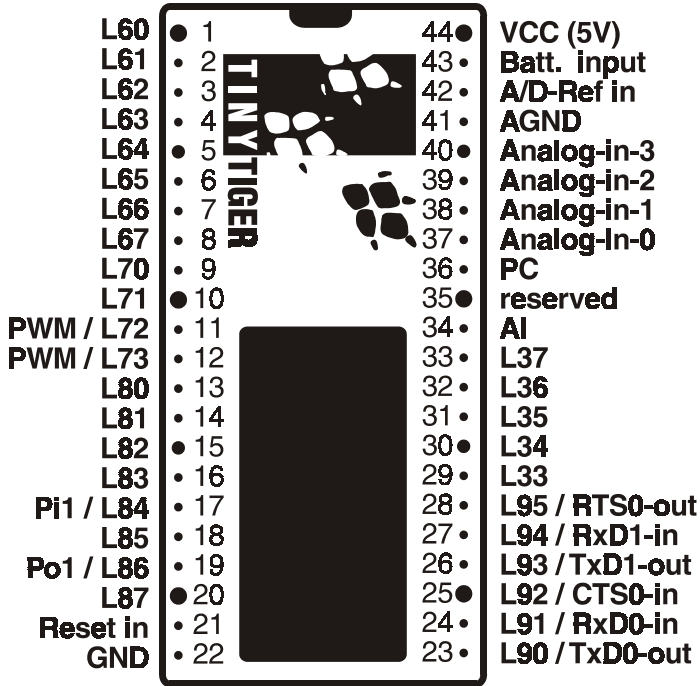
TINY-Tiger® verfügt über einen RESET-Eingang sowie einen internen RESET-Generator. Der RESET wird üblicherweise für verschiedene Aufgaben eingesetzt:

- definierte Startbedingungen des Computers beim Power-Up herstellen
- Synchronisation mehrerer Computer und logischer Einheiten beim Hochfahren
- Manueller Abbruch, „Paniktaste“ (Warmstart)
- Zusätzliche Sicherheitsfunktion mit zusätzlichem externen Watchdog.

Der Reset-Eingang des BASIC-Tiger® A ist ein digitaler Eingang, der mit einem internen Pull-up von 10k versehen ist und kann offen gelassen werden, wenn das Design keinen Reset-Eingang benötigt. Um einen Reset auszulösen, braucht der Eingang lediglich mindestens 0,4µsec gegen GND gezogen zu werden. Dieser Eingang ist nicht identisch mit dem modul-internen Reset-Netz, vielmehr wird der Reset dadurch lediglich ausgelöst, intern jedoch mindestens auf eine notwendige Zeitlänge gestreckt. Ein Reset wird ebenso ausgelöst, sobald VCC die Schwellenspannung 4,6V unterschreitet. Auch in diesem Fall ist die Länge des Reset-Impulses durch den internen Kondensator bestimmt und reicht auch bei kurzen Spannungseinbrüchen aus. Es werden Spannungseinbrüche erfasst, die 2µsec oder länger sind.

In größeren Systemen, die aus mehreren Computern und Logikeinheiten bestehen, sollte man stets einen zentralen RESET generieren und an alle Einheiten weiterleiten. Stromversorgungen geben zu diesem Zweck ein „Power-good“ Signal ab, das erst dann aktiv wird (high), wenn sich alle Ausgangsspannungen stabil aufgebaut haben.

TINY Tiger® Pinbelegung



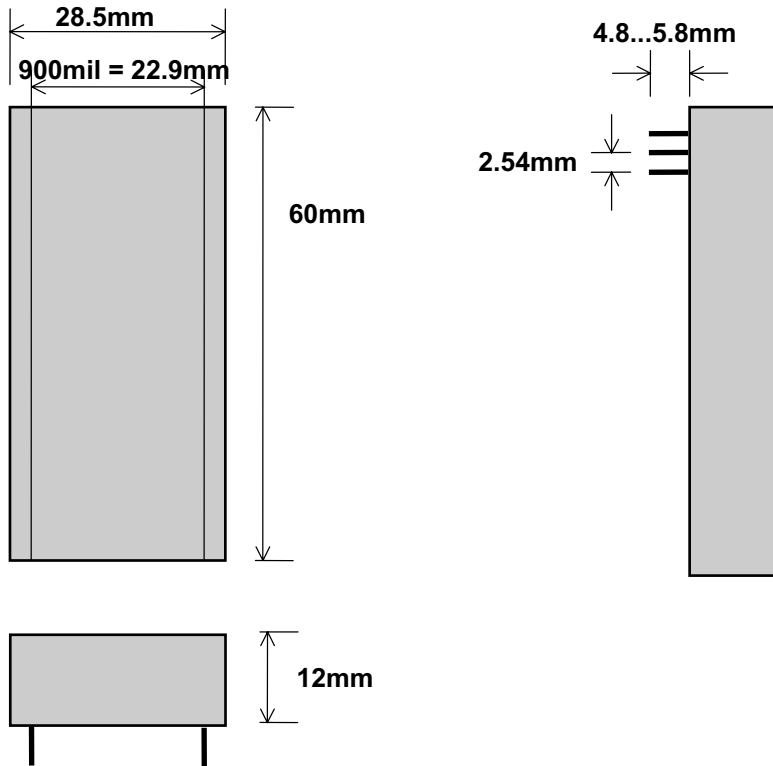
Die Pins PWM, Pi1, Po1 sowie die seriellen Pins sind digitale I/O-Pins, wenn der zugehörige Device-Treiber in der Anwendung nicht eingebunden ist.

Pin-Beschreibung

Pin-No.	Bezeichnung	Bedeutung
1	L60	Port 6 - Pin 0
2	L61	Port 6 - Pin 1
3	L62	Port 6 - Pin 2
4	L63	Port 6 - Pin 3
5	L64	Port 6 - Pin 4
6	L65	Port 6 - Pin 5
7	L66	Port 6 - Pin 6
8	L67	Port 6 - Pin 7
9	L70	Port 7 - Pin 0
10	L71	Port 7 - Pin 1
11	L72 / PWM	Port 7 - Pin 2 or PWM-output
12	L73 / PWM	Port 7 - Pin 3 or PWM-output
13	L80	Port 8 - Pin 0
14	L81	Port 8 - Pin 1
15	L82	Port 8 - Pin 2
16	L83	Port 8 - Pin 3
17	L84	Port 8 - Pin 4
18	L85	Port 8 - Pin 5
19	L86	Port 8 - Pin 6
20	L87	Port 8 - Pin 7
21	Reset in	Reset Eingang (Taster gegen GND)
22	GND	Masse

Pin-No.	Bezeichnung	Bedeutung
23	L90 / TxD0	Port 9 - Pin 0 oder Sendeleitung serieller Port 0
24	L91 / RxD0	Port 9 - Pin 1 oder Empfangsleitung serieller Port
25	L92 / CTS0	Port 9 - Pin 2 oder CTS0 (Eingang)
26	L93 TxD1	Port 9 - Pin 3 oder Sendeleitung serieller Port 1
27	L94 RxD1	Port 9 - Pin 4 oder Empfangsleitung serieller Port 1
28	L95 RTS0	Port 9 - Pin 5 oder RTS0 (Ausgang)
29	L33	Port 3 - Pin 3
30	L34	Port 3 - Pin 4
31	L35	Port 3 - Pin 5
32	L36	Port 3 - Pin 6
33	L37	Port 3 - Pin 7
34	Alarm out	Alarm-Ausgang (mit Uhr, Open-Collector; sonst NC)
35	reserved	mit VCC verbinden
36	L41 / PC	PC-Mode-Pin
37	Analog In 0	Analogeingang 0 (0V...VCC)
38	Analog In 1	Analogeingang 1 (0V...VCC)
39	Analog In 2	Analogeingang 2 (0V...VCC)
40	Analog In 3	Analogeingang 3 (0V...VCC)
41	Analog GND	Analog Masse (intern über Ferrit mit GND verbunden)
42	A/D-Ref-in	A/D-Referenzspannung: 3,5...5,0V
43	Battery in	Anschluß für externe Batterie 3,0...5,0V
44	VCC (5V)	Spannungsversorgung

Abmessungen:



4

BASIC-Tiger®-Steuercomputer

Leere Seite

4

BASIC Tiger® Modul E

Die BASIC Tiger®-Module E sind die kleinsten Module, die mit der Tiger BASIC®-Software programmiert werden.

Module der Serie BASIC Tiger® E unterscheiden sich von den T-Modulen (Tiny-Tiger®) durch die Doppelnutzung einiger Pins. Einige Pins wurden eingespart, um das Modul klein zu machen: es gibt keinen BATT-Pin, die PWM-Pins fehlen, Vref ist intern mit VCC verbunden und eine interne Uhr ist nicht erhältlich. Abgesehen von diesen Hardware-Unterschieden ist das Modul kompatibel zu den anderen Tiger-Modulen.

Die Beispiele und Applikationen in den Handbüchern sind hauptsächlich für die BASIC Tiger® A Module gemacht und einige laufen nicht auf dem BASIC Tiger® E, insbesondere wenn nur 32k RAM vorhanden sind. Durch die Doppelnutzung der Pins bei den BASIC Tiger® E müssen in einigen Fällen Maßnahmen ergriffen werden, um die gewünschte Anwendung mit diesem Modul zu realisieren. Hier zwei exemplarische Beispiele:

Verwendung der Analogen Eingänge: In jedem Fall ergeben sich durch die internen Pull-up-Widerstände der digitalen I/O-Pins andere elektrische Werte. Die Eingänge müssen niederohmig getrieben werden. Die Steuerpins des erweiterten Portsystems liegen hier. Werden diese nicht gebraucht, kann man das EPORT-System abschalten (USER_EPORT ACT, NOACTIVE). Werden sie gebraucht, kann man die Steuerfunktion auf andere Pins verlegen (DEFINE_A.INC einbinden):

Kommando	Wert	Bedeutung und Argument
BUSL	2	setze Port für Datenbus (Standard = Port 6)
CTRL	3	setze Port für Steuerleitungen des erweiterten Portsystems (Standard = 3)
DCLKB MASK	4	setze Bitmaske für DCLK
ACLB MASK	5	setze Bitmaske für ACLK
INEB MASK	6	setze Bitmaske für INE

Beispiel: setze Steuerleitungen für das erweiterte Portsystem auf Port 8, Pins 0...2:

```
#include DEFINE_A.INC

user_eport CTRL, 8
user_eport ACLB MASK, 00000001b
user_eport DCLKB MASK, 00000010b
user_eport INEB MASK, 00000100b
```

Nähere Informationen zum erweiterten I/O-System siehe im Hardware-Handbuch.

Einbinden von LCD1: standardmäßig liegt der Sound-Ausgang für Tastenklick, Beep, usw. auf dem Pin L87. Dieser wird als Ausgang den RxD0-Pin stören, sofern der serielle Kanal ebenfalls benutzt wird. Die Funktion kann auf einen anderen Pin gelegt oder abgeschaltet werden. Siehe dazu die Beschreibung des LCD1-Device-Treibers. Durch die Tastatur (=erweiterte Eingänge) sind die Steuerpins L33, L34 und L35 aktiv. Sie liegen auf den analogen Ports, die Funktion kann aber verlegt werden.

Der freie RAM-Speicher in einem 32k-TINY Tiger®-Modul E beträgt bei geladenem Programm nicht viel mehr als 10kByte. Die folgenden Hinweise sollen Ihnen helfen, ein Maximum an Leistung trotz wenig RAM zu erzielen:

4

- Die Standard-Stackgröße ist für jede Task 2kByte. Schreiben Sie in Ihren Quellcode die Zeile 'USER_STACK_SIZE nnn', um den Stack jeder Task auf bis zu 512Byte zu verkleinern.
- Die Standard-Stringgröße ist 64 Bytes. Sie sollten die Stringgröße angeben und auf die wirklich benötigte Größe begrenzen. Diese Regel gilt speziell für ARRAYs of STRINGS.
- Binden Sie nur Device-Treiber ein, die Ihre Anwendung wirklich benötigt. Speziell gepufferte Treiber wie LCD1.TDD, SER1.TDD, PRN1.TDD brauchen RAM für ihre Puffer.
- Bei der Erklärung der CALL-Instruktion finden Sie Informationen über die Benutzung des Stacks. Parameter, die an ein Unterprogramm übergeben werden sowie lokale Variable benötigen Stack entsprechend ihrer Größe.
- Während der Entwicklungsphase Ihres Projektes können Sie die Stackauslastung während der Laufzeit mit der Funktion SYSVARN abfragen (Funktionsnummern DSTACK_FILL und DSTACK_FREE).

Die folgenden technischen Daten gelten für alle BASIC Tiger®-E-Module:

Maße	ca. 39 x 28 x 10,2 mm / ca. 16g
Stromversorgung	4.7V→5.5V / ca. 45 mA
Belastbarkeit der I/O-Pins	alle Pins 1.6mA, bzw. max. 8x 3.5mA
Reset	Power-ON-Reset auf dem Modul sowie durch externen Eingang
2 serial channels L86, L87 auf RxD0, CTS0	TTL-level (Pin 15...Pin 19) Pins 15...19 können auch durch Einstellen per Software als digitale I/O verwendet werden.
A/D-Eingänge L33...L36 auf den gleichen Pins	4 Kanäle (Pin 23 bis Pin 26) Auflösung 8-, 10-Bit Eingangsspannung 0V...Vref (Vref=5V intern) Eingangswiderstand: 1M bei ausgeschaltetem Modul sehr niederohmig!

4

Technische Daten, die vom Modultyp abhängen:

RAM	32kB, 128kB, 512kB
Flash-ROM	128KB, 512kB

TINY Tiger® Modul E Pinbelegung

4

L60	●1		28●	VCC (5V)
L61	●2		27●	L37
L62	●3		26●	Analog-3, L36
L63	●4		25●	Analog-2, L35
L64	●5		24●	Analog-1, L34
L65	●6		23●	Analog-0, L33
L66	●7		22●	PC
L67	●8		21●	L85
L80	●9		20●	RESET-in
L81	●10		19●	L94, RxD1
L82	●11		18●	L93, TxD1
L83	●12		17●	L92, CTS0, L86
L84	●13		16●	L91, RxD0, L87
GND	●14		15●	L90, TxD0

BASIC Tiger® Module E Pinbeschreibung

Pin-Nr.	Name	Beschreibung
1	L60	Port 6 - Pin 0
2	L61	Port 6 - Pin 1
3	L62	Port 6 - Pin 2
4	L63	Port 6 - Pin 3
5	L64	Port 6 - Pin 4
6	L65	Port 6 - Pin 5
7	L66	Port 6 - Pin 6
8	L67	Port 6 - Pin 7
9	L80	Port 8 - Pin 0
10	L81	Port 8 - Pin 1
11	L82	Port 8 - Pin 2
12	L83	Port 8 - Pin 3
13	L84	Port 8 - Pin 4
14	GND	Masse

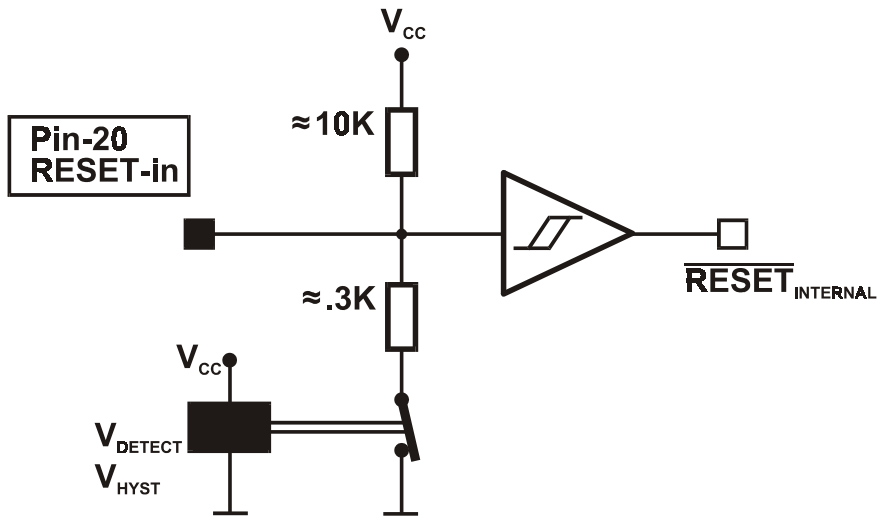
Pin-Nr.	Name	Beschreibung
15	L90 oder TxD0	Port 9 - Pin 0 oder Sendeleitung serieller Port 0
16	L91 oder RxD0 und L87	Port 9 - Pin 1 oder Empfangsleitung serieller Port 0 und Port 8 - Pin 7
17	L92 oder CTS0 und L86	Port 9 - Pin 2 oder CTS0 und Port 8 - Pin 6
18	L93 oder TxD1	Port 9 - Pin 3 oder Sendeleitung serieller Port 1
19	L94 oder RxD1	Port 9 - Pin 4 oder Empfangsleitung serieller Port 1
20	Reset in	Reset-Eingang (interner Pull-up, low aktive)
21	L85	Port 8 - Pin 5
22	L41 / PC	PC-Mode-Pin (low=PC, high=RUN)
23	Analog In 0 und L33	Analog-Eingang 0 (0V→VCC) und Port 3 - Pin 3
24	Analog In 1 und L34	Analog-Eingang 1 (0V→VCC) und Port 3 - Pin 4
25	Analog In 2 und L35	Analog-Eingang 2 (0V→VCC) und Port 3 - Pin 5
26	Analog In 3 und L36	Analog-Eingang 3 (0V→VCC) und Port 3 - Pin 6
27	L37	Port 3 - Pin 7
28	VCC (5V)	Stromversorgung

TINY Tiger® Modul E RESET-In

TINY-Tiger E verfügt über einen RESET-Eingang sowie einen internen RESET-Generator. Der RESET-Eingang kann auf verschiedene Weise beschaltet werden oder offen bleiben. Der RESET wird üblicherweise für verschiedene Aufgaben eingesetzt:

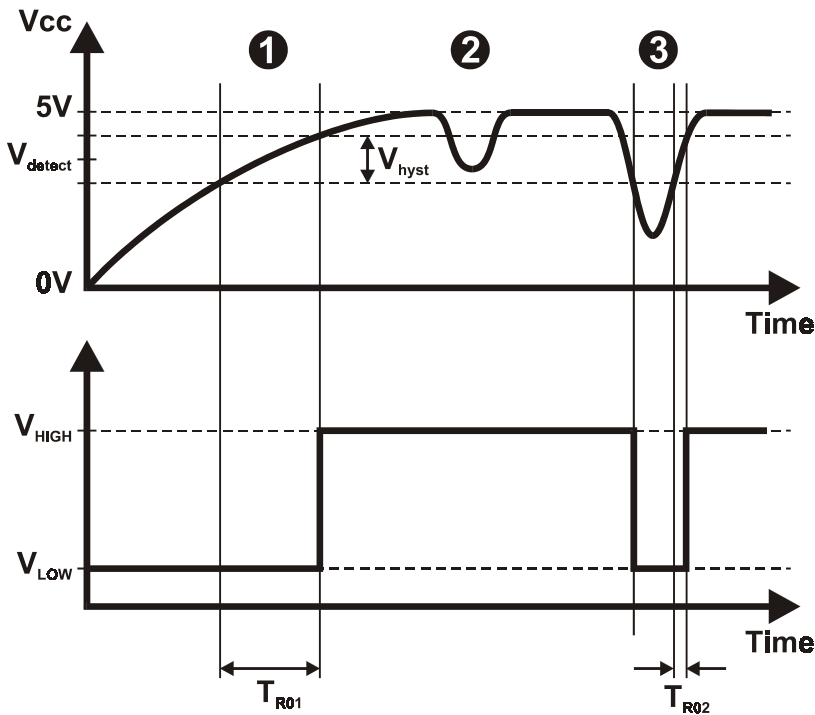
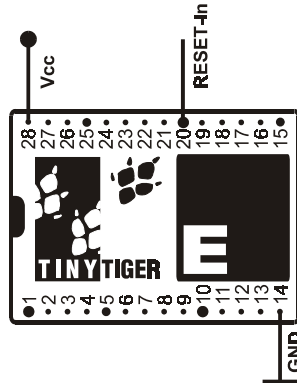
- definierte Startbedingungen des Computers beim Power-Up herstellen
- Synchronisation mehrerer Computer und logischer Einheiten beim Hochfahren
- Manueller Abbruch, „Paniktaste“ (Warmstart)
- Zusätzliche Sicherheitsfunktion mit zusätzlichem externen Watchdog.

Nachfolgendes Ersatzschaltbild gibt hierzu die Funktion des intern RESET-Generators wieder.



BASIC-Tiger®-Steuercomputer

In Stand-alone Anwendungen besteht in der Regel der Wunsch nach minimaler Zahl an Bauteilen, so dass man hier den RESET-Eingang oft einfach unbeschaltet lässt. Die sich daraus ergebenden Zusammenhänge für die Erzeugung des RESET-Signals zeigt nachfolgende Abbildung.



Die Zeit-Angaben TR01 (Power-Up) und TR02 (Störung) geben jeweils den effektiv nutzbaren Teil des RESET-Signals wieder, der nur dann zählt, wenn V_{cc} über der Mindest-Versorgungsspannung $V_{ccMin} = V_{DETECT} - 0.5 \cdot V_{HYST}$ liegt.

In der Standard-Ausführung liegen beim TINY-Tiger E die Werte für den RESET-Generator bei typisch:

- V_{DETECT} : 4.6V
- V_{HYST} : 0.36V

Eine weiterhin relevante Kenngröße ist die Mindestdauer des RESET-Signals (low-aktiv!)

- $T_{RESET_MIN} > 10 \text{ usec}$

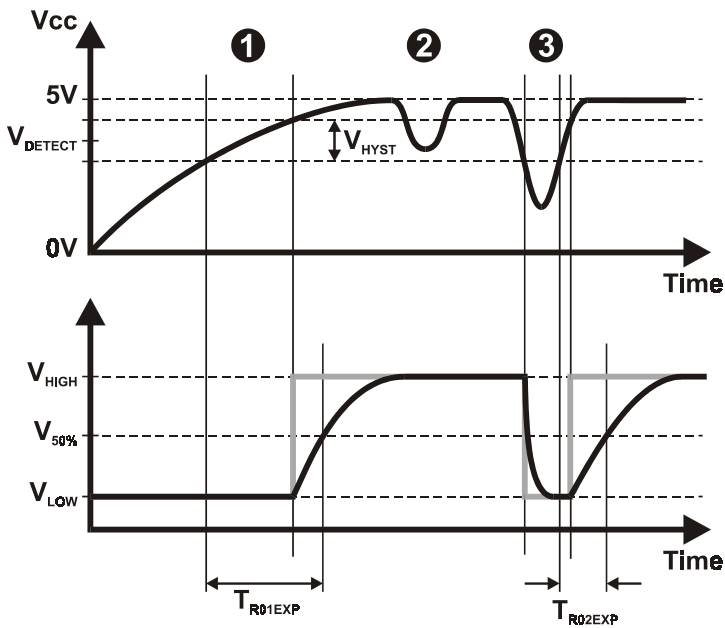
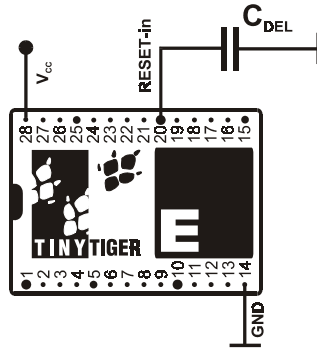
T_{RESET_MIN} wird gemessen zwischen Erreichen der Mindest-Versorgungsspannung $V_{ccMin} = V_{DETECT} - 0.5 \cdot V_{HYST}$ und dem Ende der aktiven Phase des RESET-Signals, was bei Erreichen der Spannung: $V_{DETECT} + 0.5 \cdot V_{HYST}$ eintritt. Die Einhaltung der Mindest-Dauer für das RESET-Signal ist für das einwandfreie RESETten des Moduls unbedingt erforderlich.

Hierzu sind folgende 3 Fälle zu betrachten:

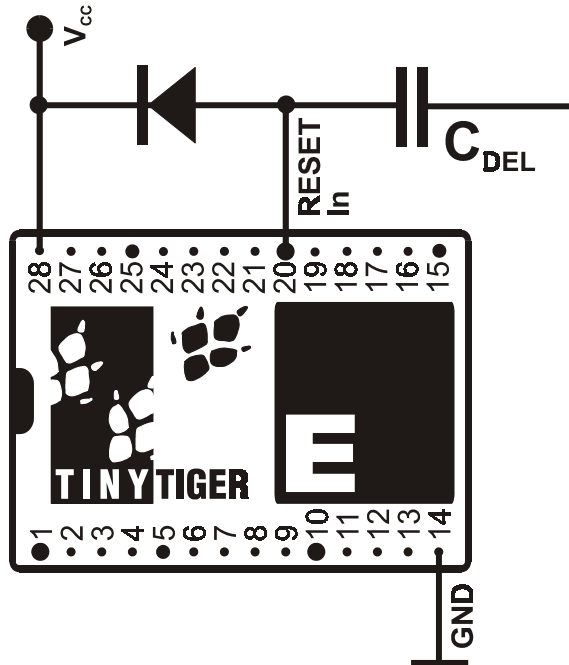
- Power-UP
- Plötzlicher Spannungseinbruch (Störung)
- Beabsichtigter RESET im Betrieb, Warmstart.

BASIC-Tiger®-Steuercomputer

Gegebenenfalls lässt sich die Dauer des RESET-Signals durch Beschaltung des RESET-Eingangs mit einem Kondensator zusätzlich verlängern. Typische Werte hierfür liegen bei 100...500 nF. Die Wirkung dieser Beschaltung zeigt folgende Abbildung:



Man beachte die Verlängerung der effektiven RESET-Signaldauer in beiden Fällen: TR01EXP (Power-Up) und TR02 EXP (Störung). Ebenso wird ersichtlich, dass besonders kurze Spannungseinbrüche nur so lange noch einen sicheren RESET auslösen, wie die Zeitdauer des Einbruchs ausreicht die externe Kapazität zu entladen. Als zusätzliche Maßnahme in solchen Fällen könnte eine Freilaufdiode helfen:



4

Schließlich wird immer dann ein externes RESET-Signal zugeführt, wenn die o.g. Bedingungen nicht erfüllt werden können. Ebenso sollte man in größeren Systemen, die aus mehreren Computern und Logikeinheiten bestehen, stets einen zentralen RESET generieren und an alle Einheiten weiterleiten. Stromversorgungen geben zu diesem Zweck ein „Power-good“ Signal ab, das erst dann aktiv wird (high), wenn sich alle Ausgangsspannungen stabil aufgebaut haben.

BASIC-Tiger®-Steuercomputer

Leere Seite

4

TCAN – CAN-Tiger

Das Modul TCAN entspricht weitgehend den Modellen der Modulreihe A mit zusätzlicher CAN-Hardware. Die Echtzeit-Uhr ist immer enthalten. Interne RS-232-Treiber sind nicht vorgesehen.

Technische Merkmale des TCAN-4/4:

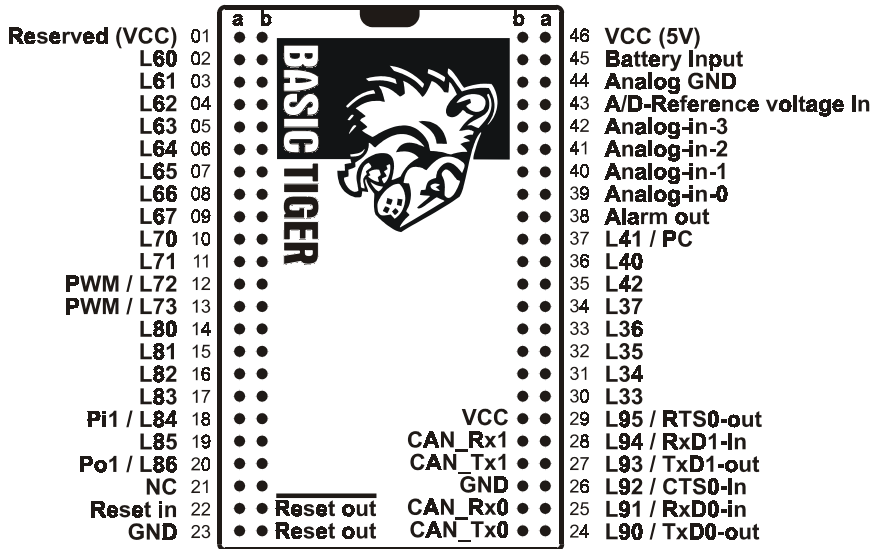
Außenmaße / Gewicht:	ca. 63 x 41 x 12 mm / ca. 50g
Stromversorgung	4,7V...5,5V / ca. 45 mA
Belastbarkeit der I/O-Pins	alle Pins 1,6mA, bzw. max. 8x 3,5mA
Reset	Power-ON-Reset auf dem Modul sowie durch externen Eingang
Batterie f. Uhr + RAM-Pufferung	Anschluß für externe Pufferbatterie (Pin 45a) U-batt 3.0...5.0V, I-batt 50µA...150µA
A/D-Eingänge	4 Kanäle (Pin 39a bis Pin 42a) Auflösung 8-, 10-Bit Eingangsspannung 0V...Vref (Vref=3.5V...5V, Iref-max=1.5mA) Eingangswiderstand: 1M bei ausgeschaltetem Modul sehr niederohmig!
PWM	2 PWM-Kanäle (Pin 12a und Pin 13a) Auflösung 6-, 7-, 8-Bit Wiederholfrequenz 1.2kHz...80kHz
RAM	512KB
Flash-ROM	512KB
Uhr	mit Kalender + Weckfunktion (Alarm aktiv low, Pin 38a)
2 serielle Kanäle	TTL-Pegel (Pin 24a...Pin 29) Pins 24...29 können auch durch Einstellen per Software als digitale I/O verwendet werden.

EMV-Hinweise: Obwohl bereits in dem Modul einiges für die ‘elektromagnetische Verträglichkeit’ und gegen Störausstrahlung getan wurde (Multi-Layerplatine mit VCC- und GND-Layer, durch Innen-Bedampfung abgeschirmtes Gehäuse, VCC-Ferrit), müssen weitere Maßnahmen getroffen werden, wenn ein CE-Zeichen für das Endgerät angestrebt wird. Das Modul betreffend werden folgende Maßnahmen vorgeschlagen:

- möglichst dicht am VCC-Pin des Moduls einen Tantal- sowie einen keramischen Kondensator vorsehen (z.B. $47\mu\text{F} + 100\text{nF}$).
- VCC über eine Drossel mit VCC verbinden (z.B. 68nH).
- Alle Ein- und Ausgänge nach Möglichkeit über 1nF oder entsprechende Varistoren mit ähnlicher Kapazität mit Masse verbinden.
- Nicht verwendete I/O-Pins per Software als Eingang initialisieren.

Es gibt kein Patentrezept, wie ein Projekt störfest und abstrahlungsarm wird. Neben den obigen Vorschlägen spielt immer auch die zusätzlich vorhandene Peripherie und Beschaltung eine Rolle.

BASIC-Tiger®-TCAN-Modul Pinbelegung



4

Die Pins PWM, Pi1, Po1 sowie die seriellen Pins sind digitale I/O-Pins, wenn der zugehörige Device-Treiber in der Anwendung nicht eingebunden ist.

Pin-Beschreibung TCAN

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
1a		reserviert	1b		unbelegt
2a	L60	Port 6 - Pin 0	2b		unbelegt
3a	L61	Port 6 - Pin 1	3b		unbelegt
4a	L62	Port 6 - Pin 2	4b		unbelegt
5a	L63	Port 6 - Pin 3	5b		unbelegt
6a	L64	Port 6 - Pin 4	6b		unbelegt
7a	L65	Port 6 - Pin 5	7b		unbelegt
8a	L66	Port 6 - Pin 6	8b		unbelegt
9a	L67	Port 6 - Pin 7	9b		unbelegt
10a	L70	Port 7 - Pin 0	10b		unbelegt
11a	L71	Port 7 - Pin 1	11b		unbelegt
12a	L72 PWM	Port 7 - Pin 2 oder PWM-Ausgang	12b		unbelegt
13a	L73 PWM	Port 7 - Pin 3 oder PWM-Ausgang	13b		unbelegt
14a	L80	Port 8 - Pin 0	14b		unbelegt
15a	L81	Port 8 - Pin 1	15b		unbelegt
16a	L82	Port 8 - Pin 2	16b		unbelegt
17a	L83	Port 8 - Pin 3	17b		unbelegt
18a	L84	Port 8 - Pin 4	18b		unbelegt
19a	L85	Port 8 - Pin 5	19b		unbelegt
20a	L86	Port 8 - Pin 6	20b		unbelegt
21a		unbelegt	21b		unbelegt
22a	Reset in	Reset-Eingang (Tas- ter gegen GND)	22b	/Reset out	Reset-Ausgang, low aktiv
23a	GND	Masse	23b	Reset out	Reset-Ausgang, high aktiv

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
24a	L90 TxD0	Port 9 - Pin 0 oder Sendeleitung serieller Port 0	24b	CAN- Tx0	CAN Tx0
25a	L91 RxD0	Port 9 - Pin 1 oder Empfangsleitung serieller Port 0	25b	CAN- Rx0	CAN Rx0
26a	L92 CTS0	Port 9 - Pin 2 oder CTS0 (Eingang)	26b	GND	Masse
27a	L93 TxD1	Port 9 - Pin 3 oder Sendeleitung serieller Port 1	27b	CAN- Tx1	CAN Tx1
28a	L94 RxD1	Port 9 - Pin 4 oder Empfangsleitung serieller Port 1	28b	CAN- Rx1	CAN Rx1
29a	L95 RTS0	Port 9 - Pin 5 oder RTS0 (Ausgang)	29b	VCC (5V)	Spannungs- versorgung
30a	L33	Port 3 - Pin 3	30b		unbelegt
31a	L34	Port 3 - Pin 4	31b		unbelegt
32a	L35	Port 3 - Pin 5	32b		unbelegt
33a	L36	Port 3 - Pin 6	33b		unbelegt
34a	L37	Port 3 - Pin 7	34b		unbelegt
35a	L42	Port 4 - Pin 2	35b		unbelegt
36a	L40	Port 4 - Pin 0	36b		unbelegt
37a	L41 PC	PC-Mode-Pin	37b		unbelegt
38a	Alarm out	Alarm Ausgang der Uhr	38b		unbelegt
39a	Analog In 0	Analogeingang 0 (0V...VCC)	39b		unbelegt
40a	Analog In 1	Analogeingang 1 (0V...VCC)	40b		unbelegt
41a	Analog In 2	Analogeingang 2 (0V...VCC)	41b		unbelegt
42a	Analog	Analogeingang 3	42b		unbelegt

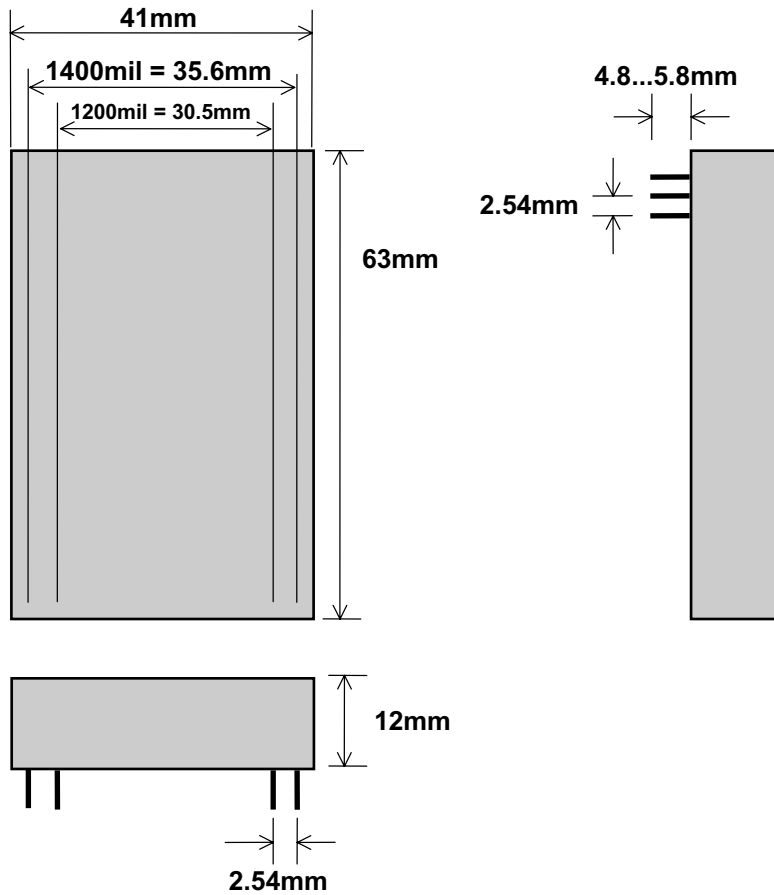
BASIC-Tiger®-Steuercomputer

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
	In 3	(0V...VCC)			

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
43a	A/D Ref-in	A/D-Referenzspannung.: 3,5...5,0V, 1,5mA max.	43b		unbelegt
44a	Analog GND	Analog Masse (intern mit GND verbunden)	44b		unbelegt
45a	Batterie in	Anschluß für externe Batterie 3,0...5,0V	45b		unbelegt
46a	VCC (5V)	Spannungsversorgung	46b		

4

Abmessungen:



4

BASIC-Tiger®-Steuercomputer

Leere Seite

4

I/O-Erweiterungsmodule

Das Tiger-BASIC[®]-System unterstützt bereits mit seiner Software die Erweiterung der I/Os des BASIC-Tiger[®]-Moduls oder Tiny-Tiger[®]-Moduls um weitere 1920 Inputs oder Outputs. Die Serie der I/O-Erweiterungsmodule für BASIC-Tiger[®] liefert die passende Hardware dazu. Die Module sind sehr kompakt und enthalten auf kleinstem Raum eine Vielzahl zusätzlicher I/O-Leitungen und Funktionen. Der Anschluß erfolgt einfach über einen 8-Bit-Port als Datenbus und einige Steuerleitungen.

Die Basisadressen der Module sind einstellbar. Auf diese Weise können verschiedene Module miteinander kombiniert werden. **Beachte:** Erweiterte Ausgänge müssen adreßmäßig unterhalb von erweiterten Eingängen liegen.

Module mit Open-Collector-Ausgängen schalten hohe Ströme und haben daher eine entsprechende Verlustleistung. Einige dieser Module haben einen eingebauten Temperaturfühler mit Schaltausgang- und Analogausgang. Mit dem Schaltausgang kann z.B. autonom ein Ventilator geschaltet werden. Über den Analogausgang wird der Temperaturverlauf kontrollierbar.

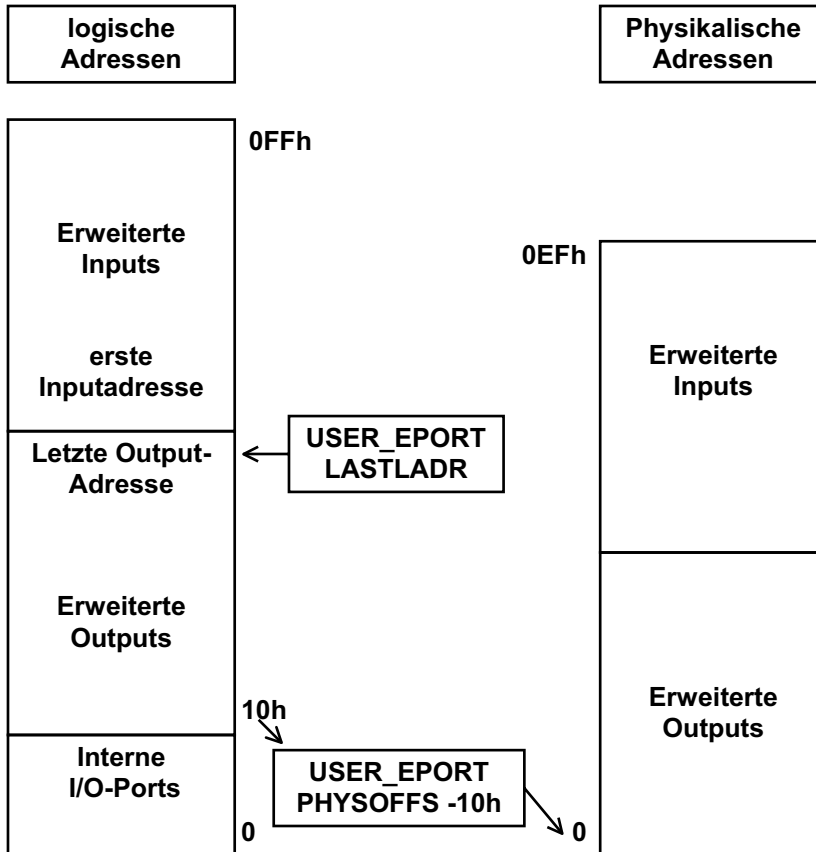
Durch die weitgehend übereinstimmende Pinbelegung der Erweiterungsmodule sind Platinen-Layouts möglich, die durch wenige Jumper verschiedene Module austauschbar machen. So kann derselbe Steckplatz eines Moduls auf der Platine sowohl für erweiterte Inputs als auch für erweiterte Outputs verwendet werden, oder in Zukunft sogar für analoge Ein/Ausgabe.

Folgende Module sind im folgenden beschrieben:

Modul	Funktion
EP1-64HDE	64 digitale Inputs
EP2-64SDA	64 digitale Outputs
EP3-32-32	32 digitale Outputs, 32 digitale Inputs
EP4-32PDA	32 OC-Outputs
EP5-32GDE	32 opto-entkoppelte Inputs
EP6-UNIVD	8x16 keyboard, 8 OC-Outputs, 8 Outputs, 8 Inputs
EP10-16PDA/GDE	16 OC-Outputs, 16 opto-entkoppelte Inputs
EP11-8AD EP12-16AD EP13-32AD EP14-64AD	8 bis 64 A/D-Eingänge mit 12-Bit-Auflösung

Adreßgenerierung

Die Abbildung zeigt, wie das EPORT-System die logischen 240 (0E0h) Erweiterungsadressen in physikalische Adressen umsetzt:

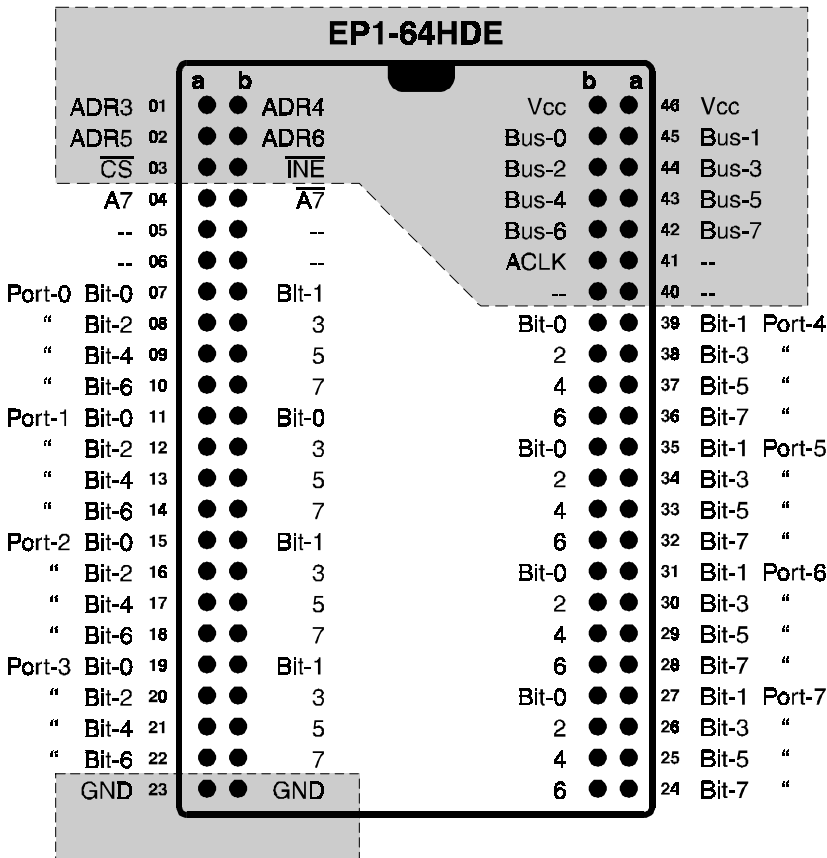


EP1-64HDE

64 digitale Eingänge

Das I/O-Erweiterungsmodul EP1-64HDE bietet 64 hochohmige digitale Inputs (0...5V, HC-MOS). Durch die einstellbare Basisadresse kann das Modul mit anderen Erweiterungen kombiniert werden.

Pinbelegung EP1-64HDE



Pin-Beschreibung EP1-64HDE

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
1a	ADR3	Adresse	1b	ADR4	AdressBit 4
2a	ADR5	Adresse	2b	ADR6	AdressBit 6
3a	-CS	ChipSelect	3b	-INE	In-Enable
4a	A7	Adressbit 7	4b	-A7	Neg.. Adressbit 7
5a	--		5b	--	
6a	--		6b	--	
7a	P0.0	Port 0 Bit0	7b	P0.1	Port0 Bit1
8a	P0.2	Port0 Bit2	8b	P0.3	Port0 Bit3
9a	P0.4	Port0 Bit4	9b	P0.5	Port0 Bit5
10a	P0.6	Port0 Bit6	10b	P0.7	Port0 Bit7
11a	P1.0	Port1 Bit0	11b	P1.1	Port1 Bit1
12a	P1.2	Port1 Bit2	12b	P1.3	Port1 Bit3
13a	P1.4	Port1 Bit4	13b	P1.5	Port1 Bit5
14a	P1.6	Port1 Bit6	14b	P1.7	Port1 Bit7
15a	P2.0	Port2 Bit0	15b	P2.1	Port2 Bit5
16a	P2.2	Port2 Bit2	16b	P2.3	Port2 Bit3
17a	P2.4	Port2 Bit4	17b	P2.5	Port2 Bit1
18a	P2.6	Port2 Bit6	18b	P2.7	Port2 Bit7
19a	P3.0	Port3 Bit0	19b	P3.1	Port3 Bit1
20a	P3.2	Port3 Bit2	20b	P3.3	Port3 Bit3
21a	P3.4	Port3 Bit4	21b	P3.5	Port3 Bit5
22a	P3.6	Port3 Bit6	22b	P3.7	Port3 Bit7
23a	GND	Masse	23b	GND	Masse

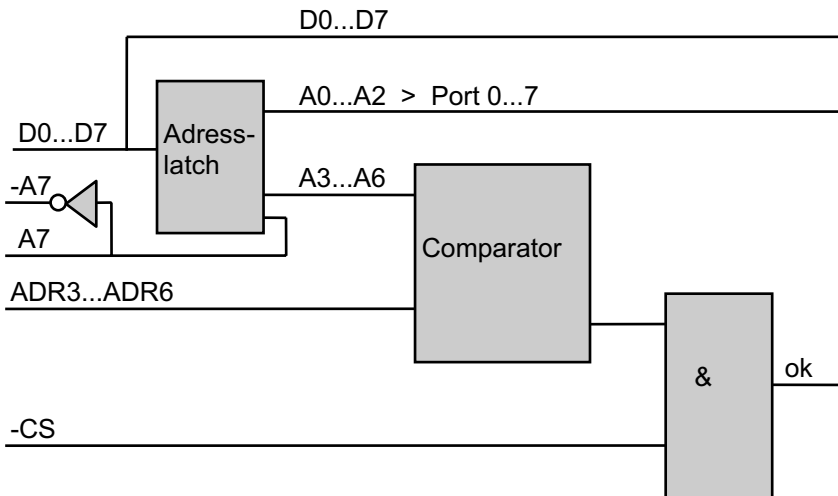
Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
24a	P7.7	Port7 Bit7	24b	P7.6	Port7 Bit6
25a	P7.5	Port7 Bit5	25b	P7.4	Port7 Bit4
26a	P7.3	Port7 Bit3	26b	P7.2	Port7 Bit2
27a	P7.1	Port7 Bit1	27b	P7.0	Port7 Bit0
28a	P6.7	Port6 Bit7	28b	P6.6	Port6 Bit6
29a	P6.5	Port6 Bit5	29b	P6.4	Port6 Bit4
30a	P6.3	Port6 Bit3	30b	P6.2	Port6 Bit2
31a	P6.1	Port6 Bit1	31b	P6.0	Port6 Bit0
32a	P5.7	Port5 Bit7	32b	P5.6	Port5 Bit6
33a	P5.5	Port5 Bit5	33b	P5.4	Port5 Bit4
34a	P5.3	Port5 Bit3	34b	P5.2	Port5 Bit2
35a	P5.1	Port5 Bit1	35b	P5.0	Port5 Bit0
36a	P4.7	Port4 Bit7	36b	P4.6	Port4 Bit6
37a	P4.5	Port4 Bit5	37b	P4.4	Port4 Bit4
38a	P4.3	Port4 Bit3	38b	P4.2	Port4 Bit2
39a	P4.1	Port4 Bit1	39b	P4.0	Port4 Bit0
40a	--		40b	--	
41a			41b	ACLK	Adressclock
42a	Bus-7	I/O-Bus Bit7	42b	Bus-6	I/O-Bus Bit6
43a	Bus-5	I/O-Bus Bit5	43b	Bus-4	I/O-Bus Bit4
44a	Bus-3	I/O-Bus Bit3	44b	Bus-2	I/O-Bus Bit2
45a	Bus-1	I/O-Bus Bit1	45b	Bus-0	I/O-Bus Bit0
46a	VCC		46b	VCC	

Adressierung EP1-64HDE

Das Modul EP1-64HDE wird direkt von dem Tiger-BASIC® EPORT-System unterstützt. Der Anschluß an ein BASIC-Tiger®-Modul A oder Tiny-Tiger®-Modul erfordert bei der Standard-Pinbenutzung folgende Pins:

Pin-Bezeichnung	BASIC-Tiger®-Pins	Modultyp-A Pin-Nr.	Tiny-Tiger Pin-Nr.
Bus-0...Bus-7	L60...L67	2...9	1...8
ACLK (Adressclock)	L33	30	29
-INE (in-enable)	L35	32	31
ADR3...ADR6	A3...A6 Inputs für Basis-Adresse des Moduls		
-CS	Chip-Select Input, low aktiv		
weitere Pins			
A7	A7-Ausgang des internen Adreßlatch		
-A7	A7-Ausgang invertiert		
Port-m Bit-n	Ports (m=0...7) mit je 8 Bits (n=0...7)		

Blockschaltbild der Adressierung des EP1-64HDE



An den Leitungen ADR3...ADR6 wird die Basis-Adresse des Erweiterungsmoduls angelegt. Die erweiterten Input-Ports belegen 8 Adressen ab der Basisadresse. Das Modul wird adressiert, wenn '-CS' auf 'low' liegt und eine Adresse im eingestellten Bereich angesprochen wird. Die Basisadresse kann um 80h angehoben werden, indem '-A7' mit '-CS' verbunden wird.

EP1-64HDE Adressierung						
-CS	ADR6	ADR5	ADR4	ADR3	Port-0	Port-7
1	x	x	x	x	—	—
0	0	0	0	0	0	7
0	0	0	0	1	8	0Fh
0	0	0	1	0	10h	17h
0	0	0	1	1	18h	1Fh
0	0	1	0	0	20h	27h
0	0	1	0	1	28h	2Fh
0	0	1	1	0	30h	37h
0	0	1	1	1	38h	3Fh
0	1	0	0	0	40h	47h
0	1	0	0	1	48h	4Fh
0	1	0	1	0	50h	57h
0	1	0	1	1	58h	5Fh
0	1	1	0	0	60h	67h
0	1	1	0	1	68h	6Fh
0	1	1	1	0	70h	77h
0	1	1	1	1	78h	7Fh

Da A7 intern nicht ausgewertet wird, spiegeln sich die Adressen ab Adresse 80h.

4

228 Wilke Technology GmbH • web: www.wilke.de • eMail: info@wilke.de

Programmbeispiel:

```

'-----
' Name: EP1-1.TIG
'-----
#INCLUDE DEFINE_A.INC          ' allgemeine Definitionen
USER_EPORT PHYSOFFS, 0F0h      ' Offset zu phys. Adr. -10h

TASK MAIN                      ' Beginn Task MAIN
' LCD-Treiber installieren (BASIC-Tiger)
  INSTALL DEVICE #1, "LCD1.TDD"
' LCD-Treiber installieren (TINY-Tiger)
'  INSTALL DEVICE #1, "LCD1.TDD", 0, 0, 0, 0, 0, 0, 80h, 8

  FOR I = 90h TO 97h           ' von Ports 0..7
    IN I,VALUE                 ' lies von Port
    PRINT #1, "Port";I-90h;"=";VALUE ' Wert ausgeben
    WAIT_DURATION 1000         ' 1 Sek. warten
  NEXT                         ' naechster Port
END                            ' Ende Task MAIN

```

4

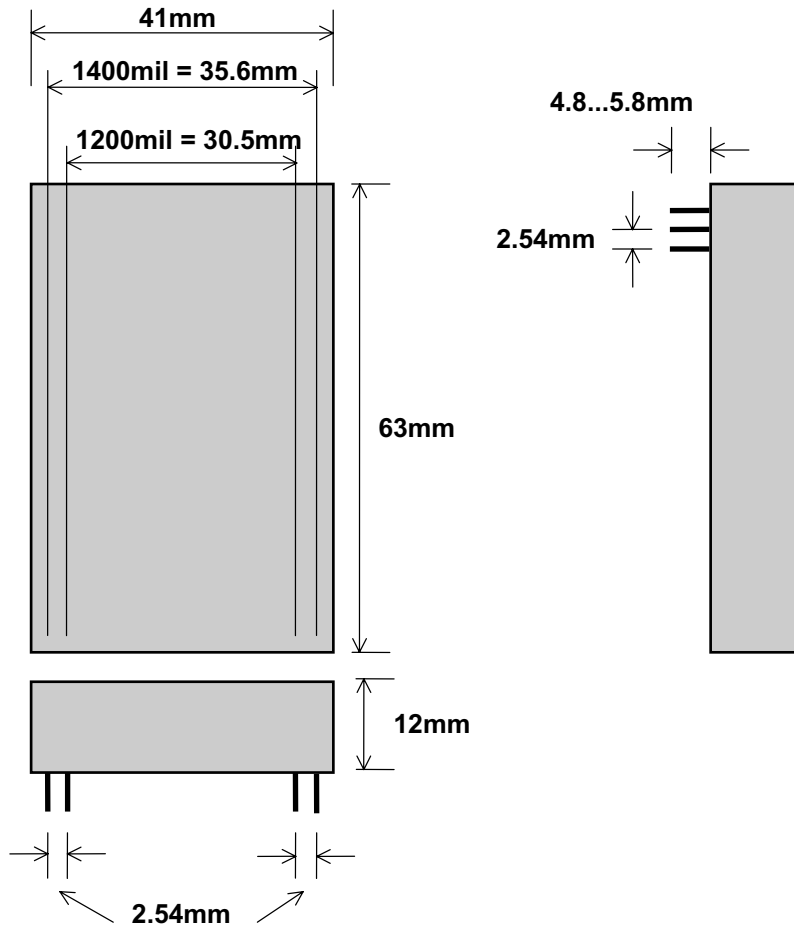
I/O-Erweiterungsmodule

Technische Daten des I/O-Erweiterungsmoduls EP1-64HDE:

Maße / Gewicht	ca. 63 x 41 x 12 mm / ca. 50g
Spannungsversorgung	4,7V...5,5V
Ruhestromaufnahme	ca. 120µA
Anzahl der erweiterten Eingänge	64
Eingangsspannung	0...5V (HC-MOS-Eingang)
Temperaturbereich	-40 bis +85°C

4

Abmessungen EP1-64HDE



4

I/O-Erweiterungsmodule

Leere Seite

4

Pin-Beschreibung EP2-64SDA

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
1a	ADR3	Adresse	1b	ADR4	AdressBit 4
2a	ADR5	Adresse	2b	ADR6	AdressBit 6
3a	-CS	ChipSelect	3b		
4a	A7	Adressbit 7	4b	-A7	Neg.. Adressbit 7
5a			5b		
6a			6b		
7a	P0.0	Port 0 Bit0	7b	P0.1	Port0 Bit1
8a	P0.2	Port0 Bit2	8b	P0.3	Port0 Bit3
9a	P0.4	Port0 Bit4	9b	P0.5	Port0 Bit5
10a	P0.6	Port0 Bit6	10b	P0.7	Port0 Bit7
11a	P1.0	Port1 Bit0	11b	P1.1	Port1 Bit1
12a	P1.2	Port1 Bit2	12b	P1.3	Port1 Bit3
13a	P1.4	Port1 Bit4	13b	P1.5	Port1 Bit5
14a	P1.6	Port1 Bit6	14b	P1.7	Port1 Bit7
15a	P2.0	Port2 Bit0	15b	P2.1	Port2 Bit5
16a	P2.2	Port2 Bit2	16b	P2.3	Port2 Bit3
17a	P2.4	Port2 Bit4	17b	P2.5	Port2 Bit1
18a	P2.6	Port2 Bit6	18b	P2.7	Port2 Bit7
19a	P3.0	Port3 Bit0	19b	P3.1	Port3 Bit1
20a	P3.2	Port3 Bit2	20b	P3.3	Port3 Bit3
21a	P3.4	Port3 Bit4	21b	P3.5	Port3 Bit5
22a	P3.6	Port3 Bit6	22b	P3.7	Port3 Bit7
23a	GND	Masse	23b	GND	Masse

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
24a	P7.7	Port7 Bit7	24b	P7.6	Port7 Bit6
25a	P7.5	Port7 Bit5	25b	P7.4	Port7 Bit4
26a	P7.3	Port7 Bit3	26b	P7.2	Port7 Bit2
27a	P7.1	Port7 Bit1	27b	P7.0	Port7 Bit0
28a	P6.7	Port6 Bit7	28b	P6.6	Port6 Bit6
29a	P6.5	Port6 Bit5	29b	P6.4	Port6 Bit4
30a	P6.3	Port6 Bit3	30b	P6.2	Port6 Bit2
31a	P6.1	Port6 Bit1	31b	P6.0	Port6 Bit0
32a	P5.7	Port5 Bit7	32b	P5.6	Port5 Bit6
33a	P5.5	Port5 Bit5	33b	P5.4	Port5 Bit4
34a	P5.3	Port5 Bit3	34b	P5.2	Port5 Bit2
35a	P5.1	Port5 Bit1	35b	P5.0	Port5 Bit0
36a	P4.7	Port4 Bit7	36b	P4.6	Port4 Bit6
37a	P4.5	Port4 Bit5	37b	P4.4	Port4 Bit4
38a	P4.3	Port4 Bit3	38b	P4.2	Port4 Bit2
39a	P4.1	Port4 Bit1	39b	P4.0	Port4 Bit0
40a			40b		
41a	DCLK	Datenclock	41b	ACLK	Adressclock
42a	Bus-7	I/O-Bus Bit7	42b	Bus-6	I/O-Bus Bit6
43a	Bus-5	I/O-Bus Bit5	43b	Bus-4	I/O-Bus Bit4
44a	Bus-3	I/O-Bus Bit3	44b	Bus-2	I/O-Bus Bit2
45a	Bus-1	I/O-Bus Bit1	45b	Bus-0	I/O-Bus Bit0
46a	VCC		46b	VCC	

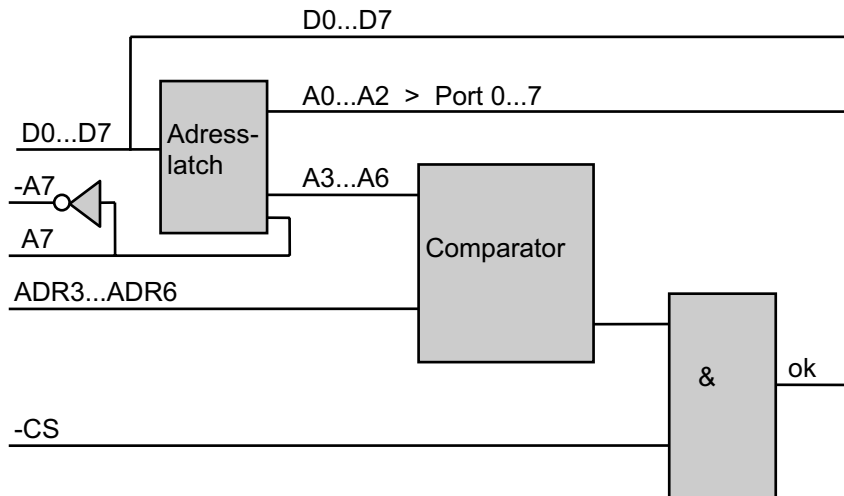
I/O-Erweiterungsmodule

Adressierung EP2-64SDA

Das Modul EP2-64SDA wird direkt von dem Tiger-BASIC[®] EPORT-System unterstützt. Der Anschluß an ein BASIC-Tiger[®]-Modul A oder Tiny-Tiger[®]-Modul erfordert bei der Standard-Pinbenutzung folgende Pins:

Pin-Bezeichnung	BASIC-Tiger [®] -Pins	Modultyp-A Pin-Nr.	Tiny-Tiger Pin-Nr.
Bus-0...Bus-7	L60...L67	2...9	1...8
ACLK (Adressclock)	L33	30	29
DCLK (Datenclock)	L34	31	30
ADR3...ADR6	A3...A6 Inputs für Basis-Adresse des Moduls		
-CS	Chip-Select Input, low aktiv		
weitere Pins			
A7	A7-Ausgang des internen Adreßlatch		
-A7	A7-Ausgang invertiert		
Port-m Bit-n	Ports (m=0...7) mit je 8 Bits (n=0...7)		

Blockschaltbild der Adressierung des EP2-64SDA



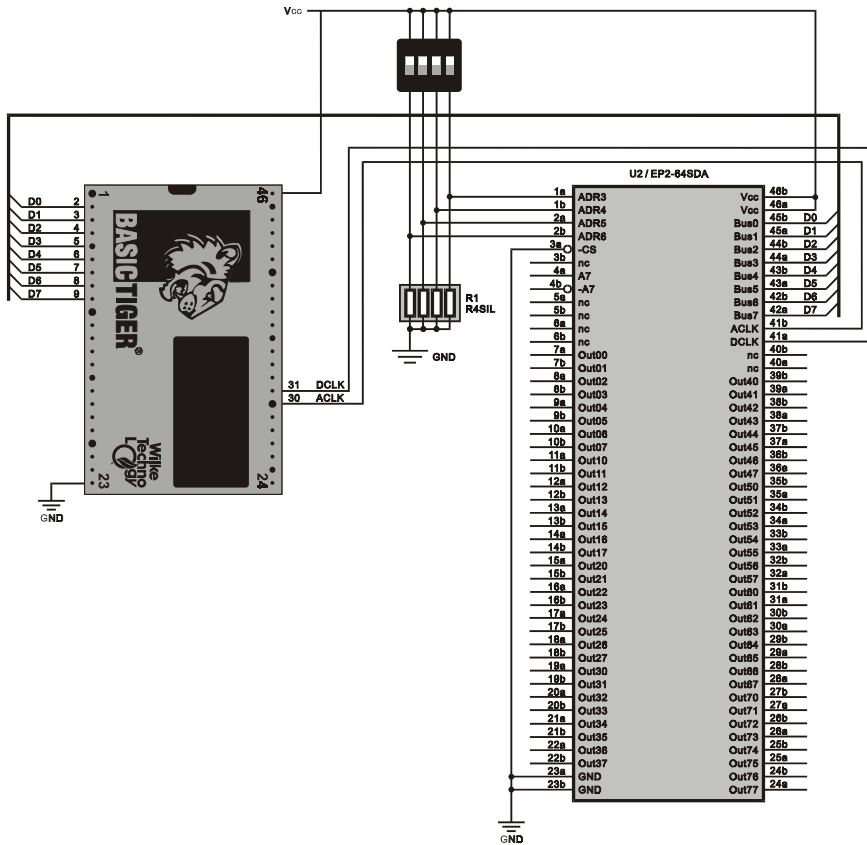
An den Leitungen ADR3...ADR6 wird die Basis-Adresse des Erweiterungsmoduls angelegt. Die erweiterten Ports belegen 8 Adressen ab der Basis-Adresse. Das Modul wird adressiert, wenn '-CS' auf 'low' liegt und eine Adresse im eingestellten Bereich angesprochen wird.

EP2-64SDA Adressierung						
-CE	ADR6	ADR5	ADR4	ADR3	Port-0	Port-7
1	x	x	x	x	—	—
0	0	0	0	0	0	7
0	0	0	0	1	8	0Fh
0	0	0	1	0	10h	17h
0	0	0	1	1	18h	1Fh
0	0	1	0	0	20h	27h
0	0	1	0	1	28h	2Fh
0	0	1	1	0	30h	37h
0	0	1	1	1	38h	3Fh
0	1	0	0	0	40h	47h
0	1	0	0	1	48h	4Fh
0	1	0	1	0	50h	57h
0	1	0	1	1	58h	5Fh
0	1	1	0	0	60h	67h
0	1	1	0	1	68h	6Fh
0	1	1	1	0	70h	77h
0	1	1	1	1	78h	7Fh

Da A7 intern nicht ausgewertet wird, spiegeln sich die Adressen ab Adresse 80h.

I/O-Erweiterungsmodule

Anschlußbeispiel EP2-64SDA:



Die Port-Adressen der erweiterten Outputs beginnen bei 0, wenn am DIP-Schalter die niedrigste Basisadresse eingestellt ist (alle DIP-Schalter OFF).

Programmbeispiel:

```

'-----
' Name: EP2-1.TIG
'-----
#INCLUDE DEFINE_A.INC          ' allgemeine Definitionen
USER_EPORT PHYSOFFS, 0F0h      ' Offset zu phys. Adr. -10h
USER_EPORT NROFOUT, 8          ' 8 erweiterte Output Ports
USER_EPORT INITIAL, 0, "&      ' initialisiere ab Adr. 0
00 01 02 03 04 05 06 07"%     ' mit dem Wert der phys. Adr.

TASK MAIN                      ' Beginn Task MAIN
    WAIT_DURATION 2000         ' 2 Sek. warten
    FOR I = 10h TO 17h         ' gebe auf Ports 0..7
        OUT I,255,I            ' logische Portadressen aus
    NEXT                       ' naechster Port
END                            ' Ende Task MAIN

```

4

Das Programm legt zunächst den Offset von logischer zu physikalischer Adresse (PHYSOFFS) sowie die Anzahl (NROFOUT) der zu Initialisierenden Output-Ports fest. Die Werte zur Initialisierung werden in der INITIAL-Anweisung definiert. Um die Start-Initialisierung sehen zu können, wartet das Programm eine Weile, bevor die Ports mit neuen Werten beschrieben werden.

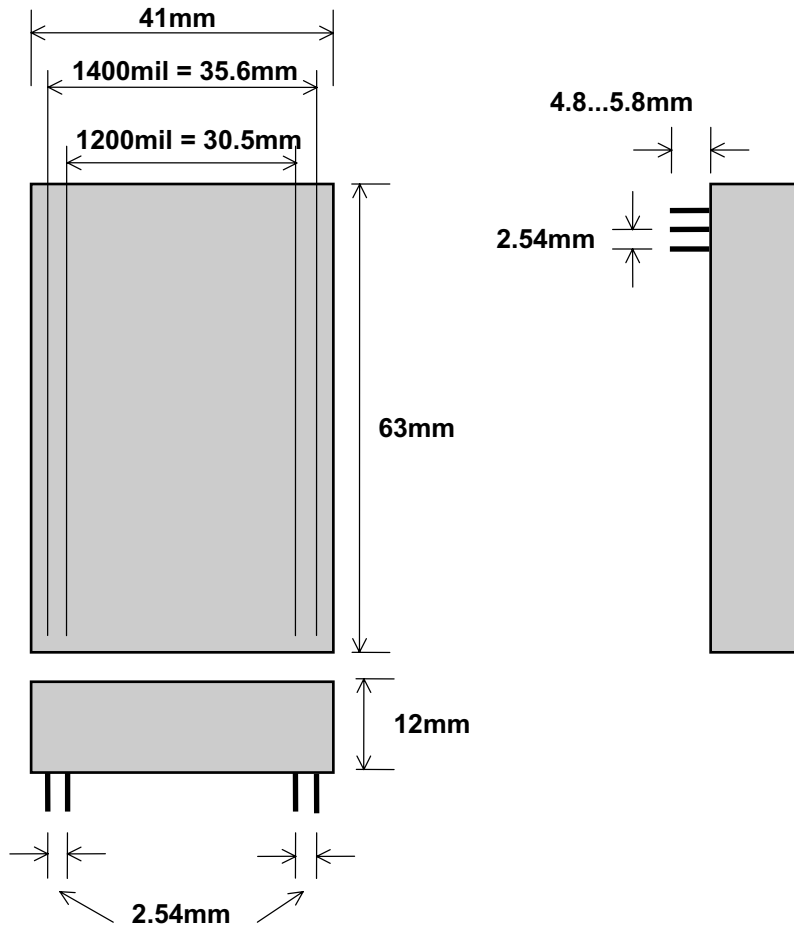
I/O-Erweiterungsmodule

Technische Daten des I/O-Erweiterungsmoduls EP2-64SDA:

Maße / Gewicht	ca. 63 x 41 x 12 mm / ca. 50g
Stromversorgung	4,7V...5,5V
Ruhestromaufnahme	ca. 120µA
Anzahl der erweiterten Ausgänge	64
max. Strom eines Outputs 0...5V	5mA
Temperaturbereich	-40 bis +85°C

4

Abmessungen EP2-64SDA:



I/O-Erweiterungsmodule

Leere Seite

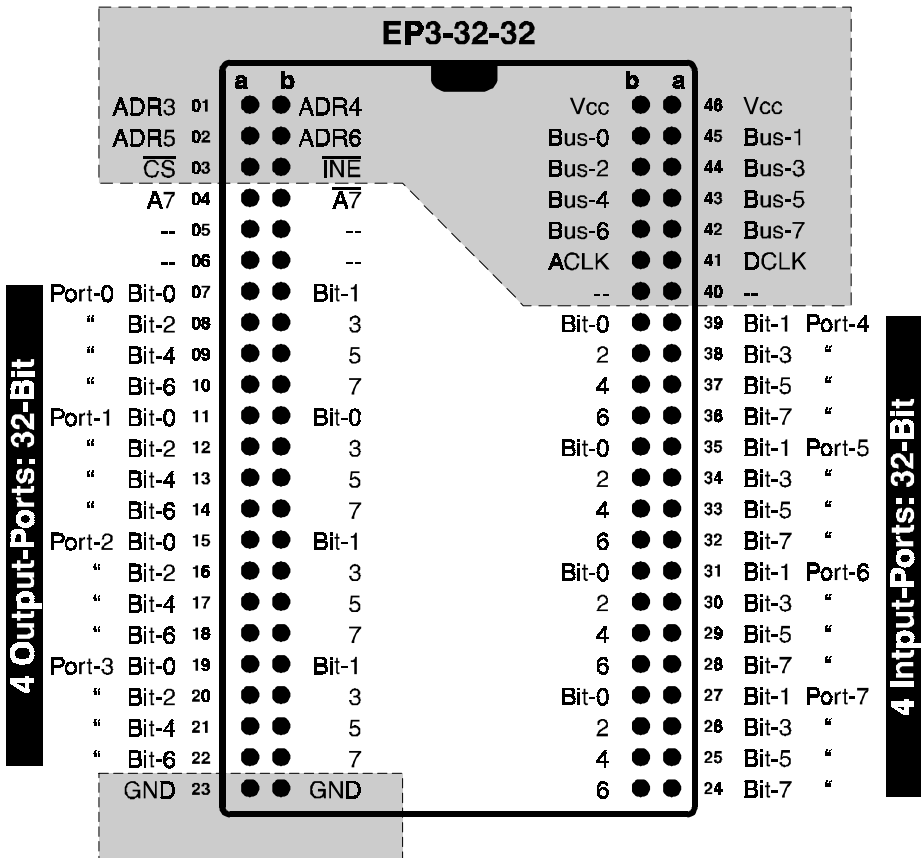
4

EP3-32-32

32 digitale Eingänge / 32 digitale Ausgänge

Das I/O-Erweiterungsmodul EP3-32-32 bietet 32 hochohmige digitale Inputs (0...5V, HC-MOS) sowie 32 digitale Outputs (0...5V, HC-MOS). Durch die einstellbare Basisadresse kann das Modul mit anderen Erweiterungen kombiniert werden.

Pinbelegung EP3-32-32



Pin-Beschreibung EP3-32-32

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
1a	ADR3	Adresse	1b	ADR4	AdressBit 4
2a	ADR5	Adresse	2b	ADR6	AdressBit 6
3a	-CS	ChipSelect	3b	-INE	In Enable
4a	A7	Adressbit 7	4b	-A7	Neg.. Adressbit 7
5a			5b		
6a			6b		
7a	P0.0	Out0 Bit0	7b	P0.1	Port0 Bit1
8a	P0.2	Out0 Bit2	8b	P0.3	Port0 Bit3
9a	P0.4	Out0 Bit4	9b	P0.5	Port0 Bit5
10a	P0.6	Out0 Bit6	10b	P0.7	Port0 Bit7
11a	P1.0	Out1 Bit0	11b	P1.1	Out1 Bit1
12a	P1.2	Out1 Bit2	12b	P1.3	Out1 Bit3
13a	P1.4	Out1 Bit4	13b	P1.5	Out1 Bit5
14a	P1.6	Out1 Bit6	14b	P1.7	Out1 Bit7
15a	P2.0	Out2 Bit0	15b	P2.1	Out2 Bit5
16a	P2.2	Out2 Bit2	16b	P2.3	Out2 Bit3
17a	P2.4	Out2 Bit4	17b	P2.5	Out2 Bit1
18a	P2.6	Out2 Bit6	18b	P2.7	Out2 Bit7
19a	P3.0	Out3 Bit0	19b	P3.1	Out3 Bit1
20a	P3.2	Out3 Bit2	20b	P3.3	Out3 Bit3
21a	P3.4	Out3 Bit4	21b	P3.5	Out3 Bit5
22a	P3.6	Out3 Bit6	22b	P3.7	Out3 Bit7
23a	GND	Masse	23b	GND	Masse

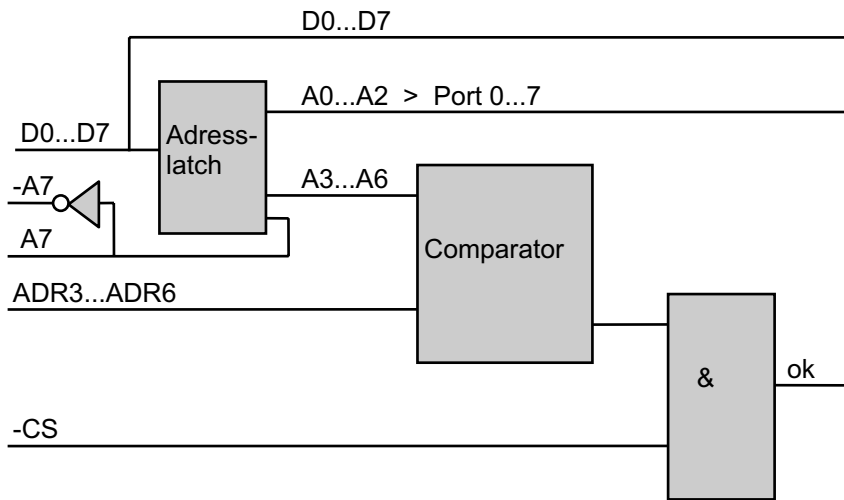
Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
24a	P7.7	In7 Bit7	24b	P7.6	In7 Bit6
25a	P7.5	In7 Bit5	25b	P7.4	In7 Bit4
26a	P7.3	In7 Bit3	26b	P7.2	In7 Bit2
27a	P7.1	In7 Bit1	27b	P7.0	In7 Bit0
28a	P6.7	In6 Bit7	28b	P6.6	In6 Bit6
29a	P6.5	In6 Bit5	29b	P6.4	In6 Bit4
30a	P6.3	In6 Bit3	30b	P6.2	In6 Bit2
31a	P6.1	In6 Bit1	31b	P6.0	In6 Bit0
32a	P5.7	In5 Bit7	32b	P5.6	In5 Bit6
33a	P5.5	In5 Bit5	33b	P5.4	In5 Bit4
34a	P5.3	In5 Bit3	34b	P5.2	In5 Bit2
35a	P5.1	In5 Bit1	35b	P5.0	In5 Bit0
36a	P4.7	In4 Bit7	36b	P4.6	In4 Bit6
37a	P4.5	In4 Bit5	37b	P4.4	In4 Bit4
38a	P4.3	In4 Bit3	38b	P4.2	In4 Bit2
39a	P4.1	In4 Bit1	39b	P4.0	In4 Bit0
40a			40b		
41a	DCLK	Datenclock	41b	ACLK	Adressclock
42a	Bus-7	I/O-Bus Bit7	42b	Bus-6	I/O-Bus Bit6
43a	Bus-5	I/O-Bus Bit5	43b	Bus-4	I/O-Bus Bit4
44a	Bus-3	I/O-Bus Bit3	44b	Bus-2	I/O-Bus Bit2
45a	Bus-1	I/O-Bus Bit1	45b	Bus-0	I/O-Bus Bit0
46a	VCC		46b	VCC	

Adressierung des EP3-32-32

Das Modul EP3-32-32 wird direkt von dem Tiger-BASIC[®] EPORT-System unterstützt. Der Anschluß an ein BASIC-Tiger[®]-Modul A oder Tiny-Tiger[®]-Modul erfordert bei der Standard-Pinbenutzung folgende Pins:

Pin-Bezeichnung	BASIC-Tiger [®] -Pins	Modultyp-A Pin-Nr.	Tiny-Tiger Pin-Nr.
Bus-0...Bus-7	L60...L67	2...9	1...8
ACLK (Adressclock)	L33	30	29
DCLK (Datenclock)	L34	31	30
-INE (in-enable)	L35	32	31
ADR3...ADR6	A3...A6 Inputs für Basis-Adresse des Moduls		
-CS	Chip-Select Input, low aktiv		
weitere Pins			
A7	A7-Ausgang des internen Adreßlatch		
-A7	A7-Ausgang invertiert		
Port-m Bit-n	Ports (m=0...3) mit je 8 Bits (n=0...7) Outputs Ports (m=4...7) mit je 8 Bits (n=0...7) Inputs		

Blockschaltbild der Adressierung des EP3-32-32:



4

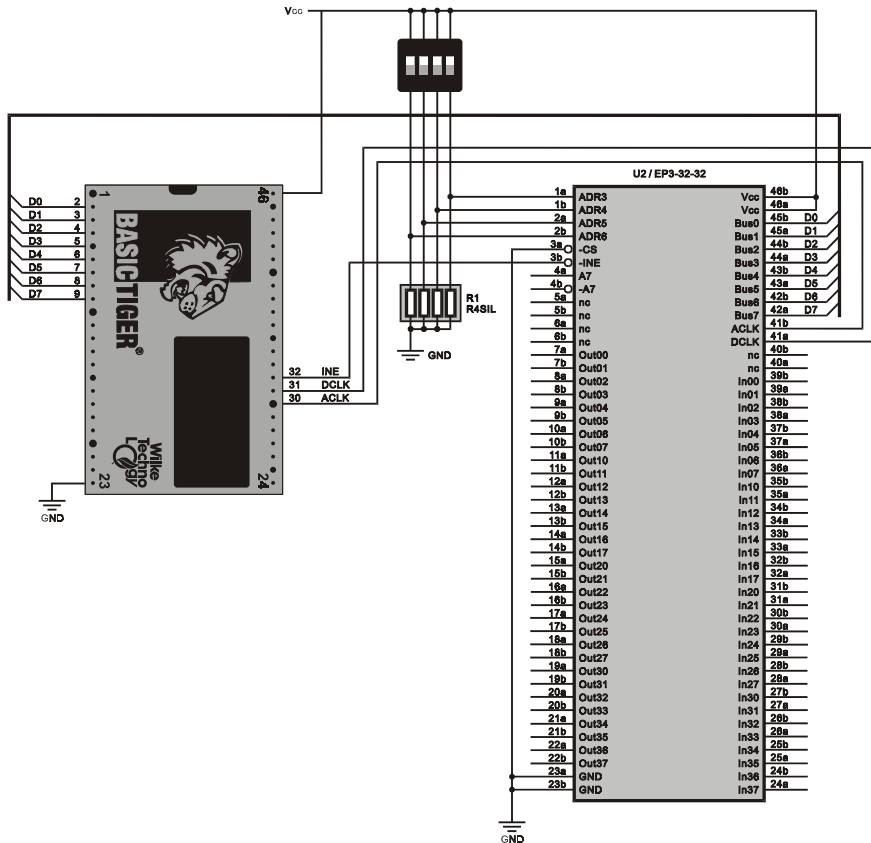
I/O-Erweiterungsmodule

An den Leitungen ADR3...ADR6 wird die Basis-Adresse des Erweiterungsmoduls angelegt. Die erweiterten Ports belegen 8 Adressen ab der Basis-Adresse. Die Input-adressen folgen unmittelbar auf die Outputadressen.

EP3-32-32 Adressierung						
-CE	ADR6	ADR5	ADR4	ADR3	Port-0	Port-7
1	x	x	x	x	—	—
0	0	0	0	0	0	7
0	0	0	0	1	8	0Fh
0	0	0	1	0	10h	17h
0	0	0	1	1	18h	1Fh
0	0	1	0	0	20h	27h
0	0	1	0	1	28h	2Fh
0	0	1	1	0	30h	37h
0	0	1	1	1	38h	3Fh
0	1	0	0	0	40h	47h
0	1	0	0	1	48h	4Fh
0	1	0	1	0	50h	57h
0	1	0	1	1	58h	5Fh
0	1	1	0	0	60h	67h
0	1	1	0	1	68h	6Fh
0	1	1	1	0	70h	77h
0	1	1	1	1	78h	7Fh

Da A7 intern nicht ausgewertet wird, spiegeln sich die Adressen ab Adresse 80h.

Anschlußbeispiel EP3-32-32:



4

Der A7-Ausgang ist in diesem Beispiel mit '-CS' verbunden, wodurch die Adressen der erweiterten Ports bei 80h beginnen, wenn am DIP-Schalter die niedrigste Basisadresse eingestellt ist (alle DIP-Schalter OFF).

I/O-Erweiterungsmodule

Programmbeispiel:

```
'-----
' Name: EP3-1.TIG
'-----
#include DEFINE_A.INC                ' allgemeine Definitionen
USER_EPORT PHYSOFFS, 0F0h           ' Offset zu phys. Adr. -10h
USER_EPORT NROFOUT, 4               ' 4 erweiterte Output Ports
USER_EPORT LASTLADR, 13h            ' letzte log. Outputadresse
USER_EPORT INITIAL, 0, "&          ' initialisiere ab Adr. 0
00 01 02 03"%                      ' mit dem Wert der phys. Adr.

TASK MAIN                          ' Beginn Task MAIN
' LCD-Treiber installieren (BASIC-Tiger)
INSTALL DEVICE #1, "LCD1.TDD"
' LCD-Treiber installieren (TINY-Tiger)
' INSTALL DEVICE #1, "LCD1.TDD", 0, 0, 0, 0, 0, 0, 80h, 8

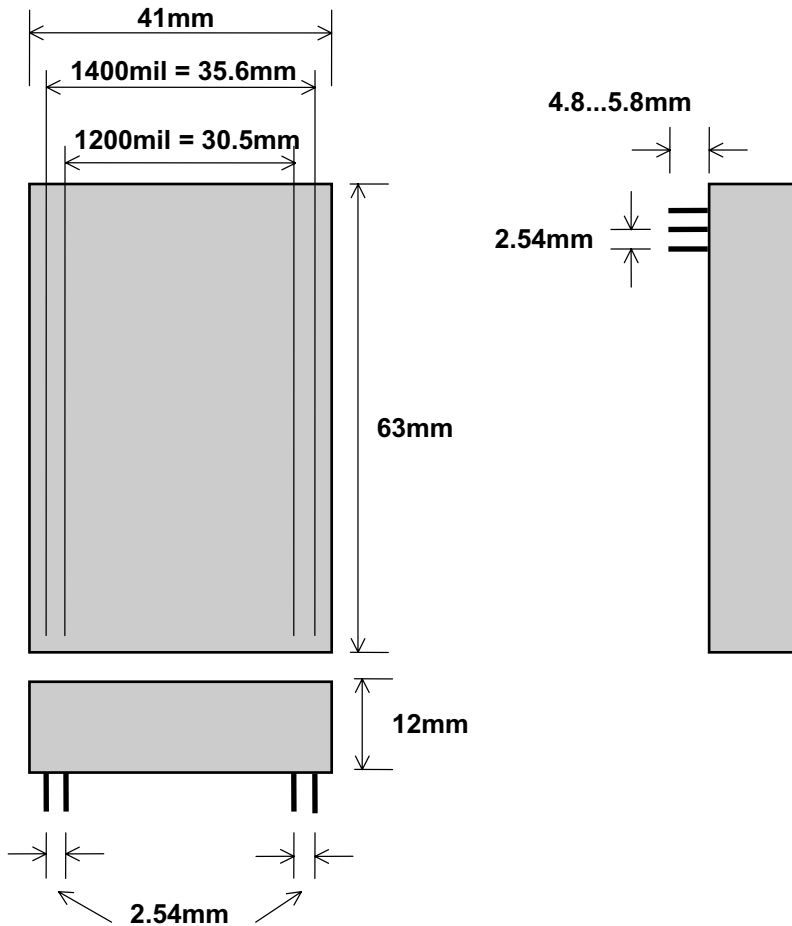
WAIT_DURATION 2000                  ' 2 Sek. warten
FOR I = 10h TO 13h                  ' gebe auf Ports 0..3
    OUT I,255,I                     ' logische Portadressen aus
NEXT                                ' naechster Port
FOR I = 14h TO 17h                  ' von Ports 4..7
    IN I,VALUE                       ' lies von Port
    PRINT #1, "Port";I-10h;"=";VALUE ' Wert ausgeben
    WAIT_DURATION 1000              ' 1 Sek. warten
NEXT                                ' naechster Port
END                                  ' Ende Task MAIN
```

Technische Daten des I/O-Erweiterungsmoduls EP3-32-32:

Maße / Gewicht	ca. 63 x 41 x 12 mm / ca. 50g
Stromversorgung	4,7V...5,5V
Ruhestromaufnahme	ca. 120µA
Anzahl der erweiterten Ausgänge	32
max. Strom eines Outputs 0...5V	5mA
Anzahl der erweiterten Eingänge	32
Eingangsspannung	0...5V (HC-MOS-Eingang)
Temperaturbereich	-40 bis +85°C

I/O-Erweiterungsmodule

Abmessungen EP3-32-32:

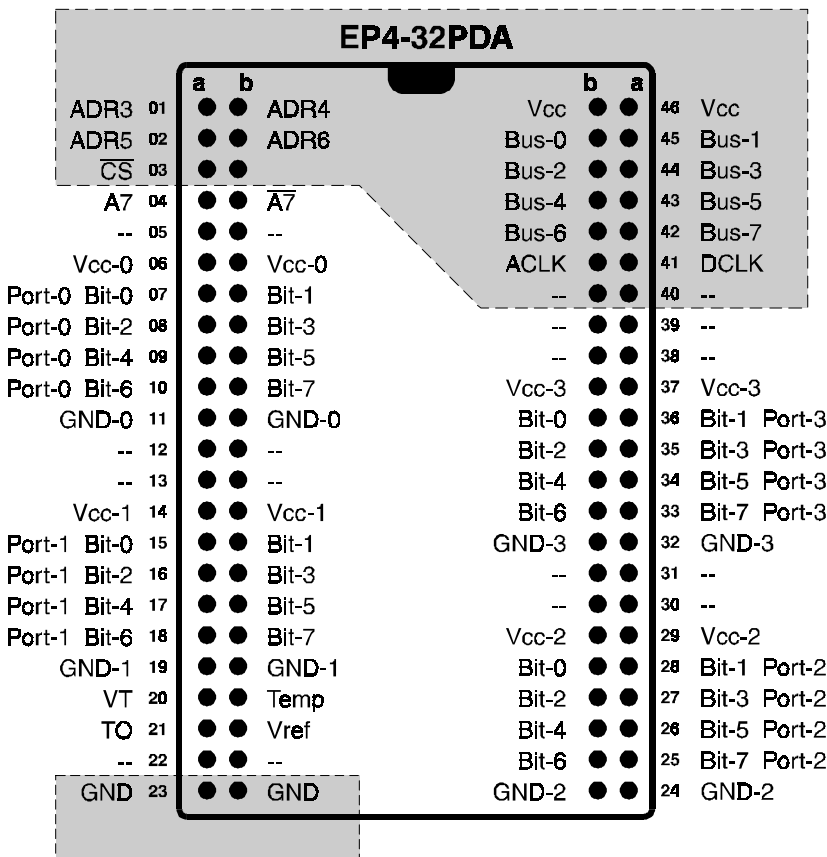


EP4-32PDA

32 Open-Collector Ausgänge

Das I/O-Erweiterungsmodul EP4-32PDA bietet 32 Open-Collector-Ausgänge (0...max. 50V, max. 500mA). Durch die einstellbare Adresse kann das Modul mit anderen Erweiterungen kombiniert werden. Auf dem Modul befindet sich ein Temperatursensor, der auf verschiedene Weise eingesetzt werden kann, um das Modul vor übermäßiger Erwärmung zu schützen.

Pinbelegung EP4-32PDA



Pin-Beschreibung EP4-32PDA

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
1a	ADR3	Adresse	1b	ADR4	AdressBit 4
2a	ADR5	Adresse	2b	ADR6	AdressBit 6
3a	-CS	ChipSelect	3b		
4a	A7	Adressbit 7	4b	-A7	Adressbit7 (negiert)
5a			5b		
6a	VCC-0	Spannung	6b	VCC-0	Spannung
7a	P0.0	Port 0 Bit0	7b	P0.1	Port0 Bit1
8a	P0.2	Port0 Bit2	8b	P0.3	Port0 Bit3
9a	P0.4	Port0 Bit4	9b	P0.5	Port0 Bit5
10a	P0.6	Port0 Bit6	10b	P0.7	Port0 Bit7
11a	GND-0	Masse	11b	GND-0	Masse
12a			12b		
13a			13b		
14a	VCC-1	Spannung	14b	VCC-1	Spannung
15a	P1.0	Port1 Bit0	15b	P1.1	Port1 Bit1
16a	P1.2	Port1 Bit2	16b	P1.3	Port1 Bit3
17a	P1.4	Port1 Bit4	17b	P1.5	Port1 Bit5
18a	P1.6	Port1 Bit6	18b	P1.7	Port1 Bit7
19a	GND-1	Masse	19b	GND-1	Masse
20a	VT	Temp.schwelle	20b	Temp	Temp. analog Out
21a	TO	Temp.-Schalt.	21b	Verf	Referenz Out
22a			22b		
23a	GND	Masse	23b	GND	Masse

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
24a	GND-2	Masse	24b	GND-2	Masse
25a	P2.7	Port2 Bit7	25b	P2.6	Port2 Bit6
26a	P2.5	Port2 Bit5	26b	P2..4	Port2 Bit4
27a	P2.3	Port2 Bit3	27b	P2.2	Port2 Bit2
28a	P2.1	Port2 Bit1	28b	P2.0	Port2 Bit0
29a	VCC-2	Spannung	29b	VCC-2	Spannung
30a			30b		
31a			31b		
32a	GND-3	Masse	32b	GND-3	Masse
33a	P3.7	Port3 Bit7	33b	P3.6	Port3 Bit6
34a	P3.5	Port3 Bit5	34b	P3.4	Port3 Bit4
35a	P3.3	Port3 Bit3	35b	P3.2	Port3 Bit2
36a	P3.1	Port3 Bit1	36b	P3.0	Port3 Bit0
37a	VCC-3		37b	VCC-3	
38a			38b		
39a			39b		
40a			40b		
41a	DCLK	Datenclock	41b	ACLK	Adressclock
42a	Bus-7	I/O-Bus Bit7	42b	Bus-6	I/O-Bus Bit6
43a	Bus-5	I/O-Bus Bit5	43b	Bus-4	I/O-Bus Bit4
44a	Bus-3	I/O-Bus Bit3	44b	Bus-2	I/O-Bus Bit2
45a	Bus-1	I/O-Bus Bit1	45b	Bus-0	I/O-Bus Bit0
46a	VCC	+5V	46b	VCC	+5V

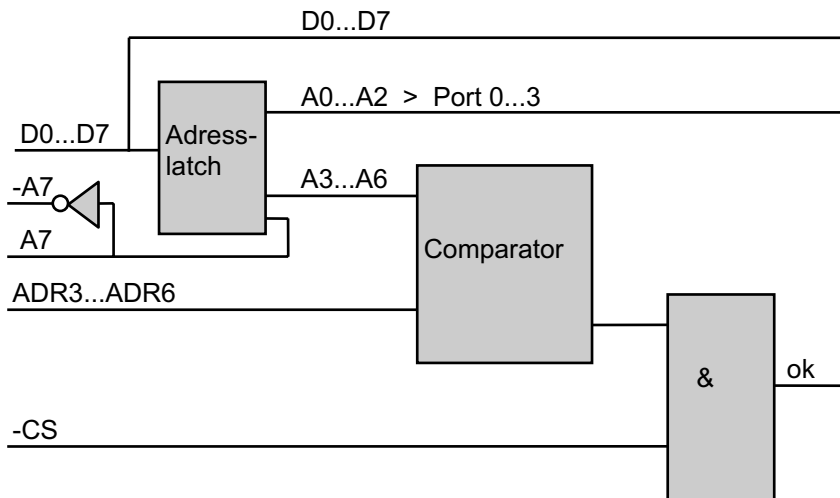
Beachte: GND-0...GND-3 sind mit GND verbunden. Jedoch sollten alle **GND-n eigene Zuleitungen** auf Platine bekommen.

Adressierung EP4-32PDA

Das Modul EP4-32PDA wird direkt von dem Tiger-BASIC[®] EPORT-System unterstützt. Der Anschluß an ein BASIC-Tiger[®]-Modul A oder Tiny-Tiger[®]-Modul erfordert bei der Standard-Pinbenutzung folgende Pins:

Pin-Bezeichnung	BASIC-Tiger [®] -Pins	Modultyp-A Pin-Nr.	Tiny-Tiger Pin-Nr.
Bus-0...Bus-7	L60...L67	2...9	1...8
ACLK (Adressclock)	L33	30	29
DCLK (Datenclock)	L34	31	30
ADR3...ADR6	A3...A6 Inputs für Basis-Adresse des Moduls		
-CS	Chip-Select Input, low aktiv		
weitere Pins			
A7	A7-Ausgang des internen Adreßlatch		
-A7	A7-Ausgang invertiert		
Port-m Bit-n	Ports (m=0...3) mit je 8 Bits (n=0...7)		
VT	Eingang zur Beeinflussung der Schaltschwelle		
Temp	analoger Temperatur-Ausgang: 395mV + 6,2mV/°C		
TO	Open-Collector-Schaltausgang mit Hysterese, active low, 50µA		
Vref	Ausgang der Referenzspannung des Temperatursensors (1,25V)		

Blockschaltbild der Adressierung des EP4-32PDA:



4

I/O-Erweiterungsmodule

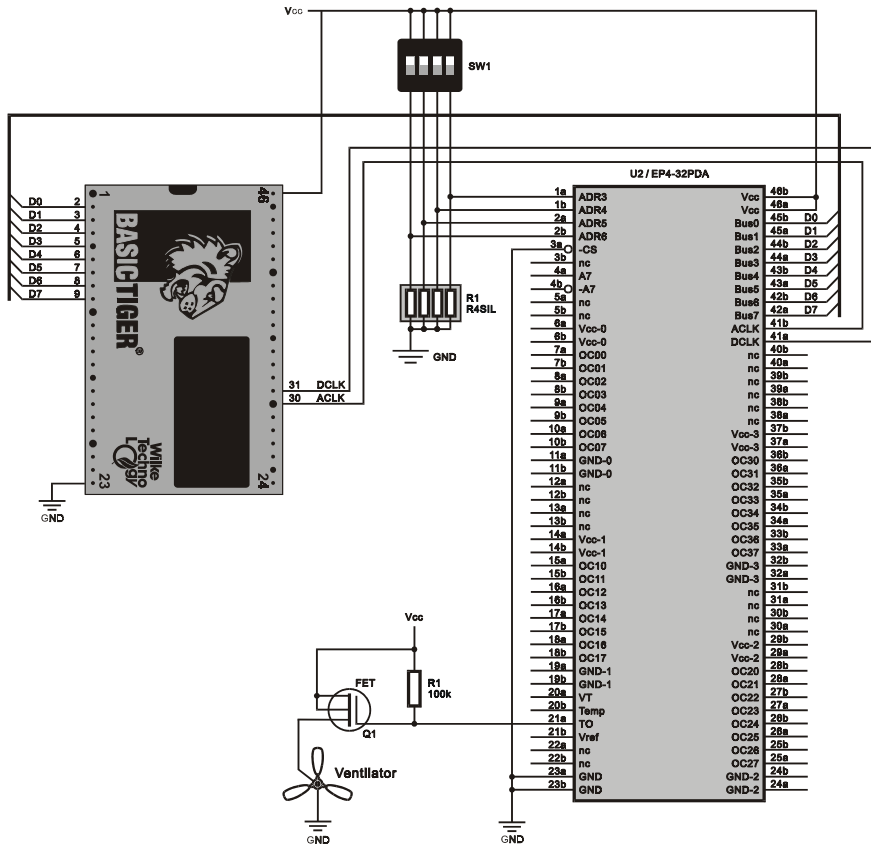
An den Leitungen ADR3...ADR6 wird die Basis-Adresse des Erweiterungsmoduls angelegt. Die erweiterten Ports belegen 8 Adressen ab der Basis-Adresse, es werden jedoch nur 4 Adressen verwendet. Das Modul wird adressiert, wenn '-CS' auf 'low' liegt und eine Adresse im eingestellten Bereich angesprochen wird.

EP4-32PDA Adressierung

-CE	ADR6	ADR5	ADR4	ADR3	Port-0	Port-3
1	x	x	x	x	—	—
0	0	0	0	0	0	3
0	0	0	0	1	8	0Bh
0	0	0	1	0	10h	13h
0	0	0	1	1	18h	1Bh
0	0	1	0	0	20h	23h
0	0	1	0	1	28h	2Bh
0	0	1	1	0	30h	33h
0	0	1	1	1	38h	3Bh
0	1	0	0	0	40h	43h
0	1	0	0	1	48h	4Bh
0	1	0	1	0	50h	53h
0	1	0	1	1	58h	5Bh
0	1	1	0	0	60h	63h
0	1	1	0	1	68h	6Bh
0	1	1	1	0	70h	73h
0	1	1	1	1	78h	7Bh

Da A7 intern nicht ausgewertet wird, spiegeln sich die Adressen ab Adresse 80h.

Anschlußbeispiel EP4-32PDA:



4

Die Port-Adressen des Moduls beginnen bei 0, wenn am DIP-Schalter die niedrigste Basisadresse eingestellt ist (alle DIP-Schalter OFF). Im Schaltungbeispiel schaltet der eingebaute Temperatursensor einen Ventilator ein, wenn die Temperatur die Schaltschwelle des eingebauten Temperatursensors übersteigt.

Temperatursensor EP4-32PDA

Wenn das Modul EP4-32PDA eingesetzt wird, um Leistungen im oberen Grenzbereich des Moduls zu schalten, kann sich das Modul sehr stark erwärmen, so daß Vorkehrungen zur Kühlung oder Reduzierung der Leistung getroffen werden müssen.

Die Leiterbahnführung des Layouts sollte bereits getrennte Versorgungswege für die port-eigenen Anschlüsse VCC-n und GND-n vorsehen, damit sich nicht alle Ströme an einigen wenigen Pins addieren.

Beachten Sie die maximal zulässige Verlustleistung eines Ports (0,4W), die nicht die Summe der zulässigen Verlustleistungen einzelner Pins (ebenfalls ca. 0,4W) entspricht.

Höhere Stöme lassen sich durch Parallelschalten mehrerer Ports erreichen. Dabei ist auf eine gleichmäßige Stromverteilung zu achten (Widerstände in den Stromzweigen).

Der interne Temperatursensor besitzt einen Schaltausgang, mit dessen Hilfe ein Ventilator geschaltet werden kann. Die Schaltschwelle des Temperatursensors ist durch externe Widerstände am Pin 'VT' beeinflussbar. Vorschlag: montieren Sie einen Pentium-Prozessor-Ventilator auf dem Modul.

Denkbar ist auch, daß die steuernde Applikation den analogen Temperatúrausgang beobachtet und dynamisch anhand der gemessenen Erwärmung die Schaltleistung drosselt sowie Warn- oder Störmeldungen absetzt.

Programmbeispiel:

```

'-----
' Name: EP4-1.TIG
'-----
#INCLUDE DEFINE_A.INC          ' allgemeine Definitionen
USER_EPORT PHYSOFFS, 0F0h      ' Offset zu phys. Adr. -10h
USER_EPORT NROFOUT, 4          ' 4 erweiterte Output Ports
USER_EPORT INITIAL, 0, "&      ' initialisiere ab Adr. 0
00 01 02 03"%                  ' mit dem Wert der phys. Adr.

TASK MAIN                      ' Beginn Task MAIN
  WAIT_DURATION 2000           ' 2 Sek. warten
  FOR I = 10h TO 13h           ' gebe auf Ports 0..3
    OUT I,255,I                ' logische Portadressen aus
  NEXT                          ' naechster Port
END                             ' Ende Task MAIN

```

4

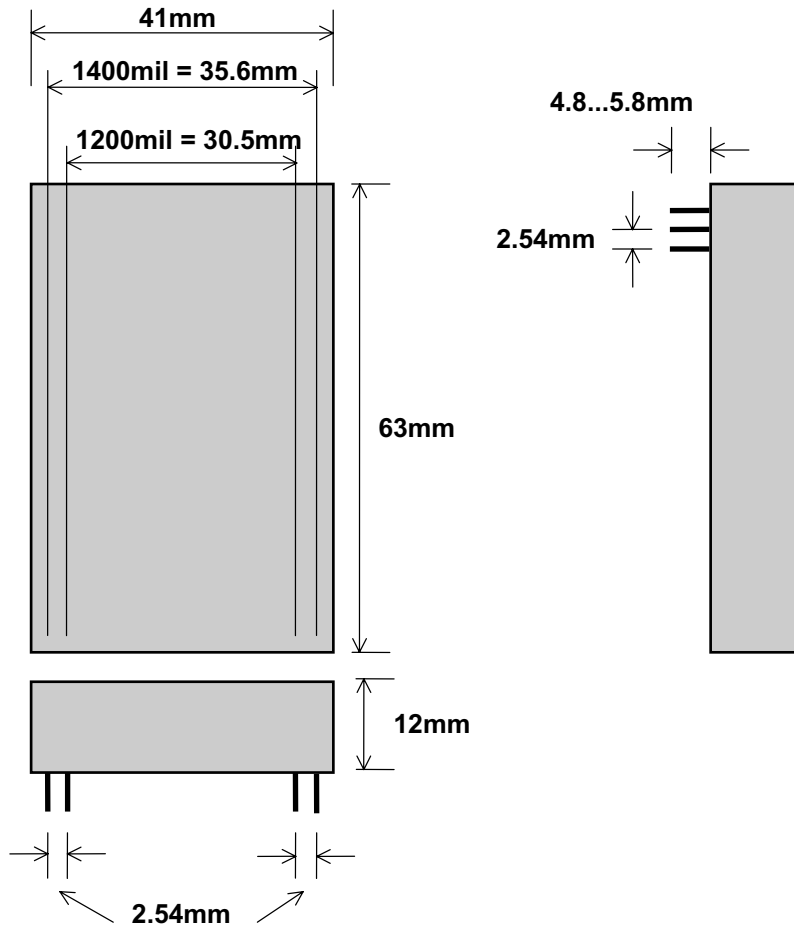
Das Programm legt zunächst den Offset von logischer zu physikalischer Adresse (PHYSOFFS) sowie die Anzahl (NROFOUT) der zu Initialisierenden Output-Ports fest. Die Werte zur Initialisierung werden in der INITIAL-Anweisung definiert. Um die Start-Initialisierung sehen zu können, wartet das Programm eine Weile, bevor die Ports mit neuen Werten beschrieben werden.

I/O-Erweiterungsmodule

Technische Daten des I/O-Erweiterungsmoduls EP4-32PDA:

Maße / Gewicht	ca. 63 x 41 x 12 mm / ca. 50g
Stromversorgung	4,7V...5,5V
Ruhestromaufnahme	ca. 12mA
Anzahl der erweiterten Ausgänge	32
abs. max. zulässiger Strom	500mA
max. Strom eines Outputs (DC)	350mA
max. Strom 8 Outputs (Duty 10%)	260mA
max. Strom 8 Outputs (Duty 50%)	90mA
max. Spannung OC-Outputs	50V
max. Verlustleistung eines Ports	0,4W
Temperaturbereich	-40 bis +85°C

Abmessungen EP4-32PDA:



I/O-Erweiterungsmodule

Leere Seite

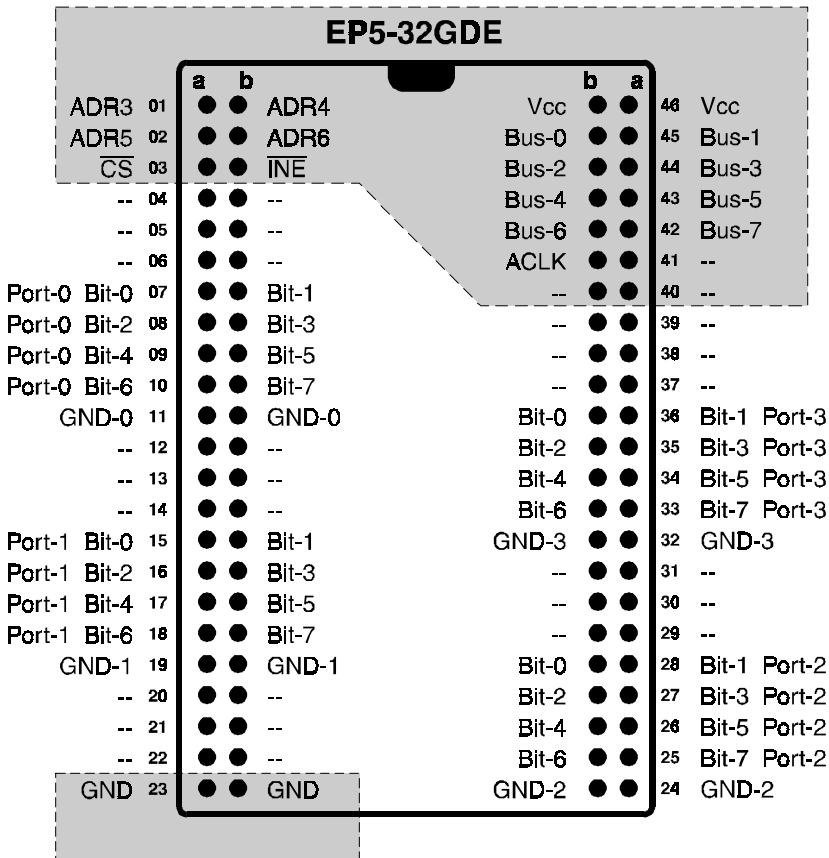
4

EP5-32GDE

32 Optokoppler Eingänge

Das I/O-Erweiterungsmodul EP5-32GDE bietet 32 opto-entkoppelte Inputs. Durch die einstellbare Adresse kann das Modul mit anderen Erweiterungen kombiniert werden.

Pinbelegung EP5-32GDE



Pin-Beschreibung EP5-32GDE

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
1a	ADR3	Adresse	1b	ADR4	AdressBit 4
2a	ADR5	Adresse	2b	ADR6	AdressBit 6
3a	-CS	ChipSelect	3b	-INE	In Enable
4a			4b		
5a			5b		
6a			6b		
7a	P0.0	Port0 Bit0	7b	P0.1	Port0 Bit1
8a	P0.2	Port0 Bit2	8b	P0.3	Port0 Bit3
9a	P0.4	Port0 Bit4	9b	P0.5	Port0 Bit5
10a	P0.6	Port0 Bit6	10b	P0.7	Port0 Bit7
11a	GND-0	Masse	11b	GND-0	Masse
12a			12b		
13a			13b		
14a			14b		
15a	P1.0	Port1 Bit0	15b	P1.1	Port1 Bit1
16a	P1.2	Port1 Bit2	16b	P1.3	Port1 Bit3
17a	P1.4	Port1 Bit4	17b	P1.5	Port1 Bit5
18a	P1.6	Port1 Bit6	18b	P1.7	Port1 Bit7
19a	GND-1	Masse	19b	GND-1	Masse
20a			20b		
21a			21b		
22a			22b		
23a	GND	Masse	23b	GND	Masse

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
24a	GND-2	Masse	24b	GND-2	Masse
25a	P2.7	Port2 Bit7	25b	P2.6	Port2 Bit6
26a	P2.5	Port2 Bit5	26b	P2.4	Port2 Bit4
27a	P2.3	Port2 Bit3	27b	P2.2	Port2 Bit2
28a	P2.1	Port2 Bit1	28b	P2.0	Port2 Bit0
29a			29b		
30a			30b		
31a			31b		
32a	GND-3	Masse	32b	GND-3	Masse
33a	P3.7	Port3 Bit7	33b	P3.6	Port3 Bit6
34a	P3.5	Port3 Bit5	34b	P3.4	Port3 Bit4
35a	P3.3	Port3 Bit3	35b	P3.2	Port3 Bit2
36a	P3.1	Port3 Bit1	36b	P3.0	Port3 Bit0
37a			37b		
38a			38b		
39a			39b		
40a			40b		
41a			41b	ACLK	Adressclock
42a	Bus-7	I/O-Bus Bit7	42b	Bus-6	I/O-Bus Bit6
43a	Bus-5	I/O-Bus Bit5	43b	Bus-4	I/O-Bus Bit4
44a	Bus-3	I/O-Bus Bit3	44b	Bus-2	I/O-Bus Bit2
45a	Bus-1	I/O-Bus Bit1	45b	Bus-0	I/O-Bus Bit0
46a	VCC	+5V	46b	VCC	+5V

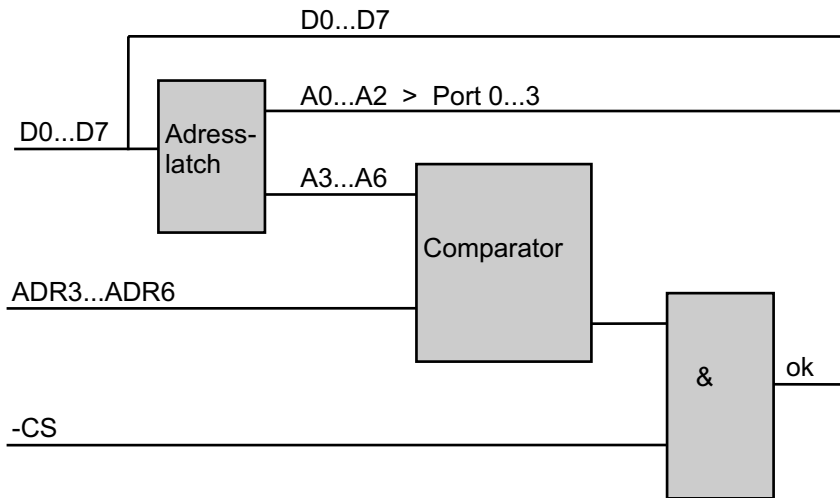
Adressierung des EP5-32GDE

Das Modul EP5-32GDE wird direkt von dem Tiger-BASIC[®] EPORT-System unterstützt. Der Anschluß an ein BASIC-Tiger[®]-Modul A oder Tiny-Tiger[®]-Modul erfordert bei der Standard-Pinbenutzung folgende Pins:

Pin-Bezeichnung	BASIC-Tiger [®] -Pins	Modultyp-A Pin-Nr.	Tiny-Tiger Pin-Nr.
Bus-0...Bus-7	L60...L67	2...9	1...8
ACLK (Adressclock)	L33	30	29
-INE (in-enable)	L35	32	31
ADR3...ADR6	A3...A6 Inputs für Basis-Adresse des Moduls		
-CS	Chip-Select Input, low aktiv		
weitere Pins			
Port-m Bit-n	Ports (m=0...3) mit je 8 Bits (n=0...7)		

Beachte: aus Gründen der Potentialtrennung fehlen einige Pins an den Modulen. Auch auf der Trägerplatine sollten die Abstände gewahrt bleiben. Kupfer-Pads, auch wenn sie nicht mit einer Buchsenleiste bestückt sind, bilden eine Brücke und verringern die Spannungsfestigkeit. Ebenso sollten Leiterbahnen und Durchkontaktierungen aus den Freiräumen ferngehalten werden.

Blockschaltbild der Adressierung des EP5-32GDE:



4

I/O-Erweiterungsmodule

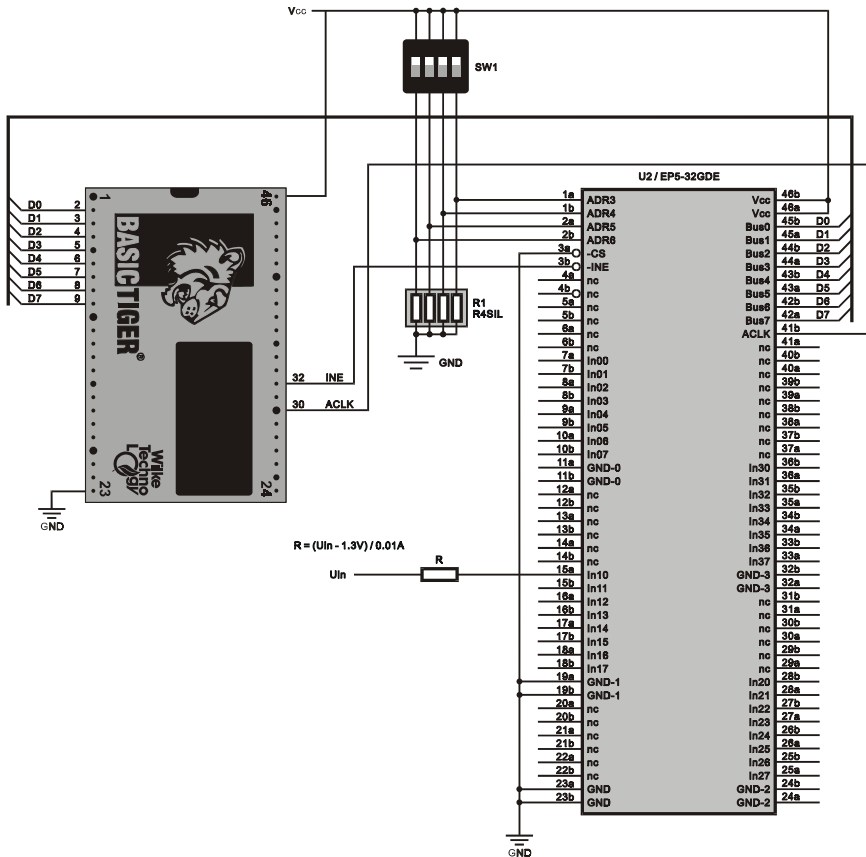
An den Leitungen ADR3...ADR6 wird die Basis-Adresse des Erweiterungsmoduls angelegt. Die erweiterten Input-Ports belegen 8 Adressen ab der Basis-Adresse, es werden jedoch nur 4 Adressen verwendet. Das Modul wird adressiert, wenn '-CS' auf 'low' liegt und eine Adresse im eingestellten Bereich angesprochen wird. Die Basisadresse kann um 80h angehoben werden, indem '-A7' mit '-CS' verbunden wird.

EP5-32GDE Adressierung

-CE	ADR6	ADR5	ADR4	ADR3	Port-0	Port-3
1	x	x	x	x	—	—
0	0	0	0	0	0	3
0	0	0	0	1	8	0Bh
0	0	0	1	0	10h	13h
0	0	0	1	1	18h	1Bh
0	0	1	0	0	20h	23h
0	0	1	0	1	28h	2Bh
0	0	1	1	0	30h	33h
0	0	1	1	1	38h	3Bh
0	1	0	0	0	40h	43h
0	1	0	0	1	48h	4Bh
0	1	0	1	0	50h	53h
0	1	0	1	1	58h	5Bh
0	1	1	0	0	60h	63h
0	1	1	0	1	68h	6Bh
0	1	1	1	0	70h	73h
0	1	1	1	1	78h	7Bh

Da A7 intern nicht ausgewertet wird, spiegeln sich die Adressen ab Adresse 80h.

Anschlußbeispiel EP5-32GDE:



4

Die Adressen des Moduls beginnen bei 0, wenn am DIP-Schalter die niedrigste Basisadresse eingestellt ist (alle DIP-Schalter OFF). Die Optokoppler-Eingänge benötigen ca. 10mA und haben keinen internen Vorwiderstand. Bei einer typischen Diodenflußspannung von 1,3V und einer Eingangsspannung U_{in} lautet die Berechnung des Vorwiderstandes:

$$R = U_{in} - 1,3V / 0,01A$$

I/O-Erweiterungsmodule

Programmbeispiel:

```
'-----
' Name: EP5-1.TIG
'-----
#include DEFINE_A.INC          ' allgemeine Definitionen
USER_EPORT PHYSOFFS, 0F0h     ' Offset zu phys. Adr. -10h

TASK MAIN                     ' Beginn Task MAIN
' LCD-Treiber installieren (BASIC-Tiger)
  INSTALL DEVICE #1, "LCD1.TDD"
' LCD-Treiber installieren (TINY-Tiger)
'  INSTALL DEVICE #1, "LCD1.TDD", 0, 0, 0, 0, 0, 0, 80h, 8

  FOR I = 90h TO 93h          ' von Ports 0..3
    IN I,VALUE                ' lies von Port
    PRINT #1, "Port";I-90h;"=";VALUE ' Wert ausgeben
    WAIT_DURATION 1000        ' 1 Sek. warten
  NEXT                        ' naechster Port
END                           ' Ende Task MAIN
```

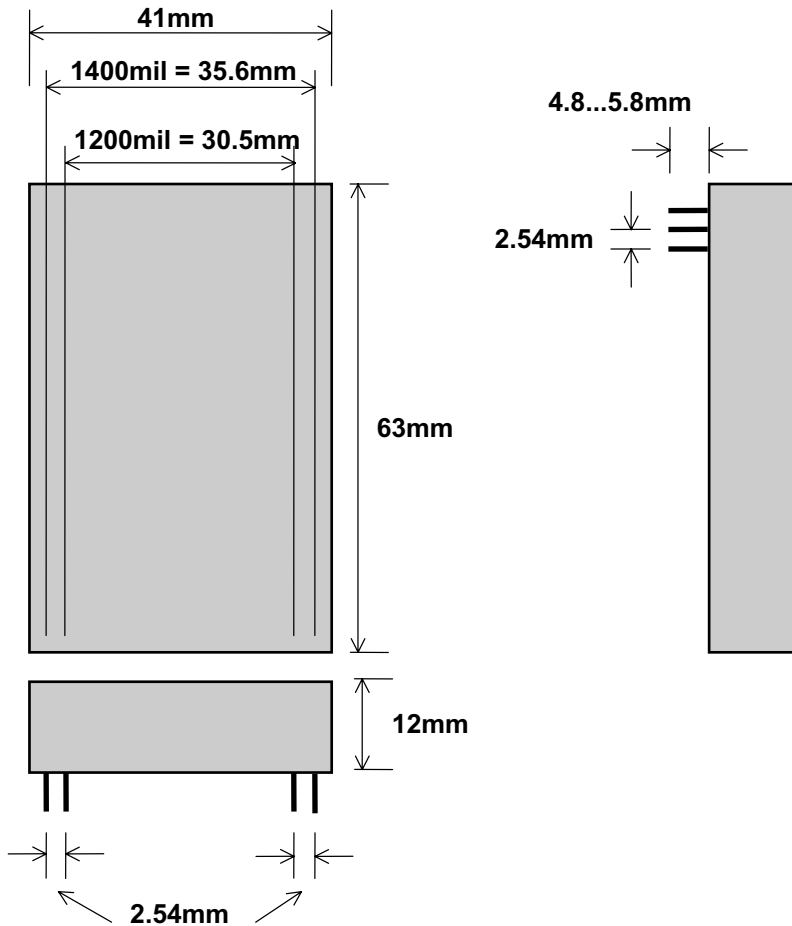
4

Technische Daten des I/O-Erweiterungsmoduls EP5-32GDE:

Maße / Gewicht	ca. 63 x 41 x 12 mm / ca. 50g
Stromversorgung	4,7V...5,5V
Ruhestromaufnahme	ca. 80µA
Anzahl der erweiterten Eingänge	32
Diodenstrom	5...50mA
Diodenflußspannung	typisch 1,3V
Temperaturbereich	-40 bis +85°C

I/O-Erweiterungsmodule

Abmessungen EP5-32GDE:

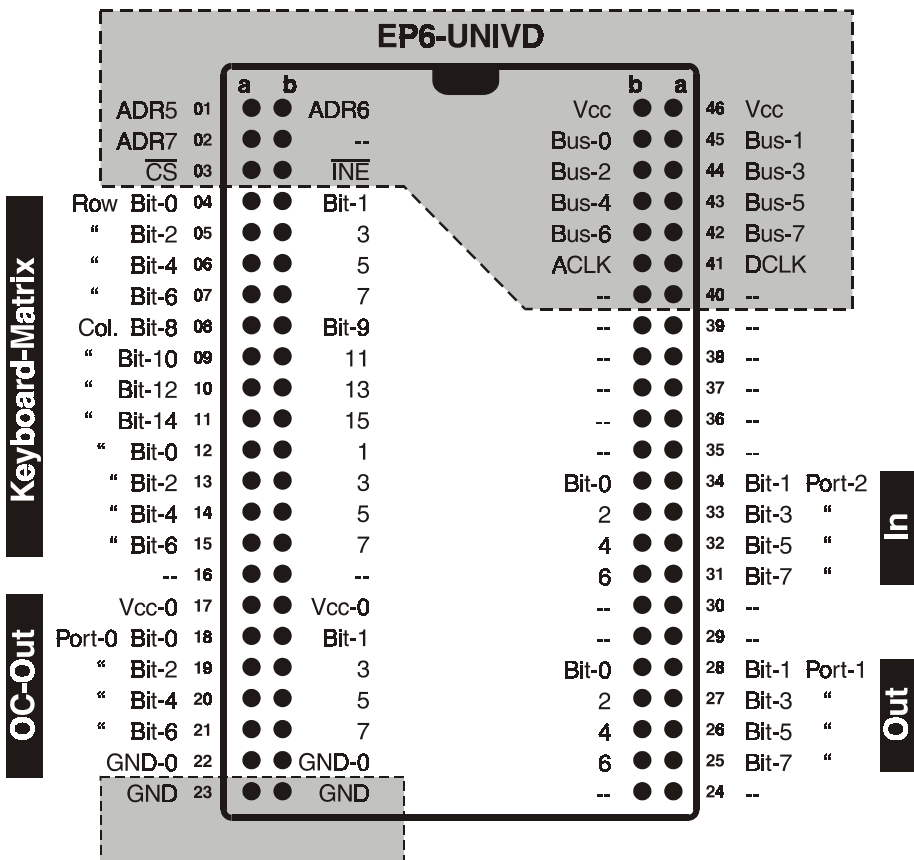


EP6-UNIVD

Universal I/O +128 Key

Das I/O-Erweiterungsmodul EP6-UNIVD bietet 8 digitale Inputs (0...5V), 8 digitale Outputs (0...5V, 5mA), 8 Open-Collector-Outputs (0...50V, max. 500mA) sowie einen Tastaturanschluß mit bis zu 128 Tasten oder DIP-Schaltern. Durch die einstellbare Adresse kann das Modul mit anderen Erweiterungen kombiniert werden.

Pinbelegung EP6-UNIVDE



Pin-Beschreibung EP6-UNIVD

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
1a	ADR5	Adresse	1b	ADR6	AdressBit 4
2a	ADR7	Adresse	2b		
3a	-CS	ChipSelect	3b	-INE	In Enable
4a	Zeile Bit0	Tastaturzeile	4b	Zeile Bit1	Tastaturzeile
5a	Zeile Bit2	Tastaturzeile	5b	Zeile Bit3	Tastaturzeile
6a	Zeile Bit4	Tastaturzeile	6b	Zeile Bit5	Tastaturzeile
7a	Zeile Bit6	Tastaturzeile	7b	Zeile Bit7	Tastaturzeile
8a	Spalte Bit0	Tastaturspalte	8b	Spalte Bit1	Tastaturspalte
9a	Spalte Bit2	Tastaturspalte	9b	Spalte Bit3	Tastaturspalte
10a	Spalte Bit4	Tastaturspalte	10b	Spalte Bit5	Tastaturspalte
11a	Spalte Bit6	Tastaturspalte	11b	Spalte Bit7	Tastaturspalte
12a	Spalte Bit8	Tastaturspalte	12b	Spalte Bit9	Tastaturspalte
13a	Spalte Bit10	Tastaturspalte	13b	Spalte Bit11	Tastaturspalte
14a	Spalte Bit12	Tastaturspalte	14b	Spalte Bit13	Tastaturspalte
15a	Spalte Bit14	Tastaturspalte	15b	Spalte Bit15	Tastaturspalte
16a			16b		
17a	VCC-0	Spannung	17b	VCC-0	Spannung
18a	P0.0	Out0 Bit0	18b	P0.1	Out0 Bit1
19a	P0.2	Out0 Bit2	19b	P0.3	Out0 Bit3
20a	P0.4	Out0 Bit4	20b	P0.5	Out0 Bit5
21a	P0.6	Out0 Bit6	21b	P0.7	Out0 Bit7
22a	GND-0	Masse	22b	GND-0	Masse
23a	GND	Masse	23b	GND	Masse

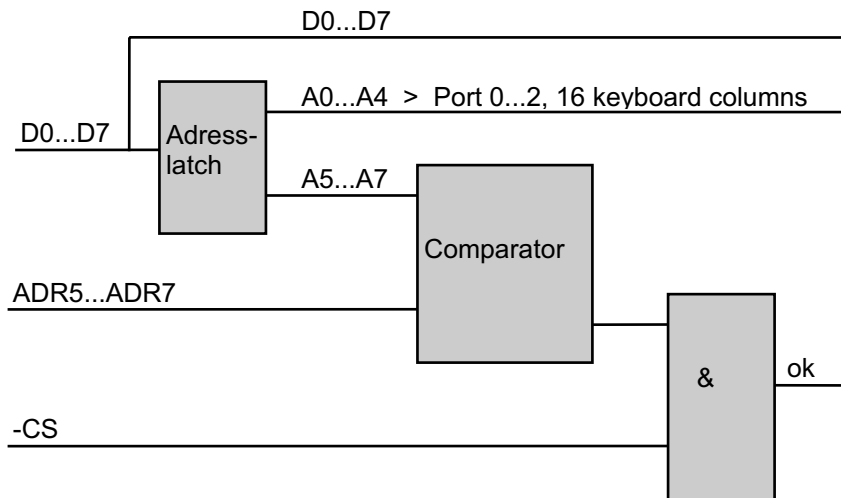
Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
24a			24b		
25a	P1.7	Out1 Bit7	25b	P1.6	Out1 Bit6
26a	P1.5	Out1 Bit5	26b	P1.4	Out1 Bit4
27a	P1.3	Out1 Bit3	27b	P1.2	Out1 Bit2
28a	P1.1	Out1 Bit1	28b	P1.0	Out1 Bit0
29a			29b		
30a			30b		
31a	P2.7	In2 Bit7	31b	P2.6	In2 Bit6
32a	P2.5	In2 Bit5	32b	P2.4	In2 Bit4
33a	P2.3	In2 Bit3	33b	P2.2	In2 Bit2
34a	P2.1	In2 Bit1	34b	P2.0	In2 Bit0
35a			35b		
36a			36b		
37a			37b		
38a			38b		
39a			39b		
40a			40b		
41a	DCLK	Datenclock	41b	ACLK	Adressclock
42a	Bus-7	I/O-Bus Bit7	42b	Bus-6	I/O-Bus Bit6
43a	Bus-5	I/O-Bus Bit5	43b	Bus-4	I/O-Bus Bit4
44a	Bus-3	I/O-Bus Bit3	44b	Bus-2	I/O-Bus Bit2
45a	Bus-1	I/O-Bus Bit1	45b	Bus-0	I/O-Bus Bit0
46a	VCC	+5V	46b	VCC	+5V

Adressierung des EP6-UNIVD

Das Modul EP6-UNIVD wird direkt von dem Tiger-BASIC[®] EPORT-System unterstützt. Der Anschluß an ein BASIC-Tiger[®]-Modul A oder Tiny-Tiger[®]-Modul erfordert bei der Standard-Pinbenutzung folgende Pins:

Pin-Bezeichnung	BASIC-Tiger [®] -Pins	Modultyp-A Pin-Nr.	Tiny-Tiger Pin-Nr.
Bus-0...Bus-7	L60...L67	2...9	1...8
ACLK (Adressclock)	L33	30	29
DCLK (Datenclock)	L34	31	30
-INE (in-enable)	L35	32	31
ADR5...ADR7	A5...A7 Inputs für Basis-Adresse des Moduls		
-CS	Chip-Select Input, low aktiv		
weitere Pins			
Row Bit-n	Tastaturzeilen (n=0...7)		
Col Bit-n	Tastaturspalten (n=0...15)		
Port-0 Bit-n	Open-Collector-Outout mit 8 Bits (n=0...7)		
Port-1 Bit-n	digitaler Output-Ports mit 8 Bits (n=0...7)		
Port-2 Bit-n	digitaler Input-Port mit 8 Bits (n=0...7)		

Blockschaltbild der Adressierung des EP6-UNIVD:



4

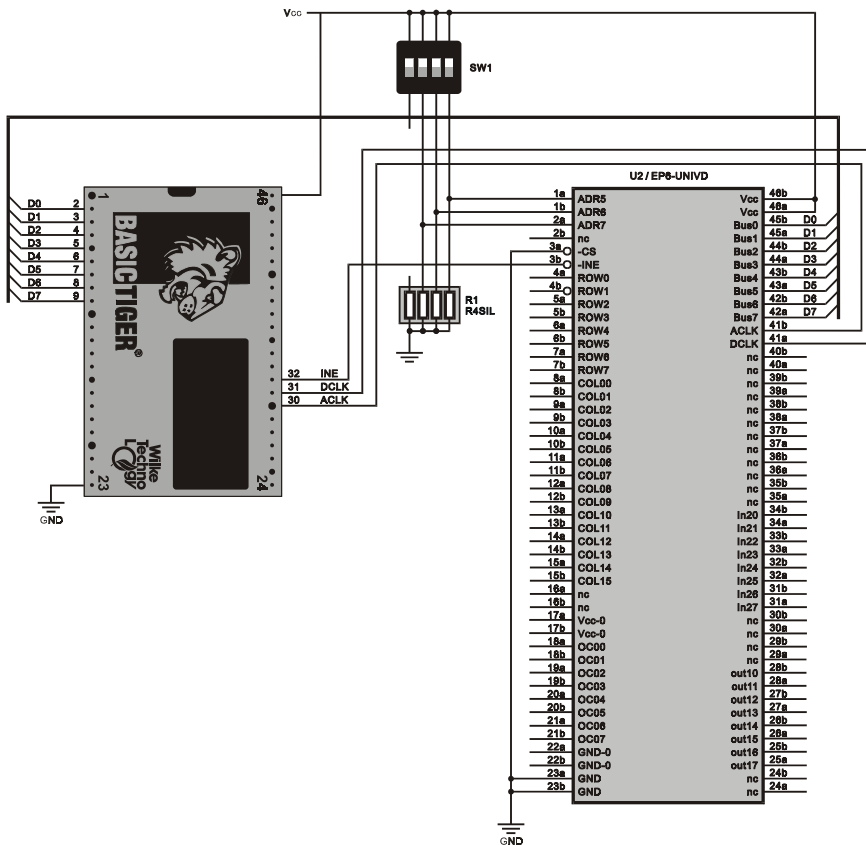
I/O-Erweiterungsmodule

An den Leitungen ADR5...ADR7 wird die Basis-Adresse des Erweiterungsmoduls angelegt. Das Modul belegt 32 Adressen ab der Basis-Adresse und wird adressiert, wenn '-CS' auf 'low' liegt und eine Adresse im eingestellten Bereich angesprochen wird.

EP6-UNIVD Adressierung						
-CE	ADR7	ADR6	ADR5	Port-0, 1 (Out)	Port-2 (In)	Keyboard-Spalte
1	x	x	x	—	—	—
0	0	0	0	0, 1	2	4-13h
0	0	0	1	20h, 21h	22h	24h-33h
0	0	1	0	40h, 41h	42h	44h-53h
0	0	1	1	60h, 61h	62h	64h-73h
0	1	0	0	80h, 81h	82h	84h-93h
0	1	0	1	0A0h, 0A1h	0A2h	0A4h-0B3h
0	1	1	0	0C0h, 0C1h	0C2h	0C4h-0D3h
0	1	1	1	0E0h, 0E1h	0E2h	0E4h-0F3h

Zunächst werden die beiden Output-Ports adressiert, dann der Input-Port. Nach einer unbelegten Adresse folgen die Tasturspalten. Die übrigen Adressen im Modul sind unbelegt.

Anschlußbeispiel EP6-UNIVD:



4

Die Port-Adressen des Moduls beginnen bei 0, wenn am DIP-Schalter die niedrigste Basisadresse eingestellt ist (alle DIP-Schalter OFF). Die Tastaturmatrix wird ohne weitere Pull-up-Widerstände angeschlossen. Die Shift-LED des Device-Treibers LCD1.TDD wird, wenn benötigt, am ersten erweiterten Ausgangsport (logische Adresse 10h) am Bit 0 angesteuert.

Das folgende Beispielprogramm dient als Basis für eigene Implementierungen. Nachdem die Einstellungen des Device-Treibers LCD1.TDD vorgenommen wurden, muß der Input-Puffer gelöscht werden, da während des Einstellens bereits falsche Zeichen gescannt sein könnten. Vor dem Löschen wird ein Scandurchgang abgewartet (normalerweise 16msec).

4

[illegible]

```

000102030405060708090A0B0C0D0E0F&      ' 00..0F
101112131415161718191A1B1C1D1E1F&      ' 10..1F
202122232425262728292A2B2C2D2E2F&      ' 20..2F
303132333435363738393A3B3C3D3E3F&      ' 30..3F
404142434445464748494A4B4C4D4E4F&      ' 40..4F
505152535455565758595A5B5C5D5E5F&      ' 50..5F
606162636465666768696A6B6C6D6E6F&      ' 60..6F
707172737475767778797A7B7C7D7E7F"%      ' 70..7F
  PRINT #DEV_NR, "<1BH>Z";A$;"<F0H>";      ' Key-Codes UN-Shifted setzen

  PRINT #DEV_NR, "<1BH>K<0><F0H>";          ' Tasten-Klick: 0 = ON
  PRINT #DEV_NR, "<1BH>r<20><5><F0H>";      ' Tasten-Repeat

' nun sind die Scan-Einstellungen ok. warte einen Scan-Durchgang...
WAIT_DURATION 20
PUT #DEV, #0, #UFCO_IBU_ERASE, 0          ' und loesche den Input-Puffer
END                                         ' = RETURN from Sub

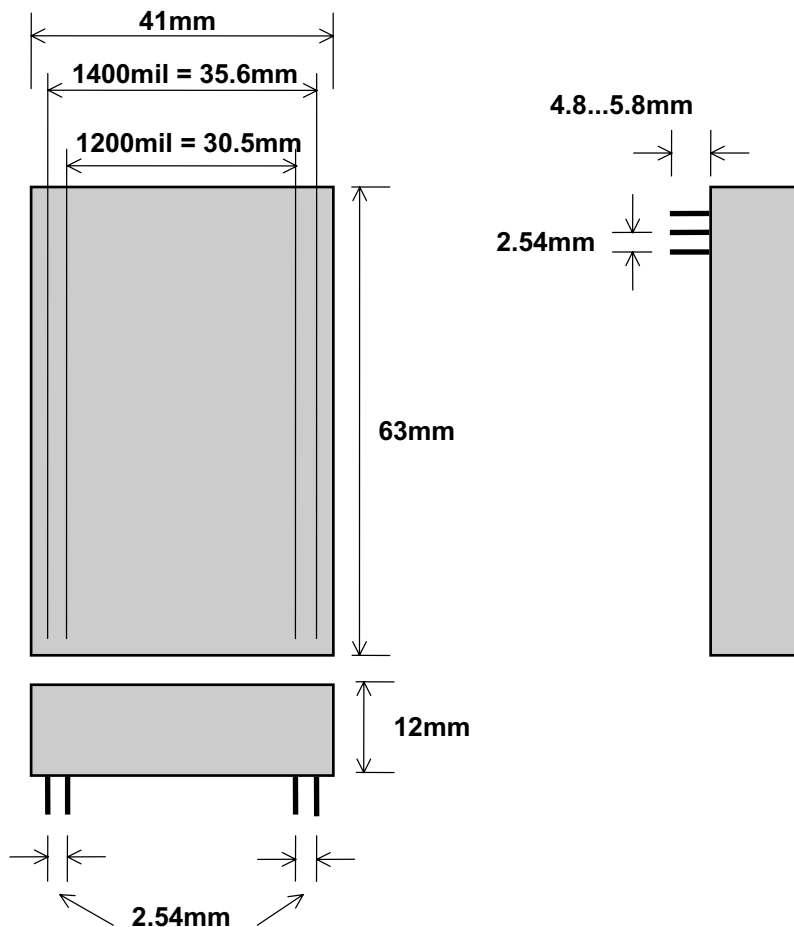
```

I/O-Erweiterungsmodule

Technische Daten des I/O-Erweiterungsmoduls EP6-UNIVD:

Maße / Gewicht	ca. 63 x 41 x 12 mm / ca. 50g
Spannungsversorgung	4,7V...5,5V
Ruhestromaufnahme	ca. 3mA
Anzahl der erw. OC-Ausgänge	8
abs. max. zulässiger Strom	500mA
max. Strom eines Outputs (DC)	350mA
max. Strom 8 Outputs (Duty 10%)	260mA
max. Strom 8 Outputs (Duty 50%)	90mA
max. Spannung OC-Outputs	50V
max. Verlustleistung eines Ports	0,4W
Anzahl der erweiterten Ausgänge	8
max. Strom eines Outputs 0...5V	5mA
Anzahl der erweiterten Eingänge	8
Anzahl Tastaturzeilen	8
Anzahl Tastaturspalten	16
Temperaturbereich	-40 bis +85°C

Abmessungen EP6-UNIVD:



4

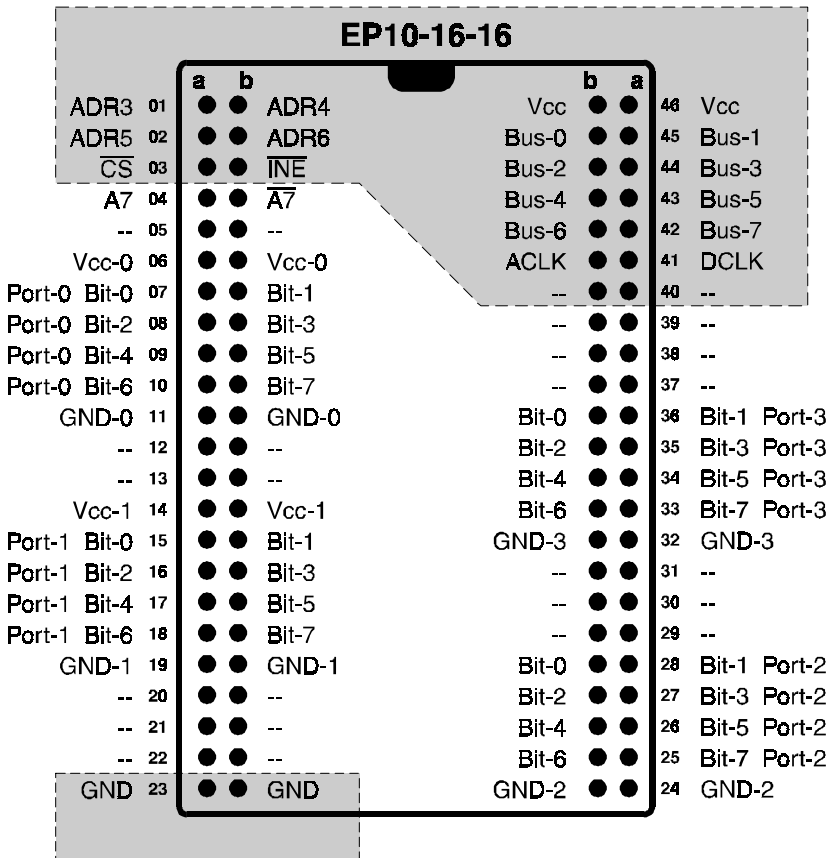
I/O-Erweiterungsmodule

Leere Seite

4

EP10-16-16**16 OptoEingänge / 16 OC-Ausgänge**

Das I/O-Erweiterungsmodul EP10-16-16 bietet 16 opto-entkoppelte Inputs sowie 16 OC-Ausgänge (0...max. 50V, max. 500mA). Durch die einstellbare Adresse kann das Modul mit anderen Erweiterungen kombiniert werden.

Pinbelegung EP10-16-16

Pin-Beschreibung EP10-16-16

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
1a	ADR3	Adresse	1b	ADR4	AdressBit 4
2a	ADR5	Adresse	2b	ADR6	AdressBit 6
3a	-CS	ChipSelect	3b	-INE	In Enable
4a	A7	Adressbit 7	4b	-A7	Neg. Adressbit 7
5a			5b		
6a	VCC-0	Spannung	6b	VCC-0	Spannung
7a	P0.0	Out 0 Bit0	7b	P0.1	Out0 Bit1
8a	P0.2	Out0 Bit2	8b	P0.3	Out0 Bit3
9a	P0.4	Out0 Bit4	9b	P0.5	Out0 Bit5
10a	P0.6	Out0 Bit6	10b	P0.7	Out0 Bit7
11a	GND-0	Masse	11b	GND-0	Masse
12a			12b		
13a			13b		
14a	VCC-1	Spannung	14b	VCC-1	Spannung
15a	P1.0	Out1 Bit0	15b	P1.1	Out1 Bit1
16a	P1.2	Out1 Bit2	16b	P1.3	Out1 Bit3
17a	P1.4	Out1 Bit4	17b	P1.5	Out1 Bit5
18a	P1.6	Out1 Bit6	18b	P1.7	Out1 Bit7
19a	GND-1	Masse	19b	GND-1	Masse
20a			20b		
21a			21b		
22a			22b		
23a	GND	Masse	23b	GND	Masse

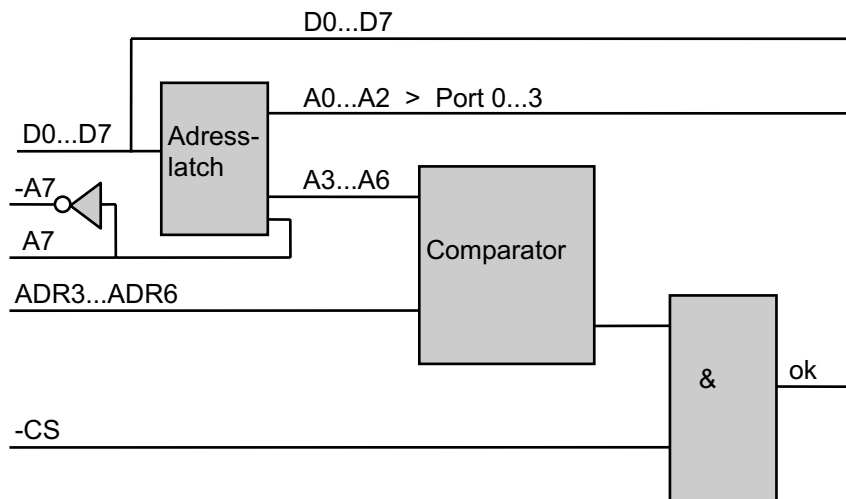
Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
24a	GND-2	Masse-2	24b	GND-2	Masse-2
25a	P2.7	In2 Bit7	25b	P2.6	In2 Bit6
26a	P2.5	In2 Bit5	26b	P2.4	In2 Bit4
27a	P2.3	In2 Bit3	27b	P2.2	In2 Bit2
28a	P2.1	In2 Bit1	28b	P2.0	In2 Bit0
29a			29b		
30a			30b		
31a			31b		
32a	GND-3	Masse-3	32b	GND-3	Masse-3
33a	P3.7	In3 Bit7	33b	P3.6	In3 Bit6
34a	P3.5	In3 Bit5	34b	P3.4	In3 Bit4
35a	P3.3	In3 Bit3	35b	P3.2	In3 Bit2
36a	P3.1	In3 Bit1	36b	P3.0	In3 Bit0
37a			37b		
38a			38b		
39a			39b		
40a			40b		
41a	DCLK	Datenclock	41b	ACLK	Adressclock
42a	Bus-7	I/O-Bus Bit7	42b	Bus-6	I/O-Bus Bit6
43a	Bus-5	I/O-Bus Bit5	43b	Bus-4	I/O-Bus Bit4
44a	Bus-3	I/O-Bus Bit3	44b	Bus-2	I/O-Bus Bit2
45a	Bus-1	I/O-Bus Bit1	45b	Bus-0	I/O-Bus Bit0
46a	VCC		46b	VCC	

Adressierung des EP10-16-16

Das Modul EP10-16-16 wird direkt von dem Tiger-BASIC[®] EPORT-System unterstützt. Der Anschluß an ein BASIC-Tiger[®]-Modul A oder Tiny-Tiger[®]-Modul erfordert bei der Standard-Pinbenutzung folgende Pins:

Pin-Bezeichnung	BASIC-Tiger [®] -Pins	Modultyp-A Pin-Nr.	Tiny-Tiger Pin-Nr.
Bus-0...Bus-7	L60...L67	2...9	1...8
ACLK (Adressclock)	L33	30	29
DCLK (Datenclock)	L34	31	30
-INE (in-enable)	L35	32	31
ADR3...ADR6	A3...A6 Inputs für Basis-Adresse des Moduls		
-CS	Chip-Select Input, low aktiv		
weitere Pins			
A7	A7-Ausgang des internen Adreßlatch		
-A7	A7-Ausgang invertiert		
Port-m Bit-n	Ports (m=0...1) mit je 8 Bits (n=0...7) OC-Outputs Ports (m=2...3) mit je 8 Bits (n=0...7) Opto-Inputs		

Blockschaltbild der Adressierung des EP10-16-16:



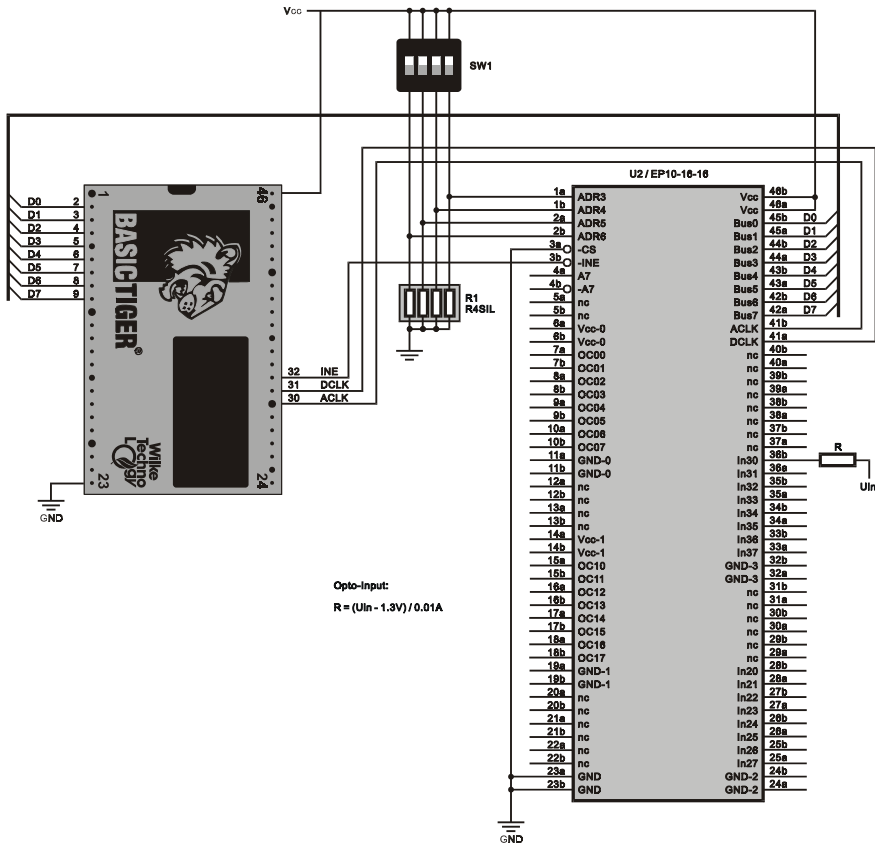
I/O-Erweiterungsmodule

An den Leitungen ADR3...ADR6 wird die Basis-Adresse des Erweiterungsmoduls angelegt. Die erweiterten Input-Ports belegen 8 Adressen ab der Basis-Adresse, es werden jedoch nur 4 Adressen verwendet. Das Modul wird adressiert, wenn '-CS' auf 'low' liegt und eine Adresse im eingestellten Bereich angesprochen wird.

EP10-16-16 Adressierung						
-CE	ADR6	ADR5	ADR4	ADR3	Port-0	Port-3
1	x	x	x	x	—	—
0	0	0	0	0	0	3
0	0	0	0	1	8	0Bh
0	0	0	1	0	10h	13h
0	0	0	1	1	18h	1Bh
0	0	1	0	0	20h	23h
0	0	1	0	1	28h	2Bh
0	0	1	1	0	30h	33h
0	0	1	1	1	38h	3Bh
0	1	0	0	0	40h	43h
0	1	0	0	1	48h	4Bh
0	1	0	1	0	50h	53h
0	1	0	1	1	58h	5Bh
0	1	1	0	0	60h	63h
0	1	1	0	1	68h	6Bh
0	1	1	1	0	70h	73h
0	1	1	1	1	78h	7Bh

Da A7 intern nicht ausgewertet wird, spiegeln sich die Adressen ab Adresse 80h.

Anschlußbeispiel EP10-16-16



4

Die Adressen des Moduls beginnen bei 0, wenn am DIP-Schalter die niedrigste Basisadresse eingestellt ist (alle DIP-Schalter OFF). Die Optokoppler-Eingänge benötigen ca. 10mA und haben keinen internen Vorwiderstand. Bei einer typischen Diodenflußspannung von 1,3V und einer Eingangsspannung U_{in} lautet die Berechnung des Vorwiderstandes:

$$R = U_{in} - 1,3V / 0,01A$$

I/O-Erweiterungsmodule

Programmbeispiel:

4

```
' -----
' Name: EP10-1.TIG
' -----
#include DEFINE_A.INC          ' allgemeine Definitionen
USER_EPORT PHYSOFFS, 0F0h      ' Offset zu phys. Adr. -10h
USER_EPORT NROFOUT, 2          ' 2 erweiterte Output Ports
USER_EPORT LASTLADR, 11h       ' letzte log. Outputadresse
USER_EPORT INITIAL, 0, "&      ' initialisiere ab Adr. 0
00 01"%                        ' mit dem Wert der phys. Adr.

TASK MAIN                      ' Beginn Task MAIN
' LCD-Treiber installieren (BASIC-Tiger)
INSTALL DEVICE #1, "LCD1.TDD"
' LCD-Treiber installieren (TINY-Tiger)
INSTALL DEVICE #1, "LCD1.TDD", 0, 0, 0, 0, 0, 0, 80h, 8

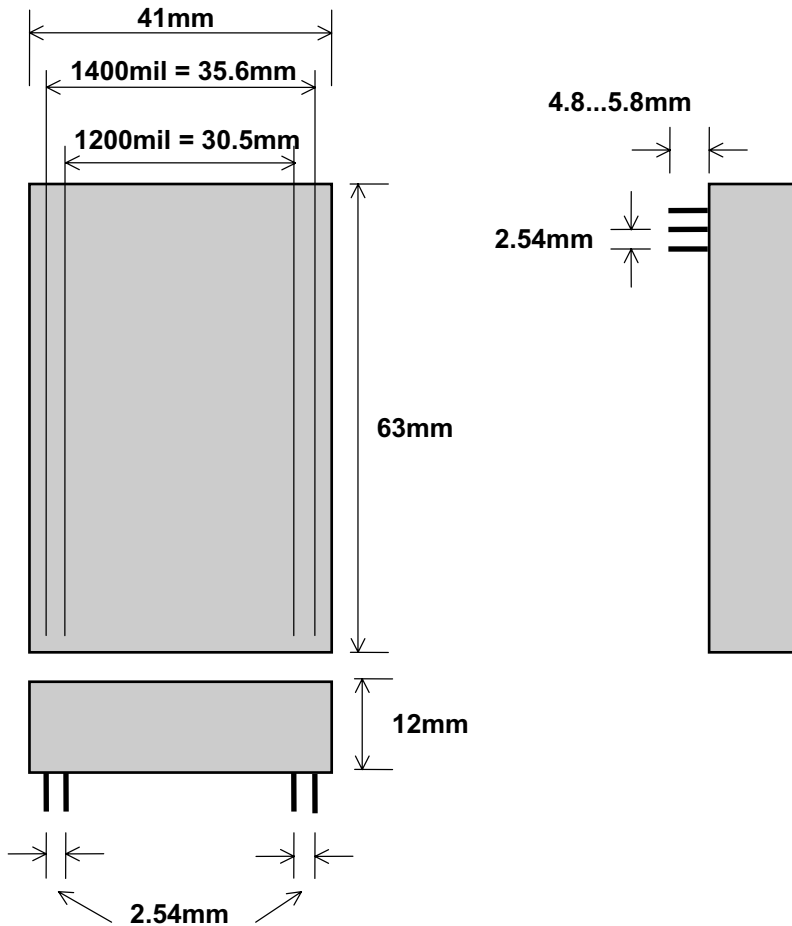
WAIT_DURATION 2000             ' 2 Sek. warten
OUT 10h,255,11h                ' gebe logische Portadr. aus
OUT 11h,255,10h                ' gebe logische Portadr. aus
IN 12h,VALUE                    ' lies von Port
PRINT #1, "Port 2 =";VALUE      ' Wert ausgeben
IN 13h,VALUE                    ' lies von Port
PRINT #1, "Port 3 =";VALUE      ' Wert ausgeben
END                             ' Ende Task MAIN
```

Technische Daten des I/O-Erweiterungsmoduls EP10-16-16:

Maße / Gewicht	ca. 63 x 41 x 12 mm / ca. 50g
Stromversorgung	4,7V...5,5V
Ruhestromaufnahme	ca. 12mA
Anzahl der erweiterten Ausgänge	16
abs. Max. zulässiger Strom	500mA
max. Strom eines Outputs (DC)	350mA
max. Strom 8 Outputs (Duty 10%)	260mA
max. Strom 8 Outputs (Duty 50%)	90mA
max. Spannung OC-Outputs	50V
max. Verlustleistung eines Ports	0,4W
Anzahl der erweiterten Eingänge	16
Diodenstrom	5...50mA
Diodenflußspannung	1,3V
Temperaturbereich	-40 bis +85°C

I/O-Erweiterungsmodule

Abmessungen EP10-16-16:



EP11-8AD, EP12-16AD, EP13-32AD, EP14-64AD

8 bis 64 A/D-Eingänge mit 12-Bit-Auflösung

Die I/O-Erweiterungsmodule EP11-8AD bis EP14-64AD bieten auf kompaktem Raum 8/16/32/64 A/D-Eingänge mit interner Referenzspannung (4.096V) und per Software programmierbaren Eingangsparametern. (0...5V/10V/±5V/±10V). Die Module kommen mit einer Betriebsspannung von 5V aus, lassen jedoch Eingangsspannungen oberhalb 5V und unterhalb GND zu.

Pinbelegung EP11-8AD bis EP14-64AD

EP11...EP14									
		a	b				b	a	
A0	01	●	●	A1		Vcc	●	●	46 Vcc
A2	02	●	●	RD		Bus-0	●	●	45 Bus-1
WR	03	●	●	HBEN		Bus-2	●	●	44 Bus-3
$\overline{CS}$	04	●	●	--		Bus-4	●	●	43 Bus-5
Ch-00	05	●	●	Ch-1		Bus-6	●	●	42 Bus-7
Ch-02	06	●	●	Ch-3		Ch-56	●	●	41 Ch-57
Ch-04	07	●	●	Ch-5		Ch-58	●	●	40 Ch-59
Ch-06	08	●	●	Ch-7		Ch-60	●	●	39 Ch-61
AGND0	09	●	●	Ch-08		Ch-62	●	●	38 Ch-63
Ch-09	10	●	●	Ch-10		AGND7	●	●	37 Ch-48
Ch-11	11	●	●	Ch-12		Ch-49	●	●	36 Ch-50
Ch-13	12	●	●	Ch-14		Ch-51	●	●	35 Ch-52
Ch-15	13	●	●	AGND1		Ch-53	●	●	34 Ch-54
Ch-16	14	●	●	Ch-17		Ch-55	●	●	33 AGND6
Ch-18	15	●	●	Ch-19		Ch-40	●	●	32 Ch-41
Ch-20	16	●	●	Ch-21		Ch-42	●	●	31 Ch-43
Ch-22	17	●	●	Ch-23		Ch-44	●	●	30 Ch-45
AGND2	18	●	●	Ch-24		Ch-46	●	●	29 Ch-47
Ch-25	19	●	●	Ch-26		AGND5	●	●	28 Ch-32
Ch-27	20	●	●	Ch-28		Ch-33	●	●	27 Ch-34
Ch-29	21	●	●	Ch-30		Ch-35	●	●	26 Ch-36
Ch-31	22	●	●	AGND3		Ch-37	●	●	25 Ch-38
GND	23	●	●	GND		Ch-39	●	●	24 AGND4

Pin-Beschreibung EP11-8AD

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
1a	ADR0	Analogportadr.	1b	ADR1	Analogportadr.
2a	ADR2	Analogportadr.	2b	-RD	Read
3a	-WR	Write	3b	HBEN	High-Byte Enable
4a	-CS	ChipSelect	4b		
5a	Ch.0	Kanal 0	5b	Ch.1	Kanal 1
6a	Ch.2	Kanal 2	6b	Ch.3	Kanal 3
7a	Ch.4	Kanal 4	7b	Ch.5	Kanal 5
8a	Ch.6	Kanal 6	8b	Ch.7	Kanal 7
9a	AGND-0	Analog Masse 0	9b	Ch.8	Kanal 8
10a	Ch.9	Kanal 9	10b	Ch.10	Kanal 10
11a	Ch.11	Kanal 11	11b	Ch.12	Kanal 12
12a	Ch.13	Kanal 13	12b	Ch.14	Kanal 14
13a	Ch.15	Kanal 15	13b	AGND-1	Analog Masse 1
14a	Ch.16	Kanal 16	14b	Ch.17	Kanal 17
15a	Ch.18	Kanal 18	15b	Ch.19	Kanal 19
16a	Ch.20	Kanal 20	16b	Ch.21	Kanal 21
17a	Ch.22	Kanal 22	17b	Ch.23	Kanal 23
18a	AGND-2	Analog Masse 2	18b	Ch.24	Kanal 24
19a	Ch.25	Kanal 25	19b	Ch.26	Kanal 26
20a	Ch.27	Kanal 27	20b	Ch.28	Kanal 28
21a	Ch.29	Kanal 29	21b	Ch.30	Kanal 30
22a	Ch.31	Kanal 31	22b	AGND-3	Analog Masse 3
23a	GND	Masse	23b	GND	Masse

EP11-8AD, EP12-16AD, EP13-32AD, EP14-64AD

Pin Nr.	Bezeichnung	Funktion	Pin Nr.	Bezeichnung	Funktion
24a	AGND-4	Analog Masse 4	24b	Ch.39	Kanal 39
25a	Ch.38	Kanal 38	25b	Ch.37	Kanal 37
26a	Ch.36	Kanal 36	26b	Ch.35	Kanal 35
27a	Ch.34	Kanal 34	27b	Ch.33	Kanal 33
28a	Ch.32	Kanal 32	28b	AGND-5	Analog Masse 5
29a	Ch.47	Kanal 47	29b	Ch.46	Kanal 46
30a	Ch.45	Kanal 45	30b	Ch.44	Kanal 44
31a	Ch.43	Kanal 43	31b	Ch.42	Kanal 42
32a	Ch.41	Kanal 41	32b	Ch.40	Kanal 40
33a	AGND-6	Analog Masse 6	33b	Ch.55	Kanal 55
34a	Ch.54	Kanal 54	34b	Ch.53	Kanal 53
35a	Ch.52	Kanal 52	35b	Ch.51	Kanal 51
36a	Ch.50	Kanal 50	36b	Ch.49	Kanal 49
37a	Ch.48	Kanal 48	37b	AGND-7	Analog Masse 7
38a	Ch.63	Kanal 63	38b	Ch.62	Kanal 62
39a	Ch.61	Kanal 61	39b	Ch.60	Kanal 60
40a	Ch.59	Kanal 59	40b	Ch.58	Kanal 58
41a	Ch.57	Kanal 57	41b	Ch.56	Kanal 56
42a	Bus-7	I/O-Bus Bit7	42b	Bus-6	I/O-Bus Bit6
43a	Bus-5	I/O-Bus Bit5	43b	Bus-4	I/O-Bus Bit4
44a	Bus-3	I/O-Bus Bit3	44b	Bus-2	I/O-Bus Bit2
45a	Bus-1	I/O-Bus Bit1	45b	Bus-0	I/O-Bus Bit0
46a	VCC		46b	VCC	

4

I/O-Erweiterungsmodule

Adressierung der EP11-EP14-AD

Die Module EP11-8AD bis EP14-64AD besitzen eine unterschiedliche Anzahl von Analogkanälen. An den Pins ADR0... ADR2 wird der Analogport ausgewählt, wenn gleichzeitig '-CS' auf 'low' liegt. Jeder Port hat wiederum 8 Kanäle. Ein Kanal wird durch Schreiben eines Steuerbytes auf den entsprechenden Port angewählt. Dieses Steuerbyte stellt auch die übrigen Parameter des Kanals ein.

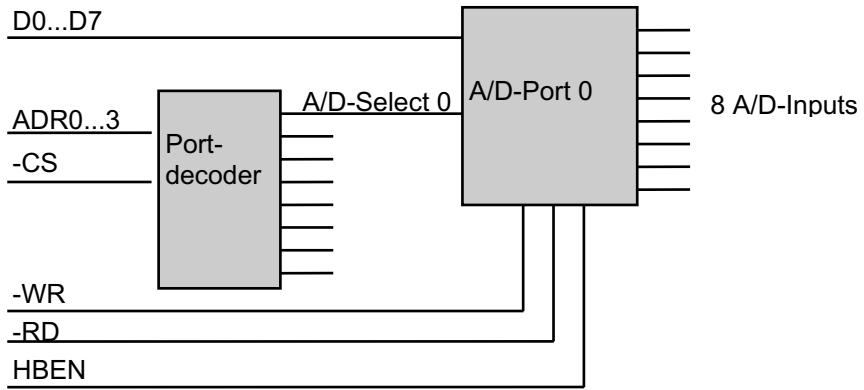
Bedeutung des Steuerbytes:

D7	D6	D5	D4	D3	D2	D1	D0
PD1	PD0	ACQM	RNG	BIP	A2	A1	A0
Power-Down-Mode 00: ungültig! 01: Normal 10: Standby 11: Power-Down		immer 0	Range und Polarität 00: 0...5V 01: 0...10V 10: $\pm 5V$ 11: $\pm 10V$		Kanal 000: Kanal 0 001: Kanal 1 010: Kanal 2 011: Kanal 3 100: Kanal 4 101: Kanal 5 110: Kanal 6 111: Kanal 7		

4

EP11-8AD, EP12-16AD, EP13-32AD, EP14-64AD

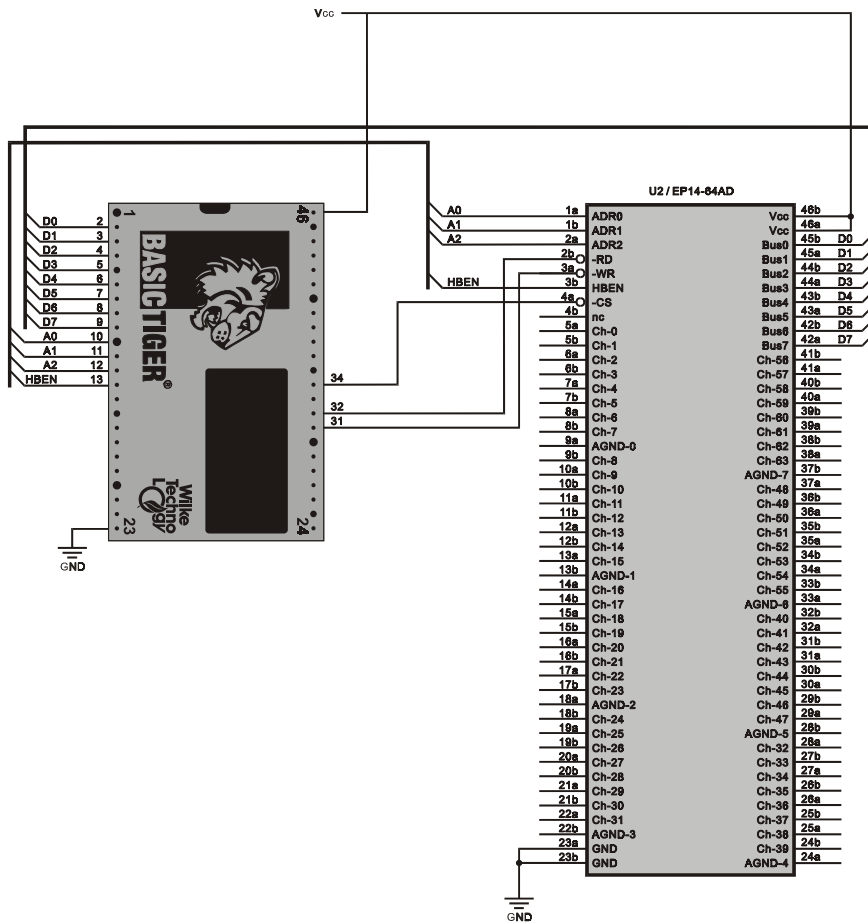
Blockschaltbild der Adressierung der EP11...EP14:



4

I/O-Erweiterungsmodule

Anschlußbeispiel EP11-8AD bis EP14-64AD



EP11-8AD, EP12-16AD, EP13-32AD, EP14-64AD

Technische Daten der I/O-Erweiterungsmodule EP11-8AD bis EP14-64AD:

	EP11	EP12	EP13	EP14
Maße / Gewicht	ca. 63 x 41 x 12 mm / ca. 50g			
Stromversorgung	4,7V...5,5V			
Ruhestromaufnahme Normalmodus, unipolar Normalmodus, bipolar	10mA 20mA	20mA 40mA	40mA 80mA	80mA 160mA
Anzahl der Analogeingänge	8	16	32	64
Genauigkeit				
Nichtlinearität Typ A	±1/2 LSB			
Nichtlinearität Typ B	±1 LSB			
Differentielle Nichtlinearität	±1 LSB			
Offset Error Typ A unipolar bipolar	±3 LSB ±5 LSB			
Offset Error Typ B unipolar bipolar	±5 LSB ±10 LSB			
Kanal-zu-Kanal Offset Error Mat- ching - unipolar bipolar	±0.1 LSB ±0.5 LSB			
Gain-Error Typ A uni- u. bipolar	±7 LSB			
Gain-Error Typ B uni- u. bipolar	±10 LSB			
Gain Temperature Coefficient unipolar bipolar	3 ppm/°C 5 ppm/°C			

4

I/O-Erweiterungsmodule

Analog-Input	
Input Current Bereiche 0...5V Bereich $\pm 5V$ Bereiche 0...10V Bereich $\pm 10V$	360 μA -600...360 μA 720 μA -1200...720 μA
Input dynamic Resistance unipolar bipolar	21 k 16 k
Input Capacitance	40 pF

4

Erweitertes I/O-System

Das erweiterte I/O-System des BASIC-Tiger[®] ermöglicht die Realisierung von weiteren Input- und Output-Ports (ePorts) mit wenigen preiswerten externen Komponenten. Es werden bis zu 1920 erweiterte Ein- oder Ausgangspins unterstützt. Vom Tiger-BASIC[®] aus erfolgt der Zugriff auf die erweiterten Ein- und Ausgänge genauso wie der Zugriff auf die internen Ports. Die Ansteuerung des erweiterten I/O-Systems verwendet den Datenbus, an dem auch andere Geräte wie LC-Display, Drucker oder parallele Eingabe angeschlossen sind. Die Geräte haben jeweils eigene Steuerleitungen zur Aktivierung. Weitere Device-Treiber werden ebenfalls diesen Datenbus verwenden. Das erweiterte I/O-System ist in Tiger-BASIC[®] integriert und ein Device-Treiber ist nicht erforderlich.

Folgende Pins werden beim Modultyp A und Tiny-Tiger[®] als Standardpins verwendet:

Bus-Bezeichnung	BASIC-Tiger [®] -Pins	Modultyp-A Pin-Nr.	Tiny-Tiger Pin-Nr.
D0...D7	L60...L67	2...9	1...8
Aclk (Adressclock)	L33	30	29
Dclk (Datenclock)	L34	31	30
-INE (in-enable) oder -keyb (Tastatur)	L35	32	31
E (LCD: enable)	L36	33	32
RS (LCD: Reg.select)	L37	34	33
beep (nicht auf J22)	L42	35	*
busy-in (PRN1)	L70	10	9
strobe-out (PRN1)	L71	11	10
busy-out (PIN1)	L80	14	13
strobe-in (PIN1)	L81	15	14

* Der Standard-Beep-Pin L42 ist bei Tiny-Tiger[®] nicht vorhanden und wird bei der Installation von LCD1.TDD auf einen anderen Pin verlegt.

Alle Leitungen werden von den Device-Treibern automatisch richtig bedient. Zur Benutzung der erweiterten I/O-Pins ist kein Device-Treiber erforderlich.

Erweitertes I/O-System

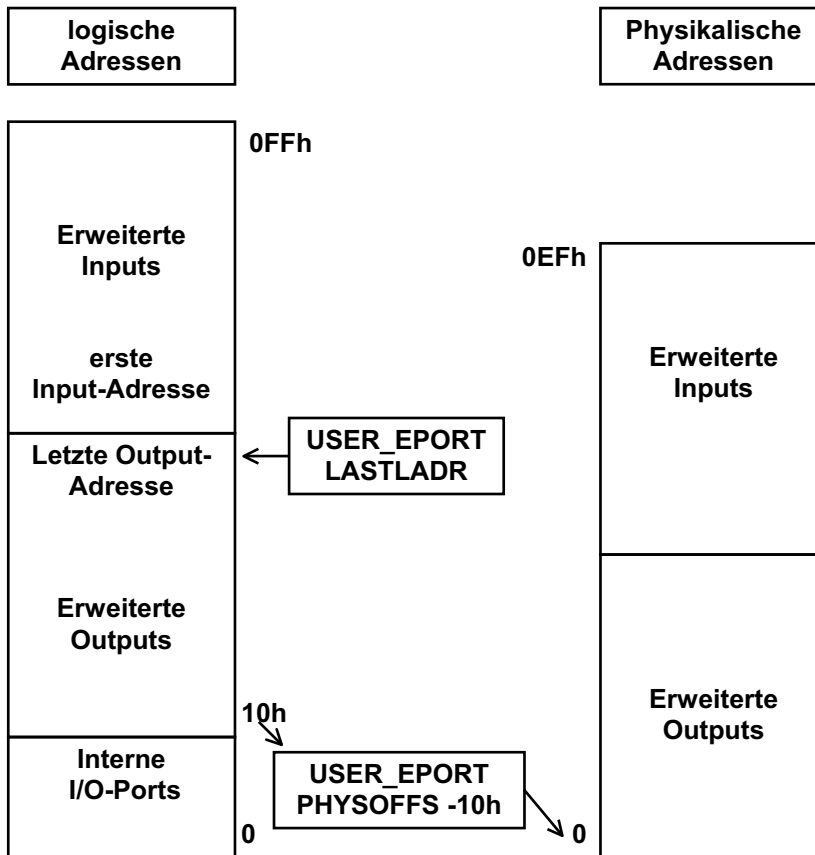
Sowohl das erweiterte Portsystem als auch Device-Treiber können Portbits gegen Zugriff schützen, so daß diese dann nicht mehr durch IN- und OUT-Instruktionen angesprochen werden können. Ausnahme: Low-Level-Instruktionen können immer auf die Ports zugreifen. Siehe auch 'Die Softwareseite der erweiterten I/O-Ports', Seite 318.



Adreß-Generierung

Da die internen I/O-Ports bereits Adressen belegen, werden die erweiterten Ports über logische Adressen angesprochen, die nicht unbedingt mit den Adressen der Hardware-Verschaltung übereinstimmen. Die Software des ePort-Systems addiert zu den logischen BYTE-Adressen einen Offset, um physikalisch die ePorts anzusprechen. Die logischen Adressen sind unterteilt in einen Adreßbereich für Output und einen Adreßbereich für Input, insgesamt von 10H (16) bis FFH (255). Auf die letzte für Output verwendete Adresse folgen Adressen für Input. Die letzte logische Adresse für Output ist setzbar.

Die Abbildung zeigt, wie das EPORT-System die logischen 240 (0E0h) Erweiterungsadressen in physikalische Adressen umsetzt:



Erweitertes I/O-System

Das Beispielprogramm zeigt, wie von Tiger-BASIC[®] aus die erweiterten Ports angesprochen werden:

```

IN 7, N          ' <----- intern-Port ----->
OUT 6, MASKE, WERT ' lies Byte von Port-7 --> N
                  ' Wert -> Port 6
                  ' 0-Bit in der Maske --> Port-Bit
                  ' aendert sich nicht

IN 90H, N        ' <----- extern-Port ----->
OUT 33H, MASKE, WERT ' lies Byte von Port-90h --> N
                  ' Wert -> Port 33H
                  ' 0-Bit in der Maske --> Port-Bit
                  ' aendert sich nicht

```

4

In der folgenden Beispieltabelle ist dargestellt, welche Adressen sich bei einem Offset von -10H = F0H und mit einer letzten logischen Output-Adresse von 8FH ergeben.

	logisch (BASIC)		physikalisch (Decodierung)	
	von Adresse	bis Adresse	von Adresse	bis Adresse
Outputs	10H (16)	8FH (139)	0H (0)	7FH (127)
Inputs	90H (144)	FFH (240)	80H (128)	EFH (240)

Auf dem Plug & Play Lab ergibt sich folgendes Bild (Offset -8):

	logisch (BASIC)		physikalisch (Decodierung)	
	von Adresse	bis Adresse	von Adresse	bis Adresse
Outputs	10H (16)	8FH (139)	8H (0)	87H (135)
Inputs	90H (144)	FFH (255)	88H (136)	F8H (255)

Auf dem Plug & Play Lab befinden sich 64 erweiterte Ausgänge. Sie werden über die Adressen 10h...17h (hexadezimal) oder 16...23 (dezimal) angesprochen. Die letzte Output-Adresse ist standardmäßig auf 1Fh gesetzt.

Erweiterte Inputs finden Sie auf dem Plug & Play Lab in abgewandelter Form als Tastatur wieder. Da die Tastatur nicht ausdekodiert ist, spiegelt sie sich im gesamten Adreßraum. Der Device-Treiber LCD1.TDD tastet die Tastaturreihen bei den logischen Adressen von 98H bis A7H ab, also physikalisch von 90H bis 9FH. Die Scan-Adressen sind jedoch auch veränderbar (ESC-k-Kommando des LCD1.TDD).

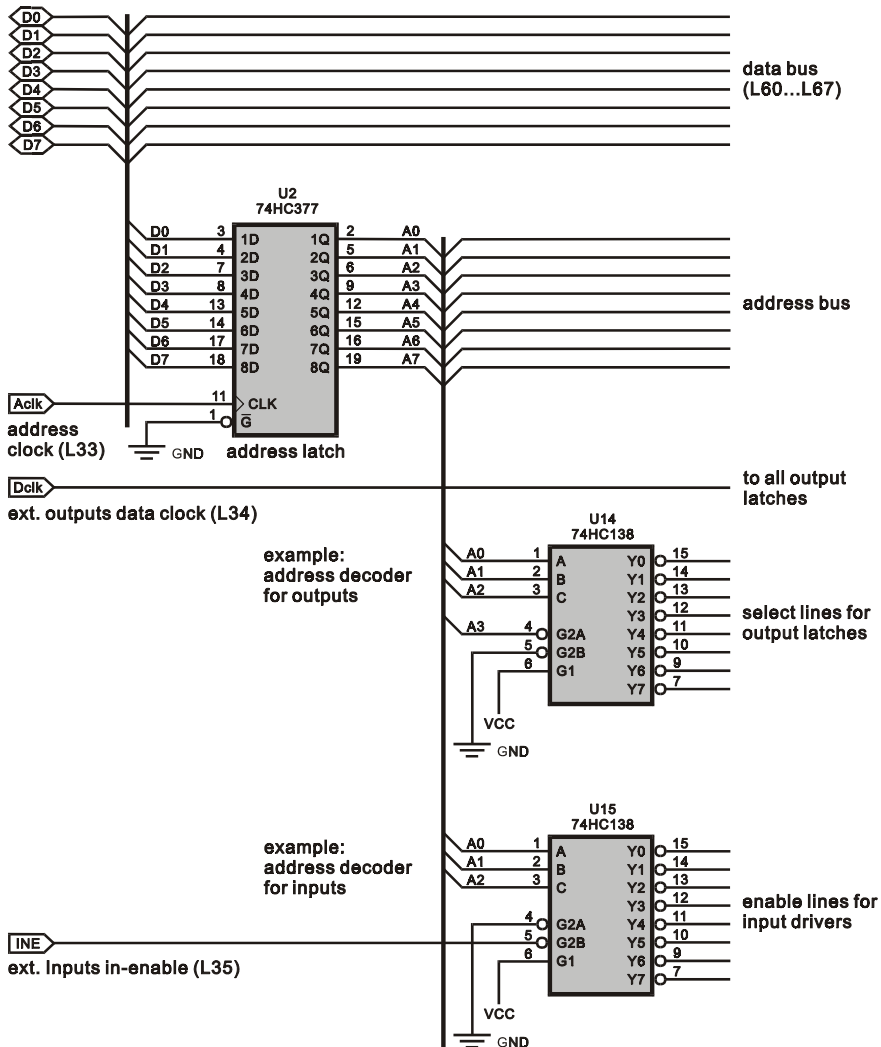
Die Abbildung zeigt die Hardware zur Erzeugung der Adressen für erweiterte I/Os. Der Adreßbus geht von einem 8-Bit-Speicher aus, in den die Adresse mit dem

‘Ack’-Signal eingeschrieben wird. Wenn notwendig, erfolgt eine weitere Decodierung der Adressen für Ausgangs-Latch-IC, Eingangs-Bustreiber-IC, etc.

Aus der Abbildung ist ersichtlich, daß das Adreß-Latch nur einmal erforderlich ist, während die Decodierung für Ausgänge und Eingänge getrennt erfolgt. Der Decoder für Eingänge wird nur aktiviert, wenn ‘in-enable’ (low)aktiv ist.

Auf den folgenden Seiten finden Sie Beispiele für diverse praktische Anwendungen.

Adreß-Generierung des ePort-Systems:



Erweiterung der Ausgänge

Die Realisierung der erweiterten Ausgänge erfolgt über Standardbauteile der 74er Reihe:

- '138 Adress-Decoder
- '377 Octal D-Latch

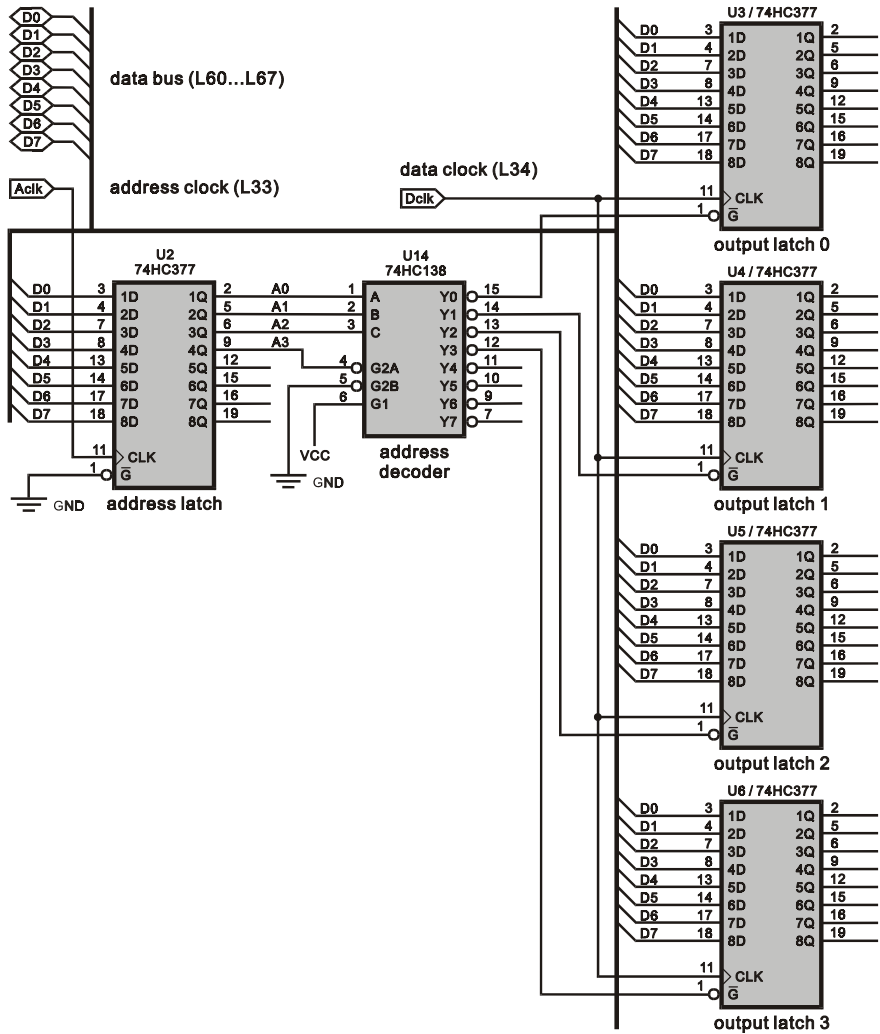
Die Adresse wird in das erste Latch 74HC377 durch einen Impuls auf der Leitung 'Aclk' übertragen. Der Adressbus ist auf den 74HC138 geführt. Bei diesem Baustein handelt es sich um einen 3 zu 8 Multiplexer. Dieser aktiviert, in Abhängigkeit der anliegenden Adresse, einen der 74HC377 über dessen Enable-Leitung. Bei den erweiterten Ausgängen werden die Daten mit 'Dclk' in das adressierte Ausgangs-Latch übertragen. In der folgenden Beispielschaltung für 32 erweiterte Ausgänge beginnen die physikalischen Adressen bei 0 und spiegeln sich im gesamten Adreßraum:

- 0...4
- 10H...14H
- 20H...24H
- ...
- 70H...74H

Bei Adressen oberhalb 80H verhindert das Signal A7 am HC138 weitere Spiegelungen. Der Offset von logischer zu physikalischer Adresse ist hier -10H (F0H). Die gesamte Bedienung aller Steuerleitungen erfolgt über das erweiterte I/O-System des BASIC-Tiger[®]-Moduls, wenn eine OUT-Instruktion mit entsprechender Adresse ausgeführt wird.

Erweitertes I/O-System

Beispielschaltung für 32 erweiterte Ausgänge auf 4 Ports (8 Ports wären möglich):



4

Erweitertes I/O-System

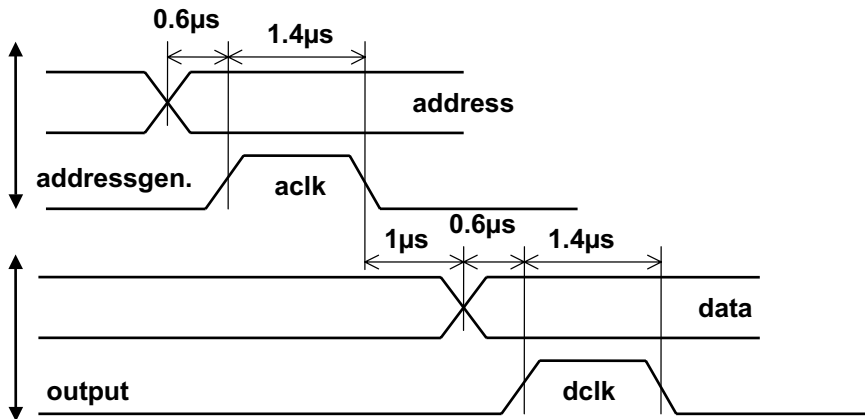
Folgende Maßnahmen in der Software sind notwendig, um die erweiterten Ausgänge anzupassen. Das Beispiel setzt voraus, daß der physikalische Offset des erweiterten Portsystems -10h ist. (Siehe auch: 'Die Softwareseite der erweiterten I/O-Ports' auf Seite 318.)

```
#INCLUDE DEFINE_A.INC
USER_EPORT PHYSOFFS, 0F0h           \ phys. Offset = -10h
USER_EPORT NROFOUT, 4               \ 4 Ports x8=32 erw. Ausgaenge
\ dieser Initialisierungsstring setzt die logische Portadresse
\ als Bitmuster auf jeden Port. Ab phys. Adresse 8 wird initialisiert.
USER_EPORT INITIAL, 8, "10 11 12 13"%

TASK MAIN
  OUT 10h, 255, 01010101b           \ Ausgabe auf ersten
                                     \ erweiterten Ausgangsport
END
```

4

Output-Timing des erweiterten I/O-Systems (BASIC-Tiger[®] Modul A oder Tiny-Tiger[®]):



Erweiterung der Eingänge

Die Realisierung der erweiterten Eingänge erfolgt über Standardbauteile der 74er Reihe:

- '138 Adress-Decoder
- '377 Octal D-Latch
- '245 Bus-Transceiver

Die Adresse wird in das erste Latch 74HC377 durch einen Impuls auf der Leitung 'Aclk' übertragen. Der Adressbus ist auf den 74HC138 geführt. Bei diesem Baustein handelt es sich um einen 3 zu 8 Multiplexer. Dieser aktiviert, in Abhängigkeit der anliegenden Adresse, einen der 74HC245 über dessen Enable-Leitung, sobald die 'in-enable'-Leitung auf 'low' geht.

In der folgenden Beispielschaltung für 32 erweiterte Eingänge beginnen die physikalischen Adressen bei 80H und spiegeln sich im gesamten Adreßraum:

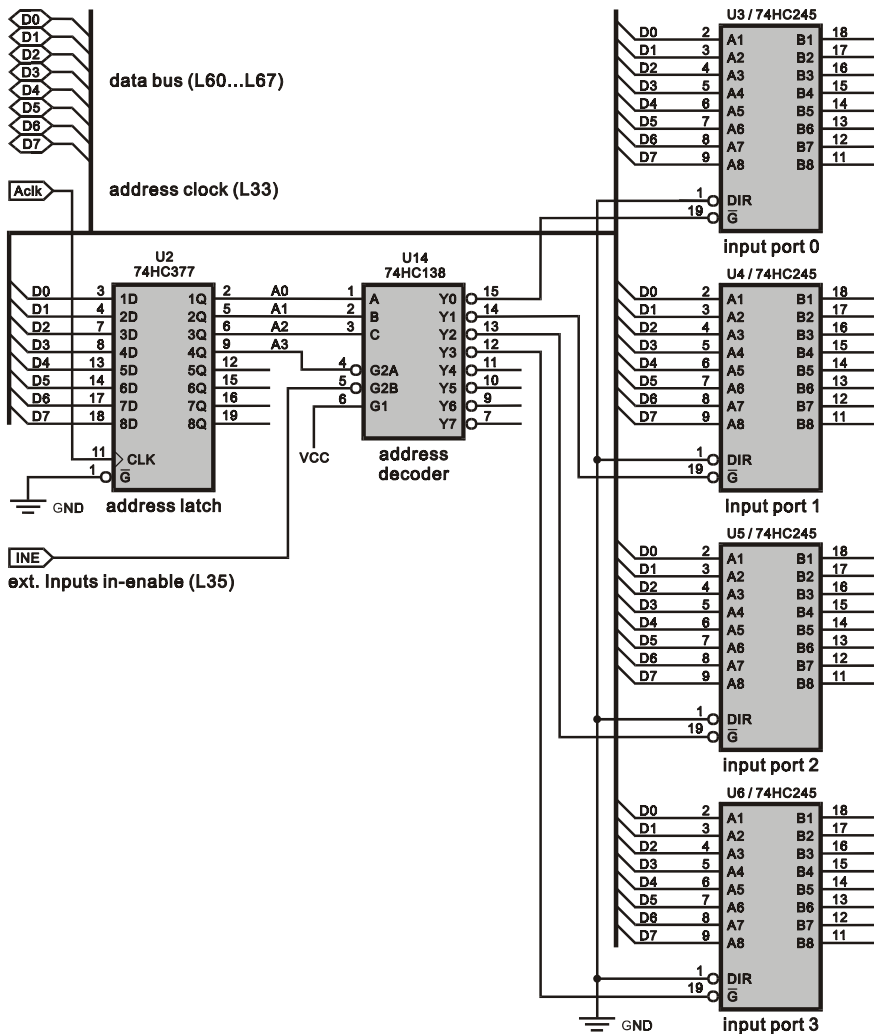
- 80H...84H
- 90H...94H
- A0H...A4H
- ...
- F0H...F4H

Bei Adressen unterhalb 80H verhindert das Signal A7 am HC138 weitere Spiegelungen. Der Offset von logischer zu physikalischer Adresse ist hier -10H (F0H). Die gesamte Bedienung aller Steuerleitungen erfolgt über das erweiterte I/O-System des BASIC-Tiger[®]-Moduls, wenn eine IN-Instruktion mit entsprechender Adresse ausgeführt wird.

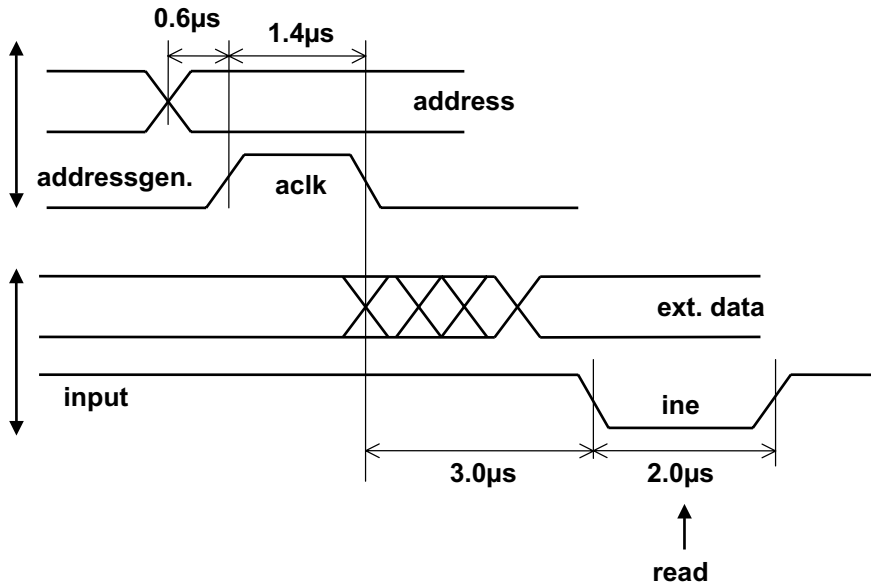
Wenn am BASIC-Tiger[®]-Modul eine Tastatur angeschlossen und der Treiber LCD1.TDD verwendet wird, sind einige Eingangsadressen durch die Tastatur belegt, so daß die erweiterten Inputs besser ausdekodiert werden müssen. Zunächst folgt eine Beispielschaltung ohne Tastatur im System.

Erweitertes I/O-System

32 erweiterte Input-Leitungen (ohne verwendung der Tastatur):



Input-Timing des erweiterten I/O-Systems (BASIC-Tiger[®] Modul A oder Tiny-Tiger[®]):



Ein Beispiel für erweiterte Eingänge und Tastatur

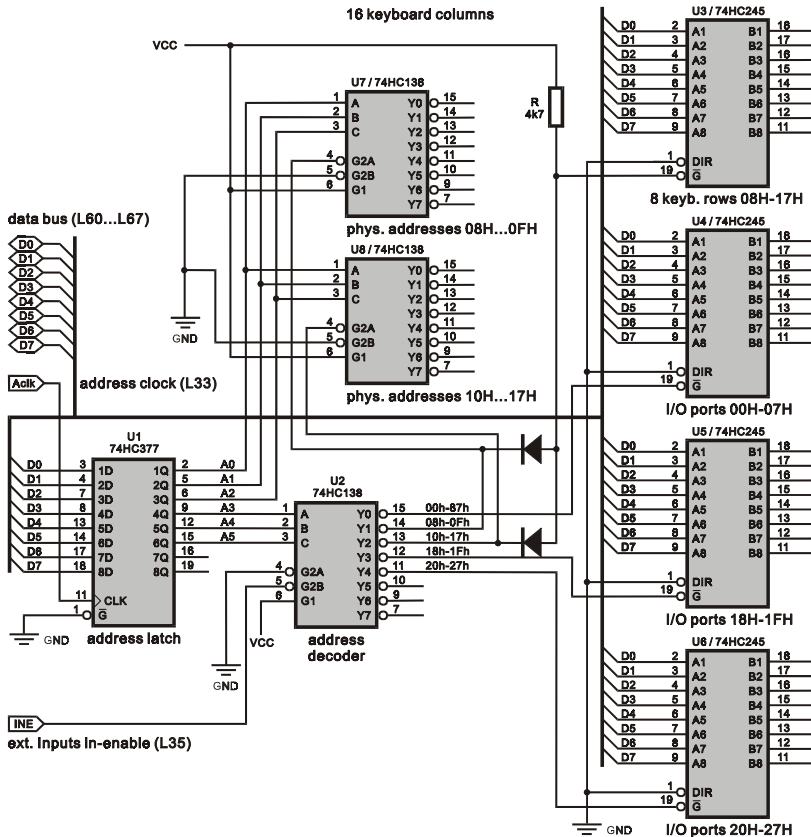
Da der Treiber LCD1.TDD einige erweiterte Input-Adressen belegt, muß bei gleichzeitiger Realisierung von Tastatur und erweiterten Input-Pins der Input-Adreßbereich besser ausdekodiert werden. Da sich auf dem Plug & Play Lab die Tastatur nicht weiter ausdekodiert ist, sind dort erweiterte Inputs ohne weitere Maßnahmen nicht zu realisieren.

Der Treiber LCD1.TDD verwendet die logischen Adressen 98H...A7H zum Scannen der Tastatur. Bei einem Offset für erweiterte I/O von -10H ergeben sich physikalische Scan-Adressen von 88H bis 97H. Auf dem Plug & Play Lab ergeben sich bei einem Offset für erweiterte I/O von -8 physikalische Scan-Adressen von 90H bis 9FH ergeben. Das Beispiel auf der folgenden Seite zeigt eine Dekodierung für eine Tastatur mit 128 Tasten und 24 erweiterte Input-Ports.

Die Adressen: Der Haupt-Adreßdekoder aktiviert grundsätzlich nur im Adreßbereich oberhalb 80H (durch A7). Dort teilt er die Adressen in Bereiche von je 8 Adressen. Von 88H bis 97H ist je eine **Tastaturspalte** durch 2 weitere HC138 sowie der oberste HC245 zum Lesen **Tastaturreihen** über die beiden Dioden aktiviert. Das ist der physikalische Adreßbereich der Tastatur. Die 3 erweiterten Input-Ports lassen sich auf je 8 Adressen gleichermaßen ansprechen:

- 80H...8FH
- 98H...9FH
- A0H...A7H

Erweiterte Eingänge zusammen mit Tastatur:



4

Die Softwareseite der erweiterten I/O-Ports

Die erweiterten I/O-Ports werden genauso vom Tiger-BASIC[®] aus angesprochen wie die internen Ports.

Da einige Adressen bereits durch interne Ports belegt sind und sich auf den externen Ports durch unvollständige Ausdekodierung Adreß-Spiegelungen ergeben können, wird die Ansteuersoftware an die wirklich vorhandenen Gegebenheiten angepaßt.

Die Compiler-Anweisung

USER_ EPORT Kommando, Argument

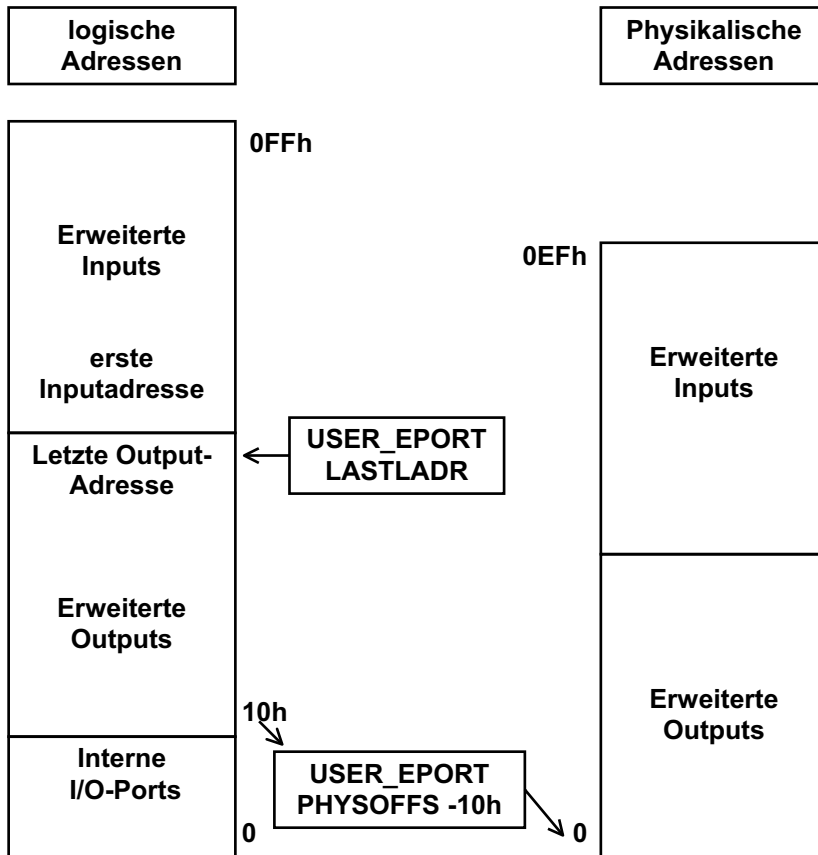
setzt System-Parameter im BASIC-Tiger[®]-Modul, die die Ansteuerung der externen Ports - und indirekt damit auch einiger interner Ports - betreffen. Die Kommandos sind in der Include-Datei DEFINE_A.INC definiert. Diese Datei muß also vor der Benutzung eines der folgenden Kommandos eingebunden sein.

Kommandos der Compiler-Anweisung USER_EPORT:

Kommando	Argument	Bedeutung
ACT		Aktivitäts-Flag zur Benutzung der erweiterten Ports.
	ACTIVE	erw. Ports werden benutzt.
	NOACTIVE	erw. Ports werden nicht benutzt
PHYSOFFS	Offset	setzt den Offset von logischer zu physikalischer Adresse
NROFOUT	Anzahl	setzt die Anzahl der Output-Ports, die initialisiert werden
LASTLADR	Adresse	gibt die letzte logische Adresse für erweiterte Outputs an. Darauf folgende Adressen sind Inputs.
INITIAL	Base, HEX-String	gibt die Initialisierungswerte für die erweiterten Ports an; beginnt ab phys. Adresse Base mit der Initialisierung
BUSL	Port-Adresse	Port für Datenbus der erweit. I/O
CTRL	Port Adresse	Port für Steuerleitungen der erweit. I/O
DCLKBMASK	DCLK-Bitmaske	Bitmaske für DCLK Daten-Clock
ACLKBMASK	ACLK-Bitmaske	Bitmaske für ACLK Adress-Clock
INEBMASK	INE-Bitmaske	Bitmaske für INE In-Enable

Erweitertes I/O-System

Die Abbildung zeigt, wie das EPORT-System die logischen 240 Erweiterungsadressen in physikalische Adressen umsetzt:



4

Erweitertes I/O-System

Nach dem Reset steht die Ansteuerung der erweiterten Ports zur Verfügung, das Aktivitäts-Flag für das EPORT-System steht auf ACTIVE. Um die betroffenen I/O-Ports als normale Ports des Controllers zu nutzen, muß das Aktivitäts-Flag zurückgesetzt werden:

USER_EPORT ACT, NOACTIVE

ACT setzt so das Aktivitäts-Flag; hier auf NOACTIVE.

USER_EPORT PHYSOFFS, 0F8H

PHYSOFFS setzt den Offset von der logischen Adresse zur physikalischen Adresse; In folgendem Beispiel: auf 0F8H (= -8).

Achtung: nicht erlaubt ist PHYSOFFS 0.

4

```
USER_EPORT PHYSOFFS, 0F8H    ` setze Offset -8 (BYTE!)
OUT 10H, Wert                 ` auf dem extended I/O-Bus wird
                              ` auf physikalische Adresse 8
                              ` geschrieben. (=10H-8)
```

Ferner ist die Anzahl der zu initialisierenden Output-Ports mit dem Kommando **NROFOUT** einstellbar:

USER_EPORT NROFOUT, Anzahl

NROFOUT stellt die Anzahl der zu initialisierenden Output-Ports ein.

Anzahl z.B. 10H (16), so werden 16 Output-Ports initialisiert.

Stellen Sie diesen Wert möglichst genau ein. Ein zu großer Wert führt zu unnötigen Operationen und kann bei unvollständiger Decodierung zu Fehlern durch Adreß-Spiegelungen führen. Wenn nur wenige ePorts verwendet werden, kann man durch unvollständige Decodierung Bauteile einsparen. Dann spiegeln sich I/O Adr-Bereiche. Beispiel:

Bei nur 8 Ports brauchen nur 3 Bits zur weiteren Dekodierung verwendet zu werden, was mit einem 74HC138 möglich ist (zB: Bit-0,1,2). Dann wiederholen sich die I/O-Adressen im Abstand von 8:

```
OUT 10H, MASKE, WERT        ` Ausgabe auf 1. ePort
OUT 18H, MASKE, WERT        ` ebenfalls Ausgabe auf 1. ePort
```


Mit der Compileranweisung `USER_EPORT NROFOUT` wird die tatsächlich ausdekodierte Anzahl Ports angegeben, so daß bei der Initialisierung auch nur diese Ports gesetzt werden:

```
USER_EPORT NROFOUT 10H ' <- Anzahl der Output-Ports
```

Die erweiterten Ausgänge werden beim Start des Tiger-BASIC[®]-Moduls initialisiert. Das Bitmuster kann mit dem Kommando **INITIAL** an Ihre Bedürfnisse angepaßt werden:

USER_EPORT INITIAL, *Base*, *String*

Base gibt an, ab welcher physikalischen Adresse initialisiert wird. Im folgenden Beispiel wird mit der Adresse 8 begonnen. Das erste Byte aus der nachfolgenden Liste wird an die physikalische Adresse 8 geschickt.

String ist ein HEX-String mit bis zu 128 Bytes (siehe ‘%’-Zeichen am String-Ende, um einen binären String zu definieren).

Im folgenden Beispiel wird auf jeden Port die physikalische Adresse geschrieben. Unabhängig von der Länge des Init-Strings werden nur soviel Ports initialisiert, wie durch ‘`USER_EPORT NROFOUT, n`’ festgelegt wurde:

```
#INCLUDE DEFINE_A.INC
USER_EPORT NROFOUT 12H ' <- Anzahl der Output-Ports
USER_EPORT INITIAL, 8, "& ' initialisiere ab phys. 8
08090A0B0C0D0E0F10111213"% ' mit diesen 12 Bytes
...
```

Die letzte Adresse für erweiterte Ausgänge wird mit dem Kommando **LASTLADR** angegeben. Alle darauf folgenden Adressen sind Input-Adressen:

USER_EPORT LASTLADR, *Adr*

Adr gibt die letzte logische Adresse für erweiterte Output-Ports an.

Portadresse und Steuerleitungen verlegen

Die Ports für Bus und Steuerleitungen sollten nicht ohne zwingenden Grund verlegt werden. Bei dem Modul BASIC-Tiger[®]-E kann es jedoch notwendig werden, die Steuerleitungen zu verlegen, da diese Pins zusammen mit dem Analogport liegen.

So wird der Steuerport auf Port 8 Bits 0...2 verlegt:

```
#INCLUDE DEFINE_A.INC
USER_EPORT CTRL 7H
USER_EPORT ACLKBMASK, 00000001b
USER_EPORT DCLKBMASK, 00000010b
USER_EPORT INEBMASK, 00000100b
...
' Steuerleitungen auf Port 7
' Bitmaske ACLK, Pin L80
' Bitmaske DCLK, Pin L81
' Bitmaske INE, Pin L82
```

Tastatur anschließen

Die Tastatur belegt pro Spalte eine erweiterte Input-Adresse. Es werden Tastaturen mit bis zu 16 Spalten à 8 Reihen, also 128 Tasten unterstützt. Jede Tastaturspalte kann auch mit einem DIP-Schalter versehen werden und wird dann entsprechend anders behandelt.

Der Tastaturtreiber verwendet einige HC-MOS-IC zur Ansteuerung:

74HC377: als Adress-Latch sowie 1 oder 2 Stück **74HC138**, um 8 oder 16 Spalten anzusteuern. Jede Spalte benötigt eine **Diode** und hat bis zu 8 Tasten.

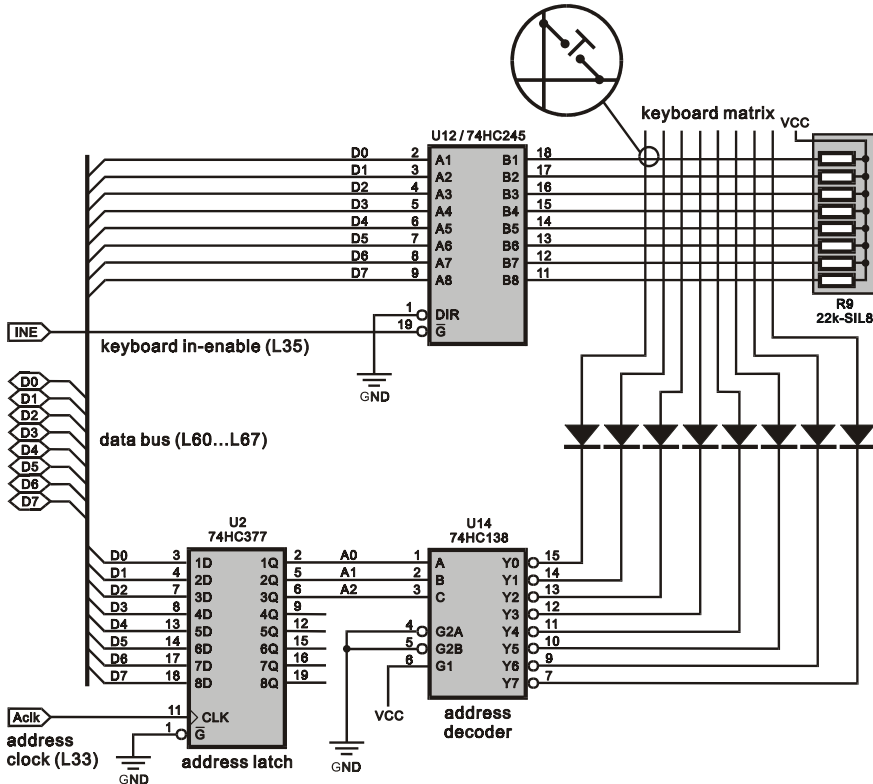
74HC245: um bei low-Signal am 'enable'-Eingang die Reihen der Tastatur auszulesen. Jede Reihe benötigt einen **Pull-up Widerstand** (z.B. 22k). Maximal 8 Reihen sind möglich.

Der Treiber erzeugt auch Töne. Dazu wird ein entsprechendes Ausgabegerät (Summer), z.B. der Beep des Plug & Play Labs, an den Pin L42 (Pin-Nr. 35) angeschlossen.

Die Shift-LED der Tastatur wird an dem ersten Bit der erweiterten Output-Ports angeschlossen.

Erweitertes I/O-System

Die Abbildung zeigt eine einfache Tastatur, unter der Voraussetzung, daß erweiterte Inputs nicht benötigt werden.



Der Device-Treiber ordnet den Tasten zunächst Scan-Codes zu, die sich aus dem Aufbau der Tastaturmatrix ergeben. Nach dem Einbinden des Treibers kann durch ein ESC-Kommando eine Tasten-Zuordnungstabelle aufgebaut werden, so daß alle Tasten den in der jeweiligen Anwendung gewünschten Code erzeugen. Optional kann eine Tastenattribut-Tabelle geladen werden.

Neben den einfachen Tasten-Codes kann eine Tabelle für Tasten mit Shift-Codes angelegt werden. Jede dieser Tabellen besteht aus 128 Bytes.

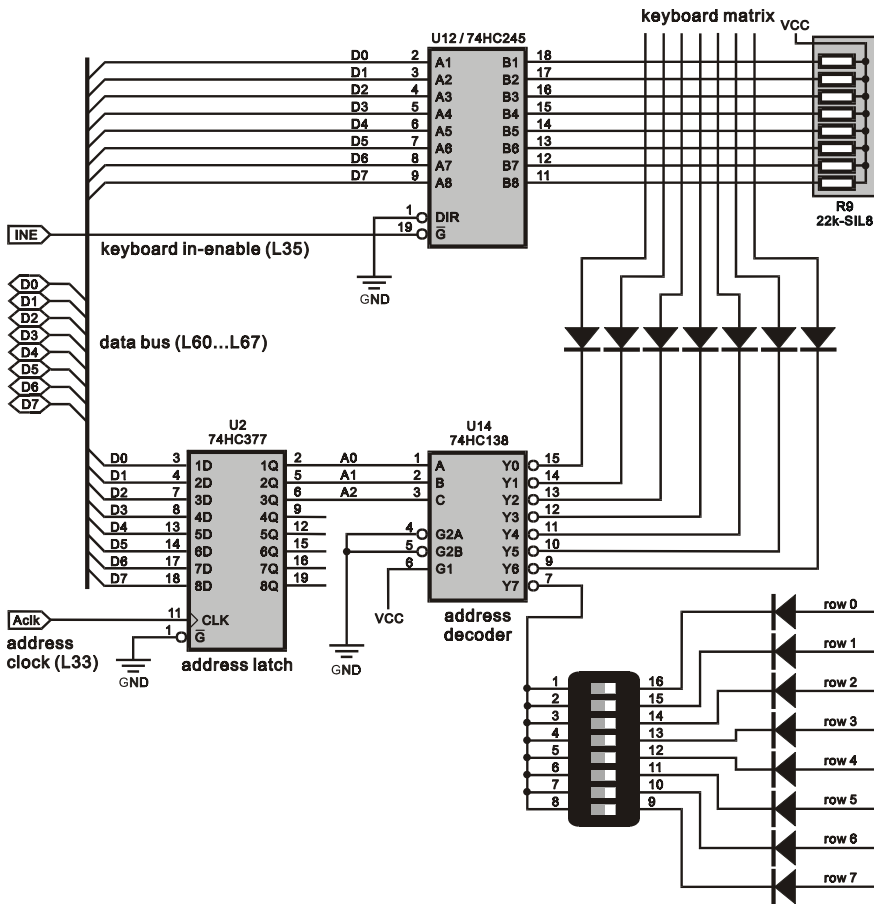
Vor der Verwendung von Shift-Codes muß mindestens eine Taste das Attribut 'Shift' bekommen. Weitere Tastenattribute beeinflussen das Auto-Wiederhol-Verhalten der Tasten.

Eine Taste mit dem Attribut CTRL (STRG) erzeugt einen Versatz von minus 40h (64dez.) zu den Tastencodes der Codierungs-Tabelle.

Bei der Anpassung einer eigenen Tastatur hilft das Programm 'KEY_NO.TIG' aus dem Verzeichnis APPLICATIONS. Auf dem Ausgabegerät werden die Scan-Codes der Tasten angezeigt. So finden Sie die Position jeder Taste in der Code-Tabelle heraus.

Erweitertes I/O-System

Die Schaltung zeigt den Anschluß eines DIP-Schalters an eine Tasterspalte.



Für jeden DIP-Schalter wird eine Diode benötigt, um Kurzschlüsse zwischen den Reihen zu verhindern.

Folgendes Beispiel zeigt, welche Maßnahmen in der Software notwendig sind, um die Tastatur anzupassen. Das Beispiel setzt voraus, dass der physikalische Offset des erweiterten Portsystems -10h ist. In der obigen Schaltung liegt der DIP-Schalter auf der Spalte 8 (Zählung beginnt mit 1).

```
#INCLUDE DEFINE_A.INC
USER_EPORT PHYSOFFS, 0F0h          ` phys. Offset = -10h

TASK MAIN
  INSTALL_DEVICE #LCD, "LCD1.TDD"
  ` ----- setze Spalte 8 auf DIP-Schalter
  PUT #LCD, "<1Bh>D<8><1><1><1><1><1><1><1><0>&
<1><1><1><1><1><1><1><1><0F0h>"
  ` logische Scan-Adressen
  PUT #LCD, "<1Bh>k<90h><91h><92h><93h><94h><95h><96h><97h>&
<90h><91h><92h><93h><94h><95h><96h><97h><0F0h>"
  ` Anpassung der Codes folgt ...

END
```

4

Tastatur anpassen

Die Tastatur wird durch Einbinden des Treibers LCD1.TDD unterstützt. Weitere Informationen zur Anpassung der Belegung der Tastatur sowie der Festlegung von Sonderfunktionen wie Shift, Strg oder DIP-Schalter finden Sie in der Beschreibung des Treibers im Device-Treiber-Handbuch.

Erweitertes I/O-System

Leere Seite

4

Bevor Sie beginnen	1
Installation	2
Entwicklungsumgebung	3
BASIC-Tiger [®] -Steuercomputer	4
Häufig gestellte Fragen - Support	5
Stichwortverzeichnis	6
Anhang	7

Leere Seite

5 Häufig gestellte Fragen

Das Programm wird aus nicht ersichtlichem Grund bei jedem Step neu kompiliert, obwohl im Listing nichts verändert wurde?

Der Compiler vergleicht das Datum Der Source-Datei mit dem Datum der zuletzt kompilierten Datei. Meist tritt dieses Problem auf, wenn Datum oder Uhrzeit verstellt wurden oder Dateien auf einen anderen Rechner transportiert wurden. Zur Abhilfe speichern Sie den Quellcode erneut, erzwingen Sie eine Compilation und laden diese in das Modul. Tritt das Problem in einem Netzwerk auf, bleibt oft nur die Lösung, den Quelltext auf den lokalen Rechner zu holen.

Ich erhalte eine Run-Time-Fehlermeldung, die mit der durch den grünen Balken angezeigten Programmstelle wirklich nichts zu tun haben kann?

Wenn das Run-Time-System einen Fehler zum PC meldet und schließlich anhält, dann steht der grüne Balken bereits auf dem nächsten Befehl (anders als beim Step-pen!). Wenn das Programm mehr als nur eine Task hat, dann kann die Stelle, die den Fehler ausgelöst hat, in der Task liegen, die vor der Stelle des grünen Balkens an der Reihe war. Sie finden die Stelle, wenn Sie mit der Taste F8 von Task zu Task weitersteppen, bis die Fehlermeldung auf eine Zeile vor dem grünen Balken paßt.

Was sind die Unterschiede zwischen BASIC-Tiger® und Tiny-Tiger®?

Tiny-Tiger® ist in einem kleineren Gehäuse untergebracht und ihm fehlen die Pins L40 und L42. Das Gehäuse ist von innen nicht mit Ferrit bedampft wie beim BASIC-Tiger® (aus EMV-Gründen). Abgesehen von diesen Unterschieden ist das Modul TNN-1/1 identisch mit dem BASIC-Tiger®-Modul ANN-1/1. Es ist weiterhin zu berücksichtigen, daß BASIC-Tiger® mit mehr Speicher erhältlich sind. Wenn sich am Ende der Entwicklung eines Projektes herausstellt, daß die Anwendung mehr Speicher braucht als erwartet, dann kann ein 'kleines' BASIC-Tiger®-Modul einfach durch ein 'großes' ersetzt werden. In Platinen-Layouts kann übrigens relativ leicht ein Steckplatz für wahlweise beide Module vorgesehen werden. Ebenso kann Tiny-Tiger® über einen Adaptersockel in einem BASIC-Tiger®-Steckplatz stecken. Allerdings wird dann etwas mehr Höhe gebraucht.

Wieviel Strom können die I/O-Pins des Moduls liefern oder ziehen?

Wenn alle Pins gleichmäßig belastet werden, sollten 1mA pro Pin nicht wesentlich überschritten werden (<1,5mA). Wenn maximal 8 Pins belastet sind, dann können diese bis zu 3,5mA ziehen oder liefern.

Was mache ich mit dem Pin 1 beim Modul ANN-X/X?

Mit VCC verbinden oder offen lassen.

Häufig gestellte Fragen

Müssen BASIC-Tiger® einen externen Watch-Dog haben?

Natürlich spricht nichts gegen eine zusätzliche Sicherheit durch einen externen Watch-Dog, aber im BASIC-Tiger® und Tiny-Tiger® ist bereits ein Watchdog enthalten.

Ich setze Port-Pins zum Beispiel mit der OUT-Instruktion, aber an den Pins tut sich nichts. Was ist der Grund?

Die Ports eines BASIC-Tiger® Moduls können grundsätzlich für ganz verschiedene Zwecke eingesetzt werden. Mit Instruktionen wie IN und OUT kann direkt auf die Ports eines Moduls zugegriffen werden - sofern diese Pins nicht schon anderweitig belegt sind. Die Mechanismen, die außerdem Ports verwenden sind: Device-Treiber sowie das ePort-System zur Erweiterung auf bis zu 1920 zusätzliche digitale Ein- oder Ausgänge.

Verwendet z.B. ein Device-Treiber wie „LCD1.TDD“ einen Port-Pin, so kann der Treiber bei Bedarf ihn für alle anderen Zugriffe (mit IN oder OUT) sperren. Es ist zwar noch erlaubt auf einen solchen Port mit OUT eine Ausgabe auszuführen, die reservierten Pins bleiben jedoch vollkommen unverändert. Dies ist ein Schutzmechanismus um die richtige Funktion der angeschlossenen Peripherie-Geräte zu sichern und es vereinfacht die Programmierung, da man selbst keine ausdrückliche Vorkehrung treffen muß um solche Pins unverändert zu lassen

Im Plug & Play Lab sind die Pins L60...L67 und L30...L37 zur Ansteuerung erweiterter Ports belegt und können somit nicht mit IN oder OUT angesprochen werden.

*Sie können auf die Aktivität erweiterter Ports mit der Compiler-Anweisung **USER_EPORT ACT, NOACTIVE** verzichten (Siehe Seite 318ff). Dann sind diese Pins wie ursprünglich erwartet ansprechbar.*

Wie spreche ich die erweiterten Outputs auf dem Plug & Play Lab an?

Genauso wie die internen Ports, lediglich mit anderer Adresse. Der mögliche Adress-Bereich für erweiterte Ports liegt zwischen: 10H...FFH (16...255). Auf dem Plug & Play Lab liegen die 8 ePort-Ausgänge im Adress-Bereich: 10H...17H (16...23). Oberhalb der Tastatur sind alle 8 x 8 Ausgangs-Bits dieser ePorts durch ein LED vertreten. Ein Befehl zur Ausgabe eines Bytes auf den untersten ePort (10H) könnte also lauten: OUT 10H,0FFH,data_byte.

Sind RAM oder Flash-Speicher über den Bus erweiterbar?

Nein. BASIC-Tiger®-Module sind autonome Computer Module, nicht CPU-Bausteine. BASIC-Tiger® haben keinen nach außen geführten Bus für Speicher-Erweiterungen, man setzt jeweils ein entsprechendes Modul mit der gewünschten Speichergöße ein. Dies stellt die hohe Zuverlässigkeit und gute EMV-Werte sicher.

In manchen Anwendungsfällen kann es jedoch sinnvoll sein zusätzlichen Speicher (RAM, ROM, Disks,...) anzuschließen z.B. um sehr große Daten-Mengen aufzunehmen oder bewegliche Speicher zu verwenden. Dies kann entweder über einen eigenen

Device-Treiber, I2C-Bus oder über entsprechende BASIC-Subroutinen erreicht werden.

Kann man auf die Hardware direkt zugreifen via PEEK und POKE?

Die Instruktionen PEEK und POKE erlauben den Zugriff auf freie Bereiche des FLASH-Speichers. Üblicherweise werden hier Daten abgelegt, die auch nach AUS und EIN schalten verfügbar sein sollen. Dies können z.B. Calibrierungs-Tabellen, registrierte Meßwerte, Datenbanken, Betriebs-Stundenzähler etc. sein. Tiger-BASIC® stellt dem Programmierer den nicht vom BASIC-Programm belegten FLASH-Bereich für diesen Zweck zur Verfügung.

Entsprechend der Eigenschaft des FLASH-Speichers stehen jeweils Bereiche in ganzen Blöcken zur Verfügung. Tiger-BASIC® sorgt automatisch dafür, daß ein BASIC-Programm sich niemals versehentlich selbst löschen kann. Die Zahl und Größe der freien Sektoren hängt vom Modultyp und der Programm-Größe ab. Der FLASH-Speicher wird stets in ganzen Sektoren gelöscht (Inhalt = FFh). Weitere Hinweise zur Verwendung des FLASH siehe Seite 181.

Warum werden nach dem Einschalten oder einem Reset die DIP-Schalter zunächst falsch gelesen?

Es wird stets das Ergebnis des letzten Scans gelesen. Nach einem Reset muss ein Scan abgewartet werden, also ca. 20msec.

Unterstützt BASIC-Tiger® die 1-wire-Chips von Dallas?

Wir empfehlen zur Unterstützung der 1-wire-Chips den ebenfalls von Dallas gelieferten Chip DS2480, der eine Umwandlung von RS-232 nach 1-wire vornimmt.

Häufig gestellte Fragen

Rat und Hilfe

Funktioniert einmal etwas nicht wie erwartet? – Dann nehmen Sie Kontakt auf zum Tiger Support Team, das Ihnen gerne mit Rat und Hilfe für Ihr Problem zur Verfügung steht.

Zur Selbsthilfe sei auch auf den Abschnitt ‚Tiger-BASIC® für Fortgeschrittene‘ hingewiesen, wo häufig gemachte Fehler kurz zusammengefasst sind.

Damit es möglichst schnell geht, senden uns Sie bei jeder Anfrage mindestens diese Informationen und Files zu:

- Lauffähiges Programm-Beispiel als „TIG“ File:
- Reduziert auf ein möglichst einfaches Programm-Beispiel bei dem der Fehler auftritt. Maximal sollte dabei eine Seite herauskommen, meist sogar nur ein paar Zeilen.
- Service-Informationen: Senden Sie uns auch die Datei, die Tiger-BASIC® mit dem Befehl ‚Service-Informationen‘ aus dem Menü ‚Hilfe‘ als Fenster öffnet.
- Welches Tiger-Modul setzen Sie ein (Typ und Version)?
- Beschreiben Sie den Fehler so genau wie möglich.
- In welchem Zusammenhang tritt das Problem auf?
- Tritt es immer nachvollziehbar auf oder nur gelegentlich?
- Wie sind Sie am besten erreichbar: E-mail, Fax- und Telefonnummern für ev. Rückfragen und Antwort nicht vergessen!

BASIC-Tiger®-Service-Hotline, Mo...Fr. 8.00–17.00 Uhr:

0241 / 15 15 99

Wilke Technology GmbH
Krefelder Str. 147
Postfach 1727

D-52070 Aachen / Germany

Tel: 0241 / 918 900
Fax: 0241 / 918 9044
eMail: support@wilke.de

Bevor Sie beginnen	1
Installation	2
Entwicklungsumgebung	3
BASIC-Tiger [®] -Steuercomputer	4
Häufig gestellte Fragen - Support	5
Stichwortverzeichnis	6
Anhang	7

Leere Seite

6 Stichwortverzeichnis

—A—

Aclk	305
ACT	320
Adressclock	305
Analog-Eingänge	119, 143
Analogverstärker	118
ANSI-Steuersequenzen	344
ASCII-Tabelle	341

—B—

BASIC Tiger® Modul A	185
BASIC Tiger® Modul E	201
BASIC Tiger® Modul T (Tiny)	193
BASIC-Tiger®-CAN-Modul	213
Batteriepufferung	102, 132
Baudot-Code	343
Beep	115, 140
Beep-Pin bei Tiny-Tiger®	147

—C—

Compiler-Fehlermeldungen	85
--------------------------------	----

—D—

Darlington-Transistoren	113, 139
Datenclock	305
Dclk	305
Debuggen	69
Debug-Mode	103, 133, 156
Device-Treiber	11
DIP-Schalter	326

—E—

EBCDIC Codes	342
Editor	
Einrücken	78
Leerzeichen	78
Editormodus	78
Editor-Tabulatoren	78
Einrücken, automatisch	78
Erw. Bus-System	125, 146, 164
Erw. I/O-Ports	
Software	318
Erweitertes I/O-System	305
Erweiterung der Ausgänge	310

Stichwortverzeichnis

Erweiterung der Eingänge 313

—F—

Fehlermeldungen, -Compiler 85

Funktionen 12

—G—

GET 12

—I—

I/O-Pins, erweitertes I/O-System 305

INE 305

in-enable 305

INITIAL 321

INPUT 12

INPUT_LINE 12

—J—

J12 Belegung 128

—K—

Kennzeichnung R+C 350

—L—

LASTLADR 321

LED Statusanzeige 162

—M—

Mikrofon 117

Multitasking 11

—N—

NROFOUT 320

—O—

Offset der phys. ePort Adressen 320

Output-Ports

erweiterte I/O-Ports initialisieren
..... 321

—P—

PC-Mode 156

PC-Mode-Pin 103, 133, 156

PHYSOFFS 320

Pinbelegung

BASIC-Tiger®-Modul A 188

EP1 223

EP10 287

EP11 297

EP2 233

EP3 243

EP4 253

EP5 265

EP6 275

TCAN 215

TINY Tiger® 196

TINY Tiger® Module E 204

Pinbelegung 'Serl 0'-Stiftleiste ... 137

Pinbelegung DB15 136

Pinbelegung DB9 108, 135

Pinbelegung des

Tastaturanschlusses 150, 168

Pinbelegung J2 170

Pinbelegung J23 152

Plug & Play Lab 99

Power-down 103, 133, 156

Power-on 103, 133, 156

Power-Verstärker 122

PRINT 12

PRINT_USING 12

Priorität 40

Prototyping-Board 129

PUT 12

PWM-Verstärker 120, 144

—R—

Relais 111

Reset 103, 133, 156

RS-232-Treiber 104, 134, 157

RUN-Mode 103, 133, 156

—S—

Schriftgröße 78

Schutz der Software im Modul 80

Serielle Anschlüsse 104, 134, 157

Sicherungsdatei 48

Softwareschutz im Modul 80

Spannungsversorgung - BASIC-

Tiger® Prototyping-Board 131

Spannungsversorgung -

Plug & Play Lab 101

Spannungsversorgung - TINY-

Tiger® Prototyping-Board 155

—T—

Tabulatoren, Editor 78

Stichwortverzeichnis

Tastatur - Hardware323
Tastenklick..... 115, 141
TCAN Pinbelegung.....215
Terminal.....171
Tiger-Shortcuts348
TINY-Tiger®193
Tiny-Tiger®-Pinbelegung196
Tiny-Tiger®-Prototyping-Board ..153
Treibertransistoren161

—U—

USER_EPORT

ACT320
 erw I/O-Ports abschalten.....320
 erweiterte I/O-Ports318
 INITIAL erweiterte I/O-Ports
 initialisieren.....321
 LASTLADR321
 NROFOUT320
 PHYSOFFS320
 USER_SECURITY80

—W—

Windows-Shortcuts346

Bevor Sie beginnen	1
Installation	2
Entwicklungsumgebung	3
BASIC-Tiger [®] -Steuercomputer	4
Häufig gestellte Fragen - Support	5
Stichwortverzeichnis	6
Anhang	7

Leere Seite

7 Anhang

ASCII-Tabelle

ASCII, gesprochen 'aski', ist eine Abkürzung für **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange'. Der ASCII-Code ist wahrscheinlich der am weitesten verbreitete Code zur Darstellung von Buchstaben, Ziffern, einigen Sonderzeichen und Steuerzeichen. Er findet im PC, Macintosh und Internet Verwendung.

CHAR	HEX	DEC	CHAR	HEX	DEC	CHAR	HEX	DEC	CHAR	HEX	DEC
NUL	00	000	SP	20	032	@	40	064		60	096
SOH	01	001	!	21	033	A	41	065	a	61	097
STX	02	002	"	22	034	B	42	066	b	62	098
ETX	03	003	#	23	035	C	43	067	c	63	099
EOT	04	004	\$	24	036	D	44	068	d	64	100
ENQ	05	005	%	25	037	E	45	069	e	65	101
ACK	06	006	&	26	038	F	46	070	f	66	102
BEL	07	007	'	27	039	G	47	071	g	67	103
BS	08	008	(	28	040	H	48	072	h	68	104
HT	09	009	)	29	041	I	49	073	i	69	105
LF	0A	010	*	2A	042	J	4A	074	j	6A	106
VT	0B	011	+	2B	043	K	4B	075	k	6B	107
FF	0C	012	,	2C	044	L	4C	076	l	6C	108
CR	0D	013	-	2D	045	M	4D	077	m	6D	109
SO	0E	014	.	2E	046	N	4E	078	n	6E	110
SI	0F	015	/	2F	047	O	4F	079	o	6F	111
DLE	10	016	0	30	048	P	50	080	p	70	112
D1	11	017	1	31	049	Q	51	081	q	71	113
D2	12	018	2	32	050	R	52	082	r	72	114
D3	13	019	3	33	051	S	53	083	s	73	115
D4	14	020	4	34	052	T	54	084	t	74	116
NAK	15	021	5	35	053	U	55	085	u	75	117
SYN	16	022	6	36	054	V	56	086	v	76	118
ETB	17	023	7	37	055	W	57	087	w	77	119
CAN	18	024	8	38	056	X	58	088	x	78	120
EM	09	025	9	39	057	Y	59	089	y	79	121
SUB	1A	026	:	3A	058	Z	5A	090	z	7A	122
ESC	1B	027	;	3B	059	[	5B	091	{	7B	123
FS	1C	028	<	3C	060	\	5C	092		7C	124
GS	1D	029	=	3D	061	]	5D	093	}	7D	125
RS	1E	030	>	3E	062	^	5E	094	~	7E	126
US	1F	031	?	3F	063	_	5F	095	DEL	7F	127

Anhang - EBCDIC Codes

EBCDIC Codes

EBCDIC, gesprochen 'eb-si-dik', ist eine Abkürzung für **E**xtended **B**inary-**C**oded **D**ecimal **I**nterchange **C**ode'. EBCDIC ist ein IBM-Code zur Darstellung von Buchstaben, Ziffern und einigen Sonderzeichen. Obwohl der Code auf großen IBM-Computern weit verbreitet ist, verwenden die meisten anderen Computer (PCs, Macintosh) den ASCII-Code.

Char	Hex	Dec	Char	Hex	Dec	Char	Hex	Dec	Char	Hex	Dec
blank	40	64	a	81	129	A	C1	193	0	F0	240
.	4B	75	b	82	130	B	C2	194	1	F1	241
<	4C	76	c	83	131	C	C3	195	2	F2	242
(	4D	77	d	84	132	D	C4	196	3	F3	243
+	4E	78	e	85	133	E	C5	197	4	F4	244
	4F	79	f	86	134	F	C6	198	5	F5	245
&	50	80	g	87	135	G	C7	199	6	F6	246
!	5A	90	h	88	136	H	C8	200	7	F7	247
\$	5B	91	i	89	137	I	C9	201	8	F8	248
*	5C	92	j	91	145	J	D1	209	9	F9	249
)	5D	93	k	92	146	K	D2	210			
;	5E	94	l	93	147	L	D3	211			
-	60	96	m	94	148	M	D4	212			
/	61	97	n	95	149	N	D5	213			
,	6B	107	o	96	150	O	D6	214			
%	6C	108	p	97	151	P	D7	215			
_	6D	109	q	98	152	Q	D8	216			
>	6E	110	r	99	153	R	D9	217			
?	6F	111	s	A2	162	S	E2	226			
:	7A	122	t	A3	163	T	E3	227			
#	7B	123	u	A4	164	U	E4	228			
@	7C	124	v	A5	165	V	E5	229			
'	7D	125	w	A6	166	W	E6	230			
=	7E	126	x	A7	167	X	E7	231			
"	7F	127	y	A8	168	Y	E8	232			
			z	A9	169	Z	E9	233			

Baudot-Code-Satz

Baudot, gesprochen 'bodoh', wurde 1874 erstmals in einem ,Telegraph' (Fernschreiber) eingesetzt, ist aber bis heute in einigen Bereichen zur datenübertragung in Gebrauch. Die Zeichen ,LTRS=Letters' und ,FIGS=Figures' schalten zwischen den beiden Zeichensätzen um. Das rechts-stehende niederwertige Bit (LSB) wird zuerst gesendet. Den Codes werden entweder Buchstaben oder Sonder- und Steuerzeichen zugeordnet.

LTRS	FIGS	HEX	BITS
A	-	03	00011
B	?	19	11001
C	:	0E	01110
D	\$	09	01001
E	3	01	00001
F	!	0D	01101
G	&	1A	11010
H	STOP	14	10100
I	8	06	00110
J	'	0B	01011
K	(	0F	01111
L	)	12	10010
M	.	1C	11100
N	,	0C	01100
O	9	18	11000
P	0	16	10110
Q	1	17	10111
R	4	0A	01010
S	BELL	05	00101
T	5	10	10000
U	7	07	00111
V	;	1E	11110
W	2	13	10011
X	/	1D	11101
Y	6	15	10101
Z	"	11	10001
n/a	n/a	00	00000
CR	CR	08	01000
LF	LF	02	00010
SP	SP	04	00100
LTRS	LTRS	1F	11111
FIGS	FIGS	1B	11011

ANSI-Steuersequenzen

Die in der Tabelle gelisteten ANSI-Steuersequenzen sind nützlich, wenn ein serielles ANSI-Terminal bzw. ein PC mit Terminalprogramm mit ANSI-Emulation angeschlossen ist. Auf dem Bildschirm lassen sich so die Ausgaben positionieren und eventuell farblich darstellen. Die Liste erhebt keinen Anspruch auf Vollständigkeit.

ANSI-Sequenzen beginnen mit ESC gefolgt von „[“. Es folgen weitere Bytes als Kommando, Parameter und eventuell ein Abschluß. In der Tabelle wird die Darstellung `<0>` für ein Byte verwendet, so wie in Strings in Tiger-BASIC[®] üblich. Folgende Abkürzungen gelten:

`<zb>` = Zeile als Binärwert
`<sb>` = Spalte als Binärwert
`<nb>` = binärer Wert.

Wird ein Wert nicht angegeben, wird der Wert ‚eins‘ angenommen. In Tiger-BASIC[®] wird eine ANSI-Sequenz zum Beispiel so ausgegeben (Cursor in Zeile 2, Spalte 3):

PUT #1, “<27>[<2>;<3>H“

ANSI-Sequenz	Funktion
ESC [2J	Löscht den Bildschirm und setzt den Cursor auf die Home -Position.
ESC [<zb>;<sb>H	Bewegt den Cursor an die angegebene Position (Home ist <1>;<1>)
ESC [<zb>;<sb>f	Bewegt den Cursor an die angegebene Position (Home ist <1>;<1>)
ESC [<nb>A	Bewegt den Cursor um n Zeilen nach oben, ohne die Spalte zu beeinflussen, bis maximal Position 1.
ESC [<nb>B	Bewegt den Cursor um n Zeilen nach unten, ohne die Spalte zu beeinflussen, bis maximal zur letzten Zeile.
ESC [<nb>C	Bewegt den Cursor um n Zeilen nach rechts, ohne die Zeile zu beeinflussen, bis maximal zur letzten Position.
ESC [<nb>D	Bewegt den Cursor um n Zeilen nach links, ohne die Zeile zu beeinflussen, bis maximal zur Position 1.
ESC [6n	Gerät gibt den Cursorpositions-Report in dieser Form aus: ESC [<zb>;<sb>R
ESC [s	Gerät speichert die aktuelle Cursorposition.
ESC [u	Gerät bewegt den Cursor an die zuletzt gespeicherte Position.
ESC [K	Löscht den Inhalt einer Zeile ab der Cursorposition bis zum Ende der Zeile.
ESC [<nb>;...;<nb>m	Funktion 'Set Graphic Rendition' schaltet grafische Attribute. Siehe Tabelle im Anschluß

Anhang - ANSI-Steuersequenzen

ESC-m-Parameter	Funktion
0	Alle Attribute aus
1	Fett ein
2	Schwach ein
3	Kursiv ein
5	Blinken ein
6	Schnelles Blinken ein
7	Invers ein
8	Verdeckt ein
30...47	Verschiedene Vorder- und Hintergrundfarben

Windows 95/98/NT Shortcuts

Die in der Tabelle stellt eine Übersicht über die wichtigsten Windows 95/98/NT Shortcuts dar. Die Liste erhebt keinen Anspruch auf Vollständigkeit.

Eingabe	Funktion
F1	Hilfe aufrufen
F10	Menüleiste (de)aktivieren
SHIFT + F10	Kontextmenü aufrufen
CTRL + C	Kopieren
CTRL + X	Ausschneiden
CTRL + V	Einfügen
CTRL + A	Alles markieren
CTRL + S	Speichern
CTRL + N	Neues Fenster öffnen
CTRL + Z	Letzte Aktion widerrufen
CTRL + ESC	Startmenü aufrufen
CTRL + F4	Aktuelles Unter-Fenster in Programm schließen
CTRL + TAB	Vorwärts Unter-Fenster / Registrierkarten wechseln
CTRL + SHIFT + TAB	Rückwärts Unter-Fenster / Registrierkarten wechseln
CTRL + O	Öffnen
CTRL + P	Drucken
CTRL + Pos1	Zum Anfang eines Dokuments
CTRL + Ende	Zum Ende eines Dokuments
CTRL + WIN + F	Suche Computer
CTRL + ALT + DEL	Öffnet Fenster „Anwendung schließen“

Anhang - Windows 95/98/NT Shortcuts

Eingabe	Funktion
WIN + TAB	Umschalten zwischen Programmen in Task-Leiste
WIN + E	Explorer starten
WIN + F1	Hilfe aufrufen
WIN	Startmenü aufrufen
WIN + BREAK	System-Eigenschaften aufrufen
WIN + D	Minimieren bzw. Wiederherstellen aller Fenster
WIN + M	Minimieren aller Fenster
WIN + SHIFT + M	Zurücknehmen minimieren aller Fenster
WIN + R	Dialogfeld „Ausführen“ aufrufen
WIN + F	Dialogfeld „Suchen“ aufrufen
ALT + TAB	Schaltet zum nächsten Programm in der Taskleiste
ALT + SHIFT + TAB	Schaltet zum vorigen Programm in der Taskleiste
ALT + F4	Programm(fenster) schließen
ALT + F10	Menüleiste einer Anwendung aktivieren
ALT + SPACE	Systemmenü der aktuellen Anwendung anzeigen
ALT + ESC	Wechselt zwischen Explorer und anderen anderen Anwendungen
ALT + -	Systemmenü des aktuellen Unterfensters anzeigen

Short-Cuts Tiger-BASIC® Version 5

Eingabe	Funktion
Pfeiltasten	Ein Zeichen vor oder zurück
Pfeiltasten	Eine Zeile vor oder zurück
STRG-Pfeiltasten	Ein Wort vor oder zurück
Bild-auf-, Bild-ab-Taste	Eine Bildschirmseite vor oder zurück
STRG-Pos1	An den Anfang des Textes
STRG-Ende	An das Ende des Textes
STRG-F7/F8	Zum nächsten/vorigen Fehler
Bildlaufleisten	Schnell vor oder zurück
Suchfunktion	An eine bestimmte Stelle im Text
Strg-S	Datei-Speichern
Strg-Z	Bearbeiten-Rückgängig
Strg-X	Bearbeiten-Ausschneiden
Strg-C	Bearbeiten-Kopieren
Strg-V	Bearbeiten-Einfügen
Strg-A	Bearbeiten-Alles markieren
Strg-F	Suchen-Suche nach
Strg-R	Suchen-Ersetzen
F3	Suche-Wiederholen
Strg-G	Gehe zu Zeile
Strg-F9	Anzeige Meldungen
Strg-M	Anzeige-Tigerstatus
ALT-F5	Anzeige-Ausdruck auswerten
Strg-F5	Anzeige-Überwachte Ausdrücke
Strg-W	Anzeige-Überwachte Ausdrücke aktualisieren
Strg-N	Anzeige-zu überwachten Ausdrücken hinzufügen
F4	Starten-Compilieren

Anhang - Short-Cuts Tiger-BASIC® Version 5

Eingabe	Funktion
F5	Starten-Ausführen
Strg-L	Starten-Programm laden
Strg-D	Starten-Programm löschen
F6	Debug-nächste Anweisung (in Task)
ALT-F6	Debug-Anweisungen ausführen (in Task)
F7	Debug-Unterprogramm überspringen (in Task)
ALT-F7	Debug-Unterprogramme überspringen (in Task)
F8	Debug-nächste Anweisung (überall)
ALT-F8	Debug-Anweisungen ausführen (überall)
Strg-T	Debug-bis zum Cursor ausführen
Strg-H	Debug-Programm anhalten
Strg-E	Debug-Programm beenden (Reset)
F2	Debug-Haltepunkt umschalten
ALT-F2	Debug-Haltepunkt hinzufügen
Strg-K	Debug-Alle Haltepunkte löschen
Strg-B	Debug-Editierschutz ein-/ausschalten
Strg-F3	Optionen-Übertragung
F1	Hilfe

Anhang - Kennzeichnung von Widerständen und Kondensatoren

Kennzeichnung von Widerständen und Kondensatoren

Kennzeichnung von Widerständen / Kondensatoren mit Angabe des Temperatur-Koeffizienten (DIN-41429 / DIN-IEC-62 / IEC 115-1-4.5)

Farbcodes

Farbe	Ziffer	Multiplikator	Toleranz	Temp.-Koeff.
schwarz	0	10^0	-	$\pm 250 * 10^{-6}/K$
braun	1	10^1	$\pm 1\%$	$\pm 100 * 10^{-6}/K$
rot	2	10^2	$\pm 2\%$	$\pm 50 * 10^{-6}/K$
orange	3	10^3	-	$\pm 15 * 10^{-6}/K$
gelb	4	10^4	-	$\pm 25 * 10^{-6}/K$
grün	5	10^5	$\pm 0,5\%$	$\pm 20 * 10^{-6}/K$
blau	6	10^6	$\pm 0,25\%$	$\pm 10 * 10^{-6}/K$
violett	7	10^7	$\pm 0,1\%$	$\pm 5 * 10^{-6}/K$
grau	8	10^8	-	$\pm 1 * 10^{-6}/K$
weiß	9	10^9	-	-
silber	-	10^{-2}	$\pm 10\%$	-
gold	-	10^{-1}	$\pm 5\%$	-

Anhang - Kennzeichnung von Widerständen und Kondensatoren

Wertkennzeichnung durch Buchstaben

(DIN 1301/12.93, EN 60 062/10.94)

Kennbuchstabe	Name	Multiplikator
y	Yocto	10^{-24}
z	Zepto	10^{-21}
a	Atto	10^{-18}
f	Femto	10^{-15}
p	Pico	10^{-12}
n	Nano	10^{-9}
μ	Mikro	10^{-6}
m	Milli	10^{-3}
c	Centi	10^{-2}
d	Dezi	10^{-1}
R,F	-	10^0
da	Deka	10^1
h	Hekto	10^2
K	Kilo	10^3
M	Mega	10^6
G	Giga	10^9
T	Tera	10^{12}
P	Peta	10^{15}
E	Exa	10^{18}
Z	Zetta	10^{21}
Y	Yotta	10^{24}

Anhang - Kennzeichnung von Widerständen und Kondensatoren

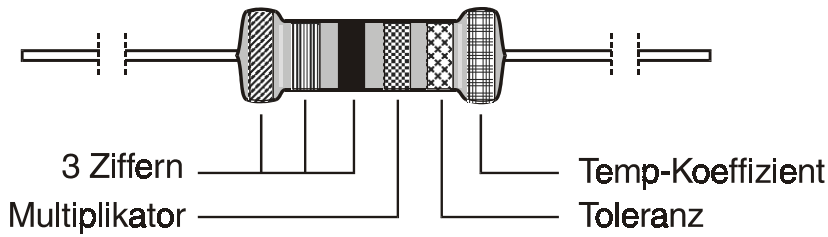
Toleranzkennzeichnung durch Buchstaben

(EN 60 062/10.94)

Abweichungen	
Toleranz	Buchstabe
Symmetrische Abweichung:	
±0,1%	B
±0,25%	C
±0,5%	D
±1%	F
±2%	G
±5%	J
±10%	K
±20%	M
±30%	N
Unsymmetrische Abweichung:	
+30...-10%	Q
+50...-10%	T
+50...-20%	S
+80...-20%	Z
Symmetrische Abweichung für Kapazitätswerte < 10pF:	
±0,1 pF	B
±0,25pF	C
±0,5pF	D
±1pF	F

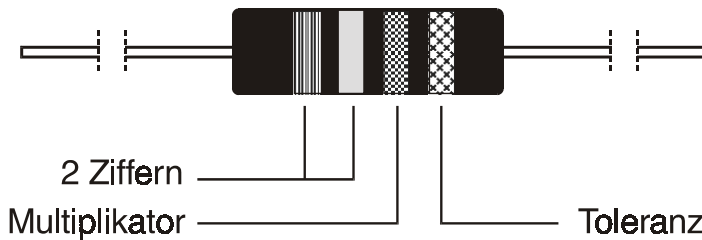
Anhang - Kennzeichnung von Widerständen und Kondensatoren

Anbringung der Farbcodes an Widerständen:



Beispiel: rot-gelb-weiß-rot-braun-rot

24,9 kOhm, +/-1%, +/-50 ppm/°K



Beispiel: gelb-violett-orange-gold

47 kOhm, +/-5%

7

Anhang - Kennzeichnung von Widerständen und Kondensatoren

Mittlere Schrittweite für Widerstands-Wachstum aufeinanderfolgender Werte:

E3	E6	E12	E24	E48	E96	E192
+115%	+47,8%	+21,2%	+10,07%	+4,914%	+2,428%	+1,206%

Normreihen für Widerstandswerte

E3	E6 ±20%	E12 ±10%	E24 ±5%	E48 ±2%	E96 ±1%	E192 ±0,5%
100	100	100	100	100	100	100
						101
					102	102
						104
				105	105	105
						106
					107	107
						109
			110	110	110	110
						111
					113	113
						114
				115	115	115
						117
					118	118
		120	120			120
				121	121	121
						123
					124	124
						126
				127	127	127
						129
			130		130	130
						132
				133	133	133
						135
					137	137
						138

Anhang - Kennzeichnung von Widerständen und Kondensatoren

E3	E6 ±20%	E12 ±10%	E24 ±5%	E48 ±2%	E96 ±1%	E192 ±0,5%
				140	140	140
						142
					143	143
						145
				147	147	147
						149
	150	150	150		150	150
						152
				154	154	154
						156
					158	158
			160			160
				162	162	162
						164
					165	165
						167
				169	169	169
						172
					174	174
						176
				178	178	178
		180	180			180
					182	182
						184
				187	187	187
						189
					191	191
						193
				196	196	196
						198
			200		200	200
						203
				205	205	205
						208
					210	210
						213

Anhang - Kennzeichnung von Widerständen und Kondensatoren

E3	E6 ±20%	E12 ±10%	E24 ±5%	E48 ±2%	E96 ±1%	E192 ±0,5%
				215	215	215
						218
220	220	220	220		221	221
						223
				226	226	226
						229
					232	232
						234
				237	237	237
			240			240
					243	243
						246
				249	249	249
						252
					255	255
						258
				261	261	261
						264
					267	267
		270	270			271
				274	274	274
						277
					280	280
						284
				287	287	287
						291
					294	294
						298
			300	301	301	301
						305
					309	309
						312
				316	316	316
						320
					324	324
						328

Anhang - Kennzeichnung von Widerständen und Kondensatoren

E3	E6 ±20%	E12 ±10%	E24 ±5%	E48 ±2%	E96 ±1%	E192 ±0,5%
	330	330	330	332	332	332
						336
					340	340
						344
				348	348	348
						352
					357	357
			360			361
				365	365	365
						370
					374	374
						379
				383	383	383
						388
		390	290		392	392
						397
				402	402	402
						407
					412	412
						417
				422	422	422
						427
			430		432	432
						437
				442	442	442
						448
					453	453
						459
				464	464	464
470	470	470	470			470
					475	475
						481
				487	487	487
						493
					499	499
						505

Anhang - Kennzeichnung von Widerständen und Kondensatoren

E3	E6 ±20%	E12 ±10%	E24 ±5%	E48 ±2%	E96 ±1%	E192 ±0,5%
			510	511	511	511
						517
					523	523
						530
				536	536	536
						542
					549	549
						556
		560	560	562	562	562
						569
					576	576
						583
				590	590	590
						597
					604	604
						612
			620	619	619	619
						626
					634	634
						642
				649	649	649
						657
					665	665
						673
	680	680	680	681	681	681
						690
					698	698
						706
				715	715	715
						723
					732	732
						741
			750	750	750	750
						759
					768	768
						777

Anhang - Kennzeichnung von Widerständen und Kondensatoren

E3	E6 ±20%	E12 ±10%	E24 ±5%	E48 ±2%	E96 ±1%	E192 ±0,5%
				787	787	787
						796
					806	806
						816
		820	820	825	825	825
						835
					845	845
						856
				866	866	866
						876
					887	887
						898
			910	909	909	909
						920
					931	931
						942
				953	953	953
						965
					976	976
						998

Anhang - Kennzeichnung von Widerständen und Kondensatoren

Leere Seite

BASIC-Tiger® Modul A – Nutzung der Pins

Übersicht der I/O-Pin-Nutzung der wichtigsten Device-Treiber. Einige Funktionen sind fest an den entsprechenden Pin gebunden (PWM, PLSO1, PLSIN1), andere sind lediglich die Standardbelegung, lassen sich aber auf andere Pins verlegen (LCD, Strobe, Busy):

Funktion	Pin-Bez.	Pin-Nr	Pin-Nr	Pin-Bez.	Funktion
	res.	1	46	VCC	
D0	L60	2	45	Batt.	
D1	L61	3	44	AGND	
D2	L62	4	43	Vref	
D3	L63	5	42	An3	
D4	L64	6	41	An2	
D5	L65	7	40	An1	
D6	L66	8	39	An0	
D7	L67	9	38	Alarm	
Busy	L70	10	37	L41/PC	
Strobe	L71	11	36	L40	
PWM0	L72	12	35	L42	Beep
PWM1	L73	13	34	L37	LCD1-RS
LCD-WR	L80	14	33	L36	LCD1-E
LCD-RD	L81	15	32	L35	INE
LCD-CE	L82	16	31	L34	Dclk
LCD-CD	L83	17	30	L33	Aclk
PLSIN1	L84	18	29	L95	RTS
	L85	19	28	L94	Rx1
PLSO1	L86	20	27	L93	Tx1
	L87	21	26	L92	CT0
	Reset-in	22	25	L91	Rx0
	GND	23	24	L90	Tx0

Anhang - BASIC-Tiger® Modul A – Nutzung der Pins

Leere Seite

Anhang - BASIC-Tiger® Modul A – Nutzung der Pins

Freie Tabelle BASIC-Tiger®-Modul A als Kopiervorlage. Projekt:

Funktion	Pin-Bez.	Pin-Nr	Pin-Nr	Pin-Bez.	Funktion
	res.	1	46	VCC	
	L60	2	45	Batt.	
	L61	3	44	AGND	
	L62	4	43	Vref	
	L63	5	42	An3	
	L64	6	41	An2	
	L65	7	40	An1	
	L66	8	39	An0	
	L67	9	38	Alarm	
	L70	10	37	L41/PC	
	L71	11	36	L40	
	L72	12	35	L42	
	L73	13	34	L37	
	L80	14	33	L36	
	L81	15	32	L35	
	L82	16	31	L34	
	L83	17	30	L33	
	L84	18	29	L95	
	L85	19	28	L94	
	L86	20	27	L93	
	L87	21	26	L92	
	Reset-in	22	25	L91	
	GND	23	24	L90	

Anhang - BASIC-Tiger® Modul A – Nutzung der Pins

Leere Seite

TINY-Tiger® – Nutzung der Pins

Übersicht der I/O-Pin-Nutzung der wichtigsten Device-Treiber. Einige Funktionen sind fest an den entsprechenden Pin gebunden (PWM, PLSO1, PLSIN1), andere sind lediglich die Standardbelegung, lassen sich aber auf andere Pins verlegen (LCD, Strobe, Busy):

Funktion	Pin-Bez.	Pin-Nr	Pin-Nr	Pin-Bez.	Funktion
D0	L60	1	44	VCC	
D1	L61	2	43	Batt.	
D2	L62	3	42	Vref	
D3	L63	4	41	AGND	
D4	L64	5	40	An3	
D5	L65	6	39	An2	
D6	L66	7	38	An1	
D7	L67	8	37	An0	
Busy	L70	9	36	L41/PC	
Strobe	L71	10	35	res.	
PWM0	L72	11	34	Alarm	
PWM1	L73	12	33	L37	LCD1-RS
LCD-WR	L80	13	32	L36	LCD1-E
LCD-RD	L81	14	31	L35	INE
LCD-CE	L82	15	30	L34	Dclk
LCD-CD	L83	16	29	L33	Aclk
PLSIN1	L84	17	28	L95	RTS
	L85	18	27	L94	Rx1
PLSO1	L86	19	26	L93	Tx1
	L87	20	25	L92	CT0
	Reset-in	21	24	L91	Rx0
	GND	22	23	L90	Tx0

Anhang - TINY-Tiger® – Nutzung der Pins

Leere Seite

Anhang - TINY-Tiger® – Nutzung der Pins

Freie Tabelle TINY-Tiger® als Kopiervorlage. Projekt:

Funktion	Pin-Bez.	Pin-Nr	Pin-Nr	Pin-Bez.	Funktion
	L60	1	44	VCC	
	L61	2	43	Batt.	
	L62	3	42	Vref	
	L63	4	41	AGND	
	L64	5	40	An3	
	L65	6	39	An2	
	L66	7	38	An1	
	L67	8	37	An0	
	L70	9	36	L41/PC	
	L71	10	35	res.	
	L72	11	34	Alarm	
	L73	12	33	L37	
	L80	13	32	L36	
	L81	14	31	L35	
	L82	15	30	L34	
	L83	16	29	L33	
	L84	17	28	L95	
	L85	18	27	L94	
	L86	19	26	L93	
	L87	20	25	L92	
	Reset-in	21	24	L91	
	GND	22	23	L90	

Anhang - TINY-Tiger® – Nutzung der Pins

Leere Seite

TINY-Tiger® Modul E – Nutzung der Pins

Übersicht der I/O-Pin-Nutzung der wichtigsten Device-Treiber. Einige Funktionen sind fest an den entsprechenden Pin gebunden (PWM, PLSO1, PLSIN1), andere sind lediglich die Standardbelegung, lassen sich aber auf andere Pins verlegen (LCD, Strobe, Busy):

Funktion	Pin-Bez.	Pin-Nr	Pin-Nr	Pin-Bez.	Funktion
D0	L60	1	28	VCC	
D1	L61	2	27	L37	LCD1-RS
D2	L62	3	26	L36/An3	LCD1-E
D3	L63	4	25	L35/An2	INE
D4	L64	5	24	L34/An1	Dclk
D5	L65	6	23	L33/An0	Aclk
D6	L66	7	22	L41/PC	
D7	L67	8	21	L85	
LCD-WR	L80	9	20	Reset-in	
LCD-RD	L81	10	19	L94	Rx1
LCD-CE	L82	11	18	L93	Tx1
LCD-CD	L83	12	17	L92/L86	CT0/PLSO1
PLSIN1	L84	13	16	L91/L87	Rx0
	GND	14	15	L90	Tx0

Anhang - TINY-Tiger® Modul E – Nutzung der Pins

Leere Seite

Anhang - TINY-Tiger® Modul E – Nutzung der Pins

Freie Tabelle TINY-Tiger®-Modul E als Kopiervorlage. Projekt:

Funktion	Pin-Bez.	Pin-Nr	Pin-Nr	Pin-Bez.	Funktion
	L60	1	28	VCC	
	L61	2	27	L37	
	L62	3	26	L36/An3	
	L63	4	25	L35/An2	
	L64	5	24	L34/An1	
	L65	6	23	L33/An0	
	L66	7	22	L41/PC	
	L67	8	21	L85	
	L80	9	20	Reset-in	
	L81	10	19	L94	
	L82	11	18	L93	
	L83	12	17	L92/L86	
	L84	13	16	L91/L87	
	GND	14	15	L90	

Anhang - TINY-Tiger® Modul E – Nutzung der Pins

Leere Seite

BASIC-Tiger®-CAN-Modul – Nutzung der Pins

Übersicht der I/O-Pin-Nutzung der wichtigsten Device-Treiber. Einige Funktionen sind fest an den entsprechenden Pin gebunden (PWM, PLSO1, PLSIN1), andere sind lediglich die Standardbelegung, lassen sich aber auf andere Pins verlegen (LCD, Strobe, Busy). **Die B-Reihe ist nicht aufgeführt**, da die meisten Pins unbelegt sind und die belegten Pins keine Variable Funktion zulassen:

Funktion	Pin-Bez.	Pin-Nr	Pin-Nr	Pin-Bez.	Funktion
	res.	1	46	VCC	
D0	L60	2	45	Batt.	
D1	L61	3	44	AGND	
D2	L62	4	43	Vref	
D3	L63	5	42	An3	
D4	L64	6	41	An2	
D5	L65	7	40	An1	
D6	L66	8	39	An0	
D7	L67	9	38	Alarm	
Busy	L70	10	37	L41/PC	
Strobe	L71	11	36	L40	
PWM0	L72	12	35	L42	Beep
PWM1	L73	13	34	L37	LCD1-RS
LCD-WR	L80	14	33	L36	LCD1-E
LCD-RD	L81	15	32	L35	INE
LCD-CE	L82	16	31	L34	Dclk
LCD-CD	L83	17	30	L33	Aclk
PLSIN1	L84	18	29	L95	RTS
	L85	19	28	L94	Rx1
PLSO1	L86	20	27	L93	Tx1
		21	26	L92	CT0
	Reset-in	22	25	L91	Rx0
	GND	23	24	L90	Tx0

Anhang - BASIC-Tiger®-CAN-Modul – Nutzung der Pins

Leere Seite

Anhang - BASIC-Tiger®-CAN-Modul – Nutzung der Pins

Freie Tabelle BASIC-Tiger®-CAN-Modul als Kopiervorlage.

Die B-Reihe ist nicht aufgeführt, da die meisten Pins unbelegt sind und die belegten Pins keine Variable Funktion zulassen: Projekt:

Funktion	Pin-Bez.	Pin-Nr	Pin-Nr	Pin-Bez.	Funktion
	res.	1	46	VCC	
	L60	2	45	Batt.	
	L61	3	44	AGND	
	L62	4	43	Vref	
	L63	5	42	An3	
	L64	6	41	An2	
	L65	7	40	An1	
	L66	8	39	An0	
	L67	9	38	Alarm	
	L70	10	37	L41/PC	
	L71	11	36	L40	
	L72	12	35	L42	
	L73	13	34	L37	
	L80	14	33	L36	
	L81	15	32	L35	
	L82	16	31	L34	
	L83	17	30	L33	
	L84	18	29	L95	
	L85	19	28	L94	
	L86	20	27	L93	
		21	26	L92	
	Reset-in	22	25	L91	
	GND	23	24	L90	

