

Bedienungsanleitung

C-CONTROL ROBOTER PROJECT 5

Best.-Nr. XXXXXX

Wichtig! Unbedingt lesen!

Bevor Sie den C-CONTROL ROBOTER PROJECT 5 (CCRP5) oder angeschlossene Geräte in Betrieb nehmen, lesen Sie bitte diese Anleitung vollständig durch! Sie erläutert Ihnen die korrekte Verwendung und weist auf mögliche Gefahren hin.

Für Schäden, die aus der Nichtbeachtung dieser Anleitung resultieren, besteht keinerlei Garantieanspruch und übernimmt Conrad Electronic keine Haftung

Inhalt

Wichtig! Unbedingt lesen!	1
Inhalt	2
Einleitung	5
Garantie.....	5
Service	5
Produktbeschreibung.....	6
Bestimmungsgemäße Verwendung.....	6
Sicherheitshinweise	6
Leistungsmerkmale.....	7
Fahrgestell und Antrieb:.....	7
Sensoren:.....	7
Steuer-Computer.....	8
Betriebsbedingungen.....	8
Handhabung.....	8
Allgemeines zum Betrieb	8
Elektrostatische Entladungen.....	8
Versorgungsspannung.....	9
Schreiben eines Programms für den Roboter.....	9
Programmierung des Roboters.....	9
Inbetriebnahme	10
Software-Installation	10
Bereitstellen der Versorgungsspannung.....	10
Verbindung des Roboters zum PC.....	10
Laden von Programmen in den Roboter.....	10
Die erste Funktionskontrolle.....	10
Laden der Akkus.....	11
Programmierung von CCRP5	11
EINFÜHRUNGEN_CC BASIC für C-Control Neulinge.....	11
Beispiele auf der CD	11
EINFÜHRUNGEN_P5 für C-Control Profis.....	12
Beispiele auf der CD	12
Erweiterte Systemressourcen	13
POWER SWITCHES	13
ACS.....	13
IRCOMM	14
NAV	16
LED CONTROL.....	16
DRIVE-CONTROL	16
Betrieb bei 12 Mhz (overclocking).....	17

Allgemeines.....	17
Einschränkungen.....	17
C-Control I/O Ressourcen	18
Belegte C-Control I/O Ressourcen	18
Freie C-Control I/O Ressourcen	18
Die Programmiersprache CCBASIC	18
Echtzeituhr.....	19
User-Bytes	19
Digitalports.....	19
Analogports	19
Aktivantenne.....	20
Frequenz Ein/Ausgänge.....	20
Interrupteingang.....	20
Expansion-Connector	20
Programmierung des C-Control Computers	20
Die Beispiele zum Erlernen von CC-BASIC	20
Was ist ein Programm?	21
Grundlegende Elemente von CCBASIC	21
Allgemeines.....	21
Bezeichner	21
Variablen und Konstanten	22
Label.....	22
Terme	22
Operanden und Operatoren	22
Funktionen.....	22
Zuweisungen.....	22
Befehle	23
Anweisungen zur Steuerung des Programmflusses	23
Compileranweisungen	23
Definition symbolischer Konstanten	23
Definition von Variablen	23
Definition von Digitalports.....	24
Definition von Analogports	25
Mathematische und logische Operatoren	25
Rangfolge von Operatoren und Funktionsaufrufen	26
Anweisungen zur Steuerung des Programmflusses	26
Kommunikation über die serielle Schmittstelle	29
Dateifunktionen	31
Portbefehle.....	32
Definition und Anwendung von Datentabellen	33
Die Echtzeituhr	34
Interner Timer, Tonerzeugung, Frequenzmessung.....	34
Einbinden von Maschinenprogrammen	35
Problemlösungen.....	36
Es lässt sich kein Programm laden	36
Die Programme lassen sich laden, aber es tut sich nichts beim Start	36
Der Roboter führt im Betrieb unkontrolliert RESETs aus	36
Im Interruptbetrieb treten unerklärliche Funktionsstörungen auf.....	36
Verschieden Sensoren zeigen falsche oder keine Werte.....	36

Bedienelemente , Anzeigen und Steckverbinder.....	37
Expansion-Connector 1.....	37
Expansion-Connector 2.....	38
Technische Daten.....	38

Einleitung

Wir danken Ihnen für Ihre Entscheidung zum Erwerb von *CCRP5*. Dieser mobile Roboter ist mit einem programmierbaren Kleincomputer versehen, der es Ihnen ermöglicht grundsätzliche Verhaltensweisen und Reaktionen des Roboters auf externe Reize selbst zu bestimmen. *CCRP5* wurde von uns mit dem Anspruch entwickelt, die hohen Erwartungen unserer Kunden an Qualität und Funktionalität zu erfüllen.

Conrad Electronic GmbH
D-92240 Hirschau

Garantie

Bevor Sie *CCRP5* oder angeschlossene Geräte in Betrieb nehmen, lesen Sie bitte diese Anleitung vollständig durch. Sie erläutert Ihnen die korrekte Verwendung und weist auf mögliche Gefahren hin. Für Schäden, die aus der Nichtbeachtung der Bedienungsanleitung resultieren, besteht keinerlei Garantieanspruch und übernehmen wir keine Haftung!

Hinweise zur beschränkten Garantie und Haftung

Das Herz des Roboters ist ein C-Control/BASIC Steuercomputer. Die im Mikroprozessor MC68HC05B6 als ROM-Maske integrierte und die zugehörigen PC-Software wird in vorliegender Form geliefert.

Conrad Electronic übernimmt keine Garantie dafür, daß die Leistungsmerkmale individuellen Ansprüchen entsprechen, oder daß die Software im Mikroprozessor und die PC-Software in jedem Fall unterbrechungs- und fehlerfrei arbeiten. Der Anwender trägt das gesamte Risiko bezüglich der Qualität und der Leistungsfähigkeit des Gerätes inklusive aller Software.

Conrad Electronic garantiert die Funktion der mitgelieferten Applikationsbeispiele unter Einhaltung der in den technischen Daten spezifizierten Bedingungen. Sollte sich der Roboter oder die PC-Software darüberhinaus als fehlerhaft oder unzureichend erweisen, so übernimmt der Kunde alle entstehenden Kosten für Service, Reparatur oder Korrektur.

Die Gewährleistung von Conrad Electronic beschränkt sich ausschließlich auf den Austausch des Gerätes innerhalb der Garantiezeit bei offensichtlichen Defekten an der Hardware, wie mechanischer Beschädigung, fehlender oder falscher Bestückung elektronischer Bauteile, ausgenommen gesockelter integrierter Schaltkreise und Steckbrücken. Es besteht keine Haftung für Schäden, die unmittelbar durch oder in Folge der Anwendung des Roboters entstehen.

Unberührt davon bleiben Ansprüche, die auf unabdingbaren gesetzlichen Vorschriften zur Produkthaftung beruhen.

Jeder *Roboter* verläßt das Werk in einwandfreiem und funktionsgeprüften Zustand!

Conrad Electronic bietet für *CCRP5* eine **Gewährleistungsdauer von 24 Monaten**. Innerhalb dieser Zeit werden eventuelle Transportschäden bei der Auslieferung, Fertigungsmängel oder Ausfälle am Gerät kostenfrei behoben.

Sollten die Leistungsmerkmale *des Roboters* Ihren individuellen Ansprüchen nicht genügen, nutzen Sie bitte unsere **Geld-Zurück-Garantie von 14 Tagen**. Senden Sie das Gerät innerhalb dieser Zeit ohne Gebrauchsspuren und in der Originalverpackung zur Erstattung des Warenwertes oder zur Verrechnung zurück. Alle Fristen gelten ab Datum der Rechnung beziehungsweise des Kassenbons.

Conrad Electronic übernimmt keine Haftung für Folgeschäden an Sachwerten oder Personen, die durch Anwendung des *Roboters* entstehen!

Service

Zu Ihrer Beratung stellt Conrad Electronic Ihnen ein kompetentes Team von Servicemitarbeitern zur Seite. Jede Anfrage wird schnellstmöglich bearbeitet. Spezialfragen werden an die Entwicklungsingenieure CTC weitergeleitet.

Um unnötige Verzögerungen zu vermeiden, möchten wir Sie jedoch bitten, vor einer Anfrage noch einmal diese Anleitung, die Online-Hilfen der Programmiersoftware, die Text- und Beispieldateien und nach Möglichkeit die Informationsseiten im Internet zu studieren. Meist findet sich so schon die Lösung eines Problems! Ihre Anfragen richten Sie bitte an unsere Abteilung Technische Kundenbetreuung.

Brief Conrad Electronic GmbH
 TKB Computer und Meßtechnik
 Klaus-Conrad-Straße 1
 92240 Hirschau

Fax 0180 / 53 12 119
Telefon 0180 / 53 12 116
Internet <http://www.conrad.de>

Produktbeschreibung

Bestimmungsgemäße Verwendung

Dieser mobile Roboter ist mit einem programmierbaren Kleincomputer versehen, der es Ihnen ermöglicht grundsätzliche Verhaltensweisen und Reaktionen des Roboters auf externe Reize selbst zu bestimmen. Der Roboter CCRP5 wurde als Experimentierplattform für den an der Thematik der Robotik interessierten Elektroniker entwickelt. Er veranschaulicht im praktischen Versuch Einflußnahme und Auswirkungen von Softwareparametern sowie über entsprechende Sensorik auch von physikalischen Größen.

Eine andere als die bestimmungsgemäße Verwendung ist nicht zulässig.

Sicherheitshinweise

Lesen Sie diesen Abschnitt besonders aufmerksam durch! Bei Nichtbeachtung der Sicherheitshinweise besteht Lebensgefahr durch einen Stromschlag oder Elektrobrand!

Bedingt durch die offene Bauform gibt es bei CCRP5 **spitze Ecken und scharfe Kanten**. Er darf daher nicht als Spielzeug für Kinder unter 8 Jahren eingesetzt werden. Beaufsichtigen Sie Kinder, die sich beim Betrieb des CCR im Raum befinden. Je nach Programmierung können unvorhergesehene Fahr- und Lenk-bewegungen auftreten.

Bei einem Raupenantrieb gibt es **zwischen Rad und Raupenband Einzugsstellen**. Diese Bereiche sind bei CCRP5 weitgehend durch die Radkästen zwischen den Rädern ausgefüllt und dadurch gesichert. Achten Sie beim Betrieb darauf, daß Sie **nicht mit den Fingern zwischen Rad und Raupenband** geraten. Beaufsichtigen Sie im Raum befindliche Kinder. Betreiben Sie den Roboter nicht, wenn sich frei laufende Kleintiere im Raum befinden.

Achtung ! Je nach Programmierung des Roboters können die Motoren unerwartet anlaufen !

Auf der Oberfläche der Hauptplatine befinden sich nicht abgedeckte Bauteile und Leiterbahnen. Verursachen Sie keine Kurzschlüsse durch versehentlich abgelegte Metallgegenstände oder Werkzeug.

Vor Inbetriebnahme des Roboters müssen alle im Erfassungsbereich befindlichen Flüssigkeitsbehälter wie z.B. Kaffeetassen, Flaschen oder Blumenvasen gesichert oder entfernt werden.

Betreiben Sie den Roboter nicht auf Tischflächen oder Bereichen, auf denen Absturzgefahr besteht. Denken Sie hierbei auch an die Kletterfähigkeit des Roboters.

Leistungsmerkmale

CCRP5 ist ein programmierbarer Kleincomputer der mit zahlreichen Sensoren bestückt und auf einem Raupenfahrgestell montiert ist. CCRP5 ist, entsprechend programmiert, ein voll funktionsfähiger Kleinroboter der auf Umweltreize ansprechen und reagieren kann.

Gleichzeitig bietet CCRP5 eine ideale Basis für eigene Erweiterungen hinsichtlich Sensoren und Aktoren z.B. für Wettbewerbe.

Die Stromversorgung erfolgt durch 6 NiCd Akkumulatoren (oder behelfsweise, mit Einschränkungen, durch 6 hochwertige Alkali-Mangan Batterien)

Fahrgestell und Antrieb:

Der Roboter CCRP5 läuft auf einem Raupenfahrgestell.

Das Fahrgestell besteht aus zwei spiegelbildlich konstruierten Bodenwannenhälften. In diese sind die Antriebsmotoren sowie die Stirnradgetriebe integriert. Die Radachsen bzw. Antriebswellen laufen in Sinterbuchsen.

Als Antriebsmotoren sind hochwertige Industrienmotoren eingesetzt, die sich durch hohe Lebensdauer und einen sehr guten Wirkungsgrad auszeichnen. Dadurch lassen sich mit einer Akkuladung lange Laufzeiten realisieren. Die Getriebe bestehen aus jeweils einem Ritzel mit 12 Zähnen auf der Motorwelle, zwei Stufenrädern mit 50/12 Zähnen und einem Zahnrad mit 50 Zähnen auf der Abtriebswelle. Alle Zahnräder haben modul 0,5 und sind aus verschleißarmen Polyamid gefertigt. Zwei Gabellichtschrangen an der Unterseite des Robot Motherboards greifen über die beiden hochgesetzten Stufenräder und ermöglichen im Zusammenwirken mit kleinen Bohrungen in den Zahnrädern eine sehr genaue Wegmessung. Die Antriebsräder mit Durchmesser 44 mm führen die Raupenbänder, die aus Gummi gefertigt werden. Damit kann der Roboter kleine Hindernisse und große Steigungen mühelos bewältigen.

In der Mitte der Bodenwanne wird der Fahrakku bzw. ein Batteriehalter, der mit Mignon Akkus bestückt wird, eingesetzt.

Das Robot Motherboard wird an der Oberseite mit 4 Schrauben befestigt und muß dann im Normalbetrieb mit Akkus nicht mehr entfernt werden.

Für höchste Mobilität wird das Fahrgestell von zwei , in der Geschwindigkeit getrennt steuerbaren, Elektromotoren angetrieben. Die Laufrichtung jeder Raupe ist schaltbar.

Sensoren:

Folgende Sensoren ermöglichen dem Programmierer sehr komplexe Interaktionen mit Umgebungsreizen und entsprechend vielseitigen Reaktionen darauf.

- 2 richtungsunterscheidende Lichtsensoren
- 2 Wegstreckenmesser mit hoher Auflösung
- ein berührungsloses IR Anti-Kollisionssystem mit schaltbarer Reichweite
- ein leistungsfähiges IR-Kommunikationssystem (Sender/Empfänger)
- Schallsensor
- Berührungssensor
- Sensor für Betriebsspannung
- Stromsensor für die Motoren
- Stromsensor für Ladestrom

Steuer-Computer

Der Steuer-Computer von CCRP5 ist ein Computer der C-Control Serie. Dieser ist ein kompakter Baustein für den universellen Einsatz in Mess-, Steuer- und Regelungsaufgaben und verfügt außerdem über die Fähigkeiten der seriellen Datenübertragung und der Datenspeicherung.

Der Steuercomputer enthält einen weiterentwickelten Mikroprozessor, der die Programmierung des Bausteins in der weitbekannten und leicht zu erlernenden Programmiersprache BASIC erlaubt. So wird der Steuercomputer durch wenige Zeilen BASIC-Quelltext zur intelligenten Alarm-anlage, zum komplexen Datenerfassungssystem, zur Steuerzentrale einer Heizungsanlage oder, wie hier, zum „Hirn“ eines kleinen Robotermodells.

Für den Kontakt zur Außenwelt stehen acht analoge Eingänge, zwei analoge Ausgänge sowie 16 frei als Ein- oder Ausgänge programmierbare Digitalports zur Verfügung.

Ein Teil dieser Ressourcen ist durch Sensorik und Antrieb belegt. Die freien Ressourcen lassen jedoch genügend Spielraum für zusätzliche Erweiterungen, einschliesslich der Verwendung einer leistungsfähigeren CPU (die vorhandene CPU wird dann mit einem speziellen Programm zum CO-Prozessor)

Betriebsbedingungen

Die geschlossene Bodenwanne schützt die Mechanik gut gegen grobe Umwelteinflüsse. Sie ist aber weder wasser- noch staubdicht. Setzen Sie den Roboter daher nur im trockenen und sauberen häuslichen Bereich ein. Schmutz, Staub, Fremdkörper und Feuchtigkeit zerstören die Mechanik.

Die zulässige Umgebungstemperatur darf während des Betriebes 0°C und 40°C nicht unter- bzw. überschreiten.

Betreiben Sie den Roboter nicht in einer Umgebung mit brennbaren oder explosionsgefährdeten Flüssigkeiten, Gasen oder Stäuben.

Der Roboter ist nicht für kommerziellen Einsatz ausgelegt.

Handhabung

Dieses Kapitel gibt einen Überblick über die Handhabung des *Roboters* und der zugehörigen Komponenten. Die nötigen Detailinformationen z.B. zur Programmierung entnehmen Sie bitte den nachfolgenden Kapiteln dieses Handbuchs bzw. den in den Beispielpogrammen enthaltenen Beschreibungen.

Allgemeines zum Betrieb

Elektrostatische Entladungen

Besonders in trockener Luft kann sich der menschliche Körper und auch der Roboter selbst (hier ist die Beschaffenheit des Bodenbelags wesentlich) elektrostatisch aufladen. Beim Kontakt mit leitenden Gegenständen baut sich diese Ladung mit einem kleinen Funken ab. Solche Entladungen beim Berühren elektronischer Bauelemente können diese zerstören. Vor dem Hantieren mit dem Gerät sollten Sie einen großen, geerdeten Gegenstand berühren (z.B.: ein PC-Metallgehäuse, eine Wasserleitung oder ein Heizungsrohr), um eventuelle Aufladungen abzubauen. Eine Entladung des Roboters selbst gegen geerdete Gegenstände ist ungefährlich kann jedoch zu Programmabstürzen oder unkontrollierter Funktion des Roboters führen

Versorgungsspannung

Alle elektrischen Verbindungen von und zum Gerät sind stets vor Anschluß der Versorgungsspannung herzustellen. Das Aufstecken oder Abziehen von Verbindungskabeln oder das Herstellen oder Lösen von Verbindungen können zur Zerstörung des Steuercomputers oder angeschlossener Geräte führen. Zur Versorgung des Roboters ist eine Gleichspannung von 7,2V vorgesehen, die mit 6 NiCd Akkumulatoren erzeugt wird. Verwenden Sie zum Laden der Akkus nur geprüfte Ladegeräte. Behelfsweise können die Akkus mit einem als Zubehör angebotenen Steckernetzteil geladen werden. Ein Anspruch auf optimale Ladung sowie ein Schutz gegen Überladung kann dann aber nicht erhoben werden. Behelfsweise kann der Roboter auch mit 6 hochwertigen Alkali-Mangan Batterien betrieben werden. Wegen des höheren Innenwiderstandes müssen Stromspitzen im Betrieb (z.B. bei abruptem Wechsel der Fahrtrichtung) programmtechnisch vermieden werden.

ACHTUNG:

- **Versuchen Sie niemals den Roboter an eine externe Spannungsversorgung anzuschliessen wenn keine Akkus eingelegt sind**
- **Versuchen Sie niemals den Roboter an eine externe Spannungsversorgung anzuschliessen wenn der EIN/AUS-Schalter auf AUS steht**
- **Schliessen Sie niemals ein anderes, als das als Zubehör empfohlene, Steckernetzteil an**

In jedem dieser Fälle besteht sonst die Gewissheit, Bauteile des Roboters durch Überspannung zu zerstören

Schreiben eines Programms für den Roboter

Für die Entwicklung des Betriebsprogramms für den Roboter steht ihnen ein komfortables Programm zur Verfügung. Dieses Programm ist mit einer Standard-Oberfläche mit Menü und Mausbedienung ausgestattet und ermöglicht die Eingabe von BASIC-Quell-code (Editor), das Übersetzen (Compiler) und das Laden (Lader) des C-Control Codes in den Roboter.

Das von Ihnen erstellte BASIC-Programm, das die Aktionen und Reaktionen des Roboters bestimmt, wird von einem Compiler in eine Folge von Befehlsbytes umgesetzt. Die Befehle und die zugehörigen Parameterbytes werden über die serielle Schnittstelle zum Steuercomputer übertragen, wo sie von dessen Betriebssystem im EEPROM-Speicherchip abgelegt werden.

Durch das C-Control Betriebssystem können Ihre Anwendungsprogramme in sehr kompakter Form gespeichert werden und belegen meist nur wenige hundert der über 8000 zur Verfügung stehenden Bytes. Damit bleibt ein großer Teil des Speicherchips frei und kann zur Aufzeichnung von Daten benutzt werden.

Um alle Ressourcen des Roboters zu nutzen steht ihnen eine Erweiterung des Betriebssystems in Form eines speziellen Treibers in Maschinencode zur Verfügung. Er wird mit dem ersten Beispiel automatisch geladen. Weitere BASIC-Unterprogramme erleichtern Ihnen den Zugriff auf diese Treiber, da oft komplizierte Parameter an den Treiber übergeben werden müssen.

ACHTUNG:

Der erste Schritt in der Initialisierung ihres Programms sollte immer die Zeile

```
REV_L=on:REV_R=on:SYS PLM_SLOW
```

enthalten. Diese Zeile initialisiert die Ports für die Richtungsumschaltung des Antriebs und die PLM Frequenz.

DER BETRIEB DER MOTOREN OHNE DIESE INITIALISIERUNG FÜHRT UNWEIGERLICH ZUR ZERSTÖRUNG DER ANTRIEBSELEKTRONIK !!

Programmierung des Roboters

Prinzipiell ist eine Verbindung von PC und Roboter nur zur Programmierung erforderlich. Anschließend kann der Roboter gestartet werden

Die Verbindung zum PC kann allerdings auch während des Programmablaufs im Roboter bestehen bleiben und z.B. zur Übertragung von Messdaten benutzt werden. Die Bewegungsfreiheit des Roboters ist dann natürlich

stark eingeschränkt, sodass diese Möglichkeit wohl zur zum Zwecke der Fehlersuche im Programm zur Anwendung kommt.

Nach Betätigung des Starttasters beginnt das Betriebssystem, die Befehle nacheinander aus dem Speicher zu lesen und abzuarbeiten, bis zum Programmende-Befehl.

Inbetriebnahme

Software-Installation

Mit dem Roboter bekommen Sie eine Installations-CD die alle benötigten Programme und Beispiele enthält. Die Anweisungen zur Installation der Software entnehmen Sie bitte der Datei INSTALL.TXT.

Konfigurieren Sie nun die Integrierte Entwicklungsumgebung (IDE) entsprechend Ihren Wünschen und der zugewiesenen Schnittstelle.

Die mit der IDE installierten Beispiele zu CCBASIC sind auf dem Roboter nicht funktionsfähig

Die Beispiele zu CCBASIC und für den Betrieb des Roboters müssen zusätzlich installiert werden.

Bereitstellen der Versorgungsspannung

- Stellen Sie sicher, dass der Roboter keine Verbindung zum PC hat.
- Stellen Sie **unbedingt** sicher, dass der EIN/AUS-Schalter auf AUS (nach vorne gekippt)steht

Entfernen Sie die vier Befestigungsschrauben der Hauptplatine und heben Sie diese **vorsichtig** ab. Legen Sie 6 (wenigstens gering vorgeladene) Mignon NiCd-Akkus **polungsrichtig** in den Batteriehalter. Wenn Sie sich von der richtigen Position der Akkus im Halter überzeugt haben, so montieren Sie die Hauptplatine wieder auf das Fahrgestell.

Verbindung des Roboters zum PC

Mit dem Roboter wurde Ihnen ein Schnittstellenkabel ausgeliefert, ein 9-poliges Nullmodemkabel (ca. 1,5 Meter) Stecken Sie nun das Nullmodemkabel an eine freie serielle Schnittstelle Ihres Computers. Viele Computer verfügen über eine 9-poligen und eine 25-polige serielle Schnittstelle. Sollte bei Ihrem Computer nur noch eine 25-polige Schnittstelle frei sein, benötigen Sie einen zusätzlichen Adapter.

Stecken Sie anschließend den 3-poligen Verbinder am Ende des Schnittstellenkabels polungsrichtig auf die Steckerleiste des Roboters. Das Kabel muß so aufgesteckt werden, dass die weiße Leitung auf der Seite der LEDs ist.

Laden von Programmen in den Roboter

Programme werden direkt aus der Entwicklungsumgebung in den Roboter geladen. Der Roboter muss dazu eingeschaltet sein und im RESET-Modus sein. Im Menü „Entwicklung“ finden sie die entsprechende Auswahl für den Download. Danach können Sie das Programm starten (START-Taste).

Wenn sie ein Programm geladen haben, welches den Antrieb aktiviert, sollten Sie vorher den Schnittstellenstecker am Roboter abziehen.

Die erste Funktionskontrolle

Schalten Sie jetzt den EIN/AUS-Schalter des Roboters auf EIN und drücken Sie die RESET-Taste auf der Platine. CCRP5 ist jetzt bereit für den Download des ersten Programms, einer kleinen Funktionskontrolle.

Laden Sie das Programm 1_EINFÜHRUNG_LEDS.BAS in die IDE. Wählen Sie dann im Menü „Entwicklung“ den Befehl „BASIC-Compiler“. Das Programm wird nun übersetzt (compiliert). Achten Sie dabei auf das Bildschirmsfenster „Meldungen“. Dort erscheinen die Meldungen des Compilers über Verlauf und Erfolg der Übersetzung. Wenn Fehler im BASIC-Quelltext enthalten sind, werden diese im Meldungsfenster aufgelistet.

In diesem Beispiel sollten keine Fehler auftreten,

Nach korrekter Übersetzung wählen Sie im Menü „Entwicklung“ den Befehl „in C-ControlUnit übertragen“. Die vom Compiler erzeugten Codes werden dann zum C-Control Steuercomputer, dem Herz des Roboters, übertragen. Meldungen über Erfolg oder Fehler bei der Übertragung erscheinen wieder im Meldungsfenster.

Wenn die Übertragung fehlerfrei erfolgt ist, drücken Sie nun den START-Taster auf der Platine um das Programm zu starten. Es erzeugt mit den vier roten LEDs ein Lauflicht, der Roboter ist bereit für den nächsten Schritt.

ACHTUNG:

Überspringen Sie dieses Beispiel nicht, weil es einen wichtigen Systemtreiber ladet.

Laden der Akkus

Wenn sie sicher sind, dass die Akkus voll geladen sind, können Sie diesen Schritt überspringen oder, wenn es nötig ist, später darauf zurück kommen.

Zur Versorgung des Roboters ist eine Gleichspannung von 7,2V vorgesehen, die mit 6 NiCd Akkumulatoren erzeugt wird. Verwenden Sie zum Laden der Akkus nur geprüfte Ladegeräte. Behelfsweise können die Akkus mit einem als Zubehör angebotenen Steckernetzteil geladen werden.

Wenn Sie sich dafür entscheiden, die Akkus mit dem Steckernetzteil zu laden, stellen wir Ihnen dazu ein spezielles Programm zur Verfügung, das den Ladevorgang anzeigt und weitgehend überwacht.

Ein Anspruch auf optimale Ladung sowie ein Schutz gegen Überladung kann dann aber nicht erhoben werden.

Stoppen Sie das, auf dem Roboter laufende Programm mit der RESET-Taste und öffnen Sie das Programm CHARGE (es ist im Ordner „TOOLS“) in der IDE. Erklärungen und Hinweise am Anfang des Programms beschreiben seine Funktion.

Laden Sie das Programm in den Roboter und starten Sie es. Während des Ladevorgangs haben Sie Gelegenheit mit den Funktionen des Programms zu spielen und sich den Quelltext anzusehen. Haben Sie noch keine Erfahrungen mit C-Control, so finden Sie eine Beschreibung der Programmiersprache im Kapitel „PROGRAMMIEREN MIT CCBASIC“ und einfache Programmbeispiele im Ordner EINFÜHRUNG_CCBASIC auf der CD

ACHTUNG:

- **Versuchen Sie niemals den Roboter an eine externe Spannungsversorgung anzuschliessen wenn keine Akkus eingelegt sind**
- **Versuchen Sie niemals den Roboter an eine externe Spannungsversorgung anzuschliessen wenn der EIN/AUS-Schalter auf AUS steht**
- **Schliessen Sie niemals ein anderes, als das als Zubehör empfohlene Steckernetzteil an**

In jedem dieser Fälle besteht sonst die Gewissheit, Bauteile des Roboters durch Überspannung zu zerstören

Programmierung von CCRP5

EINFÜHRUNGEN_CCBASIC für C-Control Neulinge

Wenn sie noch keine Erfahrung mit C-Control haben, sollten sie zuerst mit dem Kapitel „PROGRAMMIEREN MIT CCBASIC“ und den zugehörigen Beispielen im Ordner „EINFÜHRUNGEN_CCBASIC“ fortfahren.

Beispiele auf der CD

Auf der CD finden Sie 14 Beispiele, von 001.BAS bis 014.BAS, die Ihnen schrittweise die Sprache CC-BASIC erläutern.

EINFÜHRUNGEN_P5 für C-Control Profis

Auf der mitgelieferten CD finden Sie im Ordner „EINFÜHRUNGEN_P5“ eine kleine Einführung die Ihnen schrittweise den Zugriff auf die Systemressourcen von CCRP5 vorstellen, demonstrieren und erklären.

Die Beispiele zu CCRP5 beinhalten immer alle Programmteile zur Ansteuerung der Systemressourcen, weil während der Initialisierung teilweise darauf zurückgegriffen werden muss. Der relevante Code für das Beispiel ist jedoch klar herausgestellt und erklärt.

Bei allen Beispielen ist die letzte Programmzeile auskommentiert, weil sie den Systemtreiber ladet, was aber nur ein einziges mal erforderlich ist und mit dem ersten Beispiel gemacht wurde.

Beispiele auf der CD

1_EINFÜHRUNG_LEDS.BAS

Dieses Beispiel zeigt Ihnen wie die LEDs 1 bis 4 angesteuert werden. Die LEDs sind neben dem Beeper die einzige Möglichkeit zur Ausgabe.

1_EINFÜHRUNG_TOUCHSENSOR_1.BAS

Der Roboter hat als Eingabemöglichkeit einen Berührungssensor. Er ist an einen A/D-Wandler angeschlossen und bietet damit die Möglichkeit mehrere Grade der Berührung zu unterscheiden. Damit Sie einen Anhaltspunkt haben, wie sich die Art der Berührung und der A/D-Wandler Wert entsprechen, wird dieser als Balkenanzeige mit den LEDs angezeigt.

1_EINFÜHRUNG_TOUCHSENSOR_2.BAS

Dieses Beispiel zeigt, wie man diese Erkenntnisse anwendet und damit z.B. eines von vier Programmen auswählt.

2_EINFÜHRUNG_SCHALLSENSOR.BAS

Der Geräuschpegel wird ausgewertet und mit den LEDs angezeigt.

3_EINFÜHRUNG_ACS.BAS

Hier finden Sie Erklärungen zum Subsystem, zu dessen Bestandteilen auch das Anti-Kollisionssystem (ACS) gehört. Das Beispiel zeigt die Funktion des ACS. Benutzen sie das Programm um einen Eindruck zu gewinnen, was das ACS sieht, und was nicht.

3_EINFÜHRUNG_ACS_EMPFINDLICHKEIT.BAS

In diesem Beispiel wird demonstriert, wie die Empfindlichkeit des ACS eingestellt wird und wie weit der Erfassungsbereich geht.

3_EINFÜHRUNG_ACS_INTERRUPT.BAS

Das ACS kann auch im Interrupt-Modus betrieben werden. Wie es geht sehen sie hier.

4_EINFÜHRUNG_ANTRIEB.BAS

Der Roboter ist bereit für den ersten Ausflug. Eine Demo zum Fahren mit ACS zeigt die Grundlagen der Kontrolle über die Antriebsmotoren und wie Kollisionmeldungen in Ausweichmanöver umgesetzt werden

4_EINFÜHRUNG_BEWEGUNGSSENSOR_ACS.BAS

Wie das ACS geschickt als Bewegungssensor eingesetzt wird sehen Sie hier.

4_EINFÜHRUNG_ENTFERNUNGSSENSOR_ACS.BAS

In dieser Demo wird das ACS verwendet einen konstanten Abstand zu einem Objekt einzuhalten, es sozusagen zu bewachen.

5_EINFÜHRUNG_LICHTSENSOR_1 bis 3

Drei Beispiele zum Gebrauch des Lichtsensors z.B. als Bewegungsmelder oder zur Lichtstärkemessung

6_EINFÜHRUNG_WEGSTRECKENZÄHLER.BAS

Gefahrne Wegstrecken zu messen ist ein wesentlicher Punkt für die Orientierung eines Roboters im Raum.

7_EINFÜHRUNG_SPANNUNGSSENSOR.BAS

CCRP5 hat einen Sensor um die Akkuspannung zu überwachen. Informationen zum Gebrauch und Hinweise über die Wichtigkeit erhalten Sie im Beispielprogramm.

8_EINFÜHRUNG_STROMSENSOR.BAS

9_EINFÜHRUNG_IR_COMM.BAS

Grundsätzliches zum Gebrauch des IR-Kommunikationssystems

9_EINFÜHRUNG_IR_COMM_INT.BAS

Grundsätzliches zum Gebrauch des IR-Kommunikationssystems mit Interrupt

9_EINFÜHRUNG_REMOTE_CONTROL.BAS

zeigt die Anwendung des Kommunikationssystems zur Fernsteuerung des Fahrzeugs.

Erweiterte Systemressourcen

Ein wesentlicher Unterschied von CCRP5 gegenüber einer herkömmlichen C-Control-Unit liegt in den erweiterten Hardware-Ressourcen (SUBSYSTEM).

Es handelt sich dabei um einen weiteren 8 Bit Prozessor (mit festem Programm) der die Funktionseinheiten IRCOMM (zur IR-Kommunikation), ACS (das IR-Anti-Kollisionssystem) und NAV (der Wegstreckenzähler zu Navigation) bedient.

Um für den Benutzer genug freie Ports für eigene Erweiterungen zu lassen ist bei CCRP5 ein zusätzlicher (gegen über den Ressourcen die C-Control hat) 8 Bit-Ausgangsport vorhanden. Er stellt im SUBSYSTEM die Funktionseinheiten POWERSWITCHES (Schalten von Systemspannungen) und LED CONTROL (Ansteuerung der LEDs) zur Verfügung.

Der Datenaustausch zwischen C-Control und SUBSYSTEM erfolgt über den SUBSYSTEM INTERFACE-BUFFER, der durch die Bezeichner SUBCMD, HBYTE und LBYTE definiert ist.

Ein spezieller Treiber in Maschinencode (im EEPROM der C-Control) ist für den Datenaustausch zwischen C-Control und dem SUBSYSTEM zuständig. Da die technischen Zusammenhänge teilweise sehr komplex sind, stehen ihnen eine Reihe von vorgefertigten Systemroutinen (in BASIC) zur Verfügung die den Zugriff auf das SUBSYSTEM vereinfachen.

Sie können diese Routinen als festen Bestandteil des Betriebssystems betrachten, da ohne diese wichtige Funktionen nicht genutzt werden können.

Die Systemroutinen belegen die Bytes 1 bis 5. Die restlichen Bytes sind für den Benutzer frei verfügbar.

POWER SWITCHES

Bei abgeschalteten SUBSYSTEM ist die Stromaufnahme von 35 auf 12 mA reduziert. Diese Funktion ist wichtig wenn der Roboter längere Betriebszeit inaktiv verbringen soll und z.B. die Sensoren nur kurzzeitig, in grossen Zeitabständen benutzt werden.

Bei Abgeschalteten Subsystem sind damit verbundene Funktionen nicht mehr verfügbar, Zugriffe darauf führen zu Fehlfunktionen oder Programmabstürzen.

gosub SUBSYS_PWR_ON

gosub SUBSYS_PWR_OFF

ACS

Folgende Routinen gestatten Ihnen komfortablen Zugriff auf die Funktionen des Anti-Kollisionssystems:

gosub NO_ACS_INT	;Interruptbetrieb abgeschaltet
gosub ACS_INT_200	;Interrupt betrieb eingeschaltet
SYS ACS_HI	;ACS HI POWER (SEHBEREICH ca. 60cm) einschalten
SYS ACS_LO	;ACS LO POWER (SEHBEREICH ca. 30cm) einschalten
SYS ACS_MAX	;ACS MAX POWER (SEHBEREICH ca. 100cm) einschalten
SYS COMNAV_STATUS	;Abfrage des ACS/IR-COMM

Ein Aufruf der COMNAV_STATUS Routine ist immer erforderlich, wenn sie über eine Kollisionswarnung oder den Empfang eines IR-Signals informiert werden wollen.

Diese Routine liefert diese Zustände in SYSTEM_STATUS (Byte 5) zurück.

Bit 0 = linker ACS Sensor

Bit 1 = rechter ACS Sensor
 Bit 3= IR-COMM receive flag

Das Bit ist 1 wenn der jeweilige Sensor angesprochen hat, bzw empfangene Daten bereit stehen
 Die Bezeichner für diese Bits sind ACSL_F und ACSR_F sowie IR_F
 Wenn ACS und IRCOMM im Interruptbetrieb laufen, geben Ihnen diese FLAGS Auskunft, was den Interrupt ausgelöst hat. Der Interrupt wird gelöscht wenn das Statusregister mit SYS COMNAV_STATUS gelesen wird.

IRCOMM

Senden/Empfangen von Daten RC5/REC80

IRCOMM sendet und empfängt Daten entsprechend den beiden Formaten. In den Beispielen ist dargestellt, wie der Rahmen für A=30 und C=22 aussieht.

Der RC5 IR-Datenrahmen:

S	S	T	A4	A3	A2	A1	A0	C5	C4	C3	C2	C1	C0
1	1	x	1	1	1	1	0	0	1	0	1	1	0

----- 30 ----- 22 -----

S = Startbit
 T = Toggle (Wird in der Regel bei jedem erneuten Tastendruck auf die Fernbedienung getoggelt)
 A = Adresse
 C = Kommando

Der REC80 IR-Datenrahmen:

S	A4	A3	A2	A1	A0	C6	C5	C4	C3	C2	C1	C0
1	1	1	1	1	0	0	0	1	0	1	1	0

----- 30 ----- 22 -----

S = Startbit
 A = Adresse
 C = Kommando

Folgende Routinen gestatten Ihnen komfortablen Zugriff auf das IR-Kommunikationssystem.

gosub GET_IRDATA ;liefert empfangene IR-Daten in HBYTE (A) und LBYTE (C)
 (jeweils FF wenn nichts empfangen wurde)
gosub SEND_IRDATA ;Sendet die Daten in HBYTE (A) und LBYTE (C)
gosub REC80 ;Stellt das Format ein
gosub REC80_INT ;Aktiviert den Interruptbetrieb mit REC80 Format
gosub RC5 ;Stellt das Format ein
gosub RC5_INT ;Aktiviert den Interruptbetrieb mit RC5 Format
SYS COMNAV_STATUS ;Abfrage des ACS/IRCOMM (siehe ACS)

Die Formate lassen für Adresse und Kommando folgenden Wertebereich zu:

LBYTE Byte 1 (Wert 0...63 - RC5) (Wert 0...127 - REC 80)
 HBYTE Byte 2 (Wert 0...63 - RC5) (Wert 0....63 - REC 80)

Beim Zugriff auf IRCOMM gelten folgende Vereinbarungen:

- 1) Der Zugriff wird bis zu 20ms verzögert ausgeführt, wenn IRCOMM gerade ein IR-Signal empfängt oder sendet.
- 2) Zu anderer Zeit wird der Zugriff sofort ausgeführt und ist nach 3ms beendet. In dieser Zeit kann kein IR-Signal empfangen werden und geht evtl. verloren
- 3) IRCOMM kann mit gosub GET_IRDATA direkt abgefragt werden. Ist jedoch das ACS in Betrieb (und damit häufige Zugriffe mit SYS COMNAV_STATUS auf das Subsystem notwendig), ist es besser

das FLAG in SYSTEM_STATUS (siehe ACS) auszuwerten und den empfangenen Code nur im Bedarfsfall abzufragen.

- 4) IRCOMM ist wie das ACS auch interruptfähig. Der Interrupt wird gelöscht, wenn das Statusregister mit SYS COMNAV_STATUS gelesen wird. Das Ergebnis wird an SYSTEM_STATUS übertragen und die Flags geben Auskunft was den Interrupt ausgelöst hat (siehe ACS)

Senden von Telemetriedaten RC5

Wenn man sich das Datenformat ansieht, stellt man fest, dass weder in die ADRESSE, noch in das KOMMANDO ein ganzes Byte passt. Deshalb formatieren die Systemroutinen die Telemetriedaten so, dass das Datenbyte auf c0 bis a1 verteilt ist. In a2 bis a4 und T ist der Messkanal, damit man mehrere Datenkanäle übertragen kann (bis zu 16 Kanäle)

Das sieht dann so aus:

Der modifizierte RC5 IR-Datenrahmen:

S	S	T	A4	A3	A2	A1	A0	C5	C4	C3	C2	C1	C0
1	1	CH3	CH2	CH1	CH0	b7	b6	b5	b4	b3	b2	b1	b0

SYS SEND_TLM ;Formatiert und sendet die Werte in HBYTE und LBYTE
SYS SEND_SYSSTAT ;Sendet SYSTEMSTATUS (feste Kanaluweisung 0)
SYS SEND_SPEEDL ;Sendet die Werte der PLM DA/-Wandler (feste Kanaluweisung 7 und 8)
SYS SEND_SPEEDR
Gosub GET_TLM ;liest Telemetriedaten von IRCOMM und trennt Messkanal und Daten

Der Messkanal (0-15) wird in HBYTE übergeben, in LBYTE steht der Messwert (0-255), der Aufruf von SYS SEND_TLM formatiert die Daten wie oben gezeigt und sendet sie.

Sie können den Kanälen 0 bis 15 beliebige Sensoren zuordnen. Wenn die Telemetriedaten auf der Empfängerseite ausgewertet werden, muss diese Formatierung natürlich wieder rückgängig gemacht werden. Die Programmbeispiele zur Telemetrie stellen eine entsprechende Routine zur Verfügung.

Adressierter Modus

Sind mehrere Roboter in Betrieb und sollen untereinander kommunizieren, so lassen sich die Roboter jeweils mit einer Adresse versehen unter welcher sie sich im Netzwerk identifizieren. Dazu werden die beiden Formate so abgewandelt, dass jeweils eine 3Bit Sender/Empfänger-Adresse (also für bis zu 7 Einheiten) im Rahmen steckt. Damit ist dann noch Platz für ein 6 Bit-Kommando (also 64 Kommandos)

Der modifizierte RC5 IR-Datenrahmen:

S	S	T	A4	A3	A2	A1	A0	C5	C4	C3	C2	C1	C0
1	1	RX2	RX1	RX0	TX2	TX1	TX0	C5	C4	C3	C2	C1	C0

Der modifizierte REC80 IR-Datenrahmen:

S	A4	A3	A2	A1	A0	C6	C5	C4	C3	C2	C1	C0
1	RX2	RX1	RX0	TX2	TX1	TX0	C5	C4	C3	C2	C1	C0

RX ist die Adresse des Empfängers

TX ist die Adresse des Absenders

Ist diese Betriebsart aktiviert, filtert IRCOMM alle Rahmen, die nicht an den Empfänger adressiert sind automatisch aus. Die an das Subsystem, übergebenen Daten müssen aber entsprechend formatiert werden. Auch hier übernehmen kleine Unterprogramme die notwendige Formatierung.

Die RX-Adresse wird in HBYTE, das Kommando in LBYTE übergeben und mit gosub SEND_ADDRESSED_DATA gesendet. Wenn ein Datenrahmen empfangen wird, steht in HBYTE der Absender des Rahmens, in LBYTE das Kommando.

Gosub SET_ADDRESS ;Adressierten Modus einstellen LBYTE =Adresse
gosub GET_ADDRESSED_DATA ;liefert empfangene IR-Daten in HBYTE (TX) und LBYTE (C)

(jeweils FF wenn nichts empfangen wurde)

gosub SEND_ADDRESSED_DATA ;Sendet die Daten in HBBYTE (RX) und LBYTE (C)
gosub REC80 ;Stellt das Format ein
gosub REC80_INT ;Aktiviert den Interruptbetrieb mit REC80 Format
gosub RC5 ;Stellt das Format ein
gosub RC5_INT ;Aktiviert den Interruptbetrieb mit RC5 Format
SYS COMNAV_STATUS ;Abfrage des ACS/IR-COMM (siehe ACS)

NAV

Gefahrenere Distanzen zu messen ist eine wesentliche Voraussetzung für Orientierung und Navigation. Das NAV-System von CCRP5 wird mit folgenden Routinen angesprochen:

gosub CLR_DISTANCE Beide Wegstreckenzähler löschen
gosub L_DISTANCE linken Zähler abfragen
gosub R_DISTANCE rechten Zähler abfragen

Nach Aufruf der Systemroutine steht der Zählerstand in LBYTE und HBYTE. Die Auflösung ist ca. 3cm. Da es ein 16 Bit-Zähler ist, findet erst nach ca. 2km ein Überlauf statt.

LED CONTROL

Auch die LEDs werden nicht über gewöhnliche I/O-Kanäle der C-Control angesteuert. Hier ist ebenfalls ein Spezieller Treiber eingesetzt der für den Benutzer eine einfache Schnittstelle hat. Er erspart es Ihnen einzelne Bits nach komplizierten Regeln zu manipulieren

gosub LED1ON
gosub LED1OFF

Für die LEDs 2 bis 4 ist das Verfahren analog

DRIVE-CONTROL

Die Geschwindigkeit wird über die beiden PLM-Ausgänge der C-Control eingestellt (DA1 und DA2) Und erfordert keinen besonderen Treiber. Aber die Richtungsumschaltung der Motoren ist komfortabler (und im Programmcode kürzer) über Systemroutinen zu bedienen.

SYS REVR ;Kette rückwärts laufen lassen R/L
SYS REVL
SYS FWDR ;Kette vorwärts laufen lassen R/L
SYS FWDL
SYS FWD ;beide Ketten vorwärts laufen lassen
SYS REV ;beide Ketten rückwärts laufen lassen
SYS ROTR ;Auf der Stelle nach R/L drehen
SYS ROTL

ACHTUNG:

Der erste Schritt in der Initialisierung ihres Programms sollte immer die Zeile

REV_L=on:REV_R=on:SYS PLM_SLOW

enthalten. Diese Zeile initialisiert die Ports für die Richtungsumschaltung des Antriebs und die PLM Frequenz.

Initialisieren Sie die Ports auch, wenn der Betrieb der Motoren nicht geplant ist.

DER BETRIEB DER MOTOREN OHNE DIESE INITIALISIERUNG FÜHRT UNWEIGERLICH ZUR ZERSTÖRUNG DER ANTRIEBSELEKTRONIK !!

Betrieb bei 12 Mhz (overclocking)

Prinzipiell besteht die Möglichkeit CCRP5 mit 12 Mhz Systemtakt zu betreiben. Diese Option kann durch Umstecken des Jumpers 5 vor dem Programmstart gewählt werden. Zum Download eines Programms ist der reguläre 4MHz Betrieb unbedingt erforderlich, da sonst die Baudrate der Schnittstelle nicht mehr stimmt.

Allgemeines

Im 12 MHz Betrieb müssen Teile des Systemtreibers für das SUBSYSTEM stark verlangsamt werden, was einen separaten Treiber erfordert. Durch den sehr knapp bemessenen Speicherbereich im EEPROM im Controller entfallen deshalb einige unterstützende Systemroutinen und müssen in Basic ausgeführt werden. Die Systemroutinen SYS SEND_SPEEDR und SYS SEND_SPEEDL (sendet DA1 und DA2 der Motoren als Telemetrie) entfallen. Ausserdem ändern sich die Einstiegsadressen der Systemroutinen.

Sie finden einige der bereits aufgeführten Beispielprogramme im Ordner OVERCLOCKING. Laden Sie zuerst das Beispiel 1_EINFÜHRUNG_12MHZ.BAS, da es den Systemtreiber ladet und deutlich zeigt wie gross der Zuwachs an Geschwindigkeit ist. Weitere Beispiele in diesem Ordner geben Ihnen wichtige Hinweise auf Fehlerquellen im 12MHz Betrieb.

Einschränkungen

CCRP5 wurde mit grosser Sorgfalt entwickelt und getestet. Obwohl der Prozessor problemlos mit 12MHz läuft, kann Conrad Electronic für eine korrekte Funktion oder durch die Übertaktung eventuell entstehenden Schäden keinerlei Garantie oder Haftung übernehmen.

Bedingt durch die Übertaktung ändern sich alle Systemeigenschaften die Zeitmessungen mit dem internen Timer durchführen

Betroffen davon sind die Befehle:

**BAUD
BEEP
PAUSE
FREQ
SECOND
MINUTE
HOUR
DOW
MONTH
YEAR**

Alle Zeiten sind auf 1/3 verkürzt, was aber problemlos bei der Programmierung berücksichtigt werden kann. Den kleinen Nachteilen steht hier ein wirklich enormer Zuwachs an Geschwindigkeit gegenüber. Die Abarbeitung eines BASIC-Befehls dauert jetzt etwa 300us statt 1ms.

Hinweis:

Die 12 MHz Version des Treibers (P5DRIV12.S19) ist kompatibel zum 4 MHz Systemtakt. Sie können also Ihre Programme mit 4MHZ Takt laden und testen, und erst wenn das Programm befriedigend läuft den Jumper auf die Position 12 MHZ stecken. Allerdings ist der Zugriff auf IRCOMM, NAV und ACS bei 4 MHz etwa 1 ms langsamer, als mit der 4MHz Version des Treibers (P5DRIV.S19)

C-Control I/O Ressourcen

Der C-Control Computer stellt dem Benutzer eine grosse Anzahl von Ports zur Verfügung, die bei der M-Unit und Main-Unit frei verfügbar sind. Hier, bei PROJECT5, ist ein Teil dieser Ports für den Betrieb des Roboters notwendig und daher nicht, oder nur eingeschränkt verfügbar.

Belegte C-Control I/O Ressourcen

Hier finden Sie eine Übersicht der I/O-Kanäle die vom Roboter fest oder abschaltbar belegt sind. Abschaltbare Kanäle können Sie für eigene Erweiterungen benutzen, wenn der entsprechende JUMPER abgezogen ist.

PORT P1	COMNAV-INTERFACE/ IOEXPANDER	DATA
PORT P2	COMNAV-INTERFACE /	CLOCK
PORT P3	IOEXPANDER	CLOCK
PORT P4	IOEXPANDER	STROBE
PORT P5	DRIVE CONTROL	DIRECTION A
PORT P6	DRIVE CONTROL	DIRECTION B
DA1	DRIVE CONTROL	SPEED B
DA2	DRIVE CONTROL	SPEED A
AD1	MOTOR CURRENT	
AD2	CHARGE CURRENT	
AD3	BATT VOLTAGE	
AD4	MIC	abschaltbar JP 1
AD5	TOUCHSENSOR	abschaltbar JP 2
AD6	LIGHTSENSOR LEFT	abschaltbar JP 3
AD7	LIGHTSENSOR RIGHT	abschaltbar JP 4
IRQ	SUBSYSTEM INTERRUPT	

Freie C-Control I/O Ressourcen

Hier finden Sie eine Übersicht der I/O-Kanäle die vom Roboter nicht belegt sind, und für eigene Erweiterungen verwendet werden können, bzw. von Zubehör belegt werden. Diese I/O-Kanäle sind, neben anderen Signalen, an den Extension-Ports verfügbar.

PORT P7
PORT P8
PORT P9
PORT P10
PORT P11
PORT P12
PORT P13
PORT P14
PORT P15
PORT P16
AD8
DCF/FREQ1
FREQ2

Die Programmiersprache CCBASIC

Systemressourcen des C-Control Computers

Unter dem Begriff „Systemressourcen“ sind hier alle internen Funktionseinheiten zusammengefaßt, die sich aus den Eigenschaften des Mikrocontrollers ableiten oder durch das auf dem Chip maskenprogrammierte Betriebssystem zur Verfügung gestellt werden. Wie diese Systemressourcen im BASIC-Programm angesprochen werden, wird weiter unten in der Befehlsübersicht beschrieben.

Echtzeituhr

Im Hintergrund des Betriebssystems läuft ein mit 20 Millisekunden getakteter 16-Bit-Timer, dessen Wert jederzeit ausgelesen und zum Herstellen von Zeitbezügen im BASIC-Programm benutzt werden kann und Zeitbasis für die interne Echtzeituhr ist. Diese Uhr kann mit einem DCF 77 Empfänger synchronisiert werden. Die per DCF77 empfangene Zeit- und Datumsinformation wird vom Betriebssystem in sieben interne Speicherzellen (Jahr, Monat, Tag, Wochentag, Stunde, Minute, Sekunde) übertragen und bis zur nächsten Synchronisation in Portionen von 20 Millisekunden erhöht. Die Ganggenauigkeit der Echtzeituhr zwischen den Synchronisationszeitpunkten ist bestimmt durch die Abweichung des 4 MHz-Quarzes von seiner Normalfrequenz von bis zu 0,1 Promille, abhängig von Streuungen in der Serienproduktion und von der Temperatur. Das entspricht einer Abweichung von bis zu 0,36 Sekunden pro Stunde. Nach dem Zuschalten der Betriebsspannung und nach einem Reset startet die Uhr mit dem 01.01.97, 00:00:00 Uhr. Die internen Speicherzellen für Datum und Uhrzeit können vom BASIC-Programm aus gelesen und beschrieben werden. Durch das Beschreiben der Zeitspeicherzellen kann die Uhr also auch ohne DCF77-Empfang gestellt werden. Für Programmtests oder bei geringem Anspruch an die Ganggenauigkeit kann so auf die DCF77-Antenne verzichtet werden.

User-Bytes

Der Mikrocontroller MC68HC05B6 verfügt über insgesamt 240 Bytes RAM. Der C-Control Steuercomputer belegt davon größten Teil für Betriebssystemfunktionen (Stack, Timer, Uhr, DCF77-Rahmenpuffer, Schnittstellenpuffer, Zwischenspeicher für Berechnungen usw.). 24 Bytes stehen dem Anwender zur Verwendung in BASIC-Programmen zur Verfügung. Davon werden die Bytes 1 bis 5 für das erweiterte Betriebssystem von CCRP5 gebraucht und stehen dem Benutzer nicht zur freien Verfügung. Die Verwendung dieser Userbytes ist im Abschnitt zum DEFINE-Befehl weiter unten beschrieben.

Digitalports

Benutzung eines Digitalports als Eingang

Digitaleingänge werden zur Abfrage von Schaltzuständen verwendet. Wird ein Digitalport als Eingang benutzt, führt er im unbeschalteten Zustand undefinierten Pegel der z.B. mit einem PULL UP-Widerstand festgelegt werden muss. Ist beispielsweise ein Reedkontakt an diesem Portangeschlossen, wird bei offenem Schalter eine logische Eins („wahr“) vom Port gelesen, bei geschlossenem Schalter eine logische Null („falsch“). Achten Sie bitte unbedingt darauf, daß je nach Beschaltung des Ports und der logischen Aussage, die Ihr Programm beinhalten soll, der eingelesene Wert eventuell invertiert werden muß (NOT-Operator, siehe Befehlsbeschreibung)!

Benutzung eines Digitalports als Ausgang

Wird ein Digitalport als Ausgang verwendet, können daran nachfolgende ICs, Transistoren oder Low-Current-Leuchtdioden direkt betrieben werden. Der maximal zulässige Laststrom beträgt 10 mA. In jedem Fall ist eine ausreichende Strombegrenzung, zum Beispiel durch einen Widerstand, zu gewährleisten, da es sonst zur Zerstörung des Mikrocontrollers kommen kann! Innerhalb des Mikrocontrollers erfolgt die interne Beschaltung eines Digitalports als Ausgang oder Eingang beim ersten Ausführen des Anwenderprogramms. Nach dem Zuschalten der Betriebsspannung oder nach einem Reset verhalten sich alle Digitalports zunächst elektrisch als Eingang, sie führen also über einen Pullup-Widerstand High-Pegel.

Analogports

A/D-Wandler

Der C-Control/BASIC Steuercomputer verfügt über acht A/D-Ports und zwei D/A-Wandler. Bevor die A/D-Eingänge benutzt werden können, muß eine Referenz-Spannung mit dem Referenzspannungseingang des Controllers verbunden werden. Der angelegte Spannungswert gilt als Obergrenze des Messbereiches der A/D-Wandlung und entspricht dem Wandlungswert 255 (SFF hexadezimal).

Bei Ihrem Roboter ist die Referenzspannung auf 2.50V fest eingestellt. Als Referenz für das untere Ende des Messbereiches der A/D-Wandlung dient stets das Groundpotential (Masse, „Minus“) der Betriebsspannung. An

den freien A/D-Ports können Sensoren aller Art angeschlossen werden, die eine Ausgangsspannung von 0 bis 2.5 Volt liefern. In den meisten Fällen werden hier aktive Sensoren zur Anwendung kommen, um das Signal des eigentlichen Sensorelementes zu verstärken und den Ansprüchen an Auflösung, Linearität und Driftverhalten zu genügen.

D/A-Wandler

Die zwei 8-Bit-D/A-Wandler arbeiten nach dem Prinzip der Pulsweitenmodulation. In einem Zeitabschnitt (Modulationsintervall), der aus 256 Teilabschnitten besteht, wird ein D/A-Ausgang für die Dauer von sovielen Teilabschnitten high-gepulst, wie es dem 8-Bit-Wert entspricht, der zur Ausgabe bestimmt ist. Die Dauer eines Teilabschnittes beträgt Zuz, die des gesamten Modulationsintervalls 512~s (1953 Hz). Zur Demodulation, also Wandlung in ein echtes Analogsignal genügt meist ein einfaches RC-Glied.

Bei Ihrem Roboter werden die D/A-Wandler, oder in diesem Fall besser als die PWM-Ausgänge bezeichnet, dazu verwendet, die Spannung an den Motoren möglichst verlustfrei einzustellen.

Aktivantenne

Der Anschluß einer DCF77-Aktivantenne an den C-Control/BASIC Steuercomputer kann über einen Spezialport an der Buchsenleiste 2 erfolgen (DCF/Freq1) Die Antenne muß dazu über einen Open-Collector-Ausgang nach Masse verfügen, der durch das empfangene Signal geschaltet wird (low-getastet). Zum Anschluß der Aktivantenne ist unbedingt abgeschirmtes Kabel zu verwenden, da sonst besonders bei größeren Kabellängen Störimpulse eingestrahlt werden können. Bei laufenden Motoren ist ein Empfang leider nicht möglich

Frequenz Ein/Ausgänge

Frequenzmessung an Pin FREQ2

Über den Pin FREQ2 können Frequenzen von CMOS/TTL-kompatiblen Rechtecksignalen bis ca. 32000 Hz gemessen werden.

Tonausgabe am BEEP-Pin

Am BEEP-Pin können per BEEP-Befehl (siehe unten) Rechtecksignale (0/5V) ausgegeben werden, die wie bei Ihrem Roboter durch Anschluß eines piezoelektrischen Schallwandlers (Schallwandler ohne interne Elektronik zwischen BEEP und GND) als Töne hörbar gemacht werden können.

Interrupteingang

Der Interrupt-Eingang IRQ

Der IRQ-Pin ist auf der Platine des Roboters mit einem 10k-Pullup-widerstand High-gezogen und mit dem Subsystem IRCOMM/NAV verbunden. Wird beim Ausführen eines BASIC-Programms eine Low-Flanke am IRQ-Pin erkannt, so verzweigt die Abarbeitung des Programms vor dem nächsten BASIC-Befehl zu einer vom Anwender definierten Stelle (siehe INTERRUPT-Befehl).

Expansion-Connector

An den zwei zwanzigpoligen Buchsenleisten sind alle verwendbaren Ports sowie einige Systemsignale des Steuercomputers herausgeführt. Im Anhang finden Sie eine Aufstellung über die Belegung und die interne Verwendung.

Programmierung des C-Control Computers

CCBASIC ist der BASIC-Dialekt, der zur Programmierung des C-Control BASIC Steuercomputers verwendet wird. Die Syntax entspricht in etwa der des Standard-BASIC. Bei einigen Befehlen gibt es Abweichungen oder Erweiterungen, die speziell auf die Hardware des Steuercomputers zugeschnitten sind.

Die Beispiele zum Erlernen von CC-BASIC

Auf der CD finden sie kleine Beispiele, die Ihnen schrittweise den Gebrauch der BASIC-Befehle erläutern. Diese Beispiele sind erfordern zum grössten Teil eine Verbindung zum PC, da sie das „Hyperterminal“ - Sie finden es im Windows unter ->Programme ->Zubehör ->Kommunikation - als Ausgabemittel benutzen.

Starten Sie die C-CONTROL IDE , laden Sie ein BASIC-Programm aus dem Ordner „EINFÜHRUNG_CCBASIC. Wenn Sie das Programm in den Roboter laden wollen müssen Sie die jeweils Das Hyperterminal schliessen, da die IDE sonst eine belegte Schnittstelle vorfindet . Starten Sie das Hyperterminal in der Configuration 9600 Baud, 8n1. Es ist jetzt bereit für die Ausgaben, die C-Control erzeugt.

Was ist ein Programm?

Ein Programm ist die Beschreibung eines Informationsverarbeitungsprozesses. Im Laufe eines solchen Prozesses wird aus einer Menge von variablen oder konstanten Eingangswerten eine Menge von Ausgangswerten berechnet. Die Ausgangswerte sind entweder selbst Ziel der Informationsgewinnung oder dienen mittelbar zur Reaktion auf die Eingangswerte. Neben den eigentlichen Berechnungen kann ein Programm Anweisungen zum Zugriff auf die Hardware des Computers oder zur Steuerung des Programmflusses enthalten.

Ein BASIC-Programm besteht aus mehreren Zeilen sogenannten Quelltextes. Dabei enthält jede Zeile eine oder mehrere Rechen- oder Steueranweisung. Außer diesen Anweisungen selbst bestimmt ihre Reihenfolge ganz wesentlich die eingangs beschriebene Informationsverarbeitung. Die Ausführung der den Anweisungen entsprechenden Operationendurch den Steuercomputer erfolgt sequentiell, also nacheinander. Eine Folge von Programmanweisungen mit einem bestimmten Ziel nennt man auch Algorithmus. Daten sind die Objekte des Informationsverarbeitungsprozesses, sie repräsentieren die gespeicherten Informationen. Der C-Control BASIC Steuercomputer verarbeitet und speichert ausschließlich ganzzahlige numerische Daten - sogenannte „Integerzahlen“ von 1, 8 oder 16 Bit. Eine Variable von 8 Bit (Byte) kann nur nichtnegative Werte von 0 bis 255 aufnehmen. Der Wertebereich einer Integervariable von 16 Bit (Word) reicht von -32768 bis +32767. Achten Sie bei allen Berechnungen darauf, daß die Ergebnisse diese Grenzwerte nicht über- oder unterschreiten, da es sonst zu sogenannten „Überläufen“ kommt.

$a = 255 + 1$

ergibt beispielsweise für a den Wert 0 und nicht 256, wenn a nur ein Byte repräsentiert!

$a = -32768 - 1$

ergibt 32767 und nicht -32769, wenn a ein Word repräsentiert!

Grundlegende Elemente von CCBASIC

Allgemeines

Jede Programmzeile enthält eine oder mehrere Anweisung, die durch Doppelpunkte : getrennt sind. Zeilennummern, wie in älteren BASIC-Dialekten üblich, sind nicht notwendig. Werden dennoch Zeilennummern angegeben, so können diese als Sprungziel verwendet werden.

10 . . .

GOTO 10

Einen Einfluß auf die Reihenfolge der Programmoperationen haben die Nummern darüber hinaus nicht. Wenn beispielsweise im Quelltext auf eine mit 200 nummerierte Zeile eine Zeile 100 folgt, wird trotzdem die Zeile 200 vor der 100 abgearbeitet.

Kommentare können zur Erläuterung des geschriebenen Programms mit in den Quelltext aufgenommen werden und steigern dessen Lesbarkeit und Wartungsfreundlichkeit. Ein Kommentar in CCBASIC beginnt stets mit einem Hochkomma ' und erklärt den Rest der Zeile zum nicht zum Programm gehörigen Text.

a = b + c ' . . . Kommentar . . .

Bezeichner

Bezeichner sind Programmelemente aus alphanumerischen Zeichen (A bis Z, 0 bis 9) die in vom Programmierer festgelegter Weise Objekte, wie Variablen und Konstanten, bezeichnen. Label-Namen und die sogenannten „reservierten Worte“ sind ebenfalls Bezeichner. Es erfolgt keine Unterscheidung von Groß- und Kleinbuchstaben. Ein Bezeichner beginnt stets mit einem Buchstaben oder mit einem Unterstrich. Leerzeichen innerhalb eines Bezeichners sind nicht erlaubt.

Variablen und Konstanten

Variablen und Konstanten sind Objekte des Informationsverarbeitungsprozesses.

In CCBASIC speichern beide einen numerischen Wert. Während der Wert einer Konstante einmal angegeben wird und dann unverändert bleibt, kann sich der Wert einer Variablen im Lauf des Programms beliebig oft ändern. Konstanten können in CCBASIC in dezimaler, hexadezimaler und binärer Form angegeben werden. Die Syntax für Hexadezimal- und Binärzahlen sei hier am Beispiel der Zahl 46 (dezimal) gezeigt:

```
&H2E  
&B101110
```

Außerdem können per DEFINE-Zeilen (siehe unten) symbolische Konstanten vereinbart werden. Auf Variablen wird stets über ihren Bezeichner zugegriffen. Dieser Bezeichner muß vor der ersten Verwendung der Variable im Programm in einer DEFINE-Zeile definiert werden.

Label

Label markieren bestimmte Punkte in der Folge der Programmoperationen. Label sind Ziele von Sprungoperation innerhalb eines Algorithmus. In CCBASIC stehen Label am Anfang einer Zeile und beginnen stets mit einem Doppelkreuz, dann folgt - ohne Leerzeichen - der Bezeichner des Labels.

Das Beispiel zeigt die Definition des Labels „Label1“ und die Verwendung in einem Sprungbefehl:

```
#label1 . . .  
GOTO label1
```

Terme

Ein Term ergibt sofort (als Variable oder Konstante) oder durch Berechnung einen bestimmten Wert. Terme sind Teile von Anweisungen und stehen beispielsweise bei der Zuweisung eines Wertes an eine Variable rechts des Zuweisungszeichens „=“. Terme werden durch Kombinationen von Operanden und Operatoren gebildet.

```
a +b  
(ABS(x) - 13) * 10
```

Operanden und Operatoren

Ein Operand ist in der Grundform entweder eine Konstante, eine Variable oder ein Funktionsaufruf, kann aber auch selbst wieder ein aus Operanden und Operatoren zusammengesetzter Term sein. Operatoren bezeichnen Rechenoperationen, die mit den umstehenden Operanden auszuführen sind. Dabei gibt es eine definierte Rangfolge der Operatoren (siehe Befehlsbeschreibung), die die Reihenfolge der Berechnungen bestimmt.

Funktionen

Eine Funktion führt eine definierte Operation - zum Beispiel eine Berechnung - durch und liefert durch ihren Aufruf einen Ergebniswert. Die meisten Funktionen erwarten ein oder mehrere Argumente, die in runden Klammern „()“ nach dem Funktionsbezeichner übergeben werden und durch Kommas getrennt sind. Einige Funktionen werden ohne Argument aufgerufen. In diesem Fall werden keine runden Klammern geschrieben.

```
ABS (x)  
MAX(a,b)  
RAND  
EOF
```

In CCBASIC sind alle unterstützten Funktionen vordefiniert. Deren Bezeichner gehören zu den reservierten Worten. Die Formulierung anwenderdefinierter Funktionen ist in CCBASIC nicht vorgesehen.

Zuweisungen

Die Zuweisung ist die einfachste Form einer Programmanweisung. Nach dem Bezeichner einer Variablen, der ein Wert zugewiesen werden soll, folgt das Zuweisungszeichen „=“ und dann ein Term, der den zuweisenden Wert bestimmt. Eine Zuweisung entspricht damit einer einfachen mathematischen Formel.

a = 10
b = x - y
c = SQR(a*a + b*b)

Befehle

Neben den einfachen Zuweisungen sind Befehle Anweisungen zur Ausführung von Programmoperationen durch den C-Control/BASIC Steuercomputer. Befehle beginnen stets mit einem reservierten Wort. Einige Befehle erwarten einen oder mehrere Parameter zur genauen Spezifikation der auszuführenden Programmoperation. Diese Parameter werden nach dem Befehlsbezeichner und einem Leerzeichen aufgeführt und dabei durch Kommas getrennt (Ausnahme PRINT, siehe Befehlsübersicht). Im Gegensatz zu den Argumenten beim Aufruf einer Funktion stehen die Befehlsparameter nicht innerhalb runder Klammern!

RANDOMIZE
PAUSE 100
BEEP 440,50,50

Anweisungen zur Steuerung des Programmflusses

Diese Anweisungen erlauben, die Reihenfolge der an sich streng sequentiell abgearbeiteten Programmoperationen zu steuern und an Eingangswerte des Informationsverarbeitungsprozesses anzupassen. Sie bieten eine hohe Flexibilität bei der Algorithmenformulierung und sind für die Lösung mancher anwendungstechnischer Probleme sogar Grundvoraussetzung. Anweisungen zur Steuerung des Programmflusses bestehen aus einem oder mehreren reservierten Worten und erfordern in jeweils spezieller Weise eventuell weitere Angaben.

GOTO label1
IF a > b THEN GOSUB label2
FOR i = 0 TO 10 STEP 2
.._
NEXT

Compileranweisungen

Zusätzlich zu den Programmanweisungen enthält ein CCBASIC-Quelltext Compileranweisungen, die zum Beispiel zum Anlegen von Datenblocks (Tabellen) oder zur Definition von Variablen- und Konstanten dienen. Für Compileranweisungen gilt die Doppelpunktregel zum Trennen mehrerer Anweisungen in einer Zeile nicht. Es darf jeweils nur eine Compileranweisung in einer Zeile stehen. Die DEFINE-Anweisung ist eine Compileranweisung.

Definition symbolischer Konstanten

Es ist guter Programmierstil, statt „magischer“ Zahlen im Programm

IF x > 1234 THEN GOTO alarm

besser symbolische Konstanten zu verwenden. Durch Vergabe signifikanter Bezeichner für Konstanten erhöht sich die Lesbarkeit des Quelltextes. Wenn alle Konstanten global definiert werden, ist ein Programm auch leichter zu warten. Das gilt besonders, wenn ein und dieselbe Konstante mehrmals im Programm benötigt wird. Die Definition einer symbolischen Konstanten erfolgt wie folgt:

DEFINE bezeichner wert

Dabei ist wert entweder eine dezimale, hexadezimale oder binäre Zahl. So sollte das Beispiel zuvor besser

DEFINE limit 1234
...
IF x > limit THEN GOTO alarm

lauten.

Definition von Variablen

Der C-Control/BASIC Steuercomputer stellt 24 Byte-Speicherzellen seines internen Speichers (RAM) dem Anwender zur Verwendung in seinen Programmen zur Verfügung. In diesem Speicherbereich werden alle

Variablen eines BASIC-Programms gespeichert. Die 24 Bytes können je nach Bedarf auch bitweise oder als 16bit Integer (Word) verwendet werden. Im Gegensatz zum Standard-BASIC müssen in CCBASIC alle vom Programm benutzten Variablen vor ihrer ersten Verwendung definiert werden. Dabei ist der Datentyp zu spezifizieren (Bit, Byte oder Word) und kann (für Bits muß!) eine Speicherzellennummer angegeben werden. Der Anwender muß selbst darauf achten, daß keine unerwünschten Überlappungen bei der Vergabe der Speicherplätze entstehen, da es sonst zum gegenseitigen überschreiben der Variablen kommen kann. Beispielsweise belegen bit[18], byte[2] und word[1] jeweils einen Teil der Zelle 2 des Speicherbereiches.

- **Definition einer Bitvariablen:**

DEFINE bezeichner BIT [nr]

Dabei sind für nr Werte von 1 bis 192 (24 Bytes mit je 8 Bit) zulässig.

- **Definition einer Bytevariablen mit Zellennummer:**

DEFINE bezeichner BYTE[nr]

Dabei sind für nr Werte von 1 bis 24 (24 Bytes) zulässig.

- **Definition einer Integervariablen mit Zellennummer:**

DEFINE bezeichner WORD[nr]

Dabei sind für nr Werte von 1 bis 12 (ein Word belegt 2 Bytes) zulässig.

Wenn bei Byte- und Worddefinitionen die Zellenangabe [nr] weggelassen wird, übernimmt der Compiler die Aufteilung auf den Speicherbereich. Achten Sie dann darauf, daß nicht abwechselnd Bytes und Words definiert werden. Die folgenden Anweisungen

```
DEFINEa BYTE  
DEFINE b WORD  
DEFINE c BYTE  
DEFINE d WORD
```

führen zu zwei ungenutzten (verschenkten kostbaren!) Bytes, zwischen a und b sowie zwischen c und d, da Words prinzipiell an den Bytes 1,3,5,7,... usw. der 24 Bytes ausgerichtet werden. Besser wäre,

```
DEFINE b WORD  
DEFINE d WORD  
DEFINE a BYTE  
DEFINE c BYTE
```

zu schreiben. Die automatische Aufteilung der Variablen auf den Speicher durch den Compiler beginnt bei Zellennummer 1. Das obige (bessere) Beispiel belegt 6 Bytes. Bei Definition weiterer Bits, Bytes und Words mit Angabe der Zellennummer ist wieder auf unerwünschte Überlappung zu achten. Ein bereits definierter Variablenbezeichner darf nicht ein zweites Mal definiert werden.

Definition von Digitalports

In CCBASIC wird auf Ports wie auf Variablen zugegriffen. Auch hier muß jeder verwendete Port zuvor definiert sein.

- **Definition eines der 16 Digitalports:**

DEFINE bezeichner PORT[nr]

- **Definition eines 8 Bit breiten Potts:**

DEFINE bezeichner BYTEPORT[nr]

Dabei sind für nr nur die Werte 1 (Ports 1 bis 8 als Byteport) und 2 (Ports 9 bis 16) zulässig.

- **Die Wurzelfunktion SQR(x)**

liefert eine Näherung für die Quadratwurzel aus dem Argument x. Dabei werden die Nachkommastellen abgeschnitten.

- **Die Signumfunktion SGN(x)**

ergibt 1, wenn der Wert des Argumentes x größer 0, und ergibt -1, wenn der Wert kleiner 0 ist. Für x = 0 ist auch das Ergebnis der SGN-Funktion gleich 0.

- **Die Maximumfunktion MAX(x,y)**

ergibt x, wenn x > y ist, sonst y.

- **Die Minimumfunktion MIN(x,y)**

ergibt x, wenn x < y ist, sonst y.

- **Der Befehl RANDOMIZE x**

initialisiert den internen Pseudo-Zufallsgenerator des Steuercomputers mit dem Wert von x. Ein und derselbe Initialisierungswert führt stets zu einer identischen Folge von Zahlen. Die Spezialform RANDOMIZE TIMER lädt den Wert des freilaufenden Timers in den Generator.

- **Die Zufallsfunktion RAND**

liefert den nächsten Integer-Zufallswert des Pseudo-Zufallsgenerators. Die Zufallszahlen werden nach dem multiplikativen Verfahren mit anschließender Modulodivision (siehe ein gutes Mathematikbuch) aus dem jeweils vorangehenden Wert erzeugt.

Rangfolge von Operatoren und Funktionsaufrufen

Bei der Berechnung von Termen mit Operatoren und Funktionen ist deren Rangfolge von entscheidender Bedeutung. Teilausdrücke mit Operatoren von hohem Rang werden vor denen mit einem niedrigerem Rang berechnet (vergleiche Rechenregel: „Punktrechnung vor Strichrechnung“). Bei gleichrangigen Operatoren erfolgt die Berechnung von links nach rechts. Wie in der Mathematik kann jedoch durch Klammersetzung zusätzlich Einfluß auf die Berechnungsreihenfolge genommen werden. CCBASIC unterstützt maximal 3 Klammerebenen. Im Sinne der Übersichtlichkeit eines Programmes sollten jedoch „wilde“ Klammersetzungen vermieden und komplexe Berechnungen auf mehrere BASIC-Zeilen aufgeteilt werden.

Die folgende Liste zeigt die CCBASIC Operatorenrangfolge :

RANG OPERATOREN

9	()
8	Funktionsaufrufe
7	Negatives Vorzeichen
6	* / MOD SHL SHR
5	+ -
4	> >= < <= = <>
3	NOT
2	AND NAND
1	OR NOR XOR

Anweisungen zur Steuerung des Programmflusses

- **Schleife**

FOR variable = anfang TO ende STEP schrittweite

...

NEXT

Die FOR-Schleife führt die Anweisungen bis zum NEXT solange aus, bis der Wert der variable gleich dem Wert des Terms ende ist. Vor dem ersten Durchlauf wird der Wert des Terms anfang berechnet und der Schleifenvariablen zugewiesen. In jedem Durchgang wird der Wert des schrittweite- Terms zur Schleifenvariablen addiert. In der Form

FOR variable = anfang TO ende STEP schrittweite

```
...  
NEXT
```

beträgt die Schrittweite konstant 1. Die Werte des ende-Terms und des schrittweite-Terms werden mit jedem Schleifendurchlauf neu berechnet. Das gestattet eine erweiterte Kontrolle des Programmverlaufes.

FOR-Schleifen können ineinander verschachtelt werden. Die Verschachtelungstiefe ist nur durch den für die Schleifenvariablen erforderlichen Speicherplatz beschränkt.

```
FOR variable = anfang TO ende1  
  FOR variable = anfang TO ende2  
    FOR variable = anfang TO ende3  
      .....  
    NEXT  
  NEXT  
NEXT
```

Jede FOR-Schleife darf im Verlauf des Programms nur über ihre eigene NEXT-Anweisung laufen. Folgender Quelltext kann zwar kompiliert und in den Steuercomputer geladen werden, wird jedoch nicht wie vielleicht erwartet funktionieren:

```
FOR v1 = anfang1 TO ende1  
  ...  
  GOTO anothernext  
...  
NEXT  
FOR v2 = anfang 2 TO ende2  
  ...  
  #anothernext  
NEXT
```

Achten Sie außerdem auf den Wertebereich von Schleifenvariable und ende-Term!

```
DEFINE v BYTE  
FOR v = 1 TO 1000  
  ...  
NEXT
```

wird zu einer Endlosschleife, da v als Bytevariable nie den Wert 1000 erreichen kann, sondern bereits nach 255 wieder auf 0 überrollt.

• Bedingte Ausführung

```
IF bedingung THEN anweisung1  
oder  
IF bedingung THEN anweisung1 ELSE anweisung
```

Die IF...THEN...ELSE-Konstruktion ermöglicht die Anpassung des Programmflusses an Bedingungen zur Laufzeit des Programms. Als bedingung ist ein beliebiger Term einzusetzen. Ergibt dessen Berechnung einen Wert ungleich 0, dann gilt die Bedingung als erfüllt, und die anweisung1 wird ausgeführt. Werden zusätzlich ein ELSE und eine zweite Anweisung angegeben, so wird diese Anweisung alternativ ausgeführt, wenn der berechnete Term einen Wert gleich 0 ergibt. Die gesamte IF...THEN...ELSE-Konstruktion muß in einer Quelltextzeile stehen. Anweisungsblöcke (mehrere Anweisungen) nach THEN und ELSE sind nicht zulässig.

• Sprunganweisung

GOTO label

Mit der GOTO-Anweisung kann der Steuercomputer veranlaßt werden, die Programmabarbeitung an einer bestimmten Stelle fortzusetzen. Als Ziel des Sprungs wird ein Label-Bezeichner angegeben. Das Sprungziel kann sich vor oder nach der GOTO-Anweisung im Quelltext befinden.

- **Aufruf und Rückkehr aus einer Unterroutine**

Der Aufruf einer Unterroutine erfolgt mit der Anweisung

GOSUB label

Dabei ist label der Anfangspunkt der Unterroutine. In den sogenannten Unterroutinen sind Programmabschnitte zusammengefaßt, die mehrfach im Verlauf der Programmabarbeitung benötigt werden. Eine Unterroutine beginnt stets mit einem Label, enthält dann eine oder mehrere Anweisungen und abschließend ein RETURN. Nach dem RETURN wird die Programmabarbeitung mit der Anweisung nach dem GOSUB fortgesetzt. Die Programmabarbeitung darf ohne ein vorheriges GOSUB niemals an eine RETURN-Anweisung gelangen. Die maximal zulässige Verschachtelungstiefe bei Aufrufen von Unterroutinen aus Unterroutinen ist vier.

#hauptprogramm

GOSUB subl

...

#sub1

GOSUB sub2

...

RETURN

#sub2

GOSUB sub3

...

RETURN

#sub3

GOSUB sub4

...

RETURN

#sub4

...

RETURN

- **Wertgesteuerte Programmverzweigung**

ON variable GOTO label0,label1, . ..labeln

oder

Oder ON variable GOSUB label0,label1, . ..labeln

In Abhängigkeit des Wertes des Selektors variable erfolgt eine Programmverzweigung oder ein Unterroutinenaufruf zu den aufgelisteten Einsprungpunkten. Ist der Wert 0, dann wird zu label0 verzweigt, bei Wert gleich 1 zu label1 usw. Ist der Variablenwert negativ oder größer als die Anzahl der aufgeführten Sprungziele, dann wird die Programmabarbeitung ohne Verzweigung fortgesetzt.

- **Programmverzweigung nach einem Interruptsignal am Pin IRQ**

Wird während der Abarbeitung eines Programmes ein Interruptsignal (Low-Flanke) am IRQ-Pin detektiert, wird die momentan bearbeitete BASIC-Anweisung zum nächstmöglichen Zeitpunkt unterbrochen und die Programmabarbeitung an der zuvor mit dem Befehl

INTERRUPT label1

festgelegten Stelle fortgesetzt. Der Rücksprung zur Ausgangsposition erfolgt durch den Befehl

RETURN INTERRUPT

Im folgenden Beispiel wird durch jeden Interrupt eine Leuchtdiode ein/ausgeschaltet:

```

DEFINE led PORT[8]
led = OFF
‘ Festlegen der Interruptroutine
INTERRUPT switch_it
‘ Endlosschleife
#loop
GOTO loop
‘ Interruptroutine
#switch_it
tog led
RETURN INTERRUPT

```

Achtung:

Durch einen Fehler im Betriebssystem kommt es im Interruptbetrieb zu Problemen, wenn der Befehl "ON (ausdruck) GOSUB (label)" verwendet wird. Verwenden Sie "ON (ausdruck) GOTO (label)" !!

- **Programmende**

END

Gelangt der Steuercomputer im Verlauf der Programmabarbeitung zur END-Anweisung, wird die Programmabarbeitung beendet. Das System verharrt dann in einem inaktiven Zustand. Jetzt kann ein neues Anwenderprogramm übertragen oder die Ausführung per Start-Taster wieder gestartet werden.

- **Verzögerung des Programmflusses**

Die Anweisung

WAIT conditionterm unterbricht die Programmausführung solange, bis die Berechnung des conditionterm einen Wert ungleich 0 ergibt.

```

define key port[9]

```

```

...

```

```

WAIT key

```

In diesem Beispiel wird solange gewartet, bis vom Digitalport 9 ein HIGH-Pegel (= logisch 1) gelesen wird. Der PAUSE Befehl unterbricht die Programmausführung für eine gewisse Zeit. Der berechnete Wert des Parameterterms geht als Multiplikationsfaktor mit der Grundeinheit von 20 Millisekunden in die Festlegung der Pausenzeit ein.

PAUSE term

Beispielsweise wird durch den Befehl

PAUSE 50

die Programmausführung für ca. 50*20 Millisekunden = 1 Sekunde unterbrochen. Die maximale Zeitabweichung der tatsächlichen Pause vom angegebenen Wert beträgt dabei prinzipbedingt + 20 Millisekunden.

Kommunikation über die serielle Schnittstelle

- **Datenausgabe**

Die Datenausgabe erfolgt als Text über die serielle Schnittstelle des C-ControlBASIC Steuercomputers. Ist über ein Schnittstellenkabel zum Beispiel ein PC mit einem Terminalprogramm angeschlossen, können die ausgegebenen Daten dort angezeigt werden.

PRINT term

gibt das Ergebnis der Berechnung von term aus.

PRINT "text"

überträgt den in Anführungszeichen stehenden Text. In beiden Fällen wird an die Übertragung ein Zeilenvorschubzeichen angehängt, welches das Terminalprogramm veranlaßt, die nächste Ausgabe in der nächsten Bildschirmzeile vorzunehmen. Der Zeilenvorschub kann unterdrückt werden, wenn dem PRINT-Befehl nach dem Parameter (term oder „text“) ein Semikolon hinzugefügt wird.

PRINT term;

Oder :

PRINT "text";

CCBASIC unterstützt außerdem mehrere Ausgaben mit einem PRINT-Befehl, wobei die einzelnen Parameter durch Komma oder Semikolon getrennt werden. Ein Komma fügt in die Ausgabe ein Tabulatorzeichen ein, das entsprechend den Einstellungen im Terminalprogramm als eine Anzahl von Leerzeichen am Bildschirm erscheint. Sollen zwei Ausgaben ohne Zwischenraum aufeinander folgen, so sind diese im PRINT-Befehl durch ein Semikolon zu trennen.

PRINT "a= ", a

PRINT "a= "; a

Ein einzelner PRINT-Befehl ohne Parameter gibt nur einen Zeilenvorschub aus.

PRINT

• Dateneingabe

Mit dem Befehl

INPUT variable

kann ein Integerwert von der seriellen Schnittstelle gelesen und für die anschließende Weiterbearbeitung in einer Variablen gespeichert werden. Der Wert wird in einem Terminalprogramm an einem PC eingegeben und nach dem Drücken der ENTER-Taste per Schnittstellenkabel an den C-Control/BASIC Steuercomputer übertragen. Der INPUT-Befehl wartet solange, bis eine komplette Datenübertragung vom Terminal empfangen wurde. Wird der INPUT-Befehl aufgerufen, ohne daß eine Datenübertragung vom Terminal erfolgt, wird das Programm endlos an dieser Stelle stehen bleiben! Hier hilft dann nur noch der Reset-Taster und der anschließende Neustart des C-Control/BASIC Gerätes.

• Byteweise Kommunikation über die serielle Schnittstelle

Während PRINT und INPUT kurze Zeichenketten zur Darstellung eines numerischen Wertes senden beziehungsweise erwarten, kann es wünschenswert sein, einzelne Bytes seriell zu übertragen. Dafür bietet CCBA-SIC die Befehle PUT und GET.

PUT term

sendet den berechneten Wert eines Terms. Falls erforderlich, wird das Ergebnis zuvor auf den Byte Wertebereich (0...255) reduziert.

GET variable

wartet auf ein seriell empfangenes Byte und speichert den Wert dann in der angegebenen Variablen.

• Weitere Schnittstellenbefehle und -funktionen

Wie beschrieben warten INPUT und GET unter Umständen endlos auf den Empfang serieller Daten. Soll ein „Aufhängen“ des Programms in dieser Art verhindert werden, kann vor jedem von INPUT oder GET durch Aufruf der Statusfunktion RXD ermittelt werden, ob empfangene Daten zur Verfügung stehen. Die Funktion liefert in diesem Fall den Wert -1. Ist der Schnittstellenpuffer leer, so ist das Funktionsergebnis gleich 0.

if RXD then GET thebyte

Die voreingestellte Übertragungsrates der seriellen Schnittstelle beträgt für Sender und Empfänger 9600 Bit pro Sekunde (baud). Mit dem BAUD Befehl können jedoch auch andere Raten eingestellt werden. CCBASIC enthält

dafür einige vordefinierte Konstanten: R1200, R2400, R4800, R9600 für die Raten 1200 bis 9600 Bit pro Sekunde.

BAUD R2400

schaltet beispielsweise Sender und Empfänger auf die Rate von 2400 Bit pro Sekunde um. Prinzipiell sind auch andere als die vordefinierten Raten, auch für Sender und Empfänger unterschiedliche, möglich. Die Übertragungsraten der seriellen Schnittstelle werden durch Teilung aus einem internen Takt des Mikroprozessors des C-Control/BASIC Steuercomputers abgeleitet. Der dem BAUD Befehl zu übergebende Bytewert enthält die erforderlichen Teilerwerte Nxx.

b7	b6	b5	b4	b3	b2	b1	b0
NP1	NP0	NT2	NT1	NT0	NR2	NR1	NR0

Bit 7 und 6 enthalten einen für Sender und Empfänger gemeinsamen Vorteiler NP. NP kann die Werte 1, 3, 4 und 13 annehmen.

VORTEILER	NP1	NP0
1	0	0
3	0	1
4	1	0
13	1	1

NT (Bit 3 bis 5) und NR (Bit 0 bis 2) bestimmen weitere Teilerwerte, getrennt für Sender (NT) und Empfänger (NR), entsprechend folgender Codierung:

TEILER	NT2 / NR2	NT1/ NR1	NT0 / NR0
1	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

Die Übertragungsrate des Senders berechnet sich nach folgender Formel : $\text{Senderrate} = 125000 / (\text{NP} * \text{NT})$,
die des Empfängers entsprechend: $\text{Empfängerrate} = 125000 / (\text{NP} * \text{NR})$.

Die weiteren Schnittstellenparameter - 8 Datenbits, kein Paritätsbit, 1 Stopbit –sind fest und können nicht geändert werden. Die auf dem C-ControlBASIC Steuercomputer vorbereiteten Kanäle für die Handshake-Signale RTS und CTS werden in der aktuellen Version nicht verwendet und sind in CCBASIC-Programmen nicht ansprechbar.

Dateifunktionen

Die Dateifunktionen erlauben das Aufzeichnen von Messwerten oder anderen Daten oder können zum Abspeichern von Information benutzt werden, die nach Ausfall der Betriebsspannung wieder in die Programmvariablen geladen werden sollen. Der Speicherbereich im EEPROM-Chip nach dem Anwenderprogramm -meist der größte Teil - steht für diesen Zweck zur Verfügung. Der Speicherbereich wird als eine Datei verwaltet, auf die lesend oder schreibend zugegriffen werden kann, nachdem sie mit dem entsprechenden Attribut geöffnet wurde. Der Befehl zum Öffnen der Datei lautet wie folgt:

OPEN# FOR WRITE

OPEN# F OR APPEND

OPEN# FOR READ

Dabei bedeutet WRITE das Öffnen zum Schreiben mit Überschreiben eventueller alter Aufzeichnungen, APPEND das Öffnen zum Schreiben mit Anhängen der neuen an die alten Aufzeichnungen und

READ das Öffnen zum Auslesen der Aufzeichnungen. Es können nur Integerwerte gespeichert und gelesen werden. Jeder Wert belegt also 2 Bytes im EEPROM. Das Schreiben und Lesen erfolgt mit den Befehlen

PRINT# term

wobei das berechnete Ergebnis des Terms gespeichert wird, beziehungsweise

INPUT# variable

wobei variable eine definierte Integervariable des Programms bezeichnet. Schreiben in und Lesen aus der Datei erfolgt streng sequentiell. Dafür wird intern ein Dateizeiger verwaltet, der nach jedem Zugriff um 1 erhöht wird. Vor jedem Schreiben sollte geprüft werden, ob noch genügend Platz im EEPROM zur Aufnahme der Daten vorhanden ist. Dafür kann die Funktion FILEFREE abgefragt werden, die als Ergebnis die Größe des noch freien Speichers liefert (in Words). Folgendes Beispiel zeigt die Anwendung der Funktion

```
DEFINE a WORD  
DEFINE b WORD  
DEFINE c WORD  
DEFINE blocksize 3  
...  
IF FILEFREE >= blocksize THEN GOSUB writeblock  
...  
#writeblock  
PRINT# a  
PRINT# b  
PRINT# c  
RETURN
```

Vor jedem Lesen sollte geprüft werden, ob noch weitere Daten aufgezeichnet sind. Die Funktion dafür lautet EOF („end of file“). Ihr Ergebnis ist -1, wenn in der Datei keine weiteren Daten verfügbar sind, sonst 0. Die Abfrage der EOF Funktion sollte das Auslesen von Datenblocks in gleicher Weise rahmen, wie beim Schreiben der Daten. Zum obigen Beispiel würde also

```
IF NOT EOF THEN GOSUB readblock  
...  
#readblock  
INPUT# a  
INPUT# b  
INPUT# c  
RETURN
```

gehören.

Nach Beenden eines Dateizugriffs sollte die Datei sofort wieder geschlossen werden. Erst dann sind die Daten vor einem Spannungsausfall oder Reset des Systems sicher. Der Befehl dafür lautet

CLOSE#

und hat keinen Parameter.

Portbefehle

• **der Umschaltbefehl TOG**

Prinzipiell erfolgt der Zugriff auf die Ports des Steuercomputers wie auf Variablen. Um einen Digitalport einzuschalten, schreibt man

P=I und **P=O** , um ihn auszuschalten.

Um den Port umzuschalten (EIN nach AUS; AUS nach EIN), kann man schreiben

P = NOT P

oder den Befehl

TOG P

benutzen. TOG steht für englisch „toggle“. Der TOG Befehl benötigt weniger Platz im EEPROM und wird schneller als die klassische NOT-P-Konstruktion ausgeführt. Die Portvariable P darf beim TOG Befehl nur für einen einzelnen Digitalport stehen, nicht für einen Byte- oder Wordport.

• **Deaktivieren eines Ports mit DEACT**

Sobald einer Portvariablen erstmalig ein Wert zugewiesen wird, schaltet der Steuercomputer die zugehörigen Hardwarestrukturen im Prozessorchip (Transistoren) auf Ausgangsbetrieb. Es fließt also entsprechend der angeschlossenen Schaltung Strom aus bzw. in den Prozessor (max. 10 mA zulässig!). Der Befehl

DEACT portvar

deaktiviert den angegebenen Port. Das heißt, der Port wird in einen hochohmigen Zustand geschaltet und arbeitet im Eingangsbetrieb. Der DEACT Befehl darf auf einzelne Digitalports oder Byteports angewendet werden.

• **der PULSE Befehl**

Mit dem Befehl

PULSE portvar

wird ein Puls von einigen Millisekunden Breite am mit portvar bezeichneten Port ausgegeben. Das ist beispielsweise nützlich zum Schalten extern angeschlossener flankengetriggelter Logikschaltkreise. Steht der Port vor Ausführung des PULSE Befehls auf low (=0), wird ein High-Puls (0-1-0), ansonsten ein Low-Puls (1-0-1) ausgegeben. Die Portvariable darf beim PULSE Befehl nur für einen einzelnen Digitalport stehen, nicht für einen Byte- oder Wordport.

Definition und Anwendung von Datentabellen

Im Standard-BASIC dienen DATA-Zeilen zum Ablegen von konstanten Datenblöcken, auf die dann sequentiell zugegriffen werden kann. CCBASIC unterstützt keine DATA-Zeilen, bietet jedoch ein weitaus flexibleres Werkzeug zur Definition und zum Zugriff auf Datenblöcke. Konstante Daten können in Form von Tabellen abgelegt werden. Jede Tabelle bekommt einen Bezeichner (tablename) zugewiesen und kann beliebig viele Einträge enthalten, soweit der Programmspeicher Platz bietet. Jeder Dateneintrag (Cx) wird als Integerwert abgelegt und belegt somit zwei Bytes. Dabei können die Daten direkt im Quelltext aufgeführt werden.

```
TABLE tablename CO C1 C2 C3 ..  
c4 c5 . . .  
.. Cn  
TABEND
```

oder vom CCBASIC-Compiler aus einer externen Textdatei importiert werden

```
TABLE tablename "tabfilename"
```

Die Tabellendefinitionen müssen stets am Ende eines Programms, hinter dem END Befehl stehen, da die Daten nahtlos hinter den vorangehenden Codebytes im EEPROM-Speicherchip abgelegt werden. Die Programmabarbeitung darf nie über Tabellendaten laufen, da die Daten sonst als BASIC Befehle interpretiert werden würden, was sicher zum Absturz des Systems führt. Der Zugriff auf die Tabellendaten erfolgt mit dem Befehl

LOOKTAB tablename,index,variable

tablename bezeichnet eine gültige Tabelle, für index kann ein beliebiger Term stehen und die variable bezeichnet die Speicherzelle, in der das Ergebnis abgespeichert werden soll. Der berechnete Wert des index-Terms darf nicht negativ sein und maximal N-1 betragen, wenn die indizierte Tabelle N Einträge hat. Ergibt index den Wert 0, so wird CO in der angegebenen Variablen gespeichert, für index gleich 1 C1 und so weiter. Folgendes Beispiel gibt den Inhalt einer Tabelle seriell aus

```

DEFINE value WORD
DEFINE i BYTE
FOR i = 0 to 3
LOOKTAB mytab,i,value
PRINT "mytab["; i; "]="; value
NEXT
END
TABLE mytab 12 -20 0 1000
TABEND

```

Am Bildschirm des Terminalprogramms sollte erscheinen

```

mytab[0]=12
mytab[1]=-20
mytab[2]=0
mytab[3]=1000

```

Besonders nützlich erweisen sich Tabellen beim Umsetzen von A/D-Werten in echte physikalische Größen. Eine Umsetzungstabelle hat dann in der Regel 256 Einträge. Der gemessene A/D-Wert geht dann als Tabellenindex in die Bestimmung der physikalischen Größe ein.

Die Echtzeituhr

Um den Stand der internen Echtzeituhr auszulesen und zu setzen, sind folgende globale Variablen definiert:

YEAR Jahr (0...99)

MONTH Monat (1...12)

DAY Tag des Monats (1...31)

DOW Wochentag (0=Sonntag...6=Samstag)

HOURL Stunde (0...23)

MINUTE Minute (0...59)

SECOND Sekunde (0...59)

Beachten Sie bitte, daß während des Zugriffs die interne Uhr weiterläuft. Der Sekundenwert sollte daher stets zuerst ausgelesen werden. Steht er auf 59, so muß nach dem Lesen der letzten interessierenden Zeitinforation (z.B YEAR) der Sekundenwert nochmals gelesen und auf =0 getestet werden. In diesem Fall ist das Auslesen der Echtzeituhr zu wiederholen, da eine neue Minute angebrochen ist (Extremfall Silvester mit Weiterschalten aller Stellen in Uhr und Datum). Die Jahreszahl wird im C-Control System nur zweistellig abgespeichert.

Anmerkung:

Durch einen Fehler im Betriebssystem zählt DOW bis 7. Sie können diesen Fehler korrigieren, indem in jedem BASIC-Programm, dass auf den Wochentag angewiesen ist, die IF-Abfrage

```

IF DOW > 6 THEN DOW = 0

```

vor JEDEM Auslesen des DOW-Registers eingefuegt wird.

Interner Timer, Tonerzeugung, Frequenzmessung

- **Timer**

Der interne 20-Millisekunden-Timer kann über den vordefinierten Bezeichner TIMER ausgelesen werden. Der Timer ist freilaufend und kann nicht gestellt oder rückgesetzt werden.

- **Ausgabe von Tönen mit BEEP**

Der C-ControVBASIC Steuercomputer kann an einem seiner Pins (BEEP-Pin, entspricht Prozessorausgang TCMP1) Töne als Rechteckschwingungen ausgeben. Der Befehl dazu lautet

BEEP ton, tTon, tPause

Für die drei Parameter können Konstanten oder Terme eingesetzt werden. Dabei bestimmt ton die Tonhöhe nach der Formel

$$\text{ton} = 250000 / \text{freq} [\text{Hz}]$$

, tTon bestimmt die Dauer des Tons und tPause die Pause nach dem Ton. Die Einheit für die Zeitangaben beträgt 20 Millisekunden. Der Befehl

BEEP 568, 10, 3

gibt also für $10 \cdot 20 = 200$ Millisekunden einen Ton von etwa 440 Hz (Kammerton A) aus und macht danach eine Pause von $3 \cdot 20 = 60$ Millisekunden. Wenn nach einem BEEP kein weiterer BEEP folgt, kann die Pause auch auf 0 gesetzt werden. Ist für die Tonlänge 0 angegeben, wird ein Dauerton erzeugt. Der Tongenerator schaltet den Ton ein und fährt mit der Abarbeitung des BASIC-Programms fort. Mit dem Wert 0 für ton kann der Tongenerator wieder abgeschaltet werden.

• Frequenzmessung mit der Funktion FREQ

Ist am DCF77-Eingang keine Aktivantenne angeschlossen, so kann mit diesem Eingang alternativ eine Frequenzmessung erfolgen, deren Ergebnis mit der Funktion FREQ jederzeit abgefragt werden kann.

x = FREQ

Die Frequenzmessung basiert auf dem Pulszählprinzip bei einer Torzeit von 1 Sekunde. Die Messung erfolgt ständig im Hintergrund, parallel zur Abarbeitung des BASIC-Programms. Der Messbereich reicht bis etwa 5 Kilohertz mit einem Messfehler unter einem Prozent. Danach wird das Ergebnis zunehmend ungenauer.

• Stromsparmodus mit SLOWMODE

Anwendungen, die keine hohe Rechenleistung benötigen, können durch Aufruf des Befehls

SLOWMODE ON

den internen Takt des Mikroprozessors verlangsamen (1/16). In Kombination

Mit dem Abschaltbaren Subsystem läßt sich so der Leistungsbedarf des Steuercomputers nochmals senken. Sollte im Verlauf des Programms wieder eine höhere Geschwindigkeit erforderlich sein, so läßt sich mit

SLOWMODE OFF

wieder der Ausgangszustand herstellen. Programme, die serielle Datenübertragungen verwenden, sollten den SLOWMODE nicht aktivieren, da die eingestellten Übertragungsraten mit dem Prozessortakt herabgesetzt werden.

Einbinden von Maschinenprogrammen

Die folgenden Informationen richten sich an professionelle Anwender des C-Control/BASIC Steuercomputers und sind für die eigentliche BASIC-Programmierung nicht erforderlich zumal CCRP5 die für Maschinenprogramme vorgesehenen Ressourcen vollständig nutzt.

Vorausgesetzt werden die Kenntnis des internen Aufbaus des Mikrocontrollers MC68HC05B6 und Kenntnisse in der Assemblerprogrammierung dieses Controllers. Außerdem wird ein Assembler für 68HC05- benötigt. An dieser Stelle sei das Buch „Motorola 68HC05“ von Zekeriya Zengin aus dem Heise-Verlag (ISBN 3-88229-034-X, Conrad Electronie Best.-Nr.: 91 91 79) empfohlen, das den Mikrocontroller vollständig beschreibt und auf einer beiliegenden Diskette u.a. einen Assembler und zahlreiche Beispiele liefert.

Die meisten anwendungstechnischen Probleme lassen sich sicher ausschließlich durch ein BASIC-Programm lösen. Dennoch kann es vorkommen, daß für eine spezielle Aufgabe eine höhere Verarbeitungsgeschwindigkeit oder besondere Hardwarezugriffe erforderlich sind. Für diesen Fall stehen neben dem externen EEPROM-Speicherchip im Mikroprozessor selbst noch einmal 255 EEPROM Bytes zur Aufnahme von in Assembler programmierten Routinen zur Verfügung. Diese Routinen können aus dem BASIC-Programm heraus aufgerufen werden. Der Befehl dazu lautet

SYS adr

wobei adr eine Konstante ist und die Adresse bestimmt, zu der gesprungen werden soll, beispielsweise &H101, da der interne EEPROM-Bereich an dieser Adresse beginnt. Der Assemblercode muß also per ORG-Befehl an die Adresse &H101 gelegt werden. Die Rückkehr aus einer Assembleroutine zum BASIC erfolgt per RTS-Befehl. Der Datenaustausch zwischen BASIC und Assembler kann über die in der Datei SYSADR.INC aufgelisteten RAM-Adressen erfolgen.

Wie kommt der Assemblercode in den C-Control/BASIC Steuercomputer?

Die in Assembler geschriebenen Zusatzroutinen werden in einer separaten Quelltextdatei (z.B. ADDONS.ASM) gespeichert. Anschließend wird der Assembler aufgerufen, daraus eine Objektcode-Datei im S19-Format zu erstellen (-z.B. ADDONS.S19). Lesen Sie dazu die Dokumentation zu dem Ihnen zur Verfügung stehenden Assembler. Ihr BASIC-Programm, das den SYS-Befehl enthält, muß mit dem Befehl

SYSCODE "ADDONS.S19"

den generierten Code einbinden. Der SYSCODE Befehl darf nur einmal in einem CCBASIC-Programm erscheinen und sollte am Ende, noch hinter eventuellen Tabellendefinitionen stehen.

Problemlösungen

Es lässt sich kein Programm laden

- Sie haben in der IDE die falsche Schnittstelle ausgewählt
- Sie haben die Polarität des Schnittstellensteckers am Roboter vertauscht
- Es sind keine oder leere Akkus im Batteriehalter oder der Batteriehalter ist nicht angesteckt
- Die Sicherung ist defekt, oder der Roboter ist ausgeschaltet
- Der Jumper steckt nicht auf der 4MHz Position, oder fehlt
- Sie haben RESET nicht gedrückt

Die Programme lassen sich laden, aber es tut sich nichts beim Start

- Sie haben die START-Taste nicht gedrückt
- Der Systemtreiber ist nicht geladen (siehe „die erste Funktionskontrolle“)
- Es wird ein permanenter, unkontrollierter RESET ausgeführt

Der Roboter führt im Betrieb unkontrolliert RESETs aus

- Die Akkus sind schlecht geladen oder von geringer Qualität (Innenwiderstand zu gross)
- Es bestehen schlechte elektr. Verbindungen innerhalb des Batteriehalters oder zum Clips
- Hinweise in „4_EINFÜHRUNG_ANTRIEB.BAS“ wurden nicht beachtet

Im Interruptbetrieb treten unerklärliche Funktionsstörungen auf

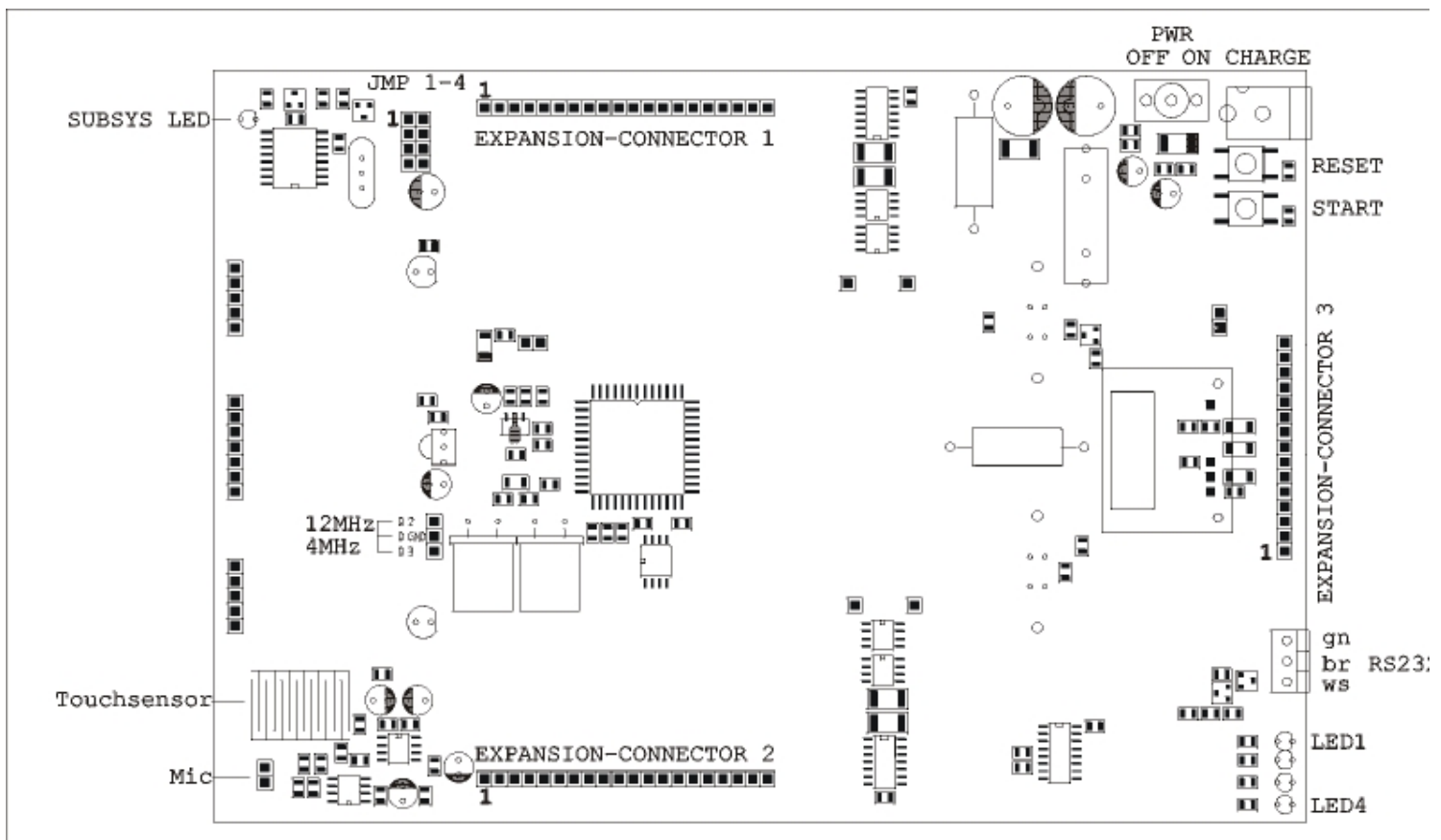
- Hinweise in „3_EINFÜHRUNG_ACS_INT.BAS“ wurden nicht beachtet

Verschieden Sensoren zeigen falsche oder keine Werte

- Es fehlen die Jumper JMP 1 bis 4

Bedienelemente , Anzeigen und Steckverbinder

Expansion-Connector 1



GND	1
AD4	2
AD5	3
AD6	4
AD7	5
AD8	6
TXD uC	7
RXD uC	8
PORT P9	9
PORT P10	10
PORT P11	11
PORT P12	12
PORT P13	13
PORT P14	14
PORT P15	15
PORT P16	16
CHARGE JACK -	17
CHARGE JACK +	18

+ BATT	19
+BATT	20

Expansion-Connector 2

GND	1
UREF	2
RESET	3
DCF/FREQ1	4
FREQ2	5
START	6
SCL	7
SDA	8
PORT P8	9
PORT P7	10
STROBE	11
SCLOCK	12
DATA	13
FT OUT	14
RS 232 TXD	15
RS 232 RXD	16
AMPL/REF POWER +	17
COMM/NAV POWER +	18
SCLOCK NEG	19
VCC	20

Technische Daten

Betriebsspannung	5V +- 10%
Stromaufnahme RX/TX	20/40 mA
Eingangswiderstand Schnittstelle (Pull down)	100 kOhm
Eingangsspegel Schnittstelle	> 3,5 V (H) < 1 V (L)
Ausgangsspegel Schnittstelle	> 4,2 V (H) < 0,4 V (L)
Empfängerempfindlichkeit	-107 dBm
Max. Sendeleistung	10 dBm
User-Interface 4Byte Frame	1,5 ms (mit C-Control)

Bedienungsanleitung

CCRP 5 Basiserweiterung

CCRP 5-BE

Best.-Nr. XXXXXX

Wichtig! Unbedingt lesen!

Bevor Sie die Basiserweiterung zu CCRP5 oder angeschlossene Geräte in Betrieb nehmen, lesen Sie bitte diese Anleitung vollständig durch! Sie erläutert Ihnen die korrekte Verwendung und weist auf mögliche Gefahren hin.

Für Schäden, die aus der Nichtbeachtung dieser Anleitung resultieren, besteht keinerlei Garantieanspruch und übernimmt Conrad Electronic keine Haftung

Inhalt

Wichtig! Unbedingt lesen!.....	1
Inhalt.....	2
Einleitung.....	2
Garantie.....	3
Service.....	3
Produktbeschreibung.....	4
Bestimmungsgemäße Verwendung.....	4
Sicherheitshinweise.....	4
Leistungsmerkmale.....	4
Ausgabemöglichkeiten:.....	4
C-Control II Adaption:.....	4
Adaption zusätzlicher Erweiterungen:.....	5
Inbetriebnahme.....	5
Montage der Basiserweiterung.....	5
Montage des LCD.....	5
Montage der C-Control II Unit.....	5
Software-Installation.....	5
Betrieb mit der CC1.....	6
Betrieb mit der CC2.....	6
CC1 Treiber GATEWAY.BAS.....	6
CC2 Treiber CCRP5.C2.....	6
Beispiele.....	6
Liste der Funktionen im Treiber CCRP5.C2.....	6
Folgende Funktionen kontrollieren die Ausgaben auf das optionale LCD.....	6
Folgende Funktionen dienen der Kontrolle der CCRP5 Hardwareresourcen:.....	8
Laden von Programmen in den Roboter.....	10
Starten von Programmen auf der C-CONTROL II.....	10
Anhalten von Programmen.....	10
Laden der Akkus.....	11
C-Control I/O Ressourcen.....	11
Belegte C-Control II I/O Ressourcen.....	11
Freie C-Control II I/O Ressourcen.....	11
Problemlösungen für den Betrieb mit der CC2.....	12
Es lässt sich kein Programm laden.....	12
Die Programme lassen sich laden, aber es tut sich nichts beim Start.....	12
Der Roboter führt im Betrieb unkontrolliert RESETs aus.....	12
Verschieden Sensoren zeigen falsche oder keine Werte.....	12
Anschlussbelegungen.....	12
Steckverbinder U 1, U 2 zur Hauptplatine.....	12
Steckverbinder C1,C2,C3 für Erweiterungen.....	13
Steckverbinder für LCD1 und LCD2.....	14

Einleitung

Wir danken Ihnen für Ihre Entscheidung zum Erwerb der Basiserweiterung *CCRP5-BE*. Sie ist Zubehör zum C-Control Roboter CCRP5 und als steckbares Modul ausgeführt.

Die Basiserweiterung stellt dem Benutzer zusätzliche Ausgabemöglichkeiten in Form von 8 LEDs und einem 2x16 Char. LC-Display zur Verfügung. Die Basiserweiterung ermöglicht ausserdem den Betrieb des Roboters mit einer C-CONTROL II Unit, die direkt aufgesteckt werden kann.

Gleichzeitig ist die Basiserweiterung Grundlage für alle von Conrad Electronic angebotenen Erweiterungen zum C-Control Roboter CCRP5.

*Conrad Electronic GmbH
D-92240 Hirschau*

Garantie

Bevor Sie die CCRP5-BE oder angeschlossene Geräte in Betrieb nehmen, lesen Sie bitte diese Anleitung vollständig durch. Sie erläutert Ihnen die korrekte Verwendung und weist auf mögliche Gefahren hin.

Für Schäden, die aus der Nichtbeachtung der Bedienungsanleitung resultieren, besteht keinerlei Garantieanspruch und wir übernehmen keine Haftung!

Hinweise zur beschränkten Garantie und Haftung

Conrad Electronic übernimmt keine Garantie dafür, daß die Leistungsmerkmale der CCRP5-BE individuellen Ansprüchen entsprechen, oder daß die Software die zu Demonstrationszwecken mitgeliefert wird, in jedem Fall unterbrechungs- und fehlerfrei arbeiten. Der Anwender trägt das gesamte Risiko bezüglich der Qualität und der Leistungsfähigkeit des Gerätes inklusive aller Software.

Conrad Electronic garantiert die Funktion der mitgelieferten Applikationsbeispiele unter Einhaltung der in den technischen Daten spezifizierten Bedingungen und mit den üblichen vorbehalten gegenüber komplexer Software. Die Gewährleistung von Conrad Electronic beschränkt sich ausschließlich auf den Austausch des Gerätes innerhalb der Garantiezeit bei offensichtlichen Defekten an der Hardware, wie mechanischer Beschädigung, fehlender oder falscher Bestückung elektronischer Bauteile, ausgenommen gesockelter integrierter Schaltkreise und Steckbrücken. Es besteht keine Haftung für Schäden, die unmittelbar durch oder in Folge der Anwendung des Roboters entstehen.

Unberührt davon bleiben Ansprüche, die auf unabdingbaren gesetzlichen Vorschriften zur Produkthaftung beruhen.

Jeder *Basiserweiterung* verläßt das Werk in einwandfreiem und funktionsgeprüften Zustand!

Conrad Electronic bietet für die CCRP5-BE eine **Gewährleistungsdauer von 24 Monaten**. Innerhalb dieser Zeit werden eventuelle Transportschäden bei der Auslieferung, Fertigungsmängel oder Ausfälle am Gerät kostenfrei behoben.

Sollten die Leistungsmerkmale *der Basiserweiterung* oder der mitgelieferten Software Ihren individuellen Ansprüchen nicht genügen, nutzen Sie bitte unsere **Geld-Zurück-Garantie von 14 Tagen**. Senden Sie das Gerät innerhalb dieser Zeit ohne Gebrauchsspuren und in der Originalverpackung zur Erstattung des Warenwertes oder zur Verrechnung zurück.

Alle Fristen gelten ab Datum der Rechnung beziehungsweise des Kassenbons.

Conrad Electronic übernimmt keine Haftung für Folgeschäden an Sachwerten oder Personen, die durch Anwendung des *Basiserweiterung* entstehen!

Service

Zu Ihrer Beratung stellt Conrad Electronic Ihnen ein kompetentes Team von Servicemitarbeitern zur Seite. Jede Anfrage wird schnellstmöglich bearbeitet. Spezialfragen werden an die Entwicklungsingenieure CTC weitergeleitet.

Um unnötige Verzögerungen zu vermeiden, möchten wir Sie jedoch bitten, vor einer Anfrage noch einmal diese Anleitung, die Online-Hilfen der Programmiersoftware, die Text- und Beispieldateien und nach Möglichkeit die Informationsseiten im Internet zu studieren. Meist findet sich so schon die Lösung eines Problems!

Ihre Anfragen richten Sie bitte an unsere Abteilung Technische Kundenbetreuung.

Brief Conrad Electronic GmbH
TKB Computer und Meßtechnik

Klaus-Conrad-Straße 1
92240 Hirschau

Fax 0180 / 53 12 119
Telefon 0180 / 53 12 116
Internet <http://www.conrad.de>

Produktbeschreibung

Bestimmungsgemäße Verwendung

Die Basiserweiterung ist Zubehör zum C-Control Roboter CCRP5 und als steckbares Modul ausgeführt. Die Basiserweiterung stellt dem Benutzer zusätzliche Ausgabemöglichkeiten in Form von 8 LEDs und einem 2x16 Char. LC-Display zur Verfügung. Die Basiserweiterung ermöglicht ausserdem den Betrieb des Roboters (durch einfaches Aufstecken) mit einer C-CONTROL II Unit. Gleichzeitig ist die Basiserweiterung Grundlage für alle von Conrad Electronic angebotenen Erweiterungen zum C-Control Roboter CCRP5.

Eine andere als die bestimmungsgemäße Verwendung ist nicht zulässig.

Sicherheitshinweise

Lesen Sie diesen Abschnitt besonders aufmerksam durch! Bei Nichtbeachtung der Sicherheitshinweise besteht Lebensgefahr durch einen Stromschlag oder Elektrobrand!

Bedingt durch die offene Bauform gibt es bei CCRP5 und der Basiserweiterung **spitze Ecken und scharfe Kanten**. Er darf daher nicht als Spielzeug für Kinder unter 8 Jahren eingesetzt werden. Beaufsichtigen Sie Kinder, die sich beim Betrieb des CCRP5 im Raum befinden. Auf der Oberfläche der Basiserweiterung befinden sich nicht abgedeckte Bauteile und Leiterbahnen. Verursachen Sie keine Kurzschlüsse durch versehentlich abgelegte Metallgegenstände oder Werkzeug.

Leistungsmerkmale

Die Basiserweiterung ist ein steckbares Modul und benötigt keine weiteren Arbeiten zur Montage.

Ausgabemöglichkeiten:

Die Ausgabemöglichkeiten des Roboters CCRP5 sind sehr beschränkt und vor allem Sensordaten lassen sich mit den LEDs nur unzureichend anzeigen. Die Basiserweiterung bietet dem Benutzer ein 2x16 Char. LCD um Betriebszustände oder Sensordaten komfortabel anzuzeigen. Weitere 8 LEDs ermöglichen z.B. das Erkennen von Betriebszuständen auch aus grösserer Entfernung. Beispielprogramme zum Betrieb des LCD und der LEDs ermöglichen auch ungeübten Benutzern den schnellen Einstieg in den Gebrauch.

C-Control II Adaption:

Die Basiserweiterung ist über die zusätzlichen Anzeigemöglichkeiten hinaus ein Adapter für den Betrieb von CCRP5 zusammen mit einer C-Control II Unit. 16-Bit Power und der grosse Speicher der CC2-Unit lassen keine Wünsche mehr offen. Alle Hardwareresourcen der CC2 sind an Steckleisten zugänglich.

Um Ihnen auch hier einen schnellen Einstieg in die Programmierung und den Betrieb des Roboters zusammen mit der CC2 zu ermöglichen, stehen Ihnen eine Anzahl von Treibern und Beispielprogrammen zur Verfügung.

Adaption zusätzlicher Erweiterungen:

Die Basiserweiterung ist Grundlage für weitere von Conrad Electronic geplanten Erweiterungen, bietet aber auch Spielraum eigene Entwicklungen zu montieren. Alle nutzbaren Ressourcen des CC1-Computers auf dem Roboter und der Aufgesteckten CC2 sind an Steckleisten zugänglich

Inbetriebnahme

Montage der Basiserweiterung

Vergewissern Sie sich, dass der Roboter abgeschaltet ist und auch sonst keine Verbindung zu anderen elektrischen Geräten besteht.

Stecken Sie die Basiserweiterung zunächst ohne Kraftaufwand in die dafür vorgesehen Steckbuchsen und vergewissern Sie sich, dass **alle** Pins der Stiftleisten exakt an den zugehörigen Buchsen liegen. Drücken Sie dann die Basiserweiterung mit erhöhtem Kraftaufwand in die Buchsenleisten.

Montage des LCD

Für die Basiserweiterung ist ein LCD als Zubehör erhältlich. Es ist kompatibel zu dem LCD das für das Application-Board (CC1 und CC2) verwendet wird.

Stecken Sie das LCD auf die mit LCD1 (wenn sie das LCD an der CC1 betreiben) oder LCD2 (für den Betrieb mit der CC2) bezeichneten Steckleiste. Achten Sie auf richtige Polung des Steckers. Das rot markierte Kabel liegt jeweils links (in Fahrtrichtung gesehen)

Fixieren Sie das LCD (z.B. mit Klebeband), damit es nicht mit Teilen der Schaltung in Berührung kommt.

Die Basiserweiterung ist damit betriebsbereit. Laden Sie dazu die Demos zur Basiserweiterung in den Roboter und starten Sie diese. Wenn Sie die Basiserweiterung nicht zum Betrieb einer CC2 Unit verwenden, ist die Montage abgeschlossen.

Montage der C-Control II Unit

Auch wenn Sie zusätzlich eine CC2-Unit aufstecken wollen, sollten Sie sich von der Funktion der Basiserweiterung zunächst überzeugen und die Beispielprogramme in den Roboter laden. Wenn Sie sich von der korrekten Funktion der Erweiterung überzeugt haben, ist es erforderlich den Treiber GATEWAY.BAS in den Roboter zu laden. Dieser Treiber (oder ein anderer Treiber Ihrer Wahl) ist zur Kommunikation mit der CC2 unbedingt erforderlich. **Erst dann sollten Sie die CC2 Unit aufstecken.**

Vergewissern Sie sich, dass der Roboter abgeschaltet ist und auch sonst keine Verbindung zu anderen elektrischen Geräten besteht.

Stecken Sie die CC2 Unit zunächst ohne Kraftaufwand in die dafür vorgesehen Steckbuchsen und vergewissern Sie sich, dass **alle** Pins der Stiftleisten exakt an den zugehörigen Buchsen liegen. Drücken Sie dann die C-CONTROL II UNIT mit erhöhtem Kraftaufwand in die Buchsenleisten.

Software-Installation

Eine Integrierte Entwicklungsumgebung (IDE) für die Programmierung der CC2-Unit wurde mit dieser geliefert. Sollten Sie die IDE zur CC2 noch nicht installiert haben, oder sonstige Fragen zur CC2 haben, sehen Sie bitte in das Handbuch der C-Control II Unit.

Mit der Basiserweiterung CCRP5-BE bekommen Sie eine Installations-CD die alle benötigten Programme und Beispiele für den Betrieb des Roboters zusammen mit der CC2 enthält.

Betrieb mit der CC1

Für den Betrieb der Basiserweiterung zusammen mit der C-Control 1 auf der Hauptplatine des CCRP5 finden Sie zwei Beispiele auf der Installations-CD.

1_ERWEITERUNG_LEDS:BAS
1_ERWEITERUNG_LCD.BAS

Laden Sie unbedingt das Beispiel zu den LEDs, da es einen neuen Treiber auf der CC1 installiert. Lesen Sie bitte auch unbedingt die Hinweise im Programm.

Verfahren Sie beim Laden von Programmen wie üblich. Der zusätzliche Schnittstellenstecker ist nur für den Download in die CC2.

Betrieb mit der CC2

Die Kontrolle über Sensoren und Antrieb ist streng an den C-Control I Prozessor auf dem Roboter gebunden und die C-Control II Unit kann darauf nicht direkt zugreifen.

Der Zugriff muss daher **immer** über den CC1 Prozessor auf dem Roboter erfolgen. Im komfortabelsten Fall ist der CC1 Prozessor dann ein richtiger Co-Prozessor, der selbst ein komplexes Programm hat und den Hauptprozessor (die CC2) nennenswert entlastet. Dafür muss das Programm für den Co-Prozessor natürlich genau auf den Anwendungszweck des Roboters zugeschnitten sein.

Sehr viel universeller ist ein „GATEWAY“ dessen alleinige Aufgabe es ist, die Ressourcen des Roboters dem Hauptprozessor zugänglich zu machen.

CC1 Treiber GATEWAY.BAS

Bei den Installationen zur Basiserweiterung auf der CD finden Sie „GATEWAY.BAS“, das genau diese Funktion erfüllt. Installieren Sie es in Ihrem Arbeitsverzeichnis der CC1 IDE und laden Sie es in den Roboter.

Benutzen Sie dazu die Schnittstelle auf dem Roboter (nicht die auf der Basiserweiterung)

Wenn Sie diesen Treiber nicht erweitern oder ändern wollen, so werden sich zukünftig alle Programmierarbeiten und Downloads auf die CC2 beziehen.

CC2 Treiber CCRP5.C2

Die CC2 kommuniziert mit dem CC1 Prozessor über SWCOM und der seriellen Schnittstelle der CC1.

Diese Kommunikation läuft nach einem strengen Protokoll ab, das für Sie sehr mühselig zu programmieren wäre. Der Treiber CCRP5.C2 stellt Ihnen deshalb eine einfache Softwareschnittstelle zur Verfügung um auf die Ressourcen des Roboters und der Basiserweiterung zugreifen zu können, ohne die Protokollierung beachten zu müssen. Ausserdem enthält es die nötigen Treiber zum Betrieb des optionalen Displays.

CCRP5.C2 ist als gemeinsames Modul vorgesehen und sollte in den entsprechenden Ordner (USERLIB) kopiert werden.

Beispiele

Auf der CD finden Sie Beispiele zur Anwendung der Funktionen im Treiber CCRP5.C2

Die Beispiele sind teilweise für den Betrieb der Basiserweiterung zusammen mit einem LCD ausgelegt, sind aber auch ohne LCD hilfreich für das Verständnis.

Liste der Funktionen im Treiber CCRP5.C2

Funktionen zu Ausgabe Ausgabe auf das optionale LCD

function init ()

Initialisierung des LCD

function clear ()
Löschen des Displays

function clear1()
Löschen von Zeile 1

function clear2()
Löschen von Zeile 2

function home ()
Cursor auf Zeile 1, Pos 1

function line2 ()
Cursor auf Zeile 2, Pos 1

function cursoron ()
Cursor einschalten

function cursoroff ()
Cursor ausschalten

function cursorleft ()
Cursor eine Position links rücken

function cursorright ()
Cursor eine Position rechts rücken

function cursorpos(byte line,byte col)
Cursor auf Zeile/Spalte setzen

function print (string s)
Ausgabe eines Strings (50 ms/16Char.)

function fprint (string s)
Ausgabe eines Strings über eine sehr schnelle Routine in Maschinencode (1,5 ms/16 Char).
Sie müssen dazu aber das File CCRP5.HEX in das Segment 3 der CC2 geladen haben.

function scrollleft ()
Displayinhalt eine Position links schieben

function scrollright ()
Displayinhalt eine Position rechts schieben

function showtime ()
Anzeige der Systemzeit

function showdate ()
Anzeige des Systemdatums

function showbar (int barlength)
Anzeige eines Balkens (0-16 Segmente)

function showport (byte port)
Anzeige eines Bytes als Binärzahl

Funktionen zur Kontrolle der CCRP5 Hardwareresourcen:

function init() returns byte

Diese Funktion initialisiert den Treiber und starten den CC1 Prozessor.
Der Rückgabewert ist nur 1 wenn die C-Control 1 aus dem SLEEP-Modus zurückkehrt und die CC2 einschaltet.
In diesem Fall wird von der CC2 auch kein RESET der CC1 veranlasst.
In allen anderen Fällen ist der Rückgabewert 0 und bei der CC1 wurde ein RESET durchgeführt.
Sehen Sie dazu auch die Funktion SLEEP.

function getCNSTAT() returns int

Diese Funktion ruft den Systemstatus der CC1 ab und gibt ihn zurück. Relevant sind dabei nur drei Bits.
Bit 0 = linkes ACS hat angesprochen
Bit 1 = rechtes ACS hat angesprochen
Bit2 = IR Daten empfangen

function FWD (byte speed_l,byte speed_r)

Diese Funktion veranlasst die Vorwärtsfahrt mit der angegebenen Geschwindigkeit(linker/rechter Motor).

function REV (byte speed_l,byte speed_r)

Diese Funktion veranlasst die Rückwärtsfahrt mit der angegebenen Geschwindigkeit.

function ROTL (byte speed_l,byte speed_r)

Diese Funktion veranlasst den Roboter zum Rotieren nach links mit der angegebenen Geschwindigkeit.

function ROTR (byte speed_l,byte speed_r)

Diese Funktion veranlasst den Roboter zum Rotieren nach rechts mit der angegebenen Geschwindigkeit

function getAMBIENT_SENS()

Ruft die Werte der Umgebungssensoren ab. Sie sind als ccrp5.LIGHTL, ccrp5.LIGHTR, ccrp5.MIC und ccrp5.TOUCH verfügbar

function getDISTANCE()

Ruft die Werte der Wegstreckenzähler ab und stellt sie in ccrp5.DISTL und ccrp5.DISTR zur Verfügung

function setACSLO()

function setACSHI()

function setACSMAX()

Funktionen zum Einstellen der ACS-Empfindlichkeit

function txIRDATA(byte address,byte command)

Die Funktion sendet die spezifizierten Daten über die IR-Kommunikationseinheit.

function rxIRDATA()

Ruft Daten von der IR-Kommunikationseinheit ab und stellt sie in ccrp5.IRADR und ccrp5.IRCMD zur Verfügung.

function LEDon(byte led)

function LEDoff(byte led)

Schaltet die LEDs 1 bis 4 auf dem Roboter.

function LEDSoft(byte led)

Schaltet alle vier LEDs aus.

function Lon(byte led)

function Loff(byte led)

Schaltet die LEDs 1 bis 8 auf der Basiserweiterung.

function LSoff(byte led)

Schaltet alle acht LEDs auf der Basiserweiterung aus.

function BEEP(int freq,byte length)

Aktiviert den Beep mit den übergebenen Parametern (entsprechend der Spezifikation für die CC1)

function clearDISTANCE()

Löscht die Wegstreckenzähler auf dem Roboter

function setRC5()**function setREC80()**

Einstellung des IR-Kommunikations –Protokolls

function SLEEP (byte hours,byte minutes)

Veranlasst den CC1 Prozessor den energiesparenden Schlafmodus für den angegebenen Zeitraum zu aktivieren.

- Die CC2 wird abgeschaltet (Speicherverlust bei Variablen !!)
- Die Sensorik wird abgeschaltet

Nach Ablauf der Zeit wird die CC2 wieder mit Spannung versorgt und startet ihr Programm. Mit dem Rückgabewert 1 der Funktion INIT weiss die CC2, dass es sich um eine Rückkehr aus dem SLEEP-Mode handelt und kein RESET durchgeführt werden darf, weil z.B. Betriebsparameter aus der CC1 zurück geladen werden. Die gelbe LED auf der Basiserweiterung zeigt, wenn die CC2 mit Spannung versorgt ist.

function setRTC()

Setzt die Systemzeit der CC1 auf die Systemzeit der CC2 (nicht aber das Datum). Damit wird eine korrekte Erhaltung der Zeit sichergestellt, wenn die CC2 während einer SLEEP-Phase abgeschaltet ist.

function getRTC()

Ladet z.B. nach Ablauf einer SLEEP-Phase die aktuelle Systemzeit von der CC1 auf die CC2 zurück.

function writeFILE(int data)

Sichert Integer-Werte auf das EEPROM in der CC1(z.B. Datensicherung vor einer SLEEP-Phase)

function readFILE()returns int

Ladet Integer-Werte vom EEPROM in der CC1(z.B. Rückladen von Daten nach einer SLEEP-Phase)

function closeFILE()

Der Aufruf ist zum Abschluss eines Datentransfers notwendig

Allgemeiner Ablauf:

writeFILE(wert1)

writeFILE(wert 2)

.

.

.

writeFILE(wert n)

close FILE()

function getSYSTEMDATA()

Stellt die von den A/D-Wandlern auf der CC1 gemessenen Rohdaten für Systemspannung, Motorstrom und Ladestrom in den Variablen ccrp.SYSVOLTAGE,ccrp5.SYSCURRENT und ccrp5.CHRCURRENT zur Verfügung. Eine Anleitung zur Umrechnung in die richtigen physikalischen Grössen finden sie bei den Beispielen.

function checkFAIL() returns int

Durch elektrostatische Entladungen während des Betriebs von CCRP5 wird die C-Control 1 in einzelnen Fällen in den RESET-Zustand versetzt und muss dann von der C-Control 2 neu initialisiert werden.

Benutzen Sie checkFAIL () um das korrekte Arbeiten der CC1 zu verifizieren. Bei korrekter Funktion gibt die Routine 0 als Wert zurück.

Laden von Programmen in den Roboter

Laden in die CC1 auf dem Roboter

Mit dem Roboter wurde Ihnen ein Schnittstellenkabel ausgeliefert, ein 9-poliges Nullmodernkabel (ca. 1,5 Meter) Um ein Programm aus der CC1-IDE in den Roboter zu laden stecken Sie es in die Buchse für die serielle Schnittstelle auf dem Roboter. Das Kabel muß so aufgesteckt werden, dass die weisse Leitung auf der Seite der LEDs ist. Die CC2-Unit darf dazu **nicht** auf der Basiserweiterung gesteckt sein

Programme werden direkt aus der Entwicklungsumgebung in den Roboter geladen. Der Roboter muss dazu eingeschaltet sein und im RESET-Modus sein. Im Menü „Entwicklung“ in der IDE finden sie die entsprechende Auswahl für den Download.

Laden in die CC2 auf der Basiserweiterung

Das gleiche Schnittstellenkabel können sie verwenden um ein Programm aus der CC2-IDE in den Roboter zu laden stecken Sie es dazu in die Buchse für die serielle Schnittstelle auf der Basiserweiterung. Das Kabel muß so aufgesteckt werden, dass die weisse Leitung auf der Seite der LEDs ist.

Programme werden direkt aus der Entwicklungsumgebung in den Roboter geladen. Der Roboter muss dazu eingeschaltet sein und im HOST-Modus sein. Sehen Sie dazu auch die entsprechenden Hinweise im Handbuch zur CC2.

Starten von Programmen auf der C-CONTROL II

Ignorieren Sie bitte die Taster auf der Platine des Roboters. Start und Reset werden ausschliesslich von der CC2 bedient !

Mit Einschalten der Betriebsspannung des Roboters stellen sich folgende Zustände ein:

- 1) Die CC1 befindet sich im RESET-Modus
- 2) Die CC2 führt einen POWER ON RESET aus und startet ihr Programm
- 3) Die Funktion INIT der CC2 startet das Programm auf der CC1 und setzt den Rückgabewert auf 0
Das Programm auf der CC2 bestimmt den weiteren Ablauf des Geschehens

Die Betriebsspannung ist eingeschaltet, das Programm wird durch RESET der CC2 gestartet.

Es wird immer ein RESET auf der CC1 durchgeführt und das Programm gestartet, ausser wenn die CC1 aus dem SLEEP-Modus zurückkehrt und die CC2 gestartet hat (der Rückgabewert der Funktion INIT ist dann 1)

Das Programm auf der CC2 bestimmt den weiteren Ablauf des Geschehens

Anhalten von Programmen

Halten Sie Programme an, indem sie den Kippschalter auf AUS stellen.

Achtung:

Wenn Sie den RESET-Taster auf dem Roboter (nicht auf der Basiserweiterung) drücken, wird die C-Control 1 auf dem Roboter das gespeicherte Programm überschreiben, weil jetzt (RESET) Daten auf der seriellen Schnittstelle als Programm-Download interpretiert werden.

Start und Reset der CC1 werden ausschliesslich von der CC2 bedient !

Laden der Akkus

Details zu den Vorschriften zum Laden von Akkus finden Sie im Handbuch zum Roboter. In den Demos zum Betrieb mit der CC2 finden Sie auch ein Beispiel das den Ladevorgang anzeigt und weitgehend überwacht. Ein Anspruch auf optimale Ladung sowie ein Schutz gegen Überladung kann dann aber nicht erhoben werden.

ACHTUNG:

- **Versuchen Sie niemals den Roboter an eine externe Spannungsversorgung anzuschliessen wenn keine Akkus eingelegt sind**
- **Versuchen Sie niemals den Roboter an eine externe Spannungsversorgung anzuschliessen wenn der EIN/AUS-Schalter auf AUS steht**
- **Schliessen Sie niemals ein anderes, als das als Zubehör empfohlene Steckernetzteil an**

In jedem dieser Fälle besteht sonst die Gewissheit, Bauteile des Roboters durch Überspannung zu zerstören

C-Control I/O Ressourcen

Der C-Control Computer stellt dem Benutzer eine grosse Anzahl von Ports zur Verfügung, die bei der M-Unit und Main-Unit frei verfügbar sind. Hier, bei CCRP5, ist ein Teil dieser Ports für den Betrieb des Roboters notwendig und daher nicht, oder nur eingeschränkt verfügbar. Mit Installation der Basiserweiterung sind die Ressourcen der C-Control I nahezu erschöpft.

Die Basiserweiterung belegt zusätzlich Port 7 (für die LEDs), Port 8 (zum Schalten der CC2 Betriebsspannung, gelbe LED) und die Ports 9 bis 16 für das LCD.

Wenn Sie das LCD nicht angesteckt haben sind die zugehörigen Ports P9-P16 frei
Frei sind ausserdem die Ports AD8,DCF und FREQ2.

Belegte C-Control II I/O Ressourcen

Hier finden Sie eine Übersicht der I/O-Kanäle der CC2 die vom Roboter fest belegt sind.

PORT P1H1	SERIAL DATA SWCOM	RX DATA
PORT P1H2	SERIAL DATA SWCOM	TX DATA
PORT P1H3	CC1 RESET SWITCH	RESET
PORT P1H4	CC1 START SWITCH	START
PORT P1L0-P1L7	LCD DRIVE	LCD

Wenn Sie das LCD nicht angesteckt haben sind die zugehörigen Ports P1L0-P1L7 frei

Freie C-Control II I/O Ressourcen

Alle I/O-Kanäle der CC2, die nicht für den Betrieb des Roboters benötigt werden sind frei und an den Steckleisten verfügbar. Eine Übersicht dazu sehen Sie im Kapitel Anschlussbelegungen.

Problemlösungen für den Betrieb mit der CC2

Es lässt sich kein Programm laden

- Sie haben in der IDE die falsche Schnittstelle ausgewählt
- Sie haben die Polarität des Schnittstellensteckers am Roboter vertauscht
- Sie verwenden die Schnittstelle zur CC1 am Roboter
- Es sind keine oder leere Akkus im Batteriehalter oder der Batteriehalter ist nicht angesteckt
- Die Sicherung ist defekt, oder der Roboter ist ausgeschaltet
- Der Jumper steckt nicht auf der 4MHz Position, oder fehlt
- Sie haben RESET und HOST nicht gedrückt (siehe Manual zur CC2)
- Manche Windows-Versionen führen zu Problemen beim Download aus der IDE, verwenden sie das Download-Tool V1.5
- Die CC2 hat kein Betriebssystem geladen (siehe Manual zur CC2)

Die Programme lassen sich laden, aber es tut sich nichts beim Start

- Sie haben die START-Taste der CC2 nicht gedrückt
- Der Systemtreiber ist nicht geladen (siehe „Betrieb mit der CC2“)
- Es wird ein permanenter, unkontrollierter RESET ausgeführt
- Sie laden Programme die nicht für den Betrieb des Roboters bestimmt sind (Demos zur Programmiersprache C2, zusammen mit der IDE installiert)
- Der Treiber GATEWAY.BAS auf der CC1 wurde zerstört weil Sie die RESET-Taste auf dem Roboter gedrückt haben (RESET und START werden ausschliesslich von der CC2 bedient !!)

Der Roboter führt im Betrieb unkontrolliert RESETs aus

- Die Akkus sind schlecht geladen oder von geringer Qualität (Innenwiderstand zu gross)
- Es bestehen schlechte elektr. Verbindungen innerhalb des Batteriehalters oder zum Clips
- Hinweise in „4_EINFÜHRUNG_ANTRIEB.BAS“ wurden nicht beachtet

Verschieden Sensoren zeigen falsche oder keine Werte

- Es fehlen die Jumper JMP 1 bis 4

Anschlussbelegungen

Steckverbinder U 1, U 2 zur Hauptplatine

Y146	GND1	(U2)	<input type="checkbox"/>
Y147	UREF	(U2)	<input type="checkbox"/>
Y148	RESET	(U2)	<input type="checkbox"/>
Y149	DCF_FREQ1	(U2)	<input type="checkbox"/>
Y150	FREQ2	(U2)	<input type="checkbox"/>
Y151	START	(U2)	<input type="checkbox"/>
Y152	SCL	(U2)	<input type="checkbox"/>
Y153	SDA	(U2)	<input type="checkbox"/>
Y154	PORT8_BUTTON	(U2)	<input type="checkbox"/>
Y155	PORT7_BUTTON	(U2)	<input type="checkbox"/>
Y156	STROBE	(U2)	<input type="checkbox"/>
Y157	SCLOCK	(U2)	<input type="checkbox"/>
Y158	DATA	(U2)	<input type="checkbox"/>
Y159	FT_OUT	(U2)	<input type="checkbox"/>
Y160	RS232_TXD	(U2)	<input type="checkbox"/>
Y161	RS232_RXD	(U2)	<input type="checkbox"/>
Y162	QP3	(U2)	<input type="checkbox"/>
Y163	COM/NAV_PWR	(U2)	<input type="checkbox"/>
Y164	SCLOCK	(U2)	<input type="checkbox"/>
Y165	VCC	(U2)	<input type="checkbox"/>

Y1	GND1	(U1)	
Y2	MIC	(U1)	
Y3	AD5	(U1)	
Y4	AD6	(U1)	
Y5	AD7	(U1)	
Y6	AD8	(U1)	
Y7	TXD	(U1)	
Y8	RXD	(U1)	
Y9	PORT9	(U1)	
Y10	PORT10	(U1)	
Y11	PORT11	(U1)	
Y12	PORT12	(U1)	
Y13	PORT13	(U1)	
Y14	PORT14	(U1)	
Y15	PORT15	(U1)	
Y16	PORT16	(U1)	
Y17	CHARGE JACK-	(U1)	
Y18	CHARGE JACK+	(U1)	
Y19	+ BAT	(U1)	
Y20	+ BAT	(U1)	

Steckverbinder C1,C2,C3 für Erweiterungen

Y38	PORT9	(C1)		Y21	GND1	(C1)
Y39	PORT10	(C1)		Y22	MIC	(C1)
Y40	PORT11	(C1)		Y23	AD5	(C1)
Y41	PORT12	(C1)		Y24	AD6	(C1)
Y42	PORT13	(C1)		Y25	AD7	(C1)
Y43	PORT14	(C1)		Y26	AD8	(C1)
Y44	PORT15	(C1)		Y27	CHARGE JACK-	(C1)
Y45	PORT16	(C1)		Y28	CHARGE JACK+	(C1)
Y46	P1L1	(C1)		Y29	P1L0	(C1)
Y47	P1L3	(C1)		Y30	P1L2	(C1)
Y48	P1L5	(C1)		Y31	P1L4	(C1)
Y49	P1L7	(C1)		Y32	P1L6	(C1)
Y50	GND	(C1)		Y33	GND	(C1)
Y51	VCC	(C1)		Y34	VCC	(C1)
Y52	GND	(C1)		Y35	GND	(C1)
Y53	+5V	(C1)		Y36	PWR	(C1)
Y54	+5V	(C1)		Y37	PWR	(C1)

Y129	GND1	(C2)		Y112	RS232 TXD	(C2)
------	------	------	--	------	-----------	------

Y130	GND1	(C2)		Y113	RS232 RXD	(C2)
Y131	GND1	(C2)		Y114	QP3	(C2)
Y132	UREF	(C2)		Y115	COM/NAV PWR	(C2)
Y133	DCF FREQ1	(C2)		Y116	VCC	(C2)
Y134	FREQ2	(C2)		Y117	VCC	(C2)
Y135	SCL	(C2)		Y118	+ BAT	(C2)
Y136	SDA	(C2)		Y119	+ BAT	(C2)
Y137	P1H.5	(C2)		Y120	P1H.0	(C2)
Y138	P1H.7	(C2)		Y121	P1H.6	(C2)
Y139	FREQ 1	(C2)		Y122	Beep	(C2)
Y140	ADC 1	(C2)		Y123	ADC 0	(C2)
Y141	ADC 3	(C2)		Y124	ADC 2	(C2)
Y142	ADC 5	(C2)		Y125	ADC 4	(C2)
Y143	ADC 7	(C2)		Y126	ADC 6	(C2)
Y144	CANH	(C2)		Y127	CANL	(C2)
Y145	nc	(C2)		Y128	nc	(C2)

Y64		Y65	GND1	(C3)
Y69		Y70	GND1	(C3)
Y74		Y75	PORT7	(C3)
Y79		Y80	PORT7	(C3)
Y85		Y86	VCC	(C3)
Y91		Y92	VCC	(C3)

Steckverbinder für LCD1 und LCD2

Y58	LCD1 P12	(13)		Y59	LCD1 P11	(14)
Y61	LCD1 P10	(11)		Y62	LCD1 P9	(12)
Y66	LCD1 nc	(9)		Y67	LCD1 nc	(10)
Y71	LCD1 nc	(7)		Y72	LCD1 nc	(8)
Y76	LCD1 P15	(5)		Y77	LCD1 P13	(6)
Y81	LCD1 P14	(3)		Y82	LCD1 POT1	(4)
Y87	LCD1 VCC	(1)		Y88	LCD1 GND	(2)

Y95	LCD2 P1L7	(13)		Y96	LCD2 P1L6	(14)
Y98	LCD2 P1L5	(11)		Y99	LCD2 P1L4	(12)
Y101	LCD2 nc	(9)		Y102	LCD2 nc	(10)
Y104	LCD2 nc	(7)		Y105	LCD2 nc	(8)
Y106	LCD2 P1L2	(5)		Y107	LCD2 P1L1	(6)
Y108	LCD2 P1L0	(3)		Y109	LCD2 POT1	(4)
Y110	LCD2 +5V	(1)		Y111	LCD2 GND	(2)