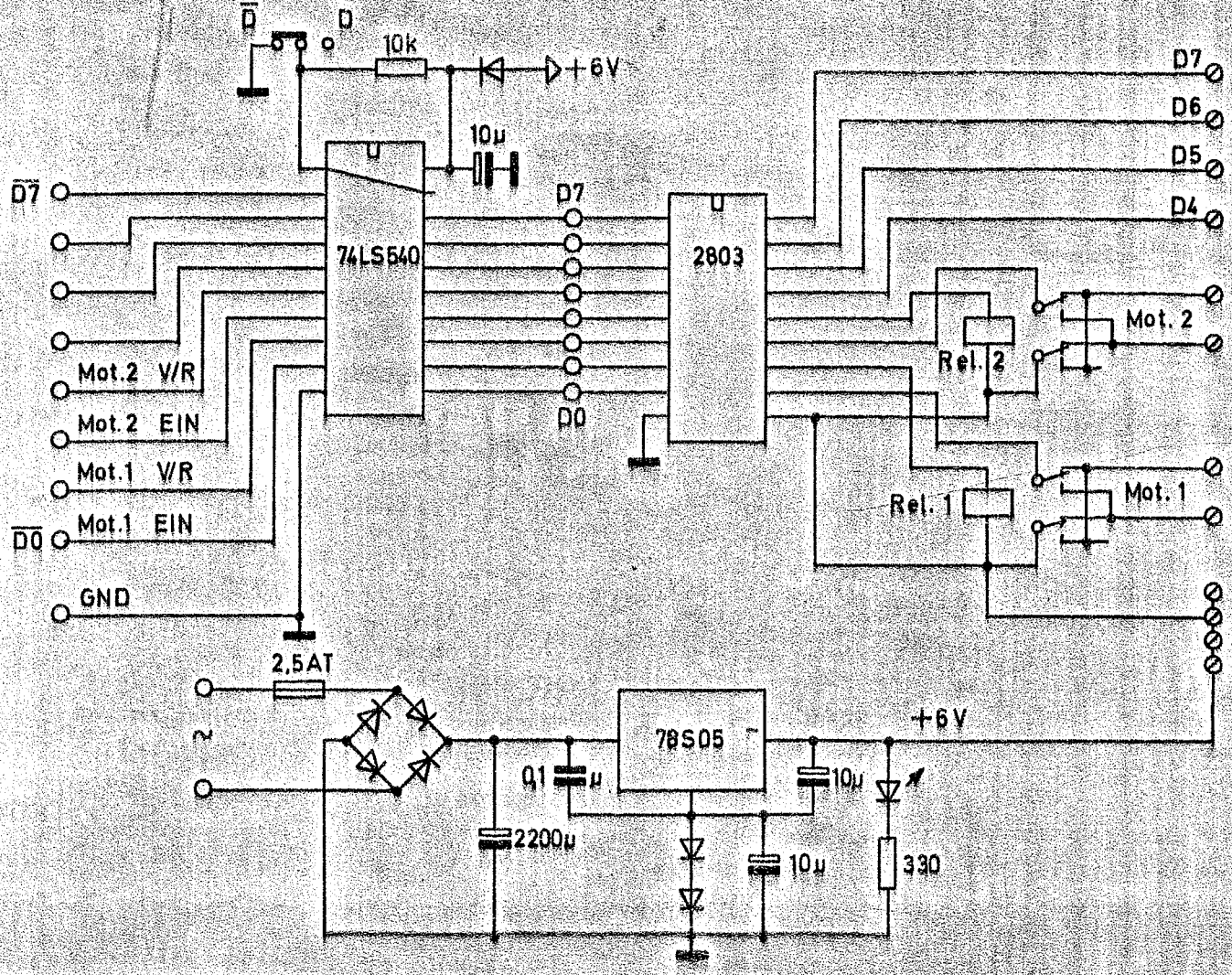
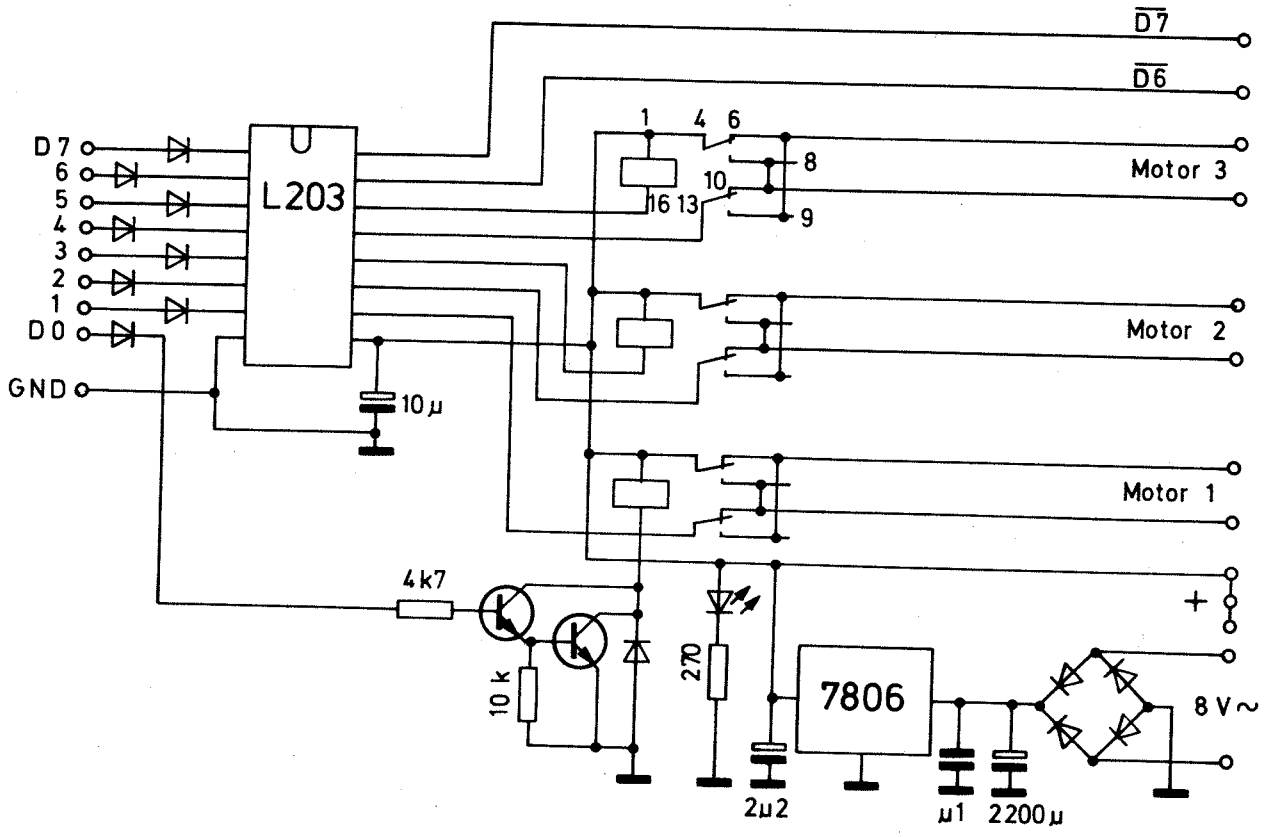
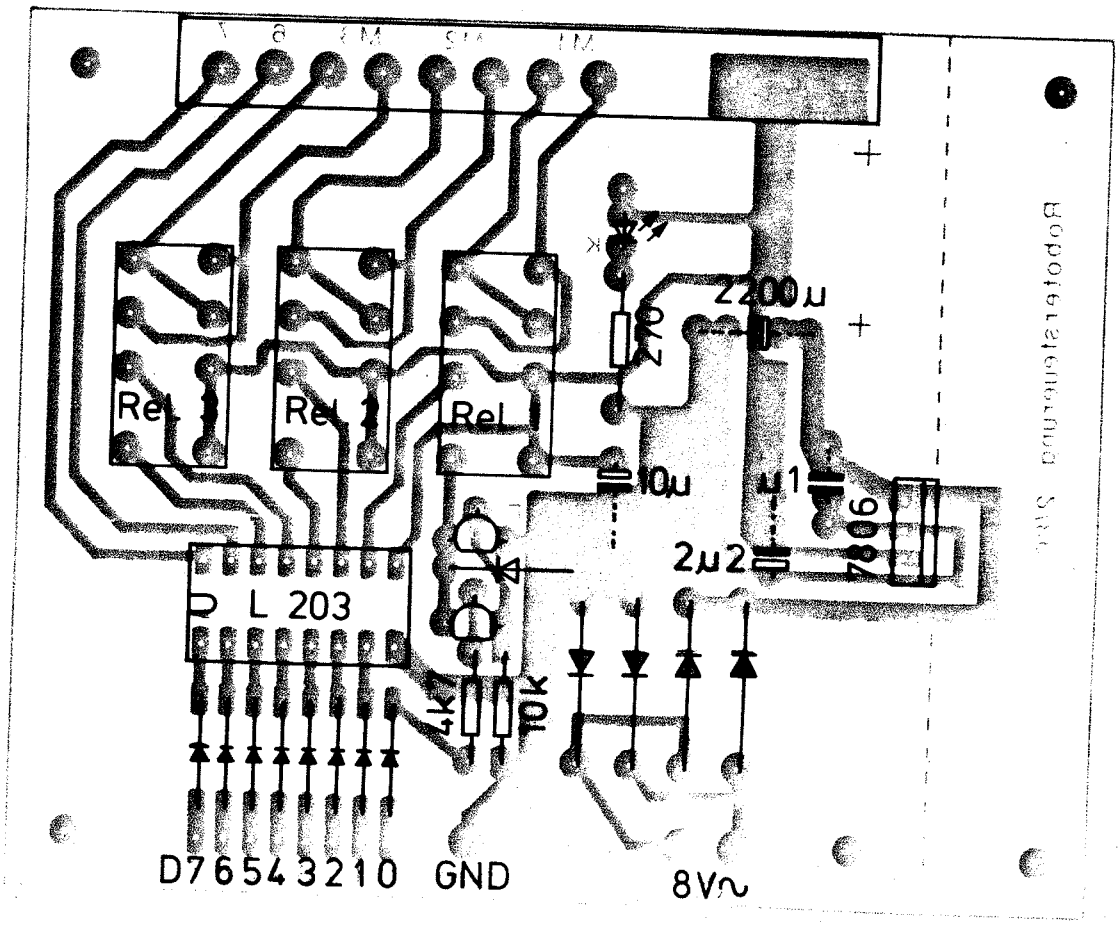
Universal interface



B&G 078702


D7
D6
D7
6
5
4
3
2
1
D0
GND
U
L203
10µ
1
4
6
8
10
16
13
9
Motor 3
Motor 2
Motor 1
4k7
10 k
270
7806
2µ2
µ1
2200µ
8 V~
+


Rel
Rel
Rel
2200µ
270
10µ
µ1
2µ2
L 203
4k7
10k
D7 6 5 4 3 2 1 0 GND
8V~
+

„Steuersprache" mit Anzeige

LABEL	ADR	HEX	BEFEHL	OPERAND	KOMMENTAR
INIT	1600	BB 95	ZBSR	*CDIS	Anzeigespeicher Ø
		BB 8F	ZBSR	*INIT	alle Register Ø
		CC Ø8 FF	STRA	RØ '08FF'	
		D4 Ø9	WRITE	RØ PORT	
		17	RETC	UN	
M1r	1610	Ø4 37	LODI	RØ 37	„M" laden
		CC Ø8 ØD	STRA	RØ Ø8ØD	für Digit 5
		Ø4 Ø6	LODI	RØ Ø6	„1" laden
		CC Ø8 ØE	STRA	RØ Ø8ØE	für Digit 4
		Ø4 5Ø	LODI	RØ 5Ø	„r" laden
		CC Ø8 ØF	STRA	RØ Ø8ØF	für Digit 3
		Ø4 Ø1	LODI	RØ Ø1	Motorwert für „RECHTS"
		3F 16 BØ	BSTA	UN „AUSGABE"	laden und ausgeben
		17	RETC	UN	zurück ins Hauptprogramm
M1L	1625	Ø4 37	LODI	RØ 37	„M" laden
		CC Ø8 ØD	STRA	RØ Ø8ØD	für Digit 5
		Ø4 Ø6	LODI	RØ Ø6	„1" laden
		CC Ø8 ØE	STRA	RØ Ø8ØE	für Digit 4
		Ø4 38	LODI	RØ 38	„L" laden
		CC Ø8 ØF	STRA	RØ Ø8ØF	für Digit 3
		Ø4 Ø3	LODI	RØ Ø3	Motorwert "LINKS"
		3F 16 BØ	BSTA	UN „Ausgabe"	laden und ausgeben
		17			zurück ins HP
M2r	1640	Ø4 37	LODI	RØ 37	„M" laden
		CC Ø8 ØD	STRA	RØ Ø8ØD	für Digit 5
		Ø4 5B	LODI	RØ 5B	„2" laden
		CC Ø8 ØE	STRA	RØ Ø8ØE	für Digit 4
		Ø4 5Ø	LODI	RØ 5Ø	„r" laden
		CC Ø8 ØF	STRA	RØ Ø8ØF	für Digit 3
		Ø4 Ø4	LODI	RØ Ø4	Motorwert für RECHTS
		3F 16 BØ	BSTA	UN „Ausgabe"	laden und ausgeben
		17	RETC	UN	zurück ins HP
M2L	1655	Ø4 37	LODI	RØ 37	„M2L" laden
		CC Ø8 ØD	STRA	RØ Ø8ØD	und an
		Ø4 5B	LODI	RØ 5B	Digit 5-3
		CC Ø8 ØE	STRA	RØ Ø8ØE	ausgeben
		Ø4 38	LODI	RØ 38	
		CC Ø8 ØF	STRA	RØ Ø8ØF	
		Ø4 ØC	LODI	RØ ØC	Motorwert LINKS
		3F 16 BØ	BSTA	UN „Ausgabe"	laden u. ausgeben
		17	RETC	UN	zurück ins HP
L1ein	1670	Ø4 38	LODI	RØ 38	„L" laden
		CC Ø8 11	STRA	RØ Ø811	für Digit 1
		Ø4 Ø6	LODI	RØ Ø6	„1" laden
		CC Ø8 12	STRA	RØ Ø812	für Digit Ø
		Ø4 4Ø	LODI	RØ 4Ø	Bit D6 setzen
		CC Ø8 FF	STRA	RØ Ø8FF	Wert in Ø8FF speichern
		17	RETC	UN	zurück ins HP
L1aus	1680	Ø4 ØØ	LODI	RØ ØØ	Anzeige „L1"
		CC Ø8 11	STRA	RØ Ø811	löschen
		CC Ø8 12	STRA	RØ Ø812	Bit D6 löschen
		CC Ø8 FF	STRA	RØ Ø8FF	und Wert in Ø8FF
		17	RETC	UN	speichern, zurück

„Steuersprache" mit Anzeige — Blatt 2

LABEL	ADR	HEX	BEFEHL	OPERAND	KOMMENTAR
L2 ein	1690	04 38	LODI	R0 38	„-2"
		CC 08 0D	STRA	R0 08 0D	für Digit 5
		04 5B	LODI	R0 5B	und 4 laden
		CC 08 0E	STRA	R0 08 0E	
		04 80	LODI	R0 80	Bit D7 setzen
		3F 16 B0	BSTA	UN „Ausgabe"	und ausgeben
		17	RETC	UN	zurück
L2 aus	16A0	04 00	LODI	R0 0	„-2" löschen
		CC 08 0D	STRA	R0 08 0D	
		CC 08 0E	STRA	R0 08 0E	
		04 01	LODI	R0 0	Bit D7 zurücksetzen
		3F 16 B0	BSTA	UN „Ausgabe"	und ausgeben
		17	RETC	UN	zurück ins HP
Ausgabe	16B0	6C 08 FF	IORA	R0 08FF	(R0) und (08FF)
		D4 09	WRTE	R0 PORT	verknüpfen und ausg.
		3F 16 E0	BSTA	UN DELAY	verzögern
		6E 08 FF	IORA	R2 08 FF	(R2) = 0 mit (08FF)
		D6 09	WRTE	R2 PORT	verknüpfen und ausgeb.
		17	RETC	UN	L1 wird „gerettet"
ENDE	16C0	04 79	LODI	R0 79	„ENDE"
		CC 08 0D	STRA	R0 08 0D	nach Digit 5-2
		CC 08 10	STRA	R0 08 10	laden und
		04 54	LODI	R0 54	für ≈ 1 sec
		CC 08 0E	STRA	R0 08 0E	anzeigen
		04 5E	LODI	R0 5E	
		CC 08 0F	STRA	R0 08 0F	
		04 00	LODI	R0 00	Interface ausschalten
		D4 09	WRTE	R0 PORT	
		3F 16 E0	BSTA	UN DELAY	
		40	HALT		Programm anhalten
DELAY	16E0	0E 08 FE	LODA	R2 08FE	der Wert in 08FE
LOOP		07 FF	LODI	R3 FF	bestimmt DELAY
		BB 8B	ZBSR	* DIS	Anzeige aktivieren
		FB 7C	BDRR	R3 LOOP	
		FA 7A	BDRR	R2 LOOP	
		17	RETC	UN	zurück
					wenn (08FE) = 18(h)
					ist DELAY ca 1 sec
					lang

EINSATZ EINER STEUERSPRACHE MIT DEM UNIVERSALINTERFACE

Die eben beschriebenen Steuerungen lassen sich mit Hilfe des Programmeproms vereinfachen. Im Adressbereich 1600 - 1700 sind 11 Unterprogramme fest abgespeichert. Sie dienen der Ansteuerung von Funktionsmodellen, die man sich z.B. aus den handelsüblichen Konstruktionsbaukästen herstellt. Die Ausgaben der "Steuersprache" erfordern wegen der Bitzuordnungen das Universalinterface zur Motorsteuerung an Port 09.
Über die Portadressen 0A und 0B können A/D Wandler betrieben werden. Durch die Wahl geeigneter Sensoren kann man z.B. Licht, Wärme und Druck als Spannungsänderungen in den Computer einlesen, die Werte auswerten und die Stellglieder (Aktoren) - hier Gleichstrommotoren - in digital arbeitenden Regelkreisen steuern.

Die Steuersprache arbeitet mit folgenden Bitzuordnungen:

Bit D0:	Motor 1 ein/aus	"1" = ein
Bit D1:	Motor 1 rechts/links	"0" = rechts; "1" = links
Bit D2:	Motor 2 ein/aus	"1" = ein
Bit D3:	Motor 2 rechts/links	"0" = rechts; "1" = links
Bit D4 und D5 nicht verwendet		
Bit D6:	Magnet o.ä. ein/aus	
Bit D7:	Lampe, Summer usw. ein/aus	

Die Verarbeitung von Bit D6 und Bit D7 ist verschieden. Die Unterschiede werden in den Programmbeschreibungen erläutert.

Die Unterprogramme sind mit einer Anzeigefunktion für das Display gekoppelt. Das ordnungsgemäße Funktionieren der Programme läßt sich somit auch auf dem Display verfolgen.

Verzeichnis und Beschreibung der Unterprogramme

Adresse	Name/Funktion	Anzeige
1600	INIT Der Anzeigespeicher 080D - 0812 wird gelöscht, die 7 Arbeisregister und die Adresse 08FF erhalten den Wert 00. Das Interface wird abgeschaltet.	keine
1610	M1 Rechtslauf Der Motor 1 läuft für eine bestimmte Zeit rechts herum. Die Dauer wird durch den Inhalt der Adresse 08FE bestimmt. Für 1s wird der Wert 18 benötigt. Er ist vor Programmbeginn einzugeben.	M1r
1625	M1 Linkslauf Motor 1 läuft links herum. Sonst wie bei 1610.	M1L

1640	M2 Rechtslauf Motor 2 läuft rechts herum.	M2r
1655	M2 Linkslauf Motor 2 läuft links herum.	M2L
1670	L1 ein Bit D6 (40hex) wird gesetzt und nach 08FF gespeichert.	L1 mit M1, M2 oder L2
1680	L1 aus Bit D6 wird gelöscht. Die Adresse 08FF erhält wieder den Wert 00.	nur M1, M2 oder L2
1690	L2 ein Eine Last (Lampe, Summer ...) wird für 1s eingeschaltet. Kürzere bzw längere Einschaltzeiten können über den Inhalt von 08FE eingestellt werden.	L2
16A0	L2 aus Die Last wird für die gewählte Zeit ausgeschaltet.	keine
16C0	ENDE Die Anzeige "Ende" erscheint auf dem Display. Nach Ablauf der voreingestellten Verzögerung erlischt die Anzeige, das Interface wird abgeschaltet und der Rechner angehalten.	EndE
16E0	DELAY Verzögerungsroutine. Der Inhalt von 08FE bestimmt die Dauer. Der Hexwert 18 gilt für 1 Sekunde. Während DELAY wird mit Hilfe der Monitorroutine DIS (BB 8B) der Inhalt des Anzeigespeichers 080D - 0812 auf dem Display angezeigt. Diese Funktion läßt sich demnach auch von anderen Programmen nutzen.	variabel

PROGRAMMBEISPIELE

Beispiel 1: Motor 1 soll 2 Sekunden rechtsdrehen und dann anhalten.

```
08FE    18                              Wert für 1s
0900    3F 16 00  BSTA,UN INIT          Steuersprache initialisieren
0903    3F 16 10  BSTA,UN M1 rechts     Motor 1 Rechtslauf für 1s
0906    3F 16 10  BSTA,UN M1 rechts     noch eine Sekunde rechtsdrehen
0909    3F 16 C0  BSTA,UN ENDE          Programm beenden
```

Beispiel 2: Motor 1 soll kontinuierlich 2s rechts-, dann 2s links-drehen.

```
08FE     18
0900     3F 16 00   BSTA,UN INIT
0903     3F 16 10   BSTA,UN M1 rechts
0906     3F 16 10   BSTA,UN M1 rechts
0909     3F 16 25   BSTA,UN M1 links
090C     3F 16 25   BSTA,UN M1 links
090F     1F 09 03   BCTA,UN '0903'        Programm wiederholen
```

Beispiel 3: Blinklicht mit L2. An D7 des Interfaces schließe man eine Lampe (z.B. 3,8 V 0,07A) gegen +6V an. Als Lampenersatz kann die LED D7 an Port 09 dienen.

```
08FE     18
0900     3F 16 00   BSTA,UN INIT
0903     3F 16 90   BSTA,UN L2 ein       Lampe für 1s einschalten
0906     3F 16 A0   BSTA,UN L2 aus       Lampe für 1s ausschalten
0909     1F 09 03   BCTA,UN '0903'        wiederholen
```

Beispiel 4: Die Lampe L2 soll 10 mal blinken. Programmierung eines Zählers mit Register R1.

```
08FE     18
0900     3F 16 00   BSTA,UN INIT
0903     05 0A      LODI,R1 0A            Zähler laden
0905     3F 16 90   BSTA,UN L2 ein
0908     3F 16 A0   BSTA,UN L2 aus
090B     FD 09 05   BDRA,R1 '0905'        Zähler - 1; wenn nicht Null,
                                          weiter bei Adresse 0905, sonst
090E     3F 16 C0   BSTA,UN ENDE          Programm beenden
```

Beispiel 5: Robotermodell aus Drehkranz mit starrem Greifarm und E-Magnet. Der Roboter soll Eisenstücke von einem Förderband nehmen und absetzen. Der Bewegungsablauf sei: M1 rechts 5s - Magnet ein - M1 links 5s - Magnet aus - Programm wiederholen.

```
08FE     78                               5 * 18 Delay jetzt 5s
0900     3F 16 00   BSTA,UN INIT
0903     3F 16 10   BSTA,UN M1 rechts
0906     3F 16 70   BSTA,UN L1 ein        Magnet ein
0909     3F 16 25   BSTA,UN M1 links      Magnet bleibt eingeschaltet
090C     3F 16 80   BSTA,UN L1 aus        Magnet aus
090F     1F 09 03   BCTA,UN '0903'        Programm wiederholen
```

Dieses Programm kann leicht mit einem Zähler (R1) kombiniert werden. Die Programmbefehle sind analog Beispiel 4 zu wählen. Der einfache Roboter kann dann so programmiert werden, daß er eine bestimmte Anzahl von Eisenstücken vom Förderband nimmt.

Beispiel 6: Temperaturregelung (Klimaanlage)
Motor 1 wird im Rechtslauf als Ventilator betrieben. Beim Überschreiten einer Grenztemperatur soll er anlaufen und die Umgebung kühlen, bis die normale Temperatur wieder erreicht ist.

Zur Temperaturmessung schließen wir einen A/D Wandler an die Portadresse 0A an; der Anschluß 0E- des A/D Wandlers muß deshalb mit dem Lötstift 0A der Portplatine verbunden werden. Mit dem DIL-Schalter wird Meßbereich 1 (0 -2,55V) eingestellt.
Als Temperatursensor dient ein NTC (einige Kiloohm), der mit einem regelbaren Widerstand von etwa 10kOhm einen Spannungsteiler zwischen +5V und 0V des Computers bildet. Der Spannungsteiler begrenzt die am NTC abfallende Spannung auf 2,55V. Mit dem Regelwiderstand kann der Skalenanfangswert der Temperatur bzw Spannung für das Programm frei gewählt werden. (vergl. Anschlußskizze für Beschaltung)

Die Einstellung des A/D Wandlers wird mit einem kleinen Testprogramm vorgenommen:

```
0900     54 0A      REDE,R0 0A          Lies den A/D Wandler nch R0
0902     D4 09      WRTE,R0 09          zeige den Binärwert am Port
0904     1F 09 00   BCTA,UN '0900'      wiederhole die Abfrage
```

Nach RUN erscheint der digitalisierte Spannungswert, der am NTC abfällt, an den 8 Portleds. Der Spannungswert ist ein Maß für die Erwärmung des NTC. Bei der Anzeige FF wurde der Meßbereich u.U. überschritten und der Regler muß soweit verstellt werden, bis die gewünschte Anzeige erreicht ist. Durch Erwärmen bzw Abkühlen des NTC prüfen wir, ob der A/D Wandler reagiert. Die Binärwerte für Normaltemperatur (20°C) und Erwärmung merken wir uns für das folgende Programm.
Der Sollwert sei z.B. 00 - 1F, der Grenzwert 20.

Wir programmieren:

```
08FE     18
0900     77 02      PPSL,COM            logischen Vergleich wählen
0902     3F 16 00   BSTA,UN INIT
0905     54 0A      REDE,R0 0A          A/D Wandler lesen
0907     E4 1F      COMI,R0 1F          Temperaturwert 1F?
0909     3D 16 10   BSTA,GT M1 rechts   wenn größer, kühlen; sonst
090C     1F 09 05   BCTA,UN '0905'      A/D Wandler weiter abfragen
```

Zur Erinnerung: Die Befehle von 0905 - 090B bilden eine logische Einheit (wenn - dann - Beziehung). Für den bedingten Sprung in das Unterprogramm gibt es 3 Möglichkeiten, die auf das Ergebnis des Vergleichs von R0 mit der Maske 1F abgestimmt sind:

3D 'Adresse' "Verzweige ins UP, wenn R0 größer als Maskenwert."
3C 'Adresse' "Verzweige ins UP, wenn R0 gleich Maskenwert."
3E 'Adresse' "Verzweige ins UP, wenn R0 kleiner als Maskenwert."

EINFACHE STEUERUNGSPROGRAMME MIT A/D WANDLUNG

COMPARE - BEFEHLE, BEDINGTE SPRUNGBEFEHLE (BCTA UND BCFA)

Bei den bisher bearbeiteten Steuerungsprogrammen lagen die Daten, die vom Rechner erfaßt und verarbeitet wurden, meistens in **digitaler** Form vor. Die Abfrage eines Positionstasters liefert z.B. immer das binäre Signal 0 oder 1. In der Mehrzahl der Anwendungen fallen aber Daten in analoger Form an. Wenn z.B. Temperaturen, Lichtintensität, Druck, Drehwinkel usw. durch geeignete Meßwertaufnehmer (Sensoren) in elektrische Signale umgewandelt werden, entstehen Spannungen, die sich **analog** mit den gemessenen Vorgängen ändern. Wenn sich z.B. bei einem Sensor eine sich ändernde Spannung zwischen 0 und 2 Volt ergibt, kann der Computer damit nichts anfangen. Die Spannungswerte müssen erst in ein Bitmuster verwandelt werden. Dazu dient der **ANALOG-DIGITAL-WANDLER** (A/D-Wandler), den wir jetzt an unseren Rechner anschließen. Die Funktion, Inbetriebnahme und Einstellung der verschiedenen Meßbereiche sind im Begleitbuch in Kapitel 11 ausführlich beschrieben. Wir konzentrieren uns also auf die Programmierung von Anwendungen mit dem A/D-Wandler.

Der A/D-Wandler kann über die Portadresse 0A gelesen werden. Wir verbinden daher den Anschluß 0A auf der Portplatine mit einem kleinen Prüfkabel mit dem Anschluß 0E des A/D-Wandlers. An die Eingangsbuchse wird ein Potentiometer von ca. 10k Ohm angeschlossen. Die Potentiometer aus den ft-Kästen **computing** sind ebenfalls geeignet. Der Mittelanschluß des Potis erhält die Signalleitung, die zur Eingangsbuchse führt. Die 0 Volt-Leitung kommt an einen äußeren Potentiometeranschluß. Der dritte Potentiometeranschluß erhält ein Prüfkabel mit Krokodilklemme, das man an eine Flachbatterie anklemmen kann. Die Flachbatterie liefert die konstante Meßspannung, die durch Verstellen des Potis zwischen 0 und 2,55 Volt verändert werden kann. Der Wert 2,55 Volt ist der Maximalwert für Meßbereich 1 (DIL-Schalter auf 1). Die Verdrahtung des Potentiometers ist aus Bild 11.24 auf Seite 209 im Begleitbuch zu entnehmen.

Wir laden jetzt ein kleines **Schleifenprogramm**, das wir in Zukunft immer als ein Prüfprogramm für den A/D-Wandler verwenden können.

```
0900 54 0A    REDE,R0 Port A      Lies den A/D-Wandler nach R0
0902 D4 09    WRTE,R0 Port 9      Zeige den Inhalt von R0 an
0904 1F 09 00 BCTA,UN '0900'      Springe an den Anfang zurück
```

An den 8 Portleds erscheinen in Abhängigkeit von der Potentiometerstellung verschiedene Binärwerte, die wir umrechnen müssen. Die zugehörige Analogspannung könnte man z.B. mit einem Drehspulmeßinstrument erfassen. Es muß parallel zu dem Teil des Widerstands angeschlossen werden, an dem die Spannung abfällt.

AUFGABE 1: Wenn eine beliebige Taste der Hex-Tastatur gedrückt wird, soll der A/D-Wandler eingelesen werden. Der Binärwert soll an den Portleds angezeigt werden. Der Binärwert ist außerdem in der Adresse 0A00 zu sichern. Danach soll der Rechner anhalten.

```
0900   BB 8D      ZBSR*KIN              Tastatur abfragen bis
                                         Taste gedrückt
0902   54 0A      REDE,R0 Port A        A/D Wandler lesen
0904   D4 09      WRTE,R0 Port 9        binär anzeigen
0906   CC 0A 00   STRA,R0 '0A00'        Wert speichern
0909   40         HALT                  Rechner anhalten
```

Für die nächste Anwendung wird das Potentiometer in eine drehbare Plattform (Drehkranz) eingebaut. Die Drehbewegung des Drehkranzes soll sich auf die Potentiometerachse übertragen. Die 256 verschiedenen Spannungswerte dienen dann als ein Maß für die **POSITION** des Drehkranzes. Der Antrieb des Drehkranzes erfolgt über einen ft-mini-mot mit Getriebe und Antriebsschnecke. Von der gewählten Übersetzung ist es abhängig, ob alle theoretisch möglichen Positionswerte erreicht werden können. Es ist außerdem zu bedenken, daß die meisten Potentiometer nur einen Einstellbereich von 270° haben. Eine vollständige 360° Drehung kann also nicht erfaßt werden. Die Positionswerte sollen bei der **Rechtsdrehung** des Drehkranzes **ansteigen.** In der **Nullposition** wird ein Taster betätigt, der neben dem Drehkranz montiert ist. Er wird an D0 von Port 08 angeschlossen.
Der beschriebene Aufbau ähnelt der **ft-Werkzeugmaschine.** Die Justierung des Potentiometers und des Starttasters nehme man mit dem kleinen Testprogramm für den A/D-Wandler von der vorigen Seite vor. Unter Umständen muß der nutzbare Bereich des Potentiometers durch Reihenschaltung eines Festwiderstands vergrößert werden. Den erforderlichen Wert muß man ausprobieren. Beim ft-Poti beträgt er einige Kiloohm.
Der Antrieb des mini-mot erfolgt über das Universalinterface. Es werden die Datenleitungen D0 und D1 benötigt. D0 steuert die Motorspannung, D1 die Richtungsumkehr durch ein Umpolrelais. Die hexadezimalen **Motorwerte** sind 01 für **Rechtslauf** und 03 für **Linkslauf.**
Auf den Drehkranz können wir z.B. das Modell einer **Radarantenne,** einer **Solarzelle** oder eines **Greifarms** setzen und folgende Steuerungsaufgabe programmieren:

Aufgabe 2: Ein Motor soll einen Drehkranz in eine vorher festgelegte Sollposition fahren. Die Position wird durch Abfrage eines Potentiometers und A/D-Wandlung der Spannungswerte bestimmt. In der Sollposition soll der Drehkranz anhalten. Vor dem Anfahren der Position wird erst die Nullstellung des Drehkranzes angefahren, sie wird durch Abfrage des Startschalters erkannt. Die Sollposition darf nicht überfahren werden.

PROGRAMMBESCHREIBUNG: Verwendung des COMPARE-Befehls

In dem folgenden Programm muß die **Sollposition** zunächst in einer Speicherstelle (z.B. Adr 0A00) gesichert werden. Der Rechner muß dann die aktuelle Position durch Abfrage des A/D-Wandlers ermitteln und mit der Sollposition **vergleichen**. Der Motor muß so lange angesteuert werden, bis der Vergleich die **Übereinstimmung** der beiden Werte ergibt. Aus dem Gesagten ist zu entnehemen, daß bei dieser Aufgabe das **ganze Bit-muster** von Bedeutung ist. Die Testbefehle für die Abfrage einzelner Bits in einem Binärwert - die TMI-Befehle - können hier nicht mehr verwendet werden. Wir benötigen besondere **COMPARE-Befehle** (engl. to compare = vergleichen).

Beim **COMZ-Befehl** wird der Inhalt eines der Vielzweckregister R1 - R3 mit dem Inhalt von Register 0 verglichen. Der Zwei-Byte- Befehl **COMI** vergleicht den Inhalt eines beliebigen Registers mit dem Bitmuster, das als **Maske** im zweiten Befehls-byte steht. Bei den Befehlen **COMR** und **COMA** werden die Register-inhalte mit dem Inhalt der Speicherstellen, die durch relative oder absolute Adressierung zu finden sind, verglichen.

Zu beachten ist die Meldung der Ergebnisse des Vergleichs. Grundsätzlich werden hierfür wiederum die Bit 6 und 7 des unteren Programm-Status-Wortes verwendet, die in der Sprache des Prozessors CONDITION CODE (Bedingungscode) genannt werden. Wenn der Inhalt des zu prüfenden Registers **größer** ist als das Bitmuster im Vergleichsregister, der Wert der Maske oder der Inhalt der angesprochenen Adresse, so wird der Bedingungscode auf **01** gesetzt. Sind beide Bitmuster **gleich**, erscheint dort **00** und wenn das zu prüfende Register **kleiner** ist, wird dort **10** eingesetzt.
Eine wichtige Besonderheit besteht darin, daß durch Setzen oder Löschen des Bit Nr.1 im unteren PSW (COM) noch fest-gelegt werden kann, ob der Prozessor beim Vergleich die Zahlen als reine Binärzahlen anzusehen hat oder aber sie als Zahlen im Zweier-Komplementsystem behandeln soll. Im ersten Fall würde z.B. die Binärzahl **10000000** größer sein als die Zahl **01000000**. Dagegen wäre als Zweier-Komplement betrachtet die erstgenannte Zahl, die einen negativen Wert darstellt, kleiner als die zweitgenannte.
In unserem Fall sind die Binärwerte als reine Binärzahlen (logischer Zahlenbereich) aufzufassen. Wir müssen deshalb Bit 1 im unteren PSW auf 1 setzen. Mit dem Zwei-Byte-Befehl PPSL,COM - Opcode 77 02 (hex) - läßt sich das PSW am ein-fachsten initialisieren. So manche Fehlfunktion eines Steuerungsprogrammes, das COMPARE-Befehle enthält, hat ihre Ursache in der Nichtbeachtung des COM-Bits im unteren PSW!

Der durch die Vergleichsbefehle veränderte Bedingungscode liefert in den folgenden Programmen wieder die Bedingungen, für bedingte Sprungbefehle (z.B. BCFA, BCTA, BCFR und BCTR).

Die genaue Befehlsbeschreibung der COMPARE-Befehle findet man auf den Seiten 327 - 329 im Begleitbuch. Hier nur eine kurze Übersicht über die ersten Befehlsbytes:

Befehl	R0	R1	R2	R3			
COMZ	E0	E1	E2	E3	(R) = Maske	CC = 00	= EQ (equal)
COMI	E4	E5	E6	E7	(R) > Maske	CC = 01	= GT (greater)
COMR	E8	E9	EA	EB	(R) < Maske	CC = 10	= LT (less than)
COMA	EC	ED	EE	EF			

PROGRAMM: Steuerung eines Drehkranzes. Positionserkennung und A/D-Wandlung.

LABEL	ADR.	DATEN	BEFEHL	KOMMENTAR
	0A00	XX	Sollposition	
Init.	0900	77 02	PPSL,COM	log. Vergleich
Start	0902	54 08	REDE,R0 Port 8	Starttaste?
	0904	F4 01	TMI, R0 H01	ja, Position anfahren
	0906	1E 09 14	BCTA,2 '0914'	nein, zur Startpos.
Links	0909	05 03	LODI,R1 H03	Linkslauf laden und
	090B	D5 09	WRTE,R1 Port 9	ausgeben, Starttaster
Abfr.	090D	54 08	REDE,R0 Port 8	abfragen bis gedrückt
	090F	F4 01	TMI, R0 H01	
	0911	1C 09 0D	BCTA,0 '090D'	zur Abfrage springen
Rechts	0914	05 01	LODI,R1 H01	Rechtslauf laden und
	0916	D5 09	WRTE,R1 Port 9	ausgeben
A/D	0918	54 0A	REDE,R0 Port A	A/D-Wandler lesen
	091A	EC 0A 00	COMA,R0 '0A00'	Wert mit Sollwert
	091D	1D 09 23	BCTA,GT 'Halt'	vergleichen, größer?
	0920	9C 09 18	BCFA,EQ '0918'	ungleich? zu A/D, sonst
Halt	0923	05 00	LODI,R1 H00	Motor stoppen und HALT
	0925	D5 09	WRTE,R1 Port 9	
	0927	40	Halt	

Die beiden Sprungbefehle an Adresse 091D bzw 0920 müssen näher erläutert werden.
Wenn die Sollposition überfahren wurde, liefert der Vergleichsbefehl in der Adresse 091A die Meldung, daß der Wert in R0 **größer** ist als in der in 0A00, den wir vorgegeben hatten. In den Bedingungscode wird also **01** geschrieben. Der jetzt folgende bedingte Sprungbefehl wird ausgeführt und der Motor kann gestoppt werden. Bei allen anderen Werten des Bedingungscodes ist der aktuelle Positionswert kleiner oder gleich dem vorgegebenen. Der BCFA-Befehl führt so lange zur Positionsabfrage - der Motor läuft weiter - bis die Bedingung **gleich** (CC = 00) erfüllt ist.
BCTA (Branch on condition true) - "Springe, wenn die Bedingung zutrifft."
BCFA (Branch on condition false) "Springe, wenn die Bedingung **nicht** zutrifft."