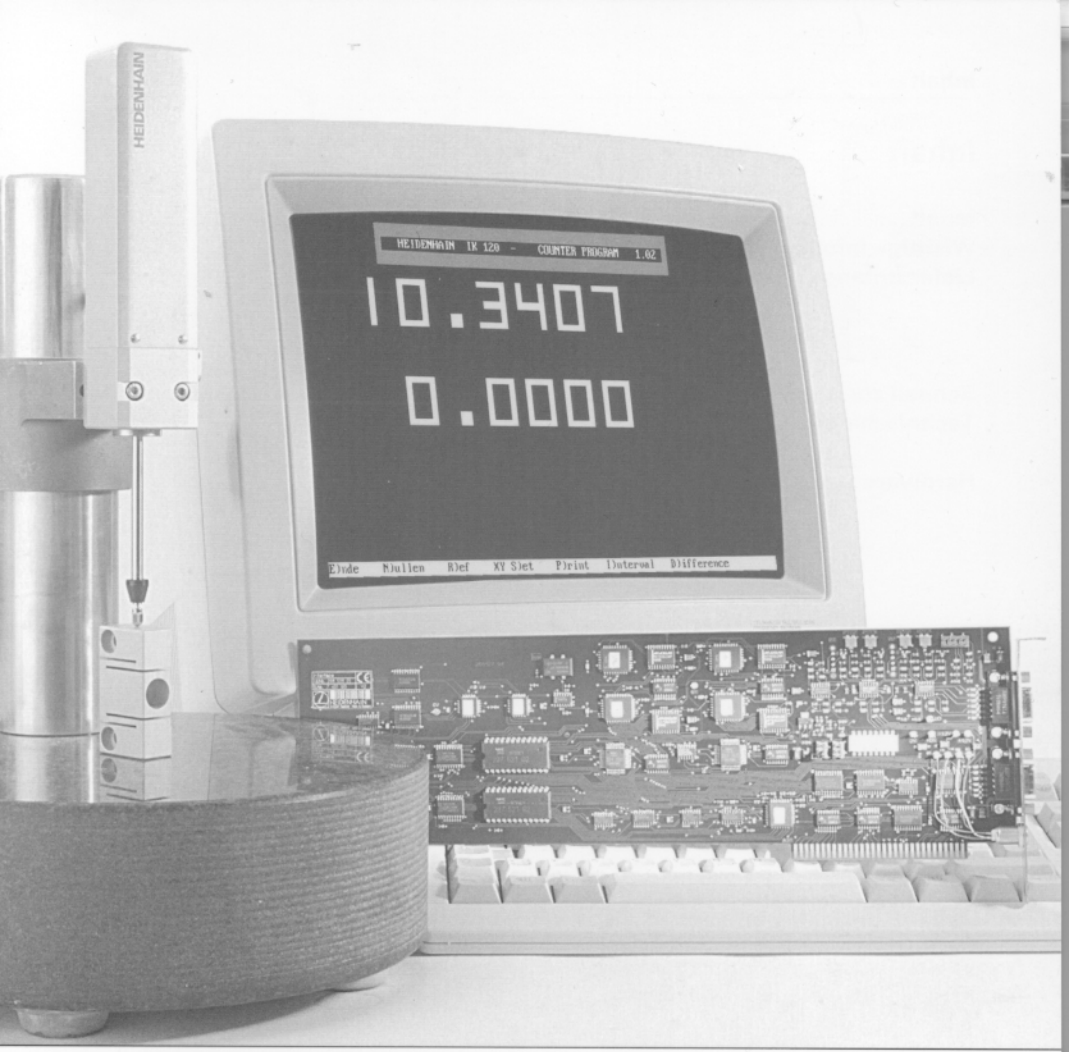


HEIDENHAIN

Benutzer-Handbuch
IK 120
PC-Zählerkarte

5/94

Inhalt

Wichtige Information für Benutzer mit 386er und 486er PC

QEMM.SYS, EMM386.SYS

Haben Sie einen PC mit 80386 oder 80486 Prozessor? Dann benutzen Sie vermutlich eines der oben genannten Speicherverwaltungs-Programme. Diese Speicherverwaltungs-Programme und ihre IK dürfen **nicht** den gleichen Speicherbereich benutzen.

Ändern Sie deshalb die Datei "**CONFIG.SYS**" beispielsweise wie folgt:

DEVICE=C:\DOS\EMM386.SYS X=C800-C840
oder
DEVICE=C:\QEMM\QEMM386.SYS X=C800-C840

Lieferumfang

Id.-Nr.	
271 208 ..	PC-Zählerkarte IK 120, Programmierbeispiele, Treiber-Software und Benutzer-Handbuch.

Hinweis zur EMV-Richtlinie 89/336/EWG

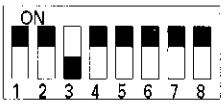
Die Bestimmungen der EMV-Richtlinie 89/336/EWG wurden mit dem COMPAQ-Rechner DESKPRO 386/20e geprüft.

Zubehör

Id.-Nr.	
309 785 ..	Adapterkabel für HEIDENHAIN-Meßsysteme; Standardlänge: 5 m; andere Längen auf Anfrage
310 195 ..	Adapterkabel für HEIDENHAIN-Meßsysteme mit APE (z.B. VM 101)
257 818 01	Zusätzlicher Sub-D-Anschluß zur Weiterführung der Meßsystemsignale des Eingangs X2 an eine weitere Anzeige oder Steuerung
309 781 ..	Adapterkabel vom zusätzlichen Sub-D-Anschluß an eine weitere Anzeige oder Steuerung
282 168 01	Stecker für die externen Funktionen am Anschluß X3 (zwei Buchsen, zwei Stifte)

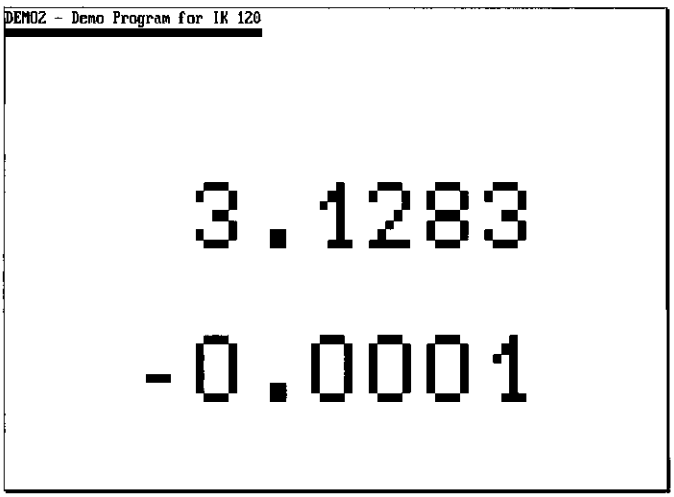
Gefahr für interne Bauteile
Die Vorsichtsmaßnahmen bei der Handhabung **elektrostatisch entladungsgefährdeter Bauelemente (ESD)** nach DIN/EN 100 015 beachten. Als Transport-Verpackung nur antistatisches Material verwenden. Beim Einbau ausreichende Erdung des Arbeitsplatzes und der Person sicherstellen.

Schnell zur ersten Anzeige



- ➤ DIP-Schalter wie folgt einstellen:
- ➤ Die Steckbrücke zum Festlegen des Interrupt-Verhaltens der IK 120 – neben den DIP-Schaltern – entfernen
- ➤ Netzstecker des PCs abstecken; PC-Gehäuse öffnen
- ➤ IK 120 in den PC einstecken und befestigen; PC-Gehäuse montieren
- ➤ Längenmeßsystem mit sinusförmigen Ausgangs-Signalen über HEIDENHAIN-Adapterkabel (Id.-Nr. 309 785 ..) an den Eingang "X1" anschließen.
- ➤ Die Diskette mit der mitgelieferten Treiber-Software in das Laufwerk A: einlegen
- ➤ Folgende DOS-Befehle eingeben:

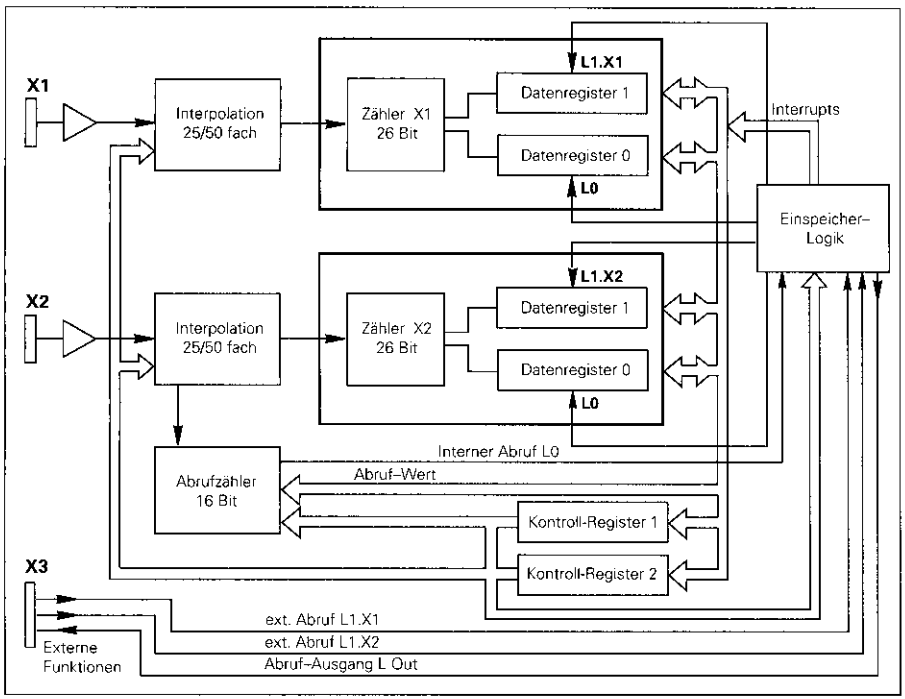
```
>A:
>CD\TP
>DEMO2
```



Falls die Meßsystembewegungen nicht am Bildschirm angezeigt werden, siehe "Was ist zu tun, wenn's nicht funktioniert?".

Technische Beschreibung der IK 120

An die PC-Zählerkarte IK 120 können zwei HEIDENHAIN-Meßsysteme mit sinusförmigen Ausgangs-Signalen angeschlossen werden. Sie wird direkt in einen Erweiterungs-Steckplatz eines XT- oder AT-kompatiblen Personal Computers gesteckt.
Die Positionen der beiden Meßsysteme werden mittels Software am PC angezeigt, im PC gespeichert und weiterverarbeitet.



Die Interpolations-Elektronik in der IK 120 unterteilt die Signalperiode des Eingangs-Signals bis zu 200fach.

Zusätzlich zu den zwei Zählern für die angeschlossenen Meßsysteme besitzt die IK 120 einen Abruf-Zähler. Dieser ermöglicht den Meßwert-Abruf in programmierbaren Weg-Intervallen.

Der Meßwert-Abruf kann auch über ein externes Schaltsignal erfolgen.

Die Begriffe "Abruf" oder "abrufen" in diesem Handbuch bedeuten, daß der Zählerwert im Daten-Register 0 oder Daten-Register 1 festgehalten wird. Dieser Zählerwert muß anschließend per Software gelesen und im PC gespeichert oder am Bildschirm angezeigt werden.

Zugriffszeit auf Meßwerte

Tests auf einem PC mit 80286-Prozessor haben gezeigt, daß mit optimierten Assembler-Routinen die Zugriffszeit zwischen zwei Meßwert-Erfassungen unter **20 µs** liegt.

Hardware

Spezifikation des PC-Bus

Die IK 120 kann in alle IBM XT, AT und 100%-IBM-kompatiblen PCs eingesetzt werden. HEIDENHAIN garantiert nicht für die einwandfreie Funktion der IK 120 mit nicht 100%-kompatiblen PCs. Die IK 120 entspricht der internationalen Norm IEEE P996 , die den XT-, AT- und ISA-Bus spezifiziert (Industrie-Standard).

Busbreite	8 Bit
Spannungsversorgung	+5 V ±5% +12 V ±5% –12 V ±5%
Leistungsaufnahme	typisch 1 Watt maximal 1,5 Watt (ohne Meßsysteme)
Slot-Ausführung	XT-kompatibel, volle Länge, 8-Bit-Slot
Speicherbedarf	1024 Bytes
Adreßbereich	C0000h bis FFFFFh

Meßsystem-Eingänge

An die IK 120 werden HEIDENHAIN-Längenmeßsysteme oder -Winkelmeßsysteme mit sinusförmigen Inkrementalsignalen I_1 und I_2, die um 90° el. zueinander phasenverschoben sind, angeschlossen. Zusätzlich steht das Referenzmarken-Signal I_0 zur Verfügung.

Signalamplituden I_1, I_2 (0°, 90°) I_0 (Referenzmarke)	 7 μA_{SS} bis 16 μA_{SS} 3,5 µA bis 8 µA
Maximale Eingangsfrequenz	Interpolation 25fach: 50 kHz Interpolation 50fach: 30 kHz
Kabellänge	max. 10 m
Signalpegel für Fehlermeldung	≤ 2,5 µA

Meßsignal-Unterteilung
Die Interpolations-Elektronik der IK 120 erzeugt aus einer Periode der Eingangs-Signale 25 oder 50 TTL-Signalperioden, die einem Zähler zugeführt werden. Der Zähler ist programmierbar und kann wahlweise eine, zwei oder vier Flanken der TTL-Signale auswerten. Das bedeutet, daß pro Periode des Eingangs-Signals bis zu 200 Impulse erzeugt werden können.

Interpolation	Auswertung	Unterteilung
25fach	1	25
	2	50
	4	100
50fach	1	50
	2	100
	4	200

Anschluß X1,X2 für Meßsysteme
Sub-D-Anschluß mit Buchseneinsatz (9polig)

Anschluß-Nr.	Belegung
1	I_1 –
2	0 V (U_N)
3	I_2 –
4	Innenschirm
5	I_0 –
6	I_1 +
7	5 V (U_P)
8	I_2 +
9	I_0 +
Gehäuse	Außenschirm

Meßsystem-Ausgang

Die Meßsystem-Signale des Eingangs X2 stehen auf der Platine der IK 120 an einem 10poligen AMP-Stecker als Ausgangs-Signale zur Verfügung. Über eine zusätzliche Kabelbaugruppe mit PLC-Slot-Abdeckung (Id.-Nr. 257 818 01) kann dieser Anschluß nach außen zu einem 9poligen Sub-D-Anschluß geführt werden. Ein Adapterkabel (Id.-Nr. 309 781 ..) zum Anschluß an HEIDENHAIN-Positionsanzeigen oder Interpolations-Elektroniken ist lieferbar.
Kabellänge: abhängig von der Eingangsschaltung der Folge-Elektronik.

Meßsystem-Ausgang (Option-Nr. 257 818 01)
Sub-D-Anschluß mit Stifteinsatz (9polig)

Anschluß-Nr.	Belegung
1	I_1 –
2	0 V (U_N)
3	I_2 –
4	nicht angeschlossen
5	I_0 –
6	I_1 +
7	nicht angeschlossen
8	I_2 +
9	I_0 +
Gehäuse	Außenschirm

Platinenstecker für Meßsystem-Ausgang
AMP mit Stifteinsatz (10polig)

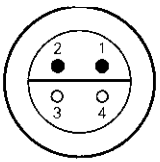
Anschluß-Nr.*	Signal
1a	nicht angeschlossen
1b	nicht angeschlossen
2a	nicht angeschlossen
2b	0 V (U_N)
3a	I_0 –
3b	I_0 +
4a	I_2 –
4b	I_2 +
5a	I_1 –
5b	I_1 +

*Die Seite mit den Verriegelungs-Stiften ist mit b bezeichnet.
Anschlüsse 1a und 1 b befinden sich auf der Seite mit der Einkerbung.

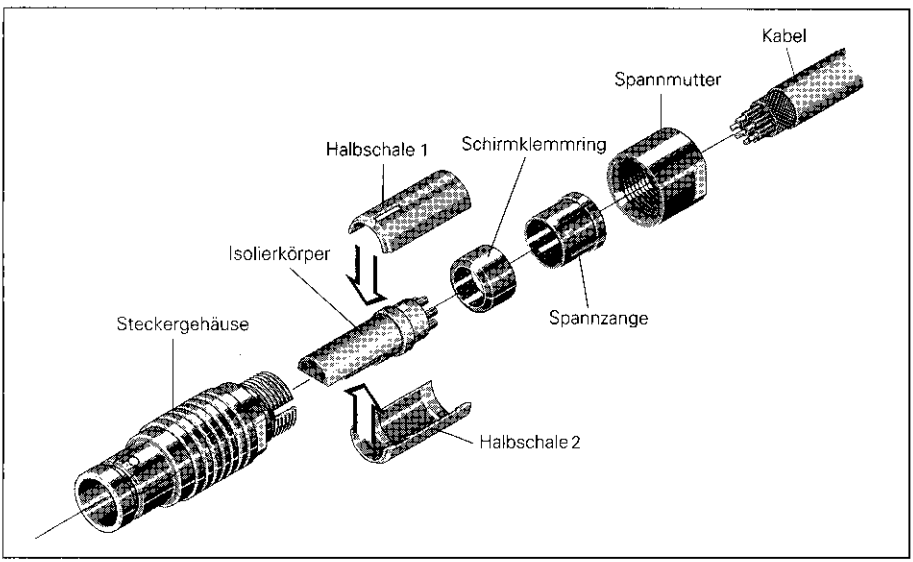
Externe Funktionen

Für externe Funktionen ist eine 4polige Flanschdose vorgesehen. Der dafür benötigte Stecker (Id.-Nr. 282 168 01) kann bei HEIDENHAIN bestellt werden.

Montage des Steckers für die externen Funktionen



Rückansicht
o = Buchse, • = Stift



Anschluß X3 für Externe Funktionen

Flanschdose mit Stift/Buchseneinsatz (4polig)

Anschluß-Nr.	Belegung
1	Eingang: Meßwert-Abruf X1 (L1.X1)
2	Eingang: Meßwert-Abruf X2 (L1.X2)
3	Ausgang: Meßwert-Abruf (LOut)
4	0 V

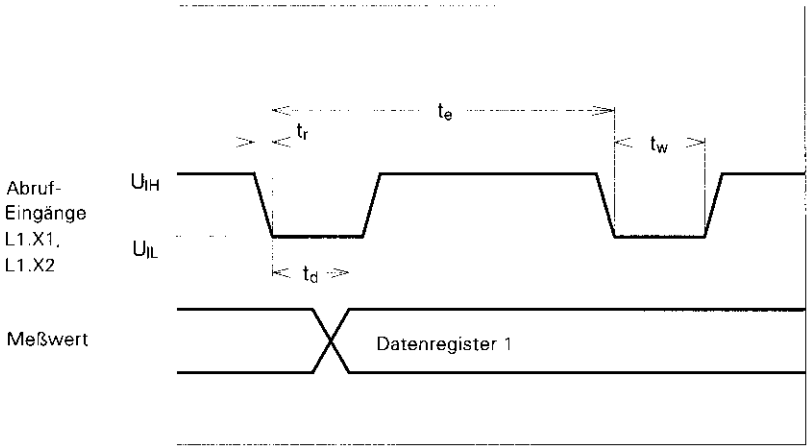
Abruf der Meßwerte über externe Eingänge
Die IK 120 hat zwei externe Eingänge an der Flanschdose X3 zum Abrufen und Speichern der Meßwerte im Daten-Register 1.

Über diese Eingänge können auch Interrupts ausgelöst werden.

Die Eingänge L1.X1 und L1.X2 sind low-aktiv; ein interner Pull-up-Widerstand hält sie auf High-Pegel. Sie können an TTL-, LS- oder CMOS-Bausteine angeschlossen werden.

Die einfachste Art, die Eingänge zu aktivieren: Eine Brücke von 0 Volt (Anschluß 4) auf den Eingang zum Abrufen.

Meßwert-Abruf über L1.X1, L2.X2: Zeitablauf und Spannungspegel



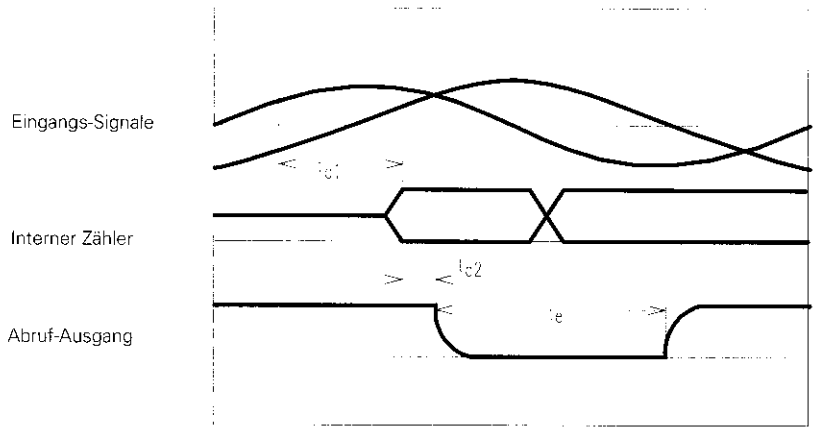
Signale / Zeit	Minimum	Maximum
U_{IH} (V)	3,15	30
U_{IL} (V)	–3,0	0,9
t_r (µs)		0,5
t_d (µs)*		0,6
t_e (µs)	40	
t_w (µs)	20	

*bei U_{IH} = 5 V und U_{IL} = 0 V

Abruf-Ausgang (LOut)
Das Ausgangs-Signal LOut wird vom internen Abruf-Zähler erzeugt und kann über die Flanschdose X3 zu weiteren IK 120 geleitet werden (Eingänge L1.X1, L1.X2), um beispielsweise die Meßwerte von verschiedenen IKs abzurufen.

LOut ist ein Open-Collector-Ausgang, der auf "Null" schaltet, wenn das interne Abruf-Signal L0 anliegt.

Abruf-Ausgang (LOut): Zeitablauf und Spannungspegel



Signale / Zeit	Minimum	Maximum
U_{OH} (V)	4,0	32
U_{OL} (V)	0	0,8 mit U_{OH}<12 1,0 mit U_{OH}>12
I_{OL} (mA)		40
t_{d1} (µs)	3,3	12
t_{d2} (ns)		200
t_e (µs)	40	

Abruf-Zähler

Die IK 120 hat einen dritten Zähler, den sogenannten Abruf-Zähler; er kann in bestimmten frei programmierbaren Abständen ein Abruf-Signal für die Zähler X1 und X2 erzeugen – beispielsweise alle 1000 Zählimpulse oder alle 0,1mm. Dieses Signal kann über den Abruf-Ausgang am Stecker X3 ausgegeben werden.

Datenbreite des Abruf-Zählers	16 Bit
Abruf-Wert	16 Bit; von 2 bis 65535
Meßsystem-Signal	nur von Eingang X2
Funktionen in Verbindung mit der Referenzmarke	Start, Stop, Rücksetzen, Laden, Abrufen
Auswertung der Meßsystem-Signale	1fach, 2fach, 4fach

Interrupts

Die IK 120 kann einen der folgenden PC-Interrupts nutzen: IRQ3, IRQ4, IRQ5, IRQ7.

Der gewünschte Interrupt wird über zwei 6polige Stecker auf der Platine – neben den DIP-Schaltern – festgelegt: durch Stecken einer Brücke.

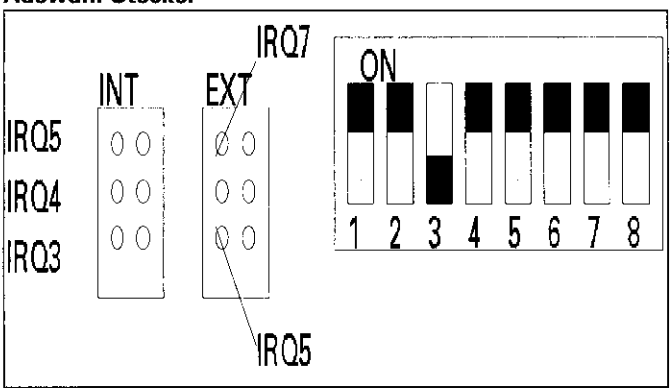
Das interne Abruf-Signal des Abruf-Zählers L0 kann auf IRQ3, IRQ4 und IRQ5 geleitet werden (mit Hilfe der Steckbrücke INT, siehe nachfolgende Zeichnung). Beim Abrufen des Meßwertes wird das **Int0**-Flip-Flop gesetzt, das der Rechner wieder rücksetzen muß.

Die externen Abruf-Signale L1.X1 und L1.X2 können auf IRQ5 und IRQ7 geleitet werden (mit Hilfe der Steckbrücke EXT, siehe nachfolgende Zeichnung). Beim Abrufen des Meßwertes wird das **Int1**-Flip-Flop gesetzt, das der Rechner wieder rücksetzen muß.

Belegung der PC-Interrupts

Interrupt	Interrupt-Nummer	Interrupt-Adresse	Normalerweise zugeordnet zu
IRQ3	0Bh	002Ch	COM2
IRQ4	0Ch	0030h	COM1
IRQ5	0Dh	0034h	LPT2
IRQ7	0Fh	003Ch	LPT1

Zuordnung der Interrupts zu den Anschlüssen der beiden Auswahl-Stecker



Es kann nur ein Interrupt – entweder intern oder extern – durch Stecken der Brücke gewählt werden. Den ausgewählten Interrupt darf keine weitere Karte im PC nutzen.

Hardware-Änderung ab Zählerkarten-Id.-Nr. 298 538 01
Die IK 120 kann einen Interrupt über das externe Latch-Eingangs-Signal L1.X1/L1.X2 generieren. Während die IK 120, Id.-Nr. 268 136 01, den Interrupt wiederholte, solange der externe Kontakt geschlossen war, generiert die IK 120, neue Hardware-Nr. 298 538 01, nur einen Interrupt – unabhängig von der Zeit, die der Kontakt geschlossen ist.

Adreßfestlegungen

Die 20-Bit-Adresse eines PCs wird aus zwei Komponenten gebildet: aus einer 16-Bit-Segment-Adresse (Abschnitts-Adresse) und aus einer 16-Bit-Offset-Adresse (Unteradresse, relative Adresse). Bei der Berechnung einer Adresse wird der Segment-Wert um vier Bits nach links geschoben (also um eine HEX-Stelle) und der Offset-Wert addiert. Eine Segment-Adresse und eine Offset-Adresse bilden zusammen eine sogenannte segmentierte Adresse (auch physikalische Adresse genannt).

Die Kommunikation zwischen IK 120 und Rechner erfolgt über RAM-Adressen mit 8-Bit-Datenbreite. Der Adreßbereich liegt im hohen Speicherbereich (über 640 Kbyte) zwischen den Adressen C0000h und FFFFFh. Dabei belegt die IK 120 1024 Bytes.

Die IK 120 wird mit der Segment-Adresse C800h ausgeliefert. Das bedeutet, daß der Adreßbereich zwischen C8000 und C83FF belegt ist.

Beispiel:
Segment-Adresse: C800
Offset-Adresse: 000A
Meist schreibt man kurz: C800:000A

Die segmentierte Adresse wird wie folgt gebildet:

```
  C8000
+ 000A
  C800A
```

DIP-Schalter für die Adressen einstellen

Die Segment-Adresse wird mit den DIP-Schaltern auf der IK 120 eingestellt.

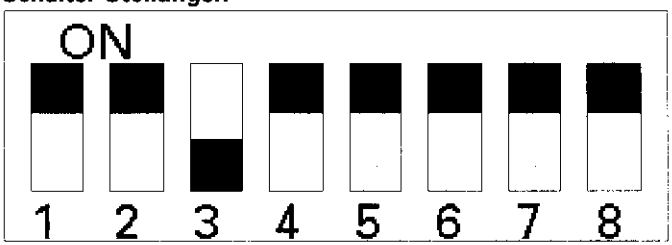
Die Adreßleitungen für die 20 Bit breite Adresse des PCs werden in diesem Handbuch mit A0 bis A19 bezeichnet.

Auf der IK sind die Adreßleitungen A19 und A18 fest auf "1" gelegt. Die Adreßleitungen A17 bis A10 sind über die DIP-Schalter S1 bis S8 einstellbar **(0=ON, 1=OFF)**.

Die nachfolgende Tabelle zeigt, wie die DIP-Schalter einzustellen sind:

Adreß-Bit	A19 A18	A17 A16	A15 A14 A13 A12	A11 A10 A9 A8
Hexadezimal	C		8	0
Binär	1 1	0 0	1 0 0 0	0 0 0 0
Schalter	fest	S1 S2	S3 S4 S5 S6	S7 S8
Schalter-Stellung		ON ON	**OFF** ON ON ON	ON ON

Schalter-Stellungen



Beispiele für Adreßeinstellungen

Segment-Adresse (Hex)	**Schalter-Stellung**							
	S1	S2	S3	S4	S5	S6	S7	S8
C8000	ON	ON	OFF	ON	ON	ON	ON	ON
C8400	ON	ON	OFF	ON	ON	ON	ON	OFF
C8800	ON	ON	OFF	ON	ON	ON	OFF	ON
CF000	ON	ON	OFF	OFF	OFF	OFF	ON	ON
D0000	ON	OFF	ON	ON	ON	ON	ON	ON
E0000	OFF	ON	ON	ON	ON	ON	ON	ON

Überprüfen Sie, ob die eingestellte Adresse nicht von einer weiteren Karte in Ihrem PC benutzt wird, da mehrere Karten nicht die gleichen Adressen benutzen dürfen.

Die Adressen von C0000h bis C6000h sind nomalerweise vom EGA-oder VGA-Grafik-Adapter belegt.

Adapterkarten (SCSI) für zusätzliche externe Speicher wie eine zusätzliche Harddisk oder optische Speicher ("GIGAFILE") verwenden oft den gleichen Speicherbereich wie die IK 120: C8000h.

Benutzen Sie nicht den Adreßbereich F0000h bis FFFFFh für die IK 120, da dieser Bereich normalerweise vom BIOS ROM belegt ist.

Adreßbelegung für den Einsatz von mehreren IK 120
Beim Einsatz von mehreren IK 120 in einem PC müssen die Karten auf verschiedene Adreßbereiche eingestellt werden.

Zählerkarte	Segment-Adresse	Adreßbereich	S1 bis S8
Erste IK 120	C8000h	C8000h bis C83FFh	0 0 1 0 0 0 0 0
Zweite IK 120	C8400h	C8400h bis C87FFh	0 0 1 0 0 0 0 1
Dritte IK 120	C8800h	C8800h bis C8BFFh	0 0 1 0 0 0 1 0
.....			

Register

In der folgenden Beschreibung werden die Offset-Adressen der Register angegeben.

Register-Übersicht

Den Adreßbereich 00h bis 0Fh belegt der Zählerbaustein für den Eingang X1 und den Adreßbereich 10h bis 1Fh der Zählerbaustein für den Eingang X2. Zwischen 20h und 70h befinden sich weitere Register zur Festlegung von Parametern, zur Steuerung des Abruf-Zählers und zum Rücksetzen eines Interrupts.

Adresse (Hex)	Register
00 bis 0F	**Zählerbaustein für Eingang X1**
00	Daten-Register 0, LS-Byte
01	Daten-Register 0
02	Daten-Register 0
03	Daten-Register 0, MS-Byte
04	Daten-Register 1, LS-Byte
05	Daten-Register 1
06	Daten-Register 1
07	Daten-Register 1, MS-Byte
08	Meßwert-Abruf für Daten-Register 0
09	Meßwert-Abruf für Daten-Register 1
0A	Referenzmarken-Register
0B	Befehlsregister für Zähler
0C	Betriebsarten-Register
0D	Löschen des Status-Registers
0E	Status-Register
0F	Initialisierungs-Register (Schreibzugriff)
0F	Zustands-Register (Lesezugriff)
10 bis 1F	**Zählerbaustein für Eingang X2** wie oben von 10 bis 1F
20	**Kontroll-Register 1**
30	**Kontroll-Register 2**
40	**Abruf-Wert LS-Byte**
50	**Abruf-Wert MS-Byte**
60	**Strobe-Register für Abrufzähler**
70	**Interrupt rücksetzen**

In der folgenden Register-Beschreibung werden aus Gründen der Übersichtlichkeit grundsätzlich nur die Adressen für den Zählerbaustein X1 (00h bis 0Fh) angegeben. Natürlich gelten die Beschreibungen auch für den Zählerbaustein X2 (10h bis 1Fh).

Daten-Register für die Zähler

Die Meßwerte werden in 26 Bit breiten Zählern erfaßt und in 32 Bit breiten Daten-Registern gespeichert. Der Meßwert benötigt nur 26 von den 32 Bits im Daten-Register. Mit den restlichen 6 Bits wird das Zweierkomplement für negative Zahlen dargestellt. Die Zähler haben also einen Werte-bereich von -2^{25} bis $+2^{25} - 1$ oder bei Winkel-Zählweise –180 000 Grad bis +179 999 Grad.

00h-03h: Daten-Register 0
Bei einem Meßwert-Abruf über das **Abruf-Register 08h** oder über den **Abruf-Zähler** wird der Meßwert im **Daten-Register 0** gespeichert.

04h-07h: Daten-Register 1
Bei einem Meßwert-Abruf über das **Abruf-Register 09h** oder bei einem **externen Abruf-Signal** wird der Meßwert im **Daten-Register 1** gespeichert.

Wenn ein Meßwert in einem Daten-Register gespeichert wurde, dann sind keine weiteren Abrufe mehr möglich, bis das obere Byte (03h oder 07h) des betreffenden Daten-Registers gelesen wurde. Nach dem Lesen des Meßwertes wird im Status-Register 0Eh das Bit D1 oder D2 rückgesetzt.

08h: Abruf-Register 0 für Software-Abruf
Ein Schreibzugriff auf dieses Register (08h für Eingang X1 und 18h für Eingang X2) speichert den Meßwert in Daten-Register 0.

09h: Abruf-Register 1 für Software-Abruf
Ein Schreibzugriff auf dieses Register (09h für Eingang X1 oder 19h für Eingang X2) speichert den Meßwert in Daten-Register 1.

0Ah: Referenzmarken-Register
Dieses Register legt Funktionen fest, die beim Überfahren der Referenzmarken ausgelöst werden. Beispielsweise kann ein Zähler gestartet, gestoppt oder genullt werden.
Wenn das Überfahren einer Referenzmarke einen Zählerwert abrufen soll, muß vorher über das Initialisierungs-Register 0Fh im Bit D4 festgelegt werden, in welchem Daten-Register der Wert zu speichern ist.

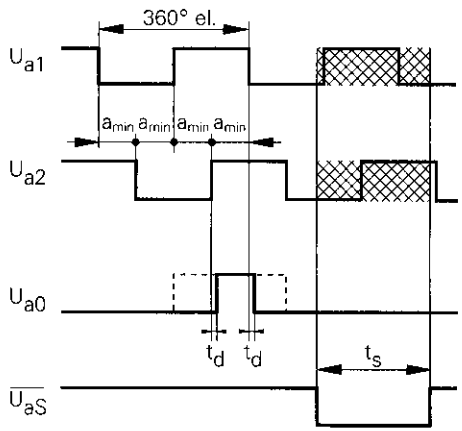
Wert (Hex)	Funktion beim Überfahren der Referenzmarke
00	keine Funktion
01	Zähler starten
02	Zähler stoppen
03	keine Funktion
04	Zähler nullen
05	Zähler nullen und starten
06	Zähler nullen und stoppen
07	Zähler nullen
08	Zählerwert abrufen
09	Zählerwert abrufen und Zähler starten
0A	Zählerwert abrufen und Zähler stoppen
0B	Zählerwert abrufen und im Daten-Register speichern
0C	Zählerwert abrufen und Zähler nullen
0D	Zählerwert abrufen, Zähler nullen und starten
0E	Zählerwert abrufen, Zähler nullen und stoppen
0F	Zählerwert abrufen und Zähler nullen

0Bh: Befehls-Register für die Zählerbausteine
Dieses Register startet, stoppt oder nullt die Zähler per Software.

Wert (Hex)	Funktion
01	Zähler starten
02	Zähler stoppen
04	Zähler nullen
05	Zähler nullen und starten
06	Zähler nullen und stoppen
07	Zähler nullen

0Ch: Betriebsarten-Register für die Zählerbausteine
Dieses Register legt die folgenden Funktionen fest:

Flankenauswertung
Durch die beiden inkrementalen Meßsystemsignale (0° el. und 90° el.) stehen pro Signalperiode des 25fach oder 50fach unterteilten Signals maximal vier Flanken zur Auswertung zur Verfügung. Die Zähler können so programmiert werden, daß sie entweder eine, zwei oder vier Flanken pro Signalperiode zählen.



Zählrichtung
Die Zählrichtung legt fest, ob die Zähler bei positiver Verfahrrichtung positiv (normal) oder negativ (invers) zählen.

Zählweise
Es können zwei Zählweisen festgelegt werden:
Linear-Zählweise: -2^{25} bis $+2^{25}-1$
Winkel-Zählweise: –180 000 bis +179 999

Wert (Hex)	Auswertung	Zählrichtung	Zählweise
00	1fach	normal	Linear
02	2fach	"	"
03	4fach	"	"
04	1fach	"	Winkel
05	2fach	"	"
07	4fach	"	"
08	1fach	invers	Linear
09	2fach	"	"
0B	4fach	"	"
0C	1fach	"	Winkel
0D	2fach	"	"
0F	4fach	"	"

0Dh: Löschen des Status-Registers
Schreiben des Wertes FFh in das Register 0Dh löscht das Status-Register.

0Eh: Status-Register (Lesezugriff)
Das Status-Register erlaubt nur Lesezugriffe. Es darf nicht beschrieben werden!

Bit	Bedeutung
D0	1 = Meßwert im Daten-Register 0 gespeichert 0 = Oberes Byte wurde aus dem Daten-Register 0 gelesen
D1	1 = Meßwert im Daten-Register 1 gespeichert 0 =Oberes Byte wurde aus dem Daten-Register 1 gelesen
D2	0 = Zähler gestoppt 1 = Zähler gestartet
D3	Meßsystemfehler 1 = Ein Meßsystemfehler wurde erkannt (Amplitude zu klein oder Eingangsfrequenz überschritten) 0 = Kein Meßsystemfehler Der Meßsystemfehler wird wie folgt rückgesetzt: Schreiben der Sequenz 1 - 0 - 1 in das Kontroll-Register – Bit D2 für Achse X1 und Bit D3 für Achse X2. Anschließend das Status-Register löschen – durch Schreiben von FFh in das Register 0Dh
D4	0
D5	1 = Auf das Referenzmarken-Register erfolgte ein Schreibzugriff Das Bit D5 wird durch Überfahren der Referenzmarke wieder rückgesetzt
D6	Logik-Pegel des 0°-Signals zum Zeitpunkt des letzten internen Abrufs
D7	Logik-Pegel des 0°-Signals zum Zeitpunkt des letzten externen Abrufs

0Fh: Initialisierungs-Register(Schreibzugriff)
Dieses Register legt den internen und externen Meßwert-Abruf fest sowie den Meßwert-Abruf über Referenzmarken.

Bit	Bedeutung
D0	0
D1	0
D2	0 = Interner Abruf über Abruf-Zähler aktiv (L0) 1 = Interner Abruf über Abruf-Zähler nicht aktiv (L0)
D3	0 = Externer Abruf aktiv (L1) 1 = Externer Abruf nicht aktiv (L1)
D4	0 = Abruf über Referenzmarke speichert in Daten-Register 0 1 = Abruf über Referenzmarke speichert in Daten-Register 1
D5	0 = Referenzmarke triggert auf die Flanke 1 = Referenzmarke triggert auf den Pegel
D6	0
D7	beliebig

0Fh: Zustands-Register (Lesezugriff)
Dieses Register enthält die Zustände verschiedener Anschlüsse des Zählerbausteins.

Bit	Bedeutung
D0	Interner Abruf (L0) ist erfolgt
D1	Externer Abruf (L1) ist erfolgt
D2	0 = Internes Abruf-Signal ist freigegeben (L0) 1 = Internes Abruf-Signal ist gesperrt (L0)
D3	0 = Externes Abruf-Signal ist freigegeben (L1) 1 = Externes Abruf-Signal ist gesperrt (L1)
D4	0 = Referenzmarke speichert im Daten-Register 0 1 = Referenzmarke speichert im Daten-Register 0
D5	Logik-Pegel der Referenzmarke
D6	Logik-Pegel des 0°-Signals
D7	Logik-Pegel des 90°-Signals

20h: Kontroll-Register 1
Die verschiedenen Bits von Kontroll-Register 1 haben folgende Funktion:

Bit	Funktion	Eingabewert
D0 D1	Interpolation, Eigang X1 Interpolation, Eingang X2	0 = 25fach, 1 = 50fach
D2 D3	Reset: Meßsystemfehler X1* Meßsystemfehler X2*	Ein Meßsystemfehler setzt Bit D3 im Status-Register 0Eh. Fehlermeldung löschen: – Bit D2 oder D3 in Kontroll-Register 20h mit 1 – 0 – 1 beschreiben. – zusätzlich Status-Register 0Dh durch Beschreiben mit FFh löschen
D4 D5	Signal-Auswertung für den Abrufzähler	00(D4, D5) = 4fach 10 = 2fach 11 = 1fach 01 = nicht verwenden
D6 D7	Freigabe: Interrupt Int0 Interrupt Int1	0 = gesperrt 1 = freigegeben

30h: Kontroll-Register 2

Funktion	Hex	Bits	Bedeutung
Meßwert-Abruf		**D3 D2 D1 D0**	*
	00	0 0 0 0	Abruf-Signale nicht freigegeben
	01	0 0 0 1	L0
	02	0 0 1 0	L1.X1
	03	0 0 1 1	L0, L1.X1
	04	0 1 0 0	L1.X2
	05	0 1 0 1	L0, L1.X2
	06	0 1 1 0	L1.X1, L1.X2
	07	0 1 1 1	L0, L1.X1, L1.X2
	08	1 0 0 0	Ein "0"-Signal, entweder am Eingang L1.X1 oder am Eingang L1.X2, ruft den Meßwert von beiden Achsen ab.
	09	1 0 0 1	L0, LOut
	0B	1 0 1 1	L0, L1.X1, LOut
	0D	1 1 0 1	L0, L1.X2, LOut
	0F	1 1 1 1	L0, L1.X1, L1.X2, LOut
		D7 D6 D5 D4	**
Befehle für den Abruf-Zähler	00	0 0 0 0	Abruf-Zähler hat keine Funktion
	10	0 0 0 1	Abruf-Zähler starten
	20	0 0 1 0	Abruf-Zähler stoppen
	30	0 0 1 1	Starten mit Strobe (Register 60h)
	40	0 1 0 0	Stoppen mit Strobe (Register 60h)
	50	0 1 0 1	Starten mit Referenzmarken-Signal
	60	0 1 1 0	Stoppen mit Referenzmarken-Signal
	70	0 1 1 1	Meßwert abrufen
	80	1 0 0 0	Abruf-Zähler nullen
	90	1 0 0 1	Nullen mit Strobe (Register 60h)
	A0	1 0 1 0	Nullen mit Referenzmarken-Signal
	B0	1 0 1 1	Abruf-Wert laden
	C0	1 1 0 0	Laden mit Strobe (Register 60h)
	D0	1 1 0 1	Laden mit Referenzmarken-Signal
	E0	1 1 1 0	Meßwert abrufen mit Strobe
	F0	1 1 1 1	Abrufen mit Referenzmarken-Signal

*L0 = Abruf-Signal vom Abruf-Zähler; aktiviert den Meßwert-Abruf von beiden Zählern

L1.X1 = Abruf-Signal von Flanschdose X3, Anschluß 1; speichert den Meßwert von Eingang X1 in Daten-Register 1

L1.X2 = Abruf-Signal von Flanschdose X3, Anschluß 2; speichert den Meßwert von Eingang X2 im Daten-Register 1

LOut = Abruf-Signal an Flanschdose X3, Anschluß 3; zur Weiterleitung an eine weitere Zählerkarte oder eine andere Elektronik

**Diese Befehle sind nur für den Abruf-Zähler wirksam und nicht für die Meßwert-Zähler X1 und X2.

40h bis 50h: Abruf-Wert

Die Register-Adressen 40h und 50h legen die Anzahl der Zählimpulse für den Abruf-Zähler fest. Beim Erreichen des Zählerwertes "Null" wird das Abruf-Signal L0 ausgegeben und der Abruf-Zähler erneut mit dem Abruf-Wert geladen. Der Abruf-Wert ist eine 16 Bit breite positive Integer-Zahl, die wie folgt gespeichert wird:

- Adresse 40h, unteres Byte des Abruf-Wertes
- Adresse 50h, oberes Byte des Abruf-Wertes

Beispiel:
Gewünscht ist ein Abruf-Signal pro Millimeter Verfahrweg. Die Teilungsperiode des Meßsystems beträgt 10 µm.
Gesucht: der Abruf-Wert bei 200fach-Unterteilung des Meßsystem-Signals.

$$\frac{1\ mm}{0.010} * 200 = 20000\ \text{Zählimpulse pro mm}$$

20000 in hexadezimaler Schreibweise ist 4E20h
Unteres Byte des Abruf-Wertes auf Adresse 40h= 20h
Oberes Byte des Abruf-Wertes auf Adresse 50h= 4Eh

60h: Strobe-Register für Abruf-Zähler

Ein Schreibzugriff auf dieses Register erzeugt ein Strobe-Signal für den Abruf-Zähler. Im Kontroll-Register 2 ist festgelegt, wie der Abruf-Zähler auf das Strobe-Signal reagiert.
Wenn beispielsweise "Meßwert abrufen mit Strobe" programmiert wird, dann werden die Meßwerte beider Zähler mit dem Strobe-Signal abgerufen.

70h: Interrupt rücksetzen

Ein Schreibzugriff auf dieses Register setzt nach einem Interrupt die Interrupt-Flip-Flops rück.

Programmierung

Die Programmierung in dieser Beschreibung wird mit allgemeingültigen Beispielen, TURBO PASCAL- und QBASIC-Beispielen gezeigt. In den allgemeingültigen Beispielen werden symbolische Befehle verwendet, beispielsweise für Schreibzugriffe WRITE und für Lesezugriffe READ. Es sollte kein Problem darstellen, diese symbolischen Befehle durch die Befehle der verwendeten Programmiersprache zu ersetzen.

Die IK 120 verwendet RAM-Speicher-Adressen und kann mit jeder Programmiersprache programmiert werden, die den "hohen Speicherbereich" ansprechen kann – also den Bereich zwischen A0000h und FFFFFh (640 Kbyte und 1024 Kbyte). Die IK 120 nutzt den Bereich ab Adresse C0000 und belegt 1024 Bytes.

Die Segment-Adresse der IK 120 wird über DIP-Schalter eingestellt. Bei den Adreß-Angaben in den folgenden Beispielen handelt es sich immer um Offset-Adressen (mit "@" gekennzeichnet) in hexadezimaler Schreibweise.

Auf der IK 120 befinden sich zwei Zählerbausteine mit den folgenden Offset-Adressen:

- Zählerbaustein X1: Adresse 00h bis 0Fh
- Zählerbaustein X2: Adresse 10h bis 1Fh

Physikalische Adresse = Segment- + Offset-Adresse

IK 120 initialisieren

Die folgenden Befehle initialisieren die Kontroll-Register:

```
WRITE   @20h  0Fh
WRITE   @20h  03h
WRITE   @20h  0Fh
```

Kontroll-Register 1
Unterteilung 200fach und mit Hilfe der Folge 1 – 0 – 1 über Bits D2 und D3 die Meldung "Meßsystemfehler" rücksetzen

```
WRITE   @30h  00h
```

Kontroll-Register 2
Meßwert-Abruf nur über Software

Die folgenden Befehle initialisieren die Zählerbausteine:

			Zähler X1
WRITE	@0Fh	00h	Die Abruf-Eingänge der Zähler aktivieren; Abruf über Referenzmarke wird im Daten-Register 0 gespeichert
WRITE	@0Dh	FFh	Status-Register löschen
WRITE	@0Ch	03h	Zähler-Betriebsart festlegen: 4fach Auswertung; Zählrichtung normal; Linear-Zählweise
WRITE	@0Ah	00h	Referenzmarken inaktiv
WRITE	@0Bh	05h	Zähler Nullen und Starten
READ	@03h	DUMMY	Daten-Register 0 aktivieren
READ	@07h	DUMMY	Daten-Register 1 aktivieren
			Zähler X2
WRITE	@1Fh	00h	
WRITE	@1Dh	FFh	
WRITE	@1Ch	03h	
WRITE	@1Ah	00h	
WRITE	@1Bh	05h	
READ	@13h	DUMMY	
READ	@17h	DUMMY	

Meßwert-Abruf über Software

Nachdem die Zähler initialisiert sind, speichern die folgenden Befehle die Zählerstände im Daten-Register 0:

WRITE	@08h	FFh	Speichert den Zählerstand X1 im Daten-Register 0
WRITE	@18h	FFh	Speichert den Zählerstand X2 im Daten-Register 0

Die Meßwerte können vom Daten-Register 0 wie folgt gelesen werden:

READ	@00h	WertX1	Liest den Zählerstand in die Variable <Wert X1>
READ	@10h	WertX2	Liest den Zählerstand in die Variable <Wert X2>

Zählerstand in Millimeter umrechnen

$$\text{Wert } [\text{mm}] = \text{Zählerstand} * \frac{\text{Teilungsperiode } [\text{mm}]}{\text{Auswertung} * \text{ Interpolation}}$$

Beispiel: Teilungsperiode = 20 µm; Auswertung = 4fach; Interpolation = 50fach

$$\text{Wert } [\text{mm}] = \text{Zählerstand} * \frac{0.020\ [\text{mm}]}{4 * 50}$$

Zählerstand in Grad umrechnen

$$\text{Wert } [^\circ] = \frac{\text{Zählerstand} * 360\ [^\circ]}{\text{Auswertung} * \text{ Interpolation} * \text{ Striche / Umdr.}}$$

Beispiel: Striche/Umdr. = 36 000; Auswertung = 4fach; Interpolation = 50fach

$$\text{Wert } [^\circ] = \frac{\text{Zählerstand} * 360\ [^\circ]}{4 * 50 * 36\ 000}$$

Meßwerte über Software abrufen und anzeigen

```
                                  Kontroll-Register 1
                                  initialisieren
WRITE   @20h  0Fh                 Unterteilung 200fach und mit
WRITE   @20h  03h                 Hilfe der Folge 1 – 0 – 1
WRITE   @20h  0Fh                 über die Bits D2 und D3.
                                  Die Meldung "Meßsystem-
                                  fehler" rücksetzen
                                  Kontroll-Register 2
                                  initialisieren
WRITE   @30h  00h                 Meßwert-Abruf nur über
                                  Software
                                  Zähler X1 initialisieren
WRITE   @0Fh  00h                 Abruf-Eingänge der Zähler
                                  aktivieren; Abruf über Re-
                                  ferenzmarke wird im Daten-
                                  Register 0 gespeichert.
WRITE   @0Dh  FFh                 Status-Register löschen
WRITE   @0Ch  03h                 4fach-Auswertung; Zählrichtung
                                  normal; Linear-Zählweise
WRITE   @0Ah  00h                 Referenzmarken inaktiv
WRITE   @0Bh  05h                 Zähler nullen und starten
READ    @03h  DUMMY               Daten-Register 0 aktivieren
READ    @07h  DUMMY               Daten-Register 1 aktivieren
                                  Zähler X2 initialisieren
WRITE   @1Fh  00h
WRITE   @1Dh  FFh
WRITE   @1Ch  03h
WRITE   @1Ah  00h
WRITE   @1Bh  05h
READ    @13h  DUMMY
READ    @17h  DUMMY
WHILE NOT KEYPRESSED DO
BEGIN
WRITE @08h FFh                    Zählerstand X1 in Daten-
                                  Register 0 speichern
WRITE @18h FFh                    Zählerstand X2 in Daten-
                                  Register 0 speichern
READ     @00h CountX1             Meßwert X1 lesen (32 Bit)
READ     @10h CountX2             Meßwert X2 lesen (32 Bit)
PRINT CountX1 * 0.02 / 200        Meßwert X1 in mm
PRINT CountX2 * 0.02 / 200        Meßwert X2 in mm
END
```

```
{Beispiel 1 in TURBO PASCAL: Meßwerte über Software abrufen und anzeigen}
PROGRAM Example1;
USES CRT;
VAR   CountX1, CountX2: longint;
BEGIN
   mem[$C800:$20]:= $0F;          {Unterteilung 200fach}
   mem[$C800:$20]:= $03;          {"Meßsystemfehler" rücksetzen}
   mem[$C800:$20]:= $0F;          {"Meßsystemfehler" rücksetzen}
   mem[$C800:$30]:= $00;          {Meßwert-Abruf über Software}
                                  {ZÄHLER X1}
   mem[$C800:$0F]:= $00;          {Abruf-Eingänge der Zähler aktivieren}
   mem[$C800:$0D]:= $FF;          {Status-Register löschen}
   mem[$C800:$0C]:= $03;          {4fach-Auswertung}
   mem[$C800:$0A]:= $00;          {Referenzmarken inaktiv}
   mem[$C800:$0B]:= $05;          {Zähler nullen und starten}
   CountX1:=   meml[$C800:$00];{Daten-Register 0 aktivieren}
   CountX1:=   meml[$C800:$04];{Daten-Register 1 aktivieren}

   mem[$C800:$1F]:= $00;          {ZÄHLER X2}
   mem[$C800:$1D]:= $FF;
   mem[$C800:$1C]:= $03;
   mem[$C800:$1A]:= $00;
   mem[$C800:$1B]:= $05;
   CountX2:=      meml[$C800:$10];
   CountX2:=      meml[$C800:$14];

   ClrScr;
   REPEAT
      mem[$C800:$08]:= $FF;          {Zählerstand X1 in Daten-Register 0}
      mem[$C800:$18]:= $FF;          {Zählerstand X2 in Daten-Register 0}
      CountX1:= meml[$C800:$00];     {Meßwert X1 lesen}
      CountX2:= meml[$C800:$10];     {Meßwert X2 lesen}
      GotoXY(10,10);
      WRITELN(  'X1= ',CountX1*0.02/200 :6:3,     {Meßwert X1 und}
              '  X2= ',CountX2*0.02/200 :6:3);    {X2 in mm ausgeben}
   UNTIL KEYPRESSED;
END.
```

'Beispiel 1 in QBASIC: Meßwerte über Software abrufen und anzeigen

```
DEF SEG = &HC800
POKE &H20, &HF
POKE &H20, &H3
POKE &H20, &HF
POKE &H30, &H0

POKE &HF, &H0
POKE &HD, &HFF
POKE &HC, &H3
POKE &HA, &H0
POKE &HB, &H5
Dummy = PEEK(&H3)
Dummy = PEEK(&H7)

POKE &H1F, &H0
POKE &H1D, &HFF
POKE &H1C, &H3
POKE &H1A, &H0
POKE &H11B, &H5
Dummy = PEEK(&H13)
Dummy = PEEK(&H17)

CLS
PRINT "Test-Programm in QBASIC: Meßwert-Abruf über Software"
DIM COUNTX1 AS LONG
DIM COUNTX2 AS LONG

WHILE INKEY$ <> CHR$(27)
    POKE &H8, &HFF
    POKE &H18, &HFF
    P00 = PEEK(0)
    P01 = PEEK(1)
    P02 = PEEK(2)
    P03 = PEEK(3)
    P10 = PEEK(&H10)
    P11 = PEEK(&H11)
    P12 = PEEK(&H12)
    P13 = PEEK(&H13)

    IF (P03 AND &H80) = &H80 THEN
      COUNTX1& = (P00 - 255) + 256 * (P01 - 255) + 65536 * (P02 - 255) +
                 16777216 * (P03 - 255) - 1
    ELSE COUNTX1& = P00 + 256 * P01 + 65536 * P02 +  16777216 * P03
    END IF
    IF (P13 AND &H80) = &H80 THEN
```

```
    COUNTX2& = (P10 - 255) + 256 * (P11 - 255) + 65536 * (P12 - 255) +
               16777216 * (P13 - 255) - 1
  ELSE COUNTX2& = P10 + 256 * P11 + 65536 * P12 + 16777216 * P13
  END IF

  LOCATE 9, 1
  PRINT USING "####.###"; COUNTX1& * .02 / 200; COUNTX2& * .02 / 200
WEND
```

Meßsystemfehler prüfen und rücksetzen

Das Bit D3 im Register 0Eh ist gesetzt, falls die Amplituden der Meßsystem-Signale zu groß sind oder die Eingangsfrequenz überschritten wurde. Dieses Fehler-Bit wird wie folgt rückgesetzt: Schreiben der Sequenz 1 – 0 – 1 in das Kontroll-Register 20h – Bit D2 für Achse X1 und Bit D3 für Achse X2. Anschließend das Status-Register 0Fh löschen –durch Schreiben von FFh in das Register 0Dh.

```
READ    @0E   STATUSX1
READ    @1E   STATUSX2

IF  STATUSX1 BIT D3=1 THEN
BEGIN
        WRITE "Meßsystemfehler in X1"
        WRITE @20h 07h
        WRITE @20h 03h          Meldung
                                "Meßsystemfehler"
        WRITE @20h 07h          rücksetzen
        WRITE @0Dh 07h
END

IF  STATUSX2 BIT D3=1 THEN
BEGIN
        WRITE "Meßsystemfehler in X2"
        WRITE @20h 0Fh
        WRITE @20h 07h          Meldung
                                "Meßsystemfehler"
        WRITE @20h 0Fh          rücksetzen
        WRITE @0Dh 0Fh
END
```

Abrufen der Meßwerte über externe Abruf-Eingänge L1.X1, L1.X2

Damit die Abruf-Eingänge L1.X1 und L1.X2 aktiv sind, muß das Kontroll-Register 30h wie folgt initialisiert werden:

```
WRITE     @30h 06h     Abruf-Eingänge L1.X1 und L1.X2
                       aktivieren
```

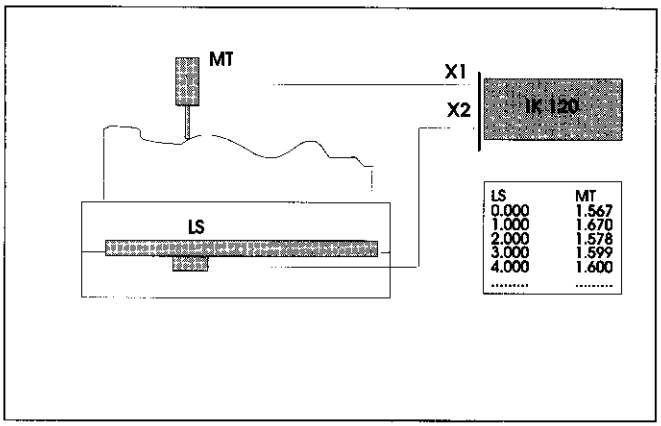
Der Meßwert eines externen Abrufs ist im Daten-Register 1 gespeichert. Gleichzeitig wird im Status-Register 0Eh das Bit D1 gesetzt. Nach dem Lesen des Meßwertes ist dieses Bit wieder rückgesetzt.

Ein externer Meßwert-Abruf wird immer im Daten-Register 1 gespeichert.

```
READ    @0Eh  STATUSX1
READ    @1Eh  STATUSX2

IF STATUSX1 BIT D1=1 THEN
BEGIN
        READ @04h  COUNTX1      Daten-Register 1 lesen
        PRINT 'X1 =', COUNTX1 * 0.02 / 200
END

IF STATUSX2 BIT D1=1 THEN
BEGIN
        READ @14h  COUNTX2      Daten-Register 1 lesen
        PRINT 'X2=', COUNTX2 * 0.02 / 200
END
```

Meßwerte über den Abruf-Zähler abrufen

Der Abruf-Zähler (16 Bit) ist der dritte Zähler auf der IK 120. Er ist parallel zum Zähler für Eingang X2 geschaltet. Er wird benutzt, um in bestimmten Abständen Meßwerte abzurufen. Dies ist beispielsweise beim Vermessen einer Achse mit einem Vergleichsmeßsystem sinnvoll.
Der Abruf-Zähler zählt zyklisch vom programmierten Abruf-Wert auf Null und erzeugt jedesmal beim Erreichen des Zählerstands Null ein internes Abruf-Signal (L0). Das Abruf-Signal L0 wird über Bits D0 bis D3 des Kontroll-Registers 2 (Register-Adresse 30h) aktiviert.



Berechnung des Abruf-Wertes:

$$\text{Impulse per Intervall} = \frac{\text{Intervall} * \text{Auswertung} * \text{Interpolation}}{\text{Signalperiode}}$$

Beispiel:
Der Abruf-Zähler soll jeden mm ein Abruf-Signal erzeugen und den Meßwert des HEIDENHAIN-METRO-Tasters speichern.

$$\text{Impulse per Intervall} = \frac{1\,[\text{mm}] * 4 * 50}{0{,}02\,[\text{mm}]}$$

= 10000
= 2710hex

Unteres Byte = 10h
Oberes Byte = 27h

Den Abruf-Zähler initialisieren:

WRITE	@40h 10h	Unteres Byte des Abruf-Wertes
WRITE	@50h 27h	Oberes Byte des Abruf-Wertes
WRITE	@20h 0Fh	4fach-Auswertung
WRITE	@30h 8Fh	Abruf-Zähler nullen und L0 aktivieren
WRITE	@30h 1Fh	Abruf-Zähler starten

Alle 10 000 Zählimpulse (2710h) oder jeden mm erzeugt der Abruf-Zähler ein Abruf-Signal, das über den Anschluß L0 an beide Zähler weitergeleitet wird. Bit D0 des Status-Registers (0Eh) zeigt an, ob ein interner Abruf über L0 erfolgt ist. Erst nach dem Lesen des Meßwertes im Daten-Register 0 kann ein weiterer Meßwert-Abruf erfolgen.

```
READ    @0E   STATUSX1
IF STATUSX1 BIT D0 = 1  THEN
BEGIN
        READ @00h          COUNTX1  Daten-Register 0
        READ @10h          COUNTX2  Daten-Register 0
        PRINT 'X1 =', COUNTX1 * 0.01 / 200
        PRINT 'X2 =', COUNTX2 * 0.01 / 200
END
```

```
{Beispiel 2 in TURBO PASCAL: Meßwert-Abruf über den Abruf-Zähler}
PROGRAM Example2;
USES CRT;
VAR   CountX1, CountX2: longint;
BEGIN
   mem[$C800:$20]:= $0F;          {Unterteilung 200fach}
   mem[$C800:$20]:= $03;          {"Meßsystemfehler" rücksetzen}
   mem[$C800:$20]:= $0F;          {"Meßsystemfehler" rücksetzen}
   mem[$C800:$30]:= $00;          {Meßwert-Abruf über Software}
                                  {ZÄHLER X1}
   mem[$C800:$0F]:= $00;          {Abruf-Eingänge der Zähler aktivieren}
   mem[$C800:$0D]:= $FF;          {Status-Register löschen}
   mem[$C800:$0C]:= $03;          {4fach-Auswertung}
   mem[$C800:$0A]:= $00;          {Referenzmarken inaktiv}
   mem[$C800:$0B]:= $05;          {Zähler nullen und starten}
   CountX1:=  meml[$C800:$00];    {Daten-Register 0 aktivieren}
   CountX1:=  meml[$C800:$04];    {Daten-Register 1 aktivieren}

   mem[$C800:$1F]:= $00;          {ZÄHLER X2}
   mem[$C800:$1D]:= $FF;
   mem[$C800:$1C]:= $03;
   mem[$C800:$1A]:= $00;
   mem[$C800:$1B]:= $05;
   CountX2:=  meml[$C800:$10];
   CountX2:=  meml[$C800:$14];
                                  {ABRUF-ZÄHLER}
   mem[$C800:$40]:= $10;          {Unteres Byte des Abruf-Wertes}
   mem[$C800:$50]:= $27;          {Oberes Byte des Abruf-Wertes}
   mem[$C800:$20]:= $0F;          {4fach Auswertung}
   mem[$C800:$30]:= $8F;          {Abruf-Zähler nullen und L0 aktivieren}
   mem[$C800:$30]:= $1F;          {Abruf-Zähler starten}

   ClrScr;
   REPEAT
      IF (mem[$C800:$0E] AND $01) =1 THEN      {Falls Meßwert gespeichert:}
      BEGIN
         CountX1:= meml[$C800:$00];     {Meßwert X1 lesen}
         CountX2:= meml[$C800:$10];     {Meßwert X2 lesen}
         WRITELN(   'MT= ',CountX1*0.01/200 :6:3,
                    '  LS= ',CountX2*0.02/200 :6:3);
      END;
   UNTIL KEYPRESSED;
END.
```

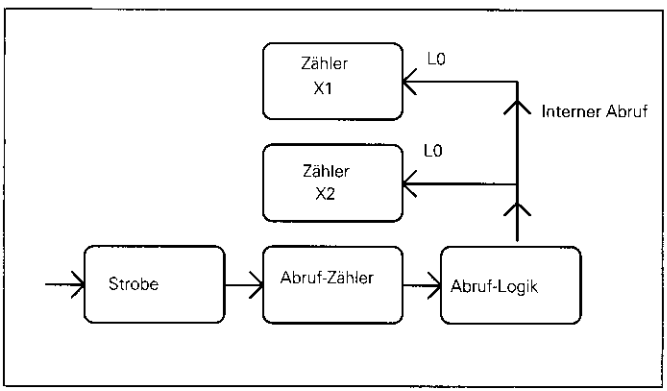
'Beispiel 2 in QBASIC: Meßwert-Abruf über den Abruf-Zähler

```
DEF SEG = &HC800
POKE &H20, &HF
POKE &H20, &H3
POKE &H20, &HF
POKE &H30, &H0

POKE &HF, &H0
POKE &HD, &HFF
POKE &HC, &H3
POKE &HA, &H0
POKE &HB, &H5
Dummy = PEEK(&H3)
Dummy = PEEK(&H7)

POKE &H1F, &H0
POKE &H1D, &HFF
POKE &H1C, &H3
POKE &H1A, &H0
POKE &H11B, &H5
Dummy = PEEK(&H13)
Dummy = PEEK(&H17)

POKE &H40, &H10
POKE &H50, &H27
POKE &H20, &HF
POKE &H30, &H8F
POKE &H30, &H1F

CLS
PRINT "Testprogramm in QBASIC für den Meßwert-Abruf über Abruf-Zähler"
DIM COUNTX1 AS LONG
DIM COUNTX2 AS LONG

WHILE INKEY$ <> CHR$(27)
    IF (PEEK(&HE) AND 1) = 1 THEN
      P00 = PEEK(0)
      P01 = PEEK(1)
      P02 = PEEK(2)
      P03 = PEEK(3)
      P10 = PEEK(&H10)
      P11 = PEEK(&H11)
      P12 = PEEK(&H12)
      P13 = PEEK(&H13)
```

```
    IF (P03 AND &H80) = &H80 THEN
      COUNTX1& = (P00 - 255) + 256 * (P01 - 255) + 65536 * (P02 - 255) +
                 16777216 * (P03 - 255) - 1
    ELSE COUNTX1& = P00 + 256 * P01 + 65536 * P02 + 16777216 * P03
    END IF

    IF (P13 AND &H80) = &H80 THEN
      COUNTX2& = (P10 - 255) + 256 * (P11 - 255) + 65536 * (P12 - 255) +
                 16777216 * (P13 - 255) - 1
    ELSE COUNTX2& = P10 + 256 * P11 + 65536 * P12 + 16777216 * P13
    END IF
    PRINT USING "####.###"; COUNTX1& * .02 / 200;
                            COUNTX2& * .02 / 200
  END IF
WEND
```

Meßwerte von zwei Zählern auf einer IK 120 simultan abrufen

Ein Schreibbefehl an das Strobe-Register 60h ruft die Meßwerte beider Zähler der IK 120 gleichzeitig ab. Dazu muß der Abruf-Zähler über das Kontroll-Register 2 (Register-Adresse 30h) programmiert werden.

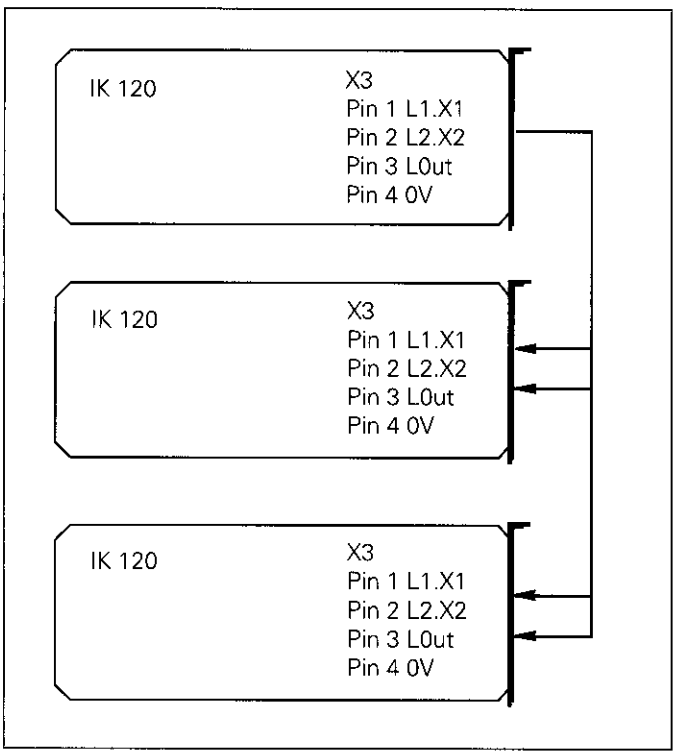


Der Meßwert-Abruf über Strobe wird wie folgt programmiert:

WRITE	@30h EFh	Meßwert abrufen mit Strobe; Abruf-Signale aktivieren
WRITE	@60h FFh	Strobe
READ	@00h COUNTX1	Meßwert X1 lesen
READ	@10h COUNTX2	Meßwert X2 lesen

Meßwerte von mehreren IK 120 abrufen

Über den Abruf-Ausgang LOut ist es möglich, die Meßwerte von mehreren IK 120 abzurufen. Dazu ist der Abruf-Ausgang LOut einer IK 120 mit den Abruf-Eingängen weiterer IK 120 zu verbinden.



Meßwert-Abruf von zwei IK 120:

WRITE	@30h EFh		Meßwert abrufen mit Strobe; Abruf-Signale aktivieren
WRITE	@430h 08h		Externe Abruf-Signale L1.X1 und L1.X2 für die zweite IK 120 aktivieren
WRITE	@60 FFh		Strobe Erste IK 120
READ	@00	COUNTX1	Daten-Register 0
READ	@00	COUNTX2	Daten-Register 0 Zweite IK 120
READ	@404	COUNTX3	Daten-Register 1
READ	@404	COUNTX4	Daten-Register 1

Interrupt-Programmierung

Die IK 120 kann auch einen Hardware-Interrupt mit dem internen Abruf-Signal L0 oder mit den externen Abruf-Signalen L1.X1 sowie L1.X2 auf dem PC-Bus erzeugen.

Folgende Interrupts können genutzt werden: IRQ3,IRQ4,IRQ5 und IRQ7. Eine Brücke auf der IK 120 legt den gewünschten Interrupt fest (siehe "Hardware").

Mit Hilfe des Interrupts kann eine Software-Routine gerufen werden, die die Meßwerte abruft.

Nur erfahrene Programmierer sollten sich an die Interrupt-Programmierung wagen, da

- die Interrupt-Programmierung sehr problematisch sein kann (Computer-Abstürze),
- nur sehr wenige Anwendungen dies erfordern und
- die zyklische Abfrage des Status-Bytes ebenfalls eine schnelle Methode zum Erkennen eines Meßwert-Abrufes ist.

Die Hardware-Interrupts gibt das Kontroll-Register 1 (Register-Adresse 20h) frei:

Bit D6 gibt **Int0** frei, der vom Abruf-Zähler erzeugt wird.

Bit D7 gibt **Int1** frei, der von den externen Abruf-Signalen erzeugt wird.

Ein externer Abruf-Impuls muß kürzer als die Interrupt-Routine sein!

Die Anwendung der Interrupt-Programmierung zeigt das Programm-Beispiel "INTERUPT.PAS" auf der mitgelieferten Diskette.

Referenzmarken nutzen

Die HEIDENHAIN-Längenmeßsysteme und -Winkelmeßsysteme besitzen eine oder mehrere – insbesondere "abstandscodierte" – Referenzmarken. Das Überfahren einer Referenzmarke erzeugt ein Signal, das diese Position als Referenzpunkt kennzeichnet. Nach dem Wiedereinschalten nach einer Stromunterbrechung kann durch das Überfahren des Referenzpunktes die festgelegte Zuordnung zwischen Positionen und Anzeigewerten wieder hergestellt werden. Bei abstandscodierten Referenzmarken genügt dazu ein Verfahrweg von maximal 20 mm.

Das Referenzmarken-Register legt die Wirkung der Referenzmarke auf die Zählerbausteine fest. So können mit dem Referenzmarken-Signal die Zähler gestartet, gestoppt, genullt oder der Zählerwert abgerufen werden.

Üblicherweise soll das Überfahren der Referenzmarken die Zähler nullen und starten. Dadurch erhält man einen einfach reproduzierbaren Bezugspunkt. Dieser Bezugspunkt kann dann – falls gewünscht – per Software durch Addieren eines Wertes verschoben werden.

Das Arbeiten mit Referenzmarken zeigt das Programm-Beispiel "CREFMARK.PAS" auf der mitgelieferten Diskette.

Software

Im Lieferumfang:Treiber-Software und Beispiele in TURBO PASCAL im Verzeichnis „TP" und MICROSOFT C im Verzeichnis „C600". Im Verzeichnis „MORE_TP" befinden sich weitere Beispiele in TURBO PASCAL.

Funktionsbeschreibung der Treiber-Software

Das folgende Kapitel beschreibt die Treiber-Software im Verzeichnis „TP" in TURBO PASCAL. Da beide Softwares die gleichen Funktions- und Variablen-Bezeichnungen verwenden, sollte das Verstehen der Treiber-Software im Verzeichnis „C600" in MICROSOFT C keine Probleme bereiten.

Variablen-Definitionen, Daten-Typen und Funktionen
Globale Variablen, Daten-Typen und Funktionen sind in folgenden Dateien definiert:

- für TURBO PASCAL in der Datei "IK120.PAS"
- für MICROSOFT C in den Dateien "IK120.H" sowie "IK120.C"

Diese Dateien müssen in die Anwendungsprogramme, die die Funktionen der Treiber-Software nutzen, eingebunden werden.

Variable:	**boardadr** Spezifiziert die Segment-Adresse der ersten IK 120. Diese Adresse wird in der Datei "K 120.PAS" auf C800 gesetzt. Sie muß geändert werden, falls auf der Platine eine andere Adresse über die DIP-Schalter eingestellt wird.
Anwendung:	In allen Funktionen der Treiber-Software
Werte:	C800h, C840h, C880h,... FFC0h

Typ:	**control** Programmierbare Funktionen der Zählerbausteine, z.B. Start, Stop, Nullen.
Anwendung:	Init_Counter, Init_Latch
Werte:	siehe nachfolgende Tabelle

Wert	Bedeutung
c_reset	Zähler nullen
c_start	Zähler starten
c_stop	Zähler stoppen
reset_start	Zähler nullen und starten
reset_stop	Zähler nullen und stoppen
RI_stop	Zähler stoppen beim Überfahren der Referenzmarke
RI_start	Zähler starten beim Überfahren der Referenzmarke
RI_reset_stop	Zähler nullen und stoppen beim Überfahren der Referenzmarke
RI_reset_start	Zähler nullen und starten beim Überfahren der Referenzmarke
RI_reset	Zähler nullen beim Überfahren der Referenzmarke
RI_latch_stop	Zählerwert beim Überfahren der Referenzmarke abrufen und in dem Daten-Register, das im Initialisierungs-Register 0Fh festgelegt wurde, speichern und Zähler stoppen
RI_latch_start	Zählerwert beim Überfahren der Referenzmarke abrufen und in dem Daten-Register, das im Initialisierungs-Register 0Fh festgelegt wurde, speichern und Zähler starten
RI_latch_reset_stop	Zählerwert beim Überfahren der Referenzmarke abrufen und in dem Daten-Register, das im Initialisierungs-Register 0Fh festgelegt wurde, speichern; Zähler stoppen und nullen
RI_latch_reset_start	Zählerwert beim Überfahren der Referenzmarke abrufen und in dem Daten-Register, das im Initialisierungs-Register 0Fh festgelegt wurde, speichern; Zähler starten und nullen
RI_latch_reset	Zählerwert beim Überfahren der Referenzmarke abrufen und in dem Daten-Register, das im Initialisierungs-Register 0Fh festgelegt wurde, speichern und Zähler nullen

Die folgenden Werte werden nur in der Funktion "Init_Latch" verwendet:

Wert	Bedeutung
c_latch	Der Abruf-Zähler erzeugt sofort ein Abruf-Signal
c_load	Den Abruf-Zähler auf einen Wert setzen
RI_load	Den Abruf-Zähler beim Überfahren der Referenzmarken auf einen Wert setzen
Strobe_Latch	Erzeugt einen Meßwert-Abruf, falls ein Schreibzugriff auf das Strobe-Register erfolgt

Typ: **eval**
Legt die Signal-Auswertung der Zähler fest
Anwendung: Init_Counter, Init_Latch, DistanceCodedREFMarks

Wert	Bedeutung
onefold	Auswertung: 1fach
twofold	Auswertung: 2fach
fourfold	Auswertung: 4fach

Typ: **direction**
Legt die Zählrichtung der Zähler fest
Anwendung: Init_Counter, Init_Latch, DistanceCodedREFMarks

Wert	Bedeutung
normal	Zähler zählt positiv bei positiver Verfahrrichtung
inverse	Zähler zählt negativ bei positiver Verfahrrichtung

Typ: **method**
Legt die Zählweise des Zählers fest (Linear- oder Winkel-Zählweise)
Anwendung: Init_Counter, Init_Latch, DistanceCodedREFMarks

Wert	Bedeutung
linear	Linear-Zählweise Bereich: -2^{25} bis $+2^{25}-1$
arc	Winkel-Zählweise Bereich: –180000 bis +179999

Typ: **interpol**
Legt die Interpolation der Meßsystem-Signale fest
Anwendung: Interpolation

Wert	Bedeutung
I_25	25fach-Interpolation
I_50	50fach-Interpolation

Typ: **Lcontrol**
Aktiviert die Abruf-Signale
Anwendung: Latch_Enable, Latch_Disable

Wert	Bedeutung
Internal	Abruf-Signal vom Abruf-Zähler (L0)
ExternalX1	Externes Abruf-Signal für ersten Zähler (L1.X1)
ExternalX2	Externes Abruf-Signal für zweiten Zähler (L1.X2)
ExternalX1X2	Externes Abruf-Signal für beide Zähler (L1.X1 und L1.X2)
Latchout	Abruf-Ausgang (LOut)

Typ: **interr**
Legt den Auslöser eines Interrupts fest
Anwendung: Interrupt_Enable

Wert	Bedeutung
Int_0	Abruf-Zähler erzeugt Interrupt
Int_1	Externe Abruf-Eingänge erzeugen Interrupt

Funktion: Init_Interface
Diese Funktion initialisiert die Zähler am Anfang eines Programms.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2

Funktion: Interpolation
Diese Funktion legt die Interpolation der gewählten Achse fest.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
m_interpol:	I_25 = 25fach-Interpolation I_50 = 50fach-Interpolation

Funktion: Init_Counter
Diese Funktion legt die wichtigsten Eigenschaften der Zählerbausteine fest.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
m_control:	Variable des Typs "control" zur Festlegung der Funktionen des Abruf-Zählers
m_eval:	Auswertung = onefold, twofold, fourfold
direction:	Zählrichtung = normal oder inverse
m_method:	Zählweise = linear oder arc

Funktion: Soft_Count
Diese Funktion speichert den Zählerstand der gewünschten Achse im spezifizierten Daten-Register und gibt den Wert in der Variablen "COUNT" als 32-Bit-Integer in Zweierkomplement-Darstellung zurück.

"Soft_Count" ist eine Kombination aus "Latch_Count" und "Read_Count".
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
Register:	0 = Daten-Register 0, 1 = Daten-Register 1

Ergebnis:

COUNT	Meßwert aus dem spezifizierten Daten-Register als 32-Bit-Variable COUNT.

Funktion: Latch_Count
Diese Funktion speichert den Zählerstand der gewünschten Achse im spezifizierten Daten-Register.

Der Meßwert kann anschließend mit der Funktion "Read_Count" gelesen werden.

Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
Register:	0 = Daten-Register 0
	1 = Daten-Register 1

Funktion: Read_Count
Diese Funktion liest den zuletzt gespeicherten Wert der gewünschten Achse aus dem spezifizierten Daten-Register und gibt den Wert in der Variablen "COUNT" als 32-Bit-Integer in Zweierkomplement-Darstellung zurück.

Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
Register:	0 = Daten-Register 0
	1 = Daten-Register 1

Ergebnis:

COUNT	Meßwert aus dem spezifizierten Daten-Register als 32-Bit-Variable COUNT

Funktion: Reset_
Diese Funktion setzt das Status-Byte des gewählten Zählers rück.

Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2

Funktion: Reset_uas
Diese Funktion setzt das Meßsystemfehler-Bit (D2 für X1oder D3 für X2) in Kontoll-Register 1 (Register-Adresse 20h) rück.

Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2

Funktion: Signal_Error
Diese Funktion gibt "true" zurück, falls in der ausgewählten Achse ein Meßsystemfehler anliegt.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
Ergebnis:	true = Meßsystemfehler
	false = Kein Meßsystemfehler

Funktion: Read_Status
Diese Funktion gibt den Inhalt des Status-Registers zurück.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
Ergebnis:	Status-Register des gewählten Zählers

Funktion: Read_ScanReg
Diese Funktion gibt den Pegel verschiedener Pins der Zählerbausteine über das Zustands-Register (Register-Adresse 0Fh) zurück.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
Ergebnis:	Zustands-Register des gewählten Zählers

Funktion: Latch_Enable
Diese Funktion gibt das Abruf-Signal für den gewählten Zähler frei.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
Lcommand:	Internal, ExternalX1, ExternalX2, ExternalX1X2, LatchOut

Funktion: Latch_Disable
Diese Funktion sperrt das Abruf-Signal für den gewählten Zähler.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
Lcommand:	Internal, ExternalX1, ExternalX2, ExternalX1X2, LatchOut

Funktion: Latched
Diese Funktion fragt ab, ob ein Meßwert-Abruf stattgefunden hat.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
latch:	0 = Abruf vom Abruf-Zähler (L0) 1 = Externer Abruf (L1.X1, L1.X2)
Ergebnis:	true = Abruf ist erfolgt false = Kein Abruf

Funktion: Init_Latch
Diese Funktion initialisiert den Abruf-Zähler und setzt den Abruf-Wert.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
m_control:	Variable des Typs CONTROL zum Nullen, Starten und Stoppen des Abruf-Zählers.
m_eval:	Auswertung = onefold, twofold, fourfold
value:	Abruf-Wert (16 Bit)

Funktion: Write_Strobe
Diese Funktion erzeugt einen Strobe auf der gewählten Karte.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2

Funktion: Interrupt_Enable
Diese Funktion gibt den gewählten Interrupt frei.
Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
m_interr:	Int_0 = Interrupt vom Abruf-Zähler (L0) Int_1 = Interrupt vom externen Abruf-Signal (L1.X1,L1.X2)

Funktion: Clear_Int
Diese Funktion setzt das Interrupt-Flip-Flop auf der IK 120 rück; sie muß nach jedem Interrupt gerufen werden.
Parameter:
axis: 0 = Zähler X1, 1 = Zähler X2

Funktion: DistanceCodedREFMarks
("DistanceCodedREFMarks" steht nur in TURBO PASCAL zur Verfügung und nicht in MICROSOFT C.)

Diese Funktion wird bei Meßsystemen mit abstandscodierten Referenzmarken eingesetzt. Sie nullt den gewählten Zähler beim Überfahren der ersten Referenzmarke. Mit dem Überfahren der zweiten Referenzmarke wird die absolute Position der ersten Referenzmarke in Signalperioden berechnet und als Funktions-Ergebnis (32-Bit-Integer) rückgeliefert. Die absolute Position kann im Anwenderprogramm durch Multiplizieren mit der Teilungsperiode berechnet werden.

Parameter:

axis:	0 = Zähler X1, 1 = Zähler X2
BasicSpacing:	1000 GP, 2000 GP, 500 GP
SubDiv:	I_25 oder I_50
Edges:	onefold, twofold, fourfold
CountDir:	normal, inverse
Meth:	linear, arc
Ergebnis:	Position der ersten überfahrenen Referenzmarke – bezogen auf den Maßstabs-Nullpunkt – in Signalperioden

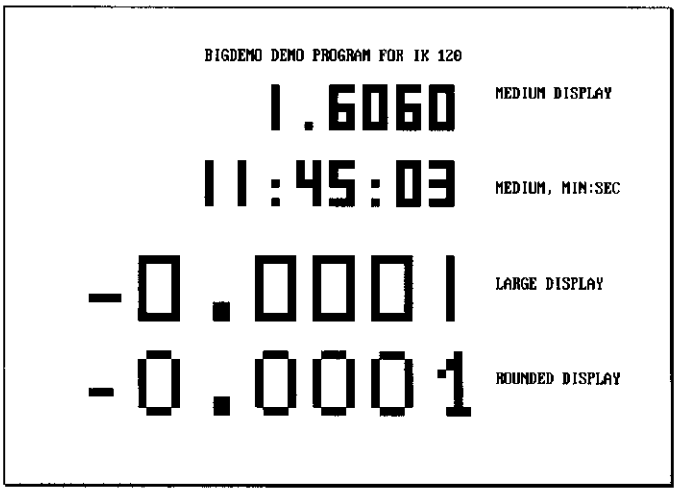
Der Parameter "BasicSpacing" (=Grundabstand) ist der konstante Abstand zwischen den Referenzmarken 0 und 2, 2 und 4, etc. Dieser Grundabstand wird in Signalperioden angegeben und beträgt bei HEIDENHAIN-Längenmeßsystemen normalerweise1 000. Andere Grundabstände wie 500 bei HEIDENHAIN-LID-Längenmeßsystemen oder 2 000 bei bestimmten HEIDENHAIN-Winkelmeßsystemen sind ebenfalls üblich. Der Grundabstand kann der Betriebs-anleitung des Meßsystems entnommen werden.

Wird diese Funktion durch Drücken einer Taste unterbrochen, dann wird der Wert 0 zurückgeliefert.

"Unit" für große Anzeige – BIGDISP.PAS

Die Datei "BIGDISP.PAS" ist eine TURBO PASCAL-"Unit" zur Anzeige einer Position mit großen, gut lesbaren Ziffern.

Das Programm "BIGDEMO.PAS" zeigt die Anwendung dieser "Unit" und die verschiedenen Anzeigemöglichkeiten.



Daten-Typen und Funktionen

Typ: **DisplayMode**
Legt die Größe der Anzeige fest
Anwendung: LDisplay

Wert	Bedeutung
Medium	Mittel
Large	Groß
Rounded	Groß mit abgerundeten Ziffern

Funktion: **LDisplay**

Parameter:

Value:	Der Wert, der angezeigt werden soll
DMode:	Art der Anzeige: Medium, Large, Rounded (siehe oben)
posx:	Horizontale Position des Zeichens links oben
posy:	Vertikale Position des Zeichens links oben
NoOfChars:	Anzahl der Zeichen, die angezeigt werden sollen, einschließlich Vorzeichen, Dezimalpunkt und Leerzeichen
DigitsAfterDecimalPoint:	Anzahl der Zeichen hinter dem Dezimalpunkt
MinutesSeconds:	True = Anzeige in Grad:Minuten:Sek. False = normale dezimale Anzeige
Flash:	True = Anzeige blinkt False = Anzeige blinkt nicht Die blinkende Anzeige erscheint üblicherweise bei einem Meßsystemfehler
LastValStr:	Diese Variable speichert den String, der auf dem Bildschirm angezeigt wird. Im Anwendungsprogramm muß für jede Achse ein eigener String mit den benötigten Zeichen initialisiert werden

"Unit" für feste Zeitintervalle - TIMER.PAS

Die Datei "TIMER.PAS" ist eine "Unit" in TURBO PASCAL, die Meßwerte in konstanten Zeit-Intervallen abruft. Diese "Unit" enthält spezielle Funktionen, mit denen Meßwerte in kürzeren Zeitabständen als 55 ms (in TURBO PASCAL üblich) abgerufen werden können.

Während der Laufzeit dieser "Unit" kann die DOS-Uhr langsamer oder auch schneller laufen.

Das Programm "MEASURE" zeigt die Anwendung dieser "Unit" (siehe Beschreibung weiter hinten):

Daten-Typen und Funktionen

Constant: **LocalMPMax**
Legt die maximale Anzahl von Meßpunkten fest, die abgerufen werden können
Gesetzt auf: 5000 Punkte

Variable: **Buffer**
Zweidimensionales Array, in dem die Zählerstände für jede Achse gespeichert werden
Definition: Buffer: array [0..LocalMpmax+1,0..1] of longint

Funktion: **SetTimerInterval**
Diese Funktion setzt den DOS-Timer-Interrupt so, daß er nicht mehr alle 55 ms erzeugt wird, sondern wie in dem Parameter "Intervall" festgelegt. DOS zählt jedoch weiterhin die Zeit in 55 ms-Schritten hoch. Deshalb läuft die interne DOS-Uhr bei Verwendung dieser Funktion nicht mehr richtig.
Parameter:
Interval: Das Zeit-Intervall zwischen zwei Meßwert-Abrufen in Millisekunden (real).

Funktion: **StartDataAquisition**
Diese Funktion speichert die im Parameter "No_of_points" festgelegte Anzahl von Meßwerten in dem Array "Buffer". Die Funktion "SetTimerInterval" muß vor dieser Funktion gerufen werden.
Parameter:
No_of_points: Anzahl der Meßwerte pro Achse

Funktion: **RemoveTimer**
Diese Funktion setzt den DOS-Timer-Interrupt wieder auf den DOS-Standard rück. Die DOS-Uhr läuft nach Aufruf dieser Funktion wieder richtig, muß jedoch gegebenenfalls auf die richtige Zeit gesetzt werden.
Parameter: none

Beispiel-Programme in TURBO PASCAL im Verzeichnis „TP"

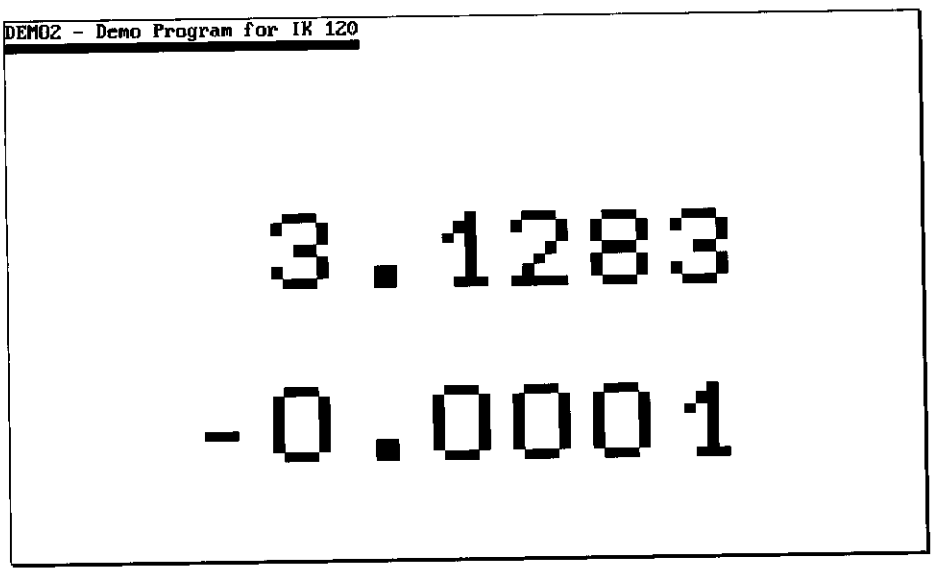
Auf der mitgelieferten Diskette sind für die Beispiele sowohl der Quell-Code „.PAS" als auch ausführbare Dateien „.EXE" unter dem Verzeichnis „TP" gespeichert.

DEMO2

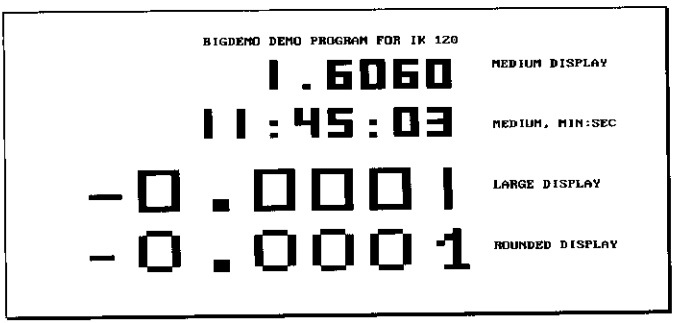
Ein einfaches Programm zur Anzeige von Positionen der Meßsysteme, die an die Eingänge X1 und X2 angeschlossen sind.

```
(*********************************************************************)
(* Dr JOHANNES HEIDENHAIN, Traunreut, Germany.                       *)
(* Demonstration program to illustrate the programming of the interface  *)
(* card IK120 using the IK120.PAS unit supplied.                     *)
(* Illustrates the use of Soft_Count                                 *)
(* Signal checking is included in this program                       *)
(*********************************************************************)
program DEMO2;

uses IK120, crt,BigDisp;

var
  Period:      real;
  SubDivision : integer;
  s,sz:        string;
  lastval :    array [0..1] of string;

procedure Display_count( axis,x,y: integer);
var Count:longint;
   Status: byte;
   Val:real;
begin
  Count := 0;
  Status := 0;
  Soft_Count(axis,Status,Count);
  Val:= Period*Count/Subdivision;
  Ldisplay(Val, Rounded,x,y,8,4,false,Signal_error(axis),lastval[axis]);
end;

(********************* main program ***********************)
BEGIN
   clrscr;
   Writeln('DEMO2 - Demo Program for IK 120');
   Writeln('_____');
   Period := 0.010;  { Grating period in mm }
   Subdivision := 200;
   m_interpol:=I_50;
```

```
  Init_Interface(0);
  Init_Interface(1);
  Interpolation(0,m_interpol);
  Interpolation(1,m_interpol);
  Init_Counter (0,reset_start,fourfold,normal,linear);
  Init_Counter (1,reset_start,fourfold,normal,linear);
  Reset_uas(0);
  Reset_uas(1);
  Reset_Status(0);
  Reset_Status(1);

  lastval[0]:='XXXXXXXXXX';
  lastval[1]:='XXXXXXXXXX';

  repeat
    Display_count(0,10,10);
    Display_count(1,10,18);
  until keypressed;
END.
```



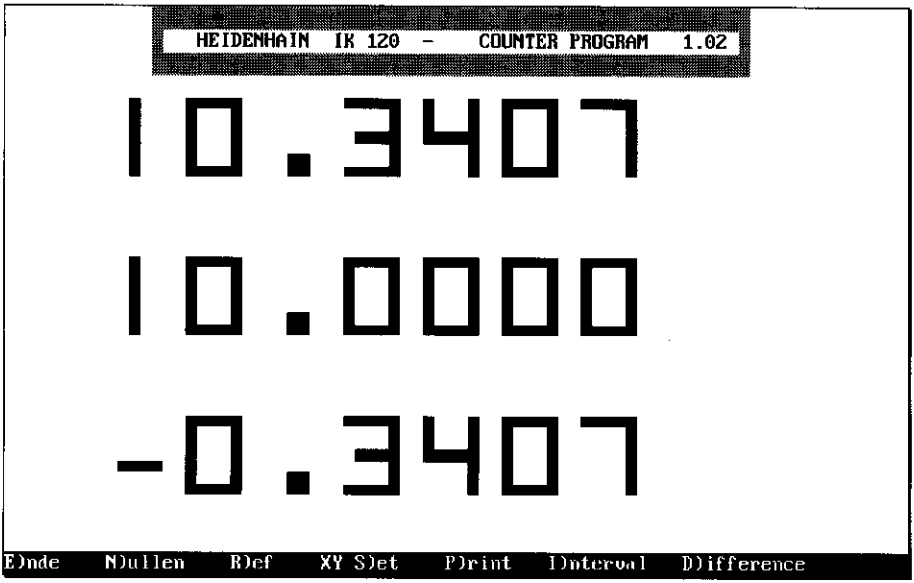
BIGDEMO
Dieses Programm zeigt die Anwendung des "Units" BIGDISP.PAS zur Anzeige von Meßwerten in großen Ziffern (siehe Beschreibung weiter oben).



COUNTER
Dieses Programm bietet die Standard-Funktionen einer Positions-Anzeige mit bis zu acht Eingängen auf vier IK 120. Funktionen:

- Nullen
- Setzen
- Große Anzeige
- Auswertung von einzelnen oder abstandscodierten Referenzmarken
- Anzeige der Meßwert-Differenz von zwei Eingängen
- Ausdrucken von Meßwerten
- Meßwert-Abruf über Abruf-Zähler

Anzeige-Beispiel



Das Programm "**COUNTER**" liest die Set-Up-Daten aus der Datei"COUNTPAR.DAT"

```
Number of Axes
2
Address of Interface card IK 120 in decimal format not HEX ( C800= 51200 )
51200
Places after the decimal point
4
Name for the axes e.g. XYZ
XYZ45678
Signal period in mm for each axes, Reference marks 0=Standard, 500, 1000, 2000
0.010 0
0.010 0
0.010 0
0.010 0
0.010 0
0.010 0
0.010 0
0.010 0
```

Diese Datei kann zur Festlegung der Grundeinstellungen geändert werden. Erklärung der Eingabewerte:

Number of axes: 1 bis 8
Legt die Anzahl der Achsen fest.

Address of the IK 120:
Spezifiziert die Segment-Adresse der ersten IK 120. Alle folgenden IK 120 haben eine um 40h höhere Segment-Adresse (muß über DIP-Schalter eingestellt werden).

Segment-Adresse	Dezimaler Eingabewert
C000	49152
C7C0	51136
C800	51200
C840	51264
C880	51328
D000	53248
E000	57344
F000	61440

Places after the decimal point:
Legt für die Anzeige die Anzahl der Stellen hinter dem Dezimalzeichen fest.

Name for the axes:
Legt die Achsbezeichnungen für die Anzeige fest, beispielsweise "XYZ45678", "ABCDEFGH" oder "12345678".

Signal period and reference mark:
In jeder Zeile wird die Signalperiode in mm und die Art der Referenzmarke festgelegt.

Durch Eingeben einer negativen Signalperiode wird die Zählrichtung umgedreht.

Referenzmarken:

0 = Einzelne Referenzmarke
1000 = Abstandscodierte Referenzmarken mit Grundabstand 1000 Teilungsperioden (z.B. Grundabstand = 20mm, Teilungsperiode = 20 µm)
2000 = Abstandscodierte Referenzmarken mit Grundabstand 2000 Teilungsperioden
500 = Abstandscodierte Referenzmarken mit Grundabstand 500 Teilungsperioden

Die Datei "COUNTPRE.DAT" enthält die absolute Position der Referenzmarke für jede Achse und wird von dem Programm "COUNTER" gelesen, falls mit Referenzmarken gearbeitet wird.

EXTLATCH
Dieses Programm erkennt externe Abruf-Signale (L1.X1, L1.X2) und zeigt den abgerufenen Wert am Bildschirm an. Ein externes Abruf-Signal kann über die 4polige Flanschdose (Anschluß X3) durch Brücken von Pin 1 und Pin 4 (Eingang X1) sowie von Pin 2 und Pin 4 (Eingang X2) erzeugt werden.

```
EXTLATCH - External latching of IK 120 using input X3

X3/1        X3/2        X3/3        X3/4
Blue        Green       Red         White
LatchInX1   LatchInX2   LatchOut    0 Volt
Axis X1 =    -0.0001
Axis X1 =     0.0000
Axis X2 =    -0.0005
Axis X2 =    -0.0005
Axis X2 =    -0.0005
```

FREEZE2
Dieses Programm zeigt die Anwendung des Abruf-Zählers in Verbindung mit der Strobe-Funktion zum simultanen Abruf des Meßwertes von zwei Achsen.
Diese Methode sollte verwendet werden, falls keine Verzögerungszeit zwischen dem Meßwert-Abruf für X1 und X2 zulässig ist.

INTERVAL
Dieses Programm zeigt die Anwendung des Abruf-Zählers in Verbindung mit der Treiber-Software für die IK 120 .

Der Abruf-Zähler wird so programmiert, daß jeden Millimeter ein Meßwert-Abruf erfolgt.

```
(* Programming the interval counter using the driver software IK 120.PAS*)
Program INTERVAL;
uses IK120, crt,BigDisp;
var
  Period:      real;
  SubDivision : integer;
  s,sz:        string;
  lastval :    array [0..1] of string;

procedure Display_count( axis,x,y: integer);
var Count:longint;
   Status: byte;
   Val:real;
begin
  Count := 0;
  Status := 0;
  Read_Count(axis,Status,Count);
  Val:= Period*Count/Subdivision;
  Ldisplay(Val, Rounded,x,y,8,4,false,Signal_error(axis),lastval[axis]);
end;
(********************** main program ************************)
BEGIN
   clrscr;
   Writeln('INTERVAL - Demo Program for IK 120');
   Writeln('_____');
   Period := 0.010;  { Grating period in mm }
   Subdivision := 200;
   m_interpol:=I_50;
   Init_Interface(0);
   Init_Interface(1);
   Interpolation(0,m_interpol);
   Interpolation(1,m_interpol);
   Init_Counter (0,RI_reset_start,fourfold,normal,linear);
   Init_Counter (1,RI_reset_start,fourfold,normal,linear);
   Reset_uas(0);
   Reset_uas(1);
   Reset_Status(0);
   Reset_Status(1);

   Latch_Enable(0,Internal);
   Init_Latch(0,C_load,fourfold,20000);
   Init_Latch(0,C_Reset,fourfold,20000);
   Init_Latch(0,RI_Start,fourfold,20000);

   lastval[0]:='XXXXXXXXXX';
   lastval[1]:='XXXXXXXXXX';
   repeat
```

```
    if latched(0,0) then
    begin
       Display_count(0,10,10);
       Display_count(1,10,18);
    end;
  until keypressed;
END.
```

CREFMARK
Dieses Programm zeigt, wie die Funktion "DistanceCodedREFMarks" zur Auswertung von abstandscodierten Referenzmarken verwendet wird.

MEASURE
Dieses Programm zeigt die Anwendung der "Unit" "TIMER.PAS" zum Abrufen von Meßwerten in bestimmten Zeitabständen. Das Zeit-Intervall wird in Millisekunden eingegeben.

Parameter:

Data file:	Die Datei, in der die Daten gespeichert werden
Number of Points:	Anzahl der Meßpunkte pro Achse, max. 5000
Interval:	Zeit-Intervall in Millisekunden
Period:	Signalperiode in mm

Das Programm kann wie folgt aufgerufen werden:

> MEASURE DATA.DAT 1000 2 0.010 0.020

Der Meßvorgang kann mit einem externen Abruf-Signal oder durch Drücken einer Taste auf dem PC gestartet werden.

```
MEASURE - Data aquisition for IK120 at constant time interval

To start measurement :
                      - press any key or
                      - use external latch input
Measurement started.
Storing point :   1000

Measurment complete.

Saving data point  1000   to  data.dat
```

Die Meßergebnisse schreibt das Programm wie folgt in eine ASCII-Datei:

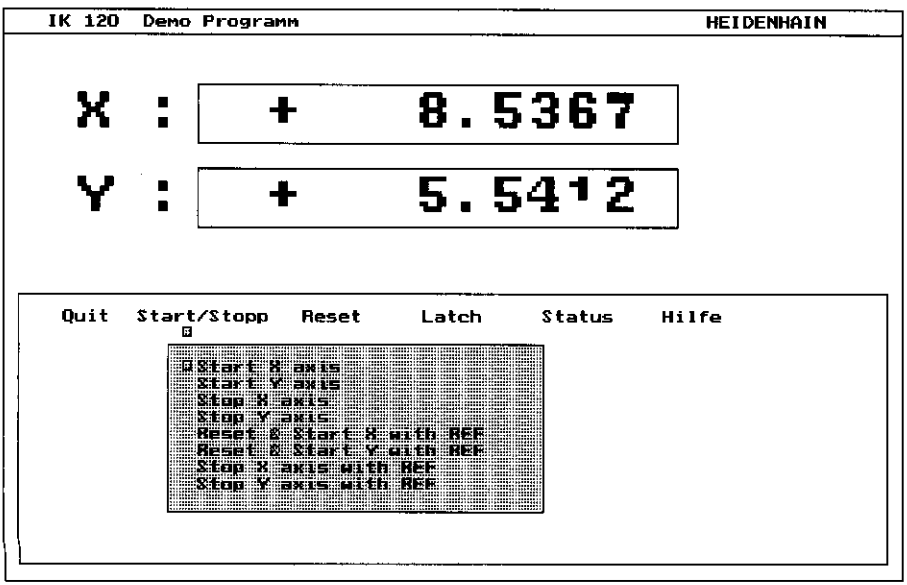
X1	X2	X2-X1
0.1115	0.0007	-0.1108
0.1122	0.0007	-0.1115
0.1127	0.0008	-0.1119
0.1133	0.0007	-0.1126
0.1140	0.0007	-0.1133
0.1146	0.0007	-0.1139
0.1150	0.0007	-0.1143
0.1156	0.0007	-0.1149
0.1163	0.0007	-0.1156

INTERRUPT

Dieses Programm demonstriert den Abruf von Meßwerten mit Hilfe einer Interrupt-Routine. Über ein externes Abruf-Signal wird ein Interrupt ausgelöst, der das gerade laufende Programm unterbricht und zur Interrupt-Routine springt. Die Interrupt-Routine liest den Meßwert und zeigt ihn am Bildschirm an.

DEMO

Mit diesem Programm können verschiedene Funktionen der IK 120 ausgewählt und angezeigt werden.



Weitere Beispiel-Programme in TURBO PASCAL im Verzeichnis „MORE_TP"

Auf der mitgelieferten Diskette sind für die Beispiele der Quell-Code „.PAS" als auch ausführbare Dateien „.EXE" unter dem Verzeichnis „MORE_TP" gespeichert.

INSTALL.EXE hilft Ihnen bei der Inbetriebnahme, zeigt einen Überblick der freien Adressen und wie man die DIP-Schalter einstellt.

IK120.EXE zeigt die Positionswerte beider Achsen und bietet viele weitere Funktionen. Da das Programm dialoggeführt ist, können Sie mit Maus oder Tastatur arbeiten. Weitere Infos finden Sie unter F1.

SAMPLE1.EXE, SAMPLE2.EXE und **SAMPLE3.EXE** sind einfache Programme, die die verschiedenen Einspeicherfunktionen zeigen.

SAMPLE4.EXE zeigt die Basisfunktionen eines Zählerprogramms.

Beispiel-Programme in MICROSOFT C im Verzeichnis „C600"

Auf der mitgelieferten Diskette sind für alle Beispiele sowohl der Quell-Code „.C" als auch ausführbare Dateien „.EXE" unter dem Verzeichnis „C600" gespeichert.
Beispiel-Programme zeigen, wie die Treiber-Softwares „IK 120.H" und „IK 120.C" genutzt werden können.

DEMO2.C

Dieses Programm zeigt die Meßwerte der Eingänge X1 und X2 am Bildschirm an.
Das folgende Beispiel zeigt, wie die Treiber-Software genutzt werden kann.

```
/*################################################################
          demo2.c      Date: 19.03.92
 Function parameter and variables defined in header ik120.h
 Functions are defined in module ik120.c
 Link and compile:  cl /AL /Oaxz /G2 /Fc demo2.c ik120.obj
################################################################ */
#include <stdio.h>
#include <float.h>
#include <ik120.h>
unsigned int  boardseg = 0xc800;      /*   Basesadress IK120    */
unsigned int  boardoff = 0;           /*   Offset boardadress   */
unsigned int  interval = 0;           /*    Latchvalue          */

extern void Init_Interface(counter);
extern void Init_Counter(counter,control,eval,direction,method);
extern unsigned long Soft_Count(counter,latch);
/* ================================================
       Simultaneous latching and reading of both axis
  ================================================ */
void main ()
{
float c_value0,c_value1;
Init_Interface(C_0);
Init_Interface(C_1);
printf ("\n\n\t Counter1\t Counter2\n");
Init_Counter(C_0,C_start,fourfold,normal,linear);
Init_Counter(C_1,C_start,fourfold,normal,linear);

while (!kbhit())
  {
    c_value0 = (float) (signed long) Soft_Count(C_0,L_0) * 0.01 / 200;
    c_value1 = (float) (signed long) Soft_Count(C_1,L_0) * 0.01 / 200;
    printf ("\r\t%12.4f\t%12.4f",c_value0, c_value1);
  }
}
```

DEMOHEX.C
Diese Programm zeigt die Meßwerte der Eingänge X1 und X2 in hexadezimaler Schreibweise am Bildschirm an.

DEMO.C
Dieses Programm zeigt die Meßwerte der Eingänge X1 und X2 am Bildschirm an und überprüft gleichzeitig die Meßsysteme auf Fehler.

Was ist zu tun, wenn's nicht funktioniert?

Wenn die IK 120 nicht funktioniert, helfen Ihnen vielleicht die folgenden Hinweise:

QEMM.SYS, EMM386.SYS

Falls Ihre Datei "CONFIG.SYS" den Aufruf für ein Speicherverwaltungs-Programm, beispielsweise "QEMM.SYS" oder "EMM386.SYS" enthält, dann darf dieses Speicherverwaltungs-Programm nicht den Adreßbereich der IK 120 belegen. Sperren Sie deshalb den Adreßbereich durch Hinzufügen des folgenden Befehls aus:

DEVICE = C:\DOS\EMM386.SYS **X=C800-C840**
oder
DEVICE = C:\QEMM\QEMM386.SYS **X=C800-C840**

Der einzugebende Adreßbereich hängt von der Einstellung der DIP-Schalter auf der IK 120 ab.

WINDOWS

Wenn Sie Programme für die IK 120 unter WINDOWS laufen lassen wollen, dann ergänzen Sie folgenden Befehl in der Datei "C:\WINDOWS\SYSTEM.INI":

[boot.description]
emmexclude=C800-C840

Der einzugebende Adreßbereich hängt von der Einstellung des DIP-Schalters auf der IK 120 ab.

Ist die Segment-Adresse der IK 120 richtig eingestellt?

Falls Ihre IK 120 nicht richtig funktioniert, dann versuchen Sie es mit einer anderen Segment-Adresse. Die Segment-Adresse wird über DIP-Schalter auf der IK 120 eingestellt (siehe "Hardware"). Achten Sie bitte darauf, daß Sie in Ihrem Programm ebenfalls die neue Segment-Adresse benutzen.

Falls Sie mit dem Programm "COUNTER" arbeiten, müssen Sie die geänderte Segment-Adresse in die Datei "COUNTER.DAT" eingeben.

Address of Interface card IK 120 in decimal format not HEX
(C800= 51200) 57344

Einige Beispiele für die dezimale Schreibweise von Segment-Adressen und die dazugehörigen DIP-Schalter-Einstellungen:

Segment-Adresse (Hex)	Segment-Adresse (Dezimal)	DIP-Schalter							
		S1	S2	S3	S4	S5	S6	S7	S8
C8000	51200	ON	ON	OFF	ON	ON	ON	ON	ON
C8400	51264	ON	ON	OFF	ON	ON	ON	ON	OFF
C8800	51328	ON	ON	OFF	ON	ON	ON	OFF	ON
CF000	52992	ON	ON	OFF	OFF	OFF	OFF	ON	ON
D0000	53248	ON	OFF	ON	ON	ON	ON	ON	ON
E0000	57344	OFF	ON	ON	ON	ON	ON	ON	ON

Große Hard-Disc oder andere zusätzliche Speichergeräte (SCSI-Interface)

PCs, die mit großen Hard-Discs oder anderen zusätzlichen Speichergeräten ausgerüstet sind, beispielsweise mit optischen Speichern, benutzen oft das SCSI-Interface. Dieses Interface arbeitet häufig mit dem Adreßbereich C8000, der IK 120 Standard-Einstellung. Versuchen Sie in diesem Fall eine andere Adreßeinstellung für die IK 120.

CONFIG.SYS, AUTOEXEC.BAT, RAMDRIVE.SYS, VDISK.SYS

Entfernen Sie alle unnötigen Befehle aus den Dateien "AUTOEXEC.BAT" und "CONFIG.SYS", insbesondere die Befehle, die im Zusammenhang mit "RAMDRIVE" oder "VDISK" stehen.

Für diese Änderungen benutzen Sie bitte einen ASCII-Editor, beispielsweise EDIT oder EDLIN. Fügen Sie bei jeder Zeile, die Sie nicht benötigen, das Wort **"REM"** hinzu. Dadurch wird dieser Befehl nicht ausgeführt. Später, wenn Sie wissen welcher Befehl Probleme verursacht hat, können Sie einfach durch Entfernen von "REM" die Befehle wieder aktivieren.

Ihr Computer muß nach Änderungen in den Dateien "AUTOEXEC.BAT" oder "CONFIG.SYS" wieder neu gestartet (gebootet) werden. Erst danach sind die Änderungen wirksam.

Beispiel für eine Änderung in der Datei "CONFIG.SYS":

REM DEVICE= C:\DOS\VDISK.SYS

Welcher andere Interface-Adapter benutzt den gleichen Adreßbereich?

Entfernen Sie nacheinander alle nicht benötigten Adapterkarten, wie beispielsweise Netzwerk-Adapter, Scanner-Interface-Karten usw. So können Sie auf einfache Art feststellen, welcher Adapter stört.

Besonders Netzwerk-Adapter benutzen oft den Adreßraum CC000 bis CCFFF und D800 bis DBFF.

Technische Daten

Mechanische Daten	
Abmessungen	340 mm x 107 mm (volle Länge)
Betriebstemperatur	0°C bis 45°C
Lagertemperatur	–30°C bis 70°C

Elektrische Daten	
Eingänge/Ausgänge	
Meßsystem-Eingang:	X1: Achse 1, Sub-D-Anschluß 9polig; X2: Achse 2, Sub-D-Anschluß 9polig; Sinus-Signale: 7 µAss bis 16 µAss Eingangs-Frequenz: max.50 kHz bei 25fach-Interpolation max. 30 kHz bei 50fach-Interpolation Kabellänge: max. 10 m
Externe Abruf-Signale:	X3: Flanschdose, 4polig; zwei Eingänge, ein Ausgang
Meßsystem-Ausgang:	Option: Zusätzlicher Sub-D-Anschluß zur Weiterführung der Meßsystem-Signale des Eingangs X2 Kabellänge: abhängig von der Eingangsschaltung der Folge-Elektronik
Signal-Interpolation	25fach oder 50fach, über Software wählbar
Signal-Auswertung	1-, 2-, 4fach
Achs-Zähler	26 Bit, 2^{-25} bis $2^{+25}-1$ oder –180 000 Grad bis +179 999 Grad
Abruf-Zähler	16 Bit
Adreßbereich	C0000h bis FFFFFh, 1 Kbyte (400h) wird belegt
Interrupts	IRQ3, IRQ4, IRQ5, IRQ7
Leistung	ca. 1 W, max. 1,5 W (ohne Meßsysteme)

Software	
Treiber-Software	in TURBO PASCAL und MICROSOFT C auf Diskette. Programmier-Beispiele für QBASIC in der Benutzer-Anleitung

Stichwortverzeichnis