SHARP

PC-1500

BASIC-ERWEITERUNGEN

INHALTSÜBERSICHT

1. Die Befehlstabelle

Jede BASIC-Tabelle besteht aus zwei Teilen, einem Steuerteil und der Befehlsliste. Diese beiden Teile werden weiter unten erläutert. Doch lassen Sie mich nun dazu Stellung nehmen, in welchen Bereichen eine solche Tabelle angesiedelt werden darf.

1.1. Mögliche Startadressen der BASIC-Tabelle

Der für das System reservierte Bereich erstreckt sich von &8000-&FFFF. Nur in diesem Bereich dürfen BASIC-Tabellen angeordnet werden. Der Bereich von &8000-&BFFF ist zusätzlich mit Hilfe des CPU-Bits PV in zwei gleiche Bereiche von jeweils parallel liegenden 16 KByte aufgegliedert. An folgenden Stellen dürfen BASIC-Tabellen beginnen.

PV gelöscht: &8000, &8800, &9000, &9800, &A000, &A800, &B000, &B800

PV gesetzt: &8000, &8800, &9000, &9800, &A000, &A800, &B000, &B800

Eine Sonderstellung nimmt die bei &C000 beginnende Tabelle ein, die nicht der Kontrolle des PV-Bit unterliegt. Diese und die oben unterstrichenen Startadressen können nicht für eigene Erweiterungen verwendet werden, da sie vom System bereits belegt sind.

1.2. Kodierung des Steuerteils der BASIC-Tabelle

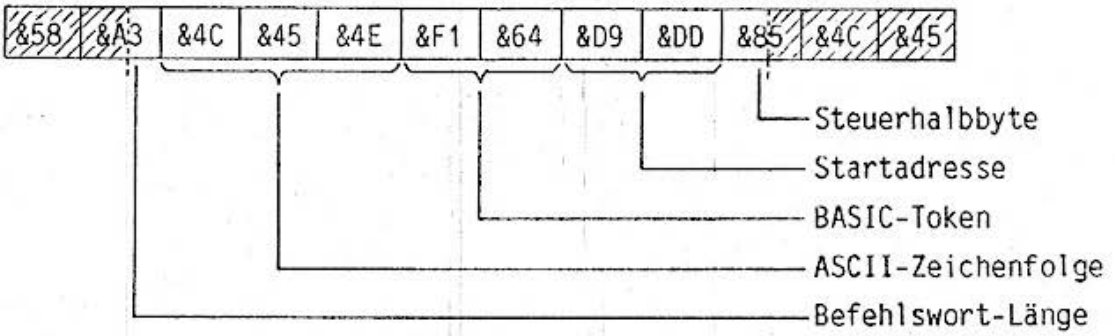
Beachten Sie bitte für die folgenden Erklärungen, daß von nun an das H-Byte der Startadresse mit ## bezeichnet wird, da es sich ja hierbei um einen variablen Parameter handelt.

Der Steuerteil der BASIC-Tabelle erstreckt sich von &##00-&##53.

&##00	&55	Erkennungszeichen, daß BASIC-Tabelle folgt; immer &55.
&##01	&07	Tabellennummer, wobei die Nummern von 1-9 vom System belegt sind und nicht verwendet werden dürfen, sonst beliebig.
&##02-&##09	&41 &42 &0D	Name der Tabelle für den OPN-Befehl. Höchstens 7 Zeichen lang und immer durch &0D abgeschlossen. Im Beispiel "AB"
&##0A-&##0C		Initialisierungsroutine bzw. Platz für Sprung dorthin (BA XXXX). Diese Routine wird bei jedem Prozessor-Reset durchlaufen
&##0D-&##0F		Platz für Sprung zur INPUT#-Routine
&##10-&##12		Platz für Sprung zur PRINT#-Routine
&##13-&##1C	&9A	Nicht verwendete Sprungbefehle; mit &9A füllen.
&##1D-&##1F		Protokoll- bzw. Trace-Vektor (XXXX FF). Bitte mit &C4 &AF &FF füllen.
&##20-&##53		Platz für 26 Buchstabenpointer mit jeweils 2 Byte Länge. Falls kein Befehl mit dem jeweiligen Anfangsbuchstaben vorhanden, mit &0000 füllen. Ansonsten muß der Pointer auf den zweiten Buchstaben des ersten vorkommenden Befehls mit diesem Buchstaben weisen. Die Pointer sind selbstverständlich alphabetisch geordnet.

1.3. Kodierung der Befehlsliste der BASIC-Tabelle

Die Aufzählung der zu einer Tabelle gehörenden Befehle erfolgt ab &##54 nach folgendem Schema. Als Beispiel wurde ein Teil der Hauptbefehlstabelle des PC-1500 verwendet:



Die Befehle müssen nicht komplett alphabetisch geordnet werden, jedoch müssen zuerst alle Befehle mit A, dann die mit B usw. angegeben werden. Falls das Halbbyte nach dem Steuerhalbbyte eines Befehls (Länge des nächsten Befehlswortes) den Wert 0 hat, so zeigt dies das Tabellenende an.

1.4. Erläuterungen zum Steuerhalbbyte

Falls das Bit 0 des Steuerhalbbytes den Wert 1 hat beginnt der nächstfolgende Befehl mit einem anderen Anfangsbuchstaben.
Zur Erklärung der Steuerbits 1-3 diene die folgende Tabelle:

B3	B2	B1	Funktionen (z.B. SIN, COS, TAN)	Prozeduren (z.B. NEW, RESTORE, PRINT)
1	0	0	Funktion benötigt genau einen Stringparameter	erlaubte Modes maskierbar (Siehe unter 1.5.)
1	0	1	Funktion benötigt genau einen numerischen Parameter	nur im laufenden Programm erlaubt (z.B. FOR, END)
1	1	0	Funktion benötigt keinen oder mehrere String- oder numerische Parameter. Anzahl und Art werden nicht überprüft.	in allen Modes lauffähig
1	1	1	nicht verwendet	Plotterbefehle

1.5. Maskierung mit Hilfe des Modebyte

Prozeduren, die nur in bestimmten Modes lauffähig sein sollen,müssen in den Steuerbits 1-3 die Kodierung 100 aufweisen. (Siehe unter 1.4.) Das erste Programmbyte einer solchen Prozedur (Beispiel: bei NEW lautet dieses Byte 00110000) muß diejenigen Bits auf 1 haben, die den erlaubten Modes entsprechen:

B7	B6	B5	B4	B3	B2	B1	B0
0	RUN	PRO	RESERVE	0	0	0	0

Die Startadresse der Prozedur liegt somit um ein Byte höher als bei den maskierbaren Befehlen in der BASIC-Tabelle angegeben.

2. Kodierung des BASIC-Token

Jeder BASIC-Token ist mit 2 Bytes kodiert, die im folgenden erläutert werden sollen.

2.1. Byte 1 des Token

Die möglichen Werte des 1. Bytes sind:

&E0, &E1, &E2, &E3, &E4, &E5, &E6, &E7,
&E8, &E9, &EA, &EB, &EC, &ED, &EE, &EF,
&F0 und &F1

Die Token-Bytes &E0-&EF sowie &F1 steuern automatisch die Suchadresse für die BASIC-Tabelle:

Start bei	&8000	&8800	&9000	&9800	&A000	&A800	&B000	&B800	&C000
PV = 0	&E0	&E1	&E2	&E3	&E4	&E5	&E6	&E7	&F1
PV = 1	&E8	&E9	&EA	&EB	&EC	&ED	&EE	&EF	&F1

Alle BASIC-Befehle, deren Tokens mit einem dieser Bytes beginnen, werden nur in der explizit durch das 1. Tokenbyte vorgegebenen Tabelle gesucht. Ist die Tabelle entweder nicht vorhanden (1. Byte = &55) oder wird der BASIC-Befehl nicht genau in dieser Tabelle gefunden, so erfolgt Fehlerausgabe der Suchprozedur (&1C).

Falls das 1. Tokenbyte &F0 ist, so dient dies zur Benutzung des gleichen BASIC-Befehls in mehreren Peripheriegeräten. So wird z.B. der Befehl LPRINT sowohl vom Drucker CE-150 als auch von der Schnittstelle CE-158 benutzt.
Jede Peripherieeinheit kann gesondert durch einen entsprechenden OPN-Befehl angesprochen werden. Der Printer CE-150 z.B. mit OPN"MGP". Der Rechner durchsucht zunächst die durch den OPN-Befehl vorgegebene Tabelle (Siehe 2.2.). Wird der Befehl jedoch in dieser Tabelle nicht gefunden, so durchsucht der Rechner in folgender Reihenfolge sämtliche BASIC-Tabellen:

&C000/PV=0
&B800-&8000/PV=0 absteigend
&B800-&8000/PV=1 absteigend

Erst, wenn der Befehl in keiner der Tabellen gefunden wurde, erfolgt Rückkehr mit Fehlerflag.

2.2. Der OPN-Befehl

Inhalt OPN-DV (&79D1)	Startadr. der BASIC-Tabelle	PV=1	PV=0	Binärcod.	Implementierung
&60	&C000	+	+	01100000	LCD
&5C	&B800		+	01011100	CMT
&DC	&B800	+		11011100	
&58	&B000		+	01011000	MGP
&D8	&B000	+		11011000	
&54	&A800		+	01010100	—
&D4	&A800	+		11010100	
&50	&A000		+	01010000	—
&D0	&A000	+		11010000	
&4C	&9800		+	01001100	
&CC	&9800	+		11001100	—
&48	&9000		+	01001000	
&C8	&9000	+		11001000	—

&44	&8800		+	01000100	
&C4	&8800	+		11000100	LPRT
&40	&8000		+	01000000	
&C0	&8000	+		11000000	COM

Wird zum Beispiel OPN"LPRT" eingegeben, so wird der Wert &C4 in das Register &79D1 geschrieben.
Bei Befehlen, die mit &F0 beginnen, beginnt der Rechner mit der Suche nach dem Befehl in der Tabelle ab &8800 mit PV=1.

2.3. Byte 2 des Token

Sämtliche Kommandos wie BEEP, FOR, NEXT, etc. haben ein 2. Tokenbyte, das im Bereich &80-&FF liegen muß. Diese Kommandos werden somit nicht durch die Arithmetikprozedur (&DE) behandelt.

Das 2. Tokenbyte für BASIC-Funktionen wie ABS, INT, SIN, etc. muß im Bereich &60-&7F liegen.

Für sämtliche Operatoren wie AND, OR sowie für operandenlose Funktionen wie MEM, TIME muß das 2. Tokenbyte im Bereich &00-&5F liegen.

Eine Mißachtung der im Kapitel 2. besprochenen Dinge führt unweigerlich zum sofortigen Absturz des Rechners.

3. Befehlsausführung

In diesem Kapitel sollen wichtige Voraussetzungen und Regeln zur Abarbeitung von BASIC-Funktionen, -Operanden und -Prozeduren durch den PC-1500 besprochen werden. Im folgenden werden solche BASIC-Worte als Funktionen bezeichnet, deren zweites Tokenbyte im Bereich &60-&7F liegt, als Prozeduren diejenigen, deren zweites Tokenbyte im Bereich &80-&FF liegt und als Operatoren diejenigen, welche ein zweites Tokenbyte im Bereich &00-&5F aufweisen.

3.1. Funktionen und Operanden

3.1.1. Sowohl Funktionen als auch Operanden werden immer als Unterprogramm formuliert, das mit dem RET-Befehl (&9A) endet.

3.1.2. Ist bei Rücksprung aus dem entsprechenden Unterprogramm das UH-Register <> 0, so wird ein Fehler mit der im UH-Register angegebenen Nummer angezeigt. Soll kein Fehler angezeigt werden, so ist das UH-Register vor Rücksprung auf Null zu setzen!

3.1.3. Beim Einsprung in das Unterprogramm enthalten die Pointerregister folgende Werte:
X = Startadresse des Unterprogramms
Y = BASIC-Token der aufgerufenen Funktion
Das U-Register und der Akkumulator enthalten keine verwendbaren Werte.

3.1.4. Liegt das zweite Tokenbyte im Bereich &60-&7F (Funktionen), so werden Operanden,wie im Steuerhalbbyte angegeben, bereitgestellt. Bei Bereitstellung eines Operanden befindet sich dieser im X-Arithmetikregister (&7A00-&7A07). Treten mehrere Operanden auf (Beispiel: MID$-Befehl), so befindet sich lediglich der letzte gefundene Operand im X-Arithmetikregister. Alle anderen werden in der Reihenfolge des Auftretens auf dem BASIC-Stack abgelegt.

3.1.5. Liegt das zweite Tokenbyte im Bereich &00-&5F (Operanden wie AND sowie operandenlose Funktionen wie INKEY$ gehören zu dieser Kategorie), so werden keine Operanden bereitgestellt.

3.2. Prozeduren

3.2.1. Prozeduren sind Maschinenprogramme, die nicht mit RET (&9A) enden.

3.2.2. Ist die Abarbeitung korrekt erfolgt (keine Fehler traten auf), erfolgt der Rücksprung aus der Prozedur mit dem Befehl CALL (&E2). Soll dagegen ein Fehler angezeigt werden, so gibt es hierfür zwei Möglichkeiten: Erfolgt der Rücksprung aus der Prozedur mit CALL (&E4), wird ein Fehler mit der Nummer 1 angezeigt. Falls eine andere Fehlernummer angezeigt werden soll, so muß diese im UH-Register abgelegt werden und die Prozedur danach mit CALL (&E0) verlassen werden.

3.2.3. Beim Einsprung in die Prozedur enthalten die Pointerregister folgende Werte:
X = Startadresse der Funktion
Y = Adresse des Bytes nach dem BASIC-Token der eben aufgerufenen Prozedur. Diese Adresse kann im BASIC-Programmtext oder im Eingabe-Puffer liegen.
Das U-Register und der Akkumulator enthalten keine verwendbaren Werte.

3.2.4. Es werden keinerlei Parameter oder Operanden bereitgestellt. Wie dies zu erledigen ist, erfahren Sie in den nachfolgenden Kapiteln.

4. Parameterübergabe für Prozeduren

Dieses Kapitel behandelt nur die Bereitstellung von Operanden/Parametern für Prozeduren. Die Übergabe bei Funktionen erfolgt nach Maßgabe des Steuerhalbbytes automatisch und wurde im Kapitel 3.1. bereits beschrieben. Beachten Sie bitte, daß für alle nachfolgenden Angaben das Y-Pointerregister immer auf den laufenden Programmtext weisen muß. Bitte berücksichtigen Sie, daß das Y-Register nach Möglichkeit bei anderweitiger Verwendung zunächst auf dem Stack gesichert werden muß. Die im folgenden angegebenen Systemunterprogramme sind nur in sehr kurzer Form beschrieben. Eine ausführliche Beschreibung finden Sie im in derselben Reihe erschienenen Info Nr. 04/83 'UNTERPROGRAMME FUNKTIONSBESCHREIBUNG".

4.1. BCD-Parameter

Ein "Binary Coded Decimal"-Parameter wird durch das Unterprogramm CALL (&DE) in das X-Arithmetikregister (&7A00-&7A07) geschrieben. Beachten Sie bitte hierzu die oben gemachten Angaben zum Y-Register.

4.2. 16 Bit-Hexadezimalzahl

Voraussetzung ist die bereits erfolgte korrekte Vorbereitung eines BCD-Parameters. Im Anschluß daran wird mit Hilfe des Unterprogramms CALL (&D0) der BCD-Parameter in einen 16 Bit-Hexparameter umgewandelt, wobei gleichzeitig ein zulässiger Wertebereich vorgegeben werden kann. Der Hexparameter wird ebenfalls im X-Arithmetikregister abgelegt.

4.3. Zeichenketten

Diese werden durch Aufruf des Unterprogramms CALL (&2E) vorbereitet. Falls dieses Unterprogramm allerdings einen numerischen Parameter vorfindet, so wird dieser bereitgestellt. Die Überprüfung, ob eine Zahl oder eine Zeichenkette gefunden wurde, obliegt also der ausführenden Prozedur. Auch ein durch CALL (&2E) ermittelter Parameter wird im X-Arithmetikregister abgelegt.

4.4. Zeilennummern und -labels

Diese sogenannten zeilenbestimmenden Parameter können durch zwei Unterprogramme bereitgestellt werden: Falls immer (auch bei mehreren mit MERGE zugeladenen Programmen im Rechner) nur die erste vorhandene Zeile mit dieser Nummer/diesem Label angesprochen werden soll und der Parameter mit ENTER (&0D) endet, so wird das Unterprogramm CALL &CC86 verwendet. Mit Hilfe des Unterprogramms CALL (&1A) wird immer nur das gerade in Ausführung befindliche Programm durchsucht. Allerdings muß der Parameter hier zuvor durch Aufruf der Prozeduren CALL (&2E) und CALL (&D0) mit Operanden &83 in dieser Reihenfolge bereitgestellt werden. In beiden Fällen wird, falls Zeile vorhanden, die Adresse der ersten Instruktion in &78A6/7 und die Zeilennummer in &78A8/9 abgelegt.

4.5. Variable

Falls Variable als Parameter verwendet werden sollen, so wird die Startadresse der Variable durch das Unterprogramm CALL (&CE) ermittelt, welches die Startadresse im U-Pointerregister übergibt. Der Inhalt der Variablen kann durch die oben erwähnten Unterprogramme behandelt werden.

5. Ergebnisrückgabe bei Funktionen

5.1. BCD-Zahl

Eine BCD-Zahl wird im X-Arithmetikregister (&7A00-&7A07) in der BCD-Form übergeben, wie im Technical Reference Manual beschrieben.

5.2. 16 Bit-Hexadezimalzahl

Eine solche Zahl muß mit "&B2" kodiert ebenfalls im X-Arithmetikregister abgelegt werden. Die Kodierung finden Sie im Technical Reference Manual.

5.3. Zeichenketten

Mit Hilfe des Unterprogramms CALL &DFB4 wird für die Ergebniszeichenkette Platz im Stringbuffer reserviert. Das Y-Pointerregister enthält dann die Startadresse im Buffer, von wo ab die Zeichenkette danach Y-registerindirekt abgelegt werden kann. Die Zeichenketten-Kodierung muß danach allerdings noch im X-Arithmetikregister abgelegt werden. Dies übernimmt vollautomatisch das Unterprogramm CALL &DFC4, dem man die Länge des gerade gespeicherten Textes im Akku übergibt.

6. Anmerkungen

Im nächsten Kapitel finden Sie Vorschläge für zu implementierende neue BASIC-Befehle, allerdings nicht die zugehörigen Maschinensprache-Routinen.
Letztere werden zu einigen der vorgeschlagenen neuen Befehle im in dieser Reihe erscheinenden Info Nr. 05/84 "BASIC-ERWEITERUNGEN" veröffentlicht und getestet freigegeben.

7. Vorschläge für neue BASIC-Befehle

Die im folgenden gemachten Vorschläge für ein stark erweitertes BASIC sind auf meine Bedürfnisse zugeschnitten und können daher spezielle Wünsche anderer Benutzer nicht berücksichtigen.

7.1. Editierhilfen

DELETE a,b	Löschen aller Zeilen von a bis b
FIND	Suchen von Zeichenketten oder BASIC-Worten
CHANGE	Ändern von Zeichenketten oder BASIC-Worten
RENUMBER a,b	Neunumerierung mit 1. Zeilennummer a Schrittweite b
KEEP	Rettet nach ERROR 44
RESET	Wiederherstellung eines Programms nach NEW
ERASE a,b	Speicherbereich von a bis b mit &00 überschreiben
BOOT a,b,c,d	Übertragen von Bereichen zwischen ME1 und ME0
MEM#	Freier Speicherplatz in ME1

7.2. Mathematische Funktionen und Operanden

HEX$(a)	Umwandlung in Hex-String
BIN$(a)	Umwandlung in Binär-String
ROUND(a,b)	Rundet a auf b Stellen nach dem Komma
MOD	Division Modulo
XOR	Exklusiv-ODER Operator
ENG$(a)	Herstellen eines Exponenten /3
ODD(a)	Prüft, ob Zahl ungerade
FRAC(a)	liefert Nachkommastellen
MIN(a,b)	kleinster Wert zweier Argumente
MAX(a,b)	größter Wert zweier Argumente
SEC(a)	Secans-Funktion
CSC(a)	Cosecans-Funktion
HSIN(a)	Sinus-Hyperbolicus
HCOS(a)	Cosinus-Hyperbolicus
HTAN(a)	Tangens-Hyperbolicus
AHSN(a)	Area-Sinus
AHCS(a)	Area-Cosinus
AHTN(a)	Area-Tangens
ETEN(a)	10^X-Funktion

7.3. Tape-Operationen

FSAVE	schnelles Sichern mit 9600 Baud
FLOAD	schnelles Laden
FCHAIN	schnelle Programmkettung
FMERGE	schnelles Zuladen

7.4. Maschinensprache-Bedienung

MONITOR	Aufruf eines Hexmonitors
ASSEMBLER	Aufruf eines Assembler
DISASSEMB a,b	Disassemblieren von a bis b
XREF	Cross-Reference-Listing

7.5. Sonstiges

GETKEY$	Warten auf nächsten Tastendruck
FAST	Sperrt sämtliche Interrupts mit Ausnahme BREAK
SLOW	Gibt die Interrupts wieder frei
INSTR$(a$,b$)	Sucht b$ in a$ und übergibt die Position, sonst 0
CIRCLE x,y,r,a,b	Zeichnet Kreis oder Kreissegment
ELLIPSE x,y,r,s,a,b	Zeichnet Ellipse oder Ellipsensegment
WIDE a$	Gibt a$ gespreizt auf die Anzeige
FREE	Anzahl freier Bytes unter Variablenberücksichtigung
SWAP v1,v2	Tauschen von Variableninhalten
ERRN	Nummer des letzten Programmfehlers
DATE$	Datum in der Form TT.MM.
TIME$	Uhrzeit in der Form SS:MM:SS
UPCR$(a$)	Umwandlung Klein- in Großbuchstaben
LWCR$(a$)	Umwandlung Groß- in Kleinbuchstaben
VER$	Version des Betriebssystems
POP	Umgeht einen Unterprogrammrücksprung
BYE	Schaltet den Rechner ab
MIDSET$(a$,b$,a)	Setzt b$ in a$ ab Position a ein.

	B7	B6	B5	B4	B3	B2	B1	B0
	80/128	40/064	20/032	10/016	08/008	04/004	02/002	01/001
764E	DEF	I	II	III	SMALL	ｶﾅ	SHIFT	BUSY
764F		RUN	PRO	RESERVE		RAD	G	DE

CP <Op1>,<Op2>

<Op1>	<	<Op2>	Z=0	C=0
<Op1>	=	<Op2>	Z=1	C=1
<Op1>	>	<Op2>	Z=0	C=1

BIT <Op1>,<Op2>

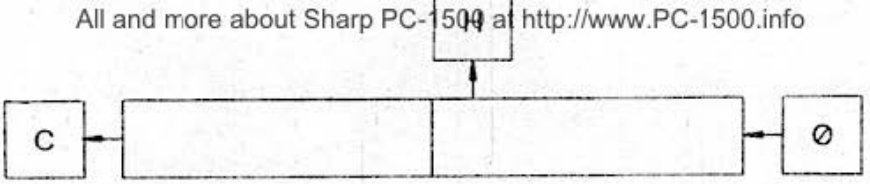
keine Übereinstimmung	Z=1
mindestens eine Übereinstimmung	Z=0

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
7	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A
C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

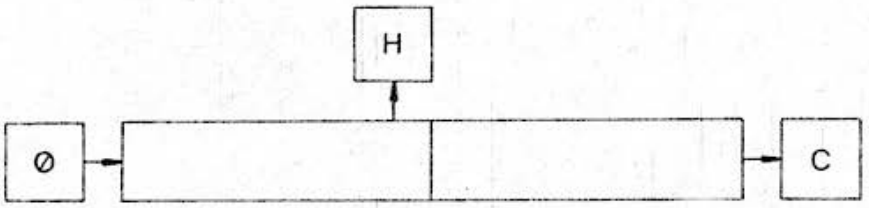
System-Flagregister

Ø	Ø	Ø	H	V	Z	I	C

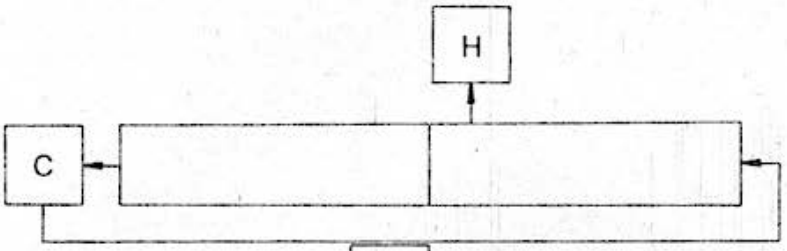
SLA



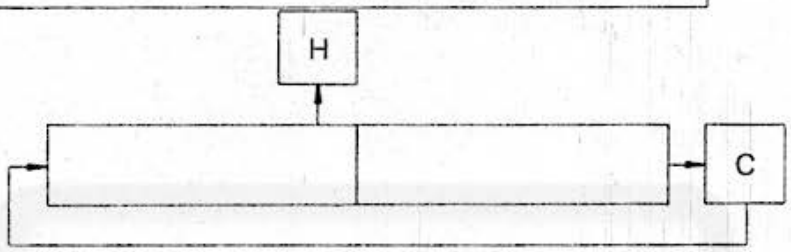
SRA



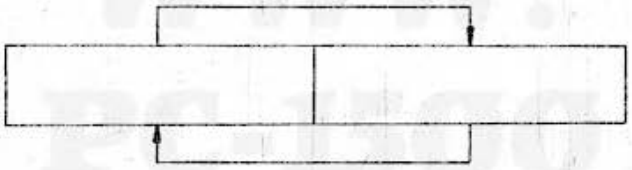
RLA



RRA

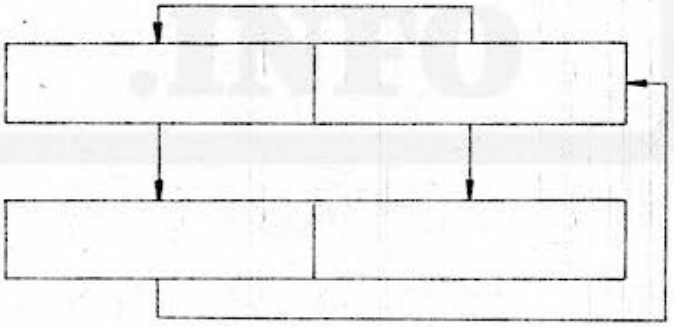


SWP

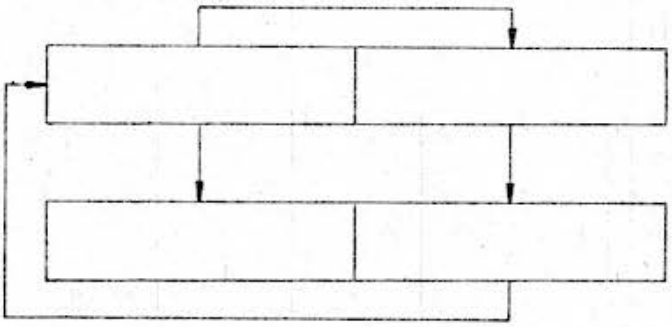


AEX

SLD



SRD



Festlegung der Druckzeichengröße CSIZE beim CE-150:

1. Allgemeines:

   Für den horizontalen Druck, d.h. ROTATEØ bzw. ROTATE2 ist maximal die Zeichengröße 54 erlaubt, da dann die Papierbreite von 216 Einheiten voll ausgenutzt wird.
   Für den vertikalen Druck, d.h. ROTATE1 bzw. ROTATE3 gilt als maximale Zeichengröße der Wert 36.

2. Eingabe der Zeichengrößen:

   Maschinenprogramme bringen den Wert in die Speicherzelle &79F4.
   Bei Benutzung in BASIC-Programmen, können die Zeichengrößen 1..9 auch direkt mit Hilfe des CSIZE-Kommandos eingegeben werden.

3. Abmessungen der entstehenden Zeichen:

Zeichenhöhe:	<Größe>*1.2 mm	<Größe>*6 Einheiten
Zeichenbreite:	<Größe>*Ø.8 mm	<Größe>*4 Einheiten
Zeilenabstand:	<Größe>*Ø.8 mm	<Größe>*4 Einheiten
Zeichenabstand:	<Größe>*Ø.4 mm	<Größe>*2 Einheiten

4. Anmerkungen:

   Nur bis zu Zeichengröße 25 wird der Zeilenvorschub vom Rechner automatisch korrekt vorgenommen. Für größere Zeichen muß der Anwender darauf achten, daß die Zeichen nicht versehentlich übereinander gedruckt werden.

DIRECTION \ LENGHT	NOT END / UP							NOT END / DOWN							END / UP							END / DOWN						
	1	2	3	4	5	6	7	1	2	3	4	5	6	7	1	2	3	4	5	6	7	1	2	3	4	5	6	7
OSTEN	01	02	03	04	05	06	07	41	42	43	44	45	46	47	81	82	83	84	85	86	87	C1	C2	C3	C4	C5	C6	C7
NORDOSTEN	09	0A	0B	0C	0D	0E	0F	49	4A	4B	4C	4D	4E	4F	89	8A	8B	8C	8D	8E	8F	C9	CA	CB	CC	CD	CE	CF
NORDEN	11	12	13	14	15	16	17	51	52	53	54	55	56	57	91	92	93	94	95	96	97	D1	D2	D3	D4	D5	D6	D7
NORDWESTEN	19	1A	1B	1C	1D	1E	1F	59	5A	5B	5C	5D	5E	5F	99	9A	9B	9C	9D	9E	9F	D9	DA	DB	DC	DD	DE	DF
WESTEN	21	22	23	24	25	26	27	61	62	63	64	65	66	67	A1	A2	A3	A4	A5	A6	A7	E1	E2	E3	E4	E5	E6	E7
SUEDWESTEN	29	2A	2B	2C	2D	2E	2F	69	6A	6B	6C	6D	6E	6F	A9	AA	AB	AC	AD	AE	AF	E9	EA	EB	EC	ED	EE	EF
SUEDEN	31	32	33	34	35	36	37	71	72	73	74	75	76	77	B1	B2	B3	B4	B5	B6	B7	F1	F2	F3	F4	F5	F6	F7
SUEDOSTEN	39	3A	3B	3C	3D	3E	3F	79	7A	7B	7C	7D	7E	7F	B9	BA	BB	BC	BD	BE	BF	F9	FA	FB	FC	FD	FE	FF

Bitte beachten Sie, daß die Codes in der Spalte END / UP eventuell nicht die gewünschte Aktion ausführen sondern für Sonderfunktionen benutzt werden. Die A Reihe z.B. wird dazu benutzt, eine ein Strich lange Ecke zu zeichnen mit einer anschließenden Geraden der angegebenen Länge. So kann man z.B. ein O mit vier Befehlen drucken lassen!

PC-1500 ERROR NUMBER TABLE

1 Syntax Error
2 No FOR or GOSUB for NEXT or RETURN
4 No DATA for READ-Instruction
5 Array declared twice
6 Array without DIM
7 Error in type
8 More than two dimensions in Array
9 Index expression out of bounds
10 No more space for variables
11 Label does not exist
12 Illegal format in USING-Instruction
13 No more space for program
14 Basic-Stack overflow by FOR or arithmetic
15 Basic-Stack overflow by GOSUB or mathematic
16 Value out of bounds
17 Type conflict of operands
18 Wrong number of arguments in function
19 Index expression out of bounds
20 Error in variable
21 Variable missed
22 No more space for loading program
23 Time expression error
25 NEW-Command out of bounds
26 Illegal instruction in this mode
27 Peripheral command allowed not yet
28 Keyword not allowed for variable assignment
30 Line number greater than 65535
32 INPUT not possible
34 Error during OPN-Command
36 USING-expression overflow
37 Max-real overflow
38 Division by zero
39 Illegal number expression for function
40 Illegal bounds during CSAVE M-Command
42 Memory overflow during LOAD-Process
43 Error during CLOAD? check
44 Checknumber error
50 Cells empty in CE-158 or switch off
51 Error in SETCOM, SETDEV or TERMINAL
52 Error in received data
53 Illegal TAB designation
58 Contents of received data not appropriate
61 Unmatched header
65 Error in input variables
67 Too much data
69 Error in parallel input port
70 Coordinate bound overflow on printer
71 Paper rewind more than 10.24 cm
72 TAB or LCURSOR expression illegal
73 Illegal instruction in this printer mode
74 Too much points after (R)LINE
76 Expression too great for this line
78 Printer error
79 Illegal position for COLOR-Command
80 Cells empty in CE-150