PHILIPS

P2509

ADVANCED BASIC INTERPRETER
OPERATOR MANUAL

5103 993 20921

Preface

This manual is intended for programmers with some experience in BASIC. It serves as an introduction to the use of Advanced BASIC on the P2000C. This includes the startup procedures explained in chapter 2. Follow the steps in this chapter to create a 'BASIC' system disk, from which BASIC will be automatically loaded and run on startup. Chapter 3 includes technical information about the P2000C system itself, which is important for the use of BASIC. This can be read in conjunction with the supplied BASIC Interpreter Reference Manual.

It is assumed that the user has a working knowledge of BASIC and CP/M, and is familiar with the P2000C microcomputer. Users who are not familiar with the P2000C or CP/M should first read the P2000C operator manual or the CP/M User Guide respectively. Other manuals which provide useful information are:

BASIC Interpreter Reference Manual (5103 993 10921)

P2000C System Reference Manual (5103 993 30421)

CP/M User Guide (103 993 21921)

CP/M Reference Guide (103 993 11921)

C

Contents

TABLE OF CONTENTS

Introduction

1 INTRODUCTION

The Advanced BASIC Interpreter enables you to develop and run programs in BASIC on your P2000C. Whereas most high level languages have to be compiled before execution, BASIC can be executed directly without compilation. The job of the BASIC interpreter is to examine each BASIC statement, and then execute it before going on to the next statement.

Introduction

1.1 ELEMENTS OF THE SYSTEM

The supplied diskette contains the Microsoft BASIC Interpreter Revision 5.21 (Advanced BASIC). This is not a system disk.

Note - The Advanced BASIC Interpreter supports double precision function results, for example, for SIN and COS (refer to the note on page 3-2 of the BASIC Interpreter Reference Manual).

examples:

```
PRINT SIN(3#)   will return a double precision
                function result

PRINT SIN(3)    will return a single precision
                function result
```

Introduction

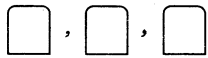
1.2 KEY REPRESENTATION

Sometimes in this manual, it will be necessary to describe a sequence of keys that you have to press. This will be represented as follows:

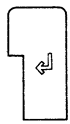
- You should press just one key:

- You should press two (or more) keys simultaneously:

- You should press several keys, one after the other:



- This is the carriage return key. Sometimes within the text it is abbreviated to (CR) or in screen displays to <CR>.



Starting Up BASIC on the P2000C

2 STARTING UP BASIC ON THE P2000C

2.1 PREPARING A BASIC SYSTEM DISK (640K)

1 First of all; please note that this section applies only to the 640K machine. For the 160K machine, start at section 2.1.1 overleaf. With either machine, you will only need to go through this procedure once. The chapter explains how to create a 'BASIC system disk' from the BASIC product disk and the CP/M operating system disk supplied with the machine.

2 To begin, start up CP/M. Slot the system disk carefully into the left-hand drive, (labelled drive'1'), press down the lever, and press the 'reset' button on the front panel of the computer. (If you have not yet made a backup copy of the CP/M disk you should do so by following the steps in chapter 2 of the CP/M User Guide), before going on.

3 The first screen display will appear after a few moments. It gives information about the keyboard, video, disks and printer. The operating system will halt at the standard CP/M prompt 'A>'.

4 Put a new disk in the right-hand drive and press down the lever. Press:

5 When you see the normal prompt 'A>', type in the name 'SYSGEN' and press the carriage return key.

6 SYSGEN is a CP/M program and it has its own prompts, (printed in green below), to which you must respond with the letters printed in black or with carriage return, when it is indicated.

```
A> SYSGEN
SYSGEN VER 2.2
SOURCE DRIVE NAME (OR RETURN TO SKIP) A
SOURCE ON A, THEN TYPE RETURN
FUNCTION COMPLETE
```


```
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)B
DESTINATION ON B, THEN TYPE RETURN
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT)
A>
```

7 You must now run the PIP program. Type 'PIP' and press carriage return.

8 You will be prompted with an asterisk. Type these characters inside the quotation marks: 'B:=A:C*.*[V]' and then press carriage return. The program is now copying the necessary files - you will see their names appearing on the screen.

9 Wait until the copying process has finished and the asterisk is displayed again. Then remove the original CP/M disk from the left-hand drive and replace it with your BASIC product disk.

10 Type 'B:=A:*.*[V]' and press carriage return. The BASIC files will now be copied. When this process is complete the asterisk will be re-displayed. Do not press carriage return. You now have your new BASIC system disk in the right-hand drive. Remove the product disk from the left-hand drive and replace it with the BASIC system disk. Press the 'reset' button to restart the operating system.

2.1.1 Preparing a BASIC System Disk (160K)

Follow steps 1 to 4 above, and then step 7. Once the 'PIP' program has been started, remove the CP/M system disk and replace it with the BASIC product disk. Type these characters inside the quotation marks: 'B:=A:*.*[V]' and then press carriage return. This copies the BASIC interpreter onto the new disk.

When copying has finished and the asterisk prompt reappears, replace the product disk in the left drive with the CP/M system disk. Leave the new 'BASIC system disk' in the right-hand drive. Go to step 2 on page 2.3 and continue from there with the configuration program.

2.2 STARTING UP BASIC ON THE P2000C

1 You are now using a disk that has both the CP/M operating system and BASIC on it, and you are at this point in CP/M. In order to complete the process of starting up BASIC you must instruct CP/M to automatically load and run the BASIC program each time that this disk is started up. To do this you must run the configuration program.

2 Since we are concerned, in this section, only with starting BASIC, the following steps show the prompts displayed by the configuration program and provide suggested responses that will satisfy the requirements of the program. Type them in exactly.

3 To Start the configuration program, type the name of the program after the CP/M prompt: A>CONFIG

Program Section	Prompt	Operator Response
Language Selection	Select or (CR)	Carriage return
Main Menu	Select:	1
System	Select, cursor left, right etc	Carriage return
Capslock	Set capslock on startup (y/n)	y
Timeout	Enter printer timeout	65 then (CR)
Autostart	Enter Autostart string	ABASIC /S:256(CR) (see next page)
Message	Enter welcome message or (CR)	your choice then (CR)
Write System	Enter floppy drive number on which..	1 (if 640K) ? (if 160K)
Confirm	Check disk etc	y
Reset System	Press RESET etc	Press 'reset'

Starting Up BASIC on the P2000C

The full list of parameters that may be entered as the autostart string is as follows:

ABASIC [<filename>] [/F:<#files>] [/M:<high mem>] [/S: <max rec lngth>] and carriage return

The options:

<filename> = if <filename> is present, the BASIC Interpreter proceeds as if a RUN filename command were typed. A default extension of .BAS is used if none is supplied, no "." appears in the filename and the filename is less than 9 characters long.

/F:<#files> = maximum number of disk data files that may be open during the execution of a BASIC program. If this is omitted, the default of 3 files takes effect. For each specified data disk file is allocated 306 bytes of memory for reading information from the disk (when max rec lngth =128). The range for <#files> is 0-15.

/M:<high mem> = highest possible memory address accessible by the BASIC Interpreter (usually used for memory allocation for machine code subroutines). If this option is omitted, high memory is set to the beginning of CP/M

/S:<max rec lngth> : maximum record length in bytes which can be specified in an OPEN statement. If this option is omitted, the default of 128 bytes takes effect.

examples:

```
ABASIC
    <#files>          = 3
    <high mem>        = (start of CP/M).1
    <max rec lngth>   = 128 bytes

ABASIC /F:5 /S:256
    <#files>          = 5
    <high mem>        = (start of CP/M).1
    <max rec lngth>   = 256 bytes

ABASIC PAYROLL /F:5 /M:&HCBFF /S:256
     The program PAYROLL.BAS will be automatically executed.
    <#files>          = 5
    <high mem>        = CBFF Hex
    <max rec lngth>   = 256
```

2.3 ADVANCED BASIC INTERPRETER DISPLAY

The ABASIC Interpreter will now be automatically loaded and run, and will also be run on any subsequent startup. The following lines will be displayed under the CP/M messages, and the system will halt at the standard BASIC prompt 'Ok'.

```
BASIC - 80 Rev. 5.21
(CP/M Version)
Copyright 1977,1978,79,80 by Microsoft
xxxxx bytes free
Ok
```

Note - The number of bytes free depends on your current configuration

3 TECHNICAL INFORMATION

This chapter contains important technical information for programmers. It describes the memory layout of BASIC on the P2000C, and gives notes on the system's treatment of ASCII codes.

3.1 MEMORY LAYOUT

3.1.1 Terminology

The memory layout is described here by hexadecimal addresses and their contents. The following terms are explained first:

Term	Meaning
Jump	A three-byte sequence which directs control to a routine. The first byte (=C3H) is the jump command and the second two bytes contain the hexadecimal address of the routine.
Pointer	A two-byte hexadecimal address.
Flag	A single byte which sets a system mode off or on.

3.1.2 Layout

Address	Contents
0000H	Jump to warm boot (Note that the command sequence 'M = 0: CALL M' has the same effect as the command 'SYSTEM')
0005H	Jump to CP/M (Refer to the CP/M Manuals for more information on CP/M)
0100H	Begin of TPA (Transient Program Area) Start of BASIC Interpreter The TPA is used by the BASIC Interpreter in the following way (starting at 100H): Administration Area: internal BASIC flags and pointers are set in this area. BASIC Interpreter: The administration area and interpreter require approximately 24K memory space. File Buffer Area: The size of this area depends on how many file buffers were allocated on initialization. You can evaluate the size of this area with the VALPTR(#1) command. BASIC Program Area: A BASIC program is written into this area. Variables: Numeric variables are allocated in this area. String pointers are also set here (for more information, see the BASIC Reference manual). Array Variables: Arrays are written into this area. String Values: BASIC dynamically allocates strings in this area. To reorganise this memory space, use the command FRE(""). Stack: The BASIC stack is used for GOSUB, FOR..NEXT and expression evaluation. The stack size can be set by the CLEAR command. The default size is 512 bytes, which is sufficient for normal application programs.

Technical Information

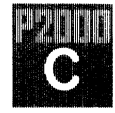
High Memory: The last accessible address for BASIC. This can also be set by the CLEAR command. The default value is (start of CP/M)-1 (see below). Machine code subroutines should be situated above the high memory address (e.g., KSAM80).

(KKKKH)

Beginning of KSAM80 (if loaded)
For 61 K Conf. ..CC00
For 62 K Conf. ..D000
For 63 K Conf. ..D400

Beginning of BDOS
For 61 K Conf. ..E006
For 62 K Conf. ..E406
For 63 K Conf. ..E806

Beginning of CBIOS
For 61 K Conf. ..EE00
For 62 K Conf. ..F200
For 63 K Conf. ..F600



3.2 Access to the Video Controller in BASIC

Information is sent to the screen using the BASIC 'PRINT [USING]' command, (see the BASIC reference manual Chapter 3.7) All commands are sent to the screen as ESCAPE sequences. It is therefore recommended that you use the following constant in all of your programs:

```
ESC$=CHR$(&H1B)
```

You can then use this constant to control screen output. The following are examples of ESCAPE sequences to control screen output:

```
PRINT ESC$;CHR(48);"P"      - This will cause every character
                              after the constant to be printed
                              in inverse video.

PRINT ESC$;CHR$(48);        - This will cause every character
CHR$(&H40)                    after the constant to be printed
                              normally.

PRINT ESC$;"c";             - This will make the cursor invisible
```

Technical Information

```
PRINT ESC$;"Y";CHR$(10+&H20);CHR$(20+&H20);"HELLO"
```

- This ESCAPE sequence will result in the word 'HELLO' being printed at line 11, column 21. It is called a Cursor Control Function. Note that it can be simplified in your program by replacing it with the following sequence of commands:

```
10 DEF FNX$(X,Y)=CHR$(&H1B)+"Y"+CHR$(X+&H20)+CHR$(Y+&H20)
```

- Later on in the program, the cursor can be controlled by using the following statement:

```
20 PRINT FNX$(10,20);"HELLO"
```

3.3 THE KEYBOARD

The codes generated by the keys on the P2000C keyboard may be set up and saved on disk by the 'Keyboard Table Editing Option' of the configuration program. This is fully documented in the CP/M Reference Guide, but it is also a self-explanatory program, and you should readily be able to set the codes to your requirements, using the translation tables contained in this manual, (this chapter pages 3-10).

3.3.1 Keyboard Control Sequences

The key:

combined with different keys generates a control sequence; the different sequences are shown overleaf.

Technical Information

CTRL A	Enters BASIC EDIT mode for the current printed line.
CTRL C	Terminates program execution.
CTRL R	Redisplays the current printed line.
CTRL S	Suspends program execution until any other key is pressed.
CTRL X	Deletes the current printed line.

3.3.2 ASCII Codes

Each ASCII character is represented by one of the 7 bit codes shown in Table 1 (there are 128 possible 7 bit codes). Fourteen of these positions are know as 'national ISO code' positions. In each national version, these positions represent different characters. The national codes are shown in Table 2.

Internally, the P2000C uses an 8 bit code (=256 positions, see Table 3).

Technical Information

POINTS TO NOTE ABOUT ASCII CODES

The PRINT USING [#filenumber] command uses special format statements (refer to the BASIC Reference Manual, Section 3.77). The ASCII backslash (\) is used to format strings. This character does not exist in all national versions because of the national ISO code restrictions (only 14 characters). The backslash must therefore be represented by the same character at the same code position in your national version.

Some national keyboard versions do not have the numeric symbol '#'. The character '£ '(pound sign) must be used instead. For example:

```
A#=1.456778
```

must be represented as follows:

```
A£ =1.456778
```

If you are writing a program for international application, be aware of the characters that will change from version to version.

	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0		P		p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3		3	C	S	c	s
4	EOT	DC4		4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB		7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K		k	
C	FF	FS	,	<	L		l	
D	CR	GS	-	=	M		m	
E	SO	RS	.	>	N		n	
F	SI	US	/	?	O		o	↵

Fig. 3.1 7 Bit ASCII Code Table

Empty spaces represent national ISO code positions.

Technical Information

HEX:	23	24	27	40	5B	5C	5D	5E	5F	60	7B	7C	7D	7E
ASCII/INT.	#	$	'	@	[	\	]	^	_	`	{	|	}	~
UK (special)	£	$	'	@	#	\	|	^	_	`	{	|	}	~
D/A	#	$	'	§	Ä	Ö	Ü	^	_	`	ä	ö	ü	ß
F/B	£	$	'	à	°	ç	§	^	_	`	é	ù	è	¨
I	£	$	'	§	°	ç	é	^	_	ù	à	ò	è	ì
E	#	$	'	@	[	Ñ	]	^	_	`	{	ñ	}	¨
S/SF	#	¤	'	É	Ä	Ö	Å	^	_	é	ä	ö	å	ü
DK/N	#	$	'	@	Æ	Ø	Å	^	_	`	æ	ø	å	~
P	£	$	'	@	Ã	Ç	Õ	^	_	`	ã	ç	õ	~
CH	£	$	'	ç	à	é	è	^	_	`	ä	ö	ü	¨
UK/NL	£	$	'	@	[	\	]	^	_	`	{	|	}	~

Fig. 3.2 National Code Table

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	DLE	SP	0	@	P	`	p	▗	▄	◇	°	≠	—	Ω	ë
1	SOH	DC1	!	1	A	Q	a	q	▖	▙	¡	±	Á	á	Æ	æ
2	STX	DC2	"	2	B	R	b	r	▐	▟	¢	²	´	î	Å	å
3	ETX	DC3	#	3	C	S	c	s	▐	▟	£	³	À	à	a	û
4	EOT	DC4	$	4	D	T	d	t	▚	▙	Õ	x	Â	â	Ô	ô
5	ENQ	NAK	%	5	E	U	e	u	▌	▙	¥	µ	Ä	ä	Ö	ö
6	ACK	SYN	&	6	F	V	f	v	▞	▟	¨	¶	É	é	Ó	ó
7	BEL	ETB	'	7	G	W	g	w	▛	▟	§	õ	È	è	Ò	ò
8	BS	CAN	(	8	H	X	h	x	▐	▟	¤	÷	Ê	ê	Ú	ú
9	HT	EM	)	9	I	Y	i	y	▛	▟	┌	┐	├	┤	ø	ø
A	LF	SUB	*	:	J	Z	j	z	▌	▟	└	┘	┬	┴	+	ı
B	VT	ESC	+	;	K	[	k	{	▛	▟	«	»	Í	í	Ω	ß
C	FF	FS	,	<	L	\	l	|	▜	▀	←	¼	Ì	ì	Ù	ù
D	CR	GS	-	=	M	]	m	}	▚	▙	↑	½	Ï	ï	Ü	ü
E	SO	RS	.	>	N	^	n	~	▟	▟	→	¾	Ç	ç	Ã	ã
F	SI	US	/	?	O	_	o	↵	▛	▉	↓	¿	Ñ	ñ	-	·

Fig. 3.3 Standard P2000 8 Bit Code Table

4 ACCESS TO THE VIDEO TERMINAL

4.1 SCREEN SPECIFICATIONS

8-bit code (national versions) 24 lines/80 characters

Character Mode

- Bidirectional scrolling
- Split-screen capability (partial scroll)
- Three Attribute modes:
  - Manual: Read and write data from/to attribute page (normal memory access).
  - Auto duplicate: The read attribute data will be duplicated.
  - Block mode: For "block moves" such as scrolling, the attribute page will be scrolled automatically.
- Attributes: Underline, Invert, Blink, 4 Intensity levels) and all combinations
- Adjustable TABs
- Teletext graphics
- Text and Attributes back-transfer from screen possible.

High Resolution Graphics Mode

- 2 selectable modes
  - 512 x 252 dots (no attributes)
  - 256 x 252 dots (3 intensity levels + background)

Each dot addressable
Simple vector handling:
- In Cartesian and Polar co-ordinates.
- Combination with character mode (characters: 21 lines/64 characters).

4.2 SCREEN AND SCREEN CODES

The screen is controlled by both single codes and ESCape sequences.

4.2.1 Single Codes

A C T I O N	Code	
Cursor Home	SOH	01H
Cursor Forward	ACK	06H
Cursor Down	LF	0AH
Cursor Up	SUB	1AH
Cursor Back	NAK	15H
Bell-Beep	BEL	07H
Backspace (same as Cursor Back)	BS	08H
TAB	TAB	09H
Clear Screen	FF	0CH
CR	CR	0DH
End of Page (column 80, row 24)	EOT	04H
Reset Terminal	CAN	18H
CAPS_LOCK	SI	0FH
Lock_Keyboard	EM	19H
Unlock Keyboard	STX	02H

4.2.2 Escape Sequences - Set Attribute

SET ATTRIBUTE = ESC,0,b
where b=attribute byte
and 0=numeric zero

An attribute can be set at any time and is valid until a new attribute is selected.

Attributes: Underline - UL
Blink - BL
Invert - INV
4 Intensities
res - reserved

Attribute byte:

	res	in1	UL	INV	res	res	BL	in2
bit:	7	6	5	4	3	2	1	0

Intensities	in1	in2
Quarter bright	0	0
Bold	0	1
Normal	1	0
Half bright	1	1

For example, the ESCape sequence ESC,0,99 (or ESC,0,&H63) would cause screen characters to be produced:

- half brightness
- underlined
- blinking

Note: To execute ESCape sequences it is necessary to use the CHR$(27) code.
The above example could be included in a BASIC program in the following way:

```
PRINT CHR$(27)+"0"+CHR$(&H63)
                or
PRINT CHR$(27)+"0"+CHR$(99)
```

4.2.3 Escape Sequences - Screen Control

A C T I O N		Code	ASCII
Cursor Addressing		ESC,Y,r,c r=row max = 24 c=column max = 80	Y=59H/89
Erase to End of Line		ESC,K	K=4BH/75
Erase to End of Screen		ESC,k	k=6BH/107
Scroll Up one Line		ESC,S	S=53H/83
Scroll Down one Line		ESC,T	T=54H/84
Set TAB at Cursor Position		ESC,I	I=49H/73
Clear TAB at Cursor Position		ESC,G	G=47H/71
Clear all TABs		ESC,g	g=67H/103
Insert Line		ESC,L	L=4CH/76
Delete Line		ESC,l	l=6CH/108
Insert Character at Cursor Position	ON	ESC,Q	Q=51H/81
	OFF	ESC,R	R=52H/82
Delete Character at Cursor Position		ESC,P	P=50H/80
Insert Character Wrap-around	ON	ESC,N	N=4EH/78
	OFF	ESC,R	R=52H/82
Delete Character Wrap-around		ESC,O	O=4FH/79
Back TAB		ESC,i	i=69H/105
Cursor Visible		ESC,C	C=43H/67
Cursor Invisible		ESC,c	
Start Teletext Graphic		ESC,1	1=31H/49
End Teletext Graphic		ESC,2	2=32H/50
Lock Area for Scrolling		ESC,A,n n=number of lines	A=41H/65
Unlock Area from cursor		ESC,a	a=61H/97
Unlock all Areas		ESC,u	u=75H/117
Send Status		ESC,?	?=3FH/63

Access to the Video Terminal

```
Send Text from Cursor Position ESC,$,nn          $=24H/36
                                 nn=number of characters
Send Attributes of Text
            from Cursor        ESC,%,nn          %=25H/37
Load User Program              ESC,p             p=70H/112
                                    (in INTEL HEX format)
End of INTEL HEX Format        ESC,:             :=3AH/58
            (exits loader, normal operation)
Execute User Program           ESC,x             x=78H/120
Load New Keyboard Table        ESC,@             @=40H/64
Load New Screen Table          ESC,!             !=21H/33
Define Caps Lock Key           ESC,+,k,nn        +=2BH/43
                (k=key,nn=upper limit (Hex))
```

4.2.4 Description of Control Codes

Cursor home:	New cursor position is column 1/ row 1
Cursor forward:	Column + 1.
Cursor down:	New line. Scroll if last line or beginning of a locked area.
Cursor up:	One line up. If 1st line then new position is the bottom line.
Cursor back:	Column - 1.
Backspace:	Same as cursor back.
TAB:	Cursor to next TAB position, default every eighth column.
Clear screen:	Erase the whole screen, cursor at home position.
CR:	Cursor at column 1 in current line.
End of page:	New cursor position is column 80 and row 24
Reset terminal:	Initialise hardware and software. After RESET allow 500ms before sending data to terminal.
Lock keyboard:	All keyboard inputs are ignored.
Unlock keyboard:	Keyboard entries are re-enabled.
Reset:	Re-initializes the hardware and software.
Set attribute:	A new attribute is used until the next "Set attribute" command.
Cursor address:	Absolute cursor address with an offset of 20H (ESC,Y,20H,20H is the "home position").

Access to the Video Terminal

Erase to end of line:	Clear all characters including cursor position to column 80.
Erase to end of screen:	Same as "Erase to end of line" to column 80, row 24.
Scroll up 1 line:	Scroll up the whole screen or area and clear last line.
Scroll down 1 line	Same as above, but scroll down.
Set TAB:	Sets a new TAB position at cursor position.
Clear TAB:	A TAB position is removed at cursor position.
Clear all TABs:	Removes all TABs.
Insert line:	Scroll down the lines from cursor line + 1, and clear cursor line.
Delete line:	Scroll up the lines from cursor line +1 to cursor line.
Insert ON:	The next character will be inserted at cursor position, the last character in the line will be lost.
Insert wrap-around:	Same as "Insert" but the last character of the screen is lost
Insert OFF:	Exit insert mode, normal overwrite.
Delete character:	Delete character at cursor position, the last character in the line will be blank.
Delete character wraparound:	Same as "Delete character" but last character of screen will be blank.

Access to the Video Terminal

Back TAB:	New cursor position is previous TAB position.
Cursor visible:	Display cursor.
Cursor invisible:	Do not display cursor.
Start Teletext graphic:	All characters between 20H & 3FH, and 60H & 7FH are interpreted as teletext characters.
End Teletext graphic:	Normal character mode.
Lock area:	From cursor line, n lines will be locked. (This area will not be scrolled by Cursor up and down, only by explicit "Scroll up (down)" command.
Unlock area:	The cursor position area is unlocked.
Unlock all:	Normal screen status.
Send status:	12 bytes terminal status information will be sent (see STATUS INFORMATION).
Send text:	nn characters from cursor position will be sent back.
Send attribute:	nn attribute bytes from cursor position will be sent.
Load user PGM:	Starts INTEL HEX FORMAT loader for down-loading a machine code program in INTEL HEX format.
End loader:	Exits the loader, enables normal operation.
Execute PGM:	Calls a previously loaded program.
Load new keyboard table:	A new keyboard table (national version) will be downloaded. It consists of 4 sub-tables (NORMAL, SHIFT, SUPER SHIFT, SUPER SHIFT-SHIFT) (see STANDARD KEY TABLE).
Load new screen table:	A new screen translation table will be loaded.
Define Caps Lock Key:	The key K is the new caps lock key. There is no default key.

4.2.5 Graphic Control

ACTION	Code	ASCII
Start high resolution mode 1 (256 x 252)	ESC,5	5=35H/53
Start high resolution mode 2 (512 x 252)	ESC,3	3=33H/51
Start character mode (end graphics)	ESC,4	4=34H/52
CARTESIAN CO-ORDINATES:		
Clear dot	ESC,d,xy	d=64H/100
Set dot	ESC,D,xy	D=44H/68
Move to	ESC,m,xy	m=6DH/109
Draw to	ESC,M,xy	M=4DH/77
Clear to	ESC,v,xy	v=76H/118
POLAR CO-ORDINATES:		
Set origin	ESC,z,xy	z=7AH/122
Move to	ESC,y,Aabs	y=79H/121
Draw to	ESC,U,Aabs	U=55H/85
Clear to	ESC,w,Aabs	w=77H/119
Set dot	ESC,F,Aabs	F=46H/70
Clear dot	ESC,f,Aabs	f=66H/102
Receive picture	ESC,r,xy,nn nn=number of bytes	r=72H/114
Send picture	ESC,t,xy,nn	t=74H/116

Note: xy = co-ordinates: 2 bytes in mode 1
3 bytes in mode 2
(low 'x' byte first, i.e., x(low), x(high), y

Aabs = A = angle ALPHA (2 bytes)
abs = absolute value (2 bytes)

For sending and receiving picture function, a byte represents the contents of the video RAM (like a dump).

Co-ordinates: x is the horizontal co-ordinate, y the vertical co-ordinate x=0, y=0 is bottom leftmost dot on the screen.

Angle ALPHA in steps of one degree (0 to 360)

4.2.6 Description of Graphic Commands

Start mode 1:	Start 256 x 252 resolution mode with 3 intensity levels. The change of this level is done by "Set attribute" command.
Start mode 2:	Start the 512 x 252 resolution mode.
Start character mode:	Exit high resolution graphic mode 1 or 2.

After a start graphic mode command the text on the screen is not cleared but will be re-arranged as 21 lines of 64 characters. The graphic screen will be cleared and the internal cursor and origin set to zero. The start character mode command (exit graphics mode) will clear the text buffer and graphics screen and set the internal cursor and origin field to zero.

CARTESIAN COORDINATES

Clear dot:	Erase pixel at screen position xy.
Set dot:	Set a pixel at xy.
Move to:	Set the internal cursor field to xy.
Draw to:	Draw a line from internal cursor field to xy, and set the internal cursor field to xy.
Clear to:	Same as "Draw to" but the line is erased.

POLAR COORDINATES

Set origin:	Set the internal origin field to xy.
Clear dot:	Erase a pixel at A (angle), abs (absolute value) according to the origin.
Set dot:	Same, but set the pixel.
Move to:	Set the internal cursor field to the calculated (using A abs, origin) value xy.
Draw to:	Draw a line to the calculated value xy and set internal cursor field to xy.

Note: Any combination of POLAR and CARTESIAN coordinates is possible!

4.3 STATUS INFORMATION

When the 'send status' code sequence ESC, ? (1BH,3FH) is sent to the terminal, the terminal status information is returned in the form of a 12 byte string, as shown below:

Byte Number		Contents
1		cursor position - column
2		cursor position - row
3		character at cursor position
4		status flag
	Bit:	
	0	1 = graphics on
	1	1 = graphic mode 2
	2	1 = teletext on
	3	1 = insert mode on
	4	1 = insert wraparound on
	5	1 = keyboard is locked
	6	reserved
	7	reserved
		internal graphic cursor field
5,6		x co-ordinate
7		y co-ordinate
8,9		free space pointer (beginning of RAM area for user program)
10,11,12		reserved for future use

The twelve bytes must be read via a normal 'keyboard read' instruction.

Example Error Message Definition

A EXAMPLE ERROR MESSAGE DEFINITION

The following is an example error message definition

```
10 CLEAR,&HCAFF,512        ' Beware not to erase other program
   parts !
20 PRINT CHR$(12)
30 ' The actual message will be displayed on line 23
   (ESC Y 22H 00H) and
32 ' inverse (ESC O P). A beep will also be activated (07).
40 A$="PRINTER ERROR.  CR  TO RETRY,  SPACE  TO IGNORE:"
50 A$=CHR$(7)+CHR$(&H1B)+"Y"+CHR$(&H20+22)+CHR$(&H20)
   +CHR$(&H1B)+"OP"+A$+CHR$(&H1B)+"O"+CHR$(&H40)
55 'Set the video write parameters
60 A$=CHR$(5)+CHR$(&H31)+CHR$(0)+CHR$(LEN(A$))+A$
65 ' POKE into area reserved for the message and set the
   pointer at 1EH
70 FOR I=1 TO LEN(A$):POKE &HCAFF+I,ASC(MID$(A$,I,1)):NEXT
80 POKE &H1E,&H0
90 POKE &H1F,&HCB
100 LPRINT                 ' Test
```

Index

INDEX

Manual Status Control Form

ADVANCED BASIC INTERPRETER OPERATOR MANUAL

12NC: 5103 992 20921

This issue comprises the following updates:

No updates

Manual Comment Form

ADVANCED BASIC INTERPRETER OPERATOR MANUAL

12 NC: 5103 992 20921

Including Update(s) No updates

Originator:

Name .....
Address.....
.....
.....
.....

Comment (if possible, add a copy of the page(s) affected by the comment, marked with the proposed changes).

Please return this form to: Österr. Philips Ind. GesmbH
MAG-PERCO / MARKETING SUPPORT
A-1100 Wien, Triesterstr. 64,
A U S T R I A