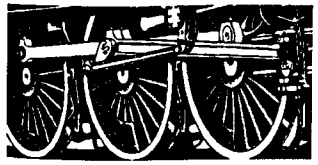


LOCOMOTIVE
SOFTWARE

Defining Character Sets

- **Extra Character Sets for an external printer**
  – using CHARKIT

  How to define new Character Set files to support extra Character Sets, fonts and printwheels on any external printer.

- **Extra Character Sets for the PCW9512 printer**
  – using MKWHEEL

  Caters for the special case of additional wheels for the PCW9512's built-in daisy-wheel printer – in the unlikely case of a printwheel being supplied for this printer which is not supported by the files on Disc 4: the Printwheels Disc.

- **Appendices**

  LocoScript Character Names; Standard Substitutions; Formal Definition of CHARKIT Character Definition files; and Troubleshooting

Note: The Character Set files created using CHARKIT and MKWHEEL will help you support printwheels and fonts on your printer. They won't enable you to select facilities (eg. 15-pitch) that are available on your printer: which of these facilities are available to you depends on the Printer Driver file you are using. The PCW External Printers Guide will tell you if you are using the Printer Driver we currently recommend for your printer.

Oct 1990

Written by Jean Gilmour, Locomotive Software
Produced and typeset electronically by Locomotive Software Ltd
Printed by Ashford Colour Press, Gosport, Hants

Published by Locomotive Software Ltd
Dorking
Surrey RH4 1YL

2nd Edition: First Published October 1990
ISBN 1 85195 046 X

Conventions

The following conventions are used in this booklet:

- Italic (slanted) text is used for descriptions of the type of information that is required, rather than the information itself. For example, *number* could be 1 or 2 or 3 etc. If the description of a single item is more than one word long, the words are joined by hyphens.
- `Text in this style` is used to indicate something to type or something displayed on the screen.
- Slanting brackets are placed round items that are optional.
- The character ↵ is used to represent a carriage return at the end of a line.

Extra Character Sets for external printers – the CHARKIT program

The CHARKIT program allows you to define new Character Sets for an external printer (ie. a printer used as an alternative to the PCW's built-in printer), describing the characters which this printer can print, for example, when a particular printwheel is fitted. (You can't, of course, print any characters other than the ones in the printer's own character set or on the printwheel you are using.) The new Character Sets can then be used as alternatives to the Character Sets provided in the supplied Printer Files. *(If you need to support an additional printwheel for the PCW9512's built-in printer, you need a different program – the MKWHEEL program – described on page 26.)*

The definition of a Character Set takes the form of a simple text file (which may be produced using LocoScript), from which the CHARKIT program extracts the information required and transforms it into the form required by LocoScript.

Before you use this program and the Printer Files on this disc, it is essential:

- to read the following background information, and
- to be using LocoScript 2 version 2.12 or later

The latest version of LocoScript 2 is supplied as part of the Printer Support Pack. If you are not using the latest version of LocoScript, you should create a new Start-of-day disc before you go any further: how to do this is described in the appropriate "Update Information" booklet supplied in this pack.

Background information

When LocoScript prints a document, it sends a long list of codes to the printer which the printer's own software decides how to interpret. These codes are simply numbers, in the range 0...255. For many reasons we find it easier to consider these values in hexadecimal (base 16), so the codes are numbers in the range &00...&FF – where the '&' signals that a hexadecimal number follows. (There is an introduction to hexadecimal numbers on page 11, alongside the description of the 'File body'.)

Some codes represent characters; others are instructions to the printer to carry out actions such as moving to the beginning of the next line. Some of the codes work together as a group to carry out further actions, such as changing the character pitch or turning on or off a print effect like Bold. These groups of codes are usually called 'Escape sequences' because the first code in the group is conventionally the code known as Escape.

Ultimately, what each code represents depends on the software that is interpreting it. Fortunately, a number of conventions have been established. In particular, the codes &00...&7F usually follow the American ASCII standard in which &00...&1F, and &7F are interpreted as instructions to carry out actions, while &20....&7E represent the letters A...Z, a...z, 0...9 and the common punctuation marks. In addition, many dot-matrix printers use the same characters and codes as major manufacturers like IBM or Epson.

Outside of these conventions, there are few universal standards and each printer has its own way of interpreting the codes it receives. In addition, ASCII itself comes in a number of different 'language' variants in which some of the codes have different meanings. For example, UK ASCII differs from US ASCII in that &23 represents £ rather than #.

On top of this, dot-matrix printers and laser printers often support more than one set of characters, which can be selected either by setting switches on the printer itself or by sending special sequences of codes. For example, on the Amstrad DMP printers you can use either technique to choose between variants of the Epson and IBM character sets. In addition, it is often possible to select a different 'language' in both these ways – which also affects the characters available and/or the codes associated with some of the characters.

For daisy-wheel printers the issue is further complicated by the multiplicity of printwheels available, each with its own selection of characters.

To cope with the variety of characters and codes in existence, LocoScript 2 records the details of each particular printer and wheel or printer and font combination as a separate 'Character Set'. Each Character set is essentially a list of the LocoScript characters which the printer is capable of reproducing, how wide they are (for proportional spacing) and how to invoke them. One Character Set for each printer is recorded in the Printer Driver file; the others are recorded in separate Character Set files each containing details of one Character Set. Each Printer Driver file is designed to cope not just with the printer whose name it takes but also as many printers compatible with it as possible, so the Character Set it contains is generally a 'lowest common denominator'.

Most dot-matrix printers provide the same characters as either an FX80 or an IBM printer, and in the majority of cases, the Character Set within the printer's .PRI file will cover the appropriate range of characters. However, while printers provide the same characters, there is a wide variation in their proportional spacing widths. Using the CHARKIT program you can produce a Character Set file containing the correct proportional spacing widths for your printer. However, in general we do not recommend proportional spacing on matrix printers unless this printer supports micro-justification, because we cannot guarantee Justification and Right Alignment.

The characters provided by a daisy-wheel printer depend on what printwheel is fitted. There is a huge diversity of wheels available, and it is here that alternative Character Sets and the CHARKIT program really come into their own. It is not necessary to have a Character Set for every printwheel, only one for each 'family' of wheels. For example, the JUKI 6100 printers often use Triumph Adler Group 2 printwheels, including Primus 10, Caroll 10, Helen 12, Madeleine PS, Tile PS and many others. All these wheels have the same characters on them, and the PS wheels use the same character widths. They can all, therefore, be used with the JUKI6100.#02 Character Set.

The name and pitch which identifies each wheel within a 'family' is known to LocoScript as the Character Style. The pitch portion of the Character Style is most important, since it tells LocoScript how wide the Underline character is and therefore affects underlining!

The function of the CHARKIT program is to take a text description of a Character Set, and produce from it a *printer.#xx* file. The bulk of this text description is simply a list of the LocoScript characters which the printer can reproduce, what code (or codes) must be sent to invoke each character, and its width if proportional spacing is used. The first part of the text description gives the Character Set file's Identity Text, the name of the Character Set, and other such information.

The CHARKIT program

The CHARKIT program on Disc 1 in the Printer Support Pack allows you to specify a new Character Set. The program runs under CP/M. It reads one file containing a definition of the Character Set (the Character Definition file) and produces another file (a *printer.#xx* file) for use by LocoScript.

The Character Definition file is a simple text file, which may be produced using LocoScript. The 'Make ASCII file' command in the Disc Manager's f1 Actions menu will convert a LocoScript document into a file which CHARKIT can read.

There are four stages to using the CHARKIT program:

- establishing the information that you need to include in the Character Definition file ie. the characters you can print, the codes the printer needs in order to print these characters and their widths (for printing the characters proportionally-spaced)
- preparing the Character Definition file (typically by using LocoScript to edit an existing Character Definition) and then making an ASCII version of this document
- creating a Printer File from Character Definition file – by running CHARKIT
- installing and using the new Printer File on your system

In the following instructions, we first describe how to set up and use the simplest sort of Character Definition file – containing only characters which require single printer codes and omitting all character width information. Later we will include character widths and show how sequences of printer codes may be used to access further characters.

Before you start preparing a Character Definition file for yourself, we suggest that you print out CHARKIT.EG on Printer Support Disc 1. CHARKIT.EG is a LocoScript version of an example Character Definition file, which is useful to look at as you read through the following instructions.

The files

The following files on Printer Support Disc 1 are connected with CHARKIT:

Group 0 – CHARKIT

CHARKIT.COM	the CHARKIT program
DAISY.BAS	the program that prints out the contents of wheels
ALLCHARS	a document containing all the LocoScript 2 Characters
TEXTCHAR	a document containing all the LocoScript 2 Characters, plus their LocoScript character names

Group 1 – SIMPLE

CHARKIT.EG	Example simple Character Definition file (LocoScript document)
TEMPLATE.STD	General-purpose basis for preparing Character Definition files

Group 2 – MATRIX

FX.EG IBM.EG	Example Character Definitions for dot-matrix printers (LocoScript documents)
TEMPLATE.STD	Suitable basis for setting up Character Definition files for IBM character sets on dot-matrix printers

Group 3 – PETAL96

PETAL.96	Example Character Definition for 96-petal printwheel (LocoScript document)
TEMPLATE.STD	Template for 96-petal Character Definition files

Group 4 – PETAL100

PETAL.100	Example Character Definition for 100-petal printwheel (LocoScript document)
TEMPLATE.STD	Template for 100-petal Character Definition files

We strongly advise you to prepare a copy of Printer Support Disc 1 (hereinafter called your CHARKIT disc), erase the contents of groups 6 and 7 from this copy (to give yourself some room) and then to do all your work with this copy. Treat the supplied disc as your Master copy, to use only to make a replacement working disc if your copy becomes damaged. (How to copy a disc is described in Session 7 of the LocoScript Tutorial.)

Establishing the information

The bulk of the information that goes into the Character Definition file is a list of all the characters you can print and the code (or codes) to be sent to the printer to make it print these characters. These character details may be given in the printer manual in a number of different forms:

I: If you are using a dot-matrix printer, the character details will probably be presented as a table showing the shape of the character and the corresponding hexadecimal or decimal code.

Generally each character that can be printed will require a single printer code. However, some printers support a number of language variants, in which the meaning of a number of codes depends on the language selected. If the different language variants can be selected by sending escape sequences to the printer, then you can make use of any additional characters they provide. (The technique used to select these characters is described in the section on 'Advanced techniques'.)

In most cases, the printer's manual will in any case tell you whether the character set provides the same characters as an Epson FX80 printer or as an IBM printer. This in turn tells you whether you should base your Character Definition on the supplied FX.EG character definition or IBM.EG one.

II: If you are using a daisy-wheel printer, its manual may give the 'printwheel' table it uses.

This will normally list the characters on the printwheel generally shipped with the printer, and the codes used to access them. It may also note where there may be different characters on other printwheels. Some daisy-wheel printers come with tables for a variety of printwheels: if you are lucky, the wheel you wish to use will be one of them!

• Don't worry about making absolutely sure that the information you've obtained at this stage is full and correct. If testing the Character Set that is produced shows up any errors, these can easily be corrected by editing the Character Definition file and then using CHARKIT to produce a new version of the Character Set file.

Generating character information (daisy-wheel printers only)

If your printer manual doesn't supply details of a particular printwheel, you should be able to generate the character information you require by using the DAISY.BAS program provided on your CHARKIT disc.

This program produces a table of the characters on the wheel and the codes used for these characters by your printer. Alongside each entry, the program prints short sequences of the character, printed assuming a PS width of 7, 6, 5, 4, and 3 PS units (1/60") respectively: these can be used to determine the correct PS width for each character on the wheel (see page 16).

Running the program

1 Connect your printer to your PCW.

On a PCW9512, printers fitted with an IBM-type Centronics connector may be attached to the 'Parallel printer' socket on the back of the machine. Otherwise, parallel printers may be attached either to this socket by an appropriate IBM-Centronics cable or to the Parallel connector on a Serial/Parallel Interface connected to the PCW's Expansion slot. Serial printers need to be attached to the serial connector on such an interface.

2 Fit the printwheel for which you want to produce the character information in the printer.

3 Load CP/M from your CP/M disc. Leave the CP/M disc in the drive after it has loaded.

The steps used to load CP/M are given in your PCW's manual.

4 Copy BASIC.COM to Drive M (the Memory disc) by typing the command:

```
PIP M:=BASIC.COM [RETURN]
```

5 Use the appropriate DEVICE command to set up CP/M to send printer output to the connector to which you have attached your printer.

- If the printer is connected to the parallel connector on the interface, type the command:

```
DEVICE LST:=CEN [RETURN]
```

- If the printer is connected to the serial connector on the interface, first use the SETSIO command to set the baud rate, parity, protocol etc. that your printer requires (as described in your CP/M User Guide); then type the command:

```
DEVICE LST:=SIO [RETURN]
```

- If the printer is connected to the Parallel printer socket on the back of a PCW9512, type the command:

```
DEVICE LST:=PAR [RETURN]
```

Note: Details of the DEVICE command and of the SETSIO command are given in the CP/M section of your PCW's manual.

6 Load some paper into your printer – either continuous stationery or a sheet of A4.

7 Insert your CHARKIT disc and press [ALT] C

8 Type `M:BASIC DAISY` [RETURN]

contd.

9 The program then asks you to identify the type of daisy-wheel printer you have:

D630 – a Diablo-type 96 Petal printer (ie. can be used with one of the following Printer Drivers: D630.PRI; D1610.PRI; IF50.PRI; IF60AX.PRI; IF60CE.PRI)

QUME – Qume-type 96 Petal printer (ie. can be used with one of the following Printer Drivers: QUME.PRI; GAKKEN.PRI)

JUKI6100 – a Juki 100 Petal printer or compatible (ie. can be used with the JUKI6100.PRI Printer Driver)

Select the option for the printer that most closely matches your printer. The program will then run – first resetting the printer and then printing a two-page table giving the character set you require.

(If the printer is off-line when the program tries to reset it, you will see a message of the form '*device* `not ready - Retry, Ignore or Cancel` '. If this appears, simply check that the printer has paper in it, set it on-line, and then type `R` to tell the program to re-try. A message of this type will also appear you specified the wrong device in your DEVICE command. **Note:** This message may also appear while the table is being printed. In this case, it indicates that the program is sending information to the printer faster than the printer can handle it. Give the printer a chance to catch up and then type `R`.)

The program pauses at the end of the first page to allow you to load fresh paper: When you are ready for it to print the second page, press the Space bar. **Note:** The message appears when the program has finished sending the first page to the printer: at this point, your printer will probably have some lines of the first page still to print and you should wait a few moments for it to finish before loading the next sheet of paper. Of course, if you are printing on continuous stationery, you can press the Space bar straight away.

The structure of the table is as follows:

Column 1	The value of code that the printer uses to access the character, written in the form specified for the CHARKIT program.
Column 2	The form of the code we recommend in the Character Definition file – single character, ! followed by single character or !'&*xx*'. *Note:* This column will be affected by the printwheel you are using: always refer to the table on page 12 for the correct code to use.
Column 3	The character accessed by the code.
Columns 4–8	Samples of the character printed assuming PS widths of 7, 6, 5, 4 and 3 PS units, respectively.

The lines of the table are in code order and are grouped in eights. This structure is also used in the Character Definition files supplied on the disc and in the Character Definition file templates, so that you can readily see how to insert the information in the table into your Character Definition file.

∞ *When the program is finished, you remain in BASIC. If you want to produce similar information about another printwheel for this printer, simply replace the printwheel in the printer and then type the command:* `RUN` [RETURN]. *To return to CP/M, type the command:* `SYSTEM` [RETURN]. *If when you return to CP/M, you want to use the built-in printer, follow this command by typing:*

`DEVICE LST:=LPT` [RETURN]

Creating the Character Definition file

The Character Definition file you need to prepare is essentially just a simple text file containing details about the Character Set file CHARKIT will make, followed by a list of printer codes and character names. However, the information it contains must be set out correctly according to a fixed set of rules if CHARKIT is to work correctly.

Each Character Definition file has two main sections:

- the 'Header' containing the basic file details
- the 'File body' containing the table of characters, codes and character widths

In this section we look at these two sections in the simplest type of Character Definition file – ie. one without either character width information or subtle methods of printing additional characters. The details given here apply to all Character Definition files.

As the Character Definition file is a simple text file, you can use just about any text editor or word processing program to prepare it as long as your final version is a simple text file free of any special commands and codes. You could, for example, use the RPED editor supplied on your CP/M disc provided the definition doesn't exceed 200 screen lines: the files this handles are all simple text files. However, we recommend using LocoScript to prepare the file in the first instance and then using the `Make ASCII file` option in the Disc Manager's f1 Actions menu to create a simple text file you need. Not only will you find the file easier to edit with LocoScript (and there is no limit on its size) but we have set up the templates on your CHARKIT disc in such a way that you can create new documents with most of the information you need already in them.

In these instructions, we assume that you will be using LocoScript.

Notes: (i) Lines of the Character Definition file can be up to 255 characters long, if necessary wrapping from one screen line onto the next. The end of each line of the file is marked by a carriage return (↵).

(ii) Any number of comments can be inserted into this file, either at the ends of lines containing character definitions or on separate lines interspersed between the definition lines. These comments have to start with a semicolon (;) but otherwise can contain any text provided the total length of the line (up to the carriage return) doesn't exceed 255 characters. Their role is to make the file readable. We don't include any comments below so that the central structure of the Character Definition file is not obscured. You can see them in action in the example Character Definition file CHARKIT.EG.

Initial steps

Note: The first time you work through these instructions, skip these initial steps. Instead edit the document CHARKIT.EG in group 1. This gives you a ready-made example document to compare with the following description.

1 Load LocoScript 2: then insert your CHARKIT disc.

2 Create a new document in the appropriate group of your CHARKIT disc and give it an appropriate name for the Character set you want to support.

For example, if you want to support 'Bilingual' wheels you might call the new document BILNGUAL.WHL.

In general, we recommend you to create your new document in the 'SIMPLE' group: this will give you a suitable basis for either a simple set of character definitions or for a more complex Character Definition file. However:

- if you have a dot-matrix printer and the character set you want to define is a version of the IBM character set, create your new document in the MATRIX group
- if you have a dot-matrix printer and the character set you want to define is a version of the Epson FX-80 character set, create your new document by copying the FX80.EG document in the MATRIX group
- if you have a 96-petal printer (eg. a Qume Sprint), create your new document in the PETAL96 group
- if you have a 100-petal printer (eg. a Juki 6100), create your new document in PETAL100 group

The document you create by following the steps given above will contain a complete Character Definition file of approximately the same form as the one you require. All you have to do to tailor it to your printwheel or printer character set is work through the file changing the details where necessary.

The Header

1 ! "*title-line1*" ↵

2 ! "*title-line2*" ↵

Title lines: The text of these lines is used in the file's Identity text which you can inspect from LocoScript's Disc Manager. This gives you a quick way of seeing what is in the file. Each *title-line* is up to 30 characters long.

3 ! "*author-name dd mmm yy*" ↵

Issue details: also included in the file's Identity text, so that you can readily see who prepared the file and when this was done. *author-name* needs to be **exactly** 20 characters long (so if you want a shorter name here, this will need to be padded out to 20 characters with spaces); *dd* is the day (01...31); *mmm* is the first three letters of the month (Jan, Feb etc.) and *yy* is the year (00...99).

Note the single spaces between the items on this line.

4 "*set-name*"*[default-PS-width]* ↵

Character Set name and default width: The Character Set name is the name by which this character set will be identified in the relevant LocoScript menus. *set-name* may be up to 12 characters long. You will probably take this name straight from the character set or the 'family' of printwheels that you are working on. For example, you might select the name 'Bilingual'.

The *default-PS-width* is only needed if character widths are being specified. Details on page 18.

5 "*style-name*" *pitch* ↵

Character Style name: *style-name* is up to 12 characters long, and *pitch* is 10, 12, 15, 17 or PS.

If you are setting up the file to support a range of printwheels, you can usually take the style name and the pitch from the wheel you will be using most often with this character set. For example, if the wheel is called some name like Letter Gothic 12, then you would use the style-name `LetterGothic` and set the pitch to 12 – just as you do when you are introducing a new printwheel for one of the supplied character sets (see Part II of the PCW External Printers Guide). If the character set is for a dot-matrix printer, you can give a dummy name like `Standard` and set any pitch. (We recommend 10.)

6 "*selection-sequence*" ↵

Printer Selection sequence: This is intended for use with printers on which you can select different character sets by sending escape sequences (see 'Advanced techniques', below). If the printer doesn't have this feature or if the character set you want can be selected by setting switches on the printer, all you need here is "" ↵

7 *[PS-units PS-origin]* ↵

Specification for Character widths: This line is only required if your Character Definition includes PS widths. Details are given on page 16.

8 *["underline-code" [width]]* ↵

Underline Character information: The first item specifies the code for the underline character. If the printer uses the ASCII code for the underline character (&5F, decimal 95), you can write it simply as: "_" ↵ Otherwise use "!'*value*'" ↵ where *value* is the printer code used for underline (written as either a decimal or a hexadecimal number – see the box on page 12). The *width* is only needed if your Character Definition includes PS widths. (Details on page 16.)

Note: If your printer has an auto-underlining facility, you can omit this line altogether. Nearly all dot-matrix printers have auto-underlining.

9 ! ↵

Marks the end of the Header part of the file.

The Header of the file which gives a straightforward definition of Bilingual wheels might therefore be:

```
1 ! "Bilingual wheels" ↵
2 ! "" ↵
3 ! "My Name                    25 Dec 90" ↵
4 "Bilingual" ↵
5 "LetterGothic" 12 ↵
6 "" ↵
```

(No Character width (PS) information, so no line '7')

```
8 "_" ↵
9 ! ↵
```

The File Body

The File Body essentially consists of lines of the following form:

code *name* *[PS-width-info]* ↵

each of which specify that this *code* should be sent to the printer whenever you want to print the LocoScript character with the given *name* and the width of the character for use in proportionally-spaced text. The *PS-width-info* is used to specify the width of the character. As this is optional, we will ignore it for the moment. (We look at character widths in detail on page 16.)

In the simplest type of character definition, the *code* is expressed as:

letter

or ! *letter*

or !'*value*'

or !'*special-name*'

where:

letter is A...Z, a...z, 0...9 or a number of the more common punctuation and other symbols. The code specified is the ASCII value for the given *letter*

!*letter* is used where the *letter* has special meaning in the character definition ie. * ! ; ". The code specified is the ASCII value for the given *letter*

!'*value*' directly specifies the code to be sent by number. *value* is the decimal or hexadecimal value of the code to be sent

!'*special-name*' specifies a code by the name of an ASCII control character (see page 25)

The *name* is expressed as "*character-name*" or "*accent-name*". The CHARKIT names for characters and accents are given in Appendix I. Any mixture of upper and lower case letters can be used when typing these names. If you specify a code for an accent then LocoScript will use this in combination with other characters to print accented characters.

For example, the line defining capital A would be:

```
A     "A Upper" ↵
```

whereas the line defining exclamation mark would be:

```
!!    "Exclamation mark" ↵
```

Most (but not all) of the codes between &20 and &7E can be expressed in terms of the ASCII character with this value. The main exceptions are the codes which vary depending on which language is selected. A case in point is the code &5B (decimal 91). In US ASCII this is a square bracket, but in other language versions of ASCII, this code is used to represent characters in common use in that language – such as é. So the way to express this code is the !'*value*' form, ie:

```
      !'&5B'  "Open Square Bracket"
or    !'91'   "Open Square Bracket"
```

The way in which we recommend you to express each of the codes required by the printer is explained in the box below (and listed in the table overleaf).

The simplest form of Character Definition file just requires you to set up one of these definitions for every LocoScript character and accent for which there is a corresponding character on the printwheel or in the printer's own character set. If you do as we suggest and use the LocoScript templates on your CHARKIT disc to set up your Character Definition file, most of these definitions will already be correctly specified. Typically, you will have to change no more than a dozen of the definitions in order to produce suitable definitions for all the characters on your printwheel.

• When you have finished preparing the LocoScript document containing your character definitions, press EXIT and save the document to disc in the usual way.

Expressing the codes required by the printer

The codes that a printer will respond to are numbers in the range 0...255. These numbers are also referred to as the 'values' of the codes.

A few of the codes with values in the range 0...127 are expressed in terms of their values, but most of them are expressed in terms of the characters etc. they represent in the ASCII character set. Details are given in the table overleaf..

The codes with values 128...255 are all expressed as !'*value*'. The *value* can be given either as a decimal number or as a hexadecimal number.

The digits of a hexadecimal number take values in the range 0...15 (just as the digits of a decimal number take values in the range 0...9). These digits are expressed as 0...9 and A...F (or a...f), where A represents the value 10, B the value 11, and so on.

Just as a two-digit decimal number is the value of the first digit times ten plus the value of the second digit, a two-digit hexadecimal number is the value of the first digit times sixteen plus the value of the second digit. For example, the hexadecimal number E7 is E * 16 + 7 = 14 * 16 + 7 = 231.

Where codes are expressed as !'*value*' and you want to express this *value* as a hexadecimal number, you write *value* as & followed by the digits of the hexadecimal number. The & simply signals that the value is a hexadecimal number.

Dec		0	16	32	48	64	80	96	112
	Hex	0	1	2	3	4	5	6	7
0	0	!'NUL'	!'DLE'	!'SP'	0	!'&40'	P	!'&60'	p
1	1	!'SOH'	!'DC1'	!!	1	A	Q	a	q
2	2	!'STX'	!'DC2'	!"	2	B	R	b	r
3	3	!'ETX'	!'DC3'	!'&23'	3	C	S	c	s
4	4	!'EOT'	!'DC4'	!'&24'	4	D	T	d	t
5	5	!'ENQ'	!'NAK'	%	5	E	U	e	u
6	6	!'ACK'	!'SYN'	&	6	F	V	f	v
7	7	!'BEL'	!'ETB'	!'&27'	7	G	W	g	w
8	8	!'BS'	!'CAN'	(	8	H	X	h	x
9	9	!'HT'	!'EM'	)	9	I	Y	i	y
10	A	!'LF'	!'SUB'	!*	:	J	Z	j	z
11	B	!'VT'	!'ESC'	+	!;	K	!'&5B'	k	!'&7B'
12	C	!'FF'	!'FS'	,	<	L	!'&5C'	l	!'&7C'
13	D	!'CR'	!'GS'	-	=	M	!'&5D'	m	!'&7D'
14	E	!'SO'	!'RS'	.	>	N	!'&5E'	n	!'&7E'
15	F	!'SI'	!'US'	/	?	O	_	o	!'DEL'

Creating the new Character Set File

Creating the Character Set file itself is firstly a matter of putting the Character Definition file into a suitable form for use with CHARKIT and then of running CHARKIT itself.

Getting the file ready for CHARKIT

So far, you have created a LocoScript document containing the information that you want in the Character Definition file. The next step is to create a simple text file version of this LocoScript document – ie. the Character Definition file itself. The steps are as follows:

1 When you return to the Disc Manager Screen, check that the File cursor is still on the Character Definition document you have just prepared, then press [f1].

2 Select the `Make ASCII file` option and press [ENTER].

3 Move the Group cursor to group 0 on your CHARKIT disc; then press [ENTER] again.

You have to pick out group 0 so that CHARKIT can access the file under CP/M.

4 When the menu appears, check that `Simple text file` is currently selected at the bottom of the menu (and the name that LocoScript is proposing to give the file is suitable) and then press [ENTER].

The Character Definition file you require will then be created and stored in group 0 on your CHARKIT disc.

5 Use the f8 Options menu to display Limbo files; then use the f3 File menu to erase these.

If you don't do this, your disc may seem full when you come to use it with the CHARKIT program. CP/M doesn't have the same system as LocoScript for clearing out Limbo files in order to make room for new files.

Running CHARKIT

To create the Character Set file:

1 Load CP/M.

2 Insert your CHARKIT disc and press [ALT]C

3 Type the command:

`CHARKIT` *printer-filename=character-filename* [RETURN]

where *printer-filename* is the name of the Character Set file you want to create and *character-filename* is the name of your Character Definition file.

The *printer-filename* must, of course, have the same main part to its name as the rest of the Printer Files you are using for this printer. The extension to the name must start with a # and the complete name must, of course, be a legal CP/M filename. For example, if you are creating an extra Printer File so that you can use a Bilingual wheel on your Juki 6100, you might call this JUKI6100.#BL. If you had called your Character Definition file BILNGUAL.WHL, the complete command would be:

`CHARKIT JUKI6100.#BL=BILNGUAL.WHL` [RETURN]

CHARKIT first displays the names of the files it will be handling and then reads the Character Definition file, interpreting it line by line. **Note:** If the Printer file you specify already exists, you'll see the message `Output file exists - Overwrite (Y/N)?`. If you type `Y`, the old file will be overwritten by the new Character Set file. If you type `N` (in fact, anything other than `Y`), you will be asked to type in another name for the new Character Set file and the original Character Set file won't be overwritten.

Any error in the definition generates the message `**ERROR FOUND**` – plus the option of having details of this and all subsequent errors sent to the printer as well as displayed on the screen. (The printer it will use is the one your system is currently set up for.) If you would like a printout of these errors, check that the printer is on-line and has paper in it; then type `Y`. If instead you type `N` (or press any other key) CHARKIT simply displays the details of the errors on the screen, but after each error it will wait for you to press a key before proceeding – to give you a chance to note down the details. Processing continues to the end of the file but no .# file is created. When it has finished, CHARKIT displays a message showing how many errors were found.

For each error, the line affected is displayed together with the appropriate error message. The point in the line at which the error occurs is marked by an exclamation mark. Details of the error messages and the likely cause of the problem are given in Appendix IV.

When no errors are found in the Character Definition file, processing carries through to the end of the file and the new .# file is stored in group 0 on your CHARKIT disc. The file you've created can now be installed and used just like any other Character Set file for this printer. Just follow the instructions given in the PCW External Printers Guide.

Important: *If you produce a new version of the Character Set file after you have started using it, you must re-select the Character Set in any document that is set up for this Character Set and that includes proportionally-spaced text. If you change the name of the Character Set itself, you will also have to install it all over again.*

Testing the new Character Set file

Before you set up any documents for your new Character Set, it is wise to check that the right characters are being printed.

We have supplied two documents that you might use for testing your Character Set: ALLCHARS and TEXTCHAR. These are stored in group 0 on your CHARKIT disc. ALLCHARS contains every character in the LocoScript 2 character set. TEXTCHAR contains a table of the characters together with their CHARKIT names.

Use the steps given in Part III of the PCW External Printers Guide to make your new Character Set the Current Character set and then print either of these documents on the appropriate printer (fitted with the appropriate printwheel). The characters printed will be the ones for which you have given character definitions and any 'Standard substitutions' CHARKIT makes for you (see Appendix II). All other characters are replaced by blanks.

With ALLCHARS, you can see whether your Character Set file is enabling you to print the right characters, simply by comparing the print-out you get with the output shown in Figure 1 opposite. (This was printed on the PCW8256/8512's built-in dot-matrix printer – which can print every character in the LocoScript 2 character set.) With TEXTCHAR, you should be able to see if anything is wrong just by inspecting the table that is printed.

Figure 1: ALLCHARS printed on the PCW8256/8512's built-in printer

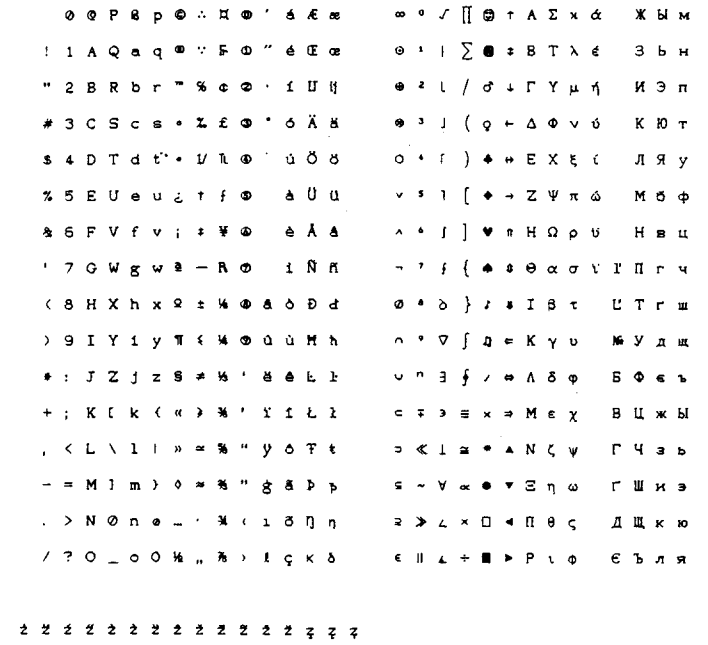
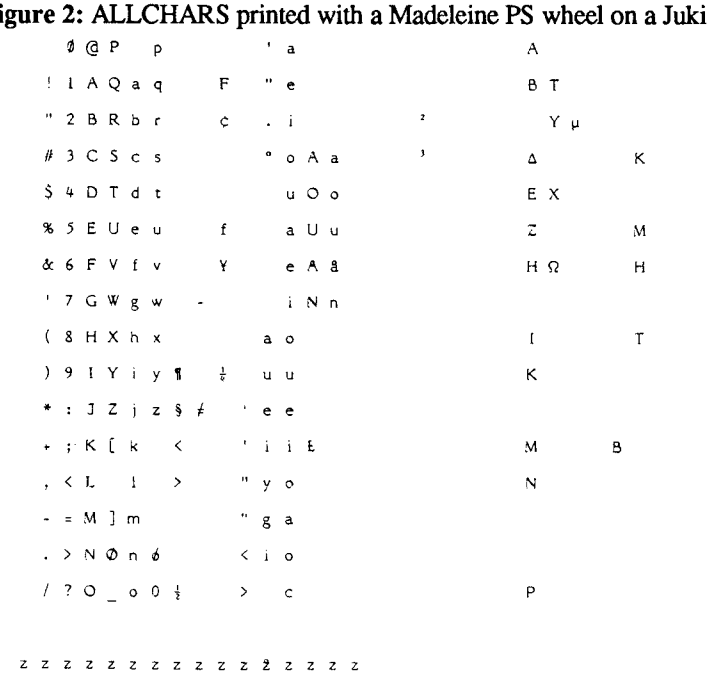


Figure 2: ALLCHARS printed with a Madeleine PS wheel on a Juki 6100



Enhancements

The instructions given above describe how to set up the simplest form of Character Set file which just associates LocoScript characters with printer codes in the most straightforward way. In this section, we describe how to build in:

- character widths – so that proportionally spaced text can be handled correctly
- characters for which there is a special accented form supplied in the printer's character set or on the printwheel

Character widths

Character widths are required for proportionally-spaced text where they enable LocoScript to calculate how to justify a line of proportionally-spaced text and where the line breaks should be. On a daisy-wheel printer, they are also used to tell the printhead how far to advance between printing one character and printing the next.

PS widths are usually only of concern when you are using a printer that supports precise character positioning (typically to 1/120th of an inch). You don't need to worry about character width at all if your printer isn't capable of such precise positioning – or you only want to use either fixed pitch text or fixed pitch printwheels (10, 12, 15 or 17 characters per inch). The characters on fixed pitch wheels are all the same width.

We do not, however, recommend proportional spacing on matrix printers unless this printer supports micro-justification. If the printer does not support micro-justification, the printhead of the dot-matrix printer cannot be positioned precisely enough to give good results and so we cannot guarantee Justification and Right Alignment.

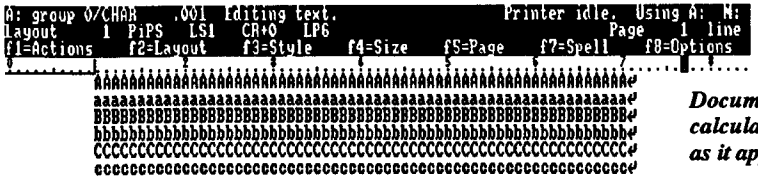
Character widths are given as a number of units to the inch and CHARKIT allows you to quote them in whatever units your printer uses. For example, if you have a laser printer that works at 300 dots per inch, you can quote the widths in 1/300ths of an inch.

Establishing character widths

The place to look first for the character widths you want is your printer's manual (or, in the case of a PS wheel, in the documentation supplied with this wheel). The widths are usually quoted as integer numbers, these being the number of standard units the character is wide. Even if there isn't a table for the particular font or wheel you want, there may well be a table which gives the widths for the standard font or wheel used on this printer. You can use these widths at least for your first version of the Character Definition file.

Unfortunately, character widths are rarely documented very well in printer manuals because they are irrelevant to most programs. You may find no mention at all of character widths. Alternatively, you may find that you are told the PS widths but not the units. In the latter case, however, it is reasonably safe to assume that the units are 1/60" for daisy-wheel printers, either 1/120" or 1/240" for 9-pin and 18-pin printers; 1/360" for 24-pin printers; and 1/300" for laser printers.

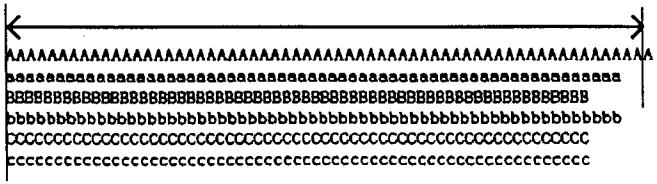
Even if no width information is given, all is not lost. If you have a dot-matrix printer or a laser printer, you can find out the PS widths by printing out a LocoScript document like the one shown in the screen dump at the top of the next page at High Quality, using a provisional version of the Character Set file (created from the Character Definition file you have set up so far) and then measuring this up.



Document used to calculate PS widths – as it appears on the screen

The way it looks when printed

Draw lines as shown in the example at the start of the line and another before the last letter. The distance between them is the measurement you need.



The important features of this document are that it contains a line of each character in the character set – either 31 or 61 characters on each line – and it has proportional spacing selected at the start. This gives you a sample of each character in the character set in your chosen font printed at its natural character width. (The numbers 31 and 61 are used because this makes the arithmetic easier as we shall see.)

The calculation of the width of a character in the appropriate printer units then goes as follows. Measure across either 30 or 60 characters as shown in the diagram above: record this measurement in inches. Now divide this measurement by the number of characters you have measured across – 30 or 60 – and then multiply by the number of printer units in an inch.

For example, if 60 of a particular character measure 5¾" (²³/₄"), then each character is ²³/₄ x ¹/₆₀" wide. If your printer is a 9-pin printer, then the printer unit is ¹/₁₂₀" and so there are 120 of these to the inch. Thus the character width is ²³/₄ x ¹/₆₀ x 120 which is approximately 11. (With a daisy-wheel printer you would multiply by 60, with a 24-pin printer you would multiply by 360, while with a laser printer you would multiply by 300.)

You simply have to do this measurement and calculation for each character in the character set. (Do this measurement carefully though: if you are too approximate in your estimates of the character widths, any justified sections of proportionally-spaced text could look distinctly ragged!)

If you have a Diablo-630 compatible printer, a Qume printer or a Juki 6100 printer, you have another option. The BASIC program supplied on your CHARKIT disc (described on page 5) not only prints out the characters on a printwheel in 'code' order; it also produces samples of the characters printed assuming PS widths of 7, 6, 5, 4 and 3 1/60ths of an inch respectively. By looking at these samples, you can judge the true PS width by assessing which sample appears best spaced. (You also have the advantage that you can guarantee that the righthand edge of any justified proportionally-spaced text really will be straight whatever widths you select because with these types of printer, the widths you set define precisely where each character will be placed.)

Note: Where possible, give the space character and the digits 0...9 the same PS width (typically 5/60", so that the spaces correspond to 12-pitch spaces). This ensures that tables of numbers print neatly. By the same token, you should also always set this width for the *default-PS-width* (see below) and for =, < and >, the three characters used to reserve space for page numbers. The positions of the digits of page numbers are worked out using the widths of the =, < and > characters, not the widths of the page numbers themselves.

The width information required in the Header

Including PS widths in your Character Set file means adding extra information to the Header part of the Character Definition file and entering character-width information in each of the lines in the File Body. The extra information you need in the Header is:

(Line 4) *"set-name" default-PS-width*↵ (just *"set-name"* if no character widths)
(Line 7) *PS-units PS-origin* ↵ (line omitted if no character widths)
(Line 8) *"underline-code" width*↵ (*width* omitted if no character widths)

Note: Line 8 is only relevant to simple daisy-wheel printers and others using SIMPLEBU and SIMPLEBS Printer Drivers. If you are setting up your definition for a printer which has auto-underlining, Line 8 can be omitted entirely.

Default-PS-width gives the PS-width (character width) that will be assumed if none has been defined. In practice, this means the size of the space left in PS text when the character you have asked to print isn't in the character set. (LocoScript leaves a space so that you can fill in these characters later – by hand, if necessary.) This width is measured in 1/240". You don't have to set a *default-PS-width*, but you are recommended to set one that is the same size as the space character and the digits. For example, if these characters have a PS-width of 5/60", that is the same as 20/240" and so the *default-PS-width* you need is 20. This is in fact the setting that LocoScript assumes if you don't specifically set a width, because it gives you a 12-pitch space – what you would normally want to leave.

PS-units specifies the fraction of an inch in which the PS-widths are given in the main part of the file. In the main, you simply need to quote whatever units your printer works in. For example, if you have a laser printer with a resolution of 300 dots per inch (ie. works in units of 1/300ths of an inch), you should make the PS-units 300. The value you specify must be an integer and this integer mustn't be greater than 3840 (corresponding to PS-widths in any unit down to 1/3840"). For daisywheels, the PS-widths for characters are normally quoted in 1/60ths of an inch, so you will usually specify *PS-units* of 60. (**Note:** The *PS-origin* setting on this line is left over from earlier versions of the CHARKIT program which forced you to work with PS widths in the range 1...15. It can be omitted from all new Character Definition files.)

The *width* defines the length of the underline character in 1/240ths of an inch. If you don't specify a width here, LocoScript assumes a setting of 20, which corresponds to 1/12th of an inch. This is the typical size for the underline character on a PS wheel.

Adding the widths to the File body

Character width information is entered by using the optional *PS-width-info* that forms the third part of any character definition:

code name PS-width-info↵

The PS-widths you quote simply need to be the ones given in your printer manual or that you have established yourself for example by using DAISY.BAS (see page 16). There are, however, some restrictions on the values you can quote:

- Each width must be an integer in the range 1...255
- The range of widths you quote must not correspond to a difference of more than 14/60" between the largest and the smallest (14/60" is just less than 1/4")
- The largest character can be no more than 78/60" wide (approx. 1.3").

In full, the *PS-width-info* can specify a different width for the character when it is printed upright and italic, and in both Draft and High Quality as follows:

code name PS-width[,[Draft-width][,[Italic-width][,Draft-italic-width]]] ↵

Note: Substitute *'Bold'* for *'Draft'* if you are describing either a laser printer font or a font on a HP DeskJet or compatible printer.

Typically, you (and printer manufacturers) just give the first of these widths – the one for upright characters in High Quality print mode. The other three widths are not needed when you are defining characters on a printwheel: they are only relevant if you are setting up a Character Set file for a dot-matrix printer or a laser printer. Even then, it is often sufficient just to give the figures for upright characters printed High Quality (so that your final High Quality version of any document will be right).

For example, the widths of the letters A...E on a Qume 96-petal PS wheel are quoted as 7, 7, 7, 7 and 6 respectively, so the lines for these characters in your Character Definition file would be:

```
A   "A Upper"     7
B   "B Upper"     7
C   "C Upper"     7
D   "D Upper"     7
E   "E Upper"     6
```

Notes: (i) If either the range of widths you quote or the largest character is too wide, you will see the message `Range of PS widths too great` *. No Character Set file will be made in this case.*

(ii) CHARKIT converts all the character widths you give to a particular value of PS-units before working on them. The calculations that this entails will be exact unless you use highly unusual PS-units. Where the widths that are actually used don't match what you specified, you will be warned of this by the message `WARNING - PS widths may not be accurate` *. The Printer file will still be made.*

Accented characters

With LocoScript 2 you can apply a wide range of accents to any character. If the accents you use are available in your printer's character set, you will generally be able to print accented characters as well.

Accented characters may be printed in two ways:

- the general approach is to print the accent and superimpose the character.
- however, the commonest ones (é, ü, ô etc.) may well be included on the printwheel or in the character set of your dot-matrix printer

No special character definitions are needed for accented characters that will be produced by printing an accent and then superimposing the character (ie. out of two separate characters on the printwheel). You just have to define the accent and the character separately (using definitions of the type described on pages 10 –12): LocoScript takes care of amalgamating the accent with the character for you.

Where an accented character is available ready made in the character set or on the printwheel, you have to take special steps. Moreover the definition you use depends on

whether the accented character is also a separate character in the LocoScript character set (something that is usually hidden from you!).

If the accented character is in the LocoScript character set, the line of the Character Definition file describing this character takes the form:

code "*accented-character-name*" *[PS-width-info]* ↵

where *code* is the internal code needed by the printer to print this character, and *accented-character-name* is the special name CHARKIT uses for the accented character. (A table of these names is given in Appendix I.)

For example, both the IBM character set and the LocoScript 2 character set include the character é. The IBM character set has the value &82 (decimal 130) for é and CHARKIT has the name ACUTE + E LOWER for this character. So if you are using an IBM character set, the line in the Character Definition file defining é might be:

```
!'&82' "Acute + e lower" ↵
```

If the accented character isn't in the list on page 56 (ie. it isn't a separate character in the character set), the line of the Character Definition file describing this character takes the form:

code "*accent-name*" + "*character-name*" ↵

where *code* is the internal code needed by the printer to print this character, *accent-name* is the CHARKIT name for the accent applied to the character, and *character-name* is the CHARKIT name for the character. (Appendix I gives details of the names to use.)

For example, if your printer's character set contained É (which is not in the LocoScript character set) with code value &A9, the line in the Character Definition file defining É might be:

```
!'&A9' "Acute" + "E Upper" ↵
```

Important: Don't include width information with any character definition like the one given here for É, which uses a combination of an accent with a character to specify a character that isn't in the LocoScript character set. The width LocoScript will use is the one you give in the definition of the unaccented character.

Advanced techniques

The Character Definition file can be further extended to cover:

- the 'extra' two or six characters on a printwheel that are accessed by short escape sequences (daisy-wheel printers only)
- characters available in different language variants of the same character set (dot-matrix printers only)
- and, briefly, characters that can be produced by more complex character definitions that involve using your printer's control codes

This section describes the advanced techniques used to incorporate these in your Character Definition file.

The 'extra' characters on a printwheel (daisy-wheel printers)

The ASCII range of character codes (&21...&7E) only provide codes for 94 characters on a printwheel, so special codes are required to access the other characters on a printwheel – two on a 96-petal wheel, six on a 100-petal wheel. The codes for these 'extra' petals are usually short Escape sequences. For example, Diablo-type printers access the characters on spokes 95 and 96 of a 96-petal wheel by the escape sequences ESC Y and ESC Z.

You don't have to specify these special 'Printer sequences' yourself. The details are recorded in the Printer Driver file for the printer. You simply write character definitions for the characters on these 'extra' petals in which the *code* is `*P`*number*, where *number* is the number of the Printer sequence you require.

For example, if a wheel had ½ on the spoke accessed by Printer sequence 1 and ¼ on the one accessed by Printer sequence 2, the character definitions you need would be:

```
*P1 "Half" ↵
*P2 "One quarter" ↵
```

Printer sequences differ for different printers even where they use the same printwheels. In particular, the same printwheels can be used on both Diablo-type and Qume-type printers, but while Diablo-type printers use the escape sequences ESC Y and ESC Z, Qume-type printers use ESC Space and ESC / to access the same petals. LocoScript has ESC Y as Printer Sequence 1 and ESC Z as Printer Sequence 2 in the Diablo Printer Driver, and ESC space and ESC / as Printer Sequences 1 and 2 in the Qume Printer Driver. This means that when you want to use a printwheel that you normally use on a Qume-type printer on a Diablo-type printer (or vice versa), you don't have to create a new file – you can simply copy the Qume file and rename it as a Diablo file.

Characters from alternative character sets (dot-matrix printers)

Many dot-matrix and laser printers support a range of different character sets and have escape sequences which allow you to shift into and out of these different character sets 'on the fly'. These character sets may contain characters that aren't available in the 'normal' character set. By incorporating the appropriate escape sequences into your Character Definition file, you can add extra characters to the range of characters you can print.

A particular example of this are the different 'language' character sets of FX-80 compatible printers (identified by the numbers 0...8). Language '0' on such printers gives you essentially US ASCII, but when one of the other languages is selected, some of the codes are used to represent characters that are in common use in that language. For example, you will rarely find a German Scharfes S in an English or American character set but it is usually included in the German variant. The escape sequence `ESC R` *number* is used to change from the current language to the language with the given number.

LocoScript lets you use these extra characters by having a system of User states, each of which is also identified by a number. You normally work in User state 0 and this gives you access to the characters in the 'normal' character set. If you want to use characters from one or more of the alternative character sets, you need to:

- define a User state corresponding to each of these alternative character sets, and
- tell LocoScript the escape sequences it needs to send to the printer, both to select the alternative character set and to go back to using the 'normal' character set.

The line that defines a User state takes the form:

`**number on-command off-command` ↵

where *number* is between 1 and 31 and identifies the User state, *on-command* is the sequence of bytes (ie. values between &00 and &FF) that the printer uses to shift into the required character set, and *off-command* is the sequence of bytes it needs to shift back out to User state 0 and the 'normal' character set.

The sequences of bytes are each written as "*byte[byte]**" (where *byte[byte]** is used to represent one or more bytes). The individual bytes are expressed in the same way as the codes we have used earlier (see pages 10–12). This will also be the way the escape sequences you need will be given in your printer's manual.

For example, an FX-80 printer uses the sequence `ESC R 2` to select the German character set and `ESC R 0` to return to the standard US set. So to define the German character set on this printer as your User state 1, you need the following definition in your file:

```
**1 "!'ESC'R!'2'" "!'ESC'R!'0'" ↵
```

(**Note:** We've defined language '2' as User state 1 here to underline that these numbers don't have to be the same. In practice, you would probably make language '2' User state 2: it is easier to remember and makes your Character Definition file easier to read.)

Once the User state has been defined, it can be used in a character definition as follows:

`*Unumber code name [PS-width-info]` ↵

with *code*, *name* and *PS-width-info* having their usual meanings, but referring to a character in the alternative character set.

The Scharfes S character has the value &7E (decimal 126) in the German character set. So having set up this as User state 1, the definition for this character would be:

```
*U1      !'&7E'  "Scharfes S" ↵
```

Note: We also recommend making the US set your normal character set and either setting the switches on your printer to match or defining a Selection Sequence in the Header section of the Character Definition file to ensure that the printer initially uses the US character set: see 'Selecting the initial character set' below.

Selecting the initial character set

Where your printer offers a choice of character sets, you need to ensure that the correct character set is selected to start with. This is particularly important when you are shifting in and out of different character sets so that you can print characters that aren't in the 'normal' character set: the initial character set defines User state 0.

The initial character set is normally defined by switches on the printer and this could mean either remembering to check the switch settings before starting to print or only ever selecting one character set this way. This is obviously inconvenient, particularly if you use your printer with different programs, each requiring different switch settings.

If your printer allows you to select character sets by sending escape sequences, then you don't have to worry about the character set that is selected by the printer switches – because you can use the Character Definition file to arrange that the correct one is selected for you. The line that enables you do this is Line 6 of the Header, which allows you to define a 'Printer Selection Sequence' that gives you the User state 0 you require. LocoScript 'primes' the printer with whatever commands are given in this sequence.

The Printer Selection Sequence is no different to on- and off-commands used above to shift into and out of other User states. Like these, it is written as "*byte[byte]**". For example, if you are defining a Character Set for use with an FX-80 compatible printer, you could use this feature to ensure that User state 0 corresponds to the US character set (as we recommend) by defining the Printer Selection Sequence:

```
"!'ESC'R!'0'"
```

More complex definitions

If a particular character that you want to print is not directly available on your printer, you may be able to generate the appropriate character by sending a whole sequence of bytes to the printer which do such things as changing the position of the printhead.

CHARKIT allows you to build highly complex sequences of bytes into your Character Definition file – up to 255 bytes long. We cannot tell you precisely what commands you will need for the characters you want to produce – only an expert on your particular printer will be able to tell you that – but we can show you the *type* of character definitions you might need. (**Note:** The maximum number of *characters* in a character definition line is 255. However, long sequences of bytes can be split over a number of lines of the character definition file, by dividing the sequence into shorter sequences, joined by `&`s. The example at the bottom of the next page shows how this is done.)

The way you define the character you want is to make the *code* in your character definition the sequence of bytes that have to be sent to the printer in order to print the character. This sequence is written as:

"*byte[byte]**" *[side-effect]**

where *byte[byte]** represents two or more *bytes* – that is, values between &00 and &FF – and *[side-effect]** represents any number of *side-effects*. We will explain what these *side-effects* are shortly.

The rules for expressing the bytes in these sequences are the same as the ones used elsewhere to express the codes the printer requires, details of which are given on pages 10–12. The sequences you will need should be given in the printer's manual.

For example, you might want to set up a special sequence so that you could print a superscript `n`, which involved selecting your printer's superscript option and changing temporarily into 12 pitch text. On an FX-80 printer, the control sequences for these two actions are `ESC S 0` to select superscript and `ESC M` to set 12 pitch text. So your special byte sequence should be `ESC S 0 ESC M`, followed by `n` (the character you want to print) and this would be written in the character definition as:

```
"!'ESC'S0!'ESC'Mn"
```

(**Note:** The bytes have been written one after another here, but you can, if you like, improve the appearance of the command by putting spaces between them. There is no confusion between these spaces and any 'space' characters in a sequence because you write those as `!'SP'`.)

While LocoScript is printing a document, it keeps careful track both of the state it needs the printer to be in at each stage (ie. which character pitch, which line pitch, which print effects etc. need to be in operation) and of the actual state the printer is in. If there is a difference between the two, it makes the necessary adjustments to the printer state before printing the next character, by sending the appropriate codes.

Sequences such as the ones described above change the printer state and you need to make LocoScript aware of these changes so that it can carry out the necessary adjustments. To do this, you follow the bytes of the commands you use to print your character by a list of these changes. These are the *side-effects* we mentioned above.

You can make a wide range of changes to the printer state and there is a special way of expressing each side-effect. For example, you record the side-effect of 12 pitch text being set in the character definition as +12 and that of setting superscript as +R. Details of the other possible side-effects are given in Appendix III, where the complete syntax of Character Definition files is given.

Putting all this together gives the character definition:

```
"!'ESC' S 0 !'ESC' M n" +12+R "Superscript n"  ↵
```

Important: There are some changes that you can make to the printer state which LocoScript cannot automatically recover from through this side-effect mechanism. These are changes to the language character set you are using and special changes to the print position. For example, it cannot automatically re-adjust after you have entered a printer's graphics mode and moved the printhead to a particular position before printing a character. The commands that make the necessary re-adjustment have to be included in the sequence.

You particularly need to be aware of this when you define extra characters by superimposing existing characters. On a daisy-wheel printer, the way to superimpose characters is to go into graphics mode first because then you get complete control over the movement of the printhead. In particular, the printhead doesn't automatically move forward after the character is printed. If you print all the 'elements' of the special character in graphics mode, you will leave the printhead in a different position to the one LocoScript expects and so you have to complete your byte sequence with an instruction to move forward to the next character position. However, you can readily get around this problem by printing all but the last 'element' in graphics mode and switching back to text mode immediately before printing the last element.

For example, to print ¼, you might print:

- a 1 one step up and one step to the left of its normal position
- a 4 one step down and one steps to the right of its normal position, and finally
- a slash in its normal position.

On a Diablo-630 compatible printer, the commands to enter and leave Graphics mode are ESC 3 and ESC 4, and once in Graphics mode, the command SP moves the printhead forward 1/60", the command BS moves it back 1/60", the command LF moves it 1/48" down the page and the command ESC LF moves it up 1/48". The character definition that prints ¼ might therefore be:

```
"!'ESC'3 !'BS' !'ESC' !'LF' 1 !'LF' !'LF' !'SP' !'SP'" &  ↵
"4 !'ESC'!'LF' !'BS' !'ESC'4 /" "One quarter"  ↵
```

Note how the control sequence has been split into two shorter sequences to spread the definition over two lines. The sequences are linked by the & at the end of the first line.

Control Code names

The following lists give the names that may be used to express control codes within Character Definition files:

(i) in order of internal value

NUL	DC1 or XON
SOH	DC2
STX	DC3 or XOFF
ETX	DC4
EOT	NAK
ENQ	SYN
ACK	ETB
BEL	CAN
BS	EM
HT	SUB
LF	ESC
VT	FS
FF	GS
CR	RS
SO	US
SI	SP
DLE	DEL

(ii) in alphabetical order of name

ACK	FS
BEL	GS
BS	HT
CAN	LF
CR	NAK
DC1	NUL
DC2	RS
DC3	SI
DC4	SO
DEL	SOH
DLE	SP
EM	STX
ENQ	SUB
EOT	SYN
ESC	US
ETB	VT
ETX	XOFF
FF	XON

Extra Character Sets for the PCW9512 built-in printer – the MKWHEEL program

This section describes what to do in the unlikely event that you have a printwheel for the PCW9512 printer for which no files have been supplied in the Printer Support Pack, for example because it has become available since this pack was produced. In general, the information given here should be ignored. In particular, it is of no use to you if you have a PCW8256/8512.

The 9512 Printwheels disc in the Printer Support Pack allows PCW9512 owners to use any printwheel on their built-in printer with both LocoScript and CP/M. Ready-made files are supplied for all the different printwheels currently available (October 1990); however, it is possible that further printwheels will become available that aren't supported by these files. If you have such a printwheel, you need to set up a Printwheel Description file for it, exactly like the SOURCE.#*xx* Printwheel Description files on the disc. This file can then be used with the SETWHEEL program to set up CP/M to use this printwheel (as described in the section on using printwheels with CP/M in the Printer Support Pack booklet) – and with another program on the Printwheels disc, the MKWHEEL program, to produce the Character Set you need to use the printwheel correctly with LocoScript.

The files

The following files on the Printwheels disc are connected with the MKWHEEL program.

Group 3 – PROGRAMS

MKWHEEL.COM	Program that creates Character Set files for use with LocoScript
MKWHEEL.EG	Example Printwheel Description file (LocoScript document)
TEMPLATE.STD	Template for preparing Printwheel Description files
ALLCHARS	A document containing all the LocoScript 2 characters
PWHLCHAR	A document containing all the LocoScript 2 characters plus their MKWHEEL names

We strongly advise you to prepare a copy of the Printwheels disc and to do all your work with this copy. Treat the supplied disc as your Master copy, to use only to make a replacement working disc if your copy becomes damaged. (How to copy a disc is described on page 109 of the PCW9512 User Guide.)

Background information

All the actions in preparing and printing documents on a printer involve using codes to represent characters and actions. For example, when you print a LocoScript document, the process of printing each character requires LocoScript to send a sequence of codes to the printer that tell it which petal on the printwheel to select, how hard to hit this petal and how far to move the printhead after printing the character.

The codes used are simply numbers, generally in the range 0...255. For many reasons we find it easier to consider these values in hexadecimal (base 16), so the codes are numbers in the range &00...&FF – where the '&' signals that a hexadecimal number follows. (There is a more detailed description of hexadecimal numbers on page 41.)

An important aspect of using a daisy-wheel printer is that the printer can only operate in terms of selecting a petal: it cannot 'see' which character is on this petal. So you need a table of information that records which character is on which petal – or rather, you need a number of such tables. Printwheels have different selections and arrangements of characters on them, depending on which family the printwheel belongs to – and, in fact, on whether it is a fixed pitch wheel or a proportionally-spaced (PS) wheel.

These tables are known as Printwheel tables and their main use is to translate the codes in your document into the codes needed by the printer. They also record how hard the petal should be hit in order to give uniform print intensity (larger characters like M have to be hit harder than smaller characters like full stops and hyphens) and the width of each character so that proportionally-spaced text can be positioned correctly.

As well as needing a table for each family of printwheels (covering both fixed pitch wheels and PS wheels), you also need a different table when you are using the CP/M operating system to the one that you use with LocoScript. CP/M and LocoScript use the same codes for a number of characters, because they both follow the American ASCII standard in which the codes &20...&7E (decimal 32...126) are used to represent the letters A...Z, a...z, 0...9 and the common punctuation marks. However this standard only covers some of the characters and codes available and outside this area, CP/M and LocoScript don't use the same codes. In addition, CP/M allows you to select from a number of different 'language' variants in which some of the codes have different meanings. For example, when the 'American' language is selected, &23 (decimal 35) represents # but when 'English' is selected it represents £. These different language details have to be built into the CP/M Printwheel table.

The Printwheel tables for your system's 'Standard' wheels (ie. printwheels of the same nationality as your system) are built into the PCW9512.PRI Printer Driver file (in the case of LocoScript) and also into the CP/M operating system itself. Tables for other wheels have to be set up separately.

Printwheel tables are too complex for you to set up directly. Instead, you set up the information that is required in a special file known as a Printwheel Description file and then special programs generate the printwheel tables you need from the information stored in the Printwheel Description file. When you are using the CP/M operating system, Printwheel tables are generated from Printwheel Description files by the SETWHEEL program described in the Printer Support Booklet. The tables for LocoScript are held as Character Sets, in special Character Set files. The program that can generate a Character Set file from the information in a Printwheel Description file is called MKWHEEL.

This section of the Defining Character Sets booklet describes how to set up a Printwheel Description file for a particular printwheel and then how to use the MKWHEEL program to generate a Character Set file for use with LocoScript. If you want to use this printwheel when you are running a CP/M program, then you simply use the Printwheel Description file you prepare with the SETWHEEL program as described in the Printer Support Pack booklet in the section on using printwheels with CP/M.

The Printwheel Description file

The bulk of the Printwheel Description file comprises lists of the characters on the printwheel. There are separate lists for Fixed Pitch wheels and Proportionally-spaced (PS) wheels of the same family because the characters are arranged differently on PS wheels, and there is another list for CP/M. Together with these lists are details of the hammer intensity for each character (ie. how hard the petal should be struck) and the widths of the characters on the PS wheel.

The Printwheel Description file is divided into the following sections:

- a Header section – which gives the name LocoScript is to use for the Character Set, and other such information
- a Fixed Pitch section – which specifies the arrangement of the characters on fixed-pitch versions of the wheel and the hammer intensity for each each character
- a PS section – which specifies the arrangement of the characters on the proportionally-spaced (PS) wheel and the width of each character
- a Substitutions section – which lets you specify characters or character combinations to use in place of a character that isn't specifically included on the printwheel
- a CP/M section – which specifies the characters on the wheel in terms of the codes used by CP/M, in all the language variants

These sections can appear in the file in any order and it is not even necessary to include all the sections – if you won't want to use the information in a particular section, then in general you can miss this out. (The exception is the Fixed Pitch section which must always be included.). For example, if you only want to use your printwheel with LocoScript, then you don't need to include the CP/M section – the file you produce can still be used with the MKWHEEL program to produce a Character Set file. (If you try using this Printwheel Description file to define an Extra wheel for CP/M, however, SETWHEEL will reject the file.) Similarly, if there isn't a PS wheel in this family of printwheels, then you don't have to include the PS section. However, you do have to ensure that the same characters are specified by each section.

Both MKWHEEL and SETWHEEL start from a copy of the English (GB) Printwheel table and substitute new values into this table according to details you give in the Printwheel Description file. If you don't specify a petal or a code or specify it incorrectly, the corresponding part of the table is left unchanged. As a side-effect of this, you don't actually have to set up a complete description of a printwheel. Instead you can just list the differences between your wheel and the England wheel: this makes your Printwheel Description very concise where both wheels contain many of the same characters. (Typically, it will be the ASCII characters that they will have in common.). To find out the details of the English Printwheel table, simply look at or print the MKWHEEL.EG example: this contains the full set of definitions needed to describe England printwheels.

Creating the Printwheel Description file

The Printwheel Description file you need to prepare must be a simple text file. It can therefore be produced using just about any text editor or word-processing program: you just have to make sure that the version of this file that you use with MKWHEEL and SETWHEEL is free of any of the special commands and codes the word processor uses. We would always recommend using LocoScript in the first instance and then using the 'Make ASCII file' option in the Disc Manager's f1 Actions menu to convert it into a file which MKWHEEL and SETWHEEL can read. **Note:** Using the RPED editor on your CP/M disc to prepare this file is not recommended because Printwheel Description files are typically over 200 lines long – ie. over the maximum size this editor can handle.

The information the Printwheel Description file contains must be set out correctly if the MKWHEEL and SETWHEEL programs are to work correctly. In the following instructions, we describe how to set up a Printwheel Description file. We assume that you will be using LocoScript to prepare this file.

Any number of comments can be inserted into this file, either at the ends of definition lines or on separate lines interspersed between the definition lines. These comments have to start with a semicolon (;). Their role is to make the file readable. We don't include any comments below so that the central structure of the Printwheel Description file is not obscured. You can see them in action in the example Printwheel Description file, MKWHEEL.EG.

IMPORTANT: The range of characters that can be used in the Printwheel Description file is limited to the following so that there is no risk of mis-interpretation:

```
A...Z, a...z, 0...9, ! " % & ' ( ) * + , - . / : ; < = > ? _
```

The names you specify for the Character Set and the Character Style are also limited to these characters.

• *Setting up a Printwheel Description file requires detailed information about the printwheel you are describing – which character is on each petal (both for fixed pitch wheels and for PS wheels), what hammer intensity should be used for each petal, the width of each character, etc. If you are lucky, these details will be provided on the printwheel's information sheet.*

Initial steps

Before you start preparing a Printwheel Description file for yourself, we suggest that you print out MKWHEEL.EG on the Printwheels disc. This is a LocoScript version of an example Printwheel Description file, which is useful to look at as you read through the following instructions.

Printwheel Description files can, if you wish, be created 'from scratch', simply by typing all the information into a fresh LocoScript document. However, because Printwheel Descriptions are rather complex, it is always better to work from an existing Printwheel Description file or from the framework for such a file. For this reason, we suggest you either edit a Printwheel Description file for a similar printwheel (ie. one that you know has many of the same characters on the same petals) or that you take advantage of the template we have set up for you in Group 3 of the Printwheels disc. This template contains an outline of a Printwheel Description. The steps to take are as follows:

1 Load LocoScript 2 and insert the Printwheels disc (press [f7] to signal the change of disc)

2 **If you are creating a completely new description:** Create a new document in group 3 (the PROGRAMS group) of the Printwheels disc and give it an appropriate name for the family of printwheels you want to support. We suggest using names like SCRIPT.#*xx* – partly to tie in with the PCW9512.#*xx* file you will ultimately create and partly to remind you that this version is a LocoScript document.

If you are basing your description on an existing file: Create a new document in *any group on the disc other than group 3* (give it an appropriate name) and then use the f1 'Insert text' option to paste in a copy of the SOURCE.#*xx* file you want to copy.

The document created by following the steps given above will contain the basis for your Printwheel Description. To tailor it to your printwheel, you simply work through the file, filling in the details and cutting out the sections you do not require. When you have finished preparing the LocoScript document containing your character definitions, press [EXIT] and save the document to disc in the usual way.

The Header section

The Header section of the Printwheel Description file contains general details that will be needed in the LocoScript Character Set file supporting your printwheel:

- the name to be used for the Character Set;
- the Character Style of one of the printwheels in the family; and
- the Identity text for the file (ie. the description you see if you 'Inspect' the file).

The structure of this section is as follows:

HEADER ↵
"*title-line1*" ↵
"*title-line2*" ↵
"*title-line3*" ↵

Title lines: The text of these lines is used in the file's Identity text which you can inspect from LocoScript's Disc Manager. This gives you a quick way of seeing what is in the file. Each *title-line* is up to 30 characters long.

"*set-name*" ↵

Character Set name: The Character Set name is the name by which this character set will be identified in the relevant LocoScript menus. *set-name* may be up to 12 characters long but your choice of name is limited to the characters listed on page 29. You will probably take this name straight from the 'family' of printwheels that you are working on. For example, you might select the name `SerboCroat` when you are describing a family of Serbo-Croat printwheels. The name you give should be unique to this Printwheel Description file – so that LocoScript can always identify the character set you need.

"*style-name*" ↵
pitch ↵

Character Style: *style-name* is up to 12 characters long, and *pitch* is 10, 12, 15 or PS. Your choice of name is limited to the characters listed on page 29.

If the file is to support a range of printwheels, take the style name and the pitch from the wheel you will be using most often with this character set. For example, if this wheel is called Letter Gothic 12, then you would use the style-name `LetterGothic` and set the pitch to 12 – just as you do when you are introducing a new printwheel for one of the supplied character sets (see Part II of the PCW External Printers Guide).

The Header of the file defining Serbo-Croat wheels might therefore be:

```
HEADER ↵
"Serbo-Croat wheels" ↵
"" ↵
"My Name                    1 Jan 88" ↵
"SerboCroat" ↵
"LetterGothic" ↵
12 ↵
```

Note: The Header section can be omitted: then the Identity text, Character Set and Character Style name are set to blank and the pitch is set to 10. The Printwheel Description won't fail in these circumstances but you will get blank names in menus. Furthermore, if the Header is omitted in more than one Printwheel Description, you will only see one blank name in each menu and you won't know which printwheel it refers to.

The Fixed Pitch section

This section must be included in any Printwheel Description file.

The Fixed Pitch section describes the arrangement of characters on fixed pitch versions of the wheel. It is also used to specify the strength with which each petal should be struck. It starts with the keyword `FIXED` (on a line on its own) and consists of definitions of the following form:

petal *character* *[hammer-intensity]* ↵

each of which specify the number of the *petal* which should be selected whenever you want to print the given LocoScript *character* and that the petal should be struck with the given *hammer-intensity*.

(The *hammer-intensity* is written between slanted square brackets to show that you don't have to specify it: the hammer intensity specified for this character on English wheels will be used instead, or failing that, a hammer intensity of 6.)

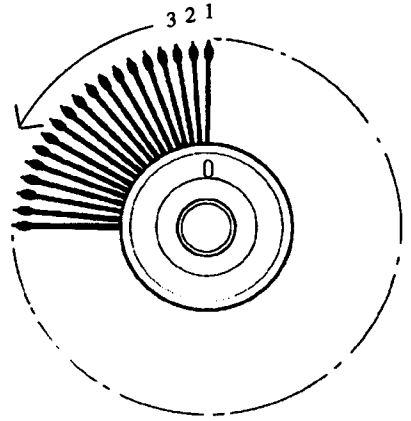
The different parts of the definition must be separated by either tabs or spaces.

Petal numbers

The printwheels used on the PCW9512's built-in printer have 100 petals.

The way to number these is from 1 to 100, starting at the petal above the rectangular index slot and counting anti-clockwise with the printwheel held face up.

Note: This way of numbering the petals should be used even when it doesn't match the numbering system specified in the documentation supplied with the printwheel.



Specifying the characters

The character in each definition is specified in one of the following ways:

- by the character itself
- by its special MKWHEEL name

The characters that can be given as the character itself are the letters A...Z and a...z, and some of the common punctuation marks, together with the characters 0...9 which you can express by writing between double quote marks ("0", for example) and some special forms for the characters that have special meaning in a Printwheel Description file viz. : ! " and ,. MKWHEEL names are used for all the rest of the characters.

In addition, there are special ways of handling accented characters and duplicated characters on the printwheel. (It is quite common for a printwheel manufacturer to put two full stops or commas on a printwheel.) These special techniques are described below.

Details of the MKWHEEL names are given in Appendix I. Any mixture of upper and lower case characters can be used when typing these names.

For example, if the first six characters on your fixed-pitch printwheel were +, V, 2, ", #, ¾, and ², then the first few lines of the 'Fixed' section might be (without the hammer intensities):

```
FIXED ↵
1     +↵                          } Standard characters which are
2     V↵                            expressed as themselves
3     "2"↵                        Number
4     !"↵                         Character with special meaning in
                                    Printwheel Description files
5     three_quarters ↵            } Characters which must be
6     superscript_2 ↵             }   described by name
```

Note: LocoScript 2 handles a very wide range of characters but there is still a chance that your wheel has a character on it that isn't in the LocoScript 2 character set. In such cases, we recommend that you specify this character as one of User Symbols 0...9. This means you automatically associate a special key combination (generally EXTRA + *number*) with this special character which will help you when you come to use it.

Accents and accented characters

As well as letters, numbers and punctuation marks, many printwheels also carry a range of accents and/or accented characters. These are specified in the Printwheel Description file in much the same way as other characters, but there are a few extra complications.

The standard way of listing accents on their own is by their MKWHEEL name, just like any other character: these names are also listed in Appendix I. For example, the name for the ` accent is GRAVE, so to specify that this accent is on petal number 7 you would put:

```
7   grave ↵
```

To specify an accented character, take the names first of the accent and then of the character and put a + between them. For example, to specify é on petal number 8 put:

```
8   acute+e ↵
```

The complication in specifying accents on their own is that you also need to consider how this accent should be positioned over other characters.

To save you work, the following defaults are applied:

- all accents except Stroke are assumed to be 'low', so that they will be raised before being amalgamated with capital letters. Stroke is assumed to be 'high', so that it is lowered before being amalgamated with lower-case o, for example.
- accents that go underneath the letter are defined to be both 'high' and 'low' so that the accent appears in the same place for both capital and lower-case letters.

As a result, you only have to specify anything special when the accent on the wheel doesn't fit these rules – for example, a 'high' Tilde (specifically for capital letters) or a 'low' Stroke.

The accent position is specified by adding an extra keyword in the definition as follows:

- To specify a 'high' accent, use: *petal* `high` *accent* ↵
- To specify a 'low' accent, use: *petal* `low` *accent* ↵

For example, to specify a 'high' Tilde on petal 11, you would put:

```
11  high tilde ↵
```

(If both a 'high' and a 'low' version of an accent appear on the same wheel, you still only have to use the high or low keyword with the one that doesn't fit the standard rules. However, you will find your file easier to read if you put the appropriate keyword in both definitions. *Note:* Such a pair of accents do *not* count as 'duplicates' as described below.)

Note: No more than 31 accented characters may be defined, but accented characters specifically included in the LocoScript character set don't count towards this total.

• As well as specifying the height of the accent on the wheel, it may also be necessary to specify the height of characters that it will be amalgamated with. Again, a range of defaults apply and you only have to specify where a character doesn't fit these rules:

- *A...Z, 0...9, = and / are assumed to be 'high' (and to require 'high' accents)*
- *everything else is assumed to be 'low' (and to require 'low' accents)*

Once more, the extra keywords to use are `high` *and* `low` *as follows:*

- *To specify a 'high' character, use: petal* `high` *character* ↵
- *To specify a 'low' character, use: petal* `low` *character* ↵

So, for example, to specify an upper-case Ishe (Cyrillic I) on petal 21, you would put:

```
21  high I_Cyrillic_upper ↵
```

Duplicated characters

If a printwheel contains two of any character, you need to specify that one petal is a Duplicate of another. The way to do this is to simply to put `DUP` before the name or code for the character in the definition for one of the two petals (which one doesn't matter). For example, if your printwheel has a comma on both petal number 12 and petal number 64, you might specify these as follows:

```
12  "," ↵
64  dup "," ↵
```

Hammer Intensities

The hammer intensity that you give for a character specifies how hard the petal should be struck. If all the petals were struck equally as hard, then smaller characters would appear very much darker than the larger characters because the pressure over the smaller area of the character would be very much greater. The result would be a very uneven quality of print. Being able to vary the hammer intensity for each character allows the print intensity to be more even.

Each hammer intensity is specified by a number in the range 4...10, 4 corresponding to the petal being struck as lightly as possible and 10 corresponding to the petal being struck heavily. Numbers between 4 and 10 tell the printer to strike the petal somewhere between the lightest and the heaviest: for example, a hammer intensity of 6 tells the printer to strike the petal a little harder than a hammer intensity of 5 but not as hard as a hammer intensity of 7. **Note:** Only whole numbers can be used.

The hammer intensity that should be used for a particular character isn't affected by the typestyle or the pitch of the wheel you are using and so should be the same across all the printwheels in the same family. A list of the hammer intensities to use should be given in the documentation supplied with the printwheel.

If no hammer intensities are listed, just leave out this part of the definition – because then the program will set the value from the English Printwheel table if the character is also on England printwheels – or failing that, give a hammer intensity of 6.

Inserting the hammer information might give you the following start to the Fixed Pitch section:

```
FIXED ↵
1    +                  5↵
2    V                  7↵
3    "2"                6↵
4    !"                 4↵
5    three_quarters     7↵
6    superscript_2      5↵
7    grave              4↵
8    acute+e            7↵
```

Hammer intensities on PS wheels

Sometimes, a printwheel manufacturer will recommend you to use a hammer intensity for a character on a PS wheel one higher than that used for the same character on a fixed pitch wheel. If this is the case, you should give first the hammer intensity for the fixed pitch wheel, then a slash and then the hammer intensity for the PS wheel. For example, if the hammer intensity recommended for V is 7 on fixed-pitch wheels (where it is on petal number 2, say) but 8 on PS wheels, then the definition for petal number 2 becomes:

```
2    V                  7/8↵
```

Note: The hammer intensity for the PS character can only be one more than that for the fixed pitch character.

The PS section

Omit this section if you don't have a PS wheel for this family of printwheels.

The PS section of the Printwheel Description file describes the arrangement of characters on PS versions of the wheel, which is typically different from their arrangement on the corresponding fixed-pitch wheels. It is also used to specify the width of each character on the PS wheel: these widths are used when printing proportionally-spaced text to calculate how to justify a line of this text and where the line breaks should be.

The PS section starts with the keyword `PS` (on a line on its own) and consists of definitions as follows:

petal character PS-width ↵

each of which specify the number of the *petal* which should be selected whenever you want to print the given LocoScript *character* and the *PS-width* of the character on the petal. This width can be any number in the range 3...8, corresponding to the width of the character in $^{1}/_{60}$ths of an inch: for example, if the character is $^{5}/_{60}$" wide ($^{1}/_{12}$"), then its *PS-width* should be given as 5. (**Note:** You don't set hammer intensities in this section of the Printwheel Description – only in the Fixed pitch section.)

Again, the different parts of the definition must be separated by either tabs or spaces.

The petal numbers and the characters are specified in exactly the same way as they are in the Fixed Pitch section of the Printwheel Description. The difference is simply the addition of character width information.

The place to look for the width information you need is in the documentation supplied with your PS wheel. (They are usually quoted as whole numbers, these being the number of $^{1}/_{60}$ths of an inch the character is wide.)

Unfortunately, however, character widths are not always documented very well by printwheel manufacturers because they are irrelevant to most programs. So you may find no mention at all of character widths. In that case, you should leave character widths out of your Printwheel Description altogether: then you will automatically retain the PS widths defined in the GB Printwheel table for those characters which are on both your wheel and the GB wheel; and any other character will be given a PS width of 5.

Note: We recommend that you always give the space character and the digits 0...9 the same PS width (typically $^{5}/_{60}$", so that the spaces correspond to 12-pitch spaces). This will ensure that tables of numbers print neatly. By the same token, you should also always set this width for =, < and >, the three characters used in LocoScript to reserve space for page numbers. The positions of the digits of page numbers are worked out using the widths of the =, < and > characters, not the widths of the page numbers themselves.

For example, suppose the first few characters on the PS version of the wheel are 0, =, M, " and # and that these have widths of 5, 5, 7, 3 and 6 respectively. The definitions for these characters in your Printwheel Description file would be:

```
1    "0"     5↵
2    =       5↵
3    M       7↵
4    !"      3↵
5    Hash    6↵
```

The Substitutions section

Only include this section if the standard substitutions (in Appendix II) aren't sufficient.

The characters that you can print at any time are essentially limited to the ones defined in the Character Set you are using, and where a document contains a LocoScript character that isn't defined in the Character Set, LocoScript will leave a blank for you to fill in the character later – by hand, if necessary.

The Fixed and PS sections of the Printwheel Description file associate one character with each petal of the printwheel and so, in theory, these are the only characters you can print when using this printwheel. However, there are a number of characters in the LocoScript character set which are the same shape or at least very similar. For example, a dash is just a slightly longer hyphen while a Greek upper-case alpha is identical to an upper-case A. In other words, there is plenty of opportunity for substituting characters on the wheel for 'missing' characters.

Built into the MKWHEEL program are details of a number of possible substitutions of a character on the wheel for another character in the LocoScript 2 character set. These are known as the Standard Substitutions and details are given in Appendix II. This section of the Printwheel Description file lets you set up further substitutions which will be used alongside the standard set. It can be used both to increase the range of characters that can be printed and to specify alternative substitutions for characters already covered by the standard substitutions: the substitutions listed in the Substitutions section are always used in preference to any corresponding standard substitution.

The characters defined through the Substitutions section fall into two categories:

- characters in the LocoScript character set that are identical or at least similar to one of the characters on the printwheel (such as Alpha-Upper and A), and
- characters that can be printed by printing an accent on top of another character. (For example, ≠ can be printed from = and a Stroke accent.)

Note: You don't need to use the Substitutions section to define the 'normal' accented characters: they are automatically taken care of by LocoScript.

The section starts with the keyword `SUBSTITUTIONS` on a line on its own. This is followed by up to 256 definitions as follows:

character-1 *character-2* ↵

where *character-1* is the character in the LocoScript character set which is not specifically included on the wheel and *character-2* is the character on the printwheel that should be used instead. The usual rules apply for specifying these LocoScript characters.

For example, if your printwheel doesn't have a ≈ ('Nearly-equals' sign) but does have a ~ ('Asymptotically-equal' sign), then you might set up the substitution:

```
Nearly_equal_to  Asymptotically_equal_to ↵
```

Where two of the printwheel characters are involved, then you give the definition as:

character-1 *accent* + *character-2* ↵

For example, if your printwheel has an = and a /, then you can define ≠ as follows:

```
Not_equal    / + =↵
```


The CP/M section

This section of the Printwheel Description file may be omitted if you don't wish to use wheels of this family while running CP/M programs.

When you are running the CP/M operating system, the codes that are used to identify the character that is to be printed are the character codes &21...&7E, together with the special escape sequences Esc H, Esc I, Esc J, Esc K, Esc Y and Esc Z (represented here by the codes &80...&85).

Each Printwheel table that is set up for use with CP/M associates characters on the printwheel with these 100 codes. In theory you could associate any character on the wheel with any code, but wherever possible the code used for each character on the printer should be the same as that used by CP/M to display the character on the screen.

In the main, the CP/M character set follows the ASCII standard in which the codes &20...&7E are used to represent the letters A...Z, a...z, 0...9 and the common punctuation marks. Where these characters also appear on the printwheel, the code to use is the standard ASCII code. The complication is that there are a number of different language versions of CP/M in which some of the codes have different meanings. For example, &5D represents] in the English version of CP/M but Ü in the German version.

These language differences affect the way certain character codes are interpreted both in your programs and at the printer. The characters associated with 82 of the codes used by the printer are the same whichever language the printer is set to use but 18 of the codes can be associated with different characters depending on which language is selected. As a result, the CP/M section of the Printwheel Description file is itself split into a number of subsections. The first of these defines the characters associated with the 82 codes common to all the languages (&21, &22, &25...&3F, &41...&5A, &5F, &61...&7A). The others define the language-variant codes, each subsection covering a different language.

The language version of CP/M being used for your programs and the language your printer works in are set independently but again it is best to ensure as far as possible that the character on the printwheel accessed by one of the language-variant codes is the one displayed on the screen when the same language is selected.

Details of codes used by CP/M are given in the table in Appendix I.1.1 of your PCW9512 User Guide. This lists the code for each character in the character set when you are using the CP/M language variant that matches the nationality of your machine. For example, the table in the UK version of this Guide lists the codes used when the UK language (Language 3) is selected. If you only use software of the same nationality as your machine, then these are the only codes you need to worry about (you can simply set all the other languages to be the same). If you use programs of other nationalities, then you need to know which characters are represented by the language-dependent codes in the corresponding language-version of CP/M: this information is given in the table on page 41).

Combine this information into a table, such as that shown overleaf.

Where CP/M uses one of the codes #21....#7E for a character on your printwheel, this is the code to use – either generally where the code isn't one of the language variant ones or in the corresponding language list. When you have slotted in this information, select the codes to use for the remaining characters on the wheel.

Example table

Hex	2	3	4	5	6	7
0		0		P		p
1	!	1	A	Q	a	q
2	"	2	B	R	b	r
3		3	C	S	c	s
4		4	D	T	d	t
5	%	5	E	U	e	u
6	&	6	F	V	f	v
7	'	7	G	W	g	w
8	(	8	H	X	h	x
9	)	9	I	Y	i	y
A	*	:	J	Z	j	z
B	+	;	K		k	
C	,	<	L		l	
D	-	=	M		m	
E	.	>	N		n	
F	/	?	O		o	

	0 USA	1 France	2 German	3 UK	4 Denmark	5 Sweden	6 Italy	7 Spain	8 Japan
&23	#	#	#	£	#	#	#	#	#
&24	$	$	$	$	$	$	$	$	$
&40	@	@	@	@	@	@	@	@	@
&5B	[	°	[	[	[	[	°	[	[
&5C	½	½	½	½	½	½	½	½	½
&5D	]	]	]	]	]	]	]	]	]
&5E	°	[	°	°	°	°	[	°	°
&60	+	+	+	+	+	+	+	+	+
&7B	¼	¼	¼	¼	¼	¼	¼	¼	¼
&7C	|	|	|	|	|	|	|	|	|
&7D	¾	¾	¾	¾	¾	¾	¾	¾	¾
&7E	£	£	£	#	£	£	£	£	£
&80	.	.	.	.	.	.	.	.	.
&81	,	,	,	,	,	,	,	,	,
&82	²	²	²	²	²	²	²	²	²
&83	³	³	³	³	³	³	³	³	³
&84	⅓	⅓	⅓	⅓	⅓	⅓	⅓	⅓	⅓
&85	⅔	⅔	⅔	⅔	⅔	⅔	⅔	⅔	⅔

The CP/M section starts with the keyword `CPM` on a line on its own, followed by definitions of the 82 codes common to all languages. This is followed by up to nine 'subsections' defining the 18 language-dependent codes, one for each language you use. Each subsection starts with a line made up of the keyword `LANGUAGE` and, optionally, the number of a language (0...8). If no number is given, all the language variants are given this definition; with a number, the definition is applied to just this particular language. If you don't want to set all nine languages individually, you should start with the definition that applies to most languages below the heading `LANGUAGE` and then follow this with separate LANGUAGE *x* sections that define the differences.

Throughout this section, the lines defining the codes have the form:

code *character*↵

each of which defines the *character* that is associated with this particular CP/M *code*. Each *character* is specified:

- either by the character itself (where appropriate)
- or by its *MKWHEEL* name

ie. in exactly the same way as they were in the Fixed Pitch section of the file. The *codes* are specified:

- either by the code itself
- or by the *ASCII* character corresponding to this code

You can only use ASCII characters for the codes which are common to all the different languages. Of these, the numbers 0...9 are written between double-quote marks and there are further special forms for writing characters that have special meaning in the Printwheel Description file. The way to specify the CP/M codes is given on pages 41–42. **Note:** If you did as we suggested and created your Printwheel Description file in group 3 (the PROGRAMS group) of the Printwheels disc you will find all these codes already in place in the file. All you have to do is to fill in the character details alongside each code.

The information to be typed into your Printwheel Description file can be taken directly from your table, the top part of which details the 82 codes that aren't affected by changes of language. Simply type each CP/M code (in the way recommended in the table on page 42) and then type the character. For example, if the beginning of your table starts:

```
&21     !
&22     "
&25     ^
&26     &
&27     '
&28     (
```

then the start of the CP/M section of the Printwheel Description file will be:

```
!!      !!↵            } ! and " have special meaning
!"      !"↵            }
%       Circumflex↵      ^ has to be specified either by its name or by
                           its LocoScript code
"&"     "&"↵             & has special meaning
'       '↵
(       (↵
```

Note: It is not at all unusual for this set of definitions to feature the same details in both columns. This is simply a result of the way CP/M follows the ASCII standard in its character set. In fact the differences in the two columns pick out the places where the ASCII character with this code is not on the printwheel and so you have used the code to represent some other character.

The language sections that follow make very little use of the characters themselves. Instead the CP/M code will probably be written as a hexadecimal number and the character will probably be specified by its MKWHEEL name.

For example, suppose the language-variant codes correspond to the following characters when Language 0 (American) is selected:

&23		#
&24		$
&40		@
&5B		[
&5C		½
&5D		]
&5E		°
&60		+
&7B		¼
&7C		|
&7D		¾
&7E		£
Esc H	(&80)	Duplicate full stop
Esc I	(&81)	Duplicate comma
Esc J	(&82)	²
Esc K	(&83)	³
Esc Y	(&84)	⅓
Esc Z	(&85)	⅔

The corresponding section of the Printwheel Description file would be

```
Language 0 ↵
&23     Hash ↵
&24     Dollar ↵
&40     At ↵
&5B     Open_square_bracket ↵
&5C     Half ↵
&5D     Close_square_bracket ↵
&5E     Degrees ↵
&60     Divide ↵
&7B     One_quarter ↵
&7C     Vertical_bar ↵
&7D     Three_quarters ↵
&7E     Pound ↵
&80     Dup . ↵                    } Duplicate characters
&81     Dup "." ↵                  }
&82     Superscript_2 ↵
&83     Superscript_3 ↵
&84     One_third ↵
&85     Two_thirds ↵
```


9010

Expressing CP/M codes

The codes in the CP/M part of the Printwheel Description file are specified:

- either by the code itself
- or by the *ASCII* character corresponding to this code

The codes that are used are numbers in the range 0..255. These numbers are also referred to as the 'values' of the codes.

The way to write each code is as & followed by the digits of the hexadecimal number. The & simply signals that the value is a hexadecimal number.

The digits of a hexadecimal number take values in the range 0...15 (just as the digits of a decimal number take values in the range 0...9). These digits are expressed as 0...9 and A...F (or a...f), where A represents the value 10, B the value 11, and so on. The CP/M codes are all two-digit hexadecimal numbers.

Just as a two-digit decimal number is the value of the first digit times ten plus the value of the second digit, a two-digit hexadecimal number is the value of the first digit times sixteen plus the value of the second digit. For example, the hexadecimal number E7 is

```
E * 16 + 7 = 14 * 16 + 7 = 231
```

Language-dependent CP/M characters

The following table gives details of the language-dependent CP/M codes and the characters displayed on the screen when the different languages are selected. (*Note:* There isn't a Japanese language set.)

Hex	US	French	German	UK	Danish	Swedish	Italian	Spanish
&23	#	#	#	£	#	#	#	Pt
&24	$	$	$	$	$		$	$
&40	@	à	§	@	@	É	@	@
&5B	[	°	Ä	|	Æ	Ä	°	¡
&5C	\	ç	Ö	\	Ø	Ö	\	Ñ
&5D	]	§	Ü	}	Å	Å	é	¿
&5E	^	^	^	^	^	Ü	^	^
&60	`	`	`	`	`	é	ù	`
&7B	{	é	ä	{	æ	ä	à	¨
&7C	|	ù	ö	|	ø	ö	ò	ñ
&7D	}	è	ü	}	å	å	è	}
&7E	~	¨	ß	~	~	ü	ì	~

CP/M codes

When you are running the CP/M operating system, characters to be printed are identified by the character codes &21...&7E, together with the codes &80...&85 representing Esc H, Esc I, Esc J, Esc K, Esc Y and Esc Z. The way to represent these codes in the Printwheel Description file is as follows:

Dec		32	48	64	80	96	112	128
	Hex	2	3	4	5	6	7	8
0	0		"0"	&40	P	&60	p	&80
1	1	!!	"1"	A	Q	a	q	&81
2	2	!"	"2"	B	R	b	r	&82
3	3	&23	"3"	C	S	c	s	&83
4	4	&24	"4"	D	T	d	t	&84
5	5	%	"5"	E	U	e	u	&85
6	6	"&"	"6"	F	V	f	v	
7	7	'	"7"	G	W	g	w	
8	8	(	"8"	H	X	h	x	
9	9	)	"9"	I	Y	i	y	
10	A	*	:	J	Z	j	z	
11	B	+	";"	K	&5B	k	&7B	
12	C	","	<	L	&5C	l	&7C	
13	D	-	=	M	&5D	m	&7D	
14	E	.	>	N	&5E	n	&7E	
15	F	/	?	O	_	o		

Making the file ready for use

So far, you have created a LocoScript document containing the information that you want in the Printwheel Description file. Before this description can be used either with MKWHEEL to produce the appropriate Character Set file or with SETWHEEL to use the wheel under CP/M, the Printwheel Description file has to be converted into a simple text file. The steps are as follows:

1 When you return to the Disc Manager Screen, check that the File cursor is still on the Printwheel Description document you have just prepared, then press [f1].

2 Select the `Make ASCII file` option and press [ENTER].

3 Check that the Group cursor is on group 3 (the PROGRAMS group) of the Printwheels disc; then press [ENTER] again.

   You have to pick out group 3 so that the MKWHEEL and SETWHEEL programs in this group can access the file under CP/M.

4 When the menu appears, set a new name for the ASCII file.

   For example, if the LocoScript Printwheel Description file is called SCRIPT.#SC, you might call the ASCII version of this file ASCII.#SC. We particularly recommend choosing names of the form *XXXXXXXX.#XX* because then when you come to make the Character Set file, you can use the short form of the MKWHEEL command (see 'Creating the file to use with LocoScript' overleaf).

   Note: If you don't set a new name here, you will lose the LocoScript version of this file – which will cause you problems if it turns out that you need to make any changes.

5 Check that `Simple text file` is currently selected at the bottom of the menu and then press [ENTER].

   The Printwheel Description file you require will then be created and stored in group 3 on the Printwheels disc.

6 Use the f8 Options menu to display Limbo files and then use the `Erase file` option in the f3 File menu to erase a few of these.

   If you don't do this, your disc may seem full when you come to use it with the MKWHEEL program. CP/M doesn't have the same system as LocoScript for clearing out Limbo files in order to make room for new files.

The file you have just created is exactly like the Printwheel Description files that were supplied on the Printwheels Disc. If you wish to use the printwheel described in the file with CP/M, you simply have to use this file with the SETWHEEL program as described in the Printer Support Pack booklet. But before you can use the printwheel to print LocoScript documents, you will need to use the MKWHEEL program to create a Character Set file for this printwheel and then install it. The steps to take are given below.

9010
9010

Creating the file to use with LocoScript

Load CP/M.

Insert the Character Sets disc and press [ALT] C

Type the command: `USER 3` [RETURN]; the system prompt should become 3A>.

This sets the current CP/M User Number to User Number 3. MKWHEEL and the Printwheel Description files are stored in group 3 on the Printwheels disc and so have User Number 3.

If there may be errors in the Printwheel Description file you are using, press [ALT] P.

This tells CP/M to 'echo' anything it displays on the screen on your printer – giving you a printed list of the errors MKWHEEL finds.

If your Printwheel Description file is called *XXXXXX.#xx*, type the command:

`MKWHEEL` *XXXXXXXX.#xx* [RETURN]

Otherwise, type the command:

`MKWHEEL PCW9512.#`*xx=printwheel-filename* [RETURN]

where `PCW9512.#`*xx* is the name of the Character Set file you want to create and *printwheel-filename* is the name of your ASCII Printwheel Description file.

For example, if your ASCII is called file ASCII.#SC, then give the command:

`MKWHEEL ASCII.#SC` [RETURN]

If you called this file SERBO_CT.ASC, the command you need might be:

`MKWHEEL PCW9512.#SC=SERBO_CT.ASC` [RETURN]

Both of these instructions will produce Character Set files called PCW9512.#SC.

Note: Be careful to choose a name for your Character Set file that you haven't used before when creating a different Character Set file: otherwise, you will simply overwrite your current Character Set file with your new one.

If appropriate, clear the echoing to the printer by pressing [ALT] P again.

The MKWHEEL program reads through the Printwheel Definition file, compiling the Printwheel table LocoScript needs from the information in the file. Provided there aren't any errors, it simply stores the finished table in the PCW9512.#*xx* file you specified (overwriting any file of this name already on the disc) and returns you to the 3A> system prompt. The new file is stored in group 3 on the Printwheels disc.

If any errors are found, no Character Set file is made. Instead, the program displays a list of these errors on the screen (possibly echoed on the printer). The errors are given in the order that MKWHEEL found them (and thus in the order they appear in your Printwheel Description file). Two lines of the display are given over to each error: the first line describes the type of error that has been found and the second gives you the position of the error. For example, if you had specified a PS width of 9 for the W character on petal number 49, you would see:

```
Error: PS width must be in the range 3...8
Error position is: 49 W    ????9
```

The `???` group is used to point out where MKWHEEL found the error. Details of the error messages and the likely cause of the problem are given in Appendix IV. Correct these errors on the LocoScript version of the file.

If more than five errors are found, MKWHEEL doesn't display all the errors because typically a number of the errors would be just side-effects of one real error. In particular, because MKWHEEL works through substituting new values in an existing Printwheel table, a simple typing error in one place can lead to MKWHEEL also reporting a 'missing' or a 'duplicate' error.

So instead of trying to sort out problems individually, simply work through the LocoScript version of the file:

- **first** checking that you have all the section keywords you need;
- **then** correcting mis-spellings and any hammer intensities or PS widths that are out of range:
- **then** try again (ie. make a new ASCII version and run this with MKWHEEL).

When no more than five errors are reported, it is unlikely that any of the section keywords are missing and you will probably just have to correct the odd mispelling of a MKWHEEL name.

Running MKWHEEL always produces a file exactly like the Character Set files that were supplied on the Printwheels Disc. To make the printwheel described in the file available from LocoScript, you just need to install it as described in Part II of the PCW External Printers Guide. However, before you start setting up many documents for this Character Set, it is wise to test your new file as described below.

Testing the new Character Set file

Before you start setting up documents for your new Character Set, you should check that the right characters are being printed. (If you set up documents and then discover a mistake, you will have to set up all these documents for the Character Set all over again after making your correction, in order to record the correct character information in the document.)

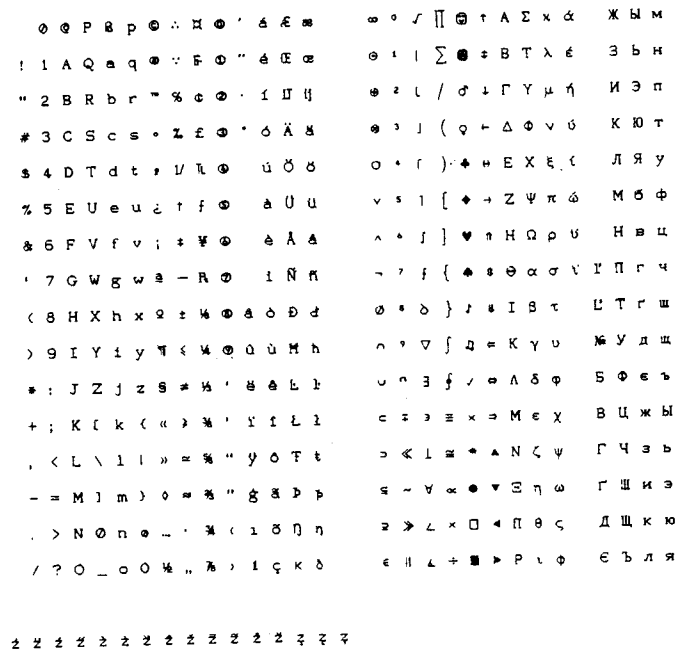
We have supplied two documents that you might use for testing your Character Set on the disc: these are the documents called ALLCHARS and PWHLCHAR stored in group 3 (the PROGRAMS group) on the Printwheels disc. The ALLCHARS document contains a copy of every character in the LocoScript 2 character set: PWHLCHAR contains a table of LocoScript character names.

If you print either of these documents on the built-in printer (fitted with the appropriate printwheel) and with LocoScript set up for your new Character Set, the characters that are printed will be the ones for which you have given definitions in the Printwheel Description file plus any 'Standard substitutions' that have been set up: all the other characters will be replaced by blanks.

With ALLCHARS, you can see whether your Character Set file is enabling you to print the right characters, simply by comparing the print out you get with Figure 1 overleaf: this shows the result of printing ALLCHARS on the PCW8256/8512's built-in printer which can print every character in the LocoScript 2 character set. With PWHLCHAR, you should be able to see if anything is wrong by just looking at the table it produces.

The way to set up LocoScript for your new Character Set for this test is to make your Character Set the Current Character Set within Printer Control State. So the steps to testing your Character Set are:

Figure 1: ALLCHARS printed on the PCW8256/8512 printer



1 Fit the printwheel in the built-in printer.

2 Press [PTR] to go into Printer Control State.

3 Use the f5 Printer menu to ensure that the Current printer is PCW9512 and to change the Current Character Set and Character Style to the ones corresponding to your printwheel. (This is described in the section on 'Setting up LocoScript for a particular printer' in Part III of the PCW External Printers Guide.)

4 Press [EXIT] to leave Printer Control State.

5 Insert the Printwheels disc (press [f7] to signal the disc change) and then print one copy of ALLCHARS or PWHLCHAR. (Draft quality print will do.)

6 Examine the printed document.

If any characters are not being printed correctly, correct the LocoScript version of your Printwheel Description file. Then when the new version is ready, make an ASCII version of this file and then use MKWHEEL to create a new version of this file. Copy the new file to group 0 of your Start-of-day disc.

Important: If you produce a new version of the Character Set file after you have started using it and you either add, remove or change any line of the file which contains PS information, you must record fresh PS width details in each document that is set up for this Character Set. This means setting up the document for the printer all over again, except that when you go into Document Set-up, you only have to bring the f6 Printing menu onto the screen and select its EXIT option. If you change the name of the Character Set itself, however, you will have to re-install the Character Set all over again.

Appendix I: LocoScript character names

Note: Where the character name is given either as the character itself, or within double quotes or as *!character*, you must copy this name exactly. Where the name is descriptive (eg. ALPHA_LOWER), any combination of upper and lower case characters can be used.

The character names

Description in User Guide	CHARKIT name	MKWHEEL name
I: Main range of characters		
Alphanumerics		
Lower case letters	A LOWER ... Z LOWER	a...z
Upper case letters	A UPPER ... Z UPPER	A...Z
Numerals	ZERO	"0"
	ONE	"1"
	TWO	"2"
	THREE	"3"
	FOUR	"4"
	FIVE	"5"
	SIX	"6"
	SEVEN	"7"
	EIGHT	"8"
	NINE	"9"
Zero (no slash)	ZERO (NO SLASH)	ZERO_WITHOUT_SLASH
ae diphthong	AE DIPHTHONG LOWER	AE_DIPHTHONG_LOWER
AE diphthong	AE DIPHTHONG UPPER	AE_DIPHTHONG_UPPER
oe diphthong	OE DIPHTHONG LOWER	OE_DIPHTHONG_LOWER
OE diphthong	OE DIPHTHONG UPPER	OE_DIPHTHONG_UPPER
Greek Characters		
Alpha	ALPHA LOWER	ALPHA_LOWER
	ALPHA UPPER	ALPHA_UPPER
Beta	BETA LOWER	BETA_LOWER
	BETA UPPER	BETA_UPPER
Gamma	GAMMA LOWER	GAMMA_LOWER
	GAMMA UPPER	GAMMA_UPPER
Delta	DELTA LOWER	DELTA_LOWER
	DELTA UPPER	DELTA_UPPER
Epsilon	EPSILON LOWER	EPSILON_LOWER
	EPSILON UPPER	EPSILON_UPPER
Zeta	ZETA LOWER	ZETA_LOWER
	ZETA UPPER	ZETA_UPPER
Eta	ETA LOWER	ETA_LOWER
	ETA UPPER	ETA_UPPER

9010 9010

Description in User Guide	CHARKIT name	MKWHEEL name
Theta	THETA LOWER	THETA_LOWER
	THETA UPPER	THETA_UPPER
Iota	IOTA LOWER	IOTA_LOWER
	IOTA UPPER	IOTA_UPPER
Kappa	KAPPA LOWER	KAPPA_LOWER
	KAPPA UPPER	KAPPA_UPPER
Lambda	LAMBDA LOWER	LAMBDA_LOWER
	LAMBDA UPPER	LAMBDA_UPPER
Mu	MU LOWER	MU_LOWER
	MU UPPER	MU_UPPER
Nu	NU LOWER	NU_LOWER
	NU UPPER	NU_UPPER
Xi	XI LOWER	XI_LOWER
	XI UPPER	XI_UPPER
Omicron	O LOWER	o
	O UPPER	O
Pi	PI LOWER	PI_LOWER
	PI UPPER	PI_UPPER
Rho	RHO LOWER	RHO_LOWER
	RHO UPPER	RHO_UPPER
Sigma	SIGMA LOWER	SIGMA_LOWER
(Alternative form)	SIGMA (ENDING)	SIGMA_ENDING
	SIGMA UPPER	SIGMA_UPPER
Tau	TAU LOWER	TAU_LOWER
	TAU UPPER	TAU_UPPER
Upsilon	UPSILON LOWER	UPSILON_LOWER
	UPSILON UPPER	UPSILON_UPPER
Phi	PHI LOWER	PHI_LOWER
(Alternative form)	PHI (SCIENTIFIC)	PHI_SCIENTIFIC
	PHI UPPER	PHI_UPPER
Chi	CHI LOWER	CHI_LOWER
	CHI UPPER	CHI_UPPER
Psi	PSI LOWER	PSI_LOWER
	PSI UPPER	PSI_UPPER
Omega	OMEGA LOWER	OMEGA_LOWER
	OMEGA UPPER	OMEGA_UPPER
Digamma	DIGAMMA LOWER	DIGAMMA_LOWER
	DIGAMMA UPPER	DIGAMMA_UPPER
Qoppa	QOPPA LOWER	QOPPA_LOWER
	QOPPA UPPER	QOPPA_UPPER
Lunate sigma	LUNATE SIGMA LOWER	LUNATE_SIGMA_LOWER
	LUNATE SIGMA UPPER	LUNATE_SIGMA_UPPER
Thousands mark	THOUSANDS MARK	THOUSANDS_MARK
Sampi	SAMPI	SAMPI
Stigma or numerical digamma	STIGMA	STIGMA
Alpha with iota subscript	ALPHA IOTA SUBSCRIPT LOWER	ALPHA_IOTA_SUBSCRIPT_LOWER
Eta with iota subscript	ETA IOTA SUBSCRIPT LOWER	ETA_IOTA_SUBSCRIPT_LOWER
Omega with iota subscript	OMEGA IOTA SUBSCRIPT LOWER	OMEGA_IOTA_SUBSCRIPT_LOWER

9010

Description in User Guide	CHARKIT name	MKWHEEL name
Cyrillic characters		
As or A	A LOWER	a
	A UPPER	A
Buki or B	B CYRILLIC LOWER	B_CYRILLIC_LOWER
	B CYRILLIC UPPER	B_CYRILLIC_UPPER
Vyedi or V	V CYRILLIC LOWER	V_CYRILLIC_LOWER
	V CYRILLIC UPPER	V_CYRILLIC_UPPER
Glagol or G; H (BR, U)	G CYRILLIC LOWER	G_CYRILLIC_LOWER
	G CYRILLIC UPPER	G_CYRILLIC_UPPER
Glagol (Ukraine) or G (BR, U)	G UKRAINIAN LOWER	G_UKRAINIAN_LOWER
	G UKRAINIAN UPPER	G_UKRAINIAN_UPPER
Dobro or D	D CYRILLIC LOWER	D_CYRILLIC_LOWER
	D CYRILLIC UPPER	D_CYRILLIC_UPPER
Yest or E; Ye (BR, R)	E LOWER	e
	E UPPER	E
Yest (Ukraine) or Ye(U)	YE UKRAINIAN LOWER	YE_UKRAINIAN_LOWER
	YE UKRAINIAN UPPER	YE_UKRAINIAN_UPPER
Zhivete or Zh	ZH LOWER	ZH_LOWER
	ZH UPPER	ZH_UPPER
Zemla or Z	Z CYRILLIC LOWER	Z_CYRILLIC_LOWER
	Z CYRILLIC UPPER	Z_CYRILLIC_UPPER
Ishe or I (Blg, R); Y (U)	I CYRILLIC LOWER	I_CYRILLIC_LOWER
	I CYRILLIC UPPER	I_CYRILLIC_UPPER
Ishe Breve or Y (postvocalic)	Use "BREVE + "I CYRILLIC LOWER";	Use BREVE + ISHE_LOWER;
	"BREVE + "I CYRILLIC UPPER"	BREVE + ISHE_UPPER
I (Ukraine) or I (BT,U)	I LOWER	i
	I UPPER	I
Kako or K	K CYRILLIC LOWER	K_CYRILLIC_LOWER
	K CYRILLIC UPPER	K_CYRILLIC_UPPER
Liudi or L	L CYRILLIC LOWER	L_CYRILLIC_LOWER
	L CYRILLIC UPPER	L_CYRILLIC_UPPER
Muislete or M	M CYRILLIC LOWER	M_CYRILLIC_LOWER
	M CYRILLIC UPPER	M_CYRILLIC_UPPER
Nash or N	N CYRILLIC LOWER	N_CYRILLIC_LOWER
	N CYRILLIC UPPER	N_CYRILLIC_UPPER
On or O	O LOWER	o
	O UPPER	O
Pokoi or P	P CYRILLIC LOWER	P_CYRILLIC_LOWER
	P CYRILLIC UPPER	P_CYRILLIC_UPPER
Rtsui or R	P LOWER	p
	P UPPER	P
Slovo or S	C LOWER	c
	C UPPER	C
Tverdo or T	T CYRILLIC LOWER	T_CYRILLIC_LOWER
	T CYRILLIC UPPER	T_CYRILLIC_UPPER
U	U CYRILLIC LOWER	U_CYRILLIC_LOWER
	U CYRILLIC UPPER	U_CYRILLIC_UPPER
Fert or F	F CYRILLIC LOWER	F_CYRILLIC_LOWER
	F CYRILLIC UPPER	F_CYRILLIC_UPPER
Kherr or Kh	X LOWER	x
	X UPPER	X

Description in User Guide	CHARKIT name	MKWHEEL name
Tsui or Ts	TS LOWER	TS_LOWER
	TS UPPER	TS_UPPER
Tsherv or Ch	CH LOWER	CH_LOWER
	CH UPPER	CH_UPPER
Sha or Sh	SH LOWER	SH_LOWER
	SH UPPER	SH_UPPER
Shtsha or Shch; Sht (Blg)	SHCH LOWER	SHCH_LOWER
	SHCH UPPER	SHCH_UPPER
Yerr or hard sign	HARD SIGN LOWER	HARD_SIGN_LOWER
	HARD SIGN UPPER	HARD_SIGN_UPPER
Yerui or Y (postcons BR, R)	Y (POSTCONSONANTAL) LOWER	Y_POSTCONSONANTAL_LOWER
	Y (POSTCONSONANTAL) UPPER	Y_POSTCONSONANTAL_UPPER
Yer or soft sign	SOFT SIGN LOWER	SOFT_SIGN_LOWER
	SOFT SIGN UPPER	SOFT_SIGN_UPPER
E	E RUSSIAN LOWER	E_RUSSIAN_LOWER
	E RUSSIAN UPPER	E_RUSSIAN_UPPER
Yu	YU LOWER	YU_LOWER
	YU UPPER	YU_UPPER
Ya	YA LOWER	YA_LOWER
	YA UPPER	YA_UPPER
Number	NUMBER	NUMBER

Other special language characters

d'	D APOSTROPHE LOWER	D_APOSTROPHE_LOWER
D stroke	D STROKE LOWER	D_STROKE_LOWER
	D STROKE UPPER	D_STROKE_UPPER
Eth	ETH	ETH
H stroke	H STROKE LOWER	H_STROKE_LOWER
	H STROKE UPPER	H_STROKE_UPPER
Dotless i (Turkish)	DOTLESS I	DOTLESS_I
IJ	IJ LOWER	IJ_LOWER
	IJ UPPER	IJ_UPPER
k (greenland)	K (GREENLAND)	K_GREENLAND
L dot	L DOT LOWER	L_DOT_LOWER
	L DOT UPPER	L_DOT_UPPER
L'	L APOSTROPHE LOWER	L_APOSTROPHE_LOWER
	L APOSTROPHE UPPER	L_APOSTROPHE_UPPER
L stroke	L STROKE LOWER	L_STROKE_LOWER
	L STROKE UPPER	L_STROKE_UPPER
Curly l (lower case only)	CURLY L	CURLY_L
Eng	ENG LOWER	ENG_LOWER
	ENG UPPER	ENG_UPPER
O stroke	O STROKE LOWER	O_STROKE_LOWER
	O STROKE UPPER	O_STROKE_UPPER
Scharfes S (lower case only)	SCHARFES S	SCHARFES_S
Thorn	THORN LOWER	THORN_LOWER
	THORN UPPER	THORN_UPPER
t'	T APOSTROPHE LOWER	T_APOSTROPHE_LOWER
T stroke	T STROKE LOWER	T_STROKE_LOWER
	T STROKE UPPER	T_STROKE_UPPER

Description in User Guide	CHARKIT name	MKWHEEL name

Textual symbols

Hyphen	HYPHEN	-
Comma	COMMA	","
Full stop	FULL STOP	.
Semicolon	SEMICOLON	";"
Colon	COLON	:
Ellipsis	ELLIPSIS	ELLIPSIS
Dash	DASH	DASH
Exclamation mark	EXCLAMATION MARK	!! or EXCLAMATION_MARK
Question mark	QUESTION MARK	?
Ampersand	AMPERSAND	"&"
Apostrophe	APOSTROPHE	'
Open single quote	OPEN SINGLE QUOTE or USER SYMBOL A	OPEN_SINGLE_QUOTE or USER_SYMBOL_A
Close single quote	CLOSE SINGLE QUOTE or USER SYMBOL B	CLOSE SINGLE_QUOTE or USER_SYMBOL_B
Double quote	DOUBLE QUOTE	!" or DOUBLE_QUOTE
Open double quote	OPEN DOUBLE QUOTE or USER SYMBOL C	OPEN_DOUBLE_QUOTE or USER_SYMBOL_C
Close double quote	CLOSE DOUBLE QUOTE or USER SYMBOL D	CLOSE_DOUBLE_QUOTE or USER_SYMBOL_D
Underline	UNDERLINE	_
Asterisk	ASTERISK	*
Hash	HASH	HASH
Slash	SLASH	/
Open parenthesis	OPEN PARENTHESIS	(
Close parenthesis	CLOSE PARENTHESIS	)
Open square bracket	OPEN SQUARE BRACKET	OPEN_SQUARE_BRACKET
Close square bracket	CLOSE SQUARE BRACKET	CLOSE_SQUARE_BRACKET
Open curly brace	OPEN CURLY BRACKET	OPEN_CURLY_BRACKET
Close curly brace	CLOSE CURLY BRACKET	CLOSE_CURLY_BRACKET
At	AT	AT
Therefore	THEREFORE	THEREFORE
Because of	BECAUSE OF	BECAUSE_OF
Care of	CARE OF	CARE_OF
Copyright	COPYRIGHT	COPYRIGHT
Registered	REGISTERED	REGISTERED
Trade mark	TRADE MARK	TRADE_MARK
Open circle	OPEN CIRCLE	OPEN_CIRCLE
Bullet	BULLET	BULLET
Paragraph (Pilcrow)	PARAGRAPH	PARAGRAPH
Section	SECTION	SECTION
Dagger	DAGGER	DAGGER
Double dagger	DOUBLE DAGGER	DOUBLE_DAGGER
Vertical bar	VERTICAL BAR	VERTICAL_BAR

Description in User Guide	CHARKIT name	MKWHEEL name
Continental textual symbols		
Open single guillemet	OPEN SINGLE GUILLEMET or USER SYMBOL E	OPEN_SINGLE_GUILLEMET or USER_SYMBOL_E
Close single guillemet	CLOSE SINGLE GUILLEMET or USER SYMBOL F	CLOSE_SINGLE_GUILLEMET or USER_SYMBOL_F
Open double guillemet	OPEN DOUBLE GUILLEMET	OPEN_DOUBLE_GUILLEMET
Close double guillemet	CLOSE DOUBLE GUILLEMET	CLOSE_DOUBLE_GUILLEMET
German open double quote	GERMAN OPEN DOUBLE QUOTE	GERMAN_OPEN_DOUBLE_QUOTE
Open query	OPEN QUERY	OPEN_QUERY
Open shriek	OPEN SHRIEK	OPEN_SHRIEK
Feminine ordinal	FEMININE ORDINAL	FEMININE_ORDINAL
Masculine ordinal	MASCULINE ORDINAL	MASCULINE_ORDINAL
Greek semicolon	GREEK SEMICOLON	GREEK_SEMICOLON
Greek thousands mark	GREEK THOUSANDS MARK	GREEK_THOUSANDS_MARK
Currency symbols		
Pound	POUND	POUND
Dollar	DOLLAR	DOLLAR
Cent	CENT	CENT
International currency symbol	INTERNATIONAL CURRENCY SYMBOL	INTERNATIONAL_CURRENCY_SYMBOL
Franc	FRANC	FRANC
Turkish pound	TURKISH POUND	TURKISH_POUND
Florin	FLORIN	FLORIN
Yen	YEN	YEN
Peseta	PESETA	PESETA
Mathematical and Technical symbols		
Plus	PLUS	+
Minus	HYPHEN	-
Multiply	ASTERISK	*
Times	TIMES	TIMES
Divide	DIVIDE	DIVIDE
Divide (Slash)	SLASH	/
Exponentiation	UP ARROW	UP_ARROW
Equals	EQUALS	=
Modulus	MODULUS	MODULUS
Backslash	BACKSLASH	BACKSLASH
Half	HALF	HALF
One eighth	ONE EIGHTH	ONE_EIGHTH
One quarter	ONE QUARTER	ONE_QUARTER
One third	ONE THIRD	ONE_THIRD
Three eighths	THREE EIGHTHS	THREE_EIGHTHS
Five eighths	FIVE EIGHTHS	FIVE_EIGHTHS
Two thirds	TWO THIRDS	TWO_THIRDS
Three quarters	THREE QUARTERS	THREE_QUARTERS
Seven eighths	SEVEN EIGHTHS	SEVEN_EIGHTHS
One upon	ONE UPON	ONE_UPON

Description in User Guide	CHARKIT name	MKWHEEL name
Superscript numerals	SUPERSCRIPT 0	SUPERSCRIPT_0
	SUPERSCRIPT 1	SUPERSCRIPT_1
	SUPERSCRIPT 2	SUPERSCRIPT_2
	SUPERSCRIPT 3	SUPERSCRIPT_3
	SUPERSCRIPT 4	SUPERSCRIPT_4
	SUPERSCRIPT 5	SUPERSCRIPT_5
	SUPERSCRIPT 6	SUPERSCRIPT_6
	SUPERSCRIPT 7	SUPERSCRIPT_7
	SUPERSCRIPT 8	SUPERSCRIPT_8
	SUPERSCRIPT 9	SUPERSCRIPT_9
Superscript n	SUPERSCRIPT N	SUPERSCRIPT_N
Decimal point	DECIMAL POINT	DECIMAL_POINT
Plus or minus	PLUS OR MINUS	PLUS_OR_MINUS
Minus or plus	MINUS OR PLUS	MINUS_OR_PLUS
Less than	LESS THAN	<
Greater than	GREATER THAN	>
Less than or equal	LESS THAN OR EQUAL	LESS_THAN_OR_EQUAL
Not equals	NOT EQUAL	NOT_EQUAL
Greater than or equal	GREATER THAN OR EQUAL	GREATER_THAN_OR_EQUAL
Approx. equal to	APPROX EQUAL TO	APPROX_EQUAL_TO
Nearly equal to	NEARLY EQUAL TO	NEARLY_EQUAL_TO
Much less than	MUCH LESS THAN	MUCH_LESS_THAN
Asymptotically equal to	ASYMPTOTICALLY EQUAL TO	ASYMPTOTICALLY_EQUAL_TO
Much greater than	MUCH GREATER THAN	MUCH_GREATER_THAN
Equivalent	EQUIVALENT	EQUIVALENT
Congruent	CONGRUENT	CONGRUENT
Proportional to	PROPORTIONAL TO	PROPORTIONAL_TO
Ohms symbol	OMEGA UPPER	OMEGA_UPPER
Degrees	DEGREES	DEGREES
Minutes	MINUTES	MINUTES
Seconds	SECONDS	SECONDS
Percent	PERCENT	%
Per thousand	PER THOUSAND	PER_THOUSAND
Parallel	PARALLEL	PARALLEL
Perpendicular	PERPENDICULAR	PERPENDICULAR
Infinity	INFINITY	INFINITY
Root	ROOT	ROOT
Angle	ANGLE	ANGLE
	MARKED ANGLE	MARKED_ANGLE
Exists	EXISTS	EXISTS
Such that	SUCH THAT	SUCH_THAT
Universal	UNIVERSAL	UNIVERSAL
Logical Or	LOGICAL OR	LOGICAL_OR
Logical And	LOGICAL AND	LOGICAL_AND
Logical Not	LOGICAL NOT	LOGICAL_NOT
Empty set	EMPTY SET	EMPTY_SET
Intersection	INTERSECTION	INTERSECTION
Union	UNION	UNION
Proper subset	PROPER SUBSET	PROPER_SUBSET
Proper superset	PROPER SUPERSET	PROPER_SUPERSET
Reflex subset	REFLEX SUBSET	REFLEX_SUBSET
Reflex superset	REFLEX SUPERSET	REFLEX_SUPERSET

Description in User Guide	CHARKIT name	MKWHEEL name
Element	ELEMENT	ELEMENT
Floor	OPEN FLOOR	OPEN_FLOOR
	CLOSE FLOOR	CLOSE_FLOOR
Ceiling	OPEN CEILING	OPEN_CEILING
	CLOSE CEILING	CLOSE_CEILING
Circle plus (special operator)	CIRCLE PLUS	CIRCLE_PLUS
Circle times (special operator)	CIRCLE TIMES	CIRCLE_TIMES
Sum	SUM	SUM
Product	PRODUCT	PRODUCT
Integral	INTEGRAL	INTEGRAL
Contour integral	CONTOUR INTEGRAL	CONTOUR_INTEGRAL
Integral (large)	INTEGRAL (LARGE)	INTEGRAL_LARGE
Contour integral (large)	CONTOUR INTEGRAL (LARGE)	CONTOUR_INTEGRAL_LARGE
Partial differential	PARTIAL DIFFERENTIAL	PARTIAL_DIFFERENTIAL
Gradient	GRADIENT	GRADIENT
Slash (large)	SLASH (LARGE)	SLASH_LARGE
Bracket (large)	OPEN BRACKET (LARGE)	OPEN_BRACKET_LARGE
	CLOSE BRACKET (LARGE)	CLOSE_BRACKET_LARGE
Square bracket (large)	OPEN SQUARE BRACKET (LARGE)	OPEN_SQUARE_BRACKET_LARGE
	CLOSE SQUARE BRACKET (LARGE)	CLOSE_SQUARE_BRACKET_LARGE
Curly brackets (large)	OPEN CURLY BRACKET (LARGE)	OPEN_CURLY_BRACKET_LARGE
	CLOSE CURLY BRACKETS (LARGE)	CLOSE_CURLY_BRACKET_LARGE

Arrows

	UP ARROW	UP_ARROW
	UP AND DOWN ARROW	UP_AND_DOWN_ARROW
	DOWN ARROW	DOWN_ARROW
	LEFT ARROW	LEFT_ARROW
	LEFT AND RIGHT ARROW	LEFT_AND_RIGHT_ARROW
	RIGHT ARROW	RIGHT_ARROW
	WIDE UP ARROW	WIDE_UP_ARROW
	WIDE UP AND DOWN ARROW	WIDE_UP_AND_DOWN_ARROW
	WIDE DOWN ARROW	WIDE_DOWN_ARROW
	WIDE LEFT ARROW	WIDE_LEFT_ARROW
	WIDE LEFT AND RIGHT ARROW	WIDE_LEFT_AND_RIGHT_ARROW
	WIDE RIGHT ARROW	WIDE_RIGHT_ARROW
	UP TRIANGLE	UP_TRIANGLE
	DOWN TRIANGLE	DOWN_TRIANGLE
	LEFT TRIANGLE	LEFT_TRIANGLE
	RIGHT TRIANGLE	RIGHT_TRIANGLE

Special Textual Symbols

Circle dot (arrow out)	CIRCLE DOT	CIRCLE_DOT
Large open circle	LARGE OPEN CIRCLE	LARGE_OPEN_CIRCLE
Large bullet	LARGE BULLET	LARGE_BULLET
Box	BOX	BOX
Filled box	FILLED BOX	FILLED_BOX
Diamond	DIAMOND	DIAMOND
Tick	TICK	TICK
Cross	CROSS	CROSS
Star	STAR	STAR

9010

Description in User Guide	CHARKIT name	MKWHEEL name
Circled digits	USER SYMBOL 0	USER_SYMBOL_0
	USER SYMBOL 1	USER_SYMBOL_1
	USER SYMBOL 2	USER_SYMBOL_2
	USER SYMBOL 3	USER_SYMBOL_3
	USER SYMBOL 4	USER_SYMBOL_4
	USER SYMBOL 5	USER_SYMBOL_5
	USER SYMBOL 6	USER_SYMBOL_6
	USER SYMBOL 7	USER_SYMBOL_7
	USER SYMBOL 8	USER_SYMBOL_8
	USER SYMBOL 9	USER_SYMBOL_9
Smiling face	SMILING FACE	SMILING_FACE
Black smiling face	BLACK SMILING FACE	BLACK_SMILING_FACE
Mars	MARS	MARS
Venus	VENUS	VENUS
Clubs	CLUBS	CLUBS
Diamonds	DIAMONDS	DIAMONDS
Hearts	HEARTS	HEARTS
Spades	SPADES	SPADES
Note	NOTE	NOTE
Two-note	TWO-NOTE	TWO-NOTE

II: Accents

Acute	ACUTE	ACUTE
Grave	GRAVE	GRAVE
Circumflex	CIRCUMFLEX	CIRCUMFLEX
Umlaut or Diaresis	UMLAUT	UMLAUT or DIARESIS
Caron or Hacek	CARON	CARON
Dot	DOT	DOT
Tilde	TILDE	TILDE
Breve	BREVE	BREVE
Double acute	DOUBLE ACUTE	DOUBLE_ACUTE
Ring	RING	RING
Macron	MACRON	MACRON
Double grave	DOUBLE GRAVE	DOUBLE_GRAVE
Cedilla	CEDILLA	CEDILLA
Ogonek	OGONEK	OGONEK
Latvian tail	LATVIAN TAIL	LATVIAN_TAIL
Dot below	DOT BELOW	DOT_BELOW
Stroke	STROKE	STROKE
Rough breathing	ROUGH BREATHING	ROUGH_BREATHING
Rough breathing with acute	ROUGH BREATHING WITH ACUTE	ROUGH_BREATHING_WITH_ACUTE
Rough breathing with grave	ROUGH BREATHING WITH GRAVE	ROUGH_BREATHING_WITH_GRAVE
Rough breathing with circumflex	ROUGH BREATHING WITH CIRCUMFLEX	ROUGH_BREATHING_WITH_CIRCUMFLEX
Rough breathing with tilde	ROUGH BREATHING WITH TILDE	ROUGH_BREATHING_WITH_TILDE
Smooth breathing	SMOOTH BREATHING	SMOOTH_BREATHING
Smooth breathing with acute	SMOOTH BREATHING WITH ACUTE	SMOOTH_BREATHING_WITH_ACUTE
Smooth breathing with grave	SMOOTH BREATHING WITH GRAVE	SMOOTH_BREATHING_WITH_GRAVE
Smooth breathing with circumflex	SMOOTH BREATHING WITH CIRCUMFLEX	SMOOTH_BREATHING_WITH_CIRCUMFLEX
Smooth breathing with tilde	SMOOTH BREATHING WITH TILDE	SMOOTH_BREATHING_WITH_TILDE
Greek circumflex	GREEK CIRCUMFLEX	GREEK_CIRCUMFLEX

CHARKIT name	MKWHEEL name

II: Accented characters in the LocoScript Character Set

ACUTE + A LOWER	ACUTE + a
ACUTE + E LOWER	ACUTE + e
ACUTE + I LOWER	ACUTE + i
ACUTE + O LOWER	ACUTE + o
ACUTE + U LOWER	ACUTE + u
GRAVE + A LOWER	GRAVE + a
GRAVE + E LOWER	GRAVE + e
GRAVE + I LOWER	GRAVE + i
GRAVE + O LOWER	GRAVE + o
GRAVE + U LOWER	GRAVE + u
CIRCUMFLEX + A LOWER	CIRCUMFLEX + a
CIRCUMFLEX + E LOWER	CIRCUMFLEX + e
CIRCUMFLEX + I LOWER	CIRCUMFLEX + i
CIRCUMFLEX + O LOWER	CIRCUMFLEX + o
CIRCUMFLEX + U LOWER	CIRCUMFLEX + u
UMLAUT + A LOWER	UMLAUT + a
UMLAUT + E LOWER	UMLAUT + e
UMLAUT + I LOWER	UMLAUT + i
UMLAUT + O LOWER	UMLAUT + o
UMLAUT + U LOWER	UMLAUT + u
UMLAUT + Y LOWER	UMLAUT + y
UMLAUT + A UPPER	UMLAUT + A
UMLAUT + O UPPER	UMLAUT + O
UMLAUT + U UPPER	UMLAUT + U
TILDE + A LOWER	TILDE + a
TILDE + N LOWER	TILDE + n
TILDE + O LOWER	TILDE + o
TILDE + N UPPER	TILDE + N
RING + A LOWER	RING + a
RING + A UPPER	RING + A
CEDILLA + C LOWER	CEDILLA + c
STROKE + O LOWER	STROKE + o
STROKE + O UPPER	STROKE + O
STROKE + L LOWER	STROKE + l
STROKE + L UPPER	STROKE + L
LATVIAN TAIL + G LOWER	LATVIAN_TAIL + g

Accented Greeks

ACUTE + ALPHA LOWER	ACUTE + ALPHA_LOWER
ACUTE + EPSILON LOWER	ACUTE + EPSILON_LOWER
ACUTE + ETA LOWER	ACUTE + ETA_LOWER
ACUTE + IOTA LOWER	ACUTE + IOTA_LOWER
ACUTE + UPSILON LOWER	ACUTE + UPSILON_LOWER
ACUTE + OMEGA LOWER	ACUTE + OMEGA_LOWER
DIARESIS + IOTA LOWER	DIARESIS + IOTA_LOWER
DIARESIS + UPSILON LOWER	DIARESIS + UPSILON_LOWER

Appendix II: Standard Substitutions

The characters that you can print at any time are essentially limited to the ones defined in the Character Set you are using, and where a document contains a LocoScript character that isn't defined in the Character Set, LocoScript will leave a blank for you to fill in the character later – by hand, if necessary.

However, CHARKIT and MKWHEEL are designed to enable you print as many of the characters in the LocoScript 2 character set as possible – not just the characters that you have specifically defined. In particular, they will always enable accented characters to be produced by printing both the accent and the unaccented character. Other characters are made available, wherever possible, by substituting an identical or similar character that has been defined in the Character Set. These substitutions will be apparent when you print the ALLCHARS document.

The following table lists the standard substitutions that are made, giving first the details of the character that may be 'missing' and then, in italic text, the substitution that may be made. Of course, if the character has been specifically defined, the substitution given here won't be used.

Alphanumerics

a...z lower	*A...Z upper*
Slashed zero	*Stroke + Unslashed zero*
Zero (no slash)	*Slashed zero*

Accents

Stroke accent	*Slash*
Accented character	*Accent + character*
High cedilla	*Low cedilla*

Greek characters

Alpha upper	*A upper*
Beta lower	*Scharfes S*
Beta upper	*B upper*
Gamma upper	*G Cyrillic upper*
Epsilon lower	*Element*
Epsilon upper	*E upper*
Zeta upper	*Z upper*
Eta upper	*H upper*
Iota upper	*I upper*
Kappa upper	*K upper*
Mu upper	*M upper*
Nu upper	*N upper*
Pi upper	*P Cyrillic upper*
Rho upper	*P upper*
Tau upper	*T upper*
Upsilon upper	*Y upper*
Phi (scientific)	*Phi lower*
Phi upper	*F Cyrillic upper*
Chi upper	*X upper*

Cyrillic characters

Vyedi or V upper	*B upper*
Glagol or G upper	*Gamma upper*
Ukrainian Yest or Ye lower	*Element*
Kako or K upper	*K upper*
Muislete or M upper	*M upper*
Nash or N upper	*H upper*
Pokoi or P upper	*Pi upper*
Tverdo or T upper	*T upper*
Fert or F upper	*Phi upper*
E Russian lower	*Such that*

(contd.)

9010
9010

Other special language characters

Dotless i lower	*I lower*
Scharfes S	*Beta lower*

Textual symbols

Hyphen	*Dash*
Dash	*Hyphen*
Open single quote	*Apostrophe*
Close single quote	*Apostrophe*
Open double quote	*Double quote*
Close double quote	*Double quote*

Continental textual symbols

Open single guillemet	*Less than*
Close single guillemet	*Greater than*
Open double guillemet	*Less than*
Close double guillemet	*Greater than*

Currency symbols

Cent	*Stroke + C lower*
Franc	*F upper*
Florin	*F lower*
Yen	*Y upper*

Mathematical and technical symbols

Modulus	*Vertical bar*
Decimal point	*Full stop*
Not equals	*Stroke + Equals*
Minutes	*Apostrophe*
Seconds	*Double quote*
Such that	*E Russian lower*
Element	*Ye Ukrainian lower*
failing that:	*Epsilon lower*

Appendix III: Formal definition of CHARKIT Character Definition file

The Character Definition file used by CHARKIT comprises a File header which contains general information about the character set, followed by a File body which describes the means by which individual characters are obtained.

General syntax

The file will be processed one line at a time. Lines must not exceed 255 characters and must end with a carriage return or CP/M end-of-file character (the latter terminates file processing). Throughout the file tabs and linefeeds will be treated as single spaces.

Blank lines can be inserted anywhere in the file and comments of the form ; *comment* can be appended to the end of any line. These are ignored when the Printer File is compiled.

In the definitions below, items enclosed in square brackets are optional. If an optional part may be repeated (and so may appear any number of times, including none at all) an asterisk is appended after the closing square bracket. For example, a string of characters requiring at least one character is written as *char[char]**.

Note: So that there is no possibility of mis-interpretation, Character Definitions cannot include any language-dependent characters. Thus only a restricted range of ASCII characters may be used in the Character Definition file –

A...Z, a...z, 0...9, ! " % & ' ( ) * + , - . / : ; < = > ? _

char	is:	any of the characters listed above
value	is:	a number in the range 0...255, expressed either as a decimal number or as a hexadecimal number (see page 11)
byte	is:	any of the characters listed above except for asterisk, exclamation mark, semicolon, double quote and single quote, which have special meaning within this syntax. The first four may be entered as !*, !!, !; and !" respectively; single quote has to be entered as !'*value*' (see below).
	or:	!*letter*
		(denotes CONTROL-*letter*. eg. !A is CONTROL-A, ASCII code 1. *letter* must be in the range A...Z)
	or:	!'*value*'
	or:	!'*control-code-name*'
		(where *control-code-name* is one of the special names for control codes – eg. BEL, ESC, etc. A list of these names is given on page 25.)

sequence is: *"byte[byte]*" [side-effect]**

(sends several bytes to the printer)

side-effect records any residual effects that an escape sequence may have on the current emphasis, pitch or line spacing. This may be any of:

	`+B`	(if bold is set)
or:	`-B`	(if bold is cleared)
or:	`+O`	(if overstrike is set)
or:	`-O`	(if overstrike is cleared)
or:	`+D`	(if draft quality is set)
or:	`-D`	(if draft quality is cleared)
or:	`+U`	(if underlining is set)
or:	`-U`	(if underlining is cleared)
or:	`+I`	(if italic is set)
or:	`-I`	(if italic is cleared)
or:	`+R`	(if raised (superscript) text is set)
or:	`-R`	(if raised (superscript) text is cleared)
or:	`+L`	(if lowered (subscript) text is set)
or:	`-L`	(if lowered (subscript) text is cleared)
or:	`+PS`	(if proportional spacing is set)
or:	`+10`	(if 10 pitch is set)
or:	`+12`	(if 12 pitch is set)
or:	`+15`	(if 15 pitch is set)
or:	`+17`	(if 17 pitch is set)
or:	`+PSW`	(if wide proportional spacing is set)
or:	`+10W`	(if wide 10 pitch is set)
or:	`+12W`	(if wide 12 pitch is set)
or:	`+15W`	(if wide 15 pitch is set)
or:	`+17W`	(if wide 17 pitch is set)
or:	`-P`	(if the horizontal pitch is left in an unknown state)
or:	`+V` *vertical-pitch*	(if a new vertical pitch is set)
or:	`-V`	(if the vertical pitch is left in an unknown state)

where *vertical-pitch* is the new vertical pitch expressed in printer units, and may take any value up to 255. The printer units are defined in the Printer Driver file: if not known or out of range, use `-V` instead. (With daisy-wheel printers, the printer units are normally those used for VMI settings and the *vertical-pitch* is the VMI that has been set.)

9010

File Header

The file header comprises a number of lines of the form given below. These must appear in the order shown, except that lines or items shown in square brackets may be omitted (whereupon the defaults detailed below will apply). Any number of comment lines may be inserted, as described above.

```
! "title-line1"
! "title-line2"
! "issue-details"
"set-name" [default-ps-width]
"style-name" [pitch]
selection-sequence
[PS-units [PS-origin]]
["underline-code" [underline-width]]
!
```

title-line1
title-line2

are lines of up to 30 chars. They form descriptive text which can be read from within LocoScript using the 'Inspect document' option.

issue-details

is a line of exactly 30 chars, which forms the final line of descriptive text which can be read from within LocoScript using the 'Inspect document' option. Its form is:

author-name day month year

with single spaces before *day*, *month* and *year*.

author-name	should be exactly 20 *chars* and should identify you or your company as the author. (Note that we have reserved the name `Locomotive Software` for our own use.)
day	is the day of the month. Values over 31 are rejected. Its form may be:
	single-space dec-digit
or:	*dec-digit dec-digit*
month	is one of the following month name abbreviations:
	`Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec`
year	is the last two digits of the year. Its form must be:
	dec-digit dec-digit

set-name

is up to 12 chars and gives the name by which this character set will be identified in the relevant LocoScript menus (eg. `UK ASCII`).

default-ps-width

is a value giving the PS-width, in 1/240", which LocoScript will assume if no PS-width has been specified. In particular, it gives the width of the gap to be left in PS text in place of any character which does not appear in the printer's character set. If omitted, a value of 20 is assumed (12 pitch space).

style-name

is up to 12 chars and gives the name of a Character Style to be associated with the Character Set (eg. `Times`). This Character Style name is automatically added to the Settings menu at the same time as the Character Set.

pitch

is the character pitch of the Character Style whose name is given by *style-name*. If not specified, 10 pitch is assumed. It may be:

`10, 12, 15, 17` or `PS`

selection-sequence

is a *sequence* which may be sent to the printer following a printer reset in order to ensure that the required character set is selected. The minimum sequence that can be specified is `""` (an empty sequence).

PS-units

expresses the units in which the PS widths of characters will be expressed within the file body. If these are not specified, it will be assumed that proportional spacing is unavailable: in which case there need be no PS width information in the file body.

The *value* specified must be an integer not greater than 3840 (corresponding to PS-widths in any unit down to 1/3840"). Wherever possible, the value should specify the units the printer itself works in.

PS-origin

specifies an offset which was used in earlier versions of CHARKIT to bring the range of PS-widths quoted in the File Body within a required range. This offset was expressed as a number of PS-units (between 0 and 63).

It is retained purely for compatibility. In all new files, it should be omitted.

underline-code

is the *byte* which must be sent to the printer in order to print an underline character. It is not required for printers having an auto-underline facility: otherwise its omission will suppress the manual underline facility.

underline-width

is a *value* giving the width in 1/240" units of the printer's underline character, for use when a PS Character Style is specified. It is not required for printers having an auto-underline facility or when manual underlining is suppressed. If unspecified, the value 20 is assumed.

File Body

The body of the character set definition file comprises a number of lines of the following forms:

1 *code name [PS-width-info]*

where

code is: *byte*

where a single byte in the range 0...255 is sent to the printer

or: *sequence*

where several bytes are sent to the printer

or: `*P`*number*

where number is in the range 1...255 and identifies one of a set of printer-specific code sequences, used principally to access the characters on the two to six 'extra' petals on a printwheel. The actual sequences of codes are held in the Printer Driver file. Note that no check is made to ensure that the requested printer sequence actually exists.

name is: "*character-name*"

where *character-name* identifies a character in the LocoScript 2 character set. These names are listed in Appendix I.

or: "*accent-name*"

where *accent-name* identifies one of the standard LocoScript accents defined in the list in Appendix I. Since these are 'dead-key' characters (on most daisy-wheel printers the accent is printed without moving and on other printers it is printed as a fixed pitch character followed by a backspace) they do not require width information.

or: "*accented-character-name*"

where *accented-character-name* identifies an accented character which exists in its own right in the LocoScript 2 character set. These names are listed in Appendix I.

or: "*accent*" + "*character-name*"

This form of the line is used when an accent–character combination which does not form part of the standard LocoScript character set is available as a special character in the printer's own set. The specified code will then be used instead of the normal mechanism, which prints the accent followed by the character. No PS-width information should be quoted as the width of the character is always assumed to be the same as the unaccented character.

Any mixture of upper or lower case is accepted, and any combination of spaces, tabs and linefeeds within a name counts as a single space: but there must be no leading or trailing spaces.

PS-width-info gives the character width in proportional spacing mode. If *PS-units* were not defined in the file header then this may be omitted: otherwise it must be included. Some printers have different PS widths for draft or italic mode. For this reason *PS-width-info* has the form:

PS-width[,[draft-width][,[ital-width][,draft-ital-width]]]

PS-width	is a *value* expressing the PS width of a printed character as an offset of between 1 and 15 *PS-units* from the *PS-origin* (see header definitions).
draft-width	is the PS-width of a draft, non-italic character. If omitted it defaults to the first value in *PS-width-info*. Its form is: *PS-width*
ital-width	is the PS width of an italic letter quality character. If omitted it defaults to the first value in *PS-width-info*. Its form is: *PS-width*
draft-ital-width	is the PS width of an italic draft quality character. If omitted it defaults to *ital-width*. Its form is: *PS-width*

Notes: (i) Substitute *'Bold'* for *'Draft'* when describing a laser printer font or a font used on a HP DeskJet or compatible printer.

(ii) The range of widths must not correspond to a difference of more than 14/60" between the largest and the smallest, and the largest character can be no more than 78/60" wide.

2 **Uuser-no code name [PS-width-info]*

where

user-no is: a *value* in the range 1 .. 31 inclusive, identifying one of the User states. The printer is first switched into the given User state by sending one or more bytes, the character is then printed by sending the given *code* and the User state is then cancelled before printing continues. (The form of a line defining a User state is given below.)

3 ***user-no on-command off-command*

where

on-command is: *sequence*

(defines the sequence of bytes selects special printer states prior to the printing of a character)

off-command is: *sequence*

(defines the sequence of bytes cancels any special printer states set by the corresponding *on-command*)

A line of this form defines one of the 31 possible User states which may be set before sending a *code* to the printer in order to print the desired character. Any residual effects on emphasis, pitch or line spacing of escape sequences used within the *on-* and *off-commands* must be recorded as *side-effects*.

Appendix IV: Troubleshooting

CHARKIT Errors

The following lists the error messages which can appear while you are running the CHARKIT program, together with a brief explanation of the probable cause.

General errors

`Byte value required`
A number in the range 0...255(&00...&FF) was expected.

`Cannot find input file`
The second of the two named files in your CHARKIT command could not be found. One likely explanation is that you have got the names of your Character Set (.#) file and your Character Definition file in the wrong order: the .# file must come first. The other possibility is that you have not given the full file specification (*drive:filename.filetype*) for the Character Definition file or you have not stored it in group 0.

`Directory full cannot open output file`
You have more than 256 files on this disc. Erase any files you don't want or move some to another disc before trying to run CHARKIT again.

`Disc error. Failed to close output file.`
`Disc error. Failed to name output file correctly.`
System errors. If repeated, consult your dealer.

`First character of input file invalid (not ASCII??)`
This error usually means that you haven't made your character definition file into a simple text (ASCII) file.

`Input line is too long`
The maximum length of any line in the character definition file is 255 characters.

`Insufficient disc space for output file`
The lack of space could well be because you have some Limbo files on the disc. Use LocoScript to erase any Limbo files (and any other files) you don't want before running CHARKIT again.

`Invalid character(s) in string`
This usually means that you have represented a special character incorrectly; for example, you may have put * instead of !*.

`Invalid character`
This usually means that you have given two (or more) characters where one was expected.

`Invalid file specification`
One or both of the file names weren't specified correctly in the command line.

`Invalid states/side effect definition`
The side-effects following an escape sequence have been specified incorrectly.

`Output file exists - Overwrite (Y/N)?`
You already have a .# file of this name on your disc. If you type `Y`, the old file will be overwritten; if you type `N`, you will be asked to type in another name for the Printer file and the original Printer file won't be overwritten.

`Premature end of file`
Incomplete Character Definition file. For example, it may consist purely of a header or the line that terminates the header may be missing.

`Syntax error`
Indicates an error in the Character Definition file not covered by one of the more precise errors quoted below. One possibility is that the file CHARKIT is trying to process is not a Character Definition file at all.

`Table space for sequences full`
The total number of bytes used in sequences defining special characters exceeds CHARKIT's limit of 2048.

`Value required`
A number in the range 0...65535 (&00...&FFFF) was expected.

Errors in the Header section of the Character Definition file

`Header terminator missing or incorrect`
The line marking the end of the Header section has been omitted. This line should contain a single `!`.

`Invalid date format`
The date does not have the standard (space)*dd*(space)*mmm*(space)*yy* format; *dd* is not in the range 01...31, *mmm* is not one of the standard month abbreviations, or *yy* is not in the range 00...99; or the date sequence in the wrong position. The space before the *dd* must be preceded by exactly 20 characters of *author-name*.

`Invalid fundamental pitch`
PS-units must be an integer and it must be no more than 3840.

`Quoted text too long`
This message is used to indicate both that a line of the text you have set up as Identity text is more than 30 characters long and that a name that you have used either as the Character Set name or the Character Style name is more than 12 characters long.

`This name is not allowed`
You may not specify `Locomotive Software` as the author of the Character Definition file.

`Value too large`
One of the items on this line is outside its permitted range of values.

Character Definition problems

`Character already defined (possibly using another name)`
This is the second definition you have given containing this LocoScript character. You can only have one definition for each LocoScript character, though you can have as many as you like for each character in the printer's character set. (The alternative name refers to the 16 characters that are both named characters and user-definable characters and so have two names.)

`Character too wide`
Character widths can be no more than 78/60".

`Duplicate user definition`
This is the second definition you have given for this particular user sequence.

`Invalid printer sequence number`
Printer sequence numbers must be in the range 1...255. **Note:** CHARKIT does not have access to the corresponding .PRI file and so cannot check that the printer sequences you specify are defined in this file.

`Invalid use of "*"`
`*` has special meaning in the Character Definition file (see the section on 'Enhancements') and so has to be written as `!*` when it is defined.

`Invalid user number`
User sequence numbers must be in the range 1...31.

`New vertical pitch is too big, Use -V instead`
The vertical pitch you have specified as a side-effect of an escape sequence is greater than 255 printer units. Giving this side-effect as `-V` records it as an unspecified vertical pitch.

`Ps width information missing`
If you have specified your chosen PS-units in the Header section of the Character Definition file, you have to specify a PS width for every character you define except accents.

`Range of PS widths too great`
The range of widths must not correspond to a difference of more than 14/60" between the smallest and the largest.

`Space must be single byte command`
'Escape sequence' definitions for the LocoScript Space character are not supported. (**Note:** The usual code for the Space character (&20) is represented in the Character Definition as !'SP'.)

`Unknown character name`
Both of these indicate that the LocoScript character name you have given is incorrect. Check the name you have used against the name in Appendix I.

`WARNING - PS widths may not be accurate`
The actual widths being used by CHARKIT don't precisely match the widths you specified. The Printer file will still be made.

9010

MKWHEEL and SETWHEEL errors

The following lists the error messages which can appear while you are running the MKWHEEL and SETWHEEL programs, together with a brief explanation of the probable cause.

Note: All the errors in the 'General errors' section cause the program to fail; an error from one of the other sections is not normally fatal and the program will still produce a Printwheel table unless a large number of such errors are found.

General errors

`Cannot open input file`
CP/M cannot find the Printwheel Description file you specified. Check the file name you specified in your command, and check that the file has User number 3 (ie. it is in group 3 on the disc). The Printwheel Description file must have the same User number as the SETWHEEL/MKWHEEL program you use it with.

`Cannot set wheel, because there is no CP/M section in the definition file`
The Printwheel Description file you are using doesn't include any CP/M definitions.

`Error writing output file : Close failed`
System error. If repeated, consult your dealer.

`Error writing output file : Directory full`
You cannot have more than 256 files on the Printwheels disc. Erase any files you don't want (in particular, LocoScript Limbo files in User 8...15) or move some to another disc before running MKWHEEL again.

`Error writing output file : Disc full`
Erase any files you don't want (in particular, LocoScript Limbo files in User 8...15) or move some to another disc before running MKWHEEL again.

`Input file is not ASCII, it is probably a LocoScript file`
The Printwheel Description file you use with MKWHEEL or SETWHEEL must be a simple ASCII text file. Try editing the file using LocoScript: if the file is a LocoScript document, the text will appear on the screen; if it is something else (eg. ASCII) you will see an Alert message telling you that this is not a LocoScript file. If necessary, re-create the ASCII version of your LocoScript file before running MKWHEEL or SETWHEEL again.

`Input file name missing`
You will get this error if you just type the command `MKWHEEL` RETURN or `SETWHEEL` RETURN. Retype your command, this time including the name of the Printwheel Description file you want the program to use.

`No output file name`
Your MKWHEEL command didn't include any Printer File name (before the =).

Header errors

`Error: Cannot recognise expected character set name string`
`Error: Cannot recognise expected character style name string`
`Error: Cannot recognise expected identity string`
One of these messages tells you that you have omitted the double quote marks around the Character Set/Style name or Identity text you have given on the relevant line of the Header section.

`Error: Cannot recognise expected pitch`
`Error: Pitch must be 10, 12, 15 or PS`
The last line of the Header section specifies an invalid pitch for the Character Style.

`Error: String exceeds maximum length allowed`
The Character Set/Style name or Identity text on this line is too long. A Character Set/Style name must be no more than 12 characters long (including spaces); each line of Identity text must be no more than 30 characters long (including spaces).

`Error: String contains invalid characters`
Only the characters listed on page 29 may be used in Character Set/Style names and Identity text.

Errors in the character definitions

`Error: Cannot recognise character`
This usually means that you have used the wrong type of definition for this part of the Printwheel Description.

`Error: Cannot recognise expected petal number`
The definition does not start with a valid petal number.

`Error: Cannot recognise expected section keyword`
This line looks like the heading to a section (ie. it isn't a definition line) but doesn't contain one of the section keywords.

`Error: Cannot use both petal` *n* `and petal` *m* `for character ....`
If two petals have the same character on them, one should be defined as a Duplicate character by preceeding the character details by the keyword `DUP`.

`Error: Character value illegal`
The code you have given is outside the range used by LocoScript (or CP/M).

`Error: Extra text found where nothing expected`
The probable explanation is that you have added a comment to this line but you have forgotten the ; that should start it.

`Error: Hammer intensity must be in range 4..10 or 4/5..10/11`
You appear to have quoted a hammer intensity outside the permitted range.

`Error: Language number must be in the range 0..8`
The number following the keyword `Language` is outside the range 0...8.

`Error: No room left for this substitution`
You cannot specify more than 256 substitutions (in addition to the built-in ones).

`Error: Petal number must be in the range 1 .. 100`
You appear to have quoted a petal number outside the range 1...100.

`Error: PS width must be in the range 3..8`
You appear to have quoted a character width outside the permitted range of 3...8.

`Error: That CP/M character must be in the CPM section`
This message appears if you include one of the language-invariant CP/M characters in a Language section.

`Error: That CP/M character must be in a LANGUAGE section`
This message appears if you include one of the language-variant CP/M characters in the main CP/M section. It may mean that you have omitted the keyword LANGUAGE at the start of the lists of langauge variant characters.

`Error: That character varies nationally, you must use a number`
You have represented one of the CP/M codes in a Language list by a character when it should be represented by its value.

`Fatal error: Too many composite characters`
You cannot specify more than 31 special accented characters (ie. accented characters that aren't included in the LocoScript character set).

`Warning: No fixed pitch petal for LocoScript 2 .... CP/M .... [language` *n*`]`
This LocoScript character is defined in the CP/M section of the file but not in the Fixed Pitch section.

`Warning: No PS petal for fixed petal` *n* `character ....`
This character which was included in the Fixed pitch section but it is not listed in the PS section.

`Warning: No PS petal for LocoScript 2 .... CP/M .... [ language` *n*`]`
This LocoScript character is defined in the CP/M section of the file but not in the PS section.

Note: The Warning messages often report errors that are a consequence of an earlier error, which meant that default information was retained in the Printwheel table.

Oct 1990
ISBN 1 85195 046 X