

7/1983/440/106



# FERRANTI PEGASUS COMPUTER

**LIBRARY  
SPECIFICATIONS**

**VOLUME I**

This document is a facsimile of the original book, transcribed by Christopher P Burton of the Computer Conservation Society in 2003 by the following method:

- Each page scanned at 200 dpi using Textbridge yielding 1-bit/pixel .tif files.
- Each image was then cropped by eye to have almost no white margins.
- Pages in the original (foolscap paper) which had text longer than A4 were cut and pasted to squeeze on to A4 size.
- Files were then saved as .gif image files.
- Word for Windows was then used to assemble the document, inserting one .gif image per page, with one inch left margin, 0.2 inch top margin, 0 right margin, 0.1 bottom margin on A4 paper. The images were ranged top left against those margins. It was necessary to fractionally reduce the size of each image to be slightly less than 11.38 inches high, rather than allow automatic fitting by Word.
- The document was saved and then output to an Apple Laserwriter II NTX but output to file, not actually printed. Requests to fix margins were not over-ruled. This created a PostScript file of the document, about 250 MB long.
- The PostScript file was then input to Frank Siegert's PStill program which converts to PDF to yield this document.

At the front of Volume 1 are loose pages of modifications.

T/1983/114/1106

PROPERTY OF THE  
SCIENCE MUSEUM  
ON LOAN TO  
I.C.L.

TELEGRAMS  
DIGIT, WESDO  
LONDON

TELEPHONE  
LANGHAM 9211

# FERRANTI LTD

ELECTRICAL & GENERAL ENGINEERS

HEAD OFFICE & WORKS  
HOLLINWOOD, LANCS  
TELEGRAMS  
FERRANTI, HOLLINWOOD  
TELEPHONE  
FAILSWORTH 2000

21 PORTLAND PLACE  
LONDON W.1

YOUR REF.

OUR REF. 441/ARB/JWM/KS

25th June 1962

To every Subscriber,

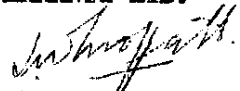
Dear Sir,

Pegasus Library Service

Recently we notified you of our decision to discontinue the annual renewal subscriptions for the Pegasus Library Service.

We feel that we have achieved our original aim which was to provide a reasonably comprehensive Library of general purpose routines and now it is apparent to us that additions are tending to be more and more specialised. At this point we consider it reasonable to round off the work in hand and to limit our future activities to the provision of a general advisory service. Naturally if any existing routines are found to contain errors we will notify Members of the service and if subsequently we are able to agree corrections with the authors we will undertake responsibility for distribution of such information.

Yours faithfully,  
FERRANTI LTD.

  
B.B. SWANN,  
SALES MANAGER,  
COMPUTER DEPT.

## ERRATA LIST NO. 3

This errata list is a complete, up-to-date list of corrections to the Pegasus Library Specifications and supersedes Errata List No.2.

An asterisk beside a correction indicates that the correction did not appear in Errata list No.2.

We should be pleased if you would check that your copies of the specifications have been amended.

<u>Specification</u>	<u>Page</u>	<u>Amendment</u>
R 1 } 3 } 5 }		Note that these routines always punch one $\phi$ (figure shift) after CR LF in order to give the printer time to complete the CR operation
1	10 Table	1.2 0.2+ 0.6 should read: 1.2 0.2+ 0.2
2057	1	Add to Section 1.3:- "Set HO = 1 to suppress optional punching".
101	1 Store: 3 Sec.6(e)	"10 blocks" should read: "11 blocks". "1.0+" should read: "1.1+"
102	1 Bottom 3 line 3	"Index 2" should read: "Issue 2" "... or entry..." should read: "... on entry..."
105	2 Sec.6(e)	"0.5" should read: "0.5+"
116	1 Time:	5 <sub>0</sub> should read: 5 <sub>c</sub> (twice)
* 200	1 Error:	R 200 has been superseded by R 200 Mk. 2 for which the error is less than $2^{-38}$ except when $p + 2^{-38} q \geq 1 - 2^{-38} + 2^{-76}$ for which numbers $p' = 1 - 2^{-38}$ .
224	2	Second line of programme should read: 224 - 04 -
* 241	2	Add Note 3 as follows:- "(3) The error is such that the exact range of the result, when using cue 02, is: $-(\frac{1}{2} + 2^{-38}) \leq p' \leq \frac{1}{2} + 2^{-38}$ If $q = 0$ and $p \neq 0$ or $q = \pm 2^{-38}$ and $ p $ is near 1.0 then $p' = \pm(\frac{1}{2} + 2^{-38})$ taking its sign from
* 242	1 Link:	"Obeyed in 1.7 - Arccos, or Arcsin if $p = 0$ or $-1.0$ 1.1 - Arcsin, unless $p = 0$ or $-1.0$ "
250	1 Uses:	Add: "X2 also is used when entry is made by cue 01"
* 251	1 Uses:	Add: "X3 also is used if $ p  \geq  q $ "
261	1	$\int_0^{\infty} \frac{2}{\pi}$ should read: $\int_0^{\infty} \frac{2}{\sqrt{\pi}}$



- \* 270 1 The sentence headed Error should be deleted and the following information inserted:-  
 "Error: Some indication of the error to be expected, when using a 9-bit exponent, is given below:
- |            |                          |   |
|------------|--------------------------|---|
| $I_0, I_1$ | for $p \geq 0$           | Error less than $1.3$ in $10^7$   |
| $K_0, K_1$ | for $p \geq 0.1$         | Error less than $1$ in $10^7$   |
| $J_0, J_1$ | for $0 \leq p \leq 25$   | Error less than $10^{-8}$   |
|            | for $25 < p \leq 250$    | Approximately 5 decimal digits correct except in the region of the zeros. |
| $Y_0, Y_1$ | for $0.1 \leq p \leq 25$ | Error less than $10^{-8}$   |
|            | for $25 < p \leq 250$    | Approximately 5 decimal digits correct except in the region of the zeros. |
- For values of  $p$  less than  $0.1$ , the errors in  $K$  and  $Y$  are larger. The table below gives approximate percentage errors.
- | $p$    | Errors in $K_0, Y_0$ | Errors in $K_1, Y_1$ |
|--------|----------------------|----------------------|
| .01    | .00001 %             | .00001 %             |
| .001   | .00005 %             | .00001 %             |
| .0001  | .006 %               | .00005 %             |
| .00001 | .6 %                 | .0004 %              |
- \* 2 Sec.3 Line 4 "... and  $|p| \geq 2^{-21} (\doteq 4.8 \times 10^{-7})$ " should read:
- \* 3 Line 2 "... and  $|p| \geq 2^{-18} (\doteq 3.8 \times 10^{-6})$ "  
 "... and  $|p| \geq 2^{-21} (\doteq 4.8 \times 10^{-7})$ " should read:  
 "... and  $|p| \geq 2^{-18} (\doteq 3.8 \times 10^{-6})$ "
- \* Sec.5 Line 5 "4.0+ 6 64 if  $|p| < 2^{-21} (K,Y)$ " should read:  
 "4.0+ 6 64 if  $|p| < 2^{-18} (K,Y)$ "
- \* 300 1 Uses: X5 is not used.  
 \* Sec.2.2 Add: "The auxiliary may not use X5 if the cue to the auxiliary is set as a programme parameter in X5."
- 323 4 Sec.5 In lines 11 and 12:-  
 "U0.0" should read: "U0.1"; "U0.1" should read: "U0.2"
- 6 Sec.9 The note to the stop in 2.7+ should read: "Loop stop if  $f(x_1, y) \geq 1.0$  or  $< -1.0$  for some value of  $x_1$ ".
- \* 400 1 Cues: Add "05 a - order partial cue  
 06 b - order partial cue"  
 1 Add at the end of section 1.1:-  
 "The routine will use the whole of blocks  $b$  to  $b-1 + 4(\frac{n+1}{8})_u$  as working space. Note in particular that when  $n$  is a multiple of 8 the routine overwrites 4 blocks after the last value of  $q$ ; for example if  $n = 16$ , blocks  $b$  to  $b + 11$  will be used."
- \* 401 1 "Rung-Kutta" should be spelt "Runge-Kutta"  
 \* 1 Cues: Add: "04 a - order partial cue  
 05 b - order partial cue"  
 1 Link: "... in U 1.7" should read: "... in U 0.7".  
 1 Add at the end of section 1.1:-  
 "The routine will use the whole of blocks  $b$  to  $b-1 + 4(\frac{n+1}{8})_u$  as working space. Note in particular that when  $n$  is a multiple of 8 the routine overwrites 4 blocks after the last value of  $q$ ; for example if  $n = 16$ , blocks  $b$  to  $b + 11$  will be used."
- \* 5 Sec.7.3 "b= 9 (mod 16)" should read: "b = 9+ (mod 16)"

- \* 402 1 Cues: Add: "04 a - order partial cue  
05 b - order partial cue"
- \* 403 1 Cues: Add: "04 a - order partial cue  
05 b - order partial cue"
- 2500 1 Add "File as 500" in top right hand corner under

**R 2500**

- \* 10 d) "Punch +.0000000" should read:  
"Punch +=.0000000"
- \* 11 Sec. 4.6 "... the 16th instruction is being obeyed!" should read:  
"... the instruction number 16 is being obeyed."
- 20 Sec. 6.4 "... the rth instruction" should read:  
"... instruction number r (the r+1th instruction)."
- 22 Sec. 6.7.2(3) "0.3+" should read: "1.3+"
- 28 Sec. 11.3 "T40.1" should read: "T42.1"  
"T40.4" should read: "T42.4"
- \* 29 Sec. 11.4 After line 4 add:  
X 48.4+

**0**

- line 9 (B 49.5+) 0 should be changed to read:  
35 2 00
- Sec. 11.5 "X 43.4+ - 53.5" should read: "X 43.4+ - 43.5"
- \* 511 3 Sec. 3.3 In the contents of X5, "m+n" should read:  
"separation, i.e., distance between leading coefficients of successive equations".
- 600 16 Sec. 13 To be inserted in the list of loop stops: "0.1 7 60  
on the operation  $n1 = \pm n2/n3$  when  $n3 = 0$ "
- 611 1 Notes(1) "0.1" should read: "0.2"
- \* 650 6 Line 6 "During input the accumulators are stored in B 0"  
should be replaced by: "During input the routine uses B 0"
- \* 10 R 686 The formula for the time of R 686, Matrix Division,  
should read: "60pg (p+3) milliseconds [ $+p^2(29p+240)$   
milliseconds for cues 01 and 02]."  
13 In programme:  
100 - 000 should read: 100 0 000  
3 3
- \* 720 2 Add Footnote:-  
"Amendment: To sort numbers in the range  
 $-s \leq x < 1.0 - s$   
C 720  
X 4+.2+  
S 1 02 (or 32 1 00 for  $-1.0 \leq x < 0$ )  
where  $s = C(S)$  and S is a special register."
- \* 721 2 Add Footnote:-  
"Amendment: To sort keywords in the range  
 $-s \leq x < 1.0 - s$   
C 721  
X 5+.2  
S 2 02  
S is a special register (or S is U 1.0 and the  
required s is set in B 5+.0 of R 721)."

\* 722 2 Add Footnote:-  
"Amendment: To sort keywords in the range  
 $-s \leq x < 1.0 - s$   
C 722  
X 2+.4+  
S 2 02 (or 32 2 00 for  $-1.0 \leq x < 0$ )  
where  $s = C(S)$  and S is a special register."

\* 723 4 Notes Add:  
"4.3 If it is required to preserve the end of the blocks containing  $A+rN-1$  and  $B+rN-1$  this may be done by inserting the following correction after the A3.

C 723  
X 8+.7 - 8+.7+  
O 5 73 5  
1 5 72 5 "

740 1 Cues 01 and 02 should start 0+ 0 72  
4 Title of the parameter list should read:

R 0	0-35
740	-04-

2901 1 "File as 901" did not print clearly  
1 Add Footnote:-  
"Note: If this programme is used before input of a programme using Assembly, it should be remembered that some locations beyond 3750 are used as working space by Assembly."

2902 1 Add Footnote as for R 2901

2903 2 Sec. 4.2 "... B 7+.5..." should read: "...B 8+.1..."  
Sec. 5.2 "... 3+.0..." should read: "... 4+.0..."  
"... 3+.1..." should read: "... 4+.1..."

2905 2 "... T 1.0 - 508.0..." should read:  
"... T1.0 - 500.0..."

\* 7927 1 Sec. 1.5 "... optional stop in 0.3..." should read:  
"... optional stop in 3.5..."  
\* Sec. 1.6 "... optional stop in 0.3..." should read:  
"... optional stop in 3.5..."

FERRANTI LTD

INDEX TO THE LIBRARY FOR THE  
FERRANTI PEGASUS COMPUTER

1. INTRODUCTION

This index gives brief details of all subroutines in the Pegasus Library, so that users may see what routines are available. The appropriate specifications should be studied before a routine is used.

2. LAYOUT OF THE INDEX

The first two columns of the index give the routine number and a brief description of each subroutine listed.

The third column indicates, by means of the following code, the state of each subroutine.

- A = Fully tested
- B = Partly tested
- C = Wholly or partly written but not tested
- D = Project

The fourth column, headed 'Spec', shows whether the specification and programme sheets have been printed.

- S = Specification issued or in preparation.
- P = Programme sheets and specification issued or in preparation.

Draft specifications are usually available for reference at the London Computer Centre.

The fifth column, headed Tapes, shows which tapes are generally issued. If there is no T in this column the relevant tape is sent when specially requested.

The sixth column, headed Store Blocks, shows the number of Main Store blocks occupied by the subroutine programme. An asterisk in this column indicates that the subroutine has an interlude obeyed during input. See section 3 below, 'Note on Interludes'.

The three columns headed 'Uses' show which Computing Store Blocks (U), Accumulators (X) and Main Store Blocks (B) are used as working space by the routine. The Main Store blocks occupied by the subroutine itself and listed in the sixth column are not included. In general the parts of the store stated to be 'used' by a subroutine are those which it may alter. For instance, X1 will *not* be listed as used if a link set before entering the routine remains unchanged in X1 on exit.

The approximate time of operation of the subroutine is given in the tenth column of the index. In order to save space a simpler but less accurate formula may be given here than in the subroutine specification. See also section 4.

The eleventh and last column of the index gives a brief explanation of the function of the subroutine. In this column the following abbreviations are used:

- SP. indicates that the subroutine is self-preserving
- CP. indicates that the routine is a complete programme
- I.O. refers to the Initial Orders.

### 3. NOTE ON INTERLUDES

Certain subroutines have interludes which are obeyed immediately the subroutine has been read in. These interludes make use of storage space following the programme of the subroutine. When the work of the interlude is completed the Transfer Address is reset and the interlude can be overwritten by the next subroutine.

The length of the interlude, if any, of the last subroutine accepted must be included when the storage space used by Assembly is considered. This interlude can be overwritten after the reading of the Library Tape is completed.

Interludes may sometimes be stored on the drum beyond 127.7 if necessary, but only if this is stated in the relevant specification or in the list below.

Existing subroutines which use such interludes are:-

<u>Subroutine</u>	<u>Length of Interlude (in Blocks)</u>	<u>Notes</u>
R 1	4	Interlude may be stored after 127.7 Includes optional parameter list
R 3	Uses R 1 and the above interlude	
R 52	4.7	No part of interlude may be stored after 127.7
R 106	1	Interlude may be stored after 127.7
R 270	1	Interlude may be stored after 127.7
R 300	6	The first block of the interlude must be stored in an address $\leq$ B 125
R 400	3	Interlude may be stored after 127.7
R 402	2	Interlude may be stored after 127.7
R 600	3	Interlude always stored in B 126.0 to 128.7
R 630	4	Interlude may be stored after 127.7
R 650	4	Interlude may be stored after 127.7
R 700	3	Interlude may be stored after 127.7
R 740	9	Interlude may be stored after 127.7

### 4. SPECIFICATIONS

The following details are assumed in individual library specifications unless information to the contrary is given.

The overflow indicator (OVR) must be clear on entry and is left clear on exit.

The link should be set as an order-pair in X1.

The subroutine is not self-preserving; it must be brought in from the Main Store each time it is used.

Nothing should be assumed about the contents of any register stated to be 'used' by the subroutine.

The time of operation is approximate and is the time from the first order of the subroutine (inclusive) to the first order of the link (exclusive).

## 5. ROUTINE NUMBERS

The isolated routines, most of which are, in fact, parts of the Initial Orders, have been given numbers between 1025 and 1099. These numbers are allocated for ease of reference only and are not used by the Assembly Routine.

Standard programmes forming part of the Library but not of a type suitable for use with Assembly are given routine numbers greater than 2000. These standard programmes have been adapted for use with the 7168-word Store and have been given corresponding numbers above 7000 (i.e. the first digit of their numbers has been changed from 2 to 7). No further reference has been made in this index to the 7168-word Store routines, as most of the details are the same as those for the routines on the 4096-word Store.

Routine numbers 850-899 inclusive have been set aside to be at the disposal of individual Pegasus users. The intention is that any subroutine which a user wishes to incorporate permanently in his Library Tape but which, for any reason, is not to be included in the standard Library as distributed by Ferranti Ltd should be allocated a number in this range. These numbers differ from the range 1000-1023 in that the latter are meant to be allocated to subroutines which the user requires temporarily rather than permanently. No programmes or specifications with numbers in either of these ranges will be generally distributed.

## 6. ACKNOWLEDGEMENTS

FERRANTI LTD gratefully acknowledge contributions from the following organisations:-

Sir W.G. Armstrong Whitworth Aircraft Ltd.; Babcock and Wilcox Ltd.; British Iron and Steel Research Association; The De Havilland Aircraft Company; National Research Development Corporation; Queen Mary College (London); Mr. F.E. Radcliffe; Robson, Morrow and Company; Royal Aircraft Establishment; United Steel Companies Ltd.; University of Durham; University of Southampton; University College of South Wales; Westland Aircraft Ltd. (Saunders-Roe Division).

## Allocation of Routine Numbers

- 0 - 99 Output**
  - 0 - 49 Output of numbers
  - 50 - 99 Non-numerical output
- 100 - 199 Input**
  - 100 - 149 Input of numbers
  - 150 - 199 Non-numerical input
- 200 - 299 Functions**
  - 200 - 219 Roots and powers
  - 220 - 239 Exponentials, logarithms, etc.
  - 240 - 259 Circular and hyperbolic functions and inverses
  - 260 - 279 Other functions of one variable
  - 280 - 299 Other functions of more than one variable
- 300 - 399 Operations**
  - 300 - 319 Quadrature and differentiation
  - 320 - 339 Interpolation and curve fitting
  - 340 - 359 Inverse interpolation, zeros of polynomials
  - 360 - 379 Power series
- 400 - 499 Differential Equations**
  - 400 - 419 Ordinary differential equations, first order
  - 420 - 439 Ordinary differential equations, linear (not first order)
  - 440 - 459 Ordinary differential equations, other
  - 460 - 479 Partial differential equations
- 500 - 599 Linear Algebra**
  - 500 - 509 General purpose schemes
  - 510 - 529 Linear equations
  - 530 - 549 Eigenvalues and eigenvectors
  - 550 - 589 Other special-purpose matrix operations
  - 590 - 599 Linear programming
- 600 - 699 Programmed Arithmetic**
  - 600 - 609 General-purpose aids to coding
  - 610 - 629 Floating-point, single-length
  - 630 - 649 Complex numbers (including floating-point)
  - 650 - 679 Multiple precision (including floating-point and complex numbers)
  - 680 - 699 Subroutines used by programmed arithmetic routines
- 700 - 799 Data Processing**
  - 700 - 719 Conversion
  - 720 - 739 Sorting and merging
  - 740 - 759 Standard processes
  - 760 - 779 File organisation
  - 780 - 799 Subroutines used by data processing routines
- 800 - 849 Applications**
- 850 - 899 Individual Pegasus users' private routines**
- 900 - 999 Miscellaneous**
  - 900 - 919 Programme Checking
  - 920 - 939 Organisational
  - 940 - 949 Non-numerical
  - 950 - 969 Number Tapes
  - 970 - 999 Miscellaneous routines
- 1000 - 1023 These numbers are available for temporary use by programmers
- 1025 - 1049 Isolated subroutines (in Initial Orders etc.)
- 1050 - 1099 Isolated test routines
- 1100 - 1199 Other test routines

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
	<b>Output</b>									
1	Print number, general purpose	A P	T		11*	0	-	0		SP. Prints scaled number from X1 with digit-layout. Parameter-list specifies styles of printing.
3	Page layout, using R1	A P	T		4*	0	1,6,7	0		SP. Uses R1 to print a scaled number and supplies page-layout characters as required (parameter-list).
4	Number Print (short)	A P	T		4	0	-	0		Prints contents of X7 as integer or fraction.
5	Double-length number print	A P	T		7	0	-	0		Prints (pq).2 <sup>38</sup> or p.2 <sup>38</sup> or q.
	Fast double-length number print (7168 store)	A P	T		7	0	-	-		Fast version of above routine to print at full speed of fast punches (7168 store only).
9	Print double-length integer	A S	T		5	0	-	0		SP. Prints (pq) as an integer.
10	Print double-length fraction	A S	T		5	0,1	-	0	43c-120	SP. Prints (p+2 <sup>-38</sup> q) as a fraction to up to 23 places. c is the number of characters punched.
11	Print floating-point numbers	A P	T		11	0	7	0		SP. Prints single length binary floating-point numbers from X1 in fixed or decimal floating point.
26	Shift and Print	A S	T		10	0,1,2,3	-	0	30(c+1)	Takes a series of numbers from the Main Store, multiplies by a scale factor and prints to a given number of significant figures. c = number of characters punched.



R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
35	Double-length print for Flexowriter	A S	T	7	0	-	0			Similar to R 5 and R 42 respectively, but printing is in the code used by the five-channel Flexowriter at the London Computer Centre.
36	f.s.d. print for Flexowriter	A S	T	4	0	6	0			
40	f.s.d. print from pence	A S	T	3	0	-	0			Prints an amount in pence as f.s.d.
42	Signed f.s.d. print from pence	A P	T	4	0	6	0	30c+19		SP. Prints an amount in pence as f.s.d. Negative amounts preceded by minus sign. c is the number of characters punched.
43	Mixed Radix Output	A S	T	4	0	-	0			Prints the contents of X7 as a mixed radix number.
45	Character Print	A S	T	4	0,1,2,3	3,4,5,6,7	-	210		Prints 6 6-bit characters on paper tape.
51	Steering routine for I.O. order output (See also R 1028)	A S	T	3	0,1	-	0			Functions as warning character P with optional printing suppressed.
52	Text output and input	A P	T	1.1*	0	-	0	30 per character		Facilitates the output of descriptive matter (e.g. English words) during the course of a programme; includes an interlude for the input of the descriptive matter.
53	Binary Punch. Steering routine for R 1033	A S	T	1	0,1,2,5	-	0	2 secs. per block		Punches out consecutive words from the Main Store in a form suitable for Binary Input, R 1031. Replaces R 50.
2054	Binary Punch without Transfer Address.	A S	T	4	All	All	-	2 secs. per block		CP. Functions like R 1033 but does not put any binary T on the output tape.

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
55	Printed Graph	A	S	T	1	0	-	0	30 per character	Plots $y/c$ against $x$ , where $c$ is the interval.
2056	Binary Punch for Isolated Store	A			13.4	All	All	-	5½ mins.	CP. For use by Maintenance Engineers. Not a general issue.
2057	Binary Punch Programme	A	S		1	All	All	0	2 secs. per block punched	CP. Makes a binary punch of all non-zero locations in the Store.
58	Binary Punch without T.A.	A	S	T	1	0,1,2	-	0	2 secs/block	Similar to R 53 but does not punch T.A.
2059	Binary Punch Programme Mk.2	A	S	T	1	All	All	0	2 secs/block	CP. Improved version of R 2057.
100	Input (See also R 1025, 1031, 1032) Input double-length integers or fractions	A	P	T	10	0,1	-	0		SP. Stores double-length numbers in consecutive pairs of Main Store locations.
101	Floating-point Input	A	P	T	11	0,1	All except 1	0.0		Floating-point numbers (each packed in one word) stored in Main Store.
102	Input Tables	A	P	T	2	All	5,6,7	0		Takes in sets of numbers, indexing first in each set. Uses Initial Orders.
103	Input Tables (Floating-point)	A	S	T	3	0,1,2	All except 1	0.0		Similar to R 102. Uses R 101.
105	Input mixed numbers	A	P	T	5	0,1	1,6,7	-		Reads double-length mixed number to (pq).
106	Modify R 105.	A	S	T	-*	-	-	-		Adapts R 105 to accept Sp as terminating character (consists solely of an interlude).

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
107	Revised mixed number input	A S T	5	0, 1	1, 6, 7	-			Improved version of R 105 and R 106.	
112	Inner Loop Integer read	A S T	2	0, 1	4, 5, 6, 7	-	5c		SP. Reads an integer into X7. Not complete - requires steering routine.	
113	Read and Scale Integers	A S T	1	0, 1	1, 4, 5, 6, 7	-	5c + 25		SP. Uses R 112. Reads integer and divides by scale factor to form fraction.	
116	Short Number read	A P T	3	0, 1, 2	1, 5, 6, 7	-	5c + 8		SP. Reads single length mixed number $N = p'/q'$ . $c = \text{number of characters read}$	
120	Slow Mid-point read	A S T	1	All	-	0	5c + 46		Uses Initial Orders Input. Reads number $N = 2^{-19} C(5.0)$ .	
121	General Single-Length Number Read	A P T	5	0, 1	6, 7	0	5c + 54		SP. Reads single length mixed number $N = p'/10^q'$	
137	Read number from handswitches	A S T	5	0	6	0			SP. Reads to X6 signed fraction or integer tapped out on the handswitches.	
140	f.s.d. input to pence	A S T	4	0, 1	7	0	5c + 50		Read f.s.d. and convert to pence. c is the number of characters read.	
142	Signed f.s.d. read to pence	A P T	6	0, 1	1	0	5c + 65		SP. Read positive or negative f.s.d. and convert to pence. c is the number of characters read.	
143	Mixed Radix Input	A S T	3	0, 1, 2	1, 6, 7	-			SP. Reads a mixed radix number and stores it in X7 as an integral number of units of the lowest denomination.	

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
148	Input f.s.d. tables to pence	A S	T	3	0,1,2	1,4,5,6,7	0	50 secs.	SP. Similar to R 102. Uses R 142.	
2150	Set Date	A S	T	1	All	All	511.6 511.7	4½ secs.	CP. Reads the date from tape and stores it in B 511.7. Clears the serial number in B 511.6.	
2151	Quick replacement of Isolated Store	A		0	All	All	0 512 -639	50 secs.	CP. For use by Maintenance Engineers. Not a general issue.	
195	Read Transactor Card	A S		3	0,1,2,5	1,5,6,7	25 locat- tions	4½ secs.	Reads a card in the transactor and stores the information in the Main Store.	
200	Square root	A	P T	1	0	5,6,7	-	40	Square root of $p + 2^{-38}q \geq 0$ . Time given applies to numbers of about $\frac{1}{4}$ to $\frac{1}{2}$ . Time increases as $p + 2^{-38}q$ decreases.	
201	Cube root	A	S T	2	0,1	5,6,7	-	80	Cube root of $p + 2^{-38}q$ . Times vary. See note to R 200.	
202	Fast square root	A	P T	2	0,1	4,5,6,7	-	30	SP. Uses more space but is faster than R 200. Time given applies to $\frac{1}{4} \leq (pq) < \frac{1}{2}$ . Time increases as $p + 2^{-38}q$ decreases.	
211	Floating-point Cube root	A	S T	3	0,1,2	1,5,6,7	-	103 - 149	$p' = \sqrt[3]{p}$ . $p'$ and $p$ are floating-point numbers.	

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
215	Complex square root	A S	T	6	0,1	All except 1	-	100 - 140	$(c + id) = \sqrt{(a + ib)}$ ; $(a + ib)$ double length.	
220	Exponential	A P	T	2	0	6,7	-	29	$p' = \frac{1}{4} \exp p$ .	
221	Logarithm, wide range	A P	T	3	0,1	5,6,7	-	34 - 50	$p' = \frac{1}{32} \log_e (pq)$ , $(pq) \geq 2^{-46}$ . Superseded by R 224.	
222	Floating-point Exponential	A S	T	4	0,1,2	1,5,6,7	-	86	$p' = \exp p$ . $p'$ and $p$ are floating-point numbers. Uses R 220. Superseded by R 225.	
223	Floating-point Logarithm	A P	T	2	0,1	1,4,5,6,7	-	68	$p' = \log_e p$ . $p'$ and $p$ are floating-point numbers. Uses R 221.	
224	Logarithm, variable range	A P	T	3	0,1	5,6,7	-	42 - 58	$p' = \frac{1}{2^\pi} \log_e (pq)$ . $\pi$ is specified by a preset parameter.	
225	Shorter F.P. Exponential	A P	T	3	0,1	1,6,7	-	71	Supersedes R 222. $p' = \exp p$ . $p$ and $p'$ are floating-point numbers. Uses R 220.	
240	Sine or cosine	A P	T	2	0,1	1,5,6,7	-	24	$p' = \sin \pi p$ or $p' = \cos \pi p$ .	
241	Inverse tangent	A P	T	4	0,1	All except 1	-	48	$p' = \frac{1}{\pi} \arctan p$ or $p' = \frac{1}{\pi} \arctan \left( \frac{p}{q} \right)$ .	
242	Inverse sine or cosine	A P	T	2	0,1	All	-	130	$p' = \frac{1}{\pi} \arcsin p$ or $p' = \frac{1}{\pi} \arccos p$ . (Uses R 200 and R 241).	

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
243	Cosine and Sine	A	P	T	3	3,4,5	1,4,5,6,7	-	42	SP. $p' = \cos \pi p$ and $q' = \sin \pi p$ .
244	Tan/Cot	A	S	T	3	0,1	1,6,7	-	29	SP. $p'/q' = \tan \pi p$ .
250	F.P. Cosine and Sine	A	S	T	8	0	6,7	-	99	$p' = \cos p$ and $q' = \sin p$ where $p', p, q'$ are floating-point numbers.
251	F.P. Arctan	A	P	T	8	0,1	4,5,6,7	-	88	$p' = \arctan p$ or $p' = \arctan \left(\frac{p}{q}\right)$ where $p', p, q$ are floating-point numbers.
260	Complete Elliptic integrals	A	S	T	4	0,1	1,4,5,6,7	-	200	$p' = \frac{\pi}{2K}; q' = \frac{2E}{\pi}$ .
261	Error Function	A	S	T	2	0,1	4,5,6,7	-	38	Given $p = \frac{1}{2}x, p' = \operatorname{erf}(x)$ .
262	Elliptic Functions	A	S	T	12	0,1,2	All	0,1	2 secs. (Average)	$x_1' = \operatorname{sn}(4Kq, p)$ $p' = \operatorname{cn}(4Kq, p)$ $q' = \operatorname{dn}(4Kq, p)$ . (Uses R 200, 240, 241, 242).
270	F.P. Bessel Functions	A	S	T	34*	0	1,6	0,1	100 (Average)	SP. $p' = f_q(p)$ for $f = I_0, I_1, K_0, K_1, J_0, J_1, Y_0, Y_1$ . $p$ and $p'$ are floating-point numbers. (Uses R 200, 220, 221, 225, 250).
300	Operations Gaussian Quadrature	A	P	T	2* + roots & weights	0,1	5,6	0	See Specification.	Evaluates $\int_{X-h}^{X+h} f(x)dx$

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
310	Differentiation of a Table	A S	T	5	All	All	-	30 $\pi$	$\pi$ is the number of points in the table.	
320	Linear Interpolation	A P	T	3	0,1	5,6,7	-	58 - 72	$p' = f(p)$ .	
321	Polynomial Interpolation	A P	T	8	0,1,2	6,7	0	About 5 $\pi^2 - 2\pi + 100$	$(\pi - 1)$ is the order of interpolation.	
322	Two way Linear Interpolation	A P	T	5	0,1,2	1,5,6,7	-	128 - 142	$p' = f(p, q)$ .	
323	Two way Polynomial Interpolation	A P	T	10	0,1,2,3	5,6	0	5.2 $\pi^3 + 6\pi^2 + 32\pi + 170$	$p' = f(p, q)$ .	
324	Floating-point Interpolation	A S	T	9	All	6,7	0	48 $\pi^2 - 9\pi + 150$	$p' = f(p)$ [Floating-point] Uses R 610 $(\pi - 1)$ is the order of interpolation	
327	Polynomial interpolation (unequal intervals)	A P	T	8	0,1,2,3	6,7	0	1.25 $\pi + 6\pi^2 + 4.5\pi + 120$	$p' = f(p)$ $\pi$ is the number of values in the table $\pi - 1$ is the order of interpolation	
328	Two way polynomial interpolation (unequal intervals)	A S	T	15	0,1,2,3,4	5,6,7	0,1	5.5( $u_x^2 + u_y^2$ ) + 18 $u_x u_y$ + 75 $u_x + 45u_y + 460$	$p' = f(p, q)$ . $u_x, u_y$ are the number of values of $x$ and $y$ used in interpolation.	
332	Straight Line Fitting (asymmetrical)	A S	T	4	0,1,4,5	6,7	0	11 $\pi + 100$	Fits a straight line $y = a_0 + a_1 x$	
333	Least Squares Parabola Fitting	A S	T	9	0,1,3,4,5	5,6,7	0	20 $\pi + 280$	Fits a curve $y = a_0 + a_1 x + a_2 x^2$	
334	Straight Line Fitting (symmetrical)	A S	T	4	0,1,3,4,5	6,7	0	13 $\pi + 100$	Fits a straight line $ax + by = 2^{-v}$ $\pi$ = number of points given	

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
340	Zero of a Function	A	P	T	5	0,1	6	1	See Specification	Finds a zero of $f(x)$ to within a tolerance specified as a preset parameter, starting from two approximations.
341	Linear inverse interpolation	A	P	T	4	0,1	6,7	0	$86 + 2.5 r$	$f(p') = p$ , (where $p$ lies between the $r^{\text{th}}$ and $(r + 1)^{\text{th}}$ entries in a table). $p = f(p'q)$ .
342	Two way linear inverse interpolation	A	S	T	11	0,1,2,3	5,6	0		
2350	Roots of Polynomials	A			95	All	All	-	$\frac{n}{2}$ sec. per iteration	CP. Double-length floating-point. Bairstow's Method. $n$ is the degree of the polynomial.
360	Power Series Economisation	A	S	T	12	All	All	-		Used with R 650. Calculates coefficients of Chebyshev polynomials, economises a power series and prints its coefficients.
361	Evaluation of Polynomial. Floating-point	A	S	T	3	All	All	-	$121n + 31$	$F(z) = \sum_{j=0}^n a_j z^{n-j}$ , where $z = p+iq$ and the coefficients $a_j$ are real. Uses R 610.
2365	Fourier Series Mk. 2.	A	†			All	All	-	$\left(4 + \frac{n}{8} + \frac{8v}{3} + \frac{\pi v}{38}\right)$ seconds	CP. Calculates Fourier coefficients from $n$ readings of values of the function at equally spaced points throughout the period. $n \leq 400$ . $v$ is the number of harmonics: $v \leq \left\lfloor \frac{n}{2} \right\rfloor$ .

† Described in CS.165, 166a.



R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
	<b>Differential Equations</b>									
400	Simultaneous first-order differential equations $n > 8, v \neq 0$	A P	T	9*	All	3, 4, 5, 6, 7	-	$38n+4t+100$	<p>The solution of <math>n</math> first-order differential equations by a modified Runge-Kutta process. Scaling factor <math>2^v</math>. Time of auxiliary = <math>t</math>. Times given are for one step of the integration.</p>	
401	ditto $n > 8, v = 0$	A P	T	6	All	4, 5, 6, 7	-	$22n+4t+100$		
402	Simultaneous first-order differential equations $n \leq 8, v \neq 0$	A P	T	5*	All	5, 6, 7	-	$20n+4t$		
403	ditto $n \leq 8, v = 0$	A P	T	4	All except 1	5, 6, 7	-	$11n+4t$		
405	As R 401, but floating-point	A P	T	14	All	All	-	$490n+4t+300$	<p>The solution of <math>n</math> first-order differential equations using Simpson's formula. Time of auxiliary = <math>t</math>. Times given are for one step of the integration.</p>	
407	As R 403, but floating-point	A P	T	10	All	All	-	$480n+4t+60$		
408	Simultaneous first-order differential equations $n \leq 8, v = 0$	A S	T	18	All	All		$19n+t+25$ $(n \leq 4)$ $19n+t+41$ $(n > 4)$		
409	ditto $n \leq 8, v \neq 0$	A S	T	20	All	All		18 - 34 extra to R 408	<p>An addition to R 408 to prevent overflow of the variables. Scaling factor <math>2^{-v}</math>.</p>	
411	Simultaneous first-order differential equations $n \leq 8$	A S	T	11	All	1, 4, 5, 6, 7	1	$19n+5t+44$	<p>Similar to R 403 but more comprehensive, using Kutta-Merson process. Time of auxiliary = <math>t</math>. Times given are for one step of the integration.</p>	

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
	<b>Linear Algebra</b>									
2500	Matrix Interpretive Scheme	A	S	T	107	All	All	-		A floating-point scheme to facilitate matrix manipulation.
2501	Error Tracer for Matrix Programmes	A	S	T	3.3	3, 4, 5	2, 3	0	2 secs. per instruction	Used with R 2500. Prints matrix instructions immediately before they are obeyed.
2502	Magnetic Tape Matrix Interpretive Scheme	A	S	T	114	All	All	-		An extension of R 2500, using magnetic tape as a backing store.
2503	Double-length Matrix Interpretive Scheme	A			107	All	All	-		Similar to R 2500 but numbers are held in three words as in R 650.
2510	Solution of simultaneous linear equations	A	†	T	21	All	All	-		CP. Solves $n$ equations with $r$ right-hand sides. If $r=1$ then $n \leq 85$ . If $r=n$ then $n \leq 50$ . (4096) If $r=1$ then $n \leq 115$ . If $r=n$ then $n \leq 67$ . (7168)
511	Simultaneous linear equations subroutine	A	S	T	15	All	All	Depends on $r$	$(3n+8r) \times (n^2+7n+5)$	Solves $n$ equations with $r$ right-hand sides.
2530	Latent roots and vectors. $n \leq 54$ .	A	S	T	119	All	All	-		CP. Determination, by iteration, of up to 16 roots and vectors of a matrix with real roots.
2532	Latent roots and vectors. $n \leq 33$ .	A	S	T	128	All	All	-		CP. Similar to 2530 but rather easier to use.
7534	7168 Latent roots and vectors. $n \leq 54$ .	A	S	T	252	All	All	-		CP. 7168 version of R 2532.

† Described in CS.132, 133, 134

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
2535	Latent roots and vectors. $n \leq 400$	A			112	All	All	-		CP. Similar to R 2530, but requires 2 magnetic tape mechanisms. Evaluates up to 24 real roots and vectors.
2536	Complex latent roots. $n \leq 48$	A			123	All	All	-		CP. Evaluates all the latent roots (real or complex) of a real matrix.
7537	Latent roots and vectors. $n \leq 54$	B								CP. An amendment to R 2530, for the 7168 Store, to enable up to 32 roots and vectors to be extracted.
550	Invert symmetric matrix. Floating-point	A	S	T	29	All	All	Depends on $n$	$14n^3 + 50n^2 + 111n + 1000$	Replaces a symmetric matrix by its inverse. Floating-point throughout. $n$ is the order of the matrix.
551	Matrix division. Floating-point	A	S	T	14	All	4, 5	0	$19n^2r + 8.5n^3 + 118n^2 + 181n$	Evaluates $A^{-1}B$ . A has dimensions $n \times n$ ; B has dimensions $n \times r$ . Floating-point throughout.
560	Multiply symmetric matrices. Floating-point	A	S	T	11	All	All	-	$26n^3 + 66n^2 + 25$	Forms the product of two symmetric matrices. Floating-point throughout. $n$ is the order of the matrices.
570	Transpose in situ. Floating-point	A	S	T	4	0, 1, 2	-	0	$44mn + 93$	Dimensions of Matrix $m \times n$ .
2590	Linear Programming. Simpfix 32 Mk. 5	A	†		60	All	All			CP. Constraints $m$ 14 30 46 Variables $n$ 191 95 63
7591	Linear Programming. 7168 Simpfix Mk. 6	A			72	All	All			CP. $m \leq 166$ $m+n \leq 205$ $mn + 9n + m \leq 5973$

† Described in CS.186, 187.

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
600	Programmed Arithmetic Autocode	A	S	T	111*	All	All		About 70 per instruction	A floating-point conversion scheme to simplify the programming of special jobs.
610	Floating-point Arithmetic	A	P	T	4	3,4,5	1,4,5,6,7	-	18	SP. Multiple-entry routine for addition, subtraction, multiplication and division of packed floating-point numbers.
611	Floating-point Square Root	A	P	T	3	0,1	1,5,6,7	-	15 - 50	$p' = \sqrt{p}$ , where $p$ and $p'$ are floating-point numbers.
612	Shorter F.P. Arithmetic	A	P	T	3	4,5	1,4,5,6,7	-	14% (+, -) 11% (x) 29 (%)	SP. Similar to R 610. Uses only U4 and 5. Will give results in non-standard form in rare special cases.
630	Complex Arithmetic Interpretive Scheme	A	S	T	40*	1,2,3,4,5	1,6,7	0		Three words per number (1 for real argument; 1 for imaginary argument; 1 for exponent).
650	Double-length floating-point Arithmetic	A	S	T	35*	1,2,3,4,5	1,6,7	0		Three words per number (2 for argument; 1 for exponent).
670	Arithmetic with Rational Fractions	A	S	T	5	0,1	1,4,5,6,7	-		
700	Data Processing Output Conversion	A	S	T	1*	0	3,4,5,6,7	-		SP. Converts an integer $< 10^6$ into 6-bit characters.

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	B	X		
710	Input Conversion	A	S	T	2	0,1,5	1,5,6,7	-		SP. Converts words of $n$ 6-bit characters into $n$ -digit integers. $n \leq 6$ .
720	Merging Sort (Maximum String)	A	S	T	7	All	All except 1	0	See Specification	Sorts positive numbers into ascending order in the Main Store.
721	Data Sort	A	S	T	8	All	All except 1	0	"	Sorts keywords, each followed by one data word.
722	Double-length keyword Sort	A	S	T	9	All	All	0	"	Sorts double-length numbers.
723	$N$ -length Sort	A	S	T	9	All	All except 1	0	"	Sorts $N$ -length numbers.
730	Sort Main File (Variable length file items)	A	S		About 65	All	All	0,511		Merging process using 4 magnetic tape units.
736	Statistical Sort (Merging)	A	S	T	10	All	All except 1	0	See Specification	Sorts keywords and adds data words when keywords are equal, using a merging process.
737	Statistical Sort (Indexing)	A	S	T	4	0,1,2	1,3,5,6,7	-	"	As R 736 but indexes keywords one by one as they are presented and sorts them later using the index.
738	Scramble Store	A	S	T	4	0,1,2,3	1,2,6,7	-		SP. Looks up random keyword, or indexes new keyword.
740	PAYE Mk. 2	A	S	T	23*	All	All	-	See Specification	Comprehensive Tax Evaluation.
741	PAYE Code Conversion	A	S	T	12	0,1,2	2,7	0		Reads tax code, looks up table A and prepares special code for R 740.

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
761	File Updating Mk. 2	A	S		53	All	All			Updates a Main File using programmers' sub-routines for the actual amendments and insertions. Permits change in length of items during amendment. Requires at least 3 magnetic tape units.
	<b>Applications</b>									
2800	Pipe Stressing Mk. 2	A	†		About 100	All	All			CP. Computes stresses for a 3-dimensional pipe system with up to 8 anchors.
2801	Frame Stressing (Livesley Method) Mk. 1	A	‡			All	All			CP. Performs elastic analysis of two-dimensional rigid frame of up to 18 joints and 56 members.
2802	Frame Stressing (Livesley Method) Mk. 2	A				All	All			CP. Performs elastic analysis of large two-dimensional rigid frames by partitioning into sub-frames.
2810	Multiple Regression Mk. 1B	A	‡			All	All			CP. ≤ 26 variables. Correlation, regression, significance tests.
2811	Multiple Regression Mk. 4	A				All	All			CP. ≤ 38 variables. Correlation and regression.
850 -899	<b>Individual Pegasus Users' Private Routines</b>									

† Described in CS.230  
 ‡ Described in CS.194  
 ‡ Described in CS.273

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
	<b>Miscellaneous</b>									
2900	Compare Tapes	A	S	T	2	All	All	-	85 characters per second	CP. Compares two tapes at high speed, stopping if they are not identical.
2901	Clear Store	A	S	T	1	All	All	All	1.6 secs.	CP. Clears whole Main Store except Date and Serial Number.
2902	Identification	A	S	T	3	All	All	All	3 secs.	CP. Writes the integers 0 to 4093 into Main Store locations 0 to 4093.
2903	Floating-point print (Non-Assembly)	A	S	T	10	All	All	-		CP. Prints floating-point numbers from the Main Store in a manner analogous to the warning character F.
2904	Clear Magnetic Tape	A	S	T	3	All	All	-	41 or 53 per section	CP. Clears sections of magnetic tape as specified by a parameter tape.
2905	Testaid Break-point	A	S	T	10	All	All	0		Aids programme development by facilitating the printing of intermediate results.
2906	Fast block transfer translation	A	S	T	3	All	All	-		CP. Translates a punch on block transfer tape more rapidly than the Initial Orders.
2907	Store Use	A	S	T	1	All	All	0	20 secs.	CP. Prints one character indicating the contents of each block in the Main Store.
2915	Independent Double-Length Fraction Print	A	S	T	7	All	All	-		CP. Prints double length fractions in a manner analogous to the warning character F.

R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
2921	Binary Translation Routine (See also R 1030, 1031).	A	S	T	17	All	All	-		CP. Translation routine for converting tape containing relative addresses and warning characters from I.O. notation to BINARY INPUT notation.
2922	End of Library	A	S	T	4	All	All	-		CP. Appears at end of Library Tape. Prints out missing tag functions if Assembly has failed to find everything it needs.
2923	Binary to Initial Orders Translation	A	S	T	18	All	All			CP.
7924	Convert programme for 7168 Store	A	S	T	54	All	All	All	3 secs. per block	CP. Converts a binary punched programme for use with the 7168 Store.
7925	Convert programme for 4096 Store	A	S	T	63	All	All	All	3 secs. per block	CP. Converse of R 7924.
7927	Binary to Maglib translation	A	S	T	11	All	All	See Spec.	About 0.5 secs. per block	CP. Records subroutines on magnetic tape in a form suitable for use with MAGLIB.
930	Check Magnetic Tape	A	S	T	6	0,1,2	1,6,7	0	130 - 1000	Checks write and 16/32 switches and optionally prints information about the tape.
951	Random Normal Deviates	A	S							A tape containing random normal deviates punched correct to two decimal places.
970	Standard atmosphere	A	S	T	6	0,1	1,5,6,7	-	53 or 109	Uses R 200 and R 220.
971	Triple Exponential Standard Atmosphere	A	S	T	4	0,1	1,4,5, 6,7		92 - 129	Uses R 220.



R	Brief Description	State	Spec.	Tapes	Store Blocks	Uses			Time (milliseconds)	Notes
						U	X	B		
980	Pseudo-Random Number Generator	A	S	T	1	5	1,2,6,7	-	5½	SP. Computes sequences of pseudo-random numbers. One number is produced and left in X2 each time subroutine is entered.
981	Random Numbers in Main Store	A	S	T	5	0,1,2,3,4	-	0	120 per block	Fills a chosen sequence of locations in the Main Store with pseudo-random digits.
990	Read/Write Magnetic Tape	A	S	T	4	All		0	48 or 64 per section	Transfers information from tape to the Main Store or from the Main Store to tape. Faster than R 1032 or R 1044.
2991	Copy Magnetic Tape	A	S	T	8	All	All	All	96 or 133 per section	CP. Copies information from one magnetic tape to another under control of a steering tape. Faster than R 1045.
2992	Compare Magnetic Tapes	A	S	T	7	All	All	All	160 or 245 per section	CP. Compares information on two magnetic tapes and prints a record of the sections which disagree.

R	Brief Description	State	Spec.	Notes
	<b>Isolated Routines</b>			
1025	Initial Orders (I.O.)	A		
1026	I.O. Integer Output	A		
1027	I.O. Fraction Output	A		
1028	I.O. Order Output	A		
1029	I.O. Date and Serial Number	A		
1030	Assembly	A		
1031	Binary Input	A		
1032	Read Magnetic Tape	A	†	Transfers information from tape to the Main Store. Requires steering tape.
1033	Binary Punch (See also R 53 to R 2059)	A	S	Replaces R 2050. Entered with A4 warning character. Punches out from the Main Store in a form suitable for Binary Input, R 1031. Time approximately 2 seconds per block punched.
	<b>Isolated Routines on 7168 Store</b>			
1040	Check Magnetic Tape	A	†	As R 930. Checks write and 16/32 switches and optionally prints information about the tape.
1041	Magnetic Tape Assembly	A	†	As R 1030 but with the library stored on magnetic tape.
	† Described in CS.127, Addendum 1			
	† Described in CS.265			

R	Brief Description	State	Spec.	Notes
1042	Read Standard Programme from Magnetic Tape	A	†	Programme called for by directive Q and routine number.
1043	7168 Read Magnetic Tape	A	†	Similar to R 1032, but different steering tape.
1044	Write to Magnetic Tape	A	†	Similar to R 1043, but transfers from the Main Store to magnetic tape.
1045	Copy Magnetic Tape	A	†	Copies sections, specified on a steering tape, from one magnetic tape to another.
1046	Print word as 6-bit characters	A	†	Prints information packed in 6-bit characters under the control of directive K.
1047	Load card Distribution and Interpretation buffers	A	≠	
1048	Load card Code Table	A	≠	

† Described in CS.265  
 † Described in CS.303

(File after index)

**Cues to Pegasus Subroutines**

When using a subroutine it is usual to call for the appropriate cue using an Assembly tag. Very occasionally it is necessary to know what the entry point is, and most specifications give this information after the cue number. The following list only includes subroutines whose specifications are incomplete in this respect.

The entry point is given in brackets after the cue number. The subroutine is entered by bringing the block specified into U0 and jumping to the order specified. Cues which do not obey this rule are given in full.

R 1	01	(0+ .2+)			
R 4	01	(0+ .3)	02	(0+ .0+)	03 (1+ .5)
	05	(0+ .7)	06	(0+ .5)	07 (1+ .4)
	09	(1+ .0+)			08 (0+ .2)
R 40	01	(0+ .0)	02	(0+ .2+)	
R 51	01	(0+ .6)	02	(0+ .6+)	03 (0+ .5+)
					04 (0+ .5)
R 52	01	(0+ .0)			
R 100	01	(0+ .0)	02	(a-order partial cue)	03 (b-order partial cue)
R 105	01	(0+ .0)			
R 140	01	(0+ .0)			
R 200	01	(0+ .0)			
R 201	01	(0+ .0)			
R 220	01	(1+ .0)			
R 260	01	(0+ .5)			

R 400	01	0+ <span style="border: 1px solid black; padding: 0 2px;">1</span> 72 1.0 0 60	02	0+ <span style="border: 1px solid black; padding: 0 2px;">1</span> 72 1.2 0 60	03	5+ <span style="border: 1px solid black; padding: 0 2px;">0</span> 72 4 34 4 01
-------	----	---	----	---	----	--

04	0+ <span style="border: 1px solid black; padding: 0 2px;">1</span> 72 4 1.2+ 0 60	05	(a-order partial cue)
		06	(b-order partial cue)

R 401	01	4+ <span style="border: 1px solid black; padding: 0 2px;">1</span> 72 1.0 0 60	02	4+ <span style="border: 1px solid black; padding: 0 2px;">1</span> 72 1.0+ 0 60	03	4+ <span style="border: 1px solid black; padding: 0 2px;">1</span> 72 4 1.5 0 60
-------	----	---	----	--	----	---

04	(a-order partial cue)	05	(b-order partial cue)
----	-----------------------	----	-----------------------

R 402	01	(0+ .0)	02	(0+ .0+)	03	(0+ .7)
	04	(a-order partial cue)	05	(b-order partial cue)		

R 403    01   (0+ .4)            02   (0+ .4+)            03   (0+ .0)  
           04   (a-order partial cue)            05   (b-order partial cue)

R 511    01   (0+ .0)            02   (0+ .2+)

R 650    01   

21+ <span style="border: 1px solid black; padding: 0 2px;">1</span> 72
1.7 0 60

            02   

21+ <span style="border: 1px solid black; padding: 0 2px;">1</span> 72
1.0 0 60

  
           03   a-order partial cue            05   (21+ .1)  
           04   b-order partial cue

R 670    01   

0+ <span style="border: 1px solid black; padding: 0 2px;">1</span> 72
1.5+ 0 60

            02   

3+ <span style="border: 1px solid black; padding: 0 2px;">1</span> 72
1.5+ 0 60

            03   

3+ <span style="border: 1px solid black; padding: 0 2px;">1</span> 72
1.3 0 60

© FERRANTI LTD 1958

*Not to be reproduced in whole or  
 in part without the prior written  
 permission of Ferranti Ltd.*

Ferranti Ltd., London Computer Centre, 21, Portland Place, LONDON W.1.

FERRANTI LTD

PEGASUS LIBRARY SPECIFICATION

Issue 1  
27.7.56.

GENERAL PURPOSE NUMBER PRINT

A self-preserving subroutine for printing scaled numbers in various styles; it may be used for integers or fractions. The routine prints from X1.

Name: NUMBER PRINT

Store: 11 blocks + parameter-list (an optional parameter-list occupies 2 blocks; this may be replaced by a programmer's parameter list, which may have any length).

Uses: U0, B0. (Self-preserving in U0).

Cue: 01 (self-preserving after one use - reentry to 0.2+)

Link: Set in X6. It may be either (a) a 'go' order-pair (which will be obeyed in 0.1), or (b) a computing-store link.

CONTENTS

	Page
1. General Description .. .. .	1
2. Method of Use .. .. .	2
3. The Standard Styles .. .. .	3
4. The Parameter-List .. .. .	5
5. The Scaling-Parameter .. .. .	5
6. The Layout-Parameter .. .. .	6
7. The Selection of a Style .. .. .	9
8. Notes .. .. .	9

APPENDICES

A. Catalogue of Layout Parameters .. .. .	10
B. Right-Hand Zero Suppression .. .. .	17
C. Detailed Description of the Layout-Parameter .. .. .	18
D. Extra Zeros .. .. .	20
E. Complete Suppression of Left-Hand Digits .. .. .	21
F. The Parameters for the Standard Styles .. .. .	22

1. General Description

1.1 R1 is designed to print scaled or unscaled fractions or integers in a variety of forms. Each time it is called in it prints the number in accumulator 1, after

multiplying it by a scaling factor. The way in which the printed number appears is determined by a layout parameter; there is provision for including any of the following:-

- (a) Printing CR LF before the number,
- (b) Preceding the number by up to 7 spaces (*initial spaces*),
- (c) Omitting left-hand zeros (*omission*) or replacing them by spaces (*suppression*) in the integral part of the number,
- (d) Printing the sign *immediately* (i.e. after the initial spaces) or just before the first significant digit (*delayed sign*) or omitting the sign entirely,
- (e) Printing up to 12 digits of the number (up to 11 if it is a fraction),
- (f) Printing the decimal point where required, or omitting it,
- (g) Omitting or suppressing right-hand zeros after the decimal point,
- (h) Printing internal spaces to divide the digits into groups,
- (i) Printing extra zeros after the number,
- (j) Following the number by up to 7 spaces (*final spaces*),
- (k) Omitting entirely the first few digits of an integer and then printing the remaining digits.

The sign (if called for) is printed as a space (for a non-negative number) or as a minus sign (for a negative number).

**1.2** The style of printing is selected from a preset-parameter list in which *two* parameters are required for each style; the first of these determines the layout (the various items listed above) and the second is the scaling factor which is used before printing. There is an optional parameter-list which includes some of the more common styles. In order to use any other style (and there are many thousand possible variants) it is necessary only to include a suitable parameter-list (of any length) in the Master-programme; this will then replace the optional list.

**1.3** The routine may be called in by a normal cue (01) and link (in X6). Alternatively a computing-store link may be used with such a cue. When it has been used it resets its own first block in U0 and it is therefore a self-preserving routine: it may be re-entered by jumping to 0.2+ (the link may be of either form). It may alternatively be used in conjunction with R3, a block layout routine, which lays out the printed page in lines and blocks as required. The way in which R1 is used with R3 is described in the specification of R3.

## **2. Method of Use**

**2.1** The routine is entered by cue 01. Before entry the number to be printed is placed in X1, the link is set in X6 and an integer which determines the style of printing is set in X7 (see Section 2.5 overleaf).

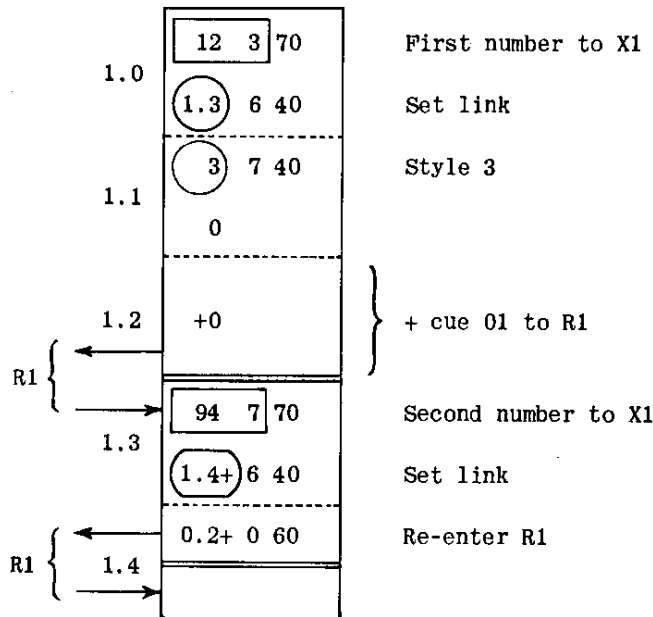
**2.2** The link may be either an order-pair, in which case it must be a 'go' order-pair, or a computing-store link (set by a 40 order). The routine is normally self-preserving and may be re-entered by a jump to 0.2+ provided U0 has not been disturbed in the meantime. The link remains in X6 on exit. An order-pair link is obeyed in 0.1.

**2.3** It should be noted that if R1 and R3 are *separately* called for by the Master-Programme, then U0 will *not* be correctly reset on leaving R1 for the Master-programme by a *computing-store link*. Furthermore the optional printing which Assembly performs during input of the programme will show addresses. etc. for R3, *whether or not* R3 has been called for (if R3 has not been called for the optional printing will show it as not requiring a parameter-list).

**2.4** The optional parameter-list of R1 permits the use of eight *standard styles*; these are numbered from 1 to 8. If other styles are needed a new parameter-list must be supplied; an interlude in R1 ensures that, in this event, the optional parameter-list is overwritten so that no space need be allowed for it. Each style requires two (consecutive) parameters and a parameter-list may provide as few or as many styles as desired.

**2.5** The positive integer  $s$  set in X7 before entry to R1 determines the style of printing; it causes R1 to select the  $s$ -th pair of parameters from the parameter-list. The first of these is the layout-parameter and the second is a scaling factor. Since  $x_7$  is restored on exit it is not necessary to reset the style number when using R1 again, unless of course it is to be changed. The styles are numbered from 1 upwards.

**2.6** As an example, suppose we require to print the numbers in locations B12.3 and B94.7 in standard style no. 3 (it is assumed that R3 is not being used):-



### 3. The Standard Styles

**3.1** Eight *standard styles* are provided by the optional parameter-list; all of these end with two spaces. In the following description a space is represented by /. Note that a positive sign is always printed as a space. The number in X1 on entry to R1 is referred to as  $z$  if it is a fraction ( $-1 \leq z < 1$ ) or as  $N$  if it is an integer, i.e. if  $z = N.2^{-38}$ . The Layout Symbols are described later (Section 3.5). The *width* is defined to be the horizontal space occupied by the printing (including spaces). Unless otherwise mentioned the sign is delayed (i.e. it is printed just before the first digit).

**3.2** The following are the specifications of the standard styles:-

- |  | <i>Specimens</i>       |
|--|------------------------|
| (1) Print $z$ to 11D (i.e. as a fraction to 11 places after the decimal point) on a new line; there is a space after the fifth digit following the point, and there is one digit (normally zero) before the point. | CR LF/0.14285/714286// |
|  | CR LF-0.25000/000000// |
|  | CR LF/0.00000/000000// |
|  | CR LF-1.00000/000000// |
| Layout Symbol: $n9.99999/999999//$   |                        |
| Width: 17.   |                        |



- |  |  |
|--|--|
| (2) As (1) but without CR LF.<br>Layout Symbol: 9.99999/999999//<br>Width: 17.   | /0.14285/714286//<br>-1.00000/000000//   |
| (3) Print $z$ to 6D.<br>Layout Symbol: 9.999999//<br>Width: 11.  | /0.142857//<br>-0.250000//<br>/0.000000//  |
| (4) Print $N/10^6$ to 6D with left-hand<br>and right-hand zero omission.<br>Layout Symbol: xxxxx0.xxxxxx//<br>Width: variable from 5 up to 16.   | /123456.789012//<br>-0.125//<br>/1030//<br>/15.25//<br>/0//                              |
| (5) Print $N$ on a new line up to<br>12 digits. Left-hand zero omission, and<br>thus a straight left-hand margin.<br>Layout Symbol: nxxxxxxxxxxx9//<br>Width: variable from 4 to 15.                   | CR LF/274877906943//<br>CR LF-274877906944//<br>CR LF-10000//<br>CR LF/24//<br>CR LF/0// |
| (6) Print $N$ with up to 12 digits, with<br>a space after the millions digit.<br>Left-hand zero suppression, and thus<br>a straight right-hand margin.<br>Layout Symbol: 333333/333339//<br>Width: 16. | -274877/906944//<br>/////12/345678//<br>//////////-950//<br>//////////0//                |
| (7) Print $N$ without sign, with up to 3<br>digits. Left-hand zero suppression.<br>Layout Symbol: +xx339//<br>Width: 5.  | 123//<br>//5//<br>//0//  |
| (8) As (7) but up to two digits.<br>Layout Symbol: +xx39//<br>Width: 4.  | /2//<br>10//   |

**3.3** Any layout with variable width should be used with caution since the layout of a page will be irregular if a number having variable width is followed on the same line by other numbers. For this reason layouts of constant width are to be preferred. The total width available on the teleprinter is 69; consequently the sum of the widths of the numbers on any one line should not exceed 69.

**3.4** The last two standard styles are intended for printing small unsigned integers. If R1 is called on to print a negative number in such a style then it will print the modulus (e.g. -10 will be printed as 10).

**3.5** The Layout Symbol is a convenient way of specifying the layout produced by any style. The symbols given above for the standard styles use the following conventions:

- 9 represents a digit which is *always printed*,
- 3 represents a digit which is *liable to suppression* (i.e. to being replaced by a space) because of left-hand or right-hand zero suppression,
- x represents a digit which is *liable to omission* for similar reasons,
- / represents a *space*,
- . represents the *decimal point*,
- n indicates that the printing is on a *new line*, i.e. is preceded by CR LF,

- + indicates that the number being printed is known to be *positive* (or zero) and its sign is therefore omitted - the absence of this character in a layout symbol is taken to indicate the normal case of a delayed sign.

It should be noted that the decimal point itself is liable to suppression if it is followed in the layout symbol by 3 or x.

#### 4. The Parameter-List †

4.1 If any style of printing is required which is not one of the eight standard styles then a parameter-list must be supplied by the programmer. Such a list may provide for as many styles as may be needed and it is arranged that it *replaces* the optional parameter-list. This means that if the programmer wishes to use any standard styles in addition to his own styles then he must include the relevant parameters in his own parameter-list. The parameters for the standard styles are given in Appendix F.

4.2 Two consecutive parameters are needed in the parameter-list for each style. If R1 is called in to print a number in style *s* then the *s*-th pair of parameters (i.e. the parameters numbered  $2s - 1$  and  $2s$ ) will be used. The first of these two parameters is called the layout-parameter and the second is the scaling-parameter. Thus to provide six different styles of printing a parameter-list containing twelve parameters would have to be supplied; since 12 is  $8 + 4$ , i.e. '14' in octal, the title of this parameter-list would be as follows:-

R	0	0	-1	4
	1	-	04	-

As usual the parameter-list may be regarded as forming a part of the master-programme. If a parameter-list is supplied then it must appear before the A2 on the programme tape; also it must start in the first 128 blocks of the main store.

4.3 The layout-parameter for a given style determines whether the printing is to be preceded by CR LF and/or by spaces, how many digits of the number are to be printed, whether these digits are to be separated into groups by spaces, where the decimal point (if any) is to appear, the treatment of non-significant left-hand zeros, etc. This parameter is normally written as a pseudo order-pair; its construction is described in Section 6.

4.4 The scaling-parameter is a negative number which is written as an integer or a fraction. The way in which its value may be determined is given in Section 5.

#### 5. Determination of the Scaling-Parameter

5.1 The scaling-parameter is the second of the two preset-parameters determining the style of printing. The method of determining this parameter may be summarized as follows ( $z$  is the number in X1 on entry to R1):-

- (a) If  $z$  is a fraction and we wish to print  $cz$  to  $n$  places after the decimal point, then the scaling parameter is to be written as the *negative integer*

$$-10^n c.$$

This must be within capacity so that  $0 < c \leq 10^{-n} \times 274877906944$ .

- (b) If  $z = N.2^{-38}$  and we wish to print  $kN$  to  $n$  places after the decimal point, then the scaling-parameter is to be written as the *negative fraction*

$$-10^n k.$$

This must be within capacity so that  $0 < k \leq 10^{-n} \leq 1$ . In particular if  $k = 1$  then the parameter is  $-1.0$ .

† NR Care must be taken not to overwrite the parameter list of R 1 at any stage.

**5.2** It will be noted that if  $z$  is a fraction then the scaling-parameter is an integer, and vice versa. One of the consequences of the above restrictions on the scaling factor  $c$  used with fractions is that  $c$  cannot be specified more precisely than to  $n$  figures after the decimal point. Note also that an integer cannot be scaled up (since  $k \leq 1$ ); it is, however, possible to print extra zeros after an integer (e.g. one can print 5 as 5000) - this can be done by the layout-parameter (see Appendix D).

**5.3** As an example, suppose we require to print  $z$  to 5D, i.e. to 5 figures after the decimal-point. The scaling-parameter is to be written as:-

-100000

If we wish to print  $19.4z$  to 3D then the parameter is:-

-19400

To print  $N = 2^{38}z$  as an integer the parameter is:-

-1.0

To print  $0.014286N$  to 1D then the parameter is:-

-0.14286

To print  $10N$  as an integer an extra zero must be supplied with the help of the layout-parameter; the scaling-parameter is:-

-1.0

**5.4** A rounded multiplication is used when R1 multiplies  $z$  by the scaling-parameter; the resulting number in X6 is then treated as an integer and its successive decimal digits are considered for printing (the decimal point is printed at the appropriate moment). It follows from this that at most 12 digits of the number can be printed (the first of these twelve can be 0, 1 or 2 only). The layout-parameter determines the treatment of these 12 digits; in particular it can be arranged that certain of them are ignored, certain are followed by spaces, one by the decimal point, and so on.

## 6. The Layout-Parameter

**6.1** The layout-parameter for any particular style is written as a pseudo order-pair. If it is a 'stop' order-pair then CR LF will be printed on entry to R1, so that the number will appear on a new line; after this the action is the same as when the parameter is a 'go' order-pair.

**6.2** We shall write the layout-parameter as follows:-

$N$	$X$	$S$	$\phi_0$	$q_0$
$\phi_1 \cdot q_1$	$\phi_2$	$q_2$	$\phi_3$	$q_3$

All the symbols here (apart from  $N$ ) stand for octal digits, as they do in an ordinary order-pair, and may therefore take values between 0 and 7. The layout-parameter should be marked as a 'stop' order-pair if required.

**6.3** The first action to be taken by R1 (after the possible CR LF) is to print some spaces; there may be none of these or any number up to 7. These are called *initial spaces* and they may be used to separate one number from another or to indent a line. The number of initial spaces is  $7-S$ .

**6.4** In order to determine the values to be assigned to the other symbols in the layout-parameter we must consider the printed form,  $Z$ , of the number. Suppose  $Z$  contains a maximum of  $m$  decimal digits, including those on either side of the decimal point. The number  $m$  may not exceed 12 (see Section 5.4) and we shall first assume that  $m$  is at least 5 (but see Section 6.12 overleaf). The constant  $X$  in the layout-parameter must be set equal to  $12-m$ . Since  $X$  is an octal digit it can take values from 0 to 7.

**6.5** At present we need consider only a few values of  $N$ , the  $a$ -order address; in fact as a general rule we write 1.2 if  $Z$  contains a decimal-point, or 1.0 if  $Z$  is an integer. These values of  $N$  provide for left-hand zero suppression or omission and for delayed printing of the sign (i.e. the sign will appear just before the first digit printed). If we wish to omit the sign entirely we need only write 0.2 (or 0.0) instead of 1.2 (or 1.0). All these values of  $N$  will ensure that the digit immediately preceding the decimal point (or the last digit of an integer) will always be printed even if it is zero) Other values of  $N$  are seldom used; they are described in Appendices B and C.

**6.6** To determine the remaining octal digits (the  $p$ 's and  $q$ 's) in the layout-parameter we must divide the  $m$  digits of the printed form of  $Z$  into groups

- (a) at the decimal-point (if any),
- (b) at any internal spaces, (an internal space is a (single) space used for layout purposes to separate groups of digits).
- (c) wherever else is required.

Each group may contain from 1 to 7 digits; there may not be more than four groups. The  $p$ 's in the layout-parameter are the *group-counters*; they specify the number of digits in each group. If there are fewer than four groups the remaining  $p$ 's should be set equal to zero (they will be referred to as *null* groups). The group-counters are examined successively by R1, starting with  $p_0$ ; if any  $p_i \neq 0$  then R1 will print (or suppress or omit) the next  $p_i$  digits, the details being determined by  $q_i$ . As a special case, if  $p_i = 0$  then R1 will (in this case *only*) print  $q_i$  spaces and obey the link. The sum of all the  $p_i$  should equal  $m$ .

**6.7** The  $q$ 's in the layout-parameter are the *group-indicators*. The following rules determine most of the  $q$ 's (in all cases assuming that the corresponding  $p \neq 0$ ):-

- (a)  $q = 0$  in the group followed by the decimal point (or in the last group of an integer).
- (b)  $q = 2$  in a group followed by a space,
- (c)  $q = 3$  in most other groups.

Each of these values of  $q$  causes non-significant left-hand zeros in the integral part of  $Z$  to be replaced by spaces; when the appropriate action has been taken at the end of the group the next group is considered. If the next group has  $p = q = 0$  the link is obeyed immediately (the same applies if there is no next group). There is also the special case

- (d)  $q = 1$ , which causes R1 to print two spaces and obey the link.

If any of these values of  $q$  are increased by 4 the effects are similar except that left-hand zeros will be omitted instead of suppressed (replaced by spaces).

**6.8** As an illustration suppose the number  $Z$  has 4 figures in front of the decimal-point and 3 after it. The printing is to be preceded by three spaces.

Since there are three initial spaces,  $S = 7 - 3 = 4$ . We wish the decimal-point and sign to appear so  $N$  is 1.2. The total number of digits in  $Z$  is  $m = 4 + 3 = 7$  so  $X = 12 - 7 = 5$ . The digits may be divided into two groups at the decimal-point so we get

$$p_0 = 4, \quad p_1 = 3, \quad p_2 = p_3 = 0.$$

The first of these groups is followed by the decimal point so we set  $q_0 = 0$ ; the second group completes the printing so we set  $q_1 = 3$ . The complete parameter is therefore:-

1.2 5 44 0
3.3 0 00 0

The layout-symbol (see Section 3.5) would be written: ///3339.999

**6.9** As a further example, let us find the layout-parameter for printing a 10-digit positive integer with a space after the 5-th digit; the printing is to be preceded by CR LF and followed by a space.

Here the parameter must be a 'stop' order-pair since we want a preliminary CR LF; there are no initial spaces so  $S = 7$ . Since we are printing an unsigned integer we must set  $N = 0.0$  (see Section 6.5). The total number of digits to be considered for printing is  $m = 10$  and therefore  $X = 12 - 10 = 2$ . There are two groups, the first is of 5 digits followed by a space, the second is of 5 digits and this ends the integer; we therefore must set

$$p_0 = 5, \quad q_0 = 2; \quad p_1 = 5, \quad q_1 = 0.$$

To obtain a single final space we can insert a null-group with

$$p_2 = 0, \quad q_2 = 1.$$

The layout-parameter is therefore written as follows:-

0.0	2	75	2.
5.0	0	10	0

The layout symbol (see Section 3.5) would be written:-

n+33333/33339/

**6.10** It frequently happens that, although the printed number  $Z$  may contain a certain maximum number of digits, some of those at the left-hand end (before the decimal point) may in practice nearly always be zero. It would be a waste of time (and space on the page) if these zeros were replaced by spaces because of zero-suppression. For example we may wish to print  $Z = 100z$  to 5D; here  $Z$  will have only two digits before the point unless  $z = -1.0$ . If the case  $z = -1.0$  is unlikely to arise we can arrange to omit the first digit of  $Z$  if it is zero; if this digit is printed it will of course disturb the layout of the page. We can do this by inserting a group with  $q = 7$ ; this value of  $q$  causes unwanted left-hand zeros in its group to be omitted (instead of being replaced by spaces); in other respects it is analogous to  $q = 3$  (see Section 6.7). A group of digits having  $q = 7$  may be called a *type-7 group*, and its digits can often be regarded as 'guard' digits to guard against exceptional numbers or programming errors.

**6.11** To illustrate these points we will now construct the parameters for the printing of  $100z$  to 5D, the printing to be preceded by two spaces. The scaling-parameter (Section 5.1a) is written as the negative integer:-

-10000000

The desired layout may be represented by the layout-symbol (see Section 3.5):-

//x39.99999

The two initial spaces may be obtained by setting  $S = 5$ . The value of  $X$  is determined by the maximum number of digits in  $Z$ , i.e.  $m = 8$  including the guard digit, so that here  $X = 12 - m = 4$ . Since the decimal-point appears we have  $N = 1.2$ . We must divide the printed digits into three groups as follows:-

$$\begin{array}{ll}
 p_0 = 1, & q_0 = 7 \quad (\text{one-digit type-7 group}), \\
 p_1 = 2, & q_1 = 0 \quad (\text{two digits followed by decimal-point}), \\
 p_2 = 5, & q_2 = 3.
 \end{array}$$

The two parameters for this style will therefore appear as follows in the parameter-list:-

1.2	4	51	7
2.0	5	30	0
-10000000			

**6.12** The introduction of a type-7 group (or the extension of an existing one) enables us to avoid the restriction temporarily imposed on  $m$  in Section 6.4 above. For example to print an unsigned two-digit integer we make  $m$  up to 5 by adding a redundant group of 3 omitted digits. The parameters will read as follows (assuming two initial spaces):-

0.0 7 53 7
2.0 0 00 0
-1.0

**6.13** Appendix A gives a catalogue of layout-parameters for many useful styles. Appendix B shows how suppression of right-hand zeros in the fractional part of a number may be effected; this facility should be used with caution. Appendix C gives a detailed description of the various component parts of the layout-parameter. Appendix D shows how one may arrange to print extra zeros at the right of a number. Appendix E describes an occasionally wanted style of printing.

## 7. The Selection of a Style

**7.1** The catalogue of styles given in Appendix A will, it is hoped, enable one to select a style to suit most requirements. This Appendix gives the appropriate layout-parameter; the scaling-parameter is easily written down from the rules given in Section 5.1 above.

**7.2** As described in Appendix A (Section A.1) all the layouts given there may be considered as having constant width. Any layout with variable width should be used with caution - see Section 3.3. On no account should the total width on any line exceed 69.

**7.3** An integer, or the integral part of a number should normally have left-hand zero suppression with delayed printing of the sign since this approximates most closely to the 'natural' way of writing numbers. This is catered for in all the layouts given in Appendix A. Occasionally it may be desirable to omit the sign or to use an 'immediate' sign - if the latter is used the signs of a column of numbers will be vertically aligned, which may be advantageous. These effects are simply obtained as described in Section A.3. The device of right-hand zero suppression or omission should be used sparingly; it is described in Appendix B.

**7.4** Two spaces are usually adequate for separating adjacent numbers on the same line; it may be advisable to use three if the numbers have many digits which are grouped by internal spaces. These separating spaces may be obtained either as initial spaces (e.g.  $S = 5$  gives two spaces before the number) or as final spaces.

## 8. Notes

**8.1** A loop-stop in U0.0 will occur (or the first digit printed will be replaced by a character which is not a digit) if the number being printed is too large for the style being used. This may arise if the quantity  $X$  in the layout-parameter is too large (see Section 6.4). The number to be printed ( $z$ ) is first multiplied by the scaling-parameter (see Section 5.4) to produce a single length integer. This integer is immediately multiplied by  $10^X$  before any other action; this results in the first  $X$  of the 12 decimal digits of the integer being removed from consideration for printing; the above undesirable effects occur if these digits are not all zero.

**8.2** If a computing-store link is used it must be set with a 40 order since the rest of  $x_6$  must be zero.

## APPENDIX A

## Catalogue of Layout-Parameters

**A.1** This appendix gives a list of some 250 layout-parameters, grouped according to the total number of digits normally printed. With few exceptions every layout given includes a 'guard digit' at the left (see Section 6.10); if this digit is never printed then all the layouts have constant width (see Section 3.3). Each layout provides for the suppression of left-hand zeros in the integral part of the printed number, but the last digit of the integral part will always be printed even if the integral part is zero (except in those layouts specifying no digits before the decimal point).

**A.2** To save space the two pseudo-orders making up the layout-parameter are written here on the same line, they should of course be punched in the usual way (with CR LF after each pseudo-order). The number of *initial spaces* has not been specified; the constant *S* should be replaced by seven minus the number of initial spaces required (see Section 6.3). The number printed will be preceded by CR LF if the parameter is a stop order-pair, e.g. if . is punched after the *a*-order of the parameter.

**A.3** The *sign* will be printed as *Space* or *Minus* just before the first digit printed (delayed sign). The sign may, if desired, be printed immediately after the initial spaces (i.e. before any spaces representing suppressed zeros) (*immediate sign*), or it may alternatively be omitted. These effects are obtained by changing the *N*-address in the *a*-order of the parameter in the following way:-

a-order <i>N</i> -address as written in catalogue (For Delayed Sign)	Replace by	
	(For Immediate Sign)	(To omit Sign)
1.0	0.0+	0.0
1.2	0.2+	0.6
1.6	1.6 (No difference)	0.6

Note that the second octal digit of *N* is unaffected by these changes (see Appendix B).

**A.4** None of the layouts given has any *final spaces* (i.e. spaces following the number). Most of the parameters given may however be changed to provide final spaces; the rules are as follows:-

- (a) If  $q = 3$  in the last group having non-zero  $p$ :-  
To get *one* final space, replace  $q = 3$  by  $q = 2$ ,  
To get *two* final spaces, replace  $q = 3$  by  $q = 1$ .
- (b) If there is a spare group (i.e. a group having  $p = 0$ ):-  
To get  $F$  final spaces, replace  $q = 0$  by  $q = F$  ( $1 \leq F \leq 7$ ).  
(Do this to the first spare group if there are more than one).

Note that final spaces cannot be provided with the four layouts marked ' $F = 0$ '.

**A.5** Most of the layouts given have one *guard digit*, which is *not* included in the total number of digits. The layouts with 3 or fewer digits necessarily include more than one guard digit (see Section 6.12). None of the layouts for 12 digits can include a guard digit, nor is one needed; the first digit of such a number can be 0, 1 or 2 only. A few other styles have no guard digit (these are marked 'No. x') as more than four groups would have been required to provide one.

**A.6** The details of each of the layouts given are specified in the layout-symbol (see Section 3.5). In these symbols:-

- / stands for *Space*,
- x stands for a digit which is liable to be *omitted*,
- 3 stands for a digit which is liable to be *replaced by Space*,
- 9 stands for a digit which is *always printed*.

**A.7** The column headed *width* gives the width normally obtained from each layout, this includes the sign but not the guard digit(s). This width may require adjustment as follows:-

- (a) If  $S \neq 7$ , the width should be increased by  $7 - S$  to allow for the *initial spaces* (Section A.2),
- (b) If  $N$  is changed to *omit* the sign, the width should be reduced by 1 (Section A.3),
- (c) If there are  $F$  *final spaces*, the width should be increased by  $F$  (Section A.4).



## Layout Parameters

Number of Digits			Layout Symbol	Width (incl. sign)	Layout-Parameter	
Total	Before Decimal Point	After Decimal Point			a-order	b-order
1	1	0	xxxx9	2	1.0 7S54	0.0 0000
	0	1	xxxx.9	3	1.6 7S44	1.3 0000
2	2	0	xxx39	3	1.0 7S37	2.0 0000
	1	1	xxx9.9	4	1.2 7S44	1.3 0000
	0	2	xxx.99	4	1.6 7S34	2.3 0000
3	3	0	xx339	4	1.0 7S27	3.0 0000
	2	1	xx39.9	5	1.2 7S27	2.0 1300
	1	2	xx9.99	5	1.2 7S34	2.3 0000
	0	3	xx.999	5	1.6 7S24	3.3 0000
4	4	0	x3339	5	1.0 7S17	4.0 0000
	3	1	x339.9	6	1.2 7S17	3.0 1300
	2	2	x39.99	6	1.2 7S17	2.0 2300
	1	3	x9.999	6	1.2 7S24	3.3 0000
	0	4	x.9999	6	1.6 7S14	4.3 0000
5	5	0	x33339	6	1.0 6S17	5.0 0000
	4	1	x3339.9	7	1.2 6S17	4.0 1300
	3	2	x339.99	7	1.2 6S17	3.0 2300
	2	3	x39.999	7	1.2 6S17	2.0 3300
	1	4	x9.9999	7	1.2 6S24	4.3 0000
	0	5	x.99999	7	1.6 6S14	5.3 0000
6	6	0	x333339	7	1.0 5S17	6.0 0000
	3,3	0	x333/339	8	1.0 5S17	3.2 3000
	1,5	0	x3/33339	8	1.0 5S17	1.2 5000
	5	1	x33339.9	8	1.2 5S17	5.0 1300
	4	2	x3339.99	8	1.2 5S17	4.0 2300
	3	3	x339.999	8	1.2 5S17	3.0 3300
	2	4	x39.9999	8	1.2 5S17	2.0 4300
	1	5	x9.99999	8	1.2 5S24	5.3 0000
	0	6	x.999999	8	1.6 5S14	6.3 0000
	0	3,3	x.999/999	9	1.6 5S14	3.2 3300
0	5,1	x.99999/9	9	1.6 5S14	5.2 1300	
7	7	0	x3333339	8	1.0 4S17	7.0 0000
	4,3	0	x3333/339	9	1.0 4S17	4.2 3000
	2,5	0	x33/33339	9	1.0 4S17	2.2 5000
	6	1	x333339.9	9	1.2 4S17	6.0 1300
	3,3	1	x333/339.9	10	1.2 4S17	3.2 3013
	1,5	1	x3/33339.9	10	1.2 4S17	1.2 5013
	5	2	x33339.99	9	1.2 4S17	5.0 2300
	4	3	x3339.999	9	1.2 4S17	4.0 3300
	3	4	x339.9999	9	1.2 4S17	3.0 4300
	2	5	x39.99999	9	1.2 4S17	2.0 5300
	1	6	x9.999999	9	1.2 4S24	6.3 0000
	1	3,3	x9.999/999	10	1.2 4S24	3.2 3300
	1	5,1	x9.99999/9	10	1.2 4S24	5.2 1300
	0	7	x.9999999	9	1.6 4S14	7.3 0000
	0	3,4	x.999/9999	10	1.6 4S14	3.2 4300
	0	4,3	x.9999/999	10	1.6 4S14	4.2 3300
0	5,2	x.99999/99	10	1.6 4S14	5.2 2300	

Layout Parameters (Contd)

Number of Digits			Layout Symbol	Width (incl. sign)	Layout-Parameter	
Total	Before Decimal Point	After Decimal Point			a-order	b-order
8	8	0	x33333339	9	1.0 3S17	3.3 5000
	3,5	0	x333/33339	10	1.0 3S17	3.2 5000
	4,4	0	x3333/3339	10	1.0 3S17	4.2 4000
	2,3,3	0	x33/333/339	11	1.0 3S17	2.2 3230
	7	1	x3333339.9	10	1.2 3S17	7.0 1300
	4,3	1	x3333/339.9	11	1.2 3S17	4.2 3013
	2,5	1	x33/33339.9	11	1.2 3S17	2.2 5013
	6	2	x333339.99	10	1.2 3S17	6.0 2300
	3,3	2	x333/339.99	11	1.2 3S17	3.2 3023
	1,5	2	x3/33339.99	11	1.2 3S17	1.2 5023
	5	3	x33339.999	10	1.2 3S17	5.0 3300
	4	4	x3339.9999	10	1.2 3S17	4.0 4300
	3	5	x339.99999	10	1.2 3S17	3.0 5300
	2	6	x39.999999	10	1.2 3S17	2.0 6300
	2	3,3	x39.999/999	11	1.2 3S17	2.0 3233
	2	5,1	x39.99999/9	11	1.2 3S17	2.0 5213
	1	7	x9.9999999	10	1.2 3S24	7.3 0000
	1	3,4	x9.999/9999	11	1.2 3S24	3.2 4300
	1	4,3	x9.9999/999	11	1.2 3S24	4.2 3300
	1	5,2	x9.99999/99	11	1.2 3S24	5.2 2300
0	8	x.99999999	10	1.6 3S14	4.3 4300	
0	3,3,2	x.999/999/99	12	1.6 3S14	3.2 3223	
0	3,5	x.999/99999	11	1.6 3S14	3.2 5300	
0	4,4	x.9999/9999	11	1.6 3S14	4.2 4300	
0	5,3	x.99999/999	11	1.6 3S14	5.2 3300	
9	9	0	x333333339	10	1.0 2S17	4.3 5000
	4,5	0	x3333/33339	11	1.0 2S17	4.2 5000
	3,3,3	0	x333/333/339	12	1.0 2S17	3.2 3230
	8	1	x33333339.9	11	1.2 2S17	3.3 5013
	3,5	1	x333/33339.9	12	1.2 2S17	3.2 5013
	4,4	1	x3333/3339.9	12	1.2 2S17	4.2 4013
	7	2	x3333339.99	11	1.2 2S17	7.0 2300
	4,3	2	x3333/339.99	12	1.2 2S17	4.2 3023
	2,5	2	x33/33339.99	12	1.2 2S17	2.2 5023
	6	3	x333339.999	11	1.2 2S17	6.0 3300
	3,3	3	x333/339.999	12	1.2 2S17	3.2 3033
	1,5	3	x3/33339.999	12	1.2 2S17	1.2 5033
	5	4	x33339.9999	11	1.2 2S17	5.0 4300
	4	5	x3339.99999	11	1.2 2S17	4.0 5300
	3	6	x339.999999	11	1.2 2S17	3.0 6300
	3	3,3	x339.999/999	12	1.2 2S17	3.0 3233
	3	5,1	x339.99999/9	12	1.2 2S17	3.0 5213
	2	7	x39.9999999	11	1.2 2S17	2.0 7300
	2	3,4	x39.999/9999	12	1.2 2S17	2.0 3243
	2	4,3	x39.9999/999	12	1.2 2S17	2.0 4233
2	5,2	x39.99999/99	12	1.2 2S17	2.0 5223	
1	8	x9.99999999	11	1.2 2S24	4.3 4300	
1	3,3,2	x9.999/999/99	13	1.2 2S24	3.2 3223	
1	3,5	x9.999/99999	12	1.2 2S24	3.2 5300	
1	4,4	x9.9999/9999	12	1.2 2S24	4.2 4300	
1	5,3	x9.99999/999	12	1.2 2S24	5.2 3300	
0	9	x.999999999	11	1.6 2S14	5.3 4300	
0	4,5	x.9999/99999	12	1.6 2S14	4.2 5300	
0	5,4	x.99999/9999	12	1.6 2S14	5.2 4300	
0	6,3	x.999999/999	12	1.6 2S14	6.2 3300	
0	3,3,3	x.999/999/999	13	1.6 2S14	3.2 3233	

F = 0

F = 0

Layout Parameters (Contd)

Number of Digits			Layout Symbol	Width (incl. sign)	Layout-Parameter	
Total	Before Decimal Point	After Decimal Point			a-order	b-order
10	10	0	x3333333339	11	1.0 1S17	5.3 5000
	5,5	0	x33333/33339	12	1.0 1S17	5.2 5000
	4,6	0	x3333/333339	12	1.0 1S17	4.2 6000
	4,3,3	0	x3333/333/339	13	1.0 1S17	4.2 3230
	9	1	x333333339.9	12	1.2 1S17	4.3 5013
	4,5	1	x3333/33339.9	13	1.2 1S17	4.2 5013
	3,3,3	1	333/333/339.9	14	1.2 2S32	3.2 3013
	3,6	1	x333/333339.9	13	1.2 1S17	3.2 6013
	8	2	x33333339.99	12	1.2 1S17	4.3 4023
	3,5	2	x333/33339.99	13	1.2 1S17	3.2 5023
	4,4	2	x3333/3339.99	13	1.2 1S17	4.2 4023
	7	3	x3333339.999	12	1.2 1S17	7.0 3300
	4,3	3	x3333/339.999	13	1.2 1S17	4.2 3033
	2,5	3	x33/33339.999	13	1.2 1S17	2.2 5033
	6	4	x333339.9999	12	1.2 1S17	6.0 4300
	3,3	4	x333/339.9999	13	1.2 1S17	3.2 3043
	1,5	4	x3/33339.9999	13	1.2 1S17	1.2 5043
	5	5	x33339.99999	12	1.2 1S17	5.0 5300
	4	6	x3339.999999	12	1.2 1S17	4.0 6300
	4	3,3	x3339.999/999	13	1.2 1S17	4.0 3233
	4	5,1	x3339.99999/9	13	1.2 1S17	4.0 5213
	3	7	x339.9999999	12	1.2 1S17	3.0 7300
	3	3,4	x339.999/9999	13	1.2 1S17	3.0 3243
	3	4,3	x339.9999/999	13	1.2 1S17	3.0 4233
	3	5,2	x339.99999/99	13	1.2 1S17	3.0 5223
	2	8	x39.99999999	12	1.2 1S17	2.0 4343
	2	3,5	x39.999/99999	13	1.2 1S17	2.0 3253
	2	4,4	x39.9999/9999	13	1.2 1S17	2.0 4243
	2	5,3	x39.99999/999	13	1.2 1S17	2.0 5233
	1	9	x9.999999999	12	1.2 1S24	5.3 4300
	1	4,5	x9.9999/99999	13	1.2 1S24	4.2 5300
	1	5,4	x9.99999/9999	13	1.2 1S24	5.2 4300
	1	6,3	x9.999999/999	13	1.2 1S24	6.2 3300
	1	3,3,3	x9.999/999/999	14	1.2 1S24	3.2 3233
	0	10	x.9999999999	12	1.6 1S14	5.3 5300
	0	5,5	x.99999/99999	13	1.6 1S14	5.2 5300
	0	4,6	x.9999/999999	13	1.6 1S14	4.2 6300
	0	6,4	x.999999/9999	13	1.6 1S14	6.2 4300
	0	3,3,4	x.999/999/9999	14	1.6 1S14	3.2 3243
	11	11	0	x33333333339	12	1.0 0S17
5,6		0	x33333/333339	13	1.0 0S17	5.2 6000
1,5,5		0	x3/33333/33339	14	1.0 0S17	1.2 5250
10		1	x3333333339.9	13	1.2 0S17	5.3 5013
5,5		1	x33333/33339.9	14	1.2 0S17	5.2 5013
4,6		1	x3333/333339.9	14	1.2 0S17	4.2 6013
4,3,3		1	3333/333/339.9	15	1.2 1S42	3.2 3013
9		2	x333333339.99	13	1.2 0S17	4.3 5023
4,5		2	x3333/33339.99	14	1.2 0S17	4.2 5023
3,3,3		2	333/333/339.99	15	1.2 1S32	3.2 3023
3,6		2	x333/333339.99	14	1.2 0S17	3.2 6023
8		3	x33333339.999	13	1.2 0S17	4.3 4033
3,5		3	x333/33339.999	14	1.2 0S17	3.2 5033
4,4		3	x3333/3339.999	14	1.2 0S17	4.2 4033
7		4	x3333339.9999	13	1.2 0S17	7.0 4300
4,3		4	x3333/339.9999	14	1.2 0S17	4.2 3043
2,5		4	x33/33339.9999	14	1.2 0S17	2.2 5043

F = 0

No x

F = 0

No x

No x

con-  
tinues

Layout Parameters (Contd)

Number of Digits			Layout Symbol	Width (incl. sign)	Layout-Parameter	
Total	Before Decimal Point	After Decimal Point			a-order	b-order
11 con- tinued	6	5	x333339.99999	13	1.2 0S17	6.0 5300
	3,3	5	x333/339.99999	14	1.2 0S17	3.2 3053
	1,5	5	x3/33339.99999	14	1.2 0S17	1.2 5053
	5	6	x33339.999999	13	1.2 0S17	5.0 6300
	5	3,3	x33339.999/999	14	1.2 0S17	5.0 3233
	5	5,1	x33339.99999/9	14	1.2 0S17	5.0 5213
	4	7	x3339.9999999	13	1.2 0S17	4.0 7300
	4	3,4	x3339.999/9999	14	1.2 0S17	4.0 3243
	4	4,3	x3339.9999/999	14	1.2 0S17	4.0 4233
	4	5,2	x3339.99999/99	14	1.2 0S17	4.0 5223
	3	8	x339.99999999	13	1.2 0S17	3.0 4343
	3	3,5	x339.999/99999	14	1.2 0S17	3.0 3253
	3	4,4	x339.9999/9999	14	1.2 0S17	3.0 4243
	3	5,3	x339.99999/999	14	1.2 0S17	3.0 5233
	2	9	x39.999999999	13	1.2 0S17	2.0 5343
	2	4,5	x39.9999/99999	14	1.2 0S17	2.0 4253
	2	5,4	x39.99999/9999	14	1.2 0S17	2.0 5243
	2	6,3	x39.999999/999	14	1.2 0S17	2.0 6233
	2	3,3,3	39.999/999/999	15	1.2 1S20	3.2 3233
	1	10	x9.9999999999	13	1.2 0S24	5.3 5300
1	5,5	x9.99999/99999	14	1.2 0S24	5.2 5300	
1	4,6	x9.9999/999999	14	1.2 0S24	4.2 6300	
1	6,4	x9.999999/9999	14	1.2 0S24	6.2 4300	
1	3,3,4	x9.999/999/9999	15	1.2 0S24	3.2 3243	
0	11	x.99999999999	13	1.6 0S14	6.3 5300	
0	5,6	x.99999/999999	14	1.6 0S14	5.2 6300	
0	6,5	x.999999/99999	14	1.6 0S14	6.2 5300	
0	5,5,1	x.99999/99999/9	15	1.6 0S14	5.2 5213	
12	12	0	333333333339	13	1.0 0S63	6.0 0000
	6,6	0	333333/333339	14	1.0 0S62	6.0 0000
	2,5,5	0	33/33333/33339	15	1.0 0S22	5.2 5000
	11	1	33333333339.9	14	1.2 0S63	5.0 1300
	5,6	1	33333/333339.9	15	1.2 0S52	6.0 1300
	1,5,5	1	3/33333/33339.9	16	1.2 0S12	5.2 5013
	10	2	3333333339.99	14	1.2 0S53	5.0 2300
	5,5	2	33333/33339.99	15	1.2 0S52	5.0 2300
	4,6	2	3333/333339.99	15	1.2 0S42	6.0 2300
	4,3,3	2	3333/333/339.99	16	1.2 0S42	3.2 3023
	9	3	333333339.999	14	1.2 0S43	5.0 3300
	4,5	3	3333/33339.999	15	1.2 0S42	5.0 3300
	3,3,3	3	333/333/339.999	16	1.2 0S32	3.2 3033
	3,6	3	333/333339.999	15	1.2 0S32	6.0 3300
	8	4	33333339.9999	14	1.2 0S33	5.0 4300
	3,5	4	333/33339.9999	15	1.2 0S32	5.0 4300
	4,4	4	3333/3339.9999	15	1.2 0S42	4.0 4300
	7	5	3333339.99999	14	1.2 0S70	5.3 0000
	4,3	5	3333/339.99999	15	1.2 0S42	3.0 5300
	2,5	5	33/33339.99999	15	1.2 0S22	5.0 5300
	6	6	333339.999999	14	1.2 0S60	6.3 0000
	3,3	6	333/339.999999	15	1.2 0S32	3.0 6300
	1,5	6	3/33339.999999	15	1.2 0S12	5.0 6300
	3,3	3,3	333/339.999/999	16	1.2 0S32	3.0 3233
	1,5	5,1	3/33339.99999/9	16	1.2 0S12	5.0 5213
5	7	33339.9999999	14	1.2 0S50	7.3 0000	
5	3,4	33339.999/9999	15	1.2 0S50	3.2 4300	
con- tinues	5	4,3	33339.9999/999	15	1.2 0S50	4.2 3300

No x

No x  
for any  
12-digit  
Layout

Layout Parameters (Contd)

Number of Digits			Layout Symbol	Width (incl. sign)	Layout-Parameter	
Total	Before Decimal Point	After Decimal Point			a-order	b-order
12 con- tinued	5	5,2	33339.99999/99	15	1.2 0S50	5.2 2300
	4	8	3339.99999999	14	1.2 0S40	3.3 5300
	4	3,5	3339.999/99999	15	1.2 0S40	3.2 5300
	4	4,4	3339.9999/9999	15	1.2 0S40	4.2 4300
	4	5,3	3339.99999/999	15	1.2 0S40	5.2 3300
	3	9	339.999999999	14	1.2 0S30	4.3 5300
	3	4,5	339.9999/99999	15	1.2 0S30	4.2 5300
	3	5,4	339.99999/9999	15	1.2 0S30	5.2 4300
	3	6,3	339.999999/999	15	1.2 0S30	6.2 3300
	3	3,3,3	339.999/999/999	16	1.2 0S30	3.2 3233
	2	10	39.9999999999	14	1.2 0S20	5.3 5300
	2	5,5	39.99999/99999	15	1.2 0S20	5.2 5300
	2	4,6	39.9999/999999	15	1.2 0S20	4.2 6300
	2	6,4	39.999999/9999	15	1.2 0S20	6.2 4300
	2	3,3,4	39.999/999/9999	16	1.2 0S20	3.2 3243
	1	11	9.99999999999	14	1.2 0S10	5.3 6300
1	5,6	9.99999/999999	15	1.2 0S10	5.2 6300	
1	6,5	9.999999/99999	15	1.2 0S10	6.2 5300	
1	5,5,1	9.99999/99999/9	16	1.2 0S10	5.2 5213	

## APPENDIX B

## Right-Hand Zero Suppression

**B.1** If desired *right-hand zeros* may be suppressed (i.e. replaced by spaces) in the fractional part of the number. This takes two forms:-

either (a) the decimal-point is printed and is followed by the first digit of the fractional part; the remaining digits of the fractional part will be liable to suppression,

or (b) right-hand zero suppression is initiated at the decimal point; if at this stage the remainder of the number to be printed is zero then the decimal-point is itself replaced by a space.

The above will be referred to as RHZ(a) and RHZ(b) respectively. These two effects may be obtained by changing the second octal digit of the  $N$ -address in the  $a$ -order of the layout-parameter. In the catalogue this is always 2 or 6 (it is 0 in integers but RHZ is here inapplicable), i.e.  $N$  is 1.2 or 1.6. We can obtain RHZ by replacing  $N$  as shown in the following table:-

$a$ -order $N$ -address as written in catalogue	Replace by	
	For RHZ(a)	For RHZ(b)
1.2	1.1	1.3
1.6	1.5	1.7*

\*Note:  $N = 1.7$  is not recommended for any layout (zero will not be printed at all).

**B.2** It is only the second octal digit of  $N$  which is to be changed in this way. The changes described in Section A.3 above for dealing with the sign may be combined with these changes; for example if a layout has  $N = 1.2$  in the catalogue and we wish to change it to give an immediate sign and RHZ(a) then we must write  $N = 0.1+$ . The complete range is shown in the following table, the first column of which gives  $N$  as shown in the catalogue of Appendix A:-

Delayed Sign			Immediate Sign			To omit Sign		
No RHZ	RHZ(a)	RHZ(b)	No RHZ	RHZ(a)	RHZ(b)	No RHZ	RHZ(a)	RHZ(b)
1.0	NA	NA	0.0+	NA	NA	0.0	NA	NA
1.2	1.1	1.3	0.2+	0.1+	0.3+	0.2	0.1	0.3
1.6	1.5	NR	1.6	1.5	NR	0.6	0.5	NR

NA means 'not applicable'.

NR means 'not recommended'.

Unless the sign is omitted these changes do not affect the width of the printing as given in the catalogue.

**B.3** For other types of layout reference should be made to Appendices C,D and E.

APPENDIX C

Detailed Description of the Layout-Parameter

C.1 A general layout-parameter may be written:-

$L.D(+)$	$X$	$S\phi_0$	$q_0(\cdot)$
$\phi_1 q_1$	$\phi_2$	$q_2 \phi_3$	$q_3$

Here the quantity  $N$  has been explicitly shown as  $L.D+$ . (Section 6.2).

- (a) If this is a 'stop' order-pair  $CR LF$  will be printed (see 6.1).
- (b) The number of *Initial Stages* is  $7-S$  (see 6.3).
- (c) If there is a  $+$  after  $L.D$  there is an *immediate sign*.
- (d) The number to be printed ( $Z$ ) is held as an integer (12 decimal digits); this is multiplied by  $10^X$  before printing cycle. This effectively removes the first  $X$  of the 12 digits from consideration (these digits must all be zero) (Section 6.4). The remaining  $m = 12-X$  digits are divided into 4 groups. Extra zeros on right can be supplied by increasing  $m$ .

C.2 We shall write LHZ for *suppression or omission of left-hand zeros* and RHZ for *suppression or omission of right-hand zeros*. LHZ may be initiated only at the start of the number (if  $L = 0$  or 1); it is terminated either (a) when a non-zero digit is encountered, or (b) the decimal-point group is met. RHZ may be initiated only at the decimal-point (if  $D$  is odd); it takes effect only when the remainder of the number being printed is zero. The octal digit  $L$  is examined before the number is printed, its effects are as follows:-

$L$	Action
0	Initiate LHZ.
1	Initiate LHZ and arrange to print a delayed sign.
2	Print all digits, even if zero.
3	Suppress or omit all digits, even if <i>not</i> zero, until stopped by the decimal-point group.

C.3 The groups of digits ( $\phi_i$  digits in the  $i$ -th group) are treated sequentially, unwanted digits (those liable to suppression or omission) are dealt with as directed by the group-indicators  $q_i$ . The  $q_i$  also determine the action to be taken at the end of the group.

Exceptionally, if a null-group is encountered (one having  $\phi_i = 0$ ) the action is to print  $q_i$  spaces and obey the link.

The following table shows the effects of various values of  $q$  in a group having  $\phi \neq 0$ .

Unwanted Digits		Action at End of Group
Suppress	Omit	
0	4	Decimal-point operations as determined by $D$ , then take next group.
1	5	Print two Spaces and obey link.
2	6	Print Space and take next group.
3	7	Take next group immediately

In all cases where there is no next group when one is called for the link is obeyed immediately. Note that  $q = 0$  (or 4) should appear once only among the  $q$ 's.

**C.4** The octal digit  $D$  is the *decimal point indicator*; it determines the action to be taken on reaching the end of the group with  $q = 0$  or 4. This group normally terminates the integral part of the number (but see Appendix E). There is a facility here for printing the last digit of the group before the point even if it would otherwise be suppressed or omitted; this is done by arranging to treat the last digit of the group as a non-zero digit. One may also start RHZ here (e.g. printing 10.5 instead of 10.500); this may take two alternative forms: (a) the decimal-point is printed and followed by at least one digit of the next group, or (b) the decimal-point itself is suppressed if all remaining digits of the number are zero. The following table summarizes the effects of various values of  $D$ :-

Treat Last Digit of Group		Other operations (in all cases stop LHZ)
as a non-zero digit	as a normal digit	
0	4	No other action (no RHZ).
1	5	Print decimal-point and first digit of next group; then start RHZ.
2	6	Print decimal-point (no RHZ).
3	7	Start immediate RHZ. If remainder of number is zero, print Space; otherwise print decimal point.

Note: The value  $D = 7$  is of doubtful utility, it may result in the number zero being printed simply as a few spaces.



APPENDIX D

Extra Zeros

D.1 If desired one can arrange to print extra zeros on the right of the number. This is done by making R1 consider more than 12 decimal digits; all digits after the twelfth will be zero. This technique may be useful with integers; one can, for example, arrange to print 5 as 5000 by considering 15 digits.

D.2 As an example suppose we wish to print  $Z = 1000 (z.2^{38})$ . We will suppose that  $z.2^{38}$  is an integer with at most three non-zero digits, i.e.  $m = 3$ . We must make  $m$  up to 5 by including two guard digits and we can then set  $X = 12-5 = 7$  (section 6.4); the quantity  $m$  does not include the extra zeros. This value of  $X$  ensures that the first 7 decimal digits are not printed; the remaining 5 digits of the 12 are now augmented to 8 to include the extra zeros. The parameters are therefore as follows (assuming two initial spaces):-

1.0 7 52 7
6.0 0 00 0
-1.0

D.3 It should be noted that the scaling-parameter is in all cases to be determined as if the extra zeros were not present.

## APPENDIX E

## Complete Suppression of Left-Hand Digits

**E.1** Appendix C (Section C.2) shows that if  $L = 3$  in the layout parameter then all digits up to the decimal-point group will be suppressed or omitted, even if they are *not* zero. This facility is useful when one wishes to print only the last few digits of an integer. This requirement may arise, for example, in printing tabular values of a slowly-changing function, the complete functional value being printed only occasionally; one may get a table having the following appearance:-

```

37474 478 483 487 492 497 502 508 513 519
37525 531 537 543 550 557 565 573 582 591
37599 608 617 626 636 646 657 667 677 688

```

**E.2** In this example we have a five-digit integer and we require a style of printing in which only the last 3 digits are printed. If we suppose that the integer is unsigned and that we want two initial spaces, the layout-parameter will be as follows:-

3.4	7	52	4
3.3	0	00	0

Here we have  $p_o = 2$ ,  $q_o = 4$  indicating a two-digit group with unwanted digits omitted (see Section C.3) which is followed by the 'decimal-point' operations determined by  $D = 4$ . The setting  $L = 3$  will cause all digits of this group to be suppressed or omitted (in this case omitted since  $q \geq 4$ ). The only effect of  $D = 4$  is to stop the omission of digits (see C.4); in this case it does not of course indicate the end of the integral part of the number. If we set  $q_o = 0$  (instead of 4) then the two left-hand digits are replaced by spaces instead of being omitted.

APPENDIX F

The Parameters for The Standard Styles

The following is the optional parameter-list included in the tape of R1.

1.2 0 71 0.	$\pi 9.99999/999999//$	} Style 1
5.2 6 10 0		
-100000000000	$= -10^{11} \times 2^{-38}$	} Style 2
1.2 0 71 0	$9.99999/999999//$	
5.2 6 10 0		} Style 3
-100000000000	$= -10^{11} \times 2^{-38}$	
1.2 5 71 0	$9.99999//$	} Style 4
6.1 0 00 0		
-1000000	$= -10^6 \times 2^{-38}$	} Style 5
1.3 0 76 4	$xxxxx9.xxxxxx//$	
6.5 0 00 0		} Style 6
-1.0		
1.0 0 77 7.	$\pi xxxxxxxxxxxx9//$	} Style 7
5.4 0 20 0		
-1.0		} Style 8
1.0 0 76 2	$333333/333339//$	
6.0 0 20 0		} Style 9
-1.0		
0.0 7 72 7	$+xx339//$	} Style 10
3.0 0 20 0		
-1.0		} Style 11
0.0 7 73 7	$+xxx39//$	
2.0 0 20 0		} Style 12
-1.0		

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
27.8.56.

## PAGE LAYOUT SUBROUTINE

A self-preserving subroutine which uses R1 to print a number from X1. Before calling in R1 to print the number, R3 will print any CR and LF characters needed so that the printed page will be laid out in rows, columns and blocks as required.

**Name:** BLOCK LAYOUT

**Store:** 4 blocks + R1, i.e. a total of 15 blocks + parameter-list for R1.

**Uses:** U0; X1, 6, 7; B0 (self-preserving in U0).

**Cues:** 01 a-order 0+ cue } see Section 2.1 below.  
02 b-order 0+ cue }

**Link:** Computing-store link set in X6.

## 1. General Description

**1.1** When called in R3 prints a single number (using R1); it prints CR LF before the first number of a new line, and an extra LF before the first number of a new block. The resulting page will have  $L$  numbers on a line and  $B$  lines in a block, or a total of  $N$  numbers in a block.  $L$ ,  $B$  and  $N$  are preset-parameters which can very easily be altered by a programme (see Section 4 overleaf).

**1.2** Normally only one of  $B$  or  $N$  will be required; whichever is not needed should be set equal to zero.

**1.3** Since the numbers concerned are actually printed by R1 any of the styles appropriate to R1 may be used with R3; for details of these styles reference should be made to the specification of R1.

## 2. Methods of Entry

**2.1** Before using R3 its block 0+ must be transferred to U0; this may be done at any convenient time by the order 0  $\boxed{0}$  72, which must be tagged so that one of the cues of R3 is added to it during input. If the order 0  $\boxed{0}$  72 is an a-order, then the order-pair containing it should be tagged by a call for cue 01 to R3; if it is a b-order, cue 02 should be called for by the tag. It will be assumed in the following description that block 0+ of R3 is in U0.

**2.2** The *normal method* of entry is best shown by an example. Suppose we wish to print the number in B43.5 in style 3 of R1 and return afterwards to the a-order in U2.6. The following orders are required:-

43 5 70	Number to X1
2.6 6 40	Set link in X6
3 7 40	Set Style in X7
0.0 0 60	Enter R3

**2.3** In order to arrange that the number starts a *fresh block* it is necessary only to set the *link* with a 42 order instead of a 40 order. For example, to print the number in U5.7 in style 5 of R1 and to start a new block, afterwards returning to the b-order in 3.0, the orders are:-

5.7 1 00	Number to X1
3.0+ 6 42	Link to X6 (new block)
5 7 40	Style 5
0.0 0 60	Enter R3

The first entry to R3 in any programme should start a fresh block (see also Section 6).

**2.4** In order to start a *new line* (and, if appropriate, a fresh block), one may set the *style* with a 42 order instead of a 40 order. For example, to start a new line by printing the number X6 in style 4 of R1, afterwards returning to the a-order in 1.4, the orders are:-

6 1 00	Number to X1
1.4 6 40	Link to X6
4 7 42	Style 4 (new line)
0.0 0 60	Enter R3

**2.5** As alternatives to the methods of entry described in sections 2.3 and 2.4 above, one may use a sequence of orders similar to that of section 2.2 but jumping to 0.4 (for a fresh block) or to 0.1 (for a fresh line). The methods described above are, however, generally more useful.

### 3. Conditions on Exit

**3.1** When the link is obeyed the whole Computing Store will have been returned to its state on entry with the exception of accumulators 1, 6 and 7 which are treated as follows (*S* denotes the style and *A(+)* the link-address:-

Accumulator	Content on Entry	Content on Exit
1	Number for printing	<i>m</i> (marker - see below)
6	$\pm A(+)$	$+A(+)$
7	$\pm S$	$+S$

**3.2** It should be noted that, if the style and the link are not to be changed, X6 and X7 are already set for printing the next number. The re-entry sequence need therefore only set the next number in X1 and jump to 0.0. This is true even if the link or the style was set by a 42 order to start a fresh line or block.

**3.3** On exit from R3 the number in X1 will have been replaced by a *marker m* which gives an indication of the state of the row and block counts. Normally (i.e. in the middle of a line)  $m = 0$ . At the end of a line or block  $m$  will have the teleprinter code for Line Feed at its least-significant end; consequently an additional Line Feed can be printed at this stage by the master-programme with a single order 16 1 10. The sign of  $m$  is positive at the end of a line and negative at the end of a block. Explicitly  $m$  has the following values:-

Normally (i.e. in the middle of a line),	$m = 0,$
At the end of a line,	$m = 13.2^{-38},$
At the end of a block,	$m = -1 + 13.2^{-38}.$

**3.4** When printing numbers systematically in a block layout, it is usually necessary to perform some special counting operations at the end of a line or block. These cases can easily be distinguished in the master-programme by testing  $m$  in X1 on exit from R3. There is often no need to keep a separate count in the master-programme, or even to use a special link. These points are illustrated in the examples given in the Appendix.

**3.5** It should be noted that the CR and LF operations needed to lay out the page are performed by R3 immediately before printing the first number in a line or a block; they will not have been done when the link is obeyed at the end of a line or block.

**3.6** Normally, none of the styles used with R3 should cause CR LF to be printed - i.e., the layout-parameters in the parameter-list for R1 should be 'go' order-pairs in those styles used with R3.

#### 4. Setting $L$ , $B$ and $N$

**4.1** The constants  $L$ ,  $B$  and  $N$  (see Section 1.1 above) are stored in the counter positions of 1+.5, 1+.6 and 1+.7 respectively. The remaining locations in this block are used to store the current values of the line and block counters and the value of the marker  $m$ ; the values of these quantities are irrelevant provided the first entry to R3 starts a new block.

**4.2** An optional parameter-list with R3 sets  $L = 4$ ,  $B = 5$ ,  $N = 0$ ; this provides a layout of 4 numbers to a line and 5 lines to a block. The normal way of setting these parameters to other values is by including a *parameter-list* with the master-programme. For example, to set  $L = 6$ ,  $B = 0$ ,  $N = 20$  the following parameter-list is required:-

R 0 0 -0 3
3 - 04 -
+6
+0
+20

These values of the parameters provide a layout with 6 numbers to a line and 20 numbers to a block; the last line of the block will contain only 2 numbers.

**4.3** One may alter the values of these parameters from tape after input of the programme. In this case we need to know where R3 is stored (the optional printing performed by Assembly gives the requisite information). Suppose, for example, that R3 starts in B64 and we wish to set  $L = 6$ ,  $B = 0$ ,  $N = 20$ . A tape punched as follows is needed:-

```
T65.5
+6
+0
+20
```

The transfer-address must be set equal to  $(b+1).5$ , where  $b$  is the block-address of R3.

**4.4** It is easy to cause the master-programme to set new values of the parameters. For example, to change all three parameters to the values  $L = 6$ ,  $B = 0$ ,  $N = 20$ :-

<table border="0" style="width: 100%; text-align: center;"> <tr><td style="border: 1px solid black; border-radius: 50%; width: 30px; height: 30px; line-height: 30px;">6</td><td style="padding: 0 10px;">5 40</td></tr> <tr><td style="border: 1px solid black; border-radius: 50%; width: 30px; height: 30px; line-height: 30px;">0</td><td style="padding: 0 10px;">6 40</td></tr> <tr><td style="border: 1px solid black; border-radius: 50%; width: 30px; height: 30px; line-height: 30px;">20</td><td style="padding: 0 10px;">7 40</td></tr> <tr><td style="padding: 0 5px;">1</td><td style="border: 1px solid black; padding: 0 5px;">7</td><td style="padding: 0 10px;">73</td></tr> </table>	6	5 40	0	6 40	20	7 40	1	7	73	<p>Set <math>L</math> in X5</p> <p>Set <math>B</math> in X6</p> <p>Set <math>N</math> in X7</p> <p>+ cue 01 or 02 to R3</p>
6	5 40									
0	6 40									
20	7 40									
1	7	73								

If this method is adopted it may be advantageous to make the number in X1 negative before the block-transfer order (73); see Section 6.

To change a single parameter, for example, to set  $N = 47$ :-

<table border="0" style="width: 100%; text-align: center;"> <tr><td style="border: 1px solid black; border-radius: 50%; width: 30px; height: 30px; line-height: 30px;">47</td><td style="padding: 0 10px;">1 40</td></tr> <tr><td style="border: 1px solid black; padding: 0 5px;">1</td><td style="border: 1px solid black; padding: 0 5px;">7</td><td style="padding: 0 10px;">71</td></tr> </table>	47	1 40	1	7	71	<p>Set <math>N</math> in X1</p> <p>+ cue 01 or 02 to R3</p>
47	1 40					
1	7	71				

**4.5** When selecting a value of  $L$  care should be taken that the styles used do not allow the total width of the printing in any line to exceed 69. The specification of R1 should be referred to.

## 5. Separate use of R1 and R3.

**5.1** It should be noted that if a master-programme calls for both R3 and R1 separately (at different points) then R1 is not a self-preserving routine if a computing-store link is used. In fact when R1 is called for under these circumstances it will leave the first block of R3 in U0; however the rest of the computing store will be untouched. If the R1 link is an order-pair then the first block of R1 will be left in U0.

## 6. Starting a New Block

**6.1** It is essential that the first number printed by R3 in a programme should start a new block; it is sometimes necessary also in the middle of a programme to arrange that a new block should be started. The way of doing this which has been explained above (Section 2.3) is sometimes inconvenient since one may wish to set R3 for a new block without calling in R3. This may be done by setting a negative number in  $B1+.1$  of R3, for example by means of the following two orders:-

32	1	00
----	---	----

Make X1 negative

1	1	71
---	---	----

+ cue 01 or 02 to R3

When this has been done R3 will start a new block when it is next called in. This method can usefully be combined with the setting of the parameters in the way described in Section 4.4 above.

## 7. Examples

**7.1** The Appendix provides illustrations of the way in which some of the features described above may be applied. The use of the marker *m* in X1 on exit from R3 (see Section 3.4 above) should be specially noted.

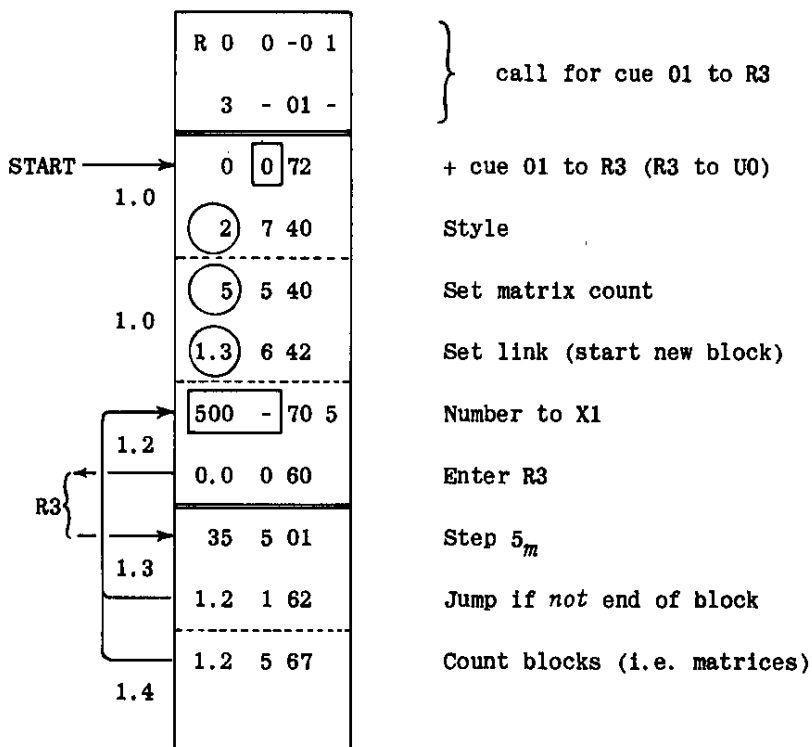


**APPENDIX**

This appendix is designed to illustrate some of the ways in which R3 may be used.

**Example 1.** To print five 4x4 matrices stored consecutively by rows starting at main-store location 500.

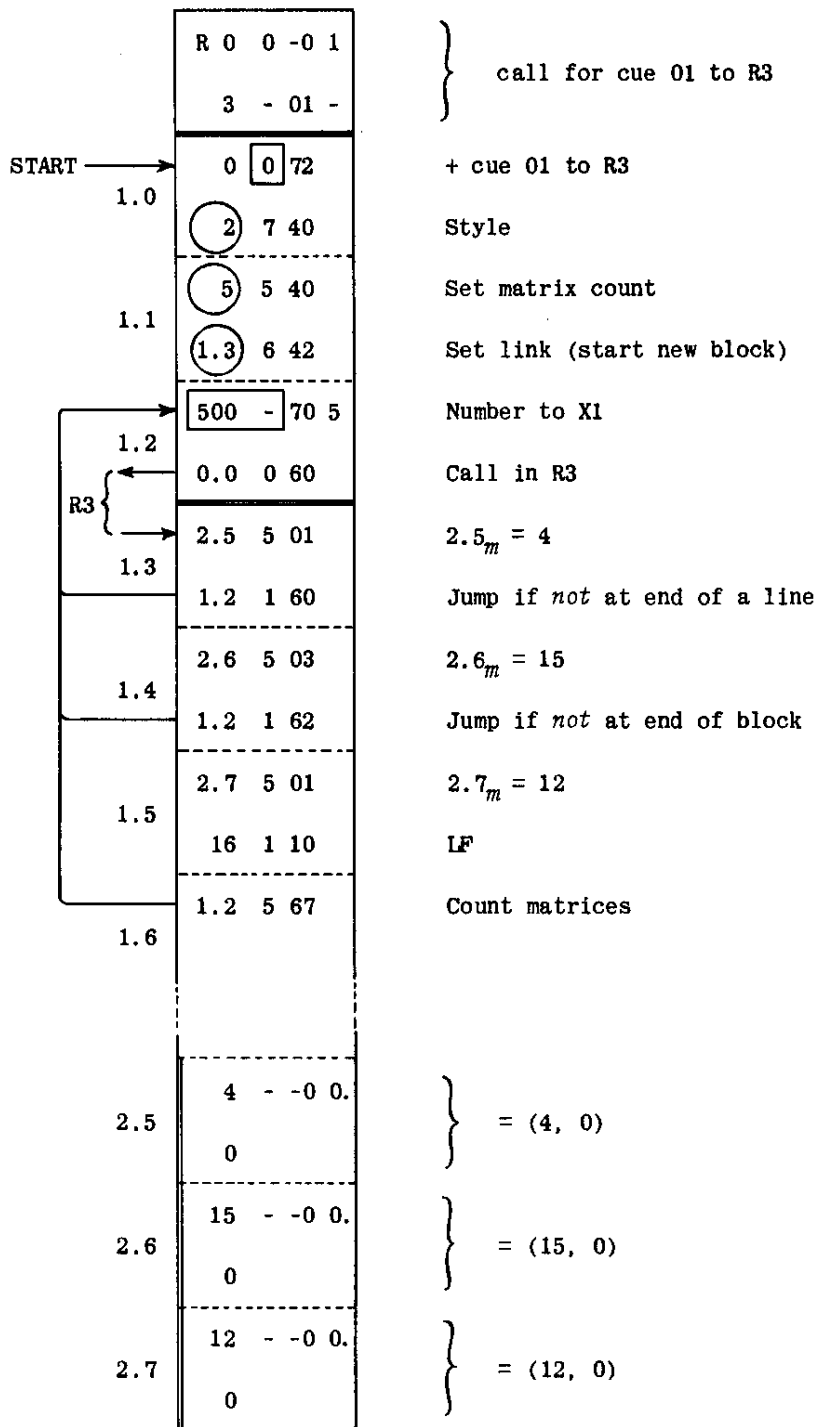
It is assumed that  $L = 4$ ,  $B = 4$ ,  $N = 0$ ; and that the printing of each number is to be done in style 2 (of R1).



Note that none of these orders may be in U0. The numbers will be printed in the following order:-

- 0 1 2 3
- 4 5 6 7
- 8 9 10 11
- 12 13 14 15
- 16 17 18 19
- 20 21 etc.

**Example 2.** To print the same five matrices stored by columns and to separate the matrices by two line-feeds. It is again assumed that  $L = 4, B = 4, N = 0$ .



The numbers will be printed in the following order:-

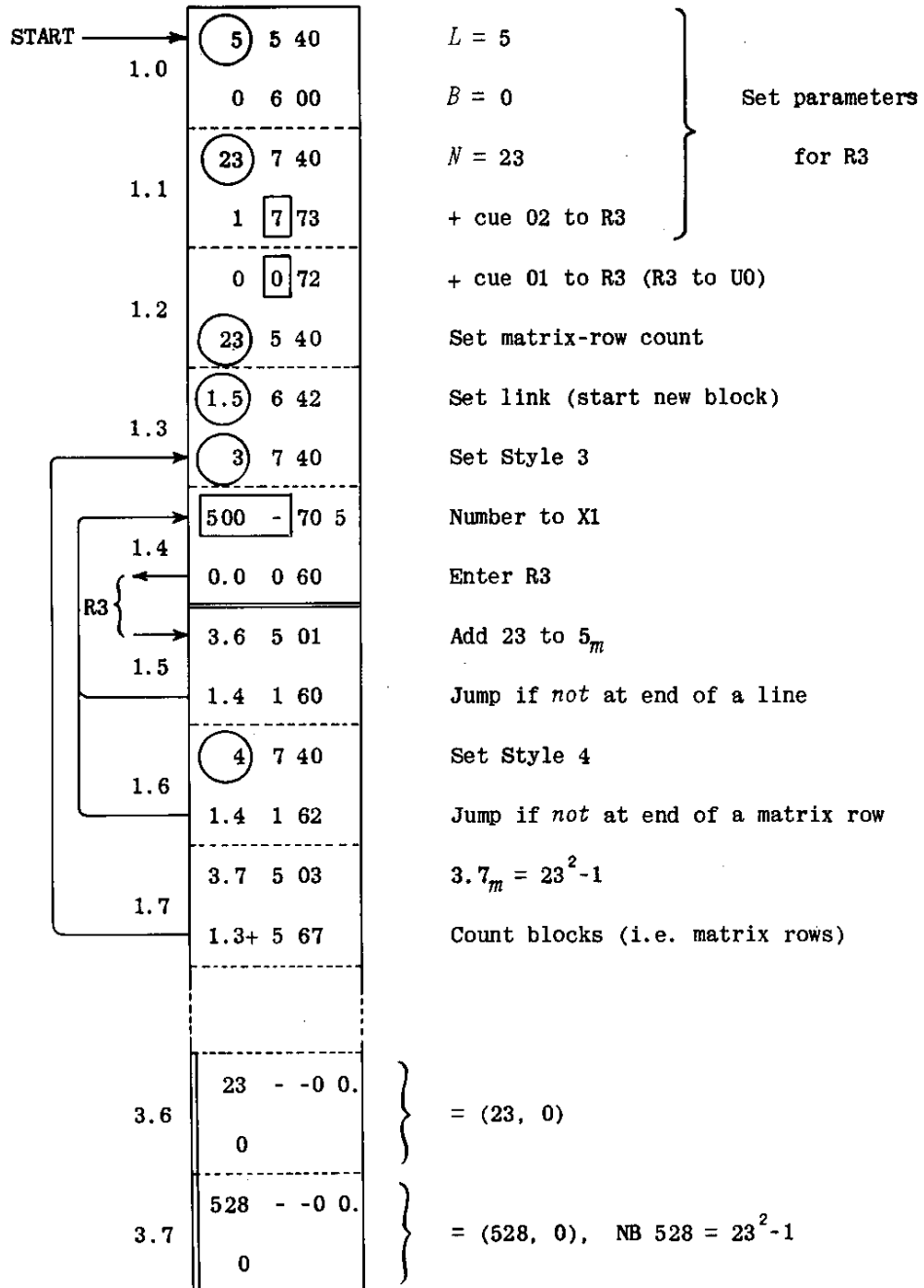
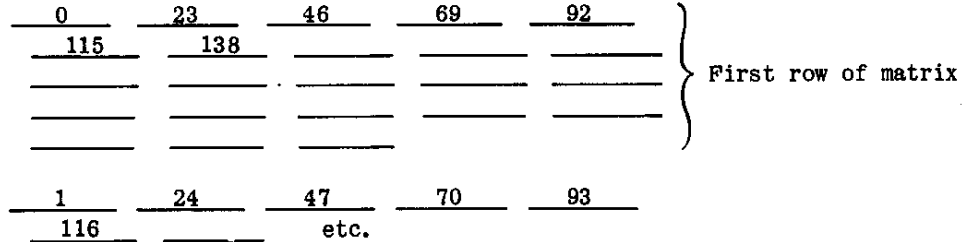
- 0 4 8 12
- 1 5 9 13
- 2 6 10 14
- 3 7 11 15

- 16 20 24 28
- 17 21 etc.

**Example 3.** To print the 23x23 matrix stored by columns starting at main-store location 500. It will be assumed that the parameter-list for R1 includes the following two styles:-

Style 3 - no initial spaces, two final spaces, } the two styles being  
 Style 4 - two initial spaces, no final spaces, } otherwise similar.

It is assumed also that the width of printing in either of these styles does not exceed 13 (including the spaces). The page layout desired is as follows:-



## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
29.11.56.

**NUMBER PRINT (SHORT)**

This is a short subroutine for printing out the contents of X7 as either an integer or a fraction. The output is in some cases suitable for re-input by the Initial Orders.

R 4 uses the Initial Orders number printing routine.

**Name:** I.O. NUMBER PUNCH  
**Store:** 4 blocks.  
**Uses:** U0; the accumulators are preserved in B0 and restored on exit. (For X1 see Section 4 below.)  
**Cues:** 01 to 09 for the various styles of printing.  
**Link:** Obeyed in 0.3 and left as obeyed in X1. See Section 4 below.

**1. Style of Printing**

- 1.1 The number to be printed must be set in X7.
- 1.2 All numbers are preceded either by CR LF or by two spaces, according to the manner of setting the link. (See Section 4.)
- 1.3 Numbers may be printed as integers or as fractions, and in either case they may be printed with or without their signs.
- 1.4 When an integer is printed, the sign (+ or -) is delayed so that it immediately precedes the first digit printed. When a fraction is printed, the sign is followed by 0 (or exceptionally 1). If the sign of a fraction is not printed, then the digit before the point is omitted.
- 1.5 The number of digits required,  $n$ , may be specified either by a preset parameter or by a programme parameter set in X2. In the case of a fraction,  $n$  is the number of digits after the decimal point. In the case of an integer, non-significant left-hand zeros are suppressed (i.e. replaced by spaces;) except that the last digit will always be printed. Thus if a column of numbers is printed using the same value of  $n$  throughout, the right-hand margin will be straight providing no integer has more than  $n$  significant digits.
- 1.6 If any integer has more than  $n$  significant digits, it will be printed in full and the right-hand margin will be disturbed accordingly.
- 1.7 There is no rounding on output.

1.8 The punched output can be used for re-input by the Initial Orders if the signs are punched and if no fraction has more than 11 digits after the point.

1.9 The following table shows the cues to be used for the various styles of printing. For cues 01 to 04,  $n$  is a preset parameter. Optional parameters are supplied, the values of which are shown in the table.

Cue	01	02	03	04	05	06	07	08	09
Integer or Fraction	F	F	I	I	F	F	I	I	I
Value of $n$	Preset parameter				$n \times 2^{-38}$ set by programme in X2				3
Optional parameters	11	12	12	12	-	-	-	-	-
Sign printed	Yes	No	Yes	No	Yes	No	Yes	No	No

## 2. Parameter List

If any of cues 01 to 04 are to be used with values of  $n$  other than those in the optional list, a parameter list should be supplied in the following form:-

R 0 0 -0 4
4 - 04 -
0
$n_1$ 0 00 0.
0
$n_2$ 0 00 0.
0
$n_3$ 0 00 0.
0
$n_4$ 0 00 0.

Where  $n_1, n_2, n_3, n_4$  are the values of  $n$  for cues 01, 02, 03, 04 respectively.

## 3. Programme Parameter

If any of cues 05 to 08 are used, the number  $n \times 2^{-38}$  must be set in X2.

## 4. Link

This must be a 'go' order pair, set in X1. If it is set in the normal way (i.e. by an 00 order) the number will be preceded by two spaces; if it is set negatively (i.e. by means of an 02 order) it will be preceded by CRLF instead. In either case the link as obeyed is left in X1. Thus, if the subroutine is to be re-entered to print a second number on the same line and the same link is required, then it need not be reset in X1.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 3  
13.11.56

**DOUBLE-LENGTH NUMBER PRINT**

Prints out the double-length number  $2^{38}$  ( $pq$ ). Can also be used for the output of single-length fractions and integers.

**Name:** D.L.PRINT

**Store:** 7 blocks

**Uses:** U0; the accumulators are preserved in B0 and restored on exit.

**Cues:** 01 (0 + .0) to precede printing by CR LF  $\emptyset$   
02 (0 + .2+) to precede printing by one space.

**Time:** For integers  $234 + 24 n_1$  milliseconds (approximately)  
For fractions  $184 + 34 n_2$  milliseconds (approximately)  
For double-length numbers  $304 + 24 n_1 + 34 n_2$  milliseconds (approximately)

Where  $n_1$  and  $n_2$  are as described in section 1 below.

**Link:** Obeyed in 0.5 and left unaltered in X1.

**1. Method of Use:**

**1.1** The quantities  $n_1 = 4_c$  and  $n_2 = 5_c$  are programme-parameters which must be set by the master-programme before entry;  $n_1$  specifies the number of digits to be printed before the decimal point and  $n_2$  the number of digits to be printed after the point.

**1.2** R 5 only inspects the seven least significant bits of X4 and X5. Therefore modifiers may be left in  $4_m$  and  $5_m$  by the master programme.

**1.3** It is also possible to make R 5 print out fractions, by setting  $n_1 = 0$ , when the contents of X6 are ignored; or integers, by setting  $n_2 = 0$ , when the contents of X7 are ignored.

**1.4** This subroutine is primarily intended for printing out numbers in unscaled form, after calculations with numbers in a scaled form. For instance, given  $x/k$  in the range

$$-1 < \frac{x}{k} < 1$$

in the computer, where  $k$  is an integer less than  $2^{38}$ , one can arrange to print out  $x$ . This is done by causing the master-programme to multiply  $(x/k)$  by the integer  $k$ , setting the programme-parameters  $n_1$  and  $n_2$  and calling in R 5.

1.5 The numbers  $n_1$  and  $n_2$  must lie in the ranges:-

$$0 \leq n_1 \leq 12 \qquad \text{and} \qquad 0 \leq n_2 \leq 11$$

## 2. Form of Printing

2.1 If  $n_1 \neq 0$ , non-significant zeros in the integral part of the number to be printed are suppressed, i.e. replaced by spaces, except that the last digit before the decimal point is always printed even if zero.

2.2 If  $n_1 = 0$ , the point is preceded by one zero regardless of the contents of X6.

2.3 If  $n_2 \neq 0$ , the number is rounded off by the addition of  $(\frac{1}{2}) 10^{-n_2}$  and printed to  $n_2$  places.

2.4 Every number is preceded by its sign, which is printed immediately before the first unsuppressed digit.

2.5 If  $n_1 > 0$  but less than the number of significant digits in the integral part of the number to be printed, the number will be printed in full and the decimal point will thus appear out of alignment if a column of numbers with the same  $n_1$  is being produced.

2.6 When  $n_1 = 0$  and  $q$  is  $-1.0$  or rounds off to  $+1.0$ , then the number printed will be  $1 - 10^{-n_2}$  ( $= 0.99 \dots 9$  to  $n_2$  places) with the correct sign.

2.7 If  $(pq)$  rounds off to  $+2^{38}$  it will be printed correctly.

## 3. Error Stops

3.1 Inadmissible settings of  $n_1$  or  $n_2$  will cause the following loop stops:-

- (a) In 0.0 if  $n_2 > 11$
- (b) In 0.3 if  $n_1 = n_2 = 0$
- (c) In 0.4+ if neither  $n_1$  nor  $n_2$  is zero and  $q < 0$ . i.e. The number is not in the standard form  $(pq)$ .
- (d) In 0.6+ if  $n_1 > 12$

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## FAST D.L. PRINT FOR 7168 WORD STORE

A fast double-length print subroutine to take advantage of the fast paper tape punches now available.

The new routine can be substituted for the old version of R 5 without any other change, provided that the programme

is to be run on the 7168 store;  
 does not alter any orders in R 5;  
 does not use the fact that the old R 5 leaves the accumulators stored in B0.

**Name:** FAST D.L. PRINT

**Store:** 7 blocks (same as old R 5) plus isolated store: B1007, 1018, 1020.0 - 1023.6.

**Uses:** U 0.  
 The accumulators, U1 and U2 are preserved in B1+, 2+ and 3+ respectively of the subroutine and restored on exit.

**Cues:** 01 (0+.0 ) Print number preceded by CR LF  $\phi$   
 02 (0+.2+) Print number preceded by Sp

**Time:** The following are the approximate times in milliseconds when the subroutine is entered by cue 01:-

a) for the Creed 25 punch (33 characters per second):

$$\left. \begin{array}{l} \text{integer: } \left( 148 + \frac{107}{4} n_1 \right) \\ \text{fraction: } (169 + 30 n_2) \\ \text{mixed number: } \left( 184 + \frac{107}{4} n_1 + 30 n_2 \right) \end{array} \right\} \begin{array}{l} -23\text{ms. for } I < 10^4, n \leq 4 \\ +16\text{ms. for more than} \\ 8 \text{ digits after the point} \end{array}$$

The times taken when entry is made by cue 02 are between 45 and 50 milliseconds less than those shown above.

b) for the Teletype punch (60 characters per second):

$$\left. \begin{array}{l} \text{integer: } \left( 121 + \frac{161}{12} n_1 \right) \\ \text{fraction: } \left( 103 + \frac{50}{3} n_2 \right) \\ \text{mixed number: } \left( 144 + \frac{161}{12} n_1 + \frac{50}{3} n_2 \right) \end{array} \right\} \begin{array}{l} -23\text{ms. for } I < 10^4, n \leq 4 \\ +16\text{ms. for more than} \\ 8 \text{ digits after the point} \end{array}$$

The times taken when entry is made by cue 02 are 18 milliseconds less than those shown above.



c) for the Soroban (150 characters per second):

$$\left. \begin{array}{l} \text{integer: } \left(104 + \frac{41}{12} n_1\right) \\ \text{fraction: } \left(65 + \frac{20}{3} n_2\right) \\ \text{mixed number: } \left(116 + \frac{41}{12} n_1 + \frac{20}{3} n_2\right) \end{array} \right\} \begin{array}{l} -23\text{ms. for } I < 10^4, n \leq 4 \\ +16\text{ms. for more than} \\ 8 \text{ digits after the point} \end{array}$$

These times also apply when entry is by cue 02.

d) for the Creed 3000 (300 characters per second):

$$\left. \begin{array}{l} \text{integer: } \left(104 + \frac{1}{12} n_1\right) \\ \text{fraction: } \left(59 + \frac{10}{3} n_2\right) \\ \text{mixed number: } \left(113 + \frac{1}{12} n_1 + \frac{10}{3} n_2\right) \end{array} \right\} \begin{array}{l} -23\text{ms. for } I < 10^4, n \leq 4 \\ +16\text{ms. for more than} \\ 8 \text{ digits after the point} \end{array}$$

These times also apply when entry is by cue 02.

**Link:** Obedy in U0.5 and left unaltered in X1 on exit.

**Specification:**

All the details of the specification, including the list of loop stops, are the same as for the old R 5.

**Alterations to the Subroutine:**

a) Replace + by Sp

C 5  
X 4+.1  
⑭ 3 41  
X 4+.7  
③ 3 43  
X 5+.7  
③ 3 43

b) Suppress + (print  $\phi$  instead)

As above, but replace ⑭ by ⑯ and ③ by ⑤ in the N position.

c) Suppress  $\phi$  after CR LF

C 5  
X 5+.2  
0

d) Suppress Sp before number (print  $\phi$  instead)

C 5  
X 0+.3  
⑯ 1 40

© FERRANTI LTD 1960

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## DOUBLE LENGTH INTEGER PRINT

A self preserving subroutine which prints out  $(pq)$  as an integer.

**Name:** D.L. INTEGER PRINT

**Store:** 5 blocks

**Uses:** U0; B0.

**Cue:** 01 (0+.0)

**Time:** The approximate time for the Creed 25 punch (33 characters per second) is

$$20m + 30q + 325 \text{ milliseconds}$$

where  $m$  is the number of digits set to be printed (see section 1.1) and  $q$  is the number of layout spaces printed between groups of digits.

For the Teletype punch and faster punches, the approximate time is

$$9m + 19q + 276 \text{ milliseconds.}$$

**Link:** Obeyed in 0.7 and left unaltered in X1.

## 1. Method of Use

1.1 The number of digits to be printed,  $m$ , must be set in  $5_c$  before entry. It may be set by the instruction

$\textcircled{m}$  5 40 to obtain printing preceded by CR LF

or  $\textcircled{m}$  5 42 to obtain printing preceded by Sp Sp.

1.2 A second programme parameter,  $n$ , must be set in  $4_c$  before entry. The number will be printed out in groups of  $n$  digits separated by a single space. (Only if  $m$  is exactly divisible by  $n$  will the first group contain the full complement of  $n$  digits).

1.3 A modifier may be left in  $4_m$ .

1.4 On exit, block 0+ is left in U0 and re-entry can be made by jumping to 0.0.

## 2. Form of Printing

2.1 The sign of the number will immediately precede the most significant digit of the number.

2.2  $m$  and  $n$  will usually take values in the range 1 to 23 inclusive. If too small a value of  $m$  is used, the number will be printed out in full, but the layout will be upset. If  $m = 0$ , CR LF will be punched followed immediately by the sign and digits of the number; no spaces will be punched and in this case only, the value of  $n$  is immaterial. If  $n = 23$  no internal spaces will occur in the number.

2.3 Provided that no internal spaces are permitted, the output is suitable for re-input by R 100.

### 3. Error Stops

3.1 If  $n = 0$  (and  $m \neq 0$ ) a loop stop will occur in 0.5 which will contain

0.5	0	5	41	4
	0.5	5	63	

3.2 If the D.L. number is not in standard form on entry, a loop stop will occur in 0.0 which will contain

0.0	0	7	73
	0.0+	7	63

3.3 If R 9 is entered with OVR set, writing with overflow occurs in 0.0 which will contain the order pair given in 3.2 above.

Author: Dr. J. Eve of Durham University.

*This specification is distributed with the kind permission of Durham University by Ferranti Ltd. as part of its service to users of Ferranti Pegasus Computers.*

*The specification must not be reproduced in whole or in part without the permission of Durham University.*

FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
4.6.57.

## PRINT DOUBLE-LENGTH FRACTIONS

A self-preserving subroutine to print  $p + 2^{-38}q$  as a fraction to up to 23 decimal places.

$$-1.0 \leq p + 2^{-38}q < +1.0$$

Name: D.L.FRACTION PRINT

Store: 5 blocks.

Uses: U 0, 1; B 0.

Cues:

01

0+	0	72
0.0	0	60

To precede printing  
by CR LF  $\phi$

02

0+	0	72
0.2+	0	60

To precede printing  
by two spaces

03 a-order partial cue.

04 b-order partial cue.

Link: Obeyed in 1.7 and left unaltered in X1.

Time: About  $43n + 100$  milliseconds, where  $n$  is the number of figures after the point.

## 1. STYLE OF PRINTING

1.1 All numbers will be preceded by their signs and will have zero (or exceptionally 1) before the point.

1.2 The number of figures,  $n$ , required after the point should be set as an integer in X5. The rest of X5 must be clear.

1.3 Numbers are rounded to the last place printed.

1.4 If a number has the value  $-1.0$  or rounds off to  $\pm 1.0$ , it will be printed correctly with  $n$  zeros after the point.

1.5 If  $n = 0$ , the number will be punched correctly as  $\pm 0$  or  $\pm 1$ . The decimal point will not be punched.

1.6 Unless  $n = 0$ , the punched output is suitable for input by R 100 if L's are inserted where necessary and if CR LF or space is punched after the last number. If  $n = 0$ , the output will be accepted by R 100, but R 100 will treat  $\pm 1$  as  $\pm 1.2^{-38}$ .

## 2. RE-ENTRY

2.1 R 10 is a self-preserving subroutine, leaving its own block 0+ in U0 on exit.

2.2 Block 0+ can also be brought into U0 without being entered by using the order

0	0	72
---	---	----

tagged by the appropriate partial cue, 03 or 04.

2.3 Once block 0+ of R 10 is in U0, the subroutine can be entered by jumping to U0.0 to precede the printing by CR LF  $\phi$  or to U0.2+ to precede the printing by two spaces.

## 3. ERROR STOPS

3.1 *Loop Stops:*

(a) In 0.1 (0.1 0 60) if  $(p + 2^{-38}q) < -1.0$ .

(b) In 0.4 (0.4 5 63) if X5 is negative on entry.

(c) In 0.5 (0.5 5 63) if  $n > 23.2^{-38}$ .

3.2 *Writing with Overflow:*

If the subroutine is entered with the OVR set, writing with overflow will occur either in 0.0 or in 0.3 (due to a block write order in 0.2+). In these cases

0.0 contains:

0	7	73
30	2	40

and 0.3 contains

14	2	40
16	2	10

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
23.1.57.

**PRINT FLOATING POINT NUMBERS**

A self-preserving subroutine which prints in fixed or decimal floating point form the binary floating point number in X1.

**Name:** F.P. PRINT MK. 3

**Store:** 11 blocks.

**Uses:** U0; B0. The accumulators are preserved in B0 and restored on exit, except that X7 is altered in some cases. See Section 5 below.

**Cues:**

01

0+	0	72
0.2	0	60

To print in floating point form as:

CR LF  $\phi$  Argument; (Space); Decimal exponent; Space; Space.

02

0+	0	72
0.4+	0	60

To print in floating point form as:

Space; Space; Argument; (Space); Decimal exponent; Space; Space.

03

0+	0	72
0.2	7	66

To print in fixed point form as:

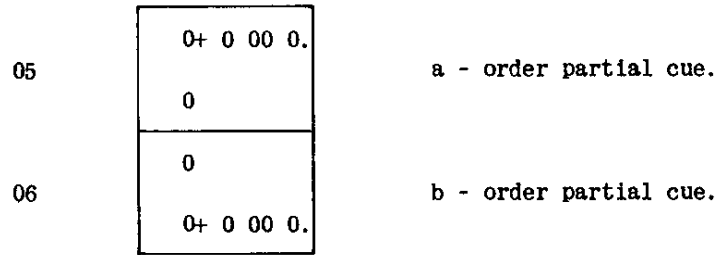
CR LF  $\phi$  Number.

04

0+	0	72
0.4+	7	66

To print in fixed point form as:

Space; Space; Number.



**Time:** In addition to the printing time, R 11 requires about

$$\frac{7 |a|}{20} \text{ milliseconds}$$

to convert the number to be printed from binary to decimal floating point, where  $a$  is the value of the binary exponent. See Section 7 for the effect of this on the reasonable size of the preset parameter.

**Link:** The link must be set in X6 where it will be left unaltered on exit. It may be a *go* order pair in which case it will be planted and obeyed in 0.0. Alternatively it may be a computing store link which will be obeyed in 1.7. A computing store link should be set with a 40 order since the rest of X6 must be clear.

**CONTENTS OF SPECIFICATION:**

Section	Page
1. Fixed and Floating Point Forms .. .. .	2
2. The Printed Form of the Argument .. .. .	2
3. The Printed Form of the Exponent .. .. .	3
4. The Contents of X5 and X7 .. .. .	3
5. Re-Entry .. .. .	3
6. Binary Floating Point Numbers .. .. .	4
7. Preset Parameter .. .. .	4
8. How to Avoid Spaces after Exponent .. .. .	5
9. A Special Case: $n_1 = n_2 = 0$ . .. .. .	5
10. Possible Writing with Overflow Stops .. .. .	6

**1. Fixed and Floating Point Forms**

Throughout this specification, to print *in floating point form* means to print a number as a separate argument and decimal exponent, and to print *in fixed point form* means to print an argument with the decimal point in the correct position.

**2. The Printed Form of the Argument**

**2.1** The value of the printed argument is determined by the two quantities  $n_1$  and  $n_2$  which are the contents of  $5_c$  and  $7_c$  respectively.

**2.2** The argument is printed with  $n_1$  decimal digits before and  $n_2$  decimal digits after the point. Non-significant zeros in the integral part of the argument will be suppressed (i.e. replaced by spaces); and the sign will appear immediately before the first digit printed. Non-significant zeros after the point will not be suppressed.

**2.3** The argument will be rounded to the last place printed.

**2.4** If the method of entry used calls for fixed point printing but  $n_1$  is less than the number of significant digits before the point, the number will be printed correctly but in floating point form with  $n_1$  places before the point in the argument.

### 3. The Printed Form of the Exponent

**3.1** When floating point output is used, the decimal exponent,  $d$ , will be printed as a signed integer. It will appear on the same line as the argument.

**3.2** If  $d < 10$  the argument and exponent will be separated by two spaces. If  $10 \leq d < 100$  they will be separated by one space. If  $d \geq 100$  there will be no spaces between them.

### 4. The Contents of X5 and X7

**4.1** The sign digits of X5 and X7 and the modifier position of X5 must always be clear on entry to R 11.

**4.2** The modifier position of X7 must also be clear when entry is by any of the ordinary cues, 01 to 04. For the state of  $7_m$  when entry is not by an ordinary cue see the following section.

### 5. Re-Entry

**5.1** R 11 is a self-preserving subroutine. It leaves its own block 0+ in U0 on exit and restores the rest of the computing store with the exception, in some cases, of X7.

**5.2** Block 0+ of R 11 can also be brought into U0 without being entered by using the order:-

0	0	72	+ partial cue.
---	---	----	----------------

This order must be tagged by cue 05 if it is an a-order or by cue 06 if it is a b-order.

**5.3** Once block 0+ of R 11 has been brought into U0 by this method or by previous use of one of the four ordinary cues, the subroutine can be entered by jumping to 0.2 to print on a new line or to 0.4+ to print on the same line as before after two spaces.

**5.4** When R 11 is entered in this way the contents of  $7_m$  determine the form of printing. If  $7_m = 0$  the number will be printed in floating point form, and if  $7_m = 1$  it will be printed in fixed point form.

**5.5** If  $7_m = 0$  and fixed point printing is required, this can be achieved by entering with a 66 order.

i.e. 

0.2	7	66
-----	---	----

or 

0.4+	7	66
------	---	----

Cues 03 and 04 use this method, and hence  $7_m$  must be clear when they are used.



5.6 When R 11 is entered by cue 03 or 04 or by a 66 order in the master programme, the accumulators are not stored until after the 66 order has been obeyed. Consequently, in these cases,  $7_m = 1$  on exit and must be cleared before re-entry by an ordinary cue or a 66 order.

6. Binary Floating Point Numbers

6.1 Although R 11 prints numbers in fixed or decimal floating point form, the numbers it handles are initially stored in X1 in binary floating point form.

6.2 A binary floating-point number  $x = A.2^a$  is held in a single word with the least-significant  $n$  bits representing the non-negative integer  $a + 2^{n-1}$ , and the most-significant  $39 - n$  bits representing the fraction  $A$ . The number of binary digits,  $n$ , allocated to the exponent is specified by a parameter-list; see section 7 below.

6.3 The exponent,  $a$ , is restricted to the range

$$-2^{n-1} < a < 2^{n-1} - 1.$$

If  $x$  is not zero then the argument,  $A$ , must satisfy

$$\begin{aligned} \frac{1}{4} < A < \frac{1}{2} & \quad \text{if } x \text{ is positive,} \\ \text{or} & \\ -\frac{1}{2} < A < -\frac{1}{4} & \quad \text{if } x \text{ is negative.} \end{aligned}$$

6.4 Zero is to be represented by  $A = 0$  and  $a = -2^{n-1}$ , i.e. by a word all of whose digits are zero.

7. Preset Parameter

7.1 If no parameter list is supplied by the programmer the value of  $n$ , (the number of digits used to represent the binary exponent), will be set as 9 by an optional parameter list.

7.2 If  $n = 9$  then the exponent may lie in the range

$$-256 \leq a \leq 255,$$

which permits the use of numbers whose absolute values lie in the range

$$2.10^{-78} < |x| < 2.10^{76}$$

approximately. The argument,  $A$ , will be represented with a precision of just under 9 decimal digits.

7.3 If  $n$  is to have any value other than 9, a parameter list of the following form must be provided:

R0	0	-0	1
11	-	04	-
$-2^{n-1}$			

7.4 In addition to printing time, R 11 takes about  $\frac{7|a|}{20}$  milliseconds to convert the

number to be printed from binary to decimal floating point, where  $a$  is the value of the binary exponent. Therefore, for any value of  $n$ , the maximum time which the conversion can take is  $\frac{7.2^{n-1}}{20}$  milliseconds. Zero is treated as a special case which

does not require conversion.

**7.5** This conversion time is negligible for any number likely to occur in the great majority of calculations; but it is about one second for numbers of the order  $10^{\pm 1000}$ . Such numbers are within range if  $n > 12$ .

**7.6** If, with  $n > 12$ , it should be required to print many numbers near the limits of the range, R 11 may be too slow. The subroutine should not normally be used with  $n > 15$  and never with  $n > 20$ .

### 8. How to Avoid Spaces after the Exponent

**8.1** The spaces after the exponent in floating point form printing may sometimes be inconvenient; for instance, when numbers are printed in a single column, these spaces are merely time wasting.

**8.2** The order pair which produces these spaces is in 10+.3 of R 11. It is:

17	4	10	3
17	4	10	3

**8.3** This can be replaced by a 'go' dummy order pair by first sending -1.0 to X1 and then using the order:

10	3	71
----	---	----

+ partial cue.

This order must be tagged by cue 05 of R 11 if it is an a-order or by cue 06 if it is a b-order.

### 9. A Special Case: $n_1 = n_2 = 0$

**9.1** The case of  $n_1 = n_2 = 0$  requires special mention. The argument will be printed with its correct sign. Apart from this, the printing will be as described in the following paragraphs.

#### 9.2 Floating Point Form Entry Used

The argument will be printed as  $\pm 0$ . The exponent printed will be that which would be correct if the argument,  $A$ , satisfied:

$$0.05 \leq |A| < 0.5$$

#### 9.3 Fixed Point Form Entry Used

- (a) If  $|\text{Number}| < 0.5$ ,  $\pm 0$  is printed.
- (b) If  $0.5 \leq |\text{Number}| < 1.5$ ,  $\pm 1$  is printed.
- (c) If  $|\text{Number}| \geq 1.5$ , the number will be printed in floating point form. It may appear either as:

Argument  $\pm 0$  together with that exponent which would be correct if the argument,  $A$ , satisfied:

$$0.1 \leq |A| < 0.5$$

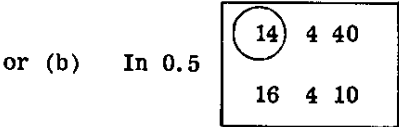
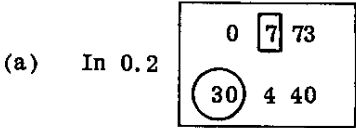
Or as:

Argument  $\pm 1$  together with that exponent which would be correct if the argument,  $A$ , satisfied:

$$0.5 \leq |A| < 1.0$$

**10. Possible Writing with Overflow Stops**

There are no loop stops in R 11, but, if the subroutine is entered with the overflow register set, one of the following writing with overflow stops will be encountered:



Due to a block write order in 0.4+

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
29.11.57.

**SHIFT AND PRINT**

This routine prints a series of numbers from the Main Store, each number being rounded off to a specified number of significant figures. Before printing, the numbers are multiplied by a scale factor of the form  $2^k$ ; a different scale factor may be used for each number. Each number is preceded by CR LF.

**Name:** SHIFT + PRINT

**Store:** 10 blocks

**Uses:** U0, 1, 2, 3; B0.

**Cues:**

01

0+	0	72
0.0	0	60

to precede printing by  $\phi$  LF.

02

1+	3	72
3.1	0	60

to precede printing by  $\phi$  only.

**Time:** Approximately  $30(c+1)$  ms. per number, where  $c$  is the number of characters punched on the tape.

**Link:** Obedied in 3.0 and left unaltered in X1.

**1. Method of Use**

The following four programme parameters must be set in X4 and 5 on entry:-

$m_1$ in 4 <sub>m</sub>	Main Store address of first number to be printed
$n_1$ in 4 <sub>c</sub>	Number of numbers to be printed
$m_2$ in 5 <sub>m</sub>	Main Store address of first scale factor
$n_2$ in 5 <sub>c</sub>	Number of significant figures required

**2. Scaling**

**2.1** Each scale factor  $2^k$  is represented by the index  $k$ , stored as an integer.  $k$  may be positive or negative. The indices for consecutive numbers are normally

stored in consecutive Main Store locations starting at  $m_2$ . The subroutine reads each number  $x$  into X7 and shifts it up or down to form  $x \cdot 2^k$  in PQ for printing.

2.2  $k$  should not exceed +38. If  $k > 38$  the number  $x \cdot 2^k$  may overflow and the subroutine will give incorrect results.

2.3  $k$  should not be less than  $-(38 - 3n_2)$  if all  $n_2$  significant figures are required. Printed numbers will not be accurate in the 12<sup>th</sup> and subsequent decimal places.

### 3. Preset-Parameters

The separation between the addresses of consecutive scale factors is specified by Preset-parameter 01. If no parameter-list is supplied the separation will be set equal to 1 by an optional parameter-list and the scale factors must be in consecutive locations.

To specify a separation of  $S$  locations the parameter-list should be punched as shown below

R 0 0 -0 1	Title
26 - 04 -	
S - -0 0.	Separation
0	

Note that if  $S = 0$  all numbers will be multiplied by the same scale factor.

### 4. Form of Printing

All numbers are preceded by CR LF sign. No spaces are printed. Some examples are given below with the number of significant figures  $n_2 = 4$ .

True number	Printed as:
+0	+0
-12.5	-12.50
+123.001	+123.0
-12345	-12350
+123.44	+123.4
-0.12344	-.1234
+0.0012304	+ .001230
+0.99996	+1.000
+123	+123
+0.125	+ .1250

Note that  $n_2$  significant figures are always printed unless the number is an exact integer less than  $10^{n_2-1}$ .

### 5. Error Stops

If R 26 is entered by cue 01 with the OVR set, writing with overflow will occur in U0.1 which will contain:-

0	7	73
13	6	40

If entry is made by cue 02 with OVR set, writing with overflow will occur in U 3.1:-

0	7	73
17	0	10

**Author:** Mr. J.V. Oldfield, Electrical Engineering Department, Queen Mary College,  
University of London.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
27.7.56

**£.S.D. PRINT FROM PENCE**

The contents of X7 are treated as being an integral number of pence and are printed out as £.s.d.

i.e. If  $X7 = X.2^{-38} = (240a + 12b + c)2^{-38}$   
where  $a, b, c$  are integers

and  $a \geq 0$ ,

$0 \leq b < 19$ ,

$0 \leq c < 11$ ,

then this subroutine prints out  $a . b . c$  with suppression of more significant zeros.

**Name:** £.S.D. PRINT (PENCE)

**Store:** 3 blocks.

**Uses:** U0; the accumulators are preserved in B0 and restored on exit.

**Cues:** 01 to precede printing by CR LF.  
02 to print on same line preceded by one space.

**Link:** Obeyed in 0.7 and left unchanged in X1.

**Notes:**

- 1)  $X$  must be positive or zero and is placed in X7. A loop stop will occur in 0.4 if  $X$  is negative.
- 2) The number of decimal places of pounds which it is required to print,  $N$ , is to be placed in X2<sub>c</sub>.

If  $a \geq 10^N$  it is printed in full and will thus appear out of alignment if a number of sterling quantities are being printed in a column.

If  $a < 10^{N-1}$  more significant zeros are suppressed.

- 3) The zero suppression provided does not cover the last place of either £, s. or d. For example,

12/6d is printed as:-

0.12. 6

and 10d. as:-

0. 0.10

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
25.6.57.

**SIGNED £.S.D. PRINT FROM PENCE**

A self-preserving subroutine. The contents of X7 are treated as being an integral number of pence and are printed as £.s.d.

i.e. If  $x_7 = \pm(240a + 12b + c)2^{-38}$   
where  $a, b, c$  are integers such that  
 $0 \leq a$   
 $0 \leq b \leq 19$   
 $0 \leq c \leq 11,$

then this subroutine prints  $a, b, c$  with suppression of more significant zeros, preceded by a minus sign if  $x_7$  is negative.

(R 42 uses R 1026, the Initial Orders Integer Print Routine.)

**Name:** SIGNED £.S.D.PRINT

**Store:** 4 blocks.

**Uses:** U0; X6; B0.

**Cues:**

01

0+	<span style="border: 1px solid black; padding: 0 2px;">0</span>	72
0.0	0	60

To precede printing by  
CR LF  $\phi$

02

0+	<span style="border: 1px solid black; padding: 0 2px;">0</span>	72
0.2	0	60

To precede printing by  
one space.

03 a-order partial cue.

04 b-order partial cue.

**Time:** About  $30N + 19$  milliseconds, where  $N$  is the total number of characters printed.

**Link:** The link should be set in X1 where it will be left unaltered on exit. It may be a *go* order pair which will be planted and obeyed in U0.7. Alternatively it may be a Computing Store link set with a *40* order, in which case it will be obeyed in U0.6+.

**Error:** None.



## 1. FORM OF OUTPUT

1.1 The number of places of pounds required,  $n$ , must be set in X2 as  $n \cdot 2^{-38}$  before entry. If  $a \geq 10^n$  or  $a < -10^n$ , it will be printed in full and will thus appear out of alignment if a number of sterling quantities are printed in a column with the same value of  $n$ .

1.2 Zeros at the more significant end of  $a$ ,  $b$  or  $c$  will be suppressed, except that the last digit in each will always be printed. For example, 12/6d. will be printed as:-

0.12. 6

and 10d. will be printed as:-

0. 0.10

1.3 Negative sums of money will be preceded by a minus sign which will appear immediately before the first digit printed. No plus signs will be printed but positive sums will be preceded by an extra space.

1.4 The punched output of R 42 is acceptable as input data for R 142 providing that CR LF or space appears after each sum.

## 2. RE-ENTRY

2.1 R 42 is self-preserving, leaving its own block 0+ in U0 on exit.

2.2 Block 0+ of R 42 can also be brought into U0 without being entered by the order

0 0 72

tagged by the appropriate partial cue, 03 or 04.

2.3 Once its block 0+ is in U0, the subroutine can be entered by jumping to U0.0 to precede the printing by CR LF  $\phi$  or to U0.2 to precede the printing by one space.

## 3. OVERFLOW STOPS

3.1 If R 42 is entered with the OVR set, writing with overflow will occur in U0.0 which will contain:-

0 7 73  
0.6 4 00

3.2 If  $x_7 = -1.0$  on entry, writing with overflow will occur in U0.5 (due to a single word write instruction in 0.4+). 0.5 will contain:-

3+ 0 72  
0.3+ 0 60

3.3 If either of these stops occur CR LF or space will already have been punched. In the case of the second stop  $\phi$  will have been punched after CR LF.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## MIXED RADIX OUTPUT

A subroutine to punch on paper tape a positive quantity in units of different denominations, such as hours and minutes. It punches the positive quantity  $q$  (in X7) in one of the three following ways:-

- (a) As an unsigned integer,  $q$ .
- (b) As two unsigned integers,  $a$  and  $b$ , satisfying

$$ar + b = q \quad 0 \leq b < r$$

where  $r$  is a preset positive integer ( $r \leq 100$ ).

- (c) As three unsigned integers,  $a$ ,  $b$  and  $c$ , satisfying

$$ast + bs + c = q \quad 0 \leq b < t, 0 \leq c < s$$

where  $s$  and  $t$  are preset positive integers ( $s, t \leq 100$ ).

The number and nature of the characters separating  $a$ ,  $b$  and  $c$  are also preset.

- Name:** MIXED RADIX OUTPUT
- Store:** 4 blocks.
- Uses:** U0; B0. X1 and X2<sub>m</sub> are altered in some cases (see section 5).
- Cues:** 01 (0+.0+) to punch  $q$  as a single integer (or to treat  $q$  in the same way as on the previous entry, see section 5.2).
- 02 (0+.0) to punch  $q$  as two integers.

03

0+	0	72
0.0	2	66

to punch  $q$  as three integers.

**Time:** The following are the approximate times in milliseconds:-

- (a) for the Creed 25 punch (33 characters per second):  $9 + 30\frac{1}{3}n$ .
- (b) for the Teletype 60 punch (60 characters per second):  $50 + 16\frac{2}{3}n$ .
- (c) for the Creed 3000 (300 characters per second):  $55 + 25d + 3\frac{1}{3}n$ .

$n$  is the total number of characters punched and  $d$  is the number of integers punched. The above times apply to the 7168-word store.

**Link:** Obeyed in 0.5 and left as obeyed in X1. See sections 4 and 5.

### 1. Method of Use

Before entry to the subroutine, the positive quantity,  $q$ , must be placed in X7. A positive integer  $D$  ( $0 \leq D \leq 12$ ) must be placed in X2 to indicate the number of digits to which  $q$  or  $a$  is to be punched (see section 2 below).  $2_m$  must be left clear (see section 5.2).

### 2. Form of Punching

2.1 If the subroutine is entered by cue 01,  $q$  will be punched as an unsigned integer to  $D$  decimal digits.

2.2 If the subroutine is entered by cue 02, two unsigned integers,  $a$  and  $b$ , will be punched, where  $a$  and  $b$  satisfy

$$ar + b = q \quad 0 \leq b < r$$

and  $r$  is a positive integer ( $\leq 100$ ) specified by preset parameter 01. The integers  $a$  and  $b$  are punched to  $D$  digits and 2 digits respectively.

2.3 If the subroutine is entered by cue 03, three unsigned integers,  $a$ ,  $b$  and  $c$ , will be punched, where  $a$ ,  $b$  and  $c$  satisfy

$$ast + bs + c = q \quad 0 \leq c < s, 0 \leq b < t$$

and  $s$  and  $st$  are positive integers ( $s, t \leq 100$ ) specified by preset parameters 02 and 03. The integer  $a$  is punched to  $D$  digits, the integers  $b$  and  $c$  to 2 digits each.

2.4 The number and nature of the separating characters between  $a$  and  $b$ , and between  $b$  and  $c$  are specified in preset parameter 04.

2.5 The first integer punched is preceded either by CR LF or by two spaces according to the manner of setting the link (see section 4).

2.6 Non significant zeros of  $a$ ,  $b$  and  $c$  are replaced by spaces except for the least significant digit which is always punched. Therefore there is right hand alignment of digits of corresponding rank in different rows. If  $a$  has more than  $D$  significant digits, all of these are punched at the expense of right hand alignment.

### 3. Parameter List

3.1 The parameter list is punched as follows:-

	R 0 0 -0 4
	43 - 04 -
01	+r
02	+s
03	+st
04	m 0 00 0.
	n 0 00

In preset parameter 04,  $n$  is the numerical value (output via register 16) of the character to be used to separate the component integers of  $q$  and  $m$  is the number of such characters.

3.2 If no parameter list is supplied, the following values are supplied by an optional parameter list:-

$r$	=	+60	
$s$	=	+12	
$st$	=	+240	
$m$	=	1	} separating characters set to be single spaces.
$n$	=	14	

In this case the subroutine may be entered:

- (i) by cue 01 to print an integer,
- (ii) by cue 02 to print an integral number of
  - (a) minutes of time as hours and minutes,
  - (b) minutes of arc as degrees and minutes,
  - (c) seconds (of time or arc) as minutes and seconds.
- (iii) by cue 03 to print an integral number of pence as pounds, shillings and pence.

#### 4. Link

The link must be a "go" order pair. If it is set in the normal way by an 00 order, the first number punched is preceded by 2 spaces. If it is set negatively by means of an 02 order, the number is preceded by CR LF.

#### 5. Contents of X1 and X2

5.1 If the link is set by an 02 order, it is negated before being obeyed and left in X1 as obeyed.

5.2 If the routine is entered by cue 02 or cue 03,  $2_m$  will contain 0.1 or 0.2 respectively on exit. The routine may be re-entered by cue 01 without resetting X2, to punch in the same style and to the same number of digits as for the previous entry.

#### 6. Alterations to the Subroutine

6.1 To punch the first digit of  $b$  or  $c$ , whether zero or not, the following amendment tape may be read in after the A3 on the master programme tape:-

```
C 43
X3+.6
0.7 0 60
Z
```

6.2 To replace a more significant digit of  $b$  or  $c$  by a character other than a space (e.g. a point), the following amendment tape may be read in after the A3 on the master programme tape:-

```
C 43
X3+.6+
(π) 7 40
Z
```

where  $n$  is the numerical value (output via register 16) of the character to be punched.

## 7. Errors

7.1 If  $x_7$  is negative on entry, there will be a loop stop in 0.0+ (0.0+ 7 63).

7.2 If  $D = 0$  or  $D > 12$ , the first integer punched will be punched to 12 digits. No catastrophic effects will occur provided that  $0 \leq D < 2^{25}$ .

7.3 The subroutine will fail if  $2_m$  is not clear before its cue is obeyed, except as explained in section 5.2.

Author: Mr. G.H.L. Buxton of Whitworth Gloster Aircraft Ltd.

*This specification is distributed with the kind permission of Whitworth Gloster Aircraft Ltd. by Ferranti Ltd. as part of its service to users of Ferranti Pegasus Computers.*

*The specification must not be reproduced in whole or in part without the permission of Whitworth Gloster Aircraft Ltd.*

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
10.4.58.

## CHARACTER PRINT

A routine to punch on paper tape 6 6-bit characters, which must be packed in X6, in the form used with the magnetic tape Bull printer.

**Name:** CHAR. PRINT

**Store:** 4 blocks

**Uses:** U0, 1, 2, 3; X3, 4, 5, 6, 7.

**Cues:** 01 (0+.0) to precede printing by CR LF.  
02 (0+.2) to print 6 characters only.

**Time:** 210 ms (Cue 01), or 150 ms (Cue 02) plus 60 ms for each sequence of letter shift characters punched. (The sequence may be only one character.)

**Link:** Obeyed in 2.0 and left unaltered in X1.

## 1. Word Layout

The word is taken to consist of 6 6-bit characters laid out as shown below:

No. of bits	3	6	6	6	6	6	6
	s	$c_1$	$c_2$	$c_3$	$c_4$	$c_5$	$c_6$

where s indicates spare bits

$c_i$  is the  $i^{\text{th}}$  character punched  $i = 1, 2, \dots, 6$

## 3. Code

The code corresponds as closely as possible to that used by the Pegasus Data Processing System with Bull printer (as described in CS.147). This does not use all the 64 possible combinations; those not used are printed out by this subroutine as \* (asterisk). Space (code 0) is printed as \* (erase), so that its presence at the end of a line can be detected. The four characters which control special printing facilities are coded as ( ) > and ≥, corresponding to 26, 10, 58 and 42 respectively. All other characters have equivalents in the teleprinter code.

#### 4. Shift

Both shifts may be used by the subroutine but a figure-shift will be punched at the end if the last character punched was in letter-shift.



## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
27.7.56.

**ORDER PRINT**

Performs the same function as warning character P with optional printing suppressed.

Prints out orders followed by CRLF with extra LF at the ends of blocks.

No CRLF before the first order printed.

**Name:** ORDER PRINT  
**Store:** 3 blocks.  
**Uses:** U0, 1; B0.  
 The accumulators are preserved in B0 and restored on exit.  
**Cues:** 01, 02, 03, 04. Please see notes below for explanations.  
**Link:** Obeyed in 1.7 and left unchanged in X1.  
**Notes:** When Cues 01, 02, or 03 are used, the contents of X7 are taken to be an order pair.

01 punches the a-order of this pair with a stop if it is a stop order-pair.

02 punches the b-order of the pair without any stop; and

03 punches the whole order pair.

04 is used to punch a series of  $n$  orders from the main store starting at the word whose address is  $m$ ; where:-

$n$  is to be specified in X7c,

$m$  is to be specified in X7m,

and the sign digit of X7 is to be 0 if the first order to be punched is the a-order in word  $m$  and 1 if the first order to be punched is the b-order in word  $m$ .

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
27.7.56.

**TEXT INPUT AND OUTPUT**

A routine to facilitate the output of descriptive matter (e.g. English words) during the course of a programme. There is an interlude for the input of the descriptive matter.

**Name:** TEXT IN/OUT

**Store:** 1 block + 1 location for the output subroutine, + index + texts (see Section 4 below).

**Uses:** U0, B0.  
During Input B0.0 is used and a total of 5 blocks and 7 locations are occupied; these are normally overwritten (except for the output subroutine) by the index and texts.

**Cue:** 01.

**Link:** Obeyed in 0.1 and is undisturbed in X1.

**Time:** The output time is roughly 30 ms per character.

**1. General Description**

**1.1** The various items of descriptive matter dealt with by R 52 are referred to as *texts*; there may be any number of them and each may be of any number of characters.

**1.2** R 52 is made up of two parts:-

- a) An output subroutine, occupying 1 block and 1 location in the main store. This is an ordinary subroutine which may be entered (by cue 01) to print a selected text.
- b) An interlude, temporarily occupying about 5 blocks, which is obeyed as soon as Assembly has read it in from the library tape. The interlude reads in the texts from the main tape-reader and stores them in a form suitable for later output.

**2. Input of the Texts**

**2.1** Each text is to be punched as if it were an N-sequence for the Initial Orders, i.e. the text should be preceded by a warning character N and terminated by two or more  $\phi$ 's (blank tape). The characters of a text are packed by R 52, seven to a word in the store, and each text starts a new word. For example, a text of 25 characters occupies 4 words of storage.

**2.2** The texts are grouped together to form a text-tape, on which the texts are numbered 0, 1, 2, . . . in the order of punching. After the last text a warning character L is to be punched. There must also be an L before the first text. The portion of tape between the two L's is read by the R 52 interlude which recognises the following warning characters only:-

N A text follows,  
 L End of texts,  
 Z Wait (77 stop); return to interlude after run → stop → run.

Each warning character is to be punched as if it were an Initial Orders warning character.

**2.3** The part of the text-tape before the first L is read by the Initial Orders; here one may punch an ordinary N-sequence, to act as the name of the text-tape, or a T-sequence (see section 4.2 below). The complete text-tape is thus made up as follows:-

N	Name of Text-Tape	} Read by Initial Orders
T	A+	
L		
N	Text Number 0	} Read by R 52 interlude
N	Text Number 1	
⋮		
N	Last Text	
L		

The vertical lines indicate blank tape. The T-sequence before the first L may be omitted if there are not more than 8 texts (see section 4.2 below).

**2.4** The text tape is read (in the main tape-reader) as soon as R 52 has been read by Assembly from the library tape. It should therefore appear after the A2 and before the A3 on the main programme tape. The interlude normally stores the texts over its own locations in the main store. When all the texts have been read in control reverts to Assembly, which then reads in more of the library tape if this is needed.

### 3. The Output Subroutine

**3.1** If at some stage in the master-programme it is desired to cause output of text number  $r$ , then it is necessary only to set  $r$  in  $6_c$  and to call in R 52 with cue 01. When called in in this way the output subroutine will punch the text, character-for-character, as it was read in. In order to obtain correct printing it is therefore essential that the text should include any characters (e.g. CR, LF Sp,  $\lambda$ ,  $\phi$ ) needed for layout or other purposes.

**3.2** The accumulators are preserved in B0. The link is obeyed in U0.1. A loop-stop may occur in U0.3+ of the output subroutine if an attempt is made to cause output of a text which has not been read in.

#### 4. Allocation of the Store

**4.1** While the R 52 interlude is reading in the texts it is simultaneously building up an index. This index contains one word for each text. The  $r$ th word in the index has (in the modifier position) the address of the first word of the  $r$ th text ( $r = 0, 1, 2, \dots$ ). The index is later used by the R 52 output subroutine.

**4.2** In the following description addresses are relative to the start of R 52.

When R 52 has just been read by Assembly the following locations in the main store are occupied:-

- a) B0+.0 to B1+.0 by the output subroutine,
- b) B1+.1 to B5+.6 by the interlude.

When the texts are read in the interlude is overwritten. The index starts at B1+.1 and occupies  $n$  words if there are  $n$  texts. The texts start after the index (not necessarily immediately after it) at an address equal to the Transfer Address (TA); this is set equal to B2+.1 before the interlude is entered, thus allowing space in the index for up to eight texts. If there are more than 8 texts the TA may be set by a warning character T on the text tape *before* the first L (see section 2.3 above). The address after this T should be relative to the start of R 52. By writing T 4+.1, for example, space is left for the index to extend up to 24 words — i.e. there may be up to 24 texts.

**4.3** As explained above, the characters of the texts are packed, seven to a word, but the last word of each text may not be completely filled. The first seven characters of the next text occupy the next available word. The TA is adjusted at the end so that the next routine read by Assembly after R 52 goes into the next available block after the texts.

**4.4** It should be noted that all  $\phi$ 's (blank tape) after the warning-character N up to the first non-blank character will form part of the text, as will also the first of the  $\phi$ 's marking the end of the text: this is analogous to the usual N-sequences of the Initial Orders. As an example consider a text punched as:-

...  $\phi\phi\phi\lambda N\phi\phi\phi$  CR LF  $\lambda$  RESULT  $\phi\phi\phi$  ...

In this case the 13 underlined characters will make up the stored form of the text — they will occupy 2 words in the main store since R 52 packs up to 7 characters in one word.

#### 5. Error Stops on Input

**5.1** The following loop-stops occur should the text-tape be mispunched:-

- a) In U1.7 — After the first L an  $\alpha$ -search is entered which ignores blank tape ( $\phi$ ), CR, LF and Er. It should at some stage read  $\lambda$  (before a warning character). If any other character is read there is a loop stop in 1.7.
- b) In U2.3 if any character other than  $\phi$  is read immediately after a warning character.
- c) In U2.6 if any warning character other than N, L or Z is read.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
11.2.57.

**BINARY PUNCH SUBROUTINE**

A subroutine to punch out consecutive words from the Main Store in a form suitable for Binary Input, R 1031.

R 53 is a steering routine using R 1033.

**Name:** BINARY PUNCH MK 4  
**Store:** 1 block.  
**Uses:** U0,1,2,5.3,5.4; B0.  
**Cue:** 01 (0+.0)  
**Time:** 2 seconds per block punched.  
**Link:** Obeyed in 0.7 and left unaltered in X1 on exit.

**Method of Use**

The addresses of the first and last words to be punched must be placed in 6m and 7m respectively before entering R 53. The rest of X6 and X7 must be clear.

The subroutine will punch six inches of blank tape followed by the contents of the specified section of the Main Store. When the output tape is read in by Binary Input the numbers (or orders) will be restored to their original locations.

If more than one section of the Main Store is required, e.g. if a series of numbers are stored in B 2.0 to B 19.4 and B 25.0 to B 32.7, the subroutine must be entered afresh to punch each section.

It may be convenient to have an optional stop or a 77 order before the subroutine is entered for the first time, in order to allow any previous output tape to be torn off.

**Note:** R 53 replaces the former R 50, which should now be considered obsolete. The specification of R 50 will not be re-issued.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
5.3.57.

**BINARY PUNCH WITHOUT TRANSFER ADDRESS  
(NON-ASSEMBLY VERSION)**

This routine may be used to punch out programme or data from the Main Store in a form suitable for BINARY INPUT, R 1031. The punching is preceded by 1.3 inches of blank tape. The Transfer Address is not punched at the front of the binary tape, which may therefore be read in to any part of the Main Store.

**Name:** BIN.PUNCH WITHOUT T.A.  
(NON-ASSEMBLY)

**Uses:** The entire Computing Store.

**Entry:** By a J-sequence read from tape. See Section 2 below.

**Exit:** After the specified section of the Main Store has been punched out, control is returned to the Initial Orders, which proceed to read more tape.

**Time:** 2 seconds for each block punched.

**Note:** Most programmes contain internal block transfers which prevent them from being moved in the Main Store. It is, however, sometimes convenient to store a programme in one section of the store and move it before obeying it.

**1. METHOD OF USE**

1.1 First prepare a steering tape as described in section 3 below.

1.2 Insert R 2054 programme tape in the main tape reader. (This tape has T 508.0 at its head. If the programme to be punched occupies this part of the store, R 2054 must be stored elsewhere. If the tape is inserted after the T 508.0, R 2054 will go into the next available block in the store, as specified by the transfer address in U5.7).

1.3 START and RUN to read in R 2054.

1.4 *Tear off previous output tape and set Handswitch 0 = 1 to suppress optional printing.*

1.5 Insert the steering tape in the main tape reader.

1.6 Operate the RUN key to read in the steering tape.

## 2. ENTRY

**2.1** R 2054 is entered by the warning-character sequence J 0+ CR LF (or J 508.0 CR LF), which is followed on the tape by two addresses specifying the first and last words to be punched out. These two addresses are separated by a minus sign and terminated by CR LF. For example, to punch out in binary the contents of B2.0 to B19.4 inclusive the following tape should be used:-

J 0+

2.0 - 19.4

This would normally be punched as follows:-

.... $\phi\phi\phi\phi$  CR LF  $\lambda$  J $\phi$  Sp 0+ CR LF 2.0 Sp - Sp 19.4 CR LF  $\phi\phi\phi\phi$ ....

All spaces here indicated are entirely optional but the clarity of the print-out of the steering tape is improved by including them.

**2.2** Only one pair of addresses can follow each J 0+. If it is required to punch out more than one section of the Main Store, a separate J 0+ sequence must be used for each section. Each section will be preceded by 1.3 inches of blank tape.

**2.3** If the Relativiser is altered after reading in the programme of R 2054 but before entering it, it cannot, of course, be entered by using J 0+. Instead the sequence J 508.0 should be used or J B.0 if the first block of R 2054 has been put in some block, B, which is other than 508.

## 3. THE STEERING TAPE

**3.1** All parts of the steering tape other than the addresses following J-sequences are read by the Initial Orders in the normal way. Therefore warning character sequences can be produced on the output tape by means of N-sequences on the steering tape.

**3.2** A D and an N-sequence should appear at the head of the output tape and an E- or J-sequence followed by blank tape and a number of erases will be required after the programme. Also some T-sequences will probably be needed.

3.3 For example, the following might be a typical steering tape for R 2054:-

```

CR LF
λ Nφφ.....φφ CR LF
λ Dφ CR LF
λ Nφ CR LF
λ XYZ. TEST 1 CR LF LF
φφ.....φφ CR LF
λ Nφφ.....φφ CR LF
λ Tφ 2.0 CR LF
φφ.....φφ CR LF
λ Jφ 0+ CR LF
2.0 - 19.4 CR LF
λ Nφφ.....φφ CR LF
λ Tφ 400.0 CR LF
φφ.....φφ CR LF
λ Jφ 0+ CR LF
25.0 - 32.7 CR LF
λ Nφφ.....φφ CR LF
λ Eφ 2.0 CR LF
φφ.....φφ CR LF
λ Nφφ.....φφ Er Er.....Er Er φφ CR LF
λ Zφ

```

The underlined characters will be reproduced on the output tape.

3.4 Programmes should be punched out together with the subroutines they use, so no A1, L, A2 or A3 will be needed.

3.5 The steering tape should end with a Z to prevent it running out of the tape reader.



FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
31.7.57

## PRINTED GRAPH

Each time this subroutine is entered it plots one point representing a dependent variable  $y$ . The  $x$ -axis is marked by a full stop and the point plotted is indicated by + or - according to the sign of  $y/c$ .  $c$  is the plotting unit.

Name: GRAPH

Store: 1 block

Uses: U0; B0.

Cue:

01

0 +	0	72
0.5	0	60

Time:  $30n$  milliseconds, where  $n$  is the number of characters printed.

Link: Obeyed in 0.5 and left unaltered in X1.

## 1. Initial Settings

1.1 The variable  $y$  must be set in X7.

1.2 The plotting unit,  $c$ , must be set in X6.  $c$  should be chosen so that  $y/c < 68.5$  for all values of  $y$  which may occur. This is because the teleprinter will not print more than 69 characters on one line.

1.3 The number of line feeds,  $l$ , required between lines of the graph must be set in X5<sub>c</sub>.  $5_m$  is not used by the subroutine and may be used by the programmer.

## 2. Layout

2.1 The  $x$ -axis runs vertically down the left hand side of the paper. The value of the dependent variable  $y$  is represented by the horizontal distance between the  $x$ -axis and the point plotted.

2.2 On entry the subroutine first prints a full stop to mark the  $x$ -axis. It then prints  $([y/c]_r - 1)$  spaces, followed by the sign of  $y/c$  and finally  $l$  line feeds.

$$[y/c]_r = y/c \text{ rounded to the nearest integer.}$$

2.3 If the interval in the abscissa is to be changed, the new value of  $l$  must be set in  $5_c$  before printing the line *preceding* the first interval of the new size.

2.4 If  $[y/c]_r = 0$  only the full stop and  $l$  line feeds will be printed.

2.5 The carriage return between lines must be supplied by the programmer, but it will often be convenient to begin each line by printing the appropriate value of  $x$ . The print routine should give constant width of printing and precede the number by CR LF. In this case an extra CR will not be required, and  $5_c$  should be set as  $(l - 1)$  instead of  $l$ .  $5_c$  may be zero if required.  $c$  must now be chosen such that  $y/c < (68.5 - n)$ , where  $n$  is the number of characters printed for  $x$ .

### 3. Incorrect Entry

3.1 If  $5_c$  is incorrectly set the subroutine may cycle from 0.1 to 0.4 printing LF's.

3.2 If  $c = 0$ , meaningless results will be produced and the OVR will be set.

3.3 If the value of  $[y/c]_r$  is unreasonably large, the subroutine will cycle in 0.0 - 0.0+ - 0.1 - 0.0 .... etc. printing spaces. The value of the counter controlling this loop may be monitored in X7.

3.4 If the subroutine is entered with the OVR set, writing with overflow will occur in 0.5 which will contain:-

0	7	73
0.5	1	10

This subroutine is based on an idea by Dr. E.S. Page of the University of Durham.

FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
11.8.58.

## BINARY PUNCH PROGRAMME

This routine provides a very simple means of obtaining a binary version of a programme. It punches out the contents of all non-zero locations in the Main Store, preceded by a clear store sequence. The form of the output is suitable for Binary Input, R 1031.

**Uses:** The entire Computing Store; B0 (but see section 2.2).

**Time:** Approximately 20 seconds plus 2 seconds for each block punched.

## 1. Method of Use

- 1.1 Run the Clear Store routine, R 2901.
- 1.2 Read the programme(s) to be binary punched.
- 1.3 Insert R 2057 in the main tape reader, and tear off previous output tape.
- 1.4 START and RUN.

## 2. Output

2.1 The first part of the R 2057 tape is copied onto the front of the output tape by an N-sequence. This consists of a heading:

N  
PUNCHED ON  
*date*  
D

followed by a programme which will clear the Main Store locations B1.0 to B511.5 inclusive when this new tape is read in later.

2.2 Next follows a binary punch of B0. B0 is then used by R 2057, and all non-zero words in the Store between B1.0 and B511.5 inclusive, are punched out. Each group of consecutive non-zero words is preceded by 1.2 inches of blank tape, directive A0 and the appropriate Transfer Address in binary. There is a checksum at the end of each group.

2.3 The output tape is terminated by a Z-directive and a blank tape and erase sequence.

2.4 The new tape may be read by operating the START and RUN keys and it will restore the contents of B0.0 to B511.5 to be the same as they were before R 2057 was read.

**Author:** Mr. C. Strachey of the National Research Development Corporation.

## PEGASUS LIBRARY SPECIFICATION

## BINARY PUNCH WITHOUT TRANSFER ADDRESS

A subroutine to punch out consecutive words from the Main Store in a form suitable for Binary Input, R 1031. Unlike R 53, R 58 does not punch the Transfer Address at the start of the tape so that the numbers (or orders) punched out can be read back to any part of the Store.

R 58 is a steering routine using R 1033.

**Name:** BIN. PUNCH (NO T.A.)

**Store:** 1 block.

**Uses:** U 0, 1, 2; B 0.

**Cue:** 01 (0+.1)

**Time:** 2 seconds per block punched.

**Link:** Obeyed in 0.3 and left unaltered in X1 on exit.

## Method of Use

Before entering R 58 the address of the first word to be punched must be placed in  $6_m$  and the total number of words in  $6_c$ .

The subroutine punches the required section of the Store preceded by two inches of blank tape and an A0 directive (for entering Binary Input); the punching is terminated by a 'binary J' (for returning to Initial Orders Input) and a checksum.

When the output tape is read the numbers (or orders) will be written to the part of the Store specified by the Transfer Address (in  $U 5.7_m$ ) and this must, therefore, be set before reading the tape.

The subroutine must be entered afresh for each section of the Store required to be punched.

**Author:** Mr. J.G.F. Francis of the National Research Development Corporation.

© FERRANTI LTD 1959

London Computer Centre,  
21, Portland Place,  
LONDON, W.1.

*Not to be reproduced in whole or  
in part without the prior written  
permission of Ferranti Ltd.*

*Issue 1  
14th April, 1959.  
J.G.F.F. M.J.M.*

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## BINARY PUNCH PROGRAMME (IMPROVED)

This routine provides a very simple means of obtaining a binary version of a programme. It punches out the contents of all non-zero locations (and also sequences of up to 4 zeros) from the Main Store, preceded by a Clear Store sequence. The form of the output is suitable for Binary Input, R 1031.

R 2059 supersedes R 2057: it is used in exactly the same way as R 2057 but it produces a shorter tape.

**Uses:** The entire Computing Store; B0 (but see section 2.2).

**Time:** Approximately 20 seconds (35 seconds on the 7168 store) plus 2 seconds for each block punched.

## 1. Method of Use

- 1.1 Run the Clear Store routine, R 2901.
- 1.2 Read the programme(s) to be binary punched.
- 1.3 Insert R 2059 in the main tape reader.
- 1.4 *Tear off previous output tape and depress handswitch 0 to suppress optional punching.*
- 1.5 START and RUN.

## 2. Output

2.1 The first part of the R 2059 tape is copied on to the front of the output tape by an N-sequence. This consists of a heading:

N	(On the 7168 store the name is preceded by
MK2 PUNCHED ON	7168)
<i>date punched</i>	
D	

followed by a programme which will clear Main Store locations B1.0 to B511.5 inclusive (B1.0 to B895.5 on the 7168 store) when this new tape is read in later.

2.2 Next follows a binary punch of B0. B0 is then used by R 2059, and all non-zero words in the Store between B1.0 and B511.5 (or B895.5) inclusive, are binary punched. Each binary punched section is preceded by 0.2 inches of blank tape, directive A0, binary Transfer Address, and terminated by J522.0+ (or 906.0+), Checksum. To reduce the number of such sections, and hence shorten the tape, single zeros and sequences of up to four consecutive zeros in between parts of the programme are punched out together with all non-zero words.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
27.7.56.

**DOUBLE LENGTH INPUT**

Reads a list of double length integers or fractions terminated by L from tape, and stores them in consecutive pairs of main store locations.

**Name:** D.L. INPUT A

**Store:** 10 blocks.

**Uses:** U0, 1; 5.0, 5.7; B0.  
The accumulators are preserved in B0 and restored on exit.

**Cue:** 01. The routine is self-preserving and may be re-entered at 0.0.

**Link:** Obeyed in 1.5 and left unchanged in X1. On reading the warning character L the accumulators are restored and the link obeyed.

**Form of Punching**

All numbers must commence with + or - and be terminated by space or CR LF.

Fractions may be punched with or without 0 before the decimal point.

Erase is ignored everywhere except between CR and LF.

LF, space,  $\phi$  and CR LF are ignored between numbers.

The list of numbers to be read is to be terminated by  $\lambda$  L  $\phi$ .

**Acceptable Numbers****a) Integers**

Integers must lie in the range

$$-2^{76} \leq x \leq +2^{76} \quad (2^{76} \doteq 7.5 \times 10^{22})$$

$+2^{76}$  is treated as  $(2^{76} - 1)$ . Otherwise there is no error.

**b) Fractions**

Up to 23 digits after the decimal point are accepted. Any further digits are ignored.

-1.0 and +1.0 are accepted.

+1.0 is treated as  $+(1 - 2^{-76})$ . Otherwise the maximum error is about  $\pm 13/24 \cdot 2^{-76}$ .

**Transfer Address:**

The numbers read in by R 100 are stored in consecutive pairs of main store locations starting at the address specified in 5.7<sub>m</sub> on entry to the subroutine.

This transfer address is stepped on two places for each number read.

**Error Stops:**

The following loop stops are encountered on overflow and unacceptable characters:-

- a) In 0.2+ on overflow or unacceptable characters in a number.
- b) In 0.4 if CR after a number is not followed by LF.
- c) In 0.4+ if . encountered while a sign is being sought.
- d) In 1.0+ if any other unacceptable character apart from a decimal digit is read while a sign is being sought.
- e) In 1.2 if decimal digit is read while sign is being sought.
- f) In 1.2+ if  $\lambda$  is not followed by  $L\phi$ .



## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
3.12.56

**FLOATING POINT INPUT**

A subroutine to read in signed numbers and convert them to packed floating point form.

**Name:** F. P. READ

**Store:** 10 blocks.

**Uses:** UO, 1; all the accumulators except X1; and B0.0.

**Cues:** 01 (0+.0) to read in one number and leave it in accumulator 6.  
02 (0+.0+) to read in a maximum of  $5_c$  numbers and store them in consecutive locations of the Main Store starting at the address specified in  $5_m$  on entry.

**Time:** About  $74 + 5r$  milliseconds, where  $r$  is the number of characters read including sign and decimal point.

**Link:** Obeyed in 0.1 on reading L or in 1.7 otherwise. Left unaltered in X1.

**1. Warning Characters**

1.1 R 101 recognizes four warning characters, L, N, Z and Q.

1.2 The functions of N and Z are exactly analogous to those of the corresponding warning characters in the Initial Orders.

1.3 If the warning character L is read the link set in X1 is immediately obeyed and no further numbers are read.

1.4 The warning character Q must be followed on the tape by a signed integer which is treated as a *block exponent*. This is a decimal integer which is added to the exponents of all following numbers until either another Q is read or the link is obeyed. If no Q is read the block exponent is assumed to be zero.

1.5 The warning characters should be punched in exactly the same way as for the Initial Orders. i.e. The character is preceded by letter shift and followed by figure shift, erase being ignored.

1.6 A CR LF between the figure shift following a Q and the ensuing block exponent is optional, and the block exponent may be terminated either by CR LF or by a space.

## 2. Form of Punching

2.1 The following are examples of permissible ways of punching:

```
+ 247.632
- 247632
+ 0.000247632
- .000247632
```

2.2 Each number must be preceded by its sign and may be terminated either by CR LF or by a space.

2.3 Fractions may be punched with or without a zero before the decimal point.

2.4 Before the sign of any number blank tape, space, LF, CR LF and erase will be ignored.

2.5 Between the sign of any number and its terminating character only decimal digits, decimal point and erase are permitted. In the case of block exponents only decimal digits and erase are permitted.

2.6 There is no limit to the number of decimal digits which may be punched in any number. Only the first eleven or twelve significant digits will be considered in forming the argument of the resulting floating point number. Subsequent digits if after the decimal point will be ignored, but subsequent digits if before the point will increase the decimal exponent.

## 3. Floating Point Numbers.

3.1 Although R 101 on first reading in a number stores it with a decimal exponent, it finally stores it in floating binary form. i.e. Each number is finally stored in the form  $A.2^a$  where  $A$  is the normalized argument which occupies the  $(39 - n)$  most significant bits of a word including the sign digit, and  $a$  is the binary exponent. The latter is stored as  $a + 2^{n-1}$  in the  $n$  least significant bits of the same word as the argument.

3.2 The value of  $n$  is specified by a preset parameter. If no parameter list is supplied by the programmer,  $n$  will be set equal to 9 by an optional parameter list.

3.3  $a$  must lie in the range

$$-2^{n-1} \leq a \leq 2^{n-1} - 1$$

Therefore, if  $n = 9$ ,

$$-256 \leq a \leq 255$$

3.4 Zero is held as  $A = 0$  and  $a = -2^{n-1}$ , giving a null word.

## 4. Parameter List

4.1 If the preset parameter,  $n$ , is to have any value other than 9, a parameter list of the following form must be provided:-

R 0 0 -0 1
101 - 04 -
$-2^{n-1}$

## 5. Programme Parameters

- 5.1** When one number only is to be read in cue 01 is used and in this case no programme parameters are needed.
- 5.2** If  $x$  numbers are to be read (where  $x > 1$ ), cue 02 is used. For cue 02 two programme parameters are needed: the Main Store address in which the first number is to be stored must be set in  $5_m$  and a counter,  $k$ , must be set in  $5_c$ .
- 5.3** If  $k > x$  or  $k = 0$ , the warning character L must be punched after the last number to be read.
- 5.4** If  $k = x$ , the link will be obeyed after reading in  $x$  numbers, and no L is necessary.
- 5.5** When cue 02 is used to read in  $x$  numbers,  $5_m$  will be increased and  $5_c$  decreased by  $x$ .
- 5.6** Note that for cue 02  $5_m$  and  $5_c$  must not both be set as zero. If they are the subroutine will behave as if cue 01 had been used.

## 6. Loop Stops

There are five loop stops which may occur in R 101. Four of these indicate inadmissible forms of punching and the fifth occurs if the binary exponent of any number is outside the permissible range. The stops are as follows:-

- (a) In 0.7 if  $\lambda$  is followed by any character other than L, N, Z or Q (or one or more erases followed by L, N, Z or Q.)
- (b) In 1.2+ if any impermissible character is encountered while the beginning of a number or a warning character sequence is being sought. Permitted characters are:

Space, erase, LF, CR LF,  $\phi$ , +, -,  $\lambda$ .

A stop also occurs here if any warning character is not followed by  $\phi$  (or by one or more erases followed by  $\phi$ ).

- (c) In 1.4 if a wrong character appears after the sign in a block exponent. Permitted characters are:
- Erase, decimal digits, space and CR LF (the last two being terminating characters.)
- (d) In 1.5 if a wrong character appears after the sign in a number other than a block exponent. Permitted characters are:
- Erase, decimal digits, decimal point, space and CR LF. (The last two being terminating characters.)
- (e) In 1.0+ if the exponent,  $a$ , of any number goes outside the range

$$-2^{n-1} \leq a \leq 2^{n-1} - 1.$$

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2.  
4.10.56.

## READ IN TABLES OF NUMBERS

Reads in one or more sets of numbers, each set being terminated by warning character L. The first number in each set is stored in the first location of the next available block of the main store. An index is formed storing the address of the first number in each set. The routine will most often be used to read numerical tables: for convenience the sets of numbers will be referred to as *tables*.

**Name:** INPUT TABLES

**Store:** Two Blocks, including an optional parameter list.

**Uses:** B0 and the entire computing store except accumulators 1,2,3 and 4.

**Cues:** 01 To store one or more tables and add their addresses to the contents  
(0+.0) of the successive locations in an index.

02 To store only one table and add its address to the contents of a  
(0+.5) specified location.

**Time:** Adds 60 milliseconds per table to the time taken to read the numbers with the initial orders alone.

**Link:** Obeyed in 0.5 and left unaltered in X1.

### 1. The Index

An outline form of the index, without addresses, must be punched and read as part of the master-programme. After the master-programme has been read the first location of the index must contain, in the counter position, the number of tables,  $t$ , to be read. The next  $t$  locations must have zero in the modifier position; they will often have a small number in the counter position for use by an interpolation routine.

The outline index must be punched as shown on the left below. After R 102 has been used the index will be as shown on the right.

$\lambda T \phi B.P$	Location	Modifier	Counter
$+t$	$B.P$	0	$t$
$+n_1$	$B.P + 1$	$A_1$	$n_1$
$+n_2$	$B.P + 2$	$A_2$	$n_2$
$+n_t$	$B.P + t$	$A_t$	$n_t$

where  $A_1, A_2, \dots, A_t$  are the addresses of the first numbers in each table.

If entry 02 is used the number  $t$  must be omitted and the 'index' will consist of one word only.

**2. Preset-Parameters**

The address,  $B.P$ , of the first word of the index is specified by a preset-parameter. If no parameter list is provided,  $B.P$  is set at 2.0 by an optional parameter list.

If it is desired to use any value of  $B.P$  other than 2.0, a parameter list of the following form must be read.

R 0 0 -0 1
102 - 04 -
$B - P0 0.$
0

**3. Layout of the Tables**

Each table must be punched in a form suitable for reading by the initial orders and must be terminated by the warning character L.

**4. Optional Stop**

There is an optional stop in 1.5 just before reading each table. This may be useful if it is required to read tables which are on separate tapes: otherwise it may be inhibited.

**5. Transfer Address**

The address of the first table is determined by the transfer address in 5.7<sup>m</sup> on entry to R 102. This address is stepped up to the start of a new block and the first table starts in that block. The numbers are stored in successive locations in the store, except that each table starts a new block.

In most programmes it is convenient to store tables immediately after the library subroutines, or the last item of information on the programme tape. The appropriate transfer address will be left in 5.7<sub>m</sub> or entry to the programme: if it is undisturbed before entry to R 102 there is no need to set 5.7 during the programme. Another method is to set a transfer address with warning character 'T' immediately before the 'E' entering the programme.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
18.2.57.

**READ TABLES OF FLOATING POINT NUMBERS**

This subroutine is designed to read in numerical tables and store them in a form suitable for floating-point interpolation routines. The first entry in each table is stored in the first location of the next available block of the Main Store. An index is formed storing the address of the first entry in each table.

**Name:** INPUT F.P. TABLES

**Store:** 3 blocks plus 11 blocks for R 101.

**Uses:** U 0,1,2; X 2,3,4,5,6,7; B 0.0

**Cues:** 01 To store one or more tables and add their addresses to the contents (0+.0) of the successive locations in an index.

02 To store only one table and add its address to the contents of a (0+.4+) specified location.

**Time:** Adds 90 milliseconds per table to the time taken to read the numbers with R 101 alone.

**Link:** Obeyed in 2.7 and left unaltered in X1.

**1. OPTIONAL STOP**

There is an optional stop in 1.5 before reading each table. This may be useful if it is required to read tables which are on separate tapes: otherwise it may be inhibited.

**2. TRANSFER ADDRESS**

The address of the first table is determined by the transfer address in U 5.7<sub>m</sub> on entry to R 103. This address is copied into X5 and stepped up to the start of a new block; the first table starts in that block. Numbers are stored in successive locations in the Main Store, except that each table starts a new block. U 5.7 is not changed by R 103. On exit 5<sub>m</sub> will contain the next available transfer address; 5<sub>c</sub> will not be clear.

In many programmes it is convenient to store tables immediately after the library subroutines or the last item of information on the programme tape. The appropriate transfer address will be left in U 5.7<sub>m</sub> on entry to the programme: if it is undisturbed before entry to R 103 there is no need to set U 5.7 during the programme.

A different transfer address can be set by a T-sequence immediately before the E-sequence entering the programme.

### 3. THE INDEX

An outline form of the index, without addresses, must be punched and read as part of the master-programme. After the master-programme has been read, the first location of the index must contain, in the counter position, the number of tables,  $t$ , to be read. The next  $t$  locations will often have a small number in the counter position for use by an interpolation routine. This number,  $n$ , must be punched as  $+n$  for a one-way table but as  $-n$  ( $n \neq 0$ ) for a two-way table.

The outline index must be punched as shown on the left below. After R 103 has been used the index will be as shown on the right.

$\lambda T \phi B.P$	Location	Modifier	Counter
$+t$	$B.P$	0	$t$
$+n_1$	$B.P + 1$	$A_1$	$n_1$
$-n_2$	$B.P + 2$	$A_2$	$n_2$
$-n_3$	$B.P + 3$	$A_3$	$n_3$
⋮	⋮	⋮	⋮
$+n_t$	$B.P + t$	$A_t$	$n_t$

where  $A_1, A_2, \dots, A_t$  are the addresses of the first numbers in each table.

If cue 02 is used the number  $t$  must be omitted and the index will consist of one word only.

### 4. LAYOUT OF THE TABLES

Each table must be punched in a form suitable for reading by R 101 and must be terminated by the warning character L. In a one-way table the first two numbers are stored as fixed-point integers. In a two-way table the first four numbers are stored as fixed-point integers. The remaining numbers in each table are stored in floating-point form: the number of digits in the exponent is determined by the preset parameter of R 101.

A more detailed description of the layout of the tables will be found in the specifications of the floating-point interpolation subroutines with which they are to be used.

### 5. PRESET PARAMETERS

The address,  $B.P$ , of the first word of the index is specified by a preset parameter. If no parameter list is provided,  $B.P$  is set at 2.0 by an optional parameter list.



If it is desired to use any value of  $B.P$  other than 2.0, a parameter list of the following form must be read.

R 0 0 -0 1
103 - 04 -
$B - P0 0.$
0

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
27.7.56.

**DOUBLE LENGTH INPUT**

This routine reads from tape a number punched in the form

Integer . Decimal

and stores the integral part in X6 and the fractional part in X7.

**Name:** D.L. INPUT B

**Store:** 5 blocks.

**Uses:** U0, 1; X 1, 6, 7.

**Cue:** 01.

**Link:** Obeyed in 1.2 on reading CR LF after a sign has previously been read.

**Time:** 44 ms + 3 ms per digit of integral part of number + 6 ms per digit of fractional part of number.

**Form of Punching**

1. All numbers must commence with + or - and be terminated by CR LF.
2. This subroutine is primarily intended for the input of numbers in an unscaled form, although pure fractions and pure integers are also accepted.

Examples of acceptable numbers:-

+ 1 2 3 . 4 5 6 C R LF

- 4 5 6 7 C R LF

+ . 7 6 5 4 C R LF

3. Before the sign the subroutine will ignore

LF, Space, Erase,  $\phi$  and CR.

4. Only decimal digits, erase and one decimal point are allowed between the sign and CR LF.

5. If more than 11 digits are read after the decimal point, those after the eleventh are ignored.

6. CR LF after a sign has been read is taken as terminating the number. The link is obeyed as soon as it has been read.

**Error Stops:**

The following loop stops are caused by misspunching and unacceptable numbers:-

- a) In U0.0+ if any character other than decimal digits, erases and one full stop is read between the sign and CR LF.
- b) In U0.1+ if a decimal digit is read before a sign has been read.
- c) In U0.2 if the final CR is not followed by LF.
- d) In U0.3 if a decimal point is read before a sign has been read.
- e) In U0.5 if any character other than LF, space, erase,  $\phi$ , CR, decimal point or a decimal digit is read before a sign has been read.
- f) In U1.5+ if the integral part of the number overflows. (The negative integer  $-2^{38}$  causes overflow).

**Author:** Mr. B. W. Gregory of Sir W. G. Armstrong Whitworth Aircraft Ltd.

FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
31.7.57.

**MODIFY R 105**

R 106 consists only of an input interlude which modifies R 105 to accept a space as a terminating character.

**Name:** D.L. INPUT C

**Store:** None. One block is temporarily occupied by the interlude during input. This block can be beyond B 127.

**Cue:**

01

+ 0
-----

**Method of Use**

1. In order to obtain the modified version of R 105, one tag calling for R 106 should be included in the master-programme. Since the cue is + 0 the address in the tag is immaterial.
2. Wherever it is required to enter the modified routine, a cue for R 105 should be called for in the usual way.
3. R 106 must not be used unless there is at least one tag calling for R 105.
4. R 106 must immediately follow R 105 on the Library Tape.

**Changes to R 105 Stops**

- a) The loop stop in 0.0+ is omitted.
- b) The loop stop 0.2 occurs if any character other than decimal digits, erases and one decimal point is read between the sign of a number and its terminating character (CR LF or space).

The other four loop stops in R 105 remain unchanged.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## REVISED DOUBLE LENGTH INPUT

R 107 combines in one routine the facilities of R 105 and R 106. It reads a number punched in the form

± Integer . Decimal

and terminated either by CR LF or by Sp, storing the integral part in X6 and the fractional part in X7.

R 107 is faster than R 105 when reading the fractional part of a number and it also produces a smaller error in the representation of the fractional part in X7.

Name: REVISED D.L. INPUT

Store: 5 blocks

Uses: U0, 1 ; X1, 6, 7.  
Plus space used by programmer's warning character routine (if any).

Cue: 01(0+.0)

Link: Obeyed in U1.2

Time: i) Mixed number or pure fraction:  
( $47+3\frac{1}{3}c$ ) ms. +  $\frac{2}{3}$  ms. extra per character of fraction.

ii) Pure integer: ( $22+3\frac{1}{3}c$ ) ms. .

$c$  is the total number of characters read.

The above times apply to the T.R.5 tape reader.

## 1. Form of Punching

1.1 All numbers must commence with + or - and be terminated either by CR LF or Sp.

1.2 This subroutine is primarily intended for the input of mixed numbers in an unscaled form, although pure fractions and pure integers are also accepted.

Examples of acceptable numbers are:-

+123.456 CR LF  
-4567 Sp  
+.7654 CR LF  
-Sp  
+1. CR LF

1.3 Before the sign the subroutine will ignore

LF, Sp, Erase,  $\phi$  and CR.

1.4 Only decimal digits, Erase and one decimal point are allowed between the sign and the terminating character.

1.5 The integral part of the number must lie in the range

$$|I| \leq 2^{38} - 1.$$

A maximum of 11 digits may appear after the decimal point: more than 11 digits will cause the link to be obeyed with the overflow indicator set and an incorrect number in X6 and 7. It is, therefore, advisable to include a test for overflow in the master programme on exit from R 107.

1.6 CR LF or Sp after a sign has been read is taken as terminating the number. The link is obeyed as soon as the terminating character has been read.

1.7 A routine to detect warning characters may be used with R 107 if required (see following section).

2. Extra Facility for Warning Characters

If it is required that certain warning characters should be recognised an entry to a "gamma-search" routine, to read and interpret the characters on tape following the letter shift, may be inserted in R 107 by punching the following sequence after the A3 on the master programme tape:-

```

C 107
X 0+.5+ - 0+.7
⊙ 7 41
B 72
U.P 7 60
0.7 0 60

```

c-27  
} Enter γ-search if λ is  
} read before + or -  
} Stop if punching error

This routine will be entered whenever a letter shift is read, provided that this does not occur in the middle of a number. It will normally read the character or sequence of characters following the letter shift on tape and interpret them as required by the programme.

The γ-search routine may be brought into any block in the Computing Store other than U0. On entry the link for R 107 will be in X1.

As an alternative to the loop stop shown above, if it is required that some other action should be taken on reading otherwise unacceptable figure shift characters the conditional jump U.P 7 60 may be replaced by U.P 0 60. The special programmer's routine may then be brought into any block in the Computing Store and will be entered with c-27 in X7.

3. Error

The error in converting the decimal number on tape to a binary number in the computer will not exceed 2<sup>-39</sup>. In R 105 this error could be as much as 2<sup>-36</sup>.

4. Stops

The following loop stops are caused by mispunching and unacceptable numbers:-

- 0.1+ 7 63 if a decimal digit is read before a sign has been read.
- 0.3 7 60 if a decimal point is read before a sign has been read.

- 0.3 7 61 if a character other than decimal digit, one decimal point, Sp, or CR followed by LF is read during a number.
- 0.5+ 0 60 if any character other than LF, Sp, Erase,  $\phi$ , CR, decimal point or decimal digit is read before a sign has been read.
- 1.5+ 0 60 if the integral part of the number overflows. (The negative integer  $-2^{38}$  causes overflow.)
- 0.7 0 60 if a  $\gamma$ -search routine is included and any character other than LF, Sp, Erase,  $\phi$ , CR, decimal point, decimal digit or  $\lambda$  is read before a sign has been read.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
16.5.57.

**INNER LOOP INTEGER READ**

This subroutine is not complete in itself and will not normally be called for by a master programme. It is intended for use by other standard library routines or possibly by programmers' subroutines.

The purpose of the routine is to read an integer from tape into X7.

**Name:** INTEREAD  
**Store:** 2 blocks.  
**Uses:** U0, 1; X4, 5, 6, 7.  
**Cues:** 01 a-order partial cue.  
02 b-order partial cue.  
**Time:** About  $5n$  milliseconds, where  $n$  is the number of characters read.

**1. FORM OF PUNCHING**

**1.1** Numbers must be punched in the form:

Sign Integer Terminating Character

**1.2** If the number is positive the sign may be omitted.

**1.3** The integer may be any number less in modulus than  $2^{38}$ , but further restrictions may be imposed by the controlling subroutine.

**1.4** The terminating character may be space or CR LF.

**1.5** Erase will be ignored everywhere except between CR and LF.

**1.6** LF, Sp, Er, CR LF may be punched between numbers.

**1.7** Blank tape may be punched between numbers provided that it is terminated by LF, Sp or CR LF. After blank tape Er will be ignored.



## 2. ERROR STOPS

0.0

0.0	4	63
10	5	41

Loop stop in 0.0 if blank tape not terminated by Sp, LF or CR LF.

0.2

0.2	6	61
5	7	01

Loop stop in 0.2 if integer  $\geq 2^{38}$  in modulus.

0.4

0.0	5	63
0.4+	5	60 6

Loop stop in 0.4+ on reading + or - in the middle of a number.

1.0

1.0	5	61
35	4	05

Loop stop in 1.0 if CR is not followed by LF.

1.3

14	5	43
1.3+	5	61

Loop stop in 1.3+ if inadmissible character on tape.

## APPENDIX

The following sections only concern programmers wishing to write a subroutine using R 112.

## A1 PRESET PARAMETERS

There is one preset parameter, 01, which is obeyed when the integer has been read. It may therefore be used to return to the programmer's subroutine. If required the parameter list should be punched as follows:

R 0 0 -0 1	}	Title
112 - 04 -		
0 72	}	Link to return to programmer's subroutine (Obeyed in 1.2)
0 60		

If no parameter list is provided the parameter will be set equal to +0 by an optional parameter list. The link may then be set by bringing block 0+ into U1 and then planting the link in 1.2.

## A2 INITIAL SETTINGS

Before entry to R 112 the following sequence of operations must be obeyed.

R 0 0 -0 2	}	Call for b-order partial cue
112 - 01 -		
R 0 3 -0 1	}	Call for a-order partial cue
112 - 01 -		
35 6 02	}	$-2^{-13} \rightarrow X6$
0 1 72		(+ cue 02) Block 0+ of R 112 to U1.
4 00	}	Set LINK for return from R 112 (omit if using preset parameter)
1.2 4 10		
0 4 00		Clear X4
0 7 00		Clear X7
1 0 72		(+ cue 01) Block 1+ of R 112 to U0
0.3 0 60		Jump to 0.3 to enter R 112

## A3 EXIT

On exit the integer read will be stored in X7.

FERRANTI LTD

PEGASUS LIBRARY SPECIFICATION

Issue 1  
20.5.57.

READ AND SCALE INTEGERS

This routine reads an integer from tape, divides it by a scale factor and leaves the answer in X1 and X7.

It is designed for reading standard tapes of numbers, such as random normal deviates, where it is important that the number of tape characters should be small.

R 113 uses R 112.

Name: INTEGER READ+SCALE

Store: 1 block, plus 2 blocks for R 112.

Uses: U0, 1; X1, 4, 5, 6, 7.

Cues:

01

0+	0	72
0.0	0	60

02 a-order partial cue.

03 b-order partial cue.

Time: About  $5n + 25$  milliseconds, where  $n$  is the total number of characters read from tape.

Link: Obeyed in 0.7. Not left in X1 on exit.

1. PRESET PARAMETER

The routine must be provided with a parameter list punched as follows:-

R	0	0	-0	1
113	-	04	-	
+m				

$m$  = scale factor by which integer is to be divided.

2. ACCEPTABLE NUMBERS

The integers punched may be any number less in modulus than the scaling factor.

### 3. FORM OF PUNCHING

The form of punching is as described in the specification of R 112.

### 4. ERROR STOPS

Punching errors will cause the error stops listed in the specification of R 112. The loop stop in 1.3+ is also used as shown below

1.3

14	5	43
1.3+	5	61

Loop stop if

- (i) Inadmissible character on tape
- or (ii) Integer  $\geq$  scaling factor.
- or (iii) OVR set on entry.

FERRANTI LTD

**PEGASUS LIBRARY SPECIFICATION**

Issue 1  
25. 7. 57.

**SHORT NUMBER READ**

A self-preserving subroutine to read a single length mixed number,  $N$ , from tape.

If there are  $m$  figures punched after the decimal point, then:-

$$p' = N.10^m.2^{-38} \quad q' = 10^m.2^{-38}$$

Thus 
$$N = \frac{p'}{q'}$$

- Name:** SHORT NUMBER READ
- Store:** 3 blocks
- Uses:** U 0, 1, 2; X 1, 5, 6, 7.

**Cues:**

01	0 + <span style="border: 1px solid black; padding: 0 2px;">0</span> 72
	0.4 0 60

- 02 a-order partial cue
- 03 b-order partial cue

**Time:** Cue 01  $5_c + 8$  milliseconds  
 Jump to 0.5  $5_c + 2$  milliseconds  
 where  $c$  is the total number of characters read from tape.

**Link:** Obeyed in 0.4. Not left in X1.

**1. Scaling**

**1.1 Mixed Numbers**

It is often convenient to store numbers in the computer with a scale factor  $2^{-n}$ . This may be arranged by obeying the following orders immediately after the subroutine:-

<div style="border: 1px solid black; border-radius: 50%; width: 20px; height: 20px; display: flex; align-items: center; justify-content: center; margin: 0 auto;"> <math>n</math> </div>	7 50
	7 6 26

$10^n.2^n.2^{-38}$  to X7

$N.2^{-n}$  to X7

n must be chosen so that:

$$| 10^n \cdot 2^n | < 2^{38}$$

$$| N \cdot 2^{-n} | < 1.0$$

$$n \leq 37$$

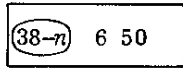
It is advisable to test the OVR after these operations.

**1.2 Fractions**

Fractions may be stored as described in section 1.1, but the shift may be omitted if n = 0.

**1.3 Integers**

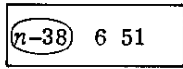
Integers may be scaled as described in section 1.1, or by the order:-



N.2<sup>-n</sup> to X6.

If n = 38, N.2<sup>-n</sup> will be left in X6 by the subroutine.

If n > 38 the following order should be used:-

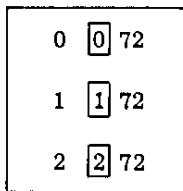


N.2<sup>-n</sup> to X6.

**2. Re-entry**

2.1 R 116 is a self-preserving subroutine, leaving its three blocks in U 0, 1, 2.

2.2 R 116 may also be brought into the Computing Store without being entered by obeying the orders



tagged by the appropriate partial cues, 02 and 03.

2.3 Once the three blocks of R 116 are in the Computing Store it can be entered by jumping to 0.5.

**3. Form of Punching**

3.1 Numbers must be punched in the form.

Sign	Integral part	•	Fractional part	Terminating character
------	---------------	---	-----------------	-----------------------

3.2 If the number is positive the sign may be omitted.

3.3 If the number is a fraction the integral part may be omitted.

- 3.4 If the number is an integer the decimal point and fractional part may be omitted.
- 3.5 The terminating character may be space or CR LF.
- 3.6 Erase will be ignored everywhere except between CR and LF.
- 3.7 LF Sp Er CRLF may be punched between numbers.
- 3.8 Blank tape may be punched between numbers provided that it is terminated by + - • LF Sp or CRLF. After blank tape Er will be ignored.

#### 4. Error Stops

- |     |  |  |
|-----|--|--|
| 2.1 | 2.2 7 63<br>2.1+ 7 61  | Loop stop in 2.1+ if two decimal points<br>are read in one number.                                       |
| 2.5 | <div style="border: 1px solid black; border-radius: 50%; width: 20px; height: 20px; display: flex; align-items: center; justify-content: center; margin-bottom: 5px;">14</div> 5 43<br>2.5+ 5 61 | Loop stop in 2.5+ if an incorrect character<br>is read.  |
| 2.7 | 0.1+ 5 60<br>2.7+ 0 60   | Loop stop in 2.7+ if CR not followed by LF<br>or OVR set on entry<br>or $m > 11$<br>or $N.10^m > 2^{38}$ |

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
6.6.57.

## SLOW MID-POINT NUMBER READ

This routine occupies only one block of the Main Store but it has several undesirable features. Routines such as R 105, 116 and 121 should normally be preferred.

R 120 reads one number from tape, multiplies it by  $2^{-19}$  and stores the result in U 5.0. It uses the Initial Orders as a subroutine.

Name: SLOW MIDPT READ

Store: 1 block

Uses: U 0, 1, 2, 3, 4, 5.0; B 0

Cue: 01

0+	2	72
2.5	0	60

Time:  $5n + 46$  milliseconds, where  $n$  is the number of characters read from tape.

Link: Obeyed in 2.5. Left in X1 on exit.

## 1. Form of Punching

1.1 Punching conventions are the same as for the Initial Orders, except that a decimal point may be punched in the middle of a number. Orders and warning-characters are not permitted.

1.2 Numbers must be punched in the form

Sign    Integral part    •    Fractional part    Terminating character

1.3 If the number is a fraction the decimal point may be punched immediately after the sign.

1.4 If the number is an integer the decimal point and fractional part may be omitted.

1.5 The number,  $N$ , must lie in the range

$$- 524288 \leq N < + 524288$$

1.6 Not more than five digits may be punched after the decimal point.



## 2. Error Stops

The usual Initial Orders loop stops will occur when tape errors are encountered, but extravagant effects may be encountered on reading  $\lambda$  (letter shift) before the start of a number.

After  $\lambda$  the Initial Orders will read the next character on the tape as a warning-character but will take the corresponding address from the handswitches. There may be some unusual optional printing, followed by a 77 stop, optional stop, loop stop, or cycling round a small loop punching blank tape. The effect of warning-character J, if HO = 0, is to enter programme at the address specified on the handswitches.

The subroutine may not be re-entered by operating the START key.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
20.5.57.

## GENERAL SINGLE-LENGTH NUMBER READ

A self-preserving subroutine to read a single-length number,  $N$ , from tape. If there are  $m$  figures punched after the decimal point, then

$$p' = N.10^m.2^{-38} \text{ and } q' = m.2^{-38}$$

A wide variety of forms of punching are permissible including many of those produced by R 1.

Name: NUMBER READ A

Store: 5 blocks.

Uses: U 0, 1; X 6, 7; B0. Also X1 if a Computing Store link is used.

Cues: 01

0+	0	72
0.5+	0	60

02 a-order partial cue.

03 b-order partial cue.

Time: About  $54 + 5n$  milliseconds, where  $n$  is the total number of characters read before and during the number.

Link: The link must be set in X1. It may be a *go* order pair in which case it will be planted and obeyed in U 1.7 and left unaltered in X1. Alternatively it may be a Computing Store link set with a *40* order. In this case it will be obeyed in 0.4 and X1 will be altered.

Error: None.

## 1. FORM OF PUNCHING

1.1 Negative numbers must be preceded by a minus sign; positive numbers can but need not begin with a plus sign. Fractions may be punched with or without a zero before the point.

1.2 Numbers must be terminated by CR LF or by *two* consecutive spaces. Erase will be ignored between the two spaces but not between CR and LF.

1.3 Before the commencement of a number (+, -, digit or decimal point) the following characters will be ignored:-

LF, space, erase, CR LF.

1.4 Blank tape ( $\phi$ ) will also be ignored except that between any  $\phi$  and the first digit of the next number at least one of the following must be read:-

+, -, ., LF, space, CR LF.

1.5 Any number of spaces may be punched between the sign of a number and the following digit or decimal point providing that no character other than erase interrupts the sequence.

1.6 Single internal spaces in a number will be ignored. This allows many of the forms of punching of R 1 to be acceptable as input data for R 121.

1.7 Apart from a sequence of spaces after the sign and single internal spaces, only decimal digits, erases and one decimal point are accepted between the commencement of a number and its terminating characters.

## 2. SIZE OF NUMBERS

The number of digits punched in any number excluding any non-significant left-hand zeros but including all right-hand zeros must not exceed 12 and, if there are 12, the first must be 1 or 2.

This is because all numbers must be such that

$$-2^{38} < N. 10^m < 2^{38}$$

and  $2^{38} = 274, 877, 906, 944.$

## 3. RE-ENTRY

3.1 R 121 is a self-preserving subroutine, leaving its own block 0+ in U 0.

3.2 Block 0+ of R 121 can also be brought into U 0 without being entered by using the order

0	0	72
---	---	----

tagged by the appropriate partial cue, 02 or 03.

3.3 Once block 0+ has been brought into U 0 by this method or by previous use of the ordinary cue, 01, the subroutine can be entered by jumping to 0.5+.

## 4. CONDITIONS ON EXIT

R 121 leaves the numbers  $N.10^m.2^{-38}$  and  $m.2^{-38}$  not only in X 6 and 7 but also in U 1.0 and 1.1 respectively.

## 5. ERROR STOPS

5.1 The following loop stops will occur if unacceptable numbers or characters are read:

- a) In 0.2 if the number is too big. (i.e. If  $|N.10^m| \geq 2^{38}$ ).
- b) In 0.7+ if any character other than +, -, ., LF, space, erase,  $\phi$ , CR or a decimal digit is read or if CR is not followed by LF before the commencement of a number.

- c) In 1.1 if more than one decimal point is read in any number.
- d) In 1.2+ unless +, -, ., LF, space or CR LF is read between any blank tape and the first digit of a number.
- e) In 1.5+ if any character other than decimal digits, decimal point, space, erase or CR is read or if CR is not followed by LF after the commencement of a number.

5.2 If the subroutine is entered with the OVR set, writing with overflow will occur in 0.6 due to a block write order in 0.5+. The Order Register at this stage will contain:-

1+	1	72
1.3	5	00

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

*Issue 1***NUMBER FROM HANDSWITCHES**

A self-preserving subroutine to read into X6 a signed fraction or integer tapped out on the handswitches.

**Name:** NO. FROM HS.

**Store:** 5 blocks.

**Uses:** U0; X6; B0.

**Cues:** 01 (0+.2+) Each character is printed as it is read.  
02 (0+.2) No printing.

**Link:** Set in X1. It may be either

- (a) a "go" order pair (which will be obeyed in 0.1)
- or (b) a Computing Store Link (only the 7 l.s. bits and the + sign are then examined).

In both cases the link will be left in X1 on exit.

**1. Significance of Handswitches**

When numbers are tapped out with this subroutine, the keys have the following significance:

H 0 to H9	Decimal digits 0 to 9	
H10	Plus sign	
H11	Minus sign	
H12	Decimal point	
H13	CR LF	} These terminate the number
H14	Space	
H15 to H19	Cancel all characters read since entry	

When using cue 01, H15 to H19 cause the printing of Er,  $\phi$ , >,  $\geq$  and \* respectively.

Note the correspondence between the handswitch number and the tape code (via register 16).

**2. Method of Use**

On entry there is an optional stop in 0.4, which is repeated in a loop if the handswitches are not clear. The handswitches should be cleared, the RUN key operated and then the number tapped out as described in Section 3.

If optional stops are inhibited the routine will loop in 0.0 or 0.4 according to whether or not the handswitches are clear (see Section 4). Clear the handswitches and tap out the number as described below.

### 3. Tapping Conventions

The number is tapped out in accordance with the usual Initial Orders conventions for punching tape, with the following exceptions:

3.1 H13 corresponds to CR LF.

3.2 If any of H15 to H19 is operated, there is a 77 stop in 0.0 and all previous input is cancelled, i.e. ineffective. This permits a fresh start to be made.

3.3 The sign may be tapped out at any time (i.e. before, after or in the middle of the digits); if there is no sign the effect is similar to tapping out a plus sign; if several signs are tapped out, only the last one is effective.

3.4 The number is terminated and the link obeyed by either H13 (CR LF) or H14 (space), both of these characters are ignored until a digit or sign has been tapped out. These characters are also ignored between a decimal point (not preceded by a digit or sign) and a digit or sign.

3.5 If cue 01 is used, each digit is printed as it is tapped out (H13 causing printing of CR LF); it is recommended that H19 should be used for "cancel" as this causes the printing of an asterisk.

### 4. Stops

0.0	0 0 77
	0.1 1 10

77 stop if H15 to H19 operated. Clear handswitches and RUN to start tapping number again.

0.0	15 4 00
	0.0 4 60

Wait for first character if handswitches clear on entry and optional stops are inhibited.

0.4	15 4 00 0.
	0.4 4 61

Optional stop on entry. Wait for handswitches to be cleared.

0.6	0.6 6 61
	① 6 40

Loop stop if two decimal points read.

0.7	0.0 0 64
	0.7+ 0 60

Loop stop if OVR set when the number is terminated; intermediate OVR is ignored and may be cancelled.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
29. 7. 56.

**£.S.D. INPUT TO PENCE**

This subroutine reads from tape a sum of money not exceeding £999,999.999.19.11 punched as  $a.b.c$  where  $a$  is £,  $b$  is shillings and  $c$  an integral number of pence and stores it in X7 as  $240a + 12b + c$

$a, b, c$  must be integers such that:-

$$0 \leq a < 10^9$$

$$0 \leq b \leq 19$$

$$0 \leq c \leq 11$$

**Name:** £.S.D. INPUT (PENCE)

**Store:** 4 blocks

**Uses:** U0, 1; X7. The accumulators are preserved in B0 and, apart from X7, restored on exit.

**Cue:** 01

**Link:** Obeyed in 1.7 and left unchanged in X1.

**Time:** Approximately  $5n + 50$  milliseconds, where  $n$  is the total number of characters to be read including full stops, spaces, erases and terminating characters.

**Notes:****1. Terminating Characters**

The first CR LF encountered after the second full stop is taken as the termination of the whole sum of money to be read.

A space will also be regarded as the terminating character of the sum if and only if at least one decimal digit has been read after the second full stop.

i.e. If  $c$  is zero and the terminating character is to be a space, then  $c$  must be punched as 0 or one or more spaces followed by 0. Whereas if  $c$  is zero and the terminating character is to be CRLF,  $c$  may be punched as just one or more spaces. ( $a$  and  $b$ , if zero, can always be punched as just one or more spaces if desired).  $a$  and  $b$  must be terminated by full stops.

**2. Erase**

Erase will be ignored anywhere except between CR and LF.

### 3. Spaces

Spaces at the more significant end of  $a$ ,  $b$  or  $c$  will be ignored. Spaces among the decimal digits of  $a$  or  $b$  or between the last decimal digit of  $a$  or  $b$  and the following full stop will be treated as errors causing a loop stop. A space after any decimal digit of  $c$  has been read will be treated as a terminating character.

4. Apart from decimal digits, any character other than those described above will be treated as an error causing a loop stop.

### 5. Loop Stops

The following is a full list of the loop stops caused by the various possible cases of incorrect or impermissible punching:-

- a) In 0.1 if  $a \geq 10^9$
- b) In 0.5+ if  $b \geq 20$
- c) In 1.0+ if impermissible character in  $c$
- d) In 1.1+ if spaces in  $a$  or  $b$  after any decimal digits have been read.
- e) In 1.2 if CR at end not followed by LF
- f) In 1.3+ if  $c \geq 12$
- g) In 1.6+ if impermissible character other than wrongly placed space in  $a$  or  $b$ .



## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
20.6.57.

**SIGNED £.S.D.READ TO PENCE**

A self-preserving subroutine to read from tape a sterling sum punched as  $a.b.c$ , where  $a$  is pounds,  $b$  is shillings and  $c$  is an integral number of pence. The number is left in X1 as  $240a + 12b + c$ .

The sum must not exceed £1,145,324,612.5.3d. in modulus.

**Name:** SIGNED £.S.D.READ

**Store:** 6 blocks.

**Uses:** U 0, 1; X1; B0.

**Cues:**

01

0+	0	72
0.5+	0	60

02 a-order partial cue.

03 b-order partial cue.

**Time:** About  $5n + 65$  milliseconds, where  $n$  is the total number of characters read; except that any erase read during, but not before, the number will add about 15 milliseconds.

**Link:** The link must be set in X6 where it will be left unaltered on exit. It may be a *go* order pair which will be planted and obeyed in U1.7. Alternatively it may be a Computing Store link set with a *40* order, in which case it will be obeyed in U0.4+.

**Error:** None.

**1. FORM OF PUNCHING**

**1.1** Negative sums of money must be preceded by a minus sign; positive sums can but need not begin with a plus sign. If  $a$  is zero and the whole number is positive, the number can commence with a full stop.

**1.2** The first CR LF encountered after the second full stop is taken as the termination of the sum of money to be read. A space after the second full stop will also be regarded as a terminating character providing that at least one decimal digit has been read in  $c$ .

1.3 If *a* or *b* is zero it may, if desired, be punched as one or more spaces or omitted altogether. This also applies to *c* if the terminating character is CR LF, but not if it is a space. *The two full stops must always be punched.*

1.4 Before the commencement of a sum of money (+, -, digit or full stop) the following characters will be ignored:-

LF, space, erase, CR LF.

1.5 Blank tape ( $\phi$ ) will also be ignored except that between any  $\phi$  and the next decimal digit at least one of the following must be read:-

+, -, full stop, LF, space, CR LF.

1.6 Any number of spaces may be punched in the following positions:-

- i) Before the first digit of *a* (both before and after the sign if any),
- ii) Between the first full stop and the first digit of *b*,
- iii) Between the second full stop and the first digit of *c*.

1.7 Apart from spaces in these places, only decimal digits, erases and two full stops are accepted between the commencement of a sum of money and its terminating character.

**2. RE-ENTRY**

2.1 R 142 is self-preserving, leaving its own block 0+ in U0 on exit.

2.2 Block 0+ of R 142 can also be brought into U0 without being entered by using the order:-

0 0 72

tagged by the appropriate partial cue, 02 or 03.

2.3 Once its block 0+ is in U0 the subroutine can be entered by jumping to U0.5+.

**3. SIZE OF NUMBERS**

Disregarding its sign, no sum of money may exceed £1,145,324,612.5.3d. i.e. Each sum must satisfy:

$$|240a + 12b + c| \leq 2^{38} - 1$$

**4. ERROR STOPS.**

**4.1 Loop Stops**

- a) In 0.2 (0.2 5 60 3) if CR is read between the commencement of a number and the second full stop.
- b) In 0.2+ (0.2+ 0 60) if any inadmissible character apart from a third full stop or a misplaced CR is read after the commencement of a number. This includes the case of CR not followed by LF at the end of the sum.

- c) In 0.3 (0.3 6 61) or in 0.5+ (0.5+ 0 60) if the sum of money exceeds the size specified in section 3.
- d) In 0.7+ (0.7+ 0 60) if any inadmissible character apart from a decimal digit following blank tape is read before the commencement of a number. This includes the case of CR not followed by LF.
- e) In 1.2+ (1.2+ 5 61) unless +, -, full stop, LF, space or CR LF is read between any  $\phi$  and the next decimal digit.
- f) In 1.4 (1.4 0 60) if a third full stop is read.

#### 4.2 Writing with Overflow.

If the subroutine is entered with the OVR set, writing with overflow will occur 0.6 (due to a block write order in 0.5+)

0.6 will contain

1+	1	72
0	2	00

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1

## MIXED RADIX INPUT

A self-preserving subroutine to read from tape a positive quantity expressed in units of different denominations. The quantity is converted into an integral number of units of the lowest denomination and placed in X7.

For example, a sterling amount punched as

$$a.b.c$$

where  $a$ ,  $b$  and  $c$  are integers representing pounds, shillings and pence, may be read and converted into pence. The subroutine will also read single integers.

Name: MIXED RADIX INPUT

Store: 3 blocks.

Uses: U0, 1, 2; X1, 6, 7. If an integer is read X6 is unused.

Cues: 01 (0+.7) Normal entry.  
02 a-order partial cue.  
03 b-order partial cue.

Time: Approximate times in milliseconds for the jump entry are:-

$$3\frac{1}{2}(c + d + 2) \text{ for T.R.5 (300 characters per second)}$$

$$5c + 2d \text{ for T.R.2 (200 characters per second)}$$

where  $c$  is the total number of characters read and  $d$  is the number of decimal points read. An extra 6 milliseconds should be added for entry by cue 01.

Link: Obeyed in 0.7; not left in X1 on exit.

## 1. Entry and Re-entry

1.1 Before entry the radices  $r$  and  $s$  must, if required, be set in X6 and X5 (see section 2).

1.2 R 143 is a self-preserving subroutine, leaving its 3 blocks in U0, U1 and U2.

1.3 It may be entered in the normal way by cue 01. Alternatively, its 3 blocks may be read into the Computing Store without being entered by the transfer orders:

0	0	72	+ partial cue
1	1	72	+ partial cue
2	2	72	+ partial cue

tagged by the appropriate partial cues.

1.4 When R 143 is in the Computing Store, it may be entered repeatedly by the jump order

1.0 0 60

provided that the contents of U0, 1 and 2 have not been overwritten since the last entry.

2. Method of Use

2.1 The subroutine may be used to read:-

2.1.1 A positive integer *a* (punched with or without sign). The integer will be left in X7.

2.1.2 A positive quantity punched in the form

$$a.b$$

where *a* and *b* are decimal integers. The integer

$$ar + b$$

will be left in X7, where *r* is a positive integer ( $0 \leq r < 2^{38}$ ) placed in X6 before entry.

2.1.3 A positive quantity punched in the form

$$a.b.c_1.c_2. .... .c_n$$

where *a*, *b*, *c*<sub>1</sub>, ..., *c*<sub>*n*</sub> are *n* + 2 unsigned decimal integers separated by *n* + 1 points. The integer

$$ars^n + bs^n + \sum_{i=1}^n c_i s^{n-i}$$

is left in X7, where *r* and *s* ( $0 \leq r, s < 2^{38}$ ) are positive integers placed in X6 and X5 respectively before entry.

For example, if a quantity

$$a.b.c$$

is read, the integer

$$ars + bs + c$$

will be left in X7.

2.2 The integer to be placed in X7 in each case must lie in the range

$$0 \leq x < 2^{38}.$$

3. Form of Punching

3.1 Each quantity should be punched as a set of integers separated by points. A + sign may (but need not) be punched before the quantity (i.e. before the first integer).

3.2 The quantity should be terminated by Sp or CR LF. An indication of which of these has been used is left in X1 (see section 4).

3.3 CR LF, LF or Sp may be punched before the sign or the first digit of the amount.

3.4 Blank tape may be punched between quantities provided that it is terminated by CR LF, LF, Sp or +. After blank tape Er will be ignored.

3.5 No characters other than decimal digits, point or erase may be punched between the sign or first digit of the quantity and its terminating character.

3.6 Erase may be punched anywhere except between CR and LF.

#### 4. Contents of X1

X1 will be clear on exit from R 143 if the quantity read was terminated by CR LF; it will contain (0.1, 0) if the terminating character was Sp.

#### 5. Loop Stops

- |                  |   |
|------------------|---|
| 0.3+ (0.3+ 1 61) | If an inadmissible character is read.   |
| 0.5 (0.5 1 61)   | If CR is not followed by LF.  |
| 1.7 (1.7 0 60)   | If an inadmissible character is read after blank tape, or the quantity is outside the range $0 \leq x < 2^{38}$ , or $r, s$ negative. |

Author: Mr. G.H.L. Buxton of Whitworth Gloster Aircraft Ltd.

*This specification is distributed with the kind permission of Whitworth Gloster Aircraft Ltd. by Ferranti Ltd. as part of its service to users of Ferranti Pegasus Computers.*

*The specification must not be reproduced in whole or in part without the permission of Whitworth Gloster Aircraft Ltd.*

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
3.10.58

## INPUT £. s. d. TABLES TO PENCE

A self-preserving subroutine which reads from tape one or more sets of sums of money, each set being terminated by warning character L. Each sum of money is punched as *a. b. c* where *a* is pounds, *b* is shillings and *c* is an integral number of pence. The first sum in each set is stored in the first location of the next available block of the Main Store. An index is formed storing the address of the first number in each set. In what follows the sets will be referred to as tables.

Name: READ £.S.D. TABLES

Store: 3 blocks + 6 blocks for R 142

Uses: U0, 1,2; X1, 4, 5, 6, 7; B0.

Cues:

01	0+ <span style="border: 1px solid black; padding: 0 2px;">2</span> 72
	2.2 0 60

02 a-order partial cue

03 b-order partial cue

Time: About  $(5n + 74)$  milliseconds for each sum of money where  $n$  is the number of characters read in a sum, plus an extra 32 milliseconds for each table.

Link: Obeyed in 2.0 and not left in X1 on exit.

## 1. FORM OF PUNCHING

Individual sums of money must be punched as described in the specification of R 142, sections 1 and 3. Each table must be terminated by a warning character L.

## 2. OPTIONAL STOP

There is an optional stop in 2.6 before reading each table. This may be useful if it is required to read tables which are on separate tapes: otherwise it may be inhibited.

## 3. TRANSFER ADDRESS

The address of the first table is determined by the transfer address in U5.7<sub>m</sub> on entry to R 148. This address is copied into X1 and stepped up to the start of a new block; the first table starts in that block. On exit U5.7<sub>m</sub> will contain the next available transfer address.

In many programmes it is convenient to store tables immediately after the library subroutines or the last item of information on the programme tape. The appropriate transfer address will be left in U5.7<sub>m</sub> on entry to the programme: if it is undisturbed before entry to R 148 there is no need to set U5.7 during the programme.

A different transfer address can be set by a T-sequence immediately before the E-sequence entering the programme.

#### 4. PRESET PARAMETERS

The address  $B.P$  of the first word of the Index is specified by a preset parameter. If no parameter list is provided,  $B.P$  is set at 2.0 by an optional parameter list.

If it is desired to use any other value than 2.0, then a parameter list of the following form must be read

R 0 0 -0 1	Title of Parameter List
148 - 04 -	
$B - P0 0.$	Address of Index
0	

#### 5. RE-ENTRY

R 148 is self-preserving, leaving its own block 0+ in U2 on exit.

Block 0+ of R 148 can also be brought into U2 without being entered, using the order 0 2 72 tagged by the appropriate partial cue 02 or 03. The subroutine may then be entered by jumping to U2.2.

#### 6. ERROR STOPS

- |      |   |  |
|------|---|--|
| 0.2  | (0.2 5 60 3)  | If CR is read between the commencement of a number and the second full stop.   |
| 0.2+ | (0.2+ 0 60)   | If any inadmissible character apart from a third full stop or a misplaced CR is read after the commencement of a number. This includes the case of CR not followed by LF at the end of a sum.                |
| 0.3  | (0.3 6 61)  | If the sum of money is too large.  |
| 0.5+ | (0.5+ 0 60)   |  |
| 1.0+ | (1.0+ 7 61)   | If any inadmissible character apart from a decimal digit following blank tape is read before the commencement of a number. This includes the case of CR not followed by LF, and $\lambda$ not followed by L. |
| 1.4  | (1.4 0 60)  | If a third full stop is read.  |
| 2.4  | ( <span style="border: 1px solid black; padding: 0 2px;">2</span> 7 71) | Writing with OVR if OVR set on entry.  |

© FERRANTI LTD 1958



FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
12.12.57.

**SET DATE**

This routine reads the date from tape and stores it in B511.7. It also sets the serial number to zero in B511.6. These are then ready for punching by the warning character D.

**Name:** SET DATE

**Store:** B1

**Uses:** The entire Computing Store; B511.6 and B511.7.

**Entry:** By E1.0 at the end of the tape.

**Method of Use**

1. First punch 8 characters on tape to represent the date. If the date consists of less than 8 characters the tape should be completed by punching 1 or 2 minus signs as shown below:-

12/12/57  
7/12/57-  
1/1/58--

The date should be preceded by blank tape and will usually be followed by a warning character Z. Thus a complete tape would be punched

Blank Tape 7/12/57- CR LF  $\lambda$  Z  $\phi$

2. Insert R 2150 in the main tape reader START and RUN.

3. When R 2150 reaches a 77 stop (E1.0), replace it by the tape containing the date. RUN to read it in.

4. When the date tape is read the 8 characters after blank tape are packed into B511.7. B511.6 is cleared and the Initial Orders START sequence is entered to read the rest of the tape. The serial number may be set equal to  $n$  by punching a sequence

T 511.6  
+ $n$

after the date. The next serial number punched will then be  $n+1$ .

**Author:** Mr. C. Strachey of the National Research and Development Corporation.

Ferranti Ltd.,  
London Computer Centre,  
21, Portland Place,  
LONDON, W.1.

Copyright Reserved

Issue 1  
12th December, 1957  
C.S. W.F.M.P.

FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

*Issue 1***READ TRANSACTOR CARD**

A subroutine to read a card in the transactor and store the information in 25 words of the Main Store. R 195 is designed to be used with the transactor as at present connected at the Ferranti Computer Centre in London. The transactor would be connected in a different way on another Pegasus.

For further information on the programming details for the transactor, reference should be made to List CS.262.

**Name:** READ TRANSACTOR CARD

**Store:** 3 blocks.

**Uses:** U0, 1, 2, 5; X1, 5, 6, 7; 25 locations in the Main Store.

**Cue:** 01 (0+.0).

**Time:** Approximately 4½ seconds.

**Link:** Obeyed in 2.2; not left in X1 on exit.

**1. Method**

**1.1** When the presence of a card has been sensed by the routine, the information from that card is read. It is stored in 25 consecutive Main Store locations, designated words 0 to 24. Word 1 does not contain any information from the card, but holds the "transactor number".

**1.2** The diagram shows the layout of the card and indicates the order in which the "words" on the card are read. Word number  $n$ , containing  $m$  bits is represented in the diagram by  $\textcircled{n} \rightarrow m \rightarrow \boxed{n}$ . In every case, the bit  $\textcircled{n}$  is placed in D0 of a Pegasus word (i.e. the sign bit), and succeeding bits are placed in D1, D2, ..... of the same word: all further bits in that word are made zero.

**1.3** A mark on the card is represented by a 1 in the corresponding bit in the store.

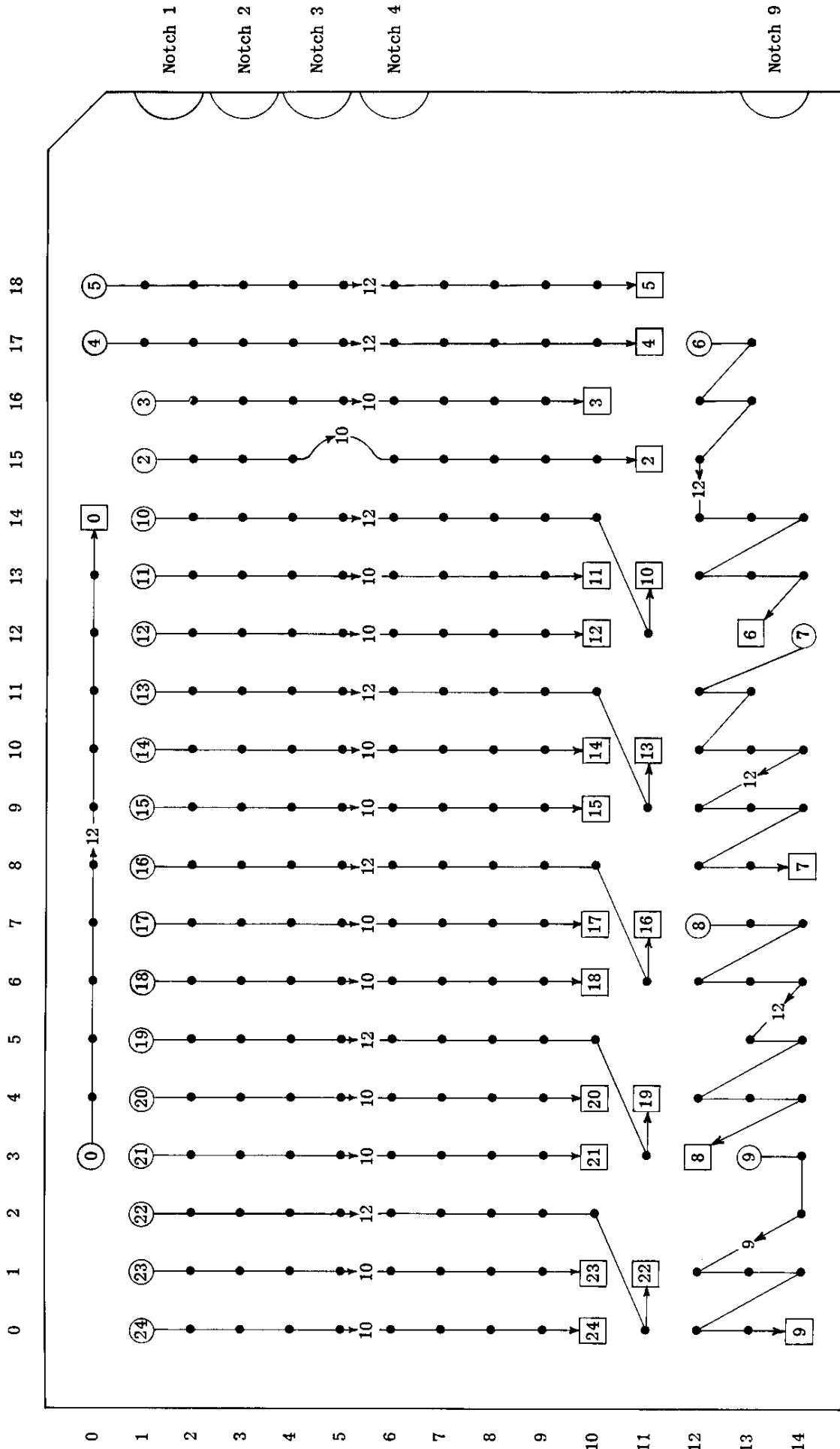
**1.4** A notch cut in the notch 1 position is represented by a 1 in D10 and a 0 in D11 of word 2; no notch is represented by a 0 in D10 and a 1 in D11.

**1.5** On exit, the last 9 words are left in the Computing Store as shown below, as well as being stored in the Main Store:

Words 16 to 23	in	U5.0 to 5.7
Word 24	in	X1 and X7

The Main Store address of word 16 is left in  $5_m$ .

**1.6** On exit from R 195 the main tape reader and the main paper tape punch will always be selected.



## 2. Preset Parameter

2.1 The Main Store block to which the first word is to be written must be specified by preset parameter 01. The first word from the transactor will be stored in the first word of the block specified. The parameter list should be punched as follows:

R 0 0 -0 1
195 - 04 -
A - 00 0.
0

A.0 is the Main Store address of the first word.

2.2 If no parameter list is supplied, the address is set equal to 512.0 by an optional parameter list.

## 3. Stop

If R 195 is entered when there is no card in the transactor, the programme will loop between 0.1 and 0.2. As soon as a card is inserted, the programme will sense the presence of the card which will then be read. A card released by the transactor but not removed from the card slot will not be read by R 195. The card must be completely removed from the slot before the insertion of it or a different card will cause R 195 to read the card.

© FERRANTI LTD 1960

*Not to be reproduced in whole or  
in part without the prior written  
permission of Ferranti Ltd.*

FERRANTI LTD

PEGASUS LIBRARY SPECIFICATION

Issue 2.  
8.10.56.

SQUARE ROOT

$$p' = \sqrt{p + 2^{-38} q}$$

Name: SQ. ROOT

Store: 1 block

Uses: U0; X5, 6, 7

Cue: 01

Time: The time taken increases as  $p + 2^{-38} q$  decreases. The following table shows the time in milliseconds for various values of  $p + 2^{-38} q$

$p + 2^{-38} q$	Time (milliseconds)
0.9	26
0.75	34
0.5	42
0.25	42
0.1	49
0.0001	88
$10^{-9}$	151
0	297

If  $p$  has its maximum value of  $1-2^{-38}$  then the answer is taken to be  $1-2^{-38}$  and the time taken is 3 milliseconds.

Link: Obeded in 0.7 and left undisturbed in X1.

Error: Not greater than  $2^{-39}$

Note: If  $p$  is negative a loop stop will occur in 0.0.

If  $p = 0$  and  $q < 0$  the subroutine will not stop but will produce a meaningless result.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
30.4.56.

## CUBE ROOT

Evaluates the cube root of the double-length number  $p + 2^{-38}q$ , and leaves the result in 6.

Name: CUBE ROOT

Store: 2 blocks

Uses: U0,1; X5,6,7,

Cue: 01

Time: An average time would not be very meaningful, as the time varies very considerably with the size of  $(pq)$ . However, the following should be a sufficient guide.

$(pq) = 0.5$	Time = 73 ms
" = 0.1	" = 99 "
" = 0.01	" = 123 "
" = 0.001	" = 161 "
" = 0.000001	" = 250 "

Link: Is obeyed in 1.4 and left undisturbed in X1

Notes: 1. The maximum error is about  $2^{-26}$ , occurring with very small numbers. For  $(pq)$  greater in magnitude than  $10^{-10}$  the error will usually be substantially less than this.

2. If  $p = 1 - 2^{-38}$  the answer is taken to be  $1 - 2^{-38}$ .

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## FAST SQUARE ROOT

$$p' = \sqrt{(pq)}$$

where  $(pq)$  is the double length number in X6 and 7. R 202 is self-preserving in U0 and 1. It uses one block more but is much faster than R 200.

**Name:** FAST SQ.ROOT

**Store:** 2 blocks

**Uses:** U0,1; X4,5,6,7.

**Cues:** 01 (0+.0) Main entry  
02 a-order partial cue  
03 b-order partial cue

**Re-entry:** If U0 and 1 are left undisturbed after using cue 01 the routine can be re-entered by jumping to U0.1.

The routine can be brought into U0 and 1 without being entered by obeying the order pair

0	0 72	+ partial cue 02 (or 03 if b-order)
1	1 72	+ partial cue 03 (or 02 if a-order)

tagged by the appropriate partial cues.

**Time:** Entry at 0.1:  $27 + \frac{3}{16} n$  milliseconds ( $-1 \leq n \leq 76$ ) where  $n$  is such that  $\frac{1}{4} \leq (pq).2^n < \frac{1}{2}$ , or  $n = 76$  if  $(pq) = 0$ .

The time must be between 27 and 42 milliseconds.

Entry by cue 01 will add 3 milliseconds to the subroutine time, plus the time to obey the cue itself.

**Link:** Obeyed in U1.7 and left unaltered in X1.

**Error:** If  $(pq) > \frac{1}{16}$ , the error will not exceed  $2^{-38}$ .

If  $(pq) \leq \frac{1}{16}$ , the error will not exceed  $2^{-39}$ .

**Stop:** In U0.1

0.1	6 63	Loop stop if $p < 0$ .
1.7	1 10	

**Note:** The sign bit of  $q$  is ignored, and the number  $(pq)$  is assumed to be a proper double length number with  $q \geq 0$ .

**Author:** Mr. J.G.F. Francis of the National Research Development Corporation.



FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## FLOATING-POINT CUBE ROOT

$$p' = \sqrt[3]{p}$$

where  $p$  and  $p'$  are floating-point numbers.

**Name:** F.P. CUBE ROOT.  
**Store:** 3 blocks.  
**Uses:** U0, 1, 2; X1, 5, 6, 7.  
**Cue:** 01 (0+.0)  
**Time:** Between 103 and 149 milliseconds.  
**Link:** Obeyed in 2.2, not left in X1 on exit.

## Preset Parameter

R 211 works in single-length floating-point arithmetic as described in the Pegasus Programming Manual and in the specifications of some floating-point subroutines (for instance R 610).

The value of  $n$ , i.e. the number of binary digits used to represent the exponent part of the floating-point numbers, is specified by a present parameter. The parameter list should be of the form

R 0 0-01
211 -04-
$-2^n$

If no parameter list is supplied  $n$  will be set equal to 9 by an optional parameter list.

## Error

The error will not exceed one in the last binary place of the argument if  $p > 0$ , or two in the last binary place if  $p < 0$ .

If  $p$  is in standard floating-point form, the answer,  $\sqrt[3]{p}$ , will also be in standard form. There is no possibility of overflow.

**Author:** Mr. E.W. Solomon of the University of Southampton.

London Computer Centre,  
68, Newman Street,  
London, W.1.

*This document is distributed with the kind permission of Southampton University by Ferranti Ltd. as part of its services to users of Ferranti Pegasus Computers. The document must not be reproduced in whole or in part without the permission of Southampton University.*

Issue 2  
22nd November, 1960.  
E. W. S. M. J. M.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## COMPLEX SQUARE ROOT

$$(c + id) = \frac{1}{2}\sqrt{(a + ib)} \quad a = a_1 + 2^{-38}a_2$$

$$b = b_1 + 2^{-38}b_2$$

The subroutine forms half the square root of the double length complex number  $a_1 + 2^{-38}a_2 + i(b_1 + 2^{-38}b_2)$ , giving a single length result,  $c + id$ .  $a_1, a_2, b_1$  and  $b_2$  must be stored in X4, 5, 6 and 7 respectively before entry. On exit,  $c$  and  $d$  are left in X4 and 5 respectively.

The real part of the square root is taken as positive.

**Name:** COMPLEX SQ. ROOT

**Store:** 6 blocks.

**Uses:** U0, 1; all the accumulators except X1.

**Cue:** 01 (4+.0).

**Time:** Between 100 and 140 milliseconds; e.g.  $a = b = \frac{1}{4}$  : 100 ms.,  
 $a = b = 0$  : 140 ms.

**Link:** Obeyed in U0.0 and left unaltered in X1.

**Error:** Less than  $2^{-39}$  except for numbers with moduli greater than about  $2^{-6}$  when the error is not greater than  $2^{-38}$ .

Normally there is no error when a number has an exact single length square root.

**Author:** Mr. J.G.F. Francis of the National Research Development Corporation.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
29.7.56.

## EXPONENTIAL

$$p' = \frac{1}{4} \exp p$$

**Name:** EXP  
**Store:** 2 blocks.  
**Uses:** U0; X6, 7.  
**Cue:** 01  
**Time:** 29 milliseconds.  
**Error:** Does not exceed  $2^{-37}$ .  
**Link:** Obeyed in 0.7 and left unchanged in X1.

## FERRANTI LTD.

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
16.10.56.

## LOGARITHM, WIDE RANGE

$$p' = \frac{1}{32} \log_e (pq)$$

where  $2^{-46} \leq (pq) < 1$

and  $q$  is positive or zero.

**Name:** LOG<sub>q</sub>

**Store:** 3 blocks

**Uses:** U 0,1; X5, 6,7

**Cue:** 01 (1 + .0)

**Time:** 34 milliseconds if  $2^{-13} \leq (pq) < 1$   
50 milliseconds if  $2^{-46} \leq (pq) < 2^{-13}$

**Link:** Obeyed in 1.7 and left unaltered in X1.

**Error:** About  $2^{-37}$

**Loop Stops:** In 0.0+ if  $(pq) < 0$   
In 1.0+ if  $0 \leq (pq) < 2^{-46}$

**Note:** Only the most significant half of the normalized form of  $(pq)$  is used in calculating the logarithm.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
27.5.57.

## FLOATING POINT EXPONENTIAL

$$p' = \exp(p)$$

where  $p$  and  $p'$  are floating point numbers (see Note 2) and  $p < 176.059$

Name: FP EXP MK 2

Store: 4 blocks + 2 blocks for R 220

Uses: U0,1,2; X1,5,6,7.

Cue: 01 (0+.0)

Time: 78 milliseconds if  $p > 0$

85 milliseconds if  $0 > p > -176.059$

13 milliseconds if  $p < -176.059$

Link: Obeyed in 2.7. Not left in X1.

- Notes: 1. R 222 uses R 220, fixed point exponential, as a subroutine.
2. The floating point numbers used with R 222 are of the same form as those normally used with R 610.

The exponent occupies the *nine* least significant bits and the argument occupies the *thirty* most significant bits including the sign digit.

Method: If  $p = A.2^a$

Then  $\exp(p) = B.2^b$

Where  $B = \frac{1}{8} \exp(R) = \frac{1}{4} \exp(R - \log_e 2)$

and  $b = \left[ \frac{p}{\log_e 2} \right] + 3$

$\left[ \frac{p}{\log_e 2} \right]$  is the integral part of  $\frac{p}{\log_e 2}$

and  $R$  is the remainder.

The subroutine evaluates  $\exp |p|$  by the above method and then takes the reciprocal if  $p < 0$ .

Error: The error in the exponential will not exceed one in the last binary place of the argument.

Loop Stop: In  $0.6+$  if  $p > (0.34386).2^9$  (= 176.059)

Ferranti Ltd.,  
London Computer Centre,  
21, Portland Place,  
LONDON, W. 1.

Issue 2  
27th May, 1957  
D.M. M.E.H.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
31.10.56.

## FLOATING POINT LOGARITHM

$$p' = \log_e p$$

$p$  and  $p'$  are packed floating point numbers of the form normally used with R 610.  
(See Note 3 below.)  $p$  must be positive and non-zero.

**Name:** FP LOG<sub>Q</sub>  
**Store:** 2 blocks + 3 blocks for R 221.  
**Uses:** U0, 1; X1, 4, 5, 6, 7.  
**Cue:** 01 (0 + .2)  
**Time:** 68 milliseconds.  
**Link:** Obeyed in 1.5. Not left in X1.

**Notes:**

- 1) Loop stops will occur
  - a) In 0.0+ if  $p < 0$
  - b) In 1.0+ if  $p = 0$
- 2) This subroutine uses R 221, fixed point logarithm, as a subroutine.
- 3) The floating point numbers used with R 223 are of the same form as those normally used with R 610.  
i.e. The exponent occupies the *nine* least significant bits and the argument occupies the thirty most significant bits including the sign digit.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
28.12.56.

## LOGARITHM, VARIABLE RANGE

$$p' = \frac{1}{2^n} \log_e (pq)$$

Where  $n$ , which must be an integer greater than zero, is specified by a preset parameter.

Only the more significant half of the normalized form of  $(pq)$  is used in calculating the logarithm.

**Name:** LOG MK.2  
**Store:** 3 blocks.  
**Uses:** U 0, 1; X 5,6,7.  
**Cue:** 01 (1 + .1)  
**Time:** 42 milliseconds if  $2^{-15} \leq (pq) < 1$   
 58 milliseconds if  $(pq) < 2^{-15}$ .  
**Link:** The link is obeyed in 1.7 and left unaltered in X1.  
**Error:** About  $2^{-37}$ .

1. Lower Limit of  $(pq)$ 

<u>Value of <math>n</math></u>	<u>Lower Limit of <math>(pq)</math></u>
6	$2^{-76}$
5	$2^{-46}$
4	$2^{-23}$
3	$2^{-11}$
2	$2^{-5}$
1	$2^{-2}$

## 2. Preset Parameter

The value of  $n$  should be set by a parameter list of the following form:

R 0 0 -0 1
<del>244</del> 224 - 04 -
$n$ 0 00 0.
0

If no parameter list is supplied by the programmer,  $n$  will be set as 5 by an optional parameter list.

## 3. Loop Stops

- (i) In 0.1+ if  $(pq) < 0$
- (ii) In 1.1+ if  $(pq)$  is positive but smaller than the lower limit for the value of  $n$  which is being used. (See paragraph 1 above).



## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
19.12.57.

## FLOATING-POINT EXPONENTIAL

$$p' = \exp(p)$$

where  $p$  and  $p'$  are packed floating-point numbers (see Section 1).

**Name:** FP EXP MK 4

**Store:** 3 blocks + 2 blocks for R 220.

**Uses:** U 0, 1; X 1, 6, 7.

**Cue:**

01

0+	1	72
1.0	0	60

**Time:** Approximately 74 milliseconds.

**Link:** Obeyed in 1.7. Not left in X1.

## 1. Parameter List

**1.1** R 225 works entirely in single-length floating-point arithmetic as described in the Pegasus Programming Manual and in the specifications of R 11, R 610 and other floating-point subroutines.

**1.2** The value of  $n$ , i.e. the number of binary digits used to represent the exponent part of the floating-point numbers, is specified by a preset parameter. The parameter list should be of the form

R 0 0 -0 1
225 - 04 -
$-2^{n-1}$

If no parameter list is supplied  $n$  will be set equal to 9 by an optional parameter list.

**2. Method**

$$\text{If } p = A.2^a$$

$$\text{then } \exp(p) = B.2^b$$

$$\text{where } B = \frac{1}{4} \exp(F \log_e 2)$$

$$b = \text{integral part of } \frac{p}{\log_e 2} + 2$$

$$F = \text{fractional part of } \frac{p}{\log_e 2}$$

for:-

$$\frac{p}{\log_e 2} = b - 2 + F$$

$$p = (b - 2) \log_e 2 + F \log_e 2$$

$$\exp(p) = \exp\{(b - 2) \log_e 2 + F \log_e 2\}$$

$$= 2^{(b-2)} \exp(F \log_e 2)$$

$$= 2^b \cdot \frac{1}{4} \exp(F \log_e 2)$$

$$= B.2^b$$

**3. Error**

The maximum error will not exceed one in the last binary place of the argument, except if underflow occurs, i.e.  $p' = 0$ . When  $n = 9$  the maximum error is little more than one half in the last binary place of the argument.

**4. Loop Stop**

A loop stop will occur in 0.6 if floating-point overflow occurs.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
23.10.56.

## SINE OR COSINE

$$\begin{aligned} \phi' &= \sin \pi \phi \\ \text{or } \phi' &= \cos \pi \phi \end{aligned}$$

**Name:** SIN/COS

**Store:** 2 blocks

**Uses:** U0,1; X1, 5, 6, 7.

**Cues:** 01 (0 + .0) for  $\sin \pi \phi$   
02 (0 + .0+) for  $\cos \pi \phi$

**Time:** 24 milliseconds.

**Link:** Obeyed in 1.1. Not left in X1.

**Error:** Less than  $5.2^{-38}$

**Notes:**

- 1) Angles are stored with a scale factor  $\pi$ . This enables them to be added or subtracted at will without concern for overflow, provided that only single-valued functions of direction are being considered.
- 2)  $\sin \frac{\pi}{2}$  and  $\cos 0$  are given as  $1 - 2^{-38}$ .

FERRANTI LTD

PEGASUS LIBRARY SPECIFICATION

Issue 2.  
10.10.56.

ARCTAN

$$p' = \frac{1}{\pi} \arctan (p)$$

$$\text{or } p' = \frac{1}{\pi} \arctan \left( \frac{p}{q} \right)$$

The result lies in the range  $-\frac{1}{2} < p' < \frac{1}{2}$

Name: ARCTAN

Store: 4 blocks

Uses: U0, 1; and all the accumulators except X1.

Cues: 01 (1+.0) for  $\frac{1}{\pi} \arctan (p)$

02 (1+.1+) for  $\frac{1}{\pi} \arctan (p/q)$

03 } partial cues, see note 1 below  
 04 }

Time: 48 ms

Link: Obeyed in 1.1 and left unaltered in X1

Error: Not greater than  $2^{-37}$ . See note 2.

Notes: (1) When evaluating  $\frac{1}{\pi} \arctan (p/q)$  (i.e. entry 02) a loop stop occurs in 1.2+ if  $p = q = 0$ .

In some cases the loop stop mentioned above may be inconvenient. For this reason the partial cues 03 and 04 have been included so that the order pair causing this stop may be altered by the master programme.

The order pair in question is:

7	3 00
1.2+	6 60

and any order pair which is to replace this pair must be of the form:

7	3 00
---	------

B.P	6 60
-----	------

where *B.P* is some address in blocks U2-5, in which is stored a routine written by the programmer to deal with the case  $\arctan (o/o)$ .

In order to write this order pair into the appropriate place in R 241 it must be placed in X1 and then written up by a single-word write order.

3	2 71
---	------

which must be tagged by 03 if it is an a-order and by 04 if it is a b-order.

- (2) The error is such that if  $|p| = 2^{-38}$ ,  $\arctan p$  will have the incorrect sign i.e.  $\frac{1}{\pi} \arctan 2^{-38}$  is given as  $-2^{-38}$

and  $\frac{1}{\pi} \arctan -2^{-38}$  is given as  $+2^{-38}$

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1.  
10.10.56.

## ARC SIN/ARC COS

$$\phi' = \frac{1}{\pi} \arcsin \phi \quad -\frac{1}{2} \leq \phi' \leq \frac{1}{2}$$

$$\text{or } \phi' = \frac{1}{\pi} \arccos \phi \quad 0 \leq \phi' < 1$$

**Name:** ARCSIN/ARCCOS

**Store:** 2 blocks  
plus 1 block for R 200  
plus 4 blocks for R 241

**Uses:** U0, 1 and all the accumulators

**Cues:** 01 (0+.1+) for  $\frac{1}{\pi} \arcsin \phi$   
02 (0+.1) for  $\frac{1}{\pi} \arccos \phi$

**Time:** About 130 milliseconds

**Link:** Obeyed in 1.7. Not left in X1.

**Error:** The error is not greater than  $2^{-37}$

**Notes:** (1) If  $\phi = -1.0$ ,  $\arccos \phi$  is taken to be  $(1-2^{-38})$   
(2) The error is such that if  $|\phi| = 2^{-38}$ ,  $\arcsin \phi$  will have the incorrect sign, i.e.  $\frac{1}{\pi} \arcsin 2^{-38}$  is given as  $-2^{-38}$   
and  $\frac{1}{\pi} \arcsin -2^{-38}$  is given as  $+2^{-38}$

(3) There are two partial cues

03 a-order partial cue  
04 b-order partial cue

(4) R 200 and R 241 are used as subroutines.

FERRANTI LTD

PEGASUS LIBRARY SPECIFICATION

Issue 1  
3.1.57.

COSINE AND SINE

$$\begin{aligned} p' &= \cos \pi \phi \\ \text{and } q' &= \sin \pi \phi \end{aligned}$$

A self-preserving subroutine.

Name: COS + SIN

Store: 3 blocks.

Uses: U 3,4,5; X 1,4,5,6,7.

Cues: 01 a-order 0+ cue.  
02 b-order 0+ cue.

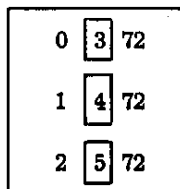
Time: About 42 milliseconds.

Link: Computing store link set in X1 with a 40 order. The rest of X1 must be zero. (The link is obeyed in 5.2+ and is not preserved).

Error: Less than  $10 \times 2^{-38}$  for both sine and cosine. A sine or cosine whose real value is  $\pm 1$  will be held as  $(1 - 2^{-38})$  with the correct sign.

Method of Use

1. Before any use is made of R 243 it must be brought into blocks 3,4,5 of the Computing Store by means of three block-transfer orders:-



These orders must be tagged in order that the correct block-address of R 243 is added to each of them; this block-address is given in cues 01 (for an a-order) and 02 (for a b-order). For example, suppose that the three block-transfer orders are in block 9+ of the master-programme with the first two of them in 9+.3 and the third as the a-order of 9+.4:-

B 9+.3	0	3	72
	1	4	72
B 9+.4	2	5	72

Then the following three tags calling for cues must be punched at the head of the master-programme:-

R 9	3	-0	1
243	-	01	-
R 9	3	-0	2
243	-	01	-
R 9	4	-0	1
243	-	01	-

Cue 01 (a-order block-address)

Cue 02 (b-order block-address)

Cue 01 (a-order block-address)

When R 243 has been brought in in this way it may be used repeatedly with Computing Store links.

2. Before each entry to R 243 a computing store link must be set in X1 with the order:-

(A)	1	40
-----	---	----

where  $A$  is the computing store address to which R 243 is to jump on exit.

3. The subroutine can then be entered at 3.0. It will place  $\cos \pi \phi$  in X6 and  $\sin \pi \phi$  in X7, finally returning control to the master programme at point  $A$ .

**Author:** Dr. D. Rogers, University College of Cardiff.



## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## TANGENT / COTANGENT

$$\frac{p'}{q'} = \tan \pi p$$

Name: TAN / COT

Store: 3 blocks.

Uses: U0, 1; X1, 6, 7.

Cue: 01(0+.3).

Time: 29 milliseconds.

Link: Obeyed in U0.0.

Notes: (1) The difficulty of presenting a result in the range  $(-\infty, \infty)$  has been avoided by producing a ratio and leaving the division to the user.

$$\text{If } p = \frac{1}{2} (\eta + \xi)$$

$$\text{and } t = \tan \left( \frac{\pi \xi}{2} \right)$$

$$\text{where } \begin{cases} \eta = -2, -1, 0, 1, 2 \\ -\frac{1}{2} \leq \xi < \frac{1}{2} \end{cases}$$

$$\text{Then } \eta \text{ even } \begin{cases} p' = -2t \\ q' = t^2 - 1 \end{cases} \quad \eta \text{ odd } \begin{cases} p' = t^2 - 1 \\ q' = 2t \end{cases}$$

$$\text{Thus } \frac{p'}{q'} = \tan \pi p \quad \text{and} \quad \frac{q'}{p'} = \cot \pi p$$

(2) Block 0+ of the subroutine is self preserving in U0 and subsequent entry may be made by jumping to 0.3.

Re-entry to 0.3+ will permit use of the previous link or a link dumped in U0.0.

© FERRANTI LTD 1962

London Computer Centre,  
68, Newman Street,  
London, W.1.

*Not to be reproduced in whole or  
in part without the prior written  
permission of Ferranti Ltd.*

Issue 1  
9th October, 1961  
T.B.W.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## FLOATING-POINT COSINE AND SINE

$$p' = \cos p$$

$$q' = \sin p$$

where  $p$ ,  $p'$  and  $q'$  are packed floating-point numbers.

**Name:** F.P. COS + SIN.

**Store:** 8 blocks, which may be stored above B 127.

**Uses:** U0; X6, 7.

**Cues:**

01	0+ <span style="border: 1px solid black; padding: 0 2px;">0</span> 72	Normal entry when R 250 is stored in B127 or below.
	0.0 0 60	

02	0 <span style="border: 1px solid black; padding: 0 2px;">0</span> 72 2	Entry with X2 containing cue 03 for R 250 stored above B127.
	0.1+ 0 60	

03	0+ - 00 0.	Modifier to be set in $2_m$ before using cue 02.
	0	

**Time:** 99 milliseconds.

**Link:** Obeyed in 0.6; left unaltered in X1,

### 1. Entry

1.1 If R 250 is stored in B127 or below, it may be entered in the usual way by obeying cue 01.

1.2 If R 250 is stored above B127, entry is made by setting cue 03 in X2 and obeying cue 02. In this case a set of orders like the following must be obeyed in order to enter the subroutine:-

	6 00	Set $\phi$
	<i>B.P</i> 2 00	Set cue 03 in X2
	1 00	Set link
	+0	+ cue 02 to R 250
<i>B.P</i>	+0	+ cue 03 to R 250

Note that cue 03 occupies only the modifier of X2, the sign and counter may be used for other purposes by the master programme.

1.3 If cue 01 is used when R 250 is stored above B127, a 77 stop will occur in the location in which the cue is obeyed, and this will be followed by an unassigned order stop on operating the RUN key. The contents of this location will be:

25	0	77
25	0	33

**2. Parameters**

2.1 R 250 works entirely in single-length floating-point arithmetic as described in the Pegasus Programming Manual, and in the specifications of R 11 and R 610.

2.2 The value of  $n$ , i.e. the number of binary digits used to represent the exponent part of the floating-point numbers, is specified by present parameter 01.

2.3 R 250 normally leaves its results in floating-point form. Fixed-point results may be obtained by using preset parameter 02 and setting  $X1 \geq 0$  on entry to the subroutine. If  $X1 \geq 0$  on entry, the subroutine will jump to  $N$ , the address specified in P.P.02, with  $\phi' = \cos \phi$  and  $q' = \sin \phi$  in fixed-point form.  $X2_{\phi}$ , 3, 4 and 5 will then have been used by the subroutine but their original values will be stored in  $B6+.2$ , 3, 4 and 5. If  $N = 0.1+$  (as in the optional parameter list) the subroutine will always produce floating-point answers, whatever the sign of  $X1$ .

2.4 The precision of the fractional part of the operand  $\phi$  is  $31-a$  binary digits, where  $a$  is the binary exponent of  $\phi$ . It is therefore desirable to define an upper limit,  $a_{max}$ , to the size of  $a$  and this is specified by preset parameter 03. This implies a lower bound of  $31-a_{max}$  bits in the precision of the sine and cosine.

2.5 The parameter list should be of the form:

	R 0 0 -0 3	
	250 - 04 -	
01	$-2^{n-1}$	$n$ = number of binary digits representing the exponent.
02	$N$ 0 00 0. 0	$N$ = address to which subroutine jumps if $X1 \geq 0$ .
03	$a_{\max}$ . 0 00 0. 0	$a_{\max}$ = largest value that the binary exponent of $\phi$ may take.

If no parameter list is supplied,  $n$  will be set equal to 9,  $N$  equal to 0.1+ and  $a_{\max}$ . equal to 21 by an optional parameter list.

### 3. Error

The error in the sine and cosine of the operand  $\phi$  should not exceed 1 in the last digit of the argument. Note, however, that the effective operand is the fractional part of  $\phi/\pi$ . If  $\phi/\pi \gg 1$  the precision of its fractional part, and hence of  $\sin \phi$  and  $\cos \phi$ , will be reduced accordingly.

### 4. Loop Stop

There will be a loop stop in 0.2+ (0.2+ 4 62) if the exponent of  $\phi$  is greater than the maximum value specified by P.P. 03.

Author: Mr. F. Burgoyne of Babcock and Wilcox Ltd.

FERRANTI LTD

PEGASUS LIBRARY SPECIFICATION

FLOATING POINT ARCTAN

$$p' = \arctan (p)$$

$$\text{or } p' = \arctan (p/q)$$

The result lies in the range  $-\pi/2 \leq p' \leq \pi/2$  (cue 01)

or  $-\pi \leq p' \leq \pi$  (cue 02)

Name: FP ARCTAN

Store: 8 blocks.

Uses: U0, 1; X4, 5, 6, 7.

Cues: 01 (0+.2) for  $\arctan (p)$ .

02 (0+.2+) for  $\arctan (p/q)$ .

03 a-order partial cue } (see Section 2)  
04 b-order partial cue }

Time: Approximately 88 milliseconds.

Link: Obeyed in U1.7 and left unaltered in X1.

1. Parameter List

1.1 R 251 works in single-length floating-point arithmetic as described in the Pegasus Programming Manual and in the specifications of R 11 and R 610.

1.2 The value of  $n$ , i.e. the number of binary digits used to represent the exponent part of the floating-point numbers, is specified by a preset parameter. The parameter list should be of the form

RO 0-01
251 -04-
$-2^{n-1}$

If no parameter list is supplied  $n$  will be set equal to 9 by an optional parameter list.

2. Loop Stop

When evaluating arctan (p/q) a loop stop occurs in 0.6+ if p = q = 0:

0.6	3	7	04
	0.6+	7	60

In some cases this loop stop may be inconvenient, and the programmer may wish to write a routine to deal with the case arctan (0/0).

The entry point (B.P) of this special routine should be somewhere in blocks U2-5. After setting the required result, p', in X6, the routine should exit by obeying the link which will be in X1.

To insert this facility into R 251, the following sequence should be read after the A3 on the master programme tape:-

```

C 251
X 3+.6+
B.P 7 60

```

As an alternative to using the above C sequence, the order pair in 3+.6 of R 251 may be replaced by using one of the partial cues. An order pair of the form

3	7	04
B.P	7	60

should be placed in X1 and written away, before the subroutine is entered, by a single word write order tagged by the appropriate partial cue:

3	6	71	+ partial cue
---	---	----	---------------

3. Range of p'

Using cue 02 the value of p' may range from -π to +π, the choice of quadrant depending on the signs of p and q. On entry by cue 01 q is set equal to +1.0 and p' must lie between -π/2 and +π/2.

4. Error

The maximum error will not exceed one in the last binary place of the argument, except if underflow occurs, i.e. p' = 0.

FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
30.11.56

## COMPLETE ELLIPTIC INTEGRALS

$$p' = \pi / 2 K(p)$$

$$\text{and } q' = 2 E(p) / \pi$$

$$\text{where } K(p) = \int_0^{\pi/2} (1 - p^2 \sin^2 x)^{-1/2} dx$$

$$\text{and } E(p) = \int_0^{\pi/2} (1 - p^2 \sin^2 x)^{1/2} dx$$

**Name:** CEL

**Store:** 4 blocks.

**Uses:** U0, 1; X1, 4, 5, 6, 7.

**Cue:** 01.

**Time:** Between 65 and 564 milliseconds, depending on  $p$ .

**Link:** Normally obeyed in 1.7. But obeyed in 1.3 if  $p = 0$  or  $-1.0$ .

**Notes:**

- (1) Both the time and the error depend on  $p$ , being small when  $p$  is near zero and large when  $p$  is near  $\pm 1$ . For  $0.01 < |p| \leq 0.99$ , the time varies from about 120 milliseconds to about 300 milliseconds, and the error is probably less than  $2 \cdot 10^{-10}$  in both  $p'$  and  $q'$ . The worst error in  $p'$  and  $q'$  for any value of  $p$  is probably less than  $3 \cdot 10^{-9}$ .
- (2) When  $p$  is near enough zero for  $\pi/2K$  and  $2E/\pi$  to round off to  $+1.0$ , the subroutine gives the answers as  $1-2^{-30}$ .

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
8.10.56.

## ERROR FUNCTION

Given  $p = x \cdot 2^{-2}$

Then  $p' = \text{erf}(x)$

$$= \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

**Name:** ERROR FN

**Store:** 2 blocks.

**Uses:** U0,1; X4, 5, 6, 7.

**Cue:** 01 (0 + .0)

**Time:** 38 milliseconds.

**Link:** Obeyed in 1.2 and left unaltered in X1.

**Error:** Less than  $\frac{1}{2} \times 10^{-6}$   
i.e. The answer is correct to six decimal places.

**Loop Stop:** There is a loop stop in 0.0 if  $p < 0$

**Method:** This subroutine uses an approximation due to Cecil Hastings Jnr.:

$$\text{erf}(x) = 1 - \frac{1}{(1 + a_1 x + a_2 x^2 + \dots + a_6 x^6)^{16}}$$

where

$a_1$	=	.0705	2307	84
$a_2$	=	.0422	8201	23
$a_3$	=	.0092	7052	72
$a_4$	=	.0001	5201	43
$a_5$	=	.0002	7656	72
$a_6$	=	.0000	4306	38

**Author:** Mr. B. W. Gregory of Sir W.G. Armstrong Whitworth Aircraft Ltd.



## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

*Issue 1*  
17. 10. 58.

## ELLIPTIC FUNCTIONS

$$x'_5 = \operatorname{sn}(4Kq, p)$$

$$p' = \operatorname{cn}(4Kq, p)$$

$$q' = \operatorname{dn}(4Kq, p)$$

where  $K(p)$  is the complete elliptic integral of the first kind.

**Name:** ELL. FCTS.

**Store:** 12 blocks + 9 blocks for other subroutines used (R 200, 240, 241, 242).

**Uses:** U0, 1, 2; B0, 1; and all the accumulators.

**Cue:** 01 (0+.0)

**Time:** Depends on  $p$  and  $q$ . An average time is 2 seconds.

**Link:** Obeyed in 2.6, not left in X1.

**Notes:**

- (1) R 240 and R 242 are used as subroutines, and these in turn use R 200 and R 241.
- (2) For the special values of  $q$  and  $p$  for which one or more of the functions round to +1, the subroutine gives  $1 - 2^{-38}$ .

**Author:** Dr. G.N. Lance of the University of Southampton.

© FERRANTI LTD 1958

*Not to be reproduced in whole or  
in part without the prior written  
permission of Ferranti Ltd.*

Ferranti Ltd., London Computer Centre, 21, Portland Place, LONDON W.1.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## FLOATING POINT BESSEL FUNCTIONS

R 270 is a self-preserving subroutine which calculates the Bessel Functions

$$p' = f_q(p)$$

where  $p$  and  $p'$  are packed floating-point numbers and  $q$  is an integer defining the type of Bessel Function according to the following code:

$$f_0 = I_0, f_1 = I_1, f_2 = K_0, f_3 = K_1, f_4 = J_0, f_5 = J_1, f_6 = Y_0, f_7 = Y_1$$

**Name:** BESSEL FUNCTIONS

**Store:** 34 blocks + 8 blocks for R 250 (these 42 blocks may be stored above B127) plus 9 blocks for other subroutines which may not be stored above B127 (R 200, 220, 221, 225).

**Uses:** U0; X1, 6; B0, 1.

**Cues:**

01	0+ - 00 0.
	0

Modifier to be set in  $2_m$  before using cue 02

02	0 0 72 2
	0.0 0 60

Enter R 270, with X2 containing cue 01

See Section 1.

R 270 is self-preserving and may be re-entered at U0.0 provided that  $2_m$  is undisturbed.

**Time:** The time taken is variable; the maximum is about 200 milliseconds but the average time is about 100 milliseconds.

**Link:** Set in X1. It may be either

- (a) a Computing Store Link, obeyed in 0.6; not preserved
- or (b) a "go" order pair, obeyed in 0.7; left in X1 on exit.

**Error:** The error is of the order of 1 in the last binary place of the argument.

### 1. Entry

R 270 is entered by setting cue 01 in X2 and obeying cue 02. A sequence of orders like the following must be obeyed in order to enter the subroutine:-

	6 00	set $p$
	7 40	set $q$
	B.P 2 00	set cue 01 in X2
	1 00	set link
	+0	+ cue 02 to R 270
B.P	+0	+ cue 01 to R 270

Note that cue 01 occupies only the modifier of X2, the sign and counter may be used for other purposes by the master programme.

**2. Parameter List**

2.1 R 270 works entirely in single length floating-point arithmetic as described in the Pegasus Programming Manual and in the specifications of R 11 and R 610.

2.2 If no parameter list is supplied by the programmer, the value of  $n$  (the number of binary digits used to represent the exponent) will be set equal to 9 by an optional parameter list.

2.3 If it is desired to use a value of  $n$  other than 9, a parameter list of the following form must be provided for R 270 but not for R 250 or 225:-

R 0 0 -0 1
270 - 04 -
$-2^{n-1}$

R 270 will use this list to provide the appropriate parameters for R 225 and R 250.

2.4 R 270 uses R 250 with the address  $N = 1.6$  in preset parameter 02, and the largest value of the exponent  $a_{max} = 21$  in present parameter 03. The master programme must therefore not attempt to alter these parameters.

**3. Permitted Limits for the Operand  $p$**

The operand,  $p$ , must satisfy the following restrictions:

$$\begin{aligned} \text{For } I(p): & \quad -4 < p < (2^{n-1} - 2) \log_e 2 \\ K(p): & \quad -4 < p < (2^{n-1} - 2) \log_e 2 \\ & \quad \text{and } |p| \geq 2^{-21} (\doteq 4.8 \times 10^{-7}) \\ J(p): & \quad -4 < p < 2^{(2^{n-1}-2)} \end{aligned}$$

$$Y(\phi): \quad -4 < \phi < 2^{(2^{n-1}-2)}$$

$$\text{and } |\phi| \geq 2^{-21} (\doteq 4.8 \times 10^{-7})$$

$$\text{If } n = 9, \quad (2^{n-1} - 2) \log_e 2 \doteq 180$$

$$2^{(2^{n-1}-2)} \doteq 2.9 \times 10^{76}$$

#### 4. Storage

If R 250 and R 270 are to be stored above B 127, the master programme must ensure that the other subroutines used by R 270 are stored below B 128. This may be done most easily by calling for R 200, 221 and 225 and ensuring that they, and R 220, are read before R 250 and 270.

#### 5. Error Stops

The following loop stops in R 200, 225 and 221 will occur if the conditions stated in section 3 above are not observed:

0.0 6 63     if  $\phi \leq -4$   
 0.6 0 60     if  $\phi \geq (2^{n-1} - 2) \log_e 2$  (I, K)  
 1.0+ 6 61     if  $|\phi| < 2^{-21}$  (K, Y)

There will be a writing with overflow stop in 0.0 if the subroutine is entered with the overflow indicator set:

0	7	73
6	4	00

**Author:** Mr. F. Burgoyne of Babcock and Wilcox Ltd.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
7.2.57.

## GAUSSIAN QUADRATURE

Evaluates  $\int_{X-h}^{X+h} f(x) dx$  and leaves the result in X6.

$h$  must be set in X6 before entry and must satisfy  $|h| < 1$ .

**Name:** GAUSS QUAD.

**Store:** 2 blocks plus list of roots and weights. During input a total of 12 blocks is used. See Section 5.

**Uses:** U0, 1; X6; B0. X5 is also used if the cue to the auxiliary is a programme parameter. See Section 4.

**Cue:** 01 (0+.1)

**Time:** The average time is approximately  
 $47 + 25[\frac{1}{2}(n+1)]$  milliseconds +  $[\frac{1}{2}(n+1)] \times$  time taken by auxiliary.

$n$  is the number of points to be used in quadrature.

**Link:** Obeyed in 0.7 and left unaltered in X1.

## 1. Method of Use

1.1  $h$  (where  $|h| < 1$ ) must be set in X6 before entry.

1.2 The function,  $f$ , is defined by an auxiliary as specified in the following Section.

## 2. The Auxiliary

2.1 The auxiliary must have the specification:

$$p' = \frac{1}{2} \{f(X+p) + f(X-p)\}$$

2.2 It may use any part of the Computing Store except X2, X3 and X4, and must not alter B0.

2.3 Return to R 300 should be made by obeying the link which is set in X1 on entry to the auxiliary.

**3. Parameter List**

3.1 The constants used in the 6,8,10,12 and 16 point Gaussian formulae appear with the subroutine on the Library Tape.

3.2 If one of these five formulae is required, the parameter list should be as follows:

	R 0 0 -0 2	
	300 - 04 -	
01	+ $n$	Formula required
02		Cue to auxiliary. See Section 4.

3.3 If any other Gaussian formula is required, the parameter list should be of the following form:

	R 0 0 - $u$ $v$	
	300 - 04 -	
	- $n$	Formula to be used.
		Cue to auxiliary. See Section 4
	$t_1$	
	$H_1$	
	$t_2$	
	$H_2$	
	etc.	

where  $t_1, t_2, etc.$  are the non-negative roots of the Legendre polynomial of degree  $n$ , and  $H_1, H_2, etc.$  are the corresponding weights. (If  $n$  is odd, the weight corresponding to  $t = 0$  must be half the value normally tabulated.)

3.4 The maximum value of  $n$  which R 300 can handle is 32.

#### 4. Cue to the Auxiliary

4.1 The cue to the auxiliary can be a preset parameter in which case the parameter list should be punched with the auxiliary so that it will have the same relativiser.

4.2 Alternatively the cue can be a programme parameter set in X5. In this case preset parameter 02 should be the order pair:

0.5	5	10
0.5	0	60

#### 5. Storage Space

5.1 During input the programme of R 300 together with an input interlude occupies 12 blocks. After input is complete the programme occupies 2 blocks and the list of roots and weights occupies the integral part of  $\frac{1}{8}(n+7)$  blocks. The transfer address is re-set when the input interlude has finished its work so that the remainder of the 12 blocks can be overwritten.

#### 6. Correcting an Error in Preset Parameter 01

6.1 If  $n$  has some value other than 6, 8, 10, 12 or 16 but preset parameter 01 has been punched as  $+n$  instead of  $-n$ , there is a 77 stop in 1.5 during input and the characters CR LF \* 300 space  $+n$  are printed.

6.2 This error can be counteracted by placing a list of the following form in the second tape reader and operating the STOP/RUN key:-

```

t1
H1
t2
H2
⋮
etc.
⋮
+ n
J 6+
L
Z

```

6.3 If further subroutines are required, the Library Tape should be re-inserted in the second tape reader when the Z at the end of this special tape is reached. It should be moved on to the beginning of the next subroutine.

#### 7. Error Stops

##### a) During Input

- i) Loop stop in 1.3 if no parameter list supplied.
- ii) 77 stop on wrong sign for preset parameter 01. See Section 6 above.

##### b) During the Subroutine

Loop stop in 0.5 on overflow if the OVR has not been cleared by the auxiliary.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
6.6.58.

## DIFFERENTIATION OF A TABLE

Given a table of a function,  $y$ , and a table of arguments,  $x$ , not necessarily at equal intervals, the subroutine calculates an approximation to  $dy/dx$  at each tabular point.

**Name:** DIFFERENTIATION 2  
**Store:** 5 blocks.  
**Uses:** The entire Computing Store including all the accumulators.  
**Cue:** 01 (0+.0)  
**Time:** About  $30n$  milliseconds.  
**Link:** Obeyed in U3.3, not left in X1.

## Method of Use

On entry the address of the first  $x$  must be in  $3_m$  and the address of the first  $y$  in  $4_m$ . The number of points,  $n$ , in the table must be in  $3_c$ , and must be greater than 1. The routine will form a table of  $2^{-v}(dy/dx)$  and place it in the address given by  $5_m$ ,  $v$  being the small integer in  $5_c$ . All three addresses must be at the beginnings of blocks.

## Scaling

If  $y$  is stored with a scale factor  $\alpha$  and  $x$  with a scale  $\beta$ ,  $dy/dx$  will have a scale factor  $2^v(\alpha/\beta)$ . The value of  $v$  should be chosen so that  $2^{-v}(\alpha/\beta)(dy/dx)$  is always within capacity. Consistent with this it should be as small as possible and in any case should not exceed about 40.

## Method

The formula used is:-

$$\left(\frac{dy}{dx}\right)_r = \frac{(x_{r+1} - x_r)(y_r - y_{r-1})}{(x_r - x_{r-1})(x_{r+1} - x_{r-1})} + \frac{(x_r - x_{r-1})(y_{r+1} - y_r)}{(x_{r+1} - x_r)(x_{r+1} - x_{r-1})}$$

for  $1 < r < n$

and

$$\left(\frac{dy}{dx}\right)_1 = \frac{y_2 - y_1}{x_2 - x_1} \quad \left(\frac{dy}{dx}\right)_n = \frac{y_n - y_{n-1}}{x_n - x_{n-1}}$$



This is equivalent to fitting a parabola through the  $r - 1^{th}$ ,  $r^{th}$  and  $r + 1^{th}$  points for a general point, and to a straight line at the end points.

**Error**

Proportional to  $d^3y/dx^3$  except at the end points.

**Overflow**

If any  $2^{-v}(dy/dx)$  exceeds capacity there will be a writing with overflow stop in either 2.7 or 3.3, due to 73 orders in 2.6+ or 3.2+.

If  $n = 0$  there will be a writing with overflow stop in 2.7.

**Loop Stop**

There is a loop stop in 0.5+ for  $n = 1$  :-

0.5	1.0 3 67
	0.5+ 0 60

**Author:** Mr. H.P. Goodman of the De Havilland Aircraft Company.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2.  
4.10.56.

## LINEAR INTERPOLATION

$$p' = f(p)$$

This subroutine estimates the value of  $f(x)$ , for given  $x$ , by linear interpolation in a table.  $f(x)$  must be tabulated at equidistant values of  $x$ , and the given  $x$  must be within the range of the table.

**Name:** LINEAR INTERPOLATION

**Store:** 3 Blocks.

**Uses:** U0, 1; X5, 6, 7.

**Cues:** 01 To interpolate in table number  $q$ , the address of which is given by the  
(0+.2)  $q^{\text{th}}$  entry in an index.  $q$  is set in  $7_c$ . The address of the index is given by preset parameter 01.

02 To interpolate in the table whose address is given in the single word  
(0+.1) index. .

03 To interpolate in the table whose address is set in  $7_m$ .  
(0+.4)

**Time:** Using cue 01 72 milliseconds.  
Using cue 02 72 milliseconds. } approximately.  
Using cue 03 58 milliseconds.

**Link:** Obeyed in 0.6 and left unaltered in X1.

**Error:** See section 6 below.

**1. Layout of the Table**

The first word in each table must be stored at the start of a block; the contents of this block must be as follows:-

A.0		$N$	= Max. Number of points needed for interpolation.
.1		$m$	= Number of values of $f(x)$ tabulated.
.2	$x_0$		= Initial value of $x$ .
.3	$h$		= $(x_{i+1} - x_i)$ , the tabular interval.
.4	$f(x_0)$	} Successive values of the function in $m$ consecutive locations.	
.5	$f(x_1)$		
.6	$f(x_2)$		
	etc.		

$N-1$  denotes the order of interpolation which must be used to obtain maximum accuracy from the table.  $N$  is not used by R 320 but it should be inserted so that the table may be used with other interpolation routines.

For the same reason, although R 320 can deal with negative values of  $h$ , the values of  $x$  should normally be in ascending order so that  $h$  is positive.

**2. The Index**

If cue 01 is used an index of tables must be stored in the master programme. This index must be arranged in the store as shown below; it can most easily be formed by using R 102.

Location	Modifier	Counter
$B.P$	0	$t$
$B.P + 1$	$A_1$	$n_1$
$B.P + 2$	$A_2$	$n_2$
⋮	⋮	⋮
$B.P + t$	$A_t$	$n_t$

$A_1, A_2, \dots, A_t$  are the addresses of the first words in each table.  $n_1, n_2, \dots, n_t$  are not used by R 320; they are used by some routines to specify the number of points to be used for interpolation.  $t$  specifies the number of tables listed in the index.

**3. Preset Parameters**

If cue 01 is to be used the address of the index must be specified by preset parameter 01. If cue 02 is used the 'index' consists of one word containing  $(A, n)$ ;

the address of this word is specified by preset parameter 01. If no parameter list is read the address will be set at 2.0.

If the given value of  $x$  is outside the range of the table, the subroutine will jump to preset parameter 02. This may be the cue for an extrapolation routine or a loop stop in 0.4: if no parameter list is supplied it will be a loop stop in 0.4.

The parameter list will normally be punched as follows:-

R 0 0 -0 2	}	Title of parameter list.
320 - 04 -		
B - P0 0.	}	Address, <i>B.P.</i> , of the index.
0		
(0.4 0 60)	}	Loop Stop or Cue for Extrapolation Routine.
0		

#### 4. Extrapolation

On obeying the cue to an extrapolation routine the contents of the following registers may be of use.

X1 Link

X5 Integral quotient  $\frac{(x - x_0)}{h}$

X6 Remainder after dividing  $x - x_0$  by  $h$

X7  $q < 0$  if  $\frac{x}{h} < \frac{x_0}{h}$

$q \geq 0$  if  $\frac{x}{h} > \frac{x_{m-1}}{h}$

U1.0 - 1.6 First block of the table, except for its last word.

U1.7 ( $A, n$ )

#### 5. Method

A tabular argument  $x_r$  is selected such that

$$x_r \leq x < x_{r+1}$$

The computer then forms  $\delta x = \frac{x - x_r}{h}$

$$\text{and } f(x) = (1 - \delta x) \cdot f(x_r) + \delta x \cdot f(x_{r+1})$$

**6. Error**

Slight inaccuracy may arise due to the fact that  $h$ , the tabular interval, is stored in binary form. An exact decimal interval may be represented in binary with an error of up to  $2^{-39}$ . This may lead to a maximum error of  $(m-1) \cdot 2^{-39}$  in estimating the position of an argument near the upper boundary of the table. If the interval is exact the error in the process will not exceed  $2 \cdot 2^{-38}$ . These errors will usually be negligible compared with that introduced by the use of a linear approximation.

It is possible for an argument which is just within the range of the original table to be rejected as just outside the range of the binary table in the computer. This type of error can be eliminated by rounding  $h$  upwards, but this has the effect of increasing the error described in the previous paragraph.

**7. Loop Stop**

0.4	0.4 0 60
	0

$x$  outside the range of the table and no extrapolation routine provided.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
12.12.56

## POLYNOMIAL INTERPOLATION

$$p' = f(p)$$

This subroutine estimates the value of  $f(x)$ , for given  $x$ , by interpolating in a table.  $f(x)$  must be tabulated at equal intervals of  $x$ , and the given  $x$  must be within the range of the table. The order of interpolation must not exceed seven.

**Name:** POLYNOMIAL INTERPOLATION

**Store:** 8 Blocks.

**Uses:** U0, 1, 2; X6, 7; B0.

**Cues:** 01 To interpolate in table number  $q$ , where  $q$  is set in  $\gamma_c$ . The number of points to be used for interpolation and the address of the table are given by the  $q^{\text{th}}$  entry in the index.

(0+.1)  
02 To interpolate in the table specified by the single word index.  
(0+.2+)

03 To interpolate in the table whose address is set in  $\gamma_m$ . The number of points to be used for interpolation is set in  $\gamma_c$ .  
(0+.4)

**Time:** Entering at 03 and using  $n$  points to estimate  $f(x)$ , (i.e. using  $(n-1)^{\text{th}}$  order interpolation), the average time would be approximately

$$5n^2 - 2n + 96 \text{ milliseconds}$$

Number of Points  $n$

Time in milliseconds

2	3	4	5	6	7	8
108	135	168	211	264	327	400

Entering at 02 would add 10 milliseconds

Entering at 01 would add 14 milliseconds

**Link:** Obeyed in 2.0 and left unaltered in X1.

**Error:** See section 7 below.

**1. Layout of the Tables**

The first word in each table must be stored in the first word of a block; the contents of this block must be as follows:-

A.0	$N$	= Maximum number of points needed for interpolation.
.1	$m$	= Number of values of $f(x)$ tabulated.
.2	$x_0$	= Smallest value of $x$ in the table.
.3	$h$	= $(x_{i+1} - x_i)$ , the tabular interval.
.4	$f(x_0)$	} Successive values of the function in $m$ consecutive locations.
.5	$f(x_1)$	
.6	$f(x_2)$	

$N-1$  denotes the order of interpolation which must be used to obtain maximum accuracy from the table. The number of points actually used for interpolation will be the lesser of  $N$  and  $n$ , where  $n$  is specified in the index or on entry to the subroutine.

The values of  $x$  must be in ascending order and the interval  $h$  is therefore always positive.

**2. Scaling**

The table must be scaled so that

$$-1.0 \leq f(x) < +1.0$$

for all values of  $x$  within the range of the table.

Provision has been made for temporary scaling if overflow occurs in the working of the subroutine. The overflow indicator will not be set on exit from the subroutine unless  $f(x)$  exceeds capacity.

**3. The Index**

If cue 01 is used an index of tables must be stored in the master programme. This index must be arranged in the store as shown below; it can most easily be formed by using R 102.

Location	Modifier	Counter
$B.P$	0	$t$
$B.P + 1$	$A_1$	$n_1$
$B.P + 2$	$A_2$	$n_2$
⋮	⋮	⋮
$B.P + t$	$A_t$	$n_t$

$A_1, A_2, \dots, A_t$  are the addresses of the first words in each table.  $n_1, n_2, \dots, n_t$  denote the number of points to be used for interpolation.  $t$  specifies the number of tables listed in the index.

#### 4. Preset Parameters

If cue 01 is to be used the address of the index must be specified by preset parameter 01. If cue 02 is used the 'index' consists of one word containing  $(A_1, n)$ ; the address of this word is specified by preset parameter 01. If no parameter list is read the address will be set at 2.0.

If the given value of  $x$  is outside the range of the table the subroutine will jump to preset parameter 02. This may be the cue for an extrapolation routine or a loop stop in 1.2: if no parameter list is supplied it will be a loop stop in 1.2.

The parameter list will normally be punched as follows:-

R 0 0 -0 2	}	Title
321 - 04 -		
B - P0 0.	}	Address B.P of the index
0		
(1.2 0 60)	}	Loop Stop or cue to extrapolation routine
0		

#### 5. Extrapolation.

On obeying the cue to an extrapolation routine the contents of the following registers may be useful.

- X2  $(A, n)$
- X3 Integral quotient  $(x - x_0)/h$
- X6 Remainder after division  $(x - x_0)/h$
- X7  $q < 0$  if  $x < x_0$   
 $q > 0$  if  $x > x_{m-1}$
- U2 First Block of the Table
- B0 The accumulators, except for X7, as stored on entry. B0.7 will contain  $(A, n)$ . The link will be in B0.1.

#### 6. Method

The number of points used for interpolation is  $n'$ , which is the lesser of  $N$  (specified in the table) and  $n$  (specified on entry to the subroutine).

The method used is Neville's iteration process. The  $n'$  points in the table are chosen so that the values of  $x_i$  are as nearly symmetrical as possible about the given value of  $x$ . Provision is made for points which lie on or near the boundary of the table.



Let the  $n'$  arguments be  $x_s, x_{s+1}, \dots, x_{s+n'-1}$

and the  $n'$  functions  $f_s^{(0)}, f_{s+1}^{(0)}, \dots, f_{s+n'-1}^{(0)}$

The recurrence relation,

$$f_{s+i}^{(t+1)} = \frac{1}{x_{s+i+t+1} - x_{s+i}} \begin{vmatrix} f_{s+i}^{(t)} & x_{s+i} - x \\ f_{s+i+1}^{(t)} & x_{s+i+t+1} - x \end{vmatrix}$$

is applied with  $t = 0, i = 0, 1, \dots, n' - 2$   
 $t = 1, i = 0, 1, \dots, n' - 3$

or, in general, with  $t = t, i = 0, 1, \dots, n' - 2 - t$

& finally with  $t = n' - 2, i = 0$ .

$f_s^{(n'-1)}$  is the required  $(n'-1)$ th order interpolate.

Since  $x_{i+1} - x_i = h$  is constant the above recurrence relation can be re-written

$$f_{s+i}^{(t+1)} = \frac{1}{(t+1)h} \begin{vmatrix} f_{s+i}^{(t)} & x_{s+i} - x \\ f_{s+i+1}^{(t)} & x_{s+i} - x + (t+1)h \end{vmatrix}$$

$$= \frac{(f_{s+i+1}^{(t)} - f_{s+i}^{(t)})(x - x_{s+i})}{(t+1)h} + f_{s+i}^{(t)}$$

It is this last formula which is used in the subroutine.

**7. Error**

The round off errors in the interpolation process will not exceed  $n'.2^{-38}$  for a well behaved function. For functions which are not well behaved the round off error will be small compared with the truncation error introduced by fitting an  $(n'-1)$ th order polynomial.

Slight inaccuracy may arise due to the fact that  $h$ , the tabular interval, is stored in binary form. An exact decimal interval may be represented in binary with an error of up to  $2^{-39}$ . This may lead to a maximum error of  $(n-1)2^{-39}$  in estimating the position of an argument near the upper boundary of the table.

It is possible for an argument, which is just below the upper boundary of the original table, to be rejected as just above the boundary of the binary table in the computer. This type of error can be eliminated by rounding  $h$  upwards, but this has the effect of increasing the error described in the previous paragraph.

**8. Order of Interpolation**

The order of interpolation required to keep the truncation error below a given value may be estimated from the differences. Let the maximum error permissible in the function be defined as the unit of measurement. The differences of the function, measured in these units, should satisfy the following conditions.

For $n' =$	2	3	4	5	6	7	8
$\delta^{n'} <$	4	60	20	500	100	3500	400

$n'$  is the number of points used for interpolation.  
 $\delta^{n'}$  is the maximum  $n'$ th order difference in the range of the table.

### 9. Error Stops

0.4	0	7	73
	0	2	72 7

Write with OVR if OVR set on entry. It may also occur if the index address is incorrect.

0.4	0.4	6	62
	7	6	41

Loop stop if  $n' > 8$

0.5	0.5	6	63
	0	7	00

Loop stop if  $n' < 2$

1.2	1.2	0	60
	0		

Loop stop if  $x$  outside the range of the table and no extrapolation routine provided.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
12.12.56.

## TWO WAY LINEAR INTERPOLATION

$$p' = f(p, q)$$

This subroutine estimates the value of  $f(x,y)$ , for given  $x$  and  $y$ , by linear interpolation in a table. The table must have a rectangular boundary and equal intervals in  $x$  and in  $y$ . The given point  $(x,y)$  must be within or on the boundary of the table.  $x$  and  $y$  must be set in accumulators 6 and 7 respectively; the answer  $f(x,y)$  is left in accumulator 6.

**Name:** TWO WAY LINEAR INTERPOLATION

**Store:** 5 Blocks.

**Uses:** U0,1 and 2. X1,5,6,7.

**Cues:** 01 To interpolate in table number  $k$ , the address of which is given by the (0+.1+)  $k^{\text{th}}$  entry in the index.  $k$  is set in  $5_c$ .

02 To interpolate in the table whose address is given in the single word (0+.1) index.

03 To interpolate in the table whose address is set in  $5_m$ .  
(0+.3+)

<b>Time:</b>	Using cue 01	142 milliseconds	} approximately.
	Using cue 02	142 milliseconds	
	Using cue 03	128 milliseconds	

**Link:** Obeyed in 0.0. Not left in X1.

**Error:** See section 6 below.

### 1. Layout of the Table

The first word in each table must be stored at the start of a block; the contents of this block must be as follows:-

A.0	$N$	Maximum number of points needed for interpolation
.1		Not used, usually punched as +0
.2	$m_x$	Number of values of $x$
.3	$m_y$	Number of values of $y$
.4	$x_0$	Initial value of $x$
.5	$y_0$	Initial value of $y$
.6	$h_x$	$x_{i+1} - x_i$ , the interval in $x$ .
.7	$h_y$	$y_{i+1} - y_i$ , the interval in $y$ .

The values of  $f(x,y)$  are stored in the next  $m_x \cdot m_y$  locations, in  $m_x$  consecutive groups of  $m_y$  values. Thus if  $A_{00}$  is the address of  $f(x_0, y_0)$ , the address,  $A_{rs}$ , of  $f(x_r, y_s)$  is  $A_{00} + r \cdot m_y + s = A + 8 + r \cdot m_y + s$  where  $A$  is the address of the table.

$N-1$  denotes the order of interpolation which should be used to obtain maximum accuracy from the table.  $N$  is not used by R 322 but it should be inserted so that the table may be used by other interpolation routines.

For the same reason, although R 322 can deal with negative values of  $h$ , the values of both  $x$  and  $y$  should normally be in ascending order, making  $h_x$  and  $h_y$  positive.

### 2. The Index

If cue 01 is used an index of tables must be stored in the master programme. This index must be arranged in the store as shown below; it can most easily be formed by using R 102.

Location	Modifier	Counter
$B.P$	0	$t$
$B.P + 1$	$A_1$	$n_1$
$B.P + 2$	$A_2$	$n_2$
⋮	⋮	⋮
$B.P + t$	$A_t$	$n_t$

$A_1, A_2 \dots A_t$  are the addresses of the first words in each table.  $n_1, n_2, \dots, n_t$  are not used by R 322; they are used by other routines to specify the number of points to be used for interpolation.  $t$  specifies the number of tables listed in the index.

### 3. Preset Parameters

If cue 01 is to be used the address of the index must be specified by preset parameter 01. If cue 02 is used the index consists of one word containing  $(A, n)$ . The address of this word is specified by preset parameter 01. If no parameter list is read the address will be set at 2.0.

If the given point  $(x, y)$  is outside the boundary of the table the subroutine will jump to preset parameter 02. This may be the cue for an extrapolation routine or a loop stop in 1.2: if no parameter list is supplied it will be a loop stop in 1.2.

The parameter list will normally be punched as follows:-

R 0 0 -0 2	}	Title
322 - 04 -		
B - P0 0.	}	Address B.P of the index
0		
( 1.2 0 60 )	}	Loop Stop or cue to extrapolation routine.
0		

### 4. Extrapolation

On obeying the cue to an extrapolation routine the contents of the following registers may be useful.

- X1  $l_p = 6$  if  $x$  (or both  $x$  and  $y$ ) are out of range  
 $l_p = 7$  if only  $y$  is out of range  
 The order 

N	1	66
---	---	----

 may be used to determine which variable is out of range.
- X5  $s_m =$  Address of the table.
- X7  $q < 0$  if  $\frac{x}{h_x} < \frac{x_0}{h_x}$  or  $\frac{y}{h_y} < \frac{y_0}{h_y}$   
 $q \geq 0$  if  $\frac{x}{h_x} > \frac{x_{m-1}}{h_x}$  or  $\frac{y}{h_y} > \frac{y_{m-1}}{h_y}$
- U0.0 Link
- U0.1 Given value of  $x$
- U0.2 Given value of  $y$
- U2 First Block of the table.

### 5. Method

The programme selects tabular arguments  $x_r$  and  $y_s$  such that

$$x_r \leq x < x_{r+1} \quad y_s \leq y < y_{s+1}$$

It finds

$$\delta x = \frac{x - x_r}{h_x} \quad \delta y = \frac{y - y_s}{h_y}$$

then

$$f(x_r, y) = (1 - \delta y)f(x_r, y_s) + \delta y f(x_r, y_{s+1})$$

$$f(x_{r+1}, y) = (1 - \delta y)f(x_{r+1}, y_s) + \delta y f(x_{r+1}, y_{s+1})$$

$$f(x, y) = (1 - \delta x) f(x_r, y) + \delta x f(x_{r+1}, y)$$

### 6. Error

Slight inaccuracy may arise due to the fact that  $h$ , the tabular interval, is stored in binary form. An exact decimal interval may be represented in binary with an error of up to  $2^{-39}$ . This may lead to a maximum error of  $(m - 1) \cdot 2^{-39}$  in estimating the position of an argument near the upper boundary of the table. If the interval were exact the error in the interpolation process would not exceed  $3 \cdot 2^{-38}$ . These errors will usually be negligible compared with that introduced by using a linear approximation.

It is possible for an argument which is just within the range of the original table to be rejected as just outside the range of the binary table in the computer. This type of error can be eliminated by rounding  $h$  upwards, but this has the effect of increasing the first error described in the previous paragraph.

### 7. Loop Stop

1.2	1.2 0 60
	0

$x$  or  $y$  outside the range of the table; no extrapolation routine provided.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
31.12.56.

## TWO WAY POLYNOMIAL INTERPOLATION

$$p' = f(p,q)$$

This subroutine estimates the value of  $f(x,y)$  for given  $x$  and  $y$ , by interpolating in a table. The table must have a rectangular boundary and equal intervals in  $x$  and in  $y$ . The given point  $(x,y)$  must be within or on the boundary of the table.  $x$  and  $y$  must be set in accumulators 6 and 7 respectively; the answer  $f(x,y)$  is left in accumulator 6.

**Name:** 2 WAY POL. INT

**Store:** 10 Blocks.

**Uses:** U0,1,2,3; X5,6; B0.

**Cues:** 01 To interpolate in table number  $k$ , where  $k$  is set in  $5_c$ . The number (0+.2+) of points to be used for interpolation and the address of the table are given by the  $k^{\text{th}}$  entry in the index.

02 To interpolate in the table specified by the single word index.  
(0+.2)

03 To interpolate in the table whose address is set in  $5_m$ . The number (0+.4+) of points to be used for interpolation is set in  $5_c$ .

**Time:** Using  $n$  values of  $x$  and of  $y$  (i.e. using  $(n-1)$ th order interpolation in both directions), the time would be approximately

$$5.2n^3 + 6n^2 + 32n + 170 \text{ milliseconds.}$$

Number of Points  $n$

Time in milliseconds

2	3	4	5	6	7	8
270	460	720	1150	1680	2430	3470

**Link:** Obeyed in 1.4 and left unaltered in X1.

**Error:** See section 7 below.

### 1. Layout of the Table

The first word in each table must be stored at the start of a block; the contents of this block must be as follows:-

A.0	N	Maximum number of points needed for interpolation
.1		Not used, usually punched as +0.
.2	$m_x$	Number of values of $x$
.3	$m_y$	Number of values of $y$
.4	$x_0$	Initial value of $x$
.5	$y_0$	Initial value of $y$
.6	$h_x$	$x_{i+1} - x_i$ , the interval in $x$ .
.7	$h_y$	$y_{i+1} - y_i$ , the interval in $y$ .

The values of  $f(x, y)$  are stored in the next  $m_x \cdot m_y$  locations, in  $m_x$  consecutive groups of  $m_y$  values. Thus if  $A_{00}$  is the address of  $f(x_0, y_0)$ , the address  $A_{rs}$  of  $f(x_r, y_s)$  is

$$A_{00} + r \cdot m_y + s = A + 8 + r \cdot m_y + s$$

where  $A$  is the address of the table.

$N-1$  denotes the order of interpolation which should be used to obtain maximum accuracy from the table. The number of points actually used for interpolation in each direction,  $n'$ , will be the lesser of  $N$  and  $n$ , where  $n$  is specified in the index or on entry to the subroutine.

The values of  $x$  and  $y$  must be in ascending order and the intervals  $h_x$  and  $h_y$  are therefore positive.

### 2. Scaling

The table must be scaled so that

$$-1.0 \leq f(x, y) < +1.0$$

for all values of  $x$  and  $y$  within the range of the table.

During the interpolation process the subroutine evaluates  $f(x_i, y)$  for  $n'$  consecutive tabular values of  $x_i$  and also for the given value of  $x$ . If the function exceeds capacity at any one of these points there will be a loop stop in 2.7+.

The overflow indicator will always be clear on exit from the subroutine.



### 3. The Index

If cue 01 is used an index of tables must be stored in the master programme. This index must be arranged in the store as shown below; it can most easily be formed by using R 102.

Location	Modifier	Counter
$B.P$	0	$t$
$B.P + 1$	$A_1$	$n_1$
$B.P + 2$	$A_2$	$n_2$
⋮	⋮	⋮
$B.P + t$	$A_t$	$n_t$

$t$  specifies the number of tables listed in the index.  $A_1, A_2, \dots, A_t$  are the addresses of the first words in each table.  $n_1 - 1, n_2 - 1, \dots, n_t - 1$  are the orders of interpolation to be used.  $n_t$  therefore represents the number of tabular values of  $x$  and of  $y$  used for interpolation in the  $t^{\text{th}}$  table.

### 4. Preset Parameters

If cue 01 is to be used, the address of the index must be specified by preset parameter 01. If cue 02 is used the index consists of one word containing  $(A, n)$ ; the address of this word is specified by preset parameter 01. If no parameter list is read the address will be set at 2.0.

If the given point  $(x, y)$  is outside the boundary of the table the subroutine will jump to preset parameter 02. This may be the cue for an extrapolation routine or a loop stop in 0.0: if no parameter list is supplied it will be a loop stop in 0.0.

The parameter list will normally be punched as follows:-

R 0 0 -0 2	}	Title
323 - 04 -		
B - P0 0.	}	Address $B.P$ of the index
0		
(0.0 0 60)	}	Loop stop or cue to extrapolation routine
( 0 )		

### 5. Extrapolation

On obeying the cue to an extrapolation routine the contents of the following registers may be of use:

X2  $2p = 6$  if  $x$  (or both  $x$  and  $y$ ) are out of range.  
 $2p = 7$  if only  $y$  is out of range.

The order 

$N$	2	66	
-----	---	----	--

 may be used to determine which variable is out of range.

X3  $n'$ , the number of points to be used for interpolation.

X5  $(A, n)$

X7  $q < 0$  if  $x < x_0$  or  $y < y_0$   
 $q \geq 0$  if  $x > x_{m-1}$  or  $y > y_{m-1}$

U0.0 Given value of  $x$

U0.1 Given value of  $y$

U3 First Block of the Table

B0 The accumulators, except for X5, as stored on entry. B0.5 will contain  $(A, n)$ . The link will be in B0.1.

### 6. Method

The number of points used for interpolation in each direction is  $n'$ , which is the lesser of  $N$  (specified in the table) and  $n$  (specified on entry to the subroutine.)

$n'$  tabular values of  $x_i$  and of  $y_i$  are chosen so that they are as nearly symmetrical as possible about the given values of  $x$  and  $y$ . Provision is made for points which lie on or near the boundary of the table.

Using  $(n' - 1)$ th order interpolation the subroutine evaluates  $f(x_i, y)$  for each of the  $n'$  selected values of  $x_i$  and for the given value of  $y$ . Finally it applies the same process to these  $n'$  values to evaluate  $f(x, y)$

The method used is Neville's iteration process, as described below:

Let the  $n'$  arguments be  $x_s, x_{s+1}, \dots, x_{s+n-1}$

and the  $n'$  functions  $f_s^{(0)}, f_{s+1}^{(0)}, \dots, f_{s+n-1}^{(0)}$

The recurrence relation,

$$f_{s+i}^{(t+1)} = \frac{1}{x_{s+i+t+1} - x_{s+i}} \begin{vmatrix} f_{s+i}^{(t)} & x_{s+i} - x \\ f_{s+i+1}^{(t)} & x_{s+i+t+1} - x \end{vmatrix}$$

is applied with  $t = 0, \quad i = 0, 1, \dots, n' - 2$   
 $t = 1, \quad i = 0, 1, \dots, n' - 3$

in general, with  $t = t \quad i = 0, 1, \dots, n' - 2 - t$

& finally with  $t = n' - 2, \quad i = 0.$

$f_s^{(n'-1)}$  is the required  $(n'-1)$ th order interpolate.

Since  $x_{i+1} - x_i = h$  is constant the above recurrence relation can be re-written

$$f_{s+i}^{(t+1)} = \frac{1}{(t+1)h} \left| \begin{array}{cc} f_{s+i}^{(t)} & x_{s+i} - x \\ f_{s+i+1}^{(t)} & x_{s+i} - x + (t+1)h \end{array} \right|$$

$$= \frac{\left( f_{s+i+1}^{(t)} - f_{s+i}^{(t)} \right) \left( x - x_{s+i} \right)}{(t+1)h} + f_{s+i}^{(t)}$$

It is this last formula which is used in the subroutine.

## 7. Error

The rounding errors in the interpolation process will not exceed  $2n' \cdot 2^{-38}$  for a well behaved function. For functions which are not well behaved the round off error will be small compared with the truncation error introduced by fitting an  $(n' - 1)^{\text{th}}$  order polynomial.

Slight inaccuracy may arise due to the fact that  $h$ , the tabular interval, is stored in binary form. An exact decimal interval may be represented in binary with an error of up to  $2^{-39}$ . This may lead to a maximum error of  $(m - 1) \cdot 2^{-39}$  in estimating the position of an argument near an upper boundary of the table.

It is possible for an argument which is just below the upper boundary of the original table to be rejected as just above the boundary of the binary table in the computer. This type of error can be eliminated by rounding  $h_x$  and  $h_y$  upwards, but this has the effect of increasing the error described in the previous paragraph.

## 8. Order of Interpolation

The order of interpolation required to keep the truncation error below a given value may be estimated as follows. Let the maximum error permissible in the function be defined as the unit of measurement. The differences of the function, measured in these units, should satisfy the following conditions:

$$\text{For } n' = \begin{array}{|c|c|c|c|c|c|c|} \hline 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ \hline \delta^{n'} x + \delta^{n'} y < & 4 & 60 & 20 & 500 & 100 & 3500 & 400 \\ \hline \end{array}$$

$\delta^{n'}$  is the maximum  $n'^{\text{th}}$  order difference in the range of the table.

## 9. Error Stops

0.0	0.0	0	60
	0		

Loop stop if  $x$  or  $y$  outside the range of the table; no extrapolation routine provided.

0.5

0	3	72	5
1+	1	72	

Writing with OVR stop due to 73 order in 0.4+. Caused by entering subroutine with OVR set or by incorrect setting of index address.

1.2

1.2	1	62	
7	1	41	

Loop stop if  $n' > 8$

1.3

1.3	1	63	
1.2	2	00	

Loop stop if  $n' < 2$

2.7+

1.0	0	64	
2.7+	0	60	

Loop stop if  $f(x_i, y) \geq 1.0$   
or  $< 1.0$   
for some value  $x_i$ .

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
28.2.57.

## FLOATING POINT POLYNOMIAL INTERPOLATION

$$p' = f(p)$$

where  $p$  and  $p'$  are in floating-point form.

This subroutine estimates the value of  $f(x)$ , for given  $x$ , by interpolating in a table.  $f(x)$  must be tabulated at equal intervals of  $x$ , and the given  $x$  must be within the range of the table. The order of interpolation must not exceed seven.

**Name:** F.P. INTERPOLATION

**Store:** 9 blocks plus 4 blocks for R 610.

**Uses:** B0 and the entire Computing Store except X1, 2, 3, 4, 5.

**Cues:** 01 To interpolate in table number  $q$ , where  $q$  is set in  $7_c$ . The number  
(0+.1+) of points to be used for interpolation and the address of the table are given by the  $q^{\text{th}}$  entry in the index.

02 To interpolate in the table specified by the single word index.  
(0+.1)

03 To interpolate in the table whose address is set in  $7_m$ . The number  
(0+.3+) of points to be used for interpolation is set in  $7_c$ .

**Time:** Approximately  $48n^2 - 9n + 150$  milliseconds, where  $n$  is the number of points used for interpolation.

Number of points  $n$

Time in milliseconds

2	3	4	5	6	7	8
325	555	880	1300	1825	2440	3150

**Link:** Obeyed in 2.0 and left unaltered in X1.

**Error:** See Section 6 below.

## 1. LAYOUT OF THE TABLES

The first word in each table must be stored in the first word of a block; the contents of this block must be as follows:-

A.0	$N$	= Maximum number of points needed for interpolation.
.1	$m$	= Number of values of $f(x)$ tabulated.
.2	$x_0$	= Smallest value of $x$ in the table.
.3	$h$	= $(x_{i+1} - x_i)$ , the tabular interval.
.4	$f(x_0)$	} Successive values of the function in $m$ consecutive locations.
.5	$f(x_1)$	
.6	$f(x_2)$	
	etc.	

$N$  and  $m$  are used as counters and must therefore be stored as fixed-point integers. All other numbers in the table must be stored as floating-point numbers. This will be arranged by R 103 if it is used to read in the tables. The number of digits in the exponent is specified by the parameter of R 610; a description of the form in which the floating-point numbers are to be stored is given in the specification of R 610.

$N-1$  denotes the order of interpolation which must be used to obtain maximum accuracy from the table. The number of points actually used for interpolation will be the lesser of  $N$  and  $n$ , where  $n$  is specified in the index or on entry to the subroutine.

The values of  $x$  must be in ascending order and the interval  $h$  is therefore always positive.

## 2. THE INDEX

If cue 01 is used an index of tables must be stored in the master programme. This index must be arranged in the store as shown below; it can most easily be formed by using R 103:-

Location	Modifier	Counter
$B.P$	0	$t$
$B.P + 1$	$A_1$	$n_1$
$B.P + 2$	$A_2$	$n_2$
⋮	⋮	⋮
$B.P + t$	$A_t$	$n_t$

$A_1, A_2, \dots, A_t$  are the addresses of the first words in each table.  $n_1, n_2, \dots, n_t$  denote the number of points to be used for interpolation.  $t$  specifies the number of tables listed in the index.

### 3. PRESET PARAMETERS

If cue 01 is to be used the address of the index must be specified by preset parameter 01. If cue 02 is used the index consists of one word containing  $(A, n)$ ; the address of this word is specified by preset parameter 01. If no parameter list is read the address will be set at 2.0.

If the given value of  $x$  is outside the range of the table the subroutine will jump to preset parameter 02. This may be the cue for an extrapolation routine or a loop stop in 0.5: if no parameter list is supplied it will be a loop stop in 0.5.

The parameter list will normally be punched as follows:-

R 0 0 -0 2	
324 - 04 -	
B - P0 0.	} Address B.P of the index
0	
(0.5 0 60)	} Loop stop or cue to extrapolation routine.
0	

### 4. EXTRAPOLATION

On obeying the cue to an extrapolation routine the contents of the following registers may be of use:

X2 Unchanged since entry

X3  $x - x_0$

X7  $q < 0$  if  $x < x_0$

$q > 0$  if  $x > x_{m-1}$

U1.7  $(A + 0.4, n)$

U2 First block of the table

U3, 4 and 5 R 610

B0 The accumulators, except for X7, as stored on entry. B0.7 will contain  $(A, n)$ . The link will be in B0.1.

### 5. METHOD

The number of points used for interpolation is  $n'$ , which is the lesser of  $N$  (specified in the table) and  $n$  (specified on entry to the subroutine).

The method used is Neville's iteration process. The  $n'$  points in the table are chosen so that the values of  $x_i$  are as nearly symmetrical as possible about the given value of  $x$ . Provision is made for points which lie on or near the boundary of the table.

Issue 2

Let the  $n'$  arguments be  $x_s, x_{s+1}, \dots, x_{s+n'-1}$

and the  $n'$  functions  $f_s^{(0)}, f_{s+1}^{(0)}, \dots, f_{s+n'-1}^{(0)}$

The recurrence relation,

$$f_{s+i}^{(t+1)} = \frac{1}{x_{s+i+t+1} - x_{s+i}} \begin{vmatrix} f_{s+i}^{(t)} & x_{s+i} - x \\ f_{s+i+1}^{(t)} & x_{s+i+t+1} - x \end{vmatrix}$$

is applied with  $t = 0, \quad i = 0, 1, \dots, n' - 2$   
 $t = 1, \quad i = 0, 1, \dots, n' - 3$

in general, with  $t = t, \quad i = 0, 1, \dots, n' - 2 - t$

& finally with  $t = n' - 2, \quad i = 0$ .

$f_s^{(n'-1)}$  is the required  $(n'-1)^{\text{th}}$  order interpolate.

Since  $x_{i+1} - x_i = h$  is constant the above recurrence relation can be re-written

$$f_{s+i}^{(t+1)} = \frac{1}{(t+1)h} \begin{vmatrix} f_{s+i}^{(t)} & x_{s+i} - x \\ f_{s+i+1}^{(t)} & x_{s+i} - x + (t+1)h \end{vmatrix}$$

$$= \frac{(f_{s+i+1}^{(t)} - f_{s+i}^{(t)})(x - x_{s+i})}{(t+1)h} + f_{s+i}^{(t)}$$

It is this last formula which is used in the subroutine.

## 6. ERROR

For a well behaved function the errors in the interpolation process will not exceed  $n'$  in the last binary place of the argument of  $f(x)$ . In the standard form of floating-point storage, with nine binary digits for the exponent, the error in  $f(x)$  will not exceed  $0.000005(n'\%)$ . For functions which are not well behaved the rounding errors will be small compared with the truncation error introduced by fitting an  $(n' - 1)^{\text{th}}$  order polynomial.

Slight inaccuracy may arise because  $h$ , the tabular interval, is stored in floating-point form. In the standard form of floating-point storage, with nine binary digits in the exponent, the error in  $h$  should not exceed  $0.000005\%$ . This may lead to a maximum error of  $0.000005(m-1)\%$  in estimating the position of an argument near the upper boundary of the table.

It is possible for an argument, which is just below the upper boundary of the original table, to be rejected as just above the floating-point table in the computer. This type of error can be eliminated by rounding  $h$  upwards, but this has the effect of increasing the error described in the previous paragraph.

## 7. ORDER OF INTERPOLATION

The order of interpolation required to keep the truncation error below a given value may be estimated from the differences. Let the maximum error permissible



in the function be defined as the unit of measurement. The differences of the function, measured in these units, should satisfy the following conditions

For $n' =$	2	3	4	5	6	7	8
$\delta^{n'} <$	4	60	20	500	100	3500	400

$n'$  is the number of points used for interpolation

$\delta^{n'}$  is the maximum  $n'$ <sup>th</sup> order difference in the range of the table.

### 8. ERROR STOPS

0.4	0 <span style="border: 1px solid black; padding: 0 2px;">2</span> 72 7 N <span style="border: 1px solid black; padding: 0 2px;">3</span> 72	Writing with OVR stop due to 73 order in 0.3+. Caused by entering with OVR set or possibly by incorrect index address.
0.4	0.4 6 62 <span style="border: 1px solid black; border-radius: 50%; padding: 0 2px;">7</span> 6 41	Loop stop if $n' > 8$
0.5	0.5 6 63 0 7 00	Loop stop if $n' < 2$
0.5	0.5 0 60 <hr style="border: 0.5px solid black;"/> 0	Loop stop if given $x$ out of range and no extrapolation routine provided.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
7. 8. 58.

## POLYNOMIAL INTERPOLATION (UNEQUAL INTERVALS)

$$p' = f(p)$$

This subroutine estimates the value of  $y = f(x)$ , for given  $x$ , by interpolating in a table of values of  $f(x)$  given at values of  $x$  which are not necessarily equidistant. The given value of  $x$  must lie within the range of the table, and the order of interpolation must not exceed 7. If  $f(x)$  is a strictly increasing function of  $x$  the subroutine may also be used for inverse interpolation, i.e., to estimate  $x$  for a given value of  $y$ .

**Name:** POLYNOMIAL INTERPOLATION (UNEQUAL INTERVALS) MK 3

**Store:** 8 blocks.

**Uses:** U0, 1, 2, 3; X6, 7; B0

**Cues:** 01 to interpolate in table number  $q$ , where  $q$  is set as  $2^{-38} q$  in X7.  
(0+.1) The number of points to be used for interpolation and the address of the table are given by the  $q^{\text{th}}$  entry in the index.

The same cue may be used to interpolate in a table specified by a single word index by setting X7 = 0 before entry.

02 to interpolate in the table whose address is set in  $7_m$ . The  
(0+.3) number of points to be used for interpolation is set in  $7_c$ .

**Time:** Entering by cue 02 the average time is

$$1.25m + 6n^2 + 4.5n + 120 \text{ milliseconds}$$

where  $m$  = no. of values in table

$n$  = no. of points used for interpolation.

Entering by cue 01 would add 12 milliseconds.

**Link:** Set in X1 and obeyed in 1.7. *It must be a 'go' order-pair.*

To interpolate directly (i.e., to estimate  $y$  for given  $x$ ) the link must be set by a 00 order in X1.

To interpolate inversely (i.e., to estimate  $x$  for given  $y$ ) the link must be set by a 02 order (i.e., negatively) in X1.

In either case the link is left as set in X1.

### 1. Layout of Tables

The first word in each table must be stored in the first location of a block. The layout of the table must be as follows.

A.0		$m$	Number of values of $x$ tabulated = number of values of $y$ tabulated Maximum number of points needed for interpolation of $y$ given $x$ . Maximum number of points needed for interpolation of $x$ given $y$ .
.1		$N_x$	
.2		$N_y$	
.3	$x_0$		Values of $x$ in $m$ consecutive locations.
.4	$x_1$		
	$x_{m-1}$		
	$y_0$		
	$y_1$		Corresponding values of $y$ in $m$ consecutive locations.
	$y_{m-1}$		

The values of  $x$  must be in strictly increasing order. If the corresponding values of  $y$  are also in strictly increasing order the table may be used for inverse interpolation, i.e. for computing  $x$  given  $y$ . The values of  $x$  must occupy consecutive locations, as also must the values of  $y$ , the first value of  $y$  following immediately after the last value of  $x$ .

$N_x - 1$ , or  $N_y - 1$  denotes the order of interpolation necessary to obtain maximum accuracy from the table when interpolating with respect to  $x$  and  $y$  respectively. The actual number of points used for interpolation is the lesser of  $n$  and  $N_x$  or  $N_y$ , where  $n$  is specified in the index or on entry to the subroutine.

If the table cannot be used for inverse interpolation (e.g., if the values of  $y$  are not in strictly increasing order) the value of  $N_y$  is irrelevant. It is recommended that in this case  $N_y$  be put equal to zero, in which case a loop stop (see paragraph 8) will occur should inverse interpolation be attempted.

### 2. Scaling

The table must be scaled so that all values of  $x$  and  $y$  within the range of the table satisfy the restrictions

$$-1.0 \leq x < 1.0$$

$$-1.0 \leq y < 1.0$$

Provision has been made for temporary scaling to prevent overflow occurring during the working of the subroutine. The overflow indicator will be set on exit from the subroutine only if the final interpolate ( $x$  or  $y$ ) does not lie in the range

$$-1 \leq y < 1 \quad \text{or} \quad -1 \leq x < 1$$

### 3. Index

If cue 01 is used an index of tables must be stored in the master programme. This index must be arranged in the store as shown below; it can most easily be formed by using R 102.

Location	Modifier	Counter
$B.P$	0	$t$
$B.P + 1$	$A_1$	$n_1$
$B.P + 2$	$A_2$	$n_2$
⋮	⋮	⋮
$B.P + t$	$A_t$	$n_t$

$A_1, A_2, \dots, A_t$  are the addresses of the first words in each table.  $n_1, n_2, \dots, n_t$  denote the number of points to be used for interpolation.  $t$  specifies the number of tables listed in the index.

Alternatively a single word index may be stored as

$$(A, n)$$

where  $A$  is the address of the first word in the table and  $n$  the number of points to be used for interpolation.

### 4. Preset Parameters

If cue 01 is to be used the address of the index must be specified by preset parameter 01. If no parameter is supplied the address will be set at 2.0.

If the given value of  $x$  (or  $y$  in the case of inverse interpolation) is outside the range of the table the subroutine will jump to preset parameter 02. This may be the cue to an extrapolation routine or a loop stop in 3.7. If no parameter list is supplied it will be a loop stop in 3.7.

The parameter list will normally be punched as follows

R 0 0 -0 2	}	Title
327 - 04 -		
$B - P0 0.$	}	Address $B.P$ of index.
0		
$(\frac{3.7 0 60}{0})$	}	Loop stop or cue to extrapolation routine
0		

### 5. Extrapolation

On obeying the cue to an extrapolation routine the contents of the following registers may be useful: (for inverse interpolation interchange  $x$  and  $y$ ).

X2  $(A(x_0), n)$

X6  $\frac{1}{2}x$

X7  $\frac{1}{2}(x-x_0)$  if  $x < x_0$  (Negative)

$\frac{1}{2}(x-x_{m-1})$  if  $x > x_{m-1}$  (Positive)

U0 Block of the table containing  $x_0$  if  $x < x_0$

Block of the table containing  $x_{m-1}$  if  $x > x_{m-1}$

U1.0  $n'$

U1.1  $m$

B0 Accumulators, except for X7, as stored on entry. B0.7 will contain  $(A, n)$ . The link will be as set in B0.1 (for inverse interpolation the link is set negatively).

### 6. Method

The number  $n'$  of points used for interpolation is the lesser of  $N_x$  or  $N_y$  (specified in the table) and  $n$  (specified on entry to the subroutine).

The method used is Neville's iterative process. In the following it is assumed that a value of  $y$  is to be estimated for a given value of  $x$ .

$n'$  points in the table are chosen so that there are, as near as possible, an equal number of points on either side of the given value of  $x$ . When  $n'$  is odd there will, where possible, be  $\frac{1}{2}(n' + 1)$  points less than  $x$  and  $\frac{1}{2}(n' - 1)$  points greater than  $x$ . Provision has been made for points which lie on or near the boundary of the table.

Let the  $n'$  arguments be  $x_S, x_{S+1}, \dots, x_{S+n'-1}$  and denote the corresponding functions by

$$y_S^{(0)}, y_{S+1}^{(0)}, \dots, y_{S+n'-1}^{(0)}$$

The recurrence relation

$$y_{S+i}^{(t+1)} = \frac{1}{x_{S+i+t+1} - x_{S+i}} \begin{vmatrix} y_{S+i}^{(t)} & x_{S+i} - x \\ y_{S+i+1}^{(t)} & x_{S+i+t+1} - x \end{vmatrix}$$

is applied with  $i = 0, 1, \dots, n'-2-t$  for each value of  $t$ , from  $t = 0$  to  $t = n'-2$

$y_S^{(n'-1)}$  is the required  $(n'-1)^{\text{th}}$  order interpolate.

### 7. Error

The round off errors in the interpolation process should not exceed  $n'.2^{-36}$  for a well behaved function. For functions which are not well behaved the round off error will be small compared with the truncation error introduced by fitting an  $(n'-1)^{\text{th}}$  order polynomial.

Accuracy will be lost if there is a very large variation in the magnitude of  $\frac{dy}{dx}$  within the range of the  $n'$  points used for interpolation. In extreme cases, when  $\frac{dy}{dx}$  is tending to infinity, all significant figures may be lost and the result will be meaningless.

Care should therefore be exercised when interpolating from tables of a function whose derivative varies greatly in magnitude over the range of the table. In particular if the derivative becomes very large in the range care must be taken when interpolating directly and using points near this region. Similarly if the derivative becomes zero or very small in the range, care must be taken when interpolating inversely.

### 8. Error Stops

0.3	0	7	73
	0	3	72 7

Writing with OVR if OVR set on entry.  
A stop may also occur here if the index address is incorrect.

1.5	1.5	1	62
	7	1	41

Loop stop if  $n' > 8$

1.6	1.6	1	63
	3	1	41

Loop stop if  $n' < 2$ . This stop will occur if inverse interpolation is attempted when  $N_y < 2$ .

1.4	3.7	0	60
	0		

Loop stop if  $x$  (or  $y$ ) outside the range of the table and no extrapolation routine provided.

Author: Mr. G.H.L. Buxton of Sir W.G. Armstrong Whitworth Aircraft Ltd.

FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

## TWO WAY POLYNOMIAL INTERPOLATION (UNEQUAL INTERVALS)

$$p' = f(p, q)$$

This subroutine estimates the value of  $z = f(x, y)$ , for given  $x$  and  $y$ , using a table of values of  $f(x, y)$  and fitting polynomials of degree specified in  $x$  and  $y$  directions. The table may be given at values of  $x$  and  $y$  not necessarily equidistant. The given point  $(x, y)$  must lie within the range of the table and the orders of interpolation (not necessarily equal) must not exceed 7.

Name: 2-WAY POLY. INT  $\neq$

Store: 15 blocks.

Uses: U0, 1, 2, 3, 4 ; X5, 6, 7 ; B0, 1.

Cue: 01 (0+. 0)

Time: The time,  $t$ , taken by the subroutine varies according to the size of the table, and is slightly longer for values near the top end. An approximate value for  $t$  is given by:-

$$t = 5.5u_x^2 + 5.5u_y^2 + 18u_xu_y + 75u_x + 45u_y + 460 \text{ milliseconds}$$

where  $u_x$  and  $u_y$  are respectively the number of values of  $x$  and  $y$  used in interpolation.

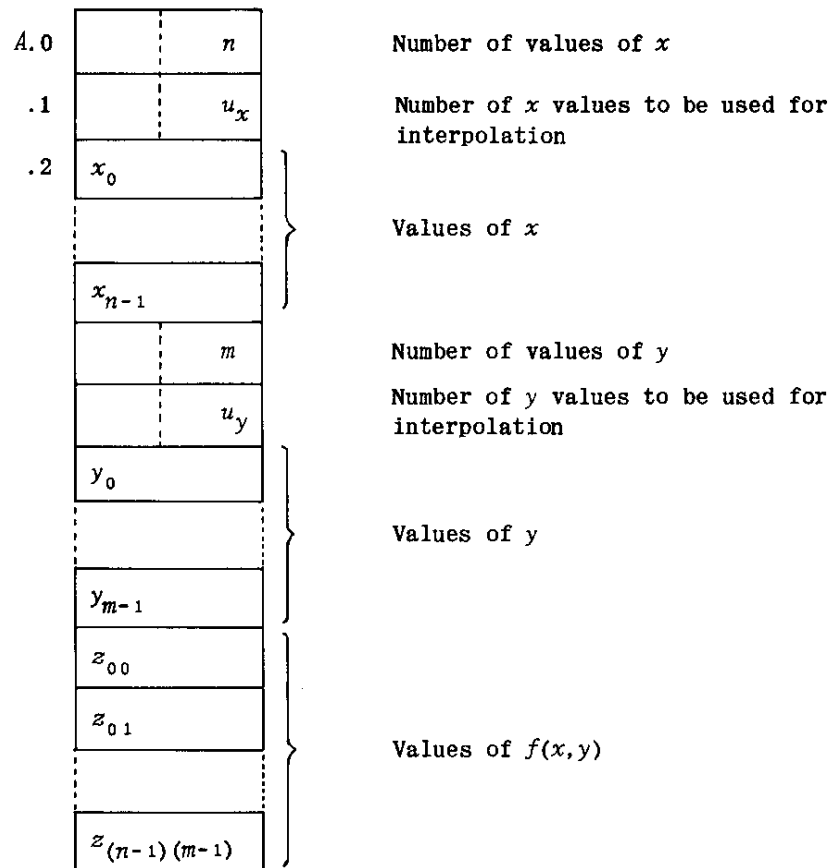
Link: Obeyed in 4.2 and left in X1 on exit.

### 1. Entry

On entry to the subroutine, the address of the table must be set in  $5_m$ , and  $5_c$  must be clear.  $x$  and  $y$  should be in X6 and X7 respectively.

### 2. Layout of the Table

The table must be stored in consecutive locations with the first word at the beginning of a block, as follows:-



The values of  $f(x,y)$  are stored in  $n$  consecutive groups of  $m$  values such that if  $A_{00}$  is the address of  $f(x_0, y_0)$ , the address  $A_{rs}$  of  $f(x_r, y_s)$  is

$$A_{00} + rm + s = A + m + n + 4 + rm + s$$

where  $A$  is the address of the table.

The order of interpolation used is  $u_x - 1$  in the  $x$  direction and  $u_y - 1$  in the  $y$  direction.

The table occupies  $mn + m + n + 4$  locations altogether.

### 3. Restrictions

- $|x_i - x_j| < 1$ ,  $|y_i - y_j| < 1$ ,  $|z_i| < 1$  for any  $i, j$ .
- The family of curves must all be continuous.
- $z$  must be a single-valued function of  $y$  for any given  $x$ .
- $z$  must be a single-valued function of  $x$  for any given  $y$ .
- $2 \leq u_x \leq 8$ ,  $2 \leq u_y \leq 8$ .

### 4. Preset Parameter

If the given point  $(x,y)$  is outside the boundary of the table the subroutine will jump to preset parameter 01. This may be the cue to an extrapolation routine or a loop stop in 1.2 : if no parameter list is supplied it will be a loop stop in 1.2.



The parameter list should be punched as follows:

R 0 0 -0 1
328 - 04 -
$\left( \begin{array}{c} 1.2 \quad 0 \quad 60 \\ \hline 0 \end{array} \right)$

Loop stop or  
cue to extrapolation routine

### 5. Extrapolation

On obeying the cue to an extrapolation routine the contents of the following registers may be of use:

X5 <sub>m</sub>	Address of table.
X6	$x - x_0$
U0.0	$x$
U0.1	$y$
U4	First block of table.
B0	The accumulators as stored on entry.
B1	U5 as stored on entry.

### 6. Method

The method is equivalent to fitting a polynomial of degree  $u_y - 1$  over the chosen range of  $y$  for each in turn of the chosen values of  $x$ , i.e. finding  $z_i = f(x_i, y)$ , and using these values to fit a polynomial of degree  $u_x - 1$  over the chosen range of  $x$ , giving  $z = f(x, y)$ .

Using the following notation:

$x_0$	.....	$x_{n-1}$	values of $x$ supplied
$y_0$	.....	$y_{m-1}$	values of $y$ supplied
$x^{(0)}$	.....	$x^{(u_x-1)}$	values of $x$ used in interpolation
$y^{(0)}$	.....	$y^{(u_y-1)}$	values of $y$ used in interpolation

$$[z_{it}] = \left[ \begin{array}{c} z_{(0,0)} \text{-----} z_{(0,m-1)} \\ \vdots \\ z_{(n-1,0)} \text{-----} z_{(n-1,m-1)} \end{array} \right] \quad \text{values of } z \text{ supplied}$$

$$[z^{(it)}] = \left[ \begin{array}{c} z^{(0,0)} \text{-----} z^{(0,u_y-1)} \\ \vdots \\ z^{(u_x-1,0)} \text{-----} z^{(u_x-1,u_y-1)} \end{array} \right] \quad \text{values of } z \text{ used in interpolation}$$

$$\prod_i = \prod_{j \neq i} \frac{x - x^{(j)}}{x^{(i)} - x^{(j)}} \quad \begin{matrix} j = 0 \dots u_x - 1 \\ i = 0 \dots u_x - 1 \end{matrix}$$

$$\prod_t = \prod_{s \neq t} \frac{y - y^{(s)}}{y^{(t)} - y^{(s)}} \quad \begin{matrix} s = 0 \dots u_y - 1 \\ t = 0 \dots u_y - 1 \end{matrix}$$

$$z_i = \sum_{t=0}^{u_y-1} z^{(it)} \prod_t \quad i = 0 \dots u_x - 1$$

the method is as follows:

- (a) Form  $\left. \begin{matrix} x^{(0)} \dots x^{(u_x-1)} \\ y^{(0)} \dots y^{(u_y-1)} \end{matrix} \right\}$  surrounding the given values of  $\begin{matrix} x \\ y \end{matrix}$

Provision is made for  $x, y$  near the end of a range.

- (b) Form  $\prod_i, \prod_t$

- (c) Form  $z_i$

- (d) Then  $z = \sum_{i=0}^{u_x-1} z_i \prod_i$

**7. Error**

An accuracy of 9 decimal places should be achieved for a well-behaved family of curves, i.e. where there are no rapid changes in the slopes over the range considered.

The round off error will normally be small compared with the truncation error introduced by using a polynomial approximation.

**8. Error Stops**

0.0	0 7 73
	1 5 73

Writing with overflow stop caused by entering the subroutine with OVR set.

1.2	1.2 0 60
	0

Loop stop if  $x$  or  $y$  is outside the range of the table and no extrapolation routine is provided.

**Author:** Mr. R.S. Neave of the De Havilland Engine Company

© FERRANTI LTD 1960

London Computer Centre,  
68, Newman Street,  
LONDON, W.1.

Not to be reproduced in whole or  
in part without the prior written  
permission of Ferranti Ltd.

Issue 1  
22nd December, 1960  
R.S.N. M.J.M.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
7. 10. 58.

## STRAIGHT LINE FITTING (ASYMMETRICAL)

Given a set of values of a function  $y$  and argument  $x$ , this routine fits a line of the form

$$y = a_0 + a_1 x$$

by the method of least squares. The values of the argument,  $x$ , need not necessarily be at equal intervals.

**Name:** ST. LINE FIT B  
**Store:** 4 blocks.  
**Uses:** U0, 1, 4, 5; X6, 7; B0.  
**Cue:** 01 (0+.0)  
**Time:** About  $(11n + 100)$  milliseconds.  
**Link:** Obeyed in 0.7 and left unaltered in X1.

## 1. METHOD OF USE

On entry the address of the first  $x$  must be in  $4_m$  and the address of the first  $y$  in  $5_m$ . The number of points,  $n$ , in the table must be in  $4_c$  and must satisfy  $2 \leq n \leq 127$ . The tables need not start at the beginning of a block.

The routine leaves  $a_0$  in X6 and  $a_1$  in X7.

## 2. OVERFLOW

The values of  $x$  and  $y$  should be scaled so that  $a_0$  and  $a_1$  are within capacity. If either  $a_0$  or  $a_1$  exceeds capacity an overflow routine is called in. The cue to the overflow routine is given by preset-parameter 01; if no parameter-list is provided it will be a loop stop in 0.4. The parameter-list should be punched as follows:-

R 0 0 -0 1
332 - 04 -
(0.4 0 60)
0

Loop stop or cue to  
overflow routine

R 332 restores the accumulators from B0 immediately before entering the overflow routine.

### 3. VALUE OF $n$

If  $n = 1$  the results will be meaningless and may overflow. If  $n = 0$  the routine will loop indefinitely between U0.0 and 0.7.

### 4. ERROR STOPS

0.0	0	7	73
	4	6	00

Writing with overflow if OVR set on entry.

0.4	0.4	0	60
	0		

Loop stop if  $a_0$  or  $a_1$  exceeds capacity and no overflow routine provided.

**Author:** Mr. H.P. Goodman of the De Havilland Aircraft Company.

© FERRANTI LTD 1958

*Not to be reproduced in whole or in part without the prior written permission of Ferranti Ltd.*

Ferranti Ltd., London Computer Centre, 21, Portland Place, LONDON W.1.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
7. 10. 58.

## LEAST SQUARES PARABOLA FITTING

Given a set of values of a function  $y$  and argument  $x$ , this routine fits a curve of the form

$$y = a_0 + a_1x + a_2x^2$$

by the method of least squares. The values of the argument  $x$  need not necessarily be at equal intervals.

**Name:** PARABOLA FITTING  
**Store:** 9 blocks.  
**Uses:** U0, 1, 3, 4, 5; X5, 6, 7; B0.  
**Cue:** 01 (0+.0)  
**Time:** About  $(280 + 20n)$  milliseconds.  
**Link:** Obeyed in 1.6 and left unaltered in X1.

## 1. METHOD OF USE

On entry the address of the first  $x$  must be in  $4_m$  and that of the first  $y$  in  $5_m$ . The number of points,  $n$ , in the table must be in  $4_c$  and must satisfy  $3 \leq n \leq 127$ .  $5_c$  is irrelevant. The tables need not start at the beginning of a block.

The routine leaves  $a_0$  in X5,  $a_1$  in X6 and  $a_2$  in X7.

## 2. METHOD

$$\text{Let } \alpha = \sum_{i=1}^n x_i \quad \beta = \sum_{i=1}^n x_i^2 \quad \gamma = \sum_{i=1}^n x_i^3 \quad \delta = \sum_{i=1}^n x_i^4$$

$$\lambda = \sum_{i=1}^n y_i \quad \mu = \sum_{i=1}^n x_i y_i \quad \nu = \sum_{i=1}^n x_i^2 y_i$$

Then  $a_0, a_1, a_2$  are the roots of:-

$$na_0 + \alpha a_1 + \beta a_2 = \lambda$$

$$\alpha a_0 + \beta a_1 + \gamma a_2 = \mu$$

$$\beta a_0 + \gamma a_1 + \delta a_2 = \nu$$

The quantities  $\alpha, \beta, \gamma, \delta, \lambda, \mu, \nu$  are first formed (suitably scaled). The equations are solved by evaluation of the various determinants concerned. Since the equations are liable to be ill conditioned, this part of the programme has been written double-length.

### 3. OVERFLOW

The scaling of the variables  $x$  and  $y$  should be chosen so that the values of  $a_0, a_1$  and  $a_2$  will be within capacity. If any of  $a_0, a_1, a_2$  are not within capacity an overflow routine is called in. The cue to the overflow routine is given by preset-parameter 01; if no parameter-list is provided it will be a loop stop in 1.3. The parameter-list should be punched as follows:-

R 0 0 -0 1
333 - 04 -
1.3 0 60
0

Loop stop or cue to overflow routine.

Note that X1, 2, 3, 4 are not usually restored on entry to the overflow routine, but they are always available in B0.1 to 0.4.

### 4. VALUE OF $n$

If  $n = 1$  or  $2$  the results will be meaningless and the overflow routine may be entered. If  $n = 0$  the routine loops indefinitely between U0.0 and 1.2.

### 5. ERROR STOPS

0.0	0 0 73
	4 6 00

Writing with overflow if OVR set on entry.

1.3	1.3 0 60
	0

Loop stop if  $a_0, a_1, a_2$  are not within capacity, and no overflow routine provided.

**Author:** Mr. H.P. Goodman of the De Havilland Aircraft Company.

© FERRANTI LTD 1958

*Not to be reproduced in whole or in part without the prior written permission of Ferranti Ltd.*

Ferranti Ltd., London Computer Centre, 21, Portland Place, LONDON W.1.

FERRANTI LTD

PEGASUS LIBRARY SPECIFICATION

Issue 1  
7.10.58.

STRAIGHT LINE FITTING (SYMMETRICAL)

Given a set of values of a function  $y$  and argument  $x$ , this routine fits a line of the form

$$ax + by = 2^{-v}$$

by the method of least squares. The values of the argument,  $x$ , need not necessarily be at equal intervals.

- Name: ST. LINE FIT
- Store: 4 blocks.
- Uses: U0, 1, 3, 4, 5; X6, 7; B0.
- Cue: 01 (0+.0)
- Time: About  $(13n + 100)$  milliseconds.
- Link: Obeyed in 0.6 and left unaltered in X1.

1. METHOD OF USE

On entry the address of the first  $x$  must be in  $4_m$  and the address of the first  $y$  in  $5_m$ . The number of points,  $n$ , in the table must be in  $4_c$  and the scale factor,  $v$  in  $5_c$ .  $n$  must satisfy  $2 < n < 127$ . The tables need not start at the beginning of a block.

The routine leaves  $a$  in X6 and  $b$  in X7.

The value of  $v$  should be chosen such that  $a$  and  $b$  are within capacity. The routine is not suitable for straight lines through or near the origin: in fact the perpendicular distance from the origin to the line must exceed  $2^{-v}$  so that  $a$  and  $b$  may satisfy  $-1.0 < a, b < +1.0$ : thus  $(\sqrt{x^2 + y^2})_{\min} > 2^{-v}$ .

2. OVERFLOW

If overflow occurs during the calculations an overflow routine is called in. The cue to the overflow routine is given by preset-parameter 01; if no parameter-list is provided it will be a loop stop in 1.6. The parameter-list should be punched as follows:-

R 0 0 -0 1
334 - 04 -
( 1.6 0 60 )
( 0 )

Loop stop or cue to overflow routine.

Note that X1, 2, 3, 4, 5 are not restored on entry to the overflow routine, but they are available in B0.1 to 0.5.

### 3. VALUE OF $n$

If  $n = 1$  the results will be meaningless and the overflow routine may be entered. If  $n = 0$  the routine will loop indefinitely between U0.0 and 1.0.

### 4. ERROR STOPS

0.0	0	7	73
	4	6	00

Writing with overflow if  
OVR set on entry.

1.6	1.6	0	60
	0		

Loop stop if  $n = 1$  or if  $a$  or  $b$   
exceeds capacity and no overflow  
routine provided.

**Author:** Mr. H.P. Goodman of the De Havilland Aircraft Company.

© FERRANTI LTD 1958

*Not to be reproduced in whole or  
in part without the prior written  
permission of Ferranti Ltd.*

Ferranti Ltd., London Computer Centre, 21, Portland Place, LONDON W.1.



## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
7.1.57.

## DETERMINATION OF THE ZERO OF A FUNCTION

This subroutine finds a zero of a function  $F(x)$ , by an iterative method, starting from two given values of  $x$  for which  $F(x)$  has opposite sign and between which at least one zero lies.

**Name:** ZERO OF F(X)  
**Store:** 5 blocks.  
**Uses:** X6; U0,1; B1 (to preserve the accumulators).  
**Cue:** 01 (3+.0; obeyed in U0.0).  
**Time:**  $50 + 2T + (56 + T)N$  milliseconds,  
 where  $T$  is the time of the auxiliary  
 and  $N$  is the number of iterations.  
**Link:** Obeyed in 0.7 and left in X1 on exit.

## 1. Method of Use

1.1 R 340 calls in, as required, an auxiliary subroutine to evaluate the function  $F(x)$ ; this auxiliary must, of course, be specially drawn up for the particular application. The auxiliary should be written so as to replace  $x_\epsilon$  by  $F(x_\epsilon)$  and then obey a link set in X1 by R 340.

1.2 Before calling in R 340, the master-programme must place in X6 and 7 two quantities  $x_a$  and  $x_b$  which are such that

- (a)  $F(x_a)$  and  $F(x_b)$  are of opposite sign,
- and (b) A zero of  $F(x)$  lies between  $x_a$  and  $x_b$ .

It should, of course, also set a link in accumulator 1. When called in R 340 will then find a zero of  $F(x)$  and will leave the corresponding value of  $x$  in X6 on exit. This value of  $x$  differs from the zero of the function by less than a positive constant  $\epsilon$ , whose value is set by a preset-parameter.

2. The Parameter-List

2.1 A parameter-list of the following form may be supplied:-

	R 0 0 -0 2	
	340 - 04 -	
01	+ 0.0002	= $\epsilon$ , tolerance on the value of the zero.
02	0 + 0 72	} Cue to auxiliary
	0.0 0 60	

The parameters written above are merely illustrative. This parameter-list should normally form part of the auxiliary, in order to obtain the correct relative address in the second parameter.

2.2 If desired, the cue to the auxiliary may be of the following form:-

02	0.7 2 10
	0.7 0 60

In which case the master-programme should set the cue to the auxiliary in X2 before calling in R 340. This allows the use of several different auxiliaries should R 340 be required to find zeros of different functions at various stages of a programme. One can use X2,3 or 4 in this way and the register (0.7 in the above illustration) may be replaced by any other ordinary register.

2.3 An optional parameter-list is provided with R 340. In this preset-parameter 01 has the value  $4 \times 2^{-38}$  and preset parameter 02 is as shown in paragraph 2.2 above.

3. The Auxiliary

3.1 The auxiliary must replace  $x_6$  by  $F(x_6)$ , which should be scaled so that it satisfies the inequality

$$-\frac{1}{2} < F(x_6) < \frac{1}{2}.$$

3.2 The auxiliary may use any of the accumulators or registers in the Computing Store and any parts of the Main Store except that it should not, of course, overwrite R 340, or B1, which is used by R 340 to preserve the accumulators. If preset-parameter 02 is of the form described in paragraph 2.2 the auxiliary should not disturb X2, or whichever accumulator is used to hold the cue.

3.3 Accumulators 2,3 and 4 are not touched by R 340 and anything placed there by the master-programme will be available to the auxiliary. Should the value of  $N$ , the number of iterations performed by R 340, be required, one of these accumulators may be used for counting; the auxiliary is called in  $N+2$  times by R 340.

4. Process

4.1 The following process is used: it is not quite second order but is better than first order.

4.2 The quantities  $x_a$  and  $x_b$  originally set by the master-programme are repeatedly replaced by other values. Suppose that  $x_a, x_b$  denote their values at the beginning of an iteration and they are replaced by  $x_a'$  and  $x_b'$ . Let the function values be  $F_a, F_b$  before and  $F_a', F_b'$  after the iteration.  $F_a$  and  $F_b$  have opposite signs.

4.3 A quantity  $x_c$  is determined by the rule

$$x_c = \frac{x_a F_b - x_b F_a}{F_b - F_a};$$

and the auxiliary is used to evaluate  $F_c$ . The following changes are then made:-

(a) If  $F_a$  and  $F_c$  have opposite signs:-

$$F_b' = F_a, \quad x_b' = x_a;$$

(b) If  $F_a$  and  $F_c$  have the same signs:-

$$F_b' = \frac{F_a F_b}{F_a + F_c}, \quad x_b' = x_b;$$

(c) In either case:-

$$F_a' = F_c; \quad x_a' = x_c.$$

It will be seen that  $F_a'$  and  $F_b'$  will have opposite signs.

4.4 The process terminates when

- either (a)  $F_c = 0$ , the result being  $x_c$ ,  
or (b)  $|x_a' - x_b'| \leq \epsilon$ , the result being  $x_a' = x_c$ .

If neither of these results holds or if overflow occurs then a new iteration is performed.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
27.12.56.

## LINEAR INVERSE INTERPOLATION

$$f(p') = p$$

This subroutine estimates the value of the argument  $x$  corresponding to a given value  $y = f(x)$  of a tabulated function.  $f(x)$  must be tabulated at equal intervals of  $x$ .

**Name:** LIN. INVERSE INT.

**Store:** 4 Blocks.

**Uses:** U0,1; X6,7; B0.

**Cues:**

- 01 (0+.1+) To interpolate in table number  $q$ , the address of which is given by the  $q^{\text{th}}$  entry in an index.  $q$  is set in  $7_c$ . The address of the index is given by preset parameter 01.
- 02 (0+.1) To interpolate in the table whose address is given in the single word index.
- 03 (0+.3+) To interpolate in the table whose address is set in  $7_m$ .

**Time:** Using cue 03 the time would be approximately

$$2.5r + 86 \text{ milliseconds}$$

when the given function lies between the  $r^{\text{th}}$  and the  $(r + 1)^{\text{th}}$  entries in the table.

Entering by cue 01 or 02 would add approximately 14 milliseconds.

**Link:** Obeyed in 1.1 and left unaltered in X1.

**Error:** See Paragraph 6 below.

**1. Layout of the Table**

The first word in each table must be stored at the start of a block; the contents of this block must be as follows:

A.0	$N$	= Maximum number of points needed for interpolation
.1	$m$	= Number of values of $f(x)$ tabulated.
.2	$x_0$	= Initial value of $x$ .
.3	$h$	= $(x_{i+1} - x_i)$ , the tabular interval.
.4	$f(x_0)$	} Successive values of the function in $m$ consecutive locations.
.5	$f(x_1)$	
.6	$f(x_2)$	
	etc.	

$N-1$  denotes the order of interpolation which must be used to obtain maximum accuracy from the table.  $N$  is not used by R 341 but should be inserted so that the table may be used with other interpolation routines.

If there are two or more values of  $x$  for which  $f(x)$  has the given value the subroutine will find the first. If  $h$  is positive this will be the smallest  $x$ , but if  $h$  is negative it will be the largest  $x$  such that  $f(x)$  has the given value. It is preferable for  $h$  to be positive so that the table may be used by other interpolation routines.

**2. The Index**

If cue 01 is to be used, an index of tables must be stored in the master programme. This index must be arranged as shown below; it can most easily be formed by using R 102.

Location	Modifier	Counter
$B.P$	0	$t$
$B.P + 1$	$A_1$	$n_1$
$B.P + 2$	$A_2$	$n_2$
⋮	⋮	⋮
$B.P + t$	$A_t$	$n_t$

$t$  specifies the number of tables listed in the index.  $A_1, A_2, \dots, A_t$  are the addresses of the first words in each table.  $n_1, n_2, \dots, n_t$  are not used by R 341; they are used by some routines to specify the number of points to be used for interpolation.

### 3. Preset Parameters

If cue 01 is to be used, the address of the index must be specified by preset parameter 01. If cue 02 is to be used the index consists of one word containing  $(A, n)$ ; the address of this word is specified by preset parameter 01. If no parameter list is read the address will be set at 2.0.

If the given value of  $f(x)$  is outside the range of the table the subroutine will jump to preset parameter 02. This may be the cue to an extrapolation routine or a loop stop in 0.6: if no parameter list is provided it will be a loop stop in 0.6.

The parameter list will normally be punched as follows:-

R 0 0 -0 2	Title
341 - 04 -	
B - P0 0.	Address B.P of the index
0	
0.6 0 60	Cue to extrapolation routine or loop stop in 0.6
0	

### 4. Extrapolation

On entering an extrapolation routine the contents of the following locations may be of use.

- X1  $(A + 0.4, 0)$  the address of  $f(x_0)$ , the first entry in the table.
- X2  $x_0$ , the initial value of  $x$  in the table.
- X3  $h$ , the interval in  $x$ .
- X6 The given value of  $y = f(x)$ .
- X7  $n_c = m$ , the number of entries in the table.  
 $n_m$  = Address of the last entry in the table.
- U1 The last block of the table.
- B0 The accumulators as stored on entry except for X7. B 0.7 contains  $(A, n)$ . The link is in B 0.1.

### 5. Method

The subroutine searches through the table, starting at  $y_0 = f(x_0)$ , until it finds two adjacent entries such that

$$y_r < y \leq y_{r+1}$$

$$\text{or } y_r > y \geq y_{r+1}$$

where  $y$  is the given value of the function.

If there are two or more values of  $r$  for which one of these conditions is true, the smaller value of  $r$  will be selected. If  $h$  is positive this means that the smaller value of  $x$  is selected. If  $h$  is negative the larger value of  $x$  is selected.

The required value of  $x$  is given by

$$x = x_{r+1} - \frac{y_{r+1} - y}{y_{r+1} - y_r} h$$

**6. Error**

Slight inaccuracy may arise due to the conversion of the tabular interval,  $h$ , from decimal to binary. An exact decimal interval may be held in the computer with an error of up to  $2^{-39}$ . The formula for  $x$  may be written

$$x = x_0 + (r + 1 - \delta)h$$

where  $0 \leq r \leq m - 2$        $0 \leq \delta \leq 1$

Thus an error of  $2^{-39}$  in  $h$  will lead to an error of about  $r.2^{-39}$  in  $x$ . The largest error will occur at the upper boundary of the table and may attain a maximum of  $(m - 1).2^{-39}$ .

Apart from the error described above the error in the interpolation process should not exceed  $2.2^{-38}$ . Thus the maximum error would be  $(m + 3).2^{-39}$ . This error will usually be small compared with that introduced by the use of a linear approximation.

If  $|x_{m-1}| \geq 1.0$ , the value of  $h$  may have to be rounded down to ensure that the calculated value of  $x_{m-1}$  is within capacity.

**7. Stops**

0.4	0 1 72 7
	0 4 00

Writing with overflow due to 73 order in 0.3+.  
Caused by entering with OVR set or possibly by incorrect setting of index address.

0.6	0.6 0 60
	0

$f(x)$  outside the range of the table and no extrapolation routine supplied.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
26.2.58

## TWO WAY LINEAR INVERSE INTERPOLATION

$$p = f(q, p') \quad \text{or} \quad p = f(p', q)$$

Given a table of  $f(x, y)$ , a value of  $f(x, y)$  and a value of  $x$  (or  $y$ ), this routine calculates the corresponding value of  $y$  (or  $x$ ). The layout of the table is the same as for R 322.  $f(x, y)$  is set in X6 and  $x$  (or  $y$ ) in X7.

**Name:** 2 WAY INV. INT.

**Store:** 11 blocks.

**Uses:** U0, 1, 2, 3; X5, 6; B0.

**Cues:**

- 01 To interpolate in table number  $k$ , the address of which is given by  
(0+.1+) the  $k^{\text{th}}$  entry in the index.  $k$  is set in  $5_c$ .
- 02 To interpolate in the table whose address is given in the single  
(0+.1) word index.
- 03 To interpolate in the table whose address is set in  $5_m$ .  
(0+.3+)

**Time:** Given  $x$ , to find  $y$ :-  
Approximately  $210 + 8t$  milliseconds ( $0 \leq t \leq m_y - 2$ ) where  $t$  is such that the given  $f(x, y)$  lies between  $f(x, y_t)$  and  $f(x, y_{t+1})$ .

Given  $y$ , to find  $x$ :-  
Approximately  $235 + 15t$  milliseconds ( $0 \leq t \leq m_x - 2$ ) where  $t$  is such that the given  $f(x, y)$  lies between  $f(x_t, y)$  and  $f(x_{t+1}, y)$ .

**Link:** Obeyed in 3.7 and left as set in X1. The link must be set as described in section 1 below.

## 1. Direction of Interpolation

The direction of interpolation is specified by the setting of the link. The link must always be a 'go' order pair.

If  $x$  is given, the link must be set in X1 in the normal way (i.e. by an 00 order) for the subroutine to evaluate  $y$ .

If  $y$  is given, the link must be set negatively in X1 (i.e. by means of an 02 order) for the subroutine to evaluate  $x$ .



**2. Layout of the Table and Index**

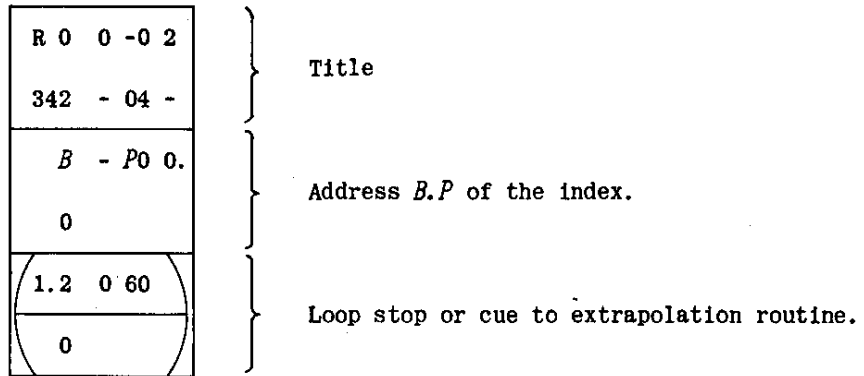
The layout of the table and the index are as described in sections 1 and 2 of the specification of R 322. R 342 will accept negative values of  $h_x$  and  $h_y$ .

**3. Preset Parameters**

If cue 01 is to be used the address of the index must be specified by preset parameter 01. If cue 02 is used the index consists of one word containing  $(A, n)$ . The address of this word is specified by preset parameter 01. If no parameter list is read the address will be set at 2.0.

If the given value of  $x$  (or  $y$ ) or of  $f(x, y)$  is outside the boundary of the table the subroutine will jump to preset parameter 02. This may be the cue for an extrapolation routine or a loop stop in 1.2: if no parameter list is supplied it will be a loop stop in 1.2.

The parameter list will normally be punched as follows:-



**4. Extrapolation**

The cue to the extrapolation routine is obeyed in 1.2. The extrapolation routine may be entered in two ways:

- (a) if the given value of  $x$  (or  $y$ ) is outside the range of the table. In this case X2 will be negative.
- (b) if the given value of  $f(x, y)$  is outside the range of the table for the given value of  $x$  (or  $y$ ). In this case X2 will be positive.

On obeying the cue to the extrapolation routine the contents of the following registers may be useful:-

In case (a): X1      Negative if  $x$  given (normal link). Positive if  $y$  given (- link).

$$X2 \begin{cases} \text{Negative} \\ -1.0 \text{ if } x \text{ given } (2_m = 0); \quad -1.0 + 2^{-13} \text{ if } y \text{ given } (2_m = 1) \end{cases}$$

X5      Address of table.

X6      Remainder after division:  $\frac{x - x_0}{h_x}$  or  $\frac{y - y_0}{h_y}$

X7       $q < 0$  if  $\frac{x}{h_x} < \frac{x_0}{h_x}$  (or  $\frac{y}{h_y} < \frac{y_0}{h_y}$ )  
 $q \geq 0$  if  $\frac{x}{h_x} > \frac{x_{m-1}}{h_x}$  (or  $\frac{y}{h_y} > \frac{y_{m-1}}{h_y}$ )

U0.0 Given value of  $f(x, y)$ .

U3 First block of table.

B0 Accumulators, except for X5, as stored on entry. B0.5 will contain the address of the table.

In case (b):

X2  $\left\{ \begin{array}{l} \text{Positive} \\ 0 \text{ if } x \text{ given, } 2^{-13} \text{ if } y \text{ given.} \end{array} \right.$

X5 Address of table.

U0.0 Given value of  $f(x, y)$ .

B0 Accumulators, except for X5, as stored on entry. B0.5 will contain the address of the table.

Note: Before returning to the main programme from an extrapolation routine it will usually be necessary to restore the accumulators from B0, including the link in B0.1 (or the negated link if  $y$  given).

### 5. Method

Suppose that  $x$  is given. The subroutine first finds an  $r$  and a  $k$  such that

$$x = x_0 + rh_x + kh_x$$

where  $r$  is an integer and  $k$  is a fraction.

Let  $z_t = f(x, y_t)$

$$= (1-k).f(x_r, y_t) + k.f(x_{r+1}, y_t) \quad t = 0, 1, 2, \dots, m_y - 1$$

The subroutine forms the  $z_t$  in turn and searches until it finds a value of  $t$  such that

$$z_t < z \leq z_{t+1} \quad \text{or} \quad z_t > z \geq z_{t+1}$$

where  $z$  is the given value of the function. The required value of  $y$  is given by

$$y = y_{t+1} - \frac{z_{t+1} - z}{z_{t+1} - z_t} \cdot h_y \quad \text{or}$$

$$y = y_0 + (t+1)h_y - \frac{(1-k).f(x_r, y_{t+1}) + k.f(x_{r+1}, y_{t+1}) - f(x, y)}{(1-k).f(x_r, y_{t+1}) + k.f(x_{r+1}, y_{t+1}) - (1-k).f(x_r, y_t) - k.f(x_{r+1}, y_t)} \cdot h_y$$

A similar formula is used for the case when  $y$  is given.

### 6. Error

The maximum rounding error in calculating  $y$ , given  $x$ , is

$$\left[ m_y + m_x \left( \frac{\partial y}{\partial x} \right)_{f=z} \right] 2^{-39}$$

For given  $y$  replace  $y$  by  $x$  and  $x$  by  $y$  in this formula.

The main contributions to this maximum error are from the rounding errors of  $2^{-39}$  which may arise in storing  $h_x$  and  $h_y$  in binary form. This error will normally be small compared with that involved in using a linear approximation.

It is possible for an argument, which is just below the upper boundary of the original table, to be rejected as just above the boundary of the binary table. This fault can be eliminated by rounding  $h$  upwards, but this has the effect of increasing the rounding errors.

### 7. Stops

0.4	0	3	72	5
	1+	1	72	

Writing with overflow due to 73 order in 0.3+ if OVR is set on entry, or possibly if the index address is set incorrectly.

1.2	1.2	0	60
	0		

Loop stop if  $x$  (or  $y$ ), or if  $f(x, y)$  is outside the range of the table and no extrapolation routine is supplied.

**Author:** Mr. H.P. Goodman of the de Havilland Aircraft Company.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
22. 4. 58.

## POWER SERIES ECONOMISATION

This routine economises a power series by means of a Chebycheff expansion, and prints the resulting coefficients. It works in double-length floating-point and uses R 650, but normally prints single-length fixed point coefficients.

Name: ECONOMISATION

Store: 12 blocks.

Uses: Some R 650 working space (see section 7).

Cue: 01 (0+.6)

Link: Not required (see section 1).

Time: Varies roughly as the square of the length of the series. Takes approximately 10 minutes for the first part of the process with a series of degree 40.

## Method of Use

1. R 360 is a double-length floating-point subroutine called in by obeying the 76-order of R 650 (see section 7 of the specification of R 650).
2. The subroutine economises over an interval  $-h$  to  $h$  symmetric about the origin.  $h$  must be set in  $S_0$  (i.e. location 0 of the R 650 working space) before entry.

The coefficients of the power series are to be placed in successive locations of the R 650 working space, i.e. in  $S_m, S_{m+3}, S_{m+6}, \dots, S_{m+3N}$ , where  $S_m$  is the address of the constant term and  $N$  is the degree of the series.  $m$  must be  $\geq 9 + \frac{1}{2}n$ , where  $n$  is the degree of the economised series (see also section 7). A word must be set in X3 containing  $S_m$  (the R 650 address of the constant term) in the modifier position, and the degree of the last term in the counter position.

3. Let the given series be  $C(x) = \sum_0^N c_i x^i$ . The subroutine first replaces the coefficients by the coefficients of  $C'(x) = \sum_0^N c'_i x^i$  such that  $C'(x) = C(hx)$ .

Let  $C'(x) = \sum_0^N a_i T_i(x)$  where the  $T_i(x)$  are Chebycheff polynomials.

Let  $b_0 = a_0$

$$b_i = 2^{i-1} a_i \quad i > 0$$

The  $c'_i$  are now replaced by the  $b_i$  and the following numbers are printed:

$$\begin{array}{r}
N-1 \quad | a_N | \\
N-2 \quad | a_N | + | a_{N-1} | \\
\text{.....} \\
i \quad | a_N | + | a_{N-1} | + \text{.....} + | a_{i+1} | \\
\text{.....} \\
0 \quad | a_N | + | a_{N-1} | + \text{.....} + | a_1 |
\end{array}$$

The numbers on the right give upper bounds for the errors committed in truncating the series  $\sum_0^N a_i T_i(x)$  after the term of degree  $N-1, N-2, \dots, i, \dots, 0$ . They are printed in floating decimal form with three significant figures.

The subroutine next punches 2 ft. of blank tape and comes to a 77 stop in 0.4+.

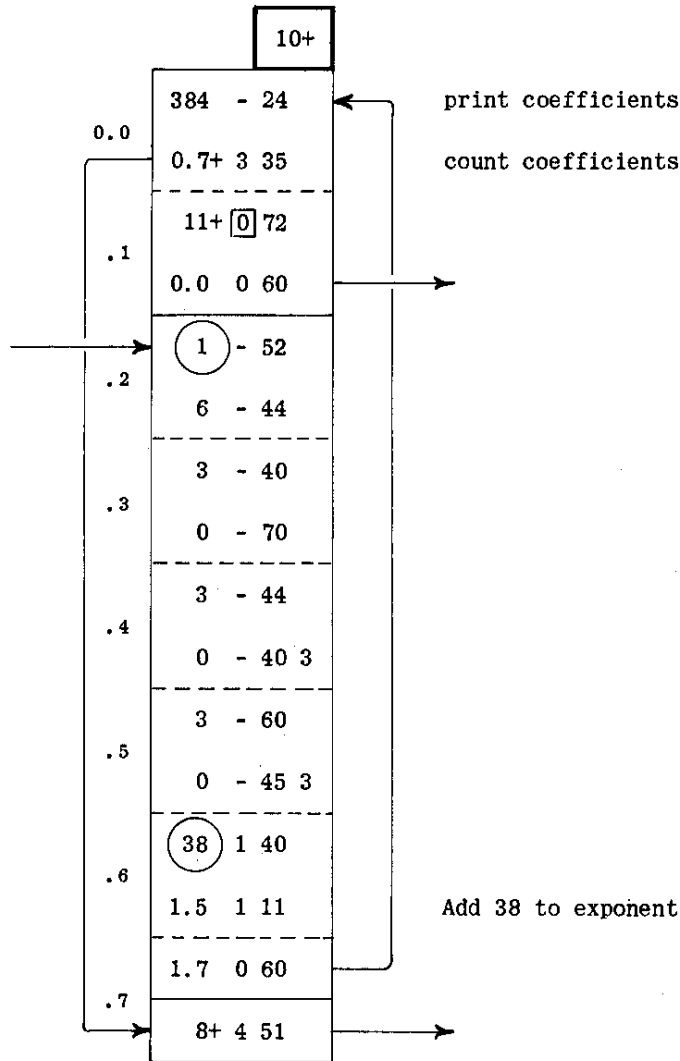
4. The operator can now decide, on the basis of the information printed, at which term to truncate the series. If it is to be truncated after the term of degree  $n$ , a tape containing CR LF +n CR LF λ L φ φ ..... must be punched and inserted in the main tape reader. On operating the STOP/RUN key the second stage of the process is carried out.

Let  $\sum_0^n a_i T_i(x) = \sum_0^n d_i (hx)^i$ . Then  $b_0, \dots, b_n$  are replaced by  $d_0, \dots, d_n$

which are the required coefficients.  $d_0 \times 2^{38}, d_1 \times 2^{38}, \dots, d_n \times 2^{38}$  are printed in this order rounded as integers, this being the most accurate form of input for Pegasus. Finally 6 inches of blank tape, 13 erases and 1½ inches of blank tape are punched; control is returned to the master programme and the next order, after the 76-order, is obeyed. The tape is suitable for re-input by Initial Orders, provided that none of the integers exceeds  $2^{38} - 1$ . If any coefficient exceeds this value, the series must be re-arranged (e.g. by halving this coefficient and using it twice). Note that accuracy will be lost if an odd number is halved.

5. The coefficients of the economised series replace the first few coefficients of the given series, starting in  $S_m$ . On exit X1 and U1.1 (the most significant part of the R 650 accumulator) contain  $(S_m, n)$ .  $S_m$  is the R 650 address of the constant term and  $n$  is the degree of the economised series.

6. Some users may wish to use a different form of output to that provided. The orders concerned are in B 10+.6-10+.7 and B 10+.0. Any alternative orders must not disturb X2 or X3. The block is as follows:



The order-pair by which the error estimates are printed is

0.5	2 - 27
	0.7 3 67

in B5+.5. Any programme to replace this printing must not disturb X2, X3, X5.

7. The subroutine up to the 77-stop uses R 650 working space S3-18. The second half of the programme uses locations S3, 6 and  $[\frac{1}{2}n]$  locations starting from S9, where  $[\frac{1}{2}n]$  is the integral part of  $\frac{1}{2}n$ .

8. Economisation in a range which is not symmetrical about the origin may be achieved by a transformation of the variable. Thus, for example, the range  $0 < x < h$  may be treated by taking  $y = \sqrt{x}$  over the range  $-\sqrt{h} < y < \sqrt{h}$ .

FERRANTI LTD

PEGASUS LIBRARY SPECIFICATION

Issue 1  
27.2.58

**POLYNOMIAL EVALUATION (M1)**

The evaluation of  $F(z) = \sum_{i=0}^n a_i z^{n-i}$ , where  $z = x + iy$  and the coefficients  $a_i$  are real. All numbers are in standard floating-point form as used in R 610.

**Name:** POLYNOMIAL EVALUATION (M1)

**Store:** 3 blocks + 4 blocks for R 610.

**Uses:** The whole Computing Store.

**Cues:**

01	$0+ \boxed{2} 72$ 2.0 0 60	When R 610 is not already in the Computing Store.
----	-------------------------------	---

02	$0+ \boxed{2} 72$ 2.1+ 0 60	When R 610 is in U3, 4, 5.
----	--------------------------------	----------------------------

<b>Time:</b>	Cue 01	Approximately $121n + 31$ milliseconds	}	where $n$ is the order of the polynomial.
	Cue 02	Approximately $121n + 8$ milliseconds		

**Link:** Normally obeyed in 1.5, but see section 2. Not left in X1.

**1. Method of Use**

Before entry, the components of  $z$ , that is  $x$  and  $y$ , which must be in standard floating-point form, must be placed in X6 and X7 respectively. The pseudo-order pair  $(A_0, n)$ , which may be either a 'stop' or a 'go' order pair, must be set in X2:  $A_0$  is the location in the Main Store of  $a_0$ , the leading coefficient, and  $n$  is the order of the polynomial.

The coefficients  $a_i$ , which must be in standard floating-point form, must be stored in successive locations commencing at  $A_0$ .

**2. Results**

The real and imaginary parts of  $F(z)$ , denoted by  $x_n$  and  $y_n$ , are normally left in X6 and 7 respectively.  $x_n$  will also be left in X3. The given values  $x$  and  $y$  will be left in U4.0 and 4.i respectively.

If the link were obeyed in 1.4 instead of 1.5,  $x_n$  and  $y_n$  would be left in X3 and 6 respectively. This may be arranged by punching a parameter list as follows:

R 0 0 -0 1	}	Title
361 - 04 -		
+0		Parameter

### 3. Floating-Point Arithmetic

R 361 works entirely in single length floating-point arithmetic as described in the Pegasus Programming Manual and in the specification of R 610. The number of binary digits used to represent the exponents of the floating-point numbers is specified by the parameters supplied for R 610. If no parameter list is supplied, it will be set as 9.

### 4. Method

The routine first sets  $x_0 = a_0$   $y_0 = 0$ .

It then forms  $x_{i+1} = x_i x - y_i y + a_{i+1}$

and  $y_{i+1} = y_i x + x_i y$

for  $i = 0, 1, 2, \dots, n$ .

then  $F(z) = x_n + iy_n$

### 5. Note on R 610

R 361 leaves R 610 in U3, 4 and 5 except that 4.0 and 4.1 are overwritten by  $x$  and  $y$ . These two locations are used only for floating-point division: R 610 may still be used for addition, subtraction and multiplication. If division is required, block 1+ of R 610 must be read into U4 again.

**Author:** Mr. R.H. Merson of R.A.E. Farnborough.



FERRANTI LTD

PEGASUS LIBRARY SPECIFICATION

Issue 2.  
17.9.56.

SIMULTANEOUS FIRST ORDER DIFFERENTIAL EQUATIONS

( $n > 8$ ;  $v \neq 0$ , i.e. scaling factor  $2^v$ )

The solution of  $n (> 8)$  equations of the form

$$\frac{dy_i}{dx} = f_i(x, y_1, y_2, \dots, y_n) \quad (i = 1, 2, \dots, n)$$

by a modified Runge-Kutta process due to C. Strachey.

Name: RUNGE-KUTTA (S).  $n > 8, v \neq 0$

Store: 9 blocks.

Uses: All the ordinary registers; X3, X4, X5 (in stage 3), X6, X7 (in stage 4).

Cues: 01 for entry with link in X1.  
02 for entry when previously set link is to be used.  
03 } for re-entry from auxiliary.  
04 }

Time:  $4A + nB + C$  milliseconds per step where:-

A is the time of the auxiliary;  
 $B = 34$  ( $0 < v \leq 3$ )  
 $B = 36$  ( $3 < v \leq 9$ )  
C varies in a complicated way with  $n$  and  $v$ , but is normally less than 100.

Link: Obeyed in U 1.7. See section 5.

1. Method of Use

1.1 R 400 takes its variables  $y_i$  straight from the Main Store; each variable  $y_i$  has associated with it the three quantities  $z, \phi$  and  $q$ , which also require space in the Main Store. The arrangement is as follows:-

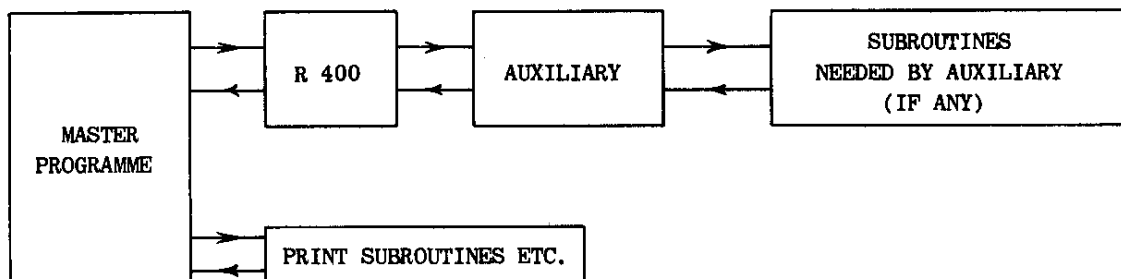
Block $b$	contains $z$	}	(values for the first eight variables)
$b + 1$	" $y$		
$b + 2$	" $\phi$		
$b + 3$	" $q$		
$b + 4$	" $z$	}	(values for the second eight variables)
$b + 5$	" $y$		
$b + 6$	" $\phi$		
$b + 7$	" $q$		
.....	etc. ....		

For  $b$  see section 3.

1.2 At the start of the range of integration the values of  $y$  must be set in the appropriate blocks and the blocks for  $p$  must be cleared; this may be done by the master-programme or by the Initial Orders. Thereafter the master-programme must leave the blocks for  $y$  and  $p$  as it finds them whenever it is called in during a range of integration; it may use the  $z$  and  $q$  blocks temporarily, but these blocks will be overwritten during the next step of integration.

1.3 Each time R 400 is called in by the master-programme the values of the variables will be advanced by one step (but see section 5.4); i.e. if  $x$  is the value of the independent variable and R 400 is called in, the values of the dependent variables  $y_i(x)$  will be replaced by those of  $y_i(x + h)$ , where  $h$  is the step length.

1.4 In order to use R 400 for a specific set of equations an auxiliary subroutine must be drawn up which defines the functions  $f_i$ . This auxiliary is called in four times during each step by R 400; it should be written in the way described in section 2 below. In addition there must of course be a master-programme, so that a block diagram of a typical programme would be as follows:-



## 2. The Auxiliary

The  $n$  functions  $f_i$ , are defined by an auxiliary subroutine (referred to as the auxiliary) which must be drawn up specially for each set of equations to be solved. It should have the following specification:-

1. Taking the values of  $y_i$  from blocks  $b + 1$ ,  $b + 5$ ,  $b + 9$ , etc., it should find  $z_i = (\frac{1}{2}hf_i)2^v$ , where  $h$  is the step length, and  $v$  a constant integer chosen as suggested in section 7.1 below, and place them in the corresponding locations in blocks,  $b$ ,  $b + 4$ ,  $b + 8$ , etc.
2. When finished it must return to R 400, by an entry consisting of two consecutive cues 03 and 04, the second of which must not be obeyed in UO.
3. It must leave the overflow indicator clear.
4. It must leave X3 and X4 as it finds them.
5. X3 contains a slightly altered form of parameter 03, i.e.  $(b - 4).0, n + 1$ , when the auxiliary is entered. This parameter can be used by the auxiliary, providing that it is finally restored to its initial value.

### 3. Preset-Parameters

The parameter-list should normally be part of the auxiliary and should be punched as follows:-

	R 0 0 -0 6 400 - 04 -	
01		Cue for auxiliary
02	(1.0 0 60) 0	Cue for overflow routine (Obeyed in 1.0)
03	b - 00 0. n	Parameter specifying first block of variables (=b) and number of variables (=n).
04	+ v	v = number of shifts
05	(1.2 2 67) (1.1 0 60)	Obeyed in 1.7 } Main Link Omit if not preset
06	( <input type="checkbox"/> 72) 0 60	Obeyed in 1.1 }

It must be read in before the programme of R 400, as this contains an interlude which consults the parameter list. It should therefore appear before the A 2 on a normal master type.

#### 4. Overflow

Preset parameter 02 is a cue obeyed when overflow occurs during R 400, as distinct from the auxiliary. The cue may lead to an overflow routine to be treated as part of the auxiliary. If such elaborate action on overflow is not required, the cue to the overflow routine in the parameter list may be a loop stop, as shown.

#### 5. The Main Link

**5.1** This is an order-pair which is obeyed in U1.7 when R 400 has finished its work; it may have the effect of returning control to the master-programme.

**5.2** If R 400 is used as a normal closed subroutine, the main link should be placed in X1 by the master-programme before calling in R 400 with cue 01.

**5.3** The main link may be preset if desired, as one order-pair in parameter 05, or as two order-pairs in parameters 05 and 06. Since the order-pairs are obeyed in U1.7 and U1.1 respectively, when both are preset one of the first order-pair must be a jump to U1.1. In this case the master-programme must be punched with the auxiliary so that both obtain the same relativiser; the notes, section 7.3, indicate a way of avoiding this restriction. If the main link is set by Assembly, R 400 must be entered from the master-programme by cue 02. The single-word main link may also be preset by previous use of entry 01.

5.4 There are two ways of setting the main link; it may be preset, i.e. set by Assembly on input, or it may be put in X1 by the master-programme before obeying the cue to R 400. In the first case R 400 should always be entered with cue 02, in the second R 400 should usually be entered with cue 01. When it is desired to use again the link which was obeyed after the last entry into R 400, the link need not be put in X1 again, provided R 400 is entered with cue 02. X1 is then available for use by the programmer.

5.5 Two order-pairs of the form shown in the parameter-list may be used to count integration steps without bringing down the master-programme each time. The effect will be that R 400 will integrate step after step until the count order is passed, when the master-programme will be entered.

## 6. The Independent Variable

The value of  $x$  is not normally used or evaluated by R 400. In general, however, it is used by the auxiliary. When it is required by the auxiliary or by the master-programme, it may be obtained by introducing an extra dependent variable satisfying

$$\frac{dy}{dx} = 1.$$

In this case the corresponding value  $z = \frac{1}{2}h.2^v$  may be set once and for all by the master-programme at the beginning of the range of integration; it will not be altered by R 400 (the auxiliary and master should be arranged so as to leave it undisturbed). It is convenient to hold the independent variable in the last block of  $y$ 's; if it is  $y_{amt+p}$ , it is then found in U5.p by the auxiliary.

If  $x$  is to be generated otherwise than as an extra dependent variable, it must be given the values  $x_0$ ,  $x_0 + \frac{1}{2}h$ ,  $x_0 + \frac{1}{4}h$ , and  $x_0 + h$  respectively when the auxiliary is entered for the first, second, third and fourth times in each step of integration, where  $x_0$  is the value of  $x$  at the beginning of the step.

## 7. Notes

7.1 **Choice of Scaling Factor.** The best choice of the positive integer  $v$  is the largest which does not cause overflow of any intermediate quantities in R 400. A good rule is to keep all the  $z_i$  less than 0.3.

The rounding errors will build up at a rate corresponding to the retention of an extra  $v$  binary digits in the variables; hence the desirability of making  $v$  as large as possible.

7.2 **Action on overflow.** It may be desired to identify the variable which has overflowed, and then re-enter R 400. The identification can be done by using the contents of X3 considered as a modifier and counter. If X3 contains  $(B,P,c)$ , then the storage location of the overflowing variable is  $B + 5.P$ , and the variable is the  $r^{\text{th}}$  where  $r = n + 1 - c$ . The variables are supposed numbered starting with the first in  $B + 1.0$ .

In order to re-enter one may write:-

0	1	72 4
1.0	0	60

A tag referring to the location of the above order-pair and calling for partial cue 05 to R 400, must then be punched in an appropriate place. The overflow routine must leave the overflow-indicator clear, and the contents of U0, U2, U3, U4, U5, X3 and X4 as it found them.

Note that the cue to the overflow routine occurs in the parameter list of R 400, together with the cue to the auxiliary and the preset link to the master-programme. Consequently it is simplest to punch the overflow routine together with the auxiliary. This is probably the most natural thing to do in any case, since it is at the time when the auxiliary is being drawn up that consideration will be given to the action needed on overflow.

**7.3 Separation of Auxiliary from Master-Programme.** If it is desired to avoid punching the auxiliary and the master-programme together, one may regard the latter as a programmer's subroutine and assign it a routine number. When used in this way the master-routine would normally have a cue-list containing two cues:-

01 for the initial entry (start of the programme),  
02 for re-entry from R 400.

In the auxiliary there would be a tag calling for cue 02 to the master-routine. The preset link of R 400 must then lead to the location of this cue in the auxiliary. In order to gain entry to the master-routine when the Initial Orders are finished, a dummy master-programme must be punched. This will consist of an empty location, with a tag calling for cue 01 to the master-routine.

Similarly, the auxiliary may contain a cue to the overflow routine if it is desired to treat the latter as separate from the auxiliary.

**7.4 Timing.** The routine is probably at its fastest if the first block of working space  $b \equiv 1+ \pmod{16}$ .

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
18.9.56

## SIMULTANEOUS FIRST-ORDER DIFFERENTIAL EQUATIONS

( $n > 8$ ;  $v = 0$ , i.e. scaling factor unity)

The solution of  $n (> 8)$  equations of the form

$$\frac{dy_i}{dx} = f_i(x, y_1, y_2, \dots, y_n) \quad (i = 1, 2, \dots, n)$$

by a modified Rung-Kutta process due to C. Strachey.

**Name:** RUNGE-KUTTA (S).  $n > 8, v = 0$

**Store:** 6 blocks.

**Uses:** All Computing Store except X2, X3.

**Cues:** 01 for entry with link in X1.  
02 for entry when previously set link is to be used.  
03 for re-entry from auxiliary.

**Time:** About  $22n + T$  milliseconds per step plus four times the time of the auxiliary.  $T$  is between 40 and 140 milliseconds.

**Link:** Obeyed in U0.7. See section 5.

## 1. Methods of Use

1.1 R 401 takes its  $n$  variables  $y_i$  straight from the Main Store; each variable  $y_i$  has associated with it the three quantities  $z, p$  and  $q$ , which also require space in the Main Store. The arrangement is as follows:-

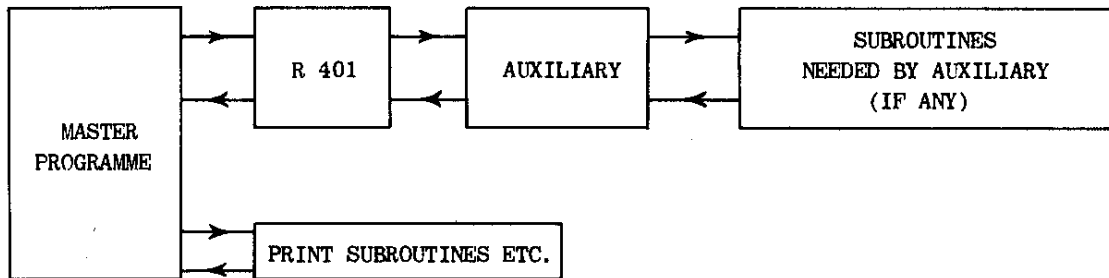
Block $b$	contains $z$	}	(values for the first eight variables)
$b + 1$	" $y$		
$b + 2$	" $p$		
$b + 3$	" $q$	}	(values for the second eight variables)
$b + 4$	" $z$		
$b + 5$	" $y$		
$b + 6$	" $p$		
$b + 7$	" $q$		
.....	etc. ....		

For  $b$  see section 3.

1.2 At the start of a range of integration the initial values of  $y$  must be set in the appropriate blocks in the Main Store; this may be done by the master-programme or, during input, by the Initial Orders. Thereafter the master-programme must leave the blocks for  $y$  as it finds them, whenever it is called in during a range of integration; it may use the  $z$ ,  $p$  and  $q$  blocks temporarily, but these blocks will be overwritten during each step of integration.

1.3 Each time R 401 is called in by the master-programme the values of the variables will be advanced by one step (but see section 5.4); i.e. if  $x$  is the value of the independent variable and R 401 is called in, the values of the dependent variables  $y_i(x)$  will be replaced by those of  $y_i(x+h)$ , where  $h$  is the step length.

1.4 In order to use R 401 for a specific set of equations, an auxiliary subroutine must be drawn up which defines the functions  $f_i$ . This auxiliary is called in four times during each step by R 401; it should be written in the way described in section 2 below. In addition, there must, of course, be a master-programme, so that a block diagram of a typical programme would be as follows:-



## 2. The Auxiliary

The  $n$  functions  $f_i$  are defined by an auxiliary subroutine (referred to as the auxiliary) which must be drawn up specially for each set of equations to be solved. It should have the following specification:-

1. Taking the values of  $y_i$  from blocks  $b + 1$ ,  $b + 5$ ,  $b + 9$ , etc. in the Main Store, it should find  $z_i = \frac{1}{2}hf_i$ , where  $h$  is the step length, and place them in the corresponding locations in blocks  $b$ ,  $b + 4$ ,  $b + 8$ , etc. To avoid overflow of intermediate quantities the  $z_i$  should be kept less than about 0.3.
2. It must not be entered in U1.
3. When finished it must return to R 401 by cue 03.
4. It must leave X4 unaltered, or restore it.
5. It must leave the overflow-indicator clear.
6. To facilitate counting it may use an altered form of parameter 04, i.e.  $((b-4.0), n+1)$ , which is placed in X5 by R 401 before entering the auxiliary.

### 3. Preset-Parameters

	R 0 0 -0 6 401 - 04 -	
01	<input type="checkbox"/> 72 0 60	Cue to Aux.
02	0 <input type="checkbox"/> 72 0.	} Cue to Aux. divided between two order pairs.
03	0 60 0.	
	0	
04	(1.4 0 60 ) ( 0 )	Cue to overflow routine. (Obeyed in 1.4)
05	<i>b</i> - 00 0. <i>n</i>	Parameter specifying first block working space (=b) and number of variables (=n).
06	(1.0+ 2 67 ) <input type="checkbox"/> 72	Main link. Omit if not preset.

### 4. Overflow

Preset-parameter 03 is a cue obeyed when overflow occurs during R 401, as distinct from the auxiliary. The cue may lead to an overflow routine, to be treated as part of the auxiliary. The cue to the overflow routine shown in the parameter-list above is a loop stop; this may be replaced by an order-pair consisting of a block-transfer and a jump.

### 5. The Main Link

5.1 This is an order pair which is obeyed in U0.7 when R 401 has finished its work; it normally has the effect of returning control to the master-programme.

5.2 If R 401 is used as a normal closed subroutine, the main link should be placed in X1 by the master-programme before calling in R 401 with cue 01.

5.3 If desired the main link may be preset as parameter 06. In this case R 401 should be entered by cue 02. The main link may also be preset by use of 01 with the link in X1, cue 02 being used thereafter to enter R 401.

5.4 If it is required that R 401 should advance the integration by several steps each time it is called in, the number of steps should be placed in  $2_c$  (or  $3_c$ ) by the master-programme before calling in R 401. The main link should then have the form of the order pair shown at 06 in the preset-parameter list. When the count order is passed, the master-programme will be re-entered in U1.0. The auxiliary must in this case leave  $2_c$  (or  $3_c$ ) as it finds it.

5.5 When the main link is preset, the master-programme must be punched with the auxiliary so that both obtain the same relativizer; the notes (section 7.2) indicate a way of avoiding this restriction.



## 6. The Independent Variable

The value of  $x$  is not normally used or evaluated by R 401. In general, however, it is used by the auxiliary. When it is required by the auxiliary or by the master-programme, it may be obtained by introducing an extra dependent variable satisfying

$$\frac{dy}{dx} = 1.$$

In this case the corresponding value  $z = \frac{1}{2}h$  may be set once and for all by the master-programme at the beginning of the range of integration; it will not be touched by R 401. (The auxiliary and the master-programme should be arranged so as to leave it undisturbed). It is convenient to hold the independent variable in the last block of  $y$ 's. If it is  $y_{8m+p}$ , it will be found by the auxiliary in U5.P.

If  $x$  is to be generated otherwise than as an extra dependent variable; it must be given the values  $x_0$ ,  $x_0 + \frac{1}{2}h$ ,  $x_0 + \frac{1}{2}h$ , and  $x_0 + h$  respectively when the auxiliary is entered for the first, second, third, and fourth times in each step of the integration; where  $x_0$  is the value of  $x$  at the beginning of the step.

## 7. Notes

### 7.1 Action on overflow

It may be desired to identify the variable which has overflowed. This can be done by using the contents of X5 considered as a modifier and counter. If X5 contains  $(B.P, c)$  on overflow, then the storage location of the overflowing variable is  $(B + 5).P$ , and the variable is the  $c$ -th, counting backwards from the last in the store.

Note that the cue to the overflow routine occurs in the parameter list of R 401, together with the cue to the auxiliary and the main link (if preset) to the master-programme. Consequently it is simplest to punch the overflow routine together with the auxiliary. This is probably the most natural thing to do in any case, since it is at the time when the auxiliary is being drawn up that consideration will be given to the action needed on overflow.

### 7.2 Separation of the Auxiliary from the Master-Programme

If it is desired to avoid punching the auxiliary and the master-programme together one may regard the latter as a programmer's subroutine and assign it a routine number. When used in this way the master-routine would normally have a cue-list containing two cues:-

- 01 for the initial entry (start of the programme),
- 02 for re-entry from R 401.

In the auxiliary there would be a tag calling for cue 02 to the master-routine. The preset link of R 401 must then lead to the location of this cue in the auxiliary. In order to gain entry to the master-routine when the Initial Orders are finished, a dummy master-programme must be punched. This will consist of an empty location, with a tag calling for cue 01 to the master-routine.

Similarly, the auxiliary may contain a cue to the overflow routine if it is desired to treat the latter as separate from the auxiliary.

### 7.3 Timing

The routine is probably at its fastest if the first block of working space  $b \equiv 9 \pmod{16}$ .

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2  
19.9.56.

## SIMULTANEOUS FIRST ORDER DIFFERENTIAL EQUATIONS

( $8 \geq n$ ;  $v \neq 0$ , i.e. scaling factor  $2^v$ )

The solution of  $n$  ( $\leq 8$ ) equations of the form

$$\frac{dy_i}{dx} = f_i(x, y_1, y_2, \dots, y_n) \quad (i = 1, 2, \dots, n)$$

by a modified Runge-Kutta process due to C. Strachey.

**Name:** RUNGE-KUTTA (S).  $8 \geq n, v \neq 0$

**Store:** 5 blocks.

**Uses:** U0, U1, the last  $n$  registers of U2, U3, U4 and U5; X5, X6, X7 (in stage 4).

**Cues:** 01 for entry with link in X1.  
02 for entry when previously set link is to be used.  
03 for re-entry from auxiliary.

**Time:** Approximately  $n(19 + \frac{1}{2}v) + 28$  milliseconds per step, plus four times the time of the auxiliary.

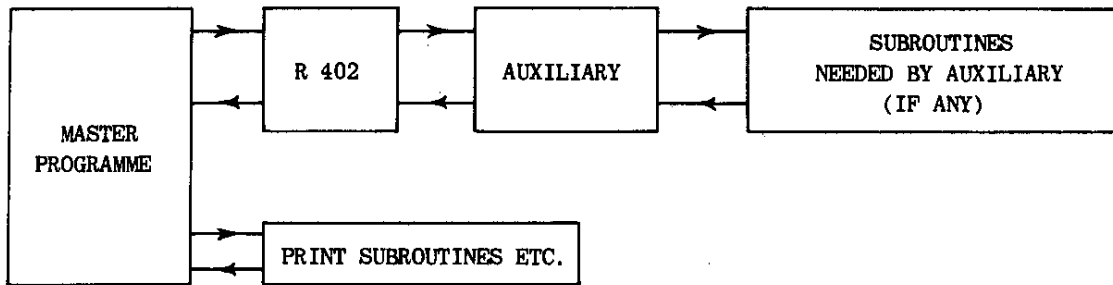
**Link:** Obeyed in U1.7. See section 5.

## 1. Method of Use

**1.1** At the start of the range of integration the master-programme must set the initial values of the variables  $y_i$  in the last  $n$  registers of U5 and clear the last  $n$  registers of U4, thereafter leaving them as it finds them.

**1.2** Each time R 402 is called in by the master-programme the values of the variables will be advanced by one step (but see section 5.2); i.e. if  $x$  is the value of the independent variable and R 402 is then called in, the values of the dependent variables  $y_i(x)$  will be replaced by  $y_i(x + h)$ , where  $h$  is the step-length.

**1.3** In order to use R 402 for a specific set of equations an auxiliary subroutine must be drawn up which defines the functions  $f_i$ . This auxiliary is called in four times during each step by R 402; it should be written in the way described in section 2 below. In addition, there must, of course, be a master-programme, so that a block-diagram of a typical programme would be as follows:-



**2. The Auxiliary**

The  $n$  functions  $f_i$  are defined by an auxiliary subroutine (referred to as the auxiliary) which must be drawn up specially for each set of equations to be solved. It should have the following specification:-

1. Taking the values of  $y_i$  from the last  $n$  registers of U5 it should find  $z_i = (h f_i) 2^v$  and place them in the corresponding registers of U3; where  $h$  is the step length and  $v$  a constant chosen as suggested in the notes.
2. When finished it must return to R 402 by cue 03 or, if U0 has not been affected, by the order:-

0.7 0 60
----------

3. It must leave  $5_m$  and the last  $n$  registers of U2, U4 and U5 as it finds them.
4. It must leave the overflow indicator clear.

**3. Preset-Parameters**

The parameter list should normally be part of the auxiliary, and should be punched as follows:-

	R 0 0 -0 5	
	402 - 04 -	
01		Cue for auxiliary.
02	( 0.5 0 60 )	Cue for overflow routine.
	( 0 )	(obeyed in 0.5)
03	$n$ - -0 0.	$n$ = number of variables
	0	
04	+ $v$	$v$ = number of shifts.
05	( 0.1 67 )	Main link (omit if not preset)
	( 2 72 )	Obeyed in 1.7.

It must be read in before the programme of R 402, as this contains an interlude which consults the parameter list. It should therefore appear before the A2 on a normal master tape.

#### 4. Overflow

Preset-parameter 02 is a cue obeyed when overflow occurs during R 402, as distinct from the auxiliary. The cue leads to an overflow routine, to be treated as part of the auxiliary. The order pair shown in the parameter list is a loop stop; for other suggestions see the notes (section 7.2).

#### 5. The Main Link

5.1 This is an order-pair which is obeyed in U1.7 when R 402 has finished its work; it has the effect of transferring control to the master-programme.

If R 402 is used as a normal closed subroutine, the main link should be placed in X1 by the master-programme before calling in R 402 with cue 01.

5.2 If desired the main link may be preset by writing it in as preset-parameter 05. In this case cue 02 should be used and the master-programme should be punched with the auxiliary so that it has the same relativizer — the notes (Section 7.3) suggest a way of avoiding this restriction.

The type of order-pair shown in the parameter list as the main link may be used in order to count integration steps without bringing down the master-programme each time. The effect will be that R 402 will integrate step after step until the count order is passed, when the master-programme will be entered in U2.0.

#### 6. The Independent Variable

The value of  $x$  is not normally used or evaluated by R 402. In general, however, it is used by the auxiliary. When it is required by the auxiliary or by the master-programme, it may be obtained by introducing an extra dependent variable satisfying

$$\frac{dy}{dx} = 1$$

In this case the corresponding value  $z = \frac{1}{2}h \cdot 2^v$  may be set once and for all by the master-programme at the beginning of the range of integration; it will not be touched by R 402 (the auxiliary and master should be arranged so as to leave it undisturbed).

If  $x$  is to be generated otherwise than as extra dependent variable, it must be given the values  $x_0$ ,  $x_0 + \frac{1}{2}h$ ,  $x_0 + \frac{1}{4}h$  and  $x_0 + h$  respectively when the auxiliary is entered for the first, second, third and fourth times in each step of the integration, where  $x_0$  is the value of  $x$  at the beginning of the step.

#### 7. Notes

7.1 **Choice of scaling factor.** The best choice of the positive integer  $v$  is the largest which does not cause overflow. A good rule is to keep all the  $z_i$  less than 0.3.

The rounding errors will build up at a rate corresponding to the retention of an extra  $v$  binary digits in the variables; hence the desirability of making  $v$  as large as possible.

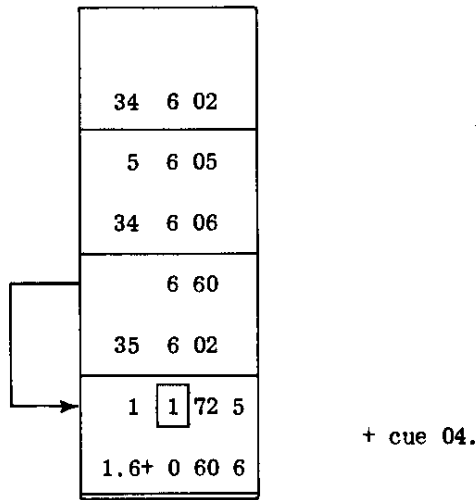
7.2 Re-entry after overflow

If overflow occurs during the calculation of one particular variable by R 402, it may be desired to use the overflow routine to indicate which variable has overflowed, and then re-enter R 402 with a view to identifying any further overflowing variables. When the dependent variable  $y_r$  stored in U5.R overflows in stage S of R 402, then

$5_B = S-1$

and  $5_p = R$

A suggested sequence for effecting re-entry to deal with the remaining variables is:-



A tag calling for cue 04 of R 402 must be punched in the appropriate place:-

Ra b -0 4
402 - 01 -

The overflow routine must leave the overflow indicator clear and the contents of  $5_m$ , U0 and the last  $n$  registers of U2, U3, U4 and U5 as it finds them.

Note that the cue to the overflow routine occurs in the parameter-list of R 402, with the cue to the auxiliary, and the preset link to the master-programme. Consequently it is simplest to punch the overflow routine together with the auxiliary. This is probably the most natural thing to do in any case, since it is at the time when the auxiliary is being drawn up that consideration will be given to the action needed on overflow.

7.3 Separation of Auxiliary from Master-Programme

If it is desired to avoid punching the auxiliary and the master-programme together, one may regard the latter as a programmer's subroutine and assign it a routine number. When used in this way the master-routine would normally have a cue-list containing two cues:-

- 01 for the initial entry (start of the programme),
- 02 for re-entry from R 402.

In the auxiliary there would be a tag calling for cue 02 to the master-routine. The preset link of R 402 must then be an order-pair leading to the location of this cue in the auxiliary. In order to gain entry to the master-routine after input when the Initial Orders are finished, a dummy master-programme must be punched. This will consist of an empty location, with a tag calling for cue 01 to the master-routine.

Similarly, the auxiliary may contain a cue to the overflow routine if it is desired to treat the latter as separate from the auxiliary.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

Issue 2.  
19.9.56.

## SIMULTANEOUS FIRST-ORDER DIFFERENTIAL EQUATIONS

( $8 \geq n$ ;  $v = 0$ , i.e. scaling factor unity)

The solution of  $n$  ( $\leq 8$ ) equations of the form

$$\frac{dy_i}{dx} = f_i(x, y_1, y_2, \dots, y_n) \quad (i = 1, 2, \dots, n)$$

by a modified Runge-Kutta process due to C. Strachey.

**Name:** RUNGE-KUTTA (S).  $8 \geq n, v=0$

**Store:** 4 blocks

**Uses:** U0 and the last  $n$  registers of U2, U3, U4 and U5; X5, X6, X7.

**Cues:** 01 for entry with link in X1.  
02 for entry when previously set link is to be used.  
03 for re-entry from auxiliary.

**Time:** About  $11n + 2$  milliseconds per step plus four times the time of the auxiliary.

**Link:** Obeyed in U0.3. See section 5.

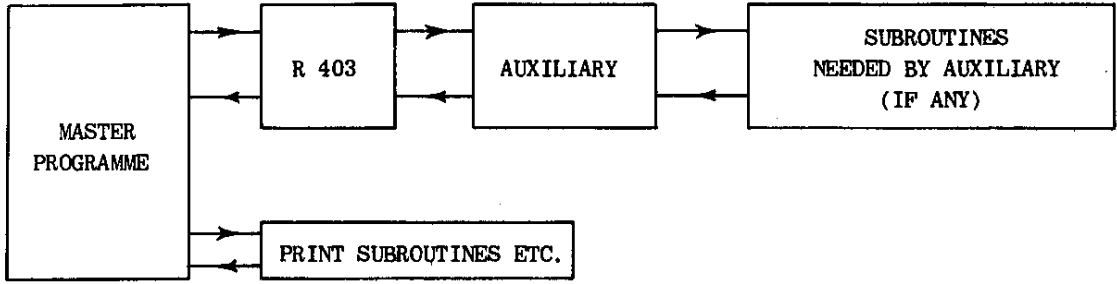
## 1. Method of Use

**1.1** At the start of the range of integration the master-programme must set the initial values of the  $n$  variables  $y_i$  in the last  $n$  registers of U5, and thereafter leave them as it finds them.

**1.2** Subsequently, each time R 403 is called in by the master-programme, the values of these variables will be advanced by one step (but see section 5.3 below); i.e. if  $x$  is the value of the independent variable when R 403 is called in, the values of the dependent variables  $y_i(x)$  will be replaced by  $y_i(x + h)$ , where  $h$  is the step-length.

**1.3** In order to use R 403 for a specific set of equations an auxiliary subroutine must be drawn up which defines the functions  $f_i$ ; this auxiliary is called in as required by R 403; it should be written in the way described in section 2 below. In addition there must of course, be a master-programme, so that a block diagram of a typical programme would be as follows:-





**2. The Auxiliary**

The  $n$  functions  $f_i$  are defined by an auxiliary subroutine (referred to as the auxiliary) which must be drawn up specially for each problem. The auxiliary is called in four times by R 403 during each step of the integration. It should have the following specification:-

1. Taking the values of  $y_i$  from the last  $n$  registers of U5 it should find  $z_i = \frac{1}{2}hf_i$ , where  $h$  is the step-length, and place them in the corresponding registers of U3. To avoid overflow of intermediate quantities the  $z_i$  should be kept less than about 0.3.
2. When finished it must return to R 403 by cue 03.
3. It must leave  $5_m$  and the last  $n$  registers of U5, U4 and U2 as it finds them.
4. It must leave OVR clear.

**3. Preset-Parameters**

The parameter list should normally be a part of the auxiliary, and should be punched as follows:-

	R 0 0 -0 5		
	403 - 04 -		Title
01		}	Cue to auxiliary
02	(0.7 0 60)		
	0	}	Cue to OVR (Loop stop shown)
03	$n$ - -0 0.		
	0		Number of variables = $n$
04		}	Main link if preset
05			
			in 0.3
			in 0.4

#### 4. Overflow

Preset-parameter 02 is a cue which will be obeyed if overflow occurs during the operation of R 403 (as distinct from the auxiliary). The order-pair shown in the parameter-list is a loop-stop. The notes below (section 7.1) contain suggestions for alternative action on overflow.

#### 5. The Main Link

**5.1** This is an order-pair which is obeyed in U0.3 after R 403 has served its purpose, in order to return control to the master-programme.

If R 403 is used as a normal closed subroutine, the main link should be placed in X1 by the master-programme before calling in R 403 with cue 01.

**5.2** If desired the main link may be preset by writing it in preset-parameter 04 (see parameter-list, section 3 above). In this case R 403 should be entered with cue 02. Alternatively two order-pairs may be preset as parameters 04 and 05; these will be obeyed in U0.3 and U0.4. The order-pair after preset-parameter 05 in the programme of R 403 is a copy of cue 02 to R 403.

Thus if R 403 is required to advance the integration by  $m$  steps each time it is called in,  $m$  may be placed in  $2_c$  (or  $3_c$  or  $4_c$ ) by the master-programme. In this case the main link could be preset by putting parameters 04 and 05 equal to:-

04	0.5 2 67 0	Jump to cue 02 unless $2_c' = 0$ (Obeyed in 0.3)
05	B+ U 72 U.P 0 60	Cue to master-programme (Obeyed in 0.4)

The auxiliary must then leave  $2_c$  as it finds it, and the master and auxiliary must be punched together so as to have the same relativizer; the notes (section 7.2) indicate a way of avoiding this restriction.

**5.3** A single word main link may also be preset by using cue 01 with the link in X1, cue 02 being used for subsequent entries to R 403.

#### 6. The Independent Variable

The value of  $x$  is not normally used or evaluated by R 403. In general, however, it is used by the auxiliary. When it is required by the auxiliary or the master-programme, it may conveniently be obtained by introducing an extra dependent variable satisfying

$$\frac{dy}{dx} = 1.$$

The corresponding value  $z = \frac{1}{2}h$  may be set once and for all by the master-programme at the beginning of the range of integration; it will not be disturbed by R 403, and the auxiliary should also be arranged so as not to disturb it.

If  $x$  is to be generated otherwise than as an extra variable, it must be given the values  $x_0$ ,  $x_0 + \frac{1}{2}h$ ,  $x_0 + \frac{1}{2}h$  and  $x_0 + h$  respectively when the auxiliary is entered for the first, second, third and fourth times in each step of the integration, where  $x_0$  is the value of  $x$  at the beginning of the step.

## 7. Notes

7.1 It may be desirable not to stop when the first variable overflows, but to complete the stage of R 403, and then take appropriate action, e.g. by printing out the names of the variables which have gone out of range. A method of identifying the variable and then re-entering R 403 in the appropriate place is required; this may be effected by using  $5_m$ , which is  $(s-1).p$  when the variable stored in  $5.p$  has overflowed during the  $s$ -th stage of integration. A suggested sequence for effecting re-entry to deal with the remaining variables if the overflow routine is obeyed in U1, is:-

5	6	00
34	6	01
①	6	51
34	6	05
②	6	51
0.2	0	60 6

The overflow routine must leave the overflow indicator clear, and the contents of  $5_m$ , U0, and the last  $n$  registers of U5, U4, U3 and U2 as it finds them. It must also form part of the auxiliary, in order to get the same relativizer.

7.2 If it is desired to avoid punching the auxiliary and master-programme together, one may regard the latter as a programmer's subroutine, and assign it a routine number. When used in this way the master-routine would normally have a cue-list containing two cues:-

01 for the initial entry (start of the programme),  
02 for re-entry from R 403.

In the auxiliary there would be a tag calling for cue 02 to the master-routine. The preset link of R 403 must then contain an order-pair leading to the location of this cue in the auxiliary. In order to gain entry to the master-routine after input when the Initial Orders are finished, a dummy master-programme must be punched. This will consist of an empty location, with a tag calling for cue 01 to the master-routine.

FERRANTI LTD.

## PEGASUS LIBRARY SPECIFICATION

Issue 1  
20.3.58.

FLOATING-POINT SIMULTANEOUS FIRST-ORDER DIFFERENTIAL EQUATIONS ( $n > 8$ )

The solution of  $n (> 8)$  equations of the form

$$\frac{dy_i}{dx} = f_i(x, y_1, y_2, \dots, y_n) \quad (i = 1, 2, \dots, n),$$

using floating-point arithmetic, by a modified Runge-Kutta process due to C. Strachey. The routine may be used with  $n \leq 8$  but will be slightly slower than R 407.

Name: F.P. RUNGE-KUTTA (S).  $n > 8$ .

Store: 14 blocks.

Uses: The entire Computing Store;  $4 \left( \frac{n+1}{8} \right)_u$  blocks of Main Store working space (see section 1.1 below).

Cues:

01	$4+ \boxed{0} 72$ $0.4 \ 0 \ 60$	For entry with link in X1.
02	$4+ \boxed{0} 72$ $0.4+ \ 0 \ 60$	For entry when previously set link is used.
03	$4+ \boxed{0} 72 \ 3$ $0.2 \ 0 \ 60$	For re-entry from the auxiliary.
04	$0+ \ 0 \ 00 \ 0.$ $0$	$a$ -order partial cue.
05	$0$ $0+ \ 0 \ 00 \ 0.$	$b$ -order partial cue.

**Time:** Approximately  $4t + 490n + 300$  milliseconds per step, where  $t$  = time for the auxiliary, excluding the time for entry and exit.

**Link:** Obeyed in 0.7. Not left in X1. See section 5.

### 1. Method of Use

1.1 R 405 takes its  $n$  variables  $y_i$  straight from the Main Store; each variable  $y_i$  has associated with it three quantities  $z$ ,  $p$  and  $q$ , which also require space in the Main Store. The arrangement is as follows:-

Block	$b$	contains	$z$	}	(Values for the first eight variables)
	$b + 1$	"	$y$		
	$b + 2$	"	$p$		
	$b + 3$	"	$q$		
	$b + 4$	"	$z$		
	$b + 5$	"	$y$		
	$b + 6$	"	$p$		
	$b + 7$	"	$q$		
	.....	etc.	.....		

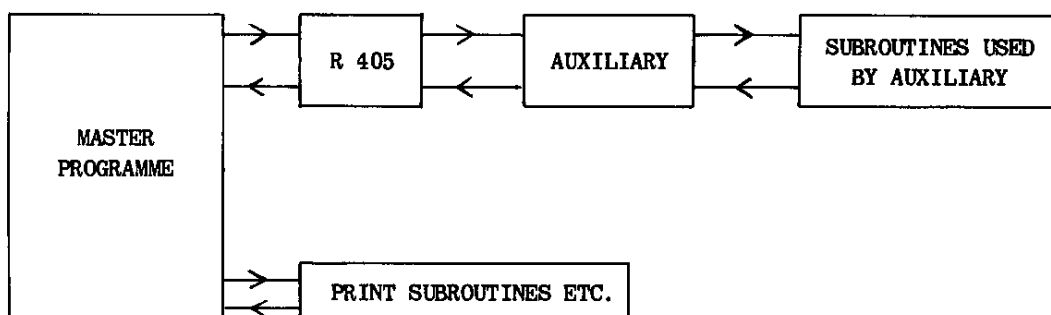
$b$  is a preset parameter set as described in section 3. The routine will use working space up to the end of block  $b - 1 + 4\left(\frac{n+1}{8}\right)_u$ , where  $\left(\frac{n+1}{8}\right)_u = \frac{n+1}{8}$  rounded

up to the next highest integer. Note in particular that when  $n$  is a multiple of 8 the routine will overwrite 4 blocks after the last value of  $q$ ; for example if  $n = 16$ , blocks  $b$  to  $b + 11$  will be overwritten.

1.2 At the start of a range of integration, the initial values of  $y$  must be set in the appropriate blocks of the Main Store by the master programme. Thereafter the master programme must leave the blocks for  $y$  as it finds them, whenever it is called in during a range of integration; it may use the  $z$ ,  $p$  and  $q$  blocks temporarily, but these blocks will be overwritten during each step of integration.

1.3 Each time R 405 is called in by the master programme the values of the variables will be advanced by one step (but see section 5.4); i.e. if  $x$  is the value of the independent variable and R 405 is called in, the values of the dependent variables  $y_i(x)$  will be replaced by those of  $y_i(x+h)$ , where  $h$  is the step length.

1.4 In order to use R 405 for a specific set of equations, an auxiliary subroutine must be drawn up which defines the functions  $f_i$ . This auxiliary is called in four times during each step of R 405; it should be written as described in section 2 below. In addition there must, of course, be a master programme, so that a block diagram of a typical programme would be as follows:-



## 2. The Auxiliary

The  $n$  functions  $f_i$  are defined by an auxiliary subroutine (referred to as the auxiliary) which must be drawn up specially for each set of equations to be solved. It may make use of the floating-point routine described in section 3.

The auxiliary should have the following specification:-

1. Taking the values of  $y_i$  from blocks  $b + 1$ ,  $b + 5$ ,  $b + 9$ , etc. in the Main Store, it should find  $z_i = \frac{1}{2}hf_i$ , where  $h$  is the step length, and place them in the corresponding locations in blocks  $b$ ,  $b + 4$ ,  $b + 8$ , etc.
2. It may use U0, 2, 3, 4, 5. If it overwrites U1 it must restore block 12+ of R 405 to U1 before returning to R 405.
3. It must leave X3 as it finds it.
4. It must leave the overflow indicator clear.
5. On entry to the auxiliary X5 contains  $\{(b - 4).0, n + 1\}$ . This may be used by the auxiliary for counting and modifying; it need not be restored on exit.
6. When finished, the auxiliary must return to R 405 by obeying cue 03.

## 3. Floating-Point Arithmetic

Once R 405 has been entered there is a self-preserving floating-point routine for addition and subtraction stored in U1. This may be used by the auxiliary or the master programme if desired. This routine uses U1; X1, 2, 4, 6, 7. Two orders (which need not form an order-pair) are needed for each entry to the routine. The first order sets a Computing Store link in X1, and is always as follows:-

A	1 40
---	------

where  $A$  is the Computing Store address to which the routine is to jump on exit.

The form of the second order depends on the operation to be carried out:-

For addition

1.2 0 60
----------

$$F(6)' = F(6) + F(7)$$

For subtraction

1.1 0 60
----------

$$F(6)' = F(6) - F(7)$$

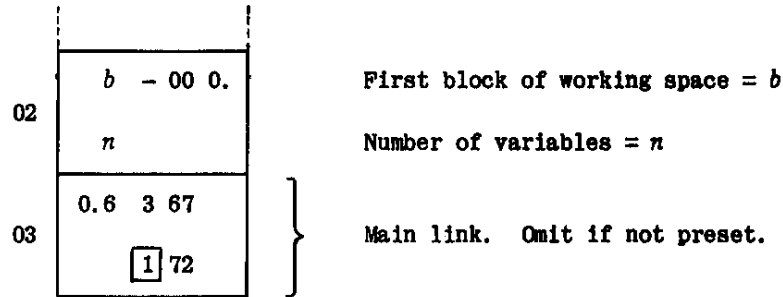
where  $F(X)$  denotes the floating-point number in accumulator  $X$ .

## 4. Preset Parameters

R 405 requires a parameter list which should normally form part of the auxiliary, and should be punched as follows:-

R 0 0 -0 3	}	Title
405 - 04 -		
<div style="display: flex; align-items: center; justify-content: center;"> <div style="border: 1px solid black; width: 15px; height: 15px; margin-right: 5px;"></div> <span>72</span> </div>	}	Cue to auxiliary
0 60		

01



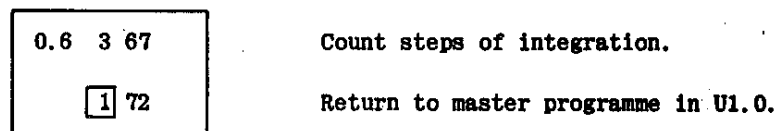
## 5. Main Link

5.1 When R 405 has completed one step in the integration process it will obey the main link in U0.7.

5.2 If R 405 is used as a normal closed subroutine, the main link should be placed in X1 before calling in R 405 with cue 01. This link will be planted in the Main Store in place of any previously set link.

5.3 If desired, the main link may be preset as one order-pair in parameter 03. In this case R 405 must be called in with cue 02. When the main link is preset, the master programme and the auxiliary should be punched together so that they both have the same relativiser in the parameter list (but see section 7.2 of R 401 for an alternative procedure). Cue 02 may also be used if it is required to use the same link as was set on the previous entry by cue 01.

5.4 The main link may be a block transfer and jump calling in the master programme. If, however, it is required to perform several steps of integration before returning to the master programme, the link may be of the form



The number of steps must be set in  $3_c$  before entry to R 405; the link is obeyed in 0.7 and return to the master programme must always be to U1.0 as shown.

## 6. The Independent Variable

The value of  $x$  is not normally used or evaluated by R 405. In general, however, it is used by the auxiliary. When it is required by the auxiliary or by the master programme, it may be obtained by introducing an extra dependent variable satisfying

$$\frac{dy}{dx} = 1.$$

In this case the corresponding value  $z = \frac{1}{2}h$  may be set once and for all by the master programme at the beginning of the range of integration; it will not be touched by R 405. (The auxiliary and the master programme should be arranged so as to leave it undisturbed). It is convenient to hold the independent variable in the last block of  $y$ 's. If it is  $y_{sm} + p$ , it will be found by the auxiliary in U5. $p$ .

If  $x$  is to be generated otherwise than as an extra dependent variable, it must be given the values  $x_0$ ,  $x_0 + \frac{1}{2}h$ ,  $x_0 + \frac{1}{2}h$ ,  $x_0 + h$  respectively when the auxiliary is entered for the first, second, third and fourth times in each step of the integration, where  $x_0$  is the value of  $x$  at the beginning of the step.

## 7. Floating-Point Working

7.1 R 405 works entirely in single length floating-point arithmetic as described in the Pegasus Programming Manual and in the specifications of R 11, R 610 and other floating-point subroutines. Nine binary digits are used to represent the exponents of the floating-point numbers.

7.2 The floating-point sequence tests for floating-point underflow but will give incorrect results if floating-point overflow occurs. In very rare special cases it is possible for the subtract sequence to set OVR, causing a writing with OVR stop in R 405.

7.3 The integer  $-2^9$  is stored in U1.0 of the routine and may be used, but not overwritten, by the auxiliary.

Author: Dr. G.N. Lance of the University of Southampton.

Ferranti Ltd.,  
London Computer Centre,  
21, Portland Place,  
LONDON, W.1.

*Issue 1*  
20th March, 1958.  
G.N.L. T.F.



FERRANTI LTD

PEGASUS LIBRARY SPECIFICATION

Issue 1  
20.3.58.

FLOATING-POINT SIMULTANEOUS FIRST-ORDER DIFFERENTIAL EQUATIONS (8 ≥ n)

The solution of n(≤ 8) equations of the form

$$\frac{dy_i}{dx} = f_i(x, y_1, y_2, \dots, y_n) \quad (i = 1, 2, \dots, n),$$

using floating-point arithmetic, by a modified Runge-Kutta process due to C. Strachey.

Name: F.P. RUNGE-KUTTA (S). 8 ≥ n.

Store: 10 blocks.

Uses: U0, 1 and the last n registers of U2, U3, U4 and U5; X1, 2, 4, 5, 6, 7.

Cues:

01	0+ 0 72 0.4+ 0 60
----	----------------------

for entry with link in X1.

02	0+ 0 72 0.5 0 60
----	---------------------

for entry when previously set link is to be used.

03	0+ 0 72 5 0.0 0 60
----	-----------------------

for re-entry from the auxiliary.

04	0+ 0 00 0. 0
----	-----------------

a-order partial cue.

05	0 0+ 0 00 0.
----	-----------------

b-order partial cue.

Time: Approximately 4t + 480n + 60 milliseconds per step, where t = time for the auxiliary, excluding the time for entry and exit.

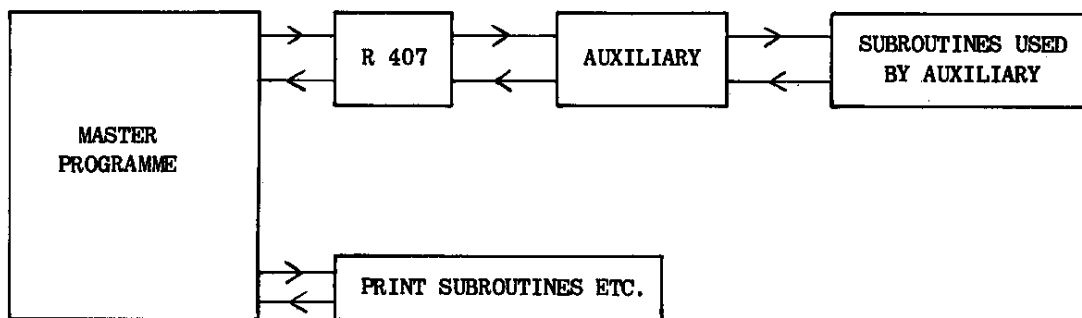
Link: Obeyed in 0.2. Not left in X1. See section 5.

## 1. Method of Use

1.1 At the start of the range of integration the master programme must set the initial values of the  $n$  variables  $y_i$  in the last  $n$  registers of U5, and thereafter leave them as it finds them.

1.2 Subsequently, each time R 407 is called in by the master programme, the values of these variables will be advanced by one step (but see section 5.4 below); i.e. if  $x$  is the value of the independent variable when R 407 is called in, the values of the dependent variables  $y_i(x)$  will be replaced by  $y_i(x + h)$ , where  $h$  is the step-length.

1.3 In order to use R 407 for a specific set of equations, an auxiliary subroutine must be drawn up which defines the functions  $f_i$ ; this auxiliary is called in as required by R 407; it should be written in the way described in section 2 below. In addition there must, of course, be a master programme, so that a block diagram of a typical programme would be as follows:-



## 2. The Auxiliary

The  $n$  functions  $f_i$  are defined by an auxiliary subroutine (referred to as the auxiliary) which must be drawn up specially for each set of equations to be solved. It may make use of the floating-point routine described in section 3.

The auxiliary is called in four times by R 407 during each step of the integration. It should have the following specification:-

1. Taking the values of  $y_i$  from the last  $n$  registers of U5, it should find  $z_i = \frac{1}{2}hf_i$ , where  $h$  is the step-length, and place them in the corresponding registers of U3.
2. It may use U0, U3 and the first  $8-n$  registers in U2, 4 and 5. If it uses U1 it must restore block 8+ of R 407 to U1 before returning to R 407.
3. It must leave  $5_m$  and the last  $n$  registers of U2, 4 and 5 as it finds them.
4. If it leaves X3 as it finds it, this accumulator will be available to the master programme for counting steps of integration. See also section 5.4.
5. It must leave the overflow indicator clear.
6. On entry to the auxiliary  $5_p$  contains  $8-n$ , and this may be used, by means of a 66 order, to count and modify through the  $n$  variables. Note, however, that the whole of  $5_m$ , not merely  $5_p$ , must be restored to its original value before returning to R 407.
7. When finished, the auxiliary must return to R 407 by obeying cue 03.
8. On entry to the auxiliary the integer  $-2^9$  is stored in U1.0. This may be used as a collating mask. If 1.0 is overwritten U1 must be restored as described in paragraph 2 above.

### 3. Floating-Point Arithmetic

Once R 407 has been entered there is a self-preserving floating-point routine for addition and subtraction stored in U1. This may be used by the auxiliary or the master programme if desired. This routine uses U1; X1, 2, 4, 6, 7. Two orders (which need not form an order-pair) are needed for each entry to the routine. The first order sets a Computing Store link in X1, and is always as follows:-

A	1 40
---	------

where  $A$  is the Computing Store address to which the routine is to jump on exit.

The form of the second order depends on the operation to be carried out:-

For addition

1.2 0 60
----------

$$F(6)' = F(6) + F(7)$$

For subtraction

1.1 0 60
----------

$$F(6)' = F(6) - F(7)$$

where  $F(X)$  denotes the floating-point number in accumulator  $X$ .

### 4. Preset Parameters

R 407 requires a parameter list which should normally form part of the auxiliary, and should be punched as follows:-

R 0 0 -0 4	}	Title
407 - 04 -		
01 <span style="border: 1px solid black; padding: 2px 5px;">72</span>	}	Cue to auxiliary
0 60		
02 $n - -0 0.$	}	Number of variables = $n$
0		
03 0.4 3 67	}	Main link if preset (see sections 5.3 and 5.4)
0		
04 <span style="border: 1px solid black; padding: 2px 5px;">72</span>	}	Obeyed in U0.2
0 60		
		}
		Obeyed in U0.3

### 5. Main Link

5.1 When R 407 has completed one step in the integration process it will obey the main link in U0.2.

5.2 If R 407 is used as a normal closed subroutine, the main link should be placed in X1 before calling in R 407 with cue 01. This link will be planted in the Main Store in place of any previously set link.

5.3 If desired, the main link may be preset as one order-pair in parameter 03 or as two order-pairs in parameters 03 and 04. In this case R 407 should be called in with cue 02. When the main link is preset, the master programme and the auxiliary should be punched together so that they both have the same relativiser in the parameter list (but see section 7.2 of R 403 for an alternative procedure). Cue 02 may also be used if it is required to use the same link as was set on the previous entry by cue 01.

5.4 The main link may be a block transfer and jump calling in the master programme. If, however, it is required to perform several steps of integration before returning to the master programme, the link may be of the form

0.4 3 67	Count steps of integration.
0 72	Return to master programme in U0.3.

The number of steps must be set in  $3_c$  before entry to R 407 and  $3_c$  must not be spoiled by the auxiliary. The link is obeyed in 0.2 and return to the master programme must always be to U0.3 as shown.

If the link is preset as two order-pairs, counting may be effected by using the orders shown in section 4.

## 6. The Independent Variable

The value of  $x$  is not normally used or evaluated by R 407. In general, however, it is used by the auxiliary. When it is required by the auxiliary or the master programme, it may conveniently be obtained by introducing an extra dependent variable satisfying.

$$\frac{dy}{dx} = 1.$$

The corresponding value  $z = \frac{1}{2}h$  may be set once and for all by the master programme at the beginning of the range of integration; it will not be disturbed by R 407, and the auxiliary should also be arranged so as not to disturb it.

If  $x$  is to be generated otherwise than as an extra variable, it must be given the values  $x_0$ ,  $x_0 + \frac{1}{2}h$ ,  $x_0 + \frac{1}{2}h$  and  $x_0 + h$  respectively when the auxiliary is entered for the first, second, third and fourth times in each step of the integration, where  $x_0$  is the value of  $x$  at the beginning of the step.

## 7. Floating-Point Working

R 407 works entirely in single length floating-point arithmetic as described in the Pegasus Programming Manual and in the specifications of R 11, R 610 and other floating-point subroutines. Nine binary digits are used to represent the exponents of the floating-point numbers. The floating-point sequence tests for floating-point underflow but will give incorrect results if floating-point overflow occurs. In very rare special cases it is possible for the subtract sequence to set OVR.

Author: Dr. G.N. Lance of the University of Southampton.

Ferranti Ltd.,  
London Computer Centre,  
21, Portland Place,  
LONDON, W.1.

Issue 1  
20th March 1958.  
G.N.L. T.F.

Copyright Reserved

FERRANTI LTD

PEGASUS LIBRARY SPECIFICATION

SIMULTANEOUS FIRST-ORDER DIFFERENTIAL EQUATIONS (8 ≥ n) (Simpson's rule with accuracy control).

The purpose of R 408 is to find the yi, whose values are given initially when x = x0, at points x0, x0 + δx, x0 + 2δx,..... xi with interval δx up to x1, from the n (≤ 8) first order differential equations.

dyi/dx = fi(x, y1, y2, ..... yn) (i = 1, 2, ..... n)

by the use of Simpson's formula, and with a total truncation error in each variable yi at the end of the range, that is when x = x1, limited to a previously fixed amount k.

A predictor-corrector technique is used, and the step length is automatically adjusted so that the rate of increase of truncation error is less than k/|x1 - x0|; but if instability occurs, that is the equations cause deviations in yi at one point to be increased at a later point, the final error may exceed k.

The length of step is variable, depending on the equations and the accuracy required, and so is the time of running. Simpson's formula has the great advantage that it needs the auxiliary to find fi only once per step.

- Name: DIFF EQUNS (SIMPSON) 8 ≥ n
Store: 18 blocks
Uses: The entire Computing Store except for the first 8 - n registers of U3, U4, U5; 3 + n blocks of the Main Store starting from B 389+ (see section 4).
Cues: 01 (0+. 0) First entry to R 408
02 (4+. 3) Re-entry from the control routine
03 (4+. 1) Re-entry from the auxiliary
04 [8+ 72 / 1.6 0 60] Used only with R 409 (see section 6 in the specification of R 409)
05 a-order partial cue
06 b-order partial cue
Time: 19n + 25 + T milliseconds per step for n ≤ 4, 19n + 41 + T milliseconds per step for n > 4, where each use of the auxiliary takes T milliseconds.
Link: Exit from R 408 is effected via the control routine by obeying preset parameter 01 (obeyed in 0.2).

**1. Method of Use**

**1.1** The master programme must place the initial  $y_i$  when  $x = x_0$  in the last  $n$  registers of U5, and set the following information in U2:

U2.0  $2^{-m} \frac{\delta x}{3}$ ,  $-3 \leq \delta x < 3$  (see section 6)

U2.1  $2^{-13-m}$

U2.2  $\eta = \frac{180 k}{|x_1 - x_0|}$  where  $10^{-9} < \eta \leq \frac{1}{2}$  (see section 7)

U2.3  $\frac{1}{2} k$  (see section 7)

$m$  is an integer  $\geq 0$  and  $2^{-m} \delta x$  defines the step length.  $m$  may be set equal to 0, but if it is set such that  $2^{-m} \delta x$  is the proper length for a fourth order integration at the beginning, the subroutine is a little quicker.

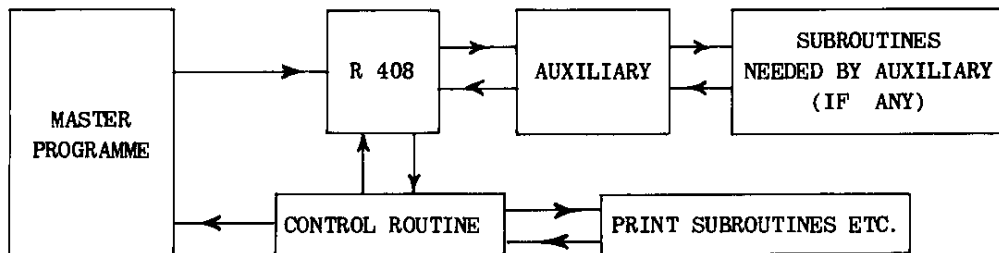
After setting the above values, the master programme must enter R 408 by cue 01.

**1.2** In order to use R 408 for a specific set of equations an auxiliary subroutine must be drawn up (see section 2). This auxiliary defines the functions  $f_i$  and is called in as required by R 408.

**1.3** R 408 enters a control routine when  $x = x_0, x_0 + \delta x, x_0 + 2\delta x, \dots$ , by obeying preset parameter 01. This routine must be provided by the user to control the number of intervals,  $\delta x$ , and to print intermediate values if required. On entry the whole Computing Store, except U0, is as it was left by the auxiliary.

The control routine may use the entire Computing Store except for X2 and the last  $n$  registers of U3, 4 and 5; it should return to R 408 by obeying cue 02.

**1.4** A block diagram of a typical programme would be as follows:



**2. The Auxiliary**

**2.1** An auxiliary routine is required to calculate the  $f_i$  from the  $y_i$  in U5 and from  $x$  given by

$$x = x_0 + 2^{13} \delta x C(2)$$

and place them in the corresponding addresses in U3. They are preserved by R 408. The inaccuracy in the  $f_i$  should be kept less than  $\eta/250$  (see section 1.1).

The cue to the auxiliary must be specified as preset parameter 02 of R 408, and the auxiliary must return to R 408 via cue 03.

**2.2** The auxiliary may use the whole Computing Store, except for X2 and the last  $n$  registers of U4 and 5.

### 3. Preset Parameters

The parameter list should normally be punched after the auxiliary and control subroutine (and shift revision if used) and should take the following form:

	R 0 0 -0 4	
	408 - 04 -	
01		} Cue to the control routine
02		
03	(8-n) - -0 0. 0	} $n =$ number of variables
04	+0	
		} or cue to Shift Revision if R 409 is used (see specification of R 409, section 6).

If the auxiliary and control routine are punched as one programme the parameter list may be punched as part of that programme, and relative addresses may then be used for PP 01 and 02. Alternatively, the appropriate relativizer may be set by a C-directive before each parameter provided that both routines have previously been read as Assembly subroutines.

### 4. Main Store Working Space

If it is inconvenient to allow R 408 to use  $3 + n$  blocks of Main Store starting at B 389+, this address can be changed by a C-sequence on the programme tape after the A2:

```
C 408
X 4+
M -000.
X 14+
M -500.
```

The new address  $M$  may be absolute, or relative to R 408. Unless  $M$  differs from 5+ by a multiple of 16 blocks there will be a loss of 16 or 32 milliseconds per step on drum transfers.

### 5. Overflow

R 408 may set the overflow indicator if any  $y_i$  exceeds the range of the machine, or if  $|f_i| > 1/6$  for any  $i$ . There will be a write with overflow stop, and the variable responsible can be identified by the position part of the modifier in X3.

R 409 is an optional addition to R 408, which will prevent overflow by shifting down the variables whenever necessary.

### 6. Rounding Errors

Owing to possible shifts down of the number  $2^{-m}\delta x/3$ , initially in U2.0, in the course of the calculation, digits may be lost which are not restored. If the very highest accuracy is required (say  $k = 10^{-10}$ ) and the steps are expected to become small, the following arrangement may be preferable. Place  $-2^{-m-b}$  in U2.0 initially instead of  $2^{-m}\delta x/3$ , and let the auxiliary place  $-2^b f_i \delta x/3$  in U3 instead of  $f_i$ , where

$b \geq 0$ . For an accuracy  $k$ ,  $\frac{60 k |\delta x| 2^b}{|x_1 - x_0|}$  is to replace  $\frac{180 k}{|x_1 - x_0|}$  in U2.2 initially.

The rounding error in the  $y_i$ , supposed random, increases in proportion to the square root of the number of steps as it accumulates in  $y_i$ , whereas that in the multiplier  $2^{-m}\delta x/3$  ( $m$  variable) is added on continually.

### 7. Truncation Error

The constant in U2.2 limits the rate of accumulation of integration error and that in U2.3 the amount of that of extrapolation. They are supposed to contribute equally to the final error, but are really independent. If both constants are large [ $C(2.2) \leq \frac{1}{2}$ ], the integration except at the start will be in equal steps  $\delta x$ .

**Author:** Mr. F.E. Radcliffe.



FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

**SIMULTANEOUS FIRST-ORDER DIFFERENTIAL EQUATIONS ( $8 \geq n$ )**  
(Simpson's rule with accuracy control and scaling)

R 409 is an optional addition to R 408 to prevent overflow of the variables. It serves to shift down the  $y_i$  and the  $f_i$  a number of places  $\nu$  so that  $2^{-\nu}f_i$  replace the  $f_i$  and  $2^{-\nu}y_i$  replace the  $y_i$  and

$$- \frac{1}{2} \leq 2^{-\nu}y_i < \frac{1}{2}$$

$$- \frac{1}{16} \leq 2^{-\nu}f_i < \frac{1}{16}$$

The shifts are made as required.

The limit of accuracy,  $k$ , used in R 408, is no longer observed, but becomes  $k \cdot 2^\nu$ . For this reason it may be advisable to print the final value of  $\nu$ .

**Name:** DIFF EQUNS (SIMPSON) WITH SHIFTS ( $8 \geq n$ )

**Store:** 2 blocks + 18 blocks for R 408

**Uses:** The entire Computing Store except for the first  $8 - n$  registers of U3, U4, U5;  $3 + n$  blocks of the Main Store starting from B 389+ relative to the beginning of R 408 (see specification of R 408).

**Cue:** 01 (0+ .1)

**Time per step** 18 milliseconds for  $n \leq 6$   
**additional to R 408:** 34 milliseconds for  $n > 6$

**Link:** R 409 automatically returns control to R 408.

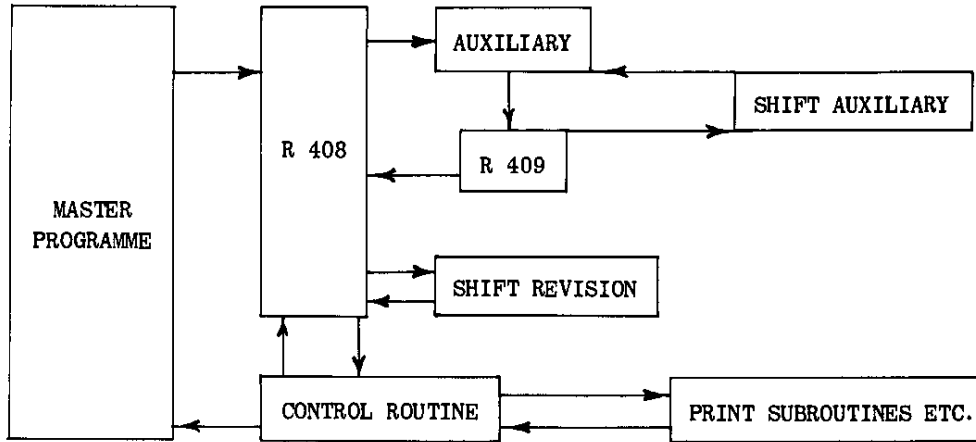
**1. Method of Use**

**1.1** The master programme must place the initial  $y_i$  (or the  $2^{-\nu}y_i$ ) in the last  $n$  registers of U5, and U2.0 to U2.3 should be set as for R 408, (section 1.1).

When  $n < 8$  an address  $K$  is to be set aside among the first  $8 - n$  registers of U3 or U5 to hold  $\nu$  in the modifier position. Initially  $K_m = 0$  or  $\nu_0$  must be set by the master programme, and afterwards  $K_m = \nu$ . When  $n = 8$ , section 6 will apply.

**1.2** In order to use R 409, in conjunction with R 408, for a specific set of equations an auxiliary subroutine must be drawn up (see section 2). This auxiliary defines the functions  $2^{-\nu}f_i$  and is called in as required by R 408. A control routine is required by R 408 to control the number of intervals,  $\delta x$ , and to print intermediate values if required.

A shift auxiliary routine (see section 5) and a shift revision routine (see section 6) may be required. A block diagram of a typical programme may then be as follows:



1.3 On returning to the control routine, numbers left by the auxiliary in X1, 6, 7, U0, 1, 2 will have been altered by R 409. It may not be assumed that any number dependent on  $\nu$  other than  $\nu$  itself in  $K_m$ ,  $2^{-\nu}y_i$  and  $2^{-\nu}f_i$  are correct.

The control routine may use the entire Computing Store except for X2,  $K_m$  and the last  $n$  registers of U3, 4 and 5: it should return to R 408 by obeying cue 02.

**2. The Auxiliary**

2.1 The auxiliary routine is required to calculate the  $2^{-\nu}f_i$  from the  $2^{-\nu}y_i$  in U5 and from  $x$  given by

$$x = x_0 + 2^{13} \delta x C(2)$$

and place them in the corresponding positions in U3.  $\nu$  is in  $K_m$ .

The cue to the auxiliary should be specified as preset parameter 02 of R 408, and the auxiliary must exit by obeying cue 01 to R 409.

2.2 The auxiliary may use the whole of the Computing Store except for X2,  $K_m$  and the last  $n$  registers of U4 and 5.

**3. Preset Parameter**

R 409 requires the following parameter list:

R	0	0	-0	1
409	-	04	-	
35	1	00		
K	1	11		

} or cue to shift auxiliary if used.

where  $K$  is an address set aside among the first  $8 - n$  registers of U3 or U5 to hold  $\nu$ .

If there is a shift auxiliary routine, the parameter list may be punched as part of that routine and preset parameter 01 (now the cue to the shift auxiliary) may contain a relative address.

The parameter list for R 408 should be punched as described in section 3 of the specification to R 408.

#### 4. Main Store Working Space

See section 4 in the specification of R 408.

#### 5. The Shift Auxiliary

If parameters dependent on  $\nu$  (other than  $2^{-\nu}f_i$  and  $2^{-\nu}y_i$ ) are used by the auxiliary or the control routine, a special routine called the shift auxiliary must be written. This must add 1 to  $\nu$ , in the modifier position of  $K$ , and re-calculate all such parameters using this new value of  $\nu$ . These parameters should preferably be stored in free registers in U3 or U5.

The cue to the shift auxiliary must be specified as preset parameter 01 to R 409. The shift auxiliary may use the whole Computing Store except for X2 and the last  $n$  registers of U4 and 5. It should exit by obeying the cue to the auxiliary routine (section 2).

#### 6. The Shift Revision

If  $\nu$  and all parameters dependent upon it are stored in U3 and U5 the shift revision process is done automatically and the user need not read the rest of this section.

When R 408 reduces the size of a step, and at the start, it may be necessary for it to go back to a previous value of  $x$ . In these circumstances the contents of U3, U5 and X2 (which are sufficient for the subroutine) are put back to what they were before. Thus when shifts are used,  $\nu$  and parameters dependent upon it, if kept in the free parts of U3 and U5, will be automatically restored to their previous values.

But if  $\nu$ , or any number dependent on  $\nu$ , is stored elsewhere its previous value will not be automatically restored. An additional programme known as the shift revision routine is then required. Its cue is specified as preset parameter 04 of R 408, written as a stop order pair. It may use the entire Computing Store except for U2, 3 and 5. It must exit by obeying cue 04 to R 408.

If  $n < 8$ ,  $\nu$  will normally be stored in U3 or U5. When the shift revision routine is entered it should first test X3. If  $x_3 \geq 0$ ,  $\nu$  has not been changed and there is no need to change the numbers dependent on it. If  $x_3 < 0$ , the routine should take  $\nu$  from  $K_m$  and use it to re-calculate numbers dependent on  $\nu$  which are not stored in U3 and U5.

If  $n = 8$ ,  $\nu$  must be held in a Main Store location which will be called the current location: this could conveniently be a word in the auxiliary programme. A second location, known as the reserve location is also required. If the shift revision routine is entered with  $x_3 \geq 0$  the number  $\nu$  in the current location must be copied into the reserve location. If  $x_3 < 0$ , the number in the reserve location must be copied into the current location as the new value of  $\nu$ , and the numbers in the Main Store dependent on  $\nu$  must be re-calculated using this new value. (Note that reserve  $\nu \leq$  current  $\nu$ ).

Author: Mr. F.E. Radcliffe.

© FERRANTI LTD 1960

Ferranti Ltd.,  
London Computer Centre,  
68, Newman Street,  
LONDON, W.1.

*Not to be reproduced in whole or  
in part without the prior written  
permission of Ferranti Ltd.*

*Issue 1*  
8th November, 1960  
F.E.R. M.J.M.

## FERRANTI LTD

## PEGASUS LIBRARY SPECIFICATION

**SIMULTANEOUS FIRST-ORDER DIFFERENTIAL EQUATIONS ( $8 \geq n$ )**  
(Kutta-Merson process).

The solution of  $n$  ( $\leq 8$ ) equations of the form

$$\frac{dy_i}{dx} = f_i(y_1, y_2, \dots, y_n) \quad (i = 1, 2, \dots, n)$$

by a Kutta process designed by R.H. Merson. The first dependent variable  $y_1$  is always a scaled version of the independent variable  $x$ , so that the subroutine is effectively restricted to the solution of up to seven first-order equations.

This is a similar subroutine to R 403, but is more comprehensive than the latter. It is designed to perform a number of steps of integration and has a built-in control routine, an automatic interval-changing routine and automatic control of printing at a specified printing interval.

**Name:** KUTTA-MERSON  $8 \geq n$

**Store:** 11 blocks.

**Uses:** U0, 1, 2 and the last  $n$  registers of U3, 4 and 5; all the accumulators except X2, 3; B1.

**Cues:**

01	(0+.0)	Initial entry						
02	(3+.2)	Re-entry after printing						
03	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td>3+</td> <td style="border: 1px solid black; padding: 2px;">0</td> <td>725</td> </tr> <tr> <td>0.0</td> <td>0</td> <td>60</td> </tr> </table>	3+	0	725	0.0	0	60	re-entry from auxiliary
3+	0	725						
0.0	0	60						

**Time:** About  $19n + 44$  milliseconds per step plus five times the time of the auxiliary.

**Link:** Exit is made via the print routine (see section 3).

**1. Method of Use**

**1.1** The user must provide an auxiliary routine (section 2) for the calculation of the functions  $f_i$  and a print routine (section 3), and must also specify certain parameters (section 5).

**1.2** The initial values of the  $y_i$  should be stored with the auxiliary as described in section 5.5.

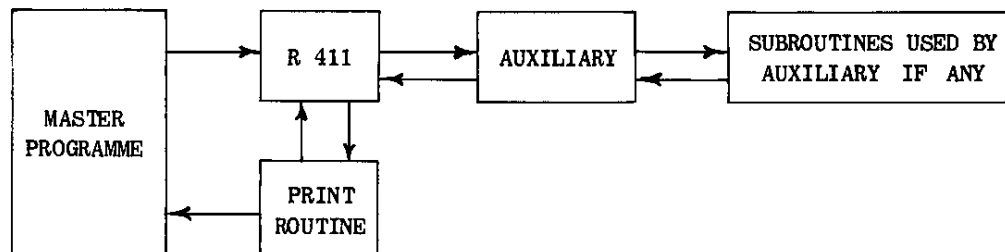
**1.3** The master programme should enter R 411 by obeying cue 01.

After certain preliminary calculations, R 411 enters the auxiliary to compute the  $f_i$ ; almost immediately after return from the auxiliary the print routine is entered. On re-entry to R 411 from the print routine, the main computation is performed, the values of the variables  $y_i(x)$  at the beginning of the step being replaced by  $y_i(x+h)$ , where  $h$  is the step length. Estimates of the elementary truncation errors in the  $y_i$  are obtained and compared with a preset maximum allowable error  $m$  chosen by the programmer. If any computed error exceeds  $m$ , the interval  $h$  is halved and the computation starts again from the beginning of the step. If all the computed errors are less than  $m/32$ , the interval  $h$  is doubled and the integration proceeds. Otherwise the computation proceeds with the same interval.

The interval  $h$  is always positive and an exact sub-multiple of 2 ( $h = 2^{-\alpha}$ ,  $\alpha \geq 0$ ). It is never held in the computer in this form, but the value of  $\alpha$  is stored in  $X4_m$ .

1.4 Exit to the master programme at the end of the integration is usually made from the print routine (section 3.3).

A block diagram of a typical programme using R 411 would be as follows:-



## 2. The Auxiliary

The functions  $f_i$  are defined by an auxiliary subroutine which must be drawn up specially for each problem. The auxiliary is called in five times by R 411 during each step of the integration. It should have the following specification:-

1. Taking the values  $y_2, y_3, \dots, y_n$  from the registers U5.6, 5.5, etc. respectively, it should compute  $z_i = hf_i/3$ , where  $h$  is the step length ( $h = 2^{-\alpha}$ , where  $\alpha$  will be in  $X4_m$ ) and place them in the corresponding registers U3.6, 3.5 etc.  $z_1$  corresponding to the independent variable  $y_1 = 2^{-\mu}x$  (see section 4), is set by R 411.
2. When finished it must return to R 411 by cue 03.
3. It must leave B1, X4, X5, U3.7 and the last  $n$  registers of U2, 4 and 5 as it finds them.

Note that the fraction  $\frac{1}{3}$  is represented most accurately by  $(2^{38} - 1)/3 = + 91625968981$ , and in this form is stored in 2+.7 of R 411.

## 3. The Print Routine

3.1 The print routine, to be drawn up by the user, should immediately follow the auxiliary and have the same relativiser. It is entered after return from the auxiliary at the beginning of a step, so that  $f$ 's as well as  $y$ 's may be printed. It is not necessarily entered at every step, control of the printing points being achieved by the parameters  $k$  and  $\alpha_p$  (section 5.4).

3.2 On entry to the print routine the values of the variables  $y_1, y_2, y_3 \dots$  are in U5.7, 5.6, 5.5, ....., and also in the corresponding positions in B1; the values of  $z_i = hf_i/3$  are in the corresponding positions of U3;  $h$ , the interval about to be tried, is represented by  $\alpha$  (where  $2^{-\alpha} = h$ ) in  $X4_m$ .

3.3 On exit, the print routine must normally return to R 411 by obeying cue 02. However, before doing this, it should test to see whether the required value of the independent variable has been reached, in which case it should exit to the master programme, the integration process having been completed.

3.4 The print routine must leave B1, X4, X5 and the last  $n$  registers of U3 and U5 as it finds them.

#### 4. The Independent Variable

The value of the independent variable  $x$ , is required by R 411 for control purposes and is obtained by introducing the dependent variable  $y_1$  as a scaled version of  $x$ , that is  $y_1 = 2^{-\mu}x$  say and the corresponding differential equation is

$$\frac{dy_1}{dx} = 2^{-\mu}$$

$\mu$  is specified in the parameter list and must be a non-negative integer.

The function  $z_1 = h \cdot 2^{-\mu}/3$  is computed by R 411 and stored in U3.7 just before the auxiliary is entered. It will sometimes be helpful to use  $z_1$  in the auxiliary (as a multiplying factor) whilst constructing the other  $z_i$ .

5. Preset Parameters

A parameter list should be drawn up as follows, and inserted after the AUXILIARY-plus-PRINT routine so that it may have the same relativiser:

	R 0 0 -1 3	Title of Parameter List
	411 - 04 -	
01	<input type="checkbox"/> 72 0 60	Cue to auxiliary
02	0.7 0 60 0	Cue to overflow routine (loop stop shown)
03	$n$ - -0 0. 0	$n$ = number of variables including $y_1$
04	<input type="checkbox"/> 72 0 60	Cue to print routine
05	$\alpha_I$ - -0 0. 0	$2^{-\alpha_I}$ = initial value of $h$
06	$\alpha_m$ - -0 0. 0	$2^{-\alpha_m} = h_m$ (maximum interval)
07	$\alpha_p$ - -0 0. 0	$2^{-\alpha_p} = h_p$ (normal printing interval)
10	$+m$	accuracy parameter
11	$k$ - -0 0. 0	sub-printing parameter
12	0 $\mu$ 0 00 0.	$y_1 = 2^{-\mu x}$
13	A 0 00 0. 0	A = number of block in which initial conditions are stored (may be relative to auxiliary if required).

5.1 The Overflow Routine

An overflow routine may be written, which will be entered if overflow occurs during the operation of R 411 (but not if overflow occurs during the auxiliary). The cue to this routine is set as P.P.02, and is obeyed in UO.7. If no special action is required on overflow it is suggested that a loop stop is used as shown above.

## 5.2 The Initial Interval

An initial trial value of the interval  $h$  must be specified by the parameter  $\alpha_I$ , where  $2^{-\alpha_I} = h_I$ .  $\alpha_I$  must be a non-negative integer and  $\alpha_I + \mu < 38$  (for if  $\alpha_I + \mu \geq 38$ ,  $\Delta y_1$  will be effectively zero and the computation will stick).

## 5.3. The Maximum Interval

Even apart from printing requirements it is not always desirable to use as great an interval as the normal procedure would allow. Provision is made for the user to restrict the maximum interval, if required, by setting the parameter  $\alpha_m$ , where  $2^{-\alpha_m} = h_{\max}$ .  $\alpha_m$  must be non-negative.

If no special restriction is required, set  $\alpha_m = 0$ .

## 5.4. Printing Parameters

The parameters  $\alpha_p$  and  $k$  determine whether or not the printing routine is entered at each point of the computation.  $h_p = 2^{-\alpha_p}$  is called the *normal printing interval*, and  $\alpha_p$  must be non-negative. At certain stages of the computation the variables may be changing more rapidly than normal and it may be required to print at a finer interval than the normal printing interval over this region. The facility to do this is provided by the parameter  $k$ , the action of which is described by the following rule: printing will occur if and only if  $x$ , the independent variable, is an exact multiple of the lesser of  $h_p$  and  $2^k h$ , when  $h$  is the interval which has just been used. The interpretation of this rule is as follows:

if $k = 0$	printing will occur at every step;
if $k = 1$	printing will occur at every other step if $h \leq \frac{1}{2} h_p$ , but at every step if $h = h_p$ ;
if $k = 2$	printing will occur once every four steps if $h \leq \frac{1}{4} h_p$ , but at the normal interval if $h > \frac{1}{4} h_p$ ; etc.
if $k = 38$	printing will take place only at the normal interval $h_p$ .

## 5.5. Initial conditions

The initial conditions for the problem, that is the values of  $y_i$  ( $i = 1, \dots, n$ ) must be stored in the last  $n$  locations of a block with address  $A$ , so that  $y_1$  is in location  $A.7$ ,  $y_2$  in  $A.6$ ,  $y_3$  in  $A.5$  etc.  $A$  is specified in P.P.13.

There is a restriction on the initial value of  $y_1$  (that is, effectively on the starting value of  $x$ ).  $2^k y_1$  ( $\neq x$ ) must either be zero or an exact multiple of the initial interval  $h_I = 2^{-\alpha_I}$ . If this restriction is not obeyed, there will be a loop stop in U1.1.

## 5.6. Accuracy Parameter $m$

The parameter  $m$ , which must be a positive integer or fraction, controls the interval of integration and hence the accuracy with which the computation is carried out. The interval  $h$  is chosen such that  $m/32 < |r|_{\max} < m$ , where  $|r|_{\max}$  is the modulus of the truncation error, in one step, of the variable for which  $|r|$  is greatest.

$m$  must be appreciably greater than the rounding errors in the auxiliary. If  $m$  is too small, the computation will stick as the interval  $\Delta y_1$  becomes zero.



### 5.7. Storage of Parameters

To examine or change the initial conditions when the programme is in the computer, it may be useful to note that

	$m$ is stored in 1+.7 of R 411
$(\alpha_I, 0)$	in 1+.6
$(\alpha_p, 0)$	in 1+.5 and 10+.6
$(\alpha_m, 0)$	in 1+.4
$(k, 0)$	in 10+.5

### 6. Truncation Errors

The truncation error at the end of one step is of the order of  $h^5$ . Provided that non-linearities in the differential equations are not pronounced during one interval of integration, the truncation error in  $y_i$  at the end of one step is approximately

$$\epsilon_i = - \frac{1}{720} h^5 \frac{d^5 y_i}{dx^5}$$

where the derivative is evaluated at the beginning of the step.

The relationship between the overall errors in the variables after many steps of integration and the accuracy parameter  $m$  depends on the particular equations being solved. If the solution is damped and non-oscillatory, the error may amount to a few times  $m$  in the worst case. If the solution is oscillatory, but not divergent, then errors in the amplitudes (or envelope functions) and in the frequencies may both amount to a few times  $m$ . The behaviour of the error when the solution is divergent has not yet been studied, but it is possible for the error to be much greater in this case.

### 7. Comparison with the Runge-Kutta Process

The stability of an integration process can be measured by the stability boundary. For many processes this is roughly a semi-circle in the left half-plane with its centre at the origin. For the Runge-Kutta process the radius of this semi-circle is about 2.8, so that the computation will only be stable if  $h < 2.8 |\lambda|^{-1}$  for every latent root  $\lambda$  of the matrix of coefficients (see R.H. Merson, 1954). For the Kutta-Merson process the radius of stability is about 3.3, so that this process is slightly more stable than the Runge-Kutta process.

Since the elementary truncation error of the Runge-Kutta process is approximately  $\frac{1}{120} h^5 \frac{d^5 y_i}{dx^5}$  it follows that the interval used with this process must be  $\frac{1}{\sqrt[5]{6}} = \frac{1}{1.43}$  times that used with the Kutta-Merson process to give the same accuracy. However, the latter requires five auxiliary computations per step compared with four in the former case, so that for straightforward computation with fixed interval, the Kutta-Merson process is only  $1.43 \times .8 = 1.14$  times as fast. The advantage claimed for the present programme is in the case where the interval is being controlled, for with the Runge-Kutta process much extra computation has to be done to determine the elementary truncation error.

**List of References**

1. Merson, R.H.      An operational method for the study of integration processes.  
Proceedings of a Symposium on Data Processing at Weapons  
Research Est., Salisbury, South Australia, June 1957  
(Session I, programming and mathematics).
2. Gray, H.J.        Numerical methods in digital real-time simulation.  
Quart. App. Math. Vol.XII. July, 1954, pp. 133-140.
3. Gurk, H.M.        The use of stability charts in the synthesis of numerical  
quadrature formulae.  
Quart. App. Math. Vol.XIII. March, 1955. pp. 73-78.
4. Merson, R.H.      The stability of the Runge-Kutta method of solution of linear  
differential equations.  
R.A.E. Tech. Note G.W.320. June, 1954.

**Author:**    Mr. R.H. Merson of the Royal Aircraft Establishment.

**CROWN COPYRIGHT RESERVED**

*This document is distributed with the kind permission of  
the Minister of Aviation by Ferranti Ltd. as part of its  
services to users of Ferranti Pegasus Computers.*

*The document must not be reproduced in whole or in part  
without the permission of the Minister of Aviation.*

Ferranti Ltd.,  
London Computer Centre,  
68, Newman Street,  
LONDON, W.1.

*Issue 1*  
8th November, 1960  
R.H.M.            M.J.M.

END OF VOLUME 1