TRANSCODE

A SYSTEM OF CODING FOR THE FERRANTI MARK I COMPUTER

BY

J.N.P. HUME

B.H. WORSLEY

COMPUTATION CENTRE

UNIVERSITY OF TORONTO

CANADA

FERUT LIBRARY ROUTINE

Class Composite **Date** 1 Oct. 1954 **Name** TRANSCODE

Purpose TRANSCODE is a comprehensive system of routines which enables a Ferranti Mark I Computer to utilize a three-address, mnemonic code of instructions operating on floating-decimal numbers. Tape controls are provided for facilitating organization of the programme and input of both data and instructions.

TRANSCODE is intended to provide a system of coding which is simple to learn and to apply, and which requires only a cursory knowledge of the specifications of the computer. This system will first be described, assuming such a background.

An appendix is also included for the benefit of programmers familiar with the Ferranti machine code and the Toronto system of Input and Organization. However, a study of this appendix is not necessary for learning to operate in the TRANSCODE manner.

TRANSCODE is expected to be particularly useful in solving single-shot problems, and in doing exploratory work in production problems for which scaling presents unusual difficulties. It is believed that the main loss of speed in working with TRANSCODE rather than in the machine code is only that inherent in the use of a floating-point system of notation.

Source J.N.P.H. and B.H.W. (Toronto)

Operational instructions. Thus a "programme" is written in the form

Instruction number	Instruction	Three-Address Operand		
001	LOOP	012.0	000.6	000.0
002	MULT	X12.6	Y12.6	Z12.6
		and generally		
00n	ABCD	$\alpha.B_\alpha$	$\beta.B_\beta$	$\gamma.B_\gamma$

where ABCD is a 4-letter word indicating what the instruction does,

α, β, γ are 3-character entities indicating either an address or a decimal integer,

B_α, B_β, B_γ are single decimal digits.

The code of operational instructions is given on pages 5 and 6.

Numbers

TRANSCODE deals with numbers in floating decimal form. The general form is

$$\pm\ a.bb\ \ldots b \times 10^{\pm n}$$

where a = 1,2,3 -- 9; b = 0,1,2--9 and n = 0,1,2, -- . All numbers entering into a calculation should be divided into three classes as follows:

1. Data, which must be grouped into sets called pages. Two types of pages, X and Y, are available simultaneously in the electronic store, although many such pages may be stored initially on the drum. The contents of each X or Y page may also be transferred to or from the drum at any stage of the calculation.

2. Constants, only one such set per programme, which are input initially and separately from the data, and are distributed automatically to be available as required by the programme.

3. Intermediate Values, which may temporarily be placed on a Z page in the electronic store. The Z page can thus be thought of as temporary working space, and may not be transferred to and from the drum.

Hence all numbers can be called upon by the operational instructions according to their addresses, α, β or γ where these are of the form:

X01	Y01	C01	Z01
X02	Y02	C02	Z02
			Z13
X21	Y21	C21	

X,Y and Z addresses can be B-modified in the manner described on page 7. They can be filled as the result of obeying an operational instruction. C addresses cannot be B-modified, and may not be filled as the result of obeying an operational instruction.

Any given calculation proceeds by the execution of its programme. However, it is necessary to arrange for the input and organization of the programme, data, and associated constants. In TRANSCODE, this is greatly facilitated by the provision of a set of Tape Controls which operate from the input tape, and which should now be referred to. See page 4.

All pages of data are read into the electronic store by means of NUMB, and must immediately be stored on the magnetic drum by means of the DRUM control. X and Y pages of data are indistinguishable during input. The necessary set of constants is read in by means of CNST, and automatically stored.

The set of operational instructions constituting the programme is input by means of INST. Only one such set is permissible, and must be punched without interruption. Spaces may precede any of the Tape Controls, and a STOP (I//G) inserted whenever it is desired to stop the tape from being read-in.

For lengthy programmes it may be convenient to break the set of instructions down into sub-sections. Each of these may be written separately in TRANSCODE, and prepared for further use as described in the Section under "Write-Tape Procedure". Library tapes are also provided for the more commonly occurring processes such as evaluating the simple functions, integrating differential equations, etc. A list of all functions to be used in a programme should be made out, and these numbered consecutively beginning at .001. Tapes for these may then be inserted, in any sequence, directly into the complete problem tape, provided each is preceded by $\left\{\begin{array}{l}\text{Spaces}\\ \text{FNTN 00n}\\ \text{Spaces}\end{array}\right\}$ and read in

before ENTR. A stop (FF/G) is provided at the end of each FNTN tape so that it is not necessary to punch a STOP. These functions are called into a calculation by means of FNTN used as an operational instruction.

INST oop	To read in the programme. INST 00p should be followed directly by p operational instructions, punched without instruction number or decimal pbint, and terminated by QUIT. p = 001, 002, 003 ---.
CNST abb---b±n± abb---b±n± ⋮ abb---b±n±	To read in the set of constants. Assimilation by the programme is automatic.
NUMB abb---b±n± ⋮ abb---b±n±	To read in a page of at most 21 numbers to the electronic store. Should be followed by a DRUM order.
DRUM 00m	To copy a set of numbers in the electronic store to the magnetic drum position m = 001, 002 ---. Drum positions are divided into two sets. See p.13, A.8.
FNTN 00∝	To copy a library function tape, or the write-taped instructions for a sub-section of the programme, into function location ∝ in the magnetic store. ∝ = 001, 002 ---.
STOP	To stop the tape from being read in (I//G).
ENTR	To initiate translation of the instructions, read in by means of the above tape controls, into the form required by the machine, and to start the actual calculation. A stop (ØØ/L) separates the translation from the calculation.
REEN	To re-start the operation manually just after the ØØ/L stop of ENTR. This assumes that the entire process up to this point has once taken place successfully, and that no one else has used the computer in the meantime.
Blank tape	No effect. Can precede any tape control.
KOPY	Initiates the output of all translated instructions into a form suitable for re-input. See pages 13 and appendix. Read in FNTN 000 tape to KOPY.

General Form	ABCD	$\alpha.B_\alpha$	$\beta.B_\beta$	$\gamma.B_\gamma$

ADDN $[\alpha.B_\alpha] + [\beta.B_\beta] \rightarrow \gamma.B_\gamma$	α,β = X, Y, Z or C addresses. γ = X, Y or Z address. B_α, B_β, B_γ are B-line mods. (C address can't be B-mod) * $\alpha.B_\alpha$ and $\beta.B_\beta$ remain unaltered.
SUBT $[\alpha.B_\alpha] - [\beta.B_\beta] \rightarrow \gamma.B_\gamma$ MULT $[\alpha.B_\alpha] \times [\beta.B_\beta] \rightarrow \gamma.B_\gamma$ DIVD $[\alpha.B_\alpha] \div [\beta.B_\beta] \rightarrow \gamma.B_\gamma$ 1/2 QRT $\sqrt{[\alpha.B_\alpha]} \rightarrow \gamma.B_\gamma$ KOMP $\lvert[\alpha.B_\alpha]\rvert - [\beta.B_\beta] \rightarrow \gamma.B_\gamma$	Addresses as for ADDN. 1/2 QRT assume $[\alpha.B_\alpha] \geq$ zero, $N\frac{1}{2}/G$ stop if $[\alpha.B_\alpha]$ negative.
ZERO $\alpha.B_\alpha$	Place zero in storage location α of the XY or Z type, B-mod.
OVER $[\alpha.B_\alpha] \rightarrow \gamma.B_\gamma$	α = X, Y, Z or C address. γ = X, Y or Z address. B_α, B_γ are B - line mods. (C address can't be B-mod) $[\alpha.B_\alpha]$ remains unaltered.
LOOP $\alpha.0$ 0.B 0.0	Prepare to cycle through the subsequent set of instructions α times, using B-line B to alter any X, Y or Z addresses modified by this B line. The set of instructions should be terminated by a B-conditional TRNS order involving the same B-line. See p. 7.
TRNS $\alpha.0$ 0.0 0.0	Jump (i.e. transfer control) unconditionally to instruction number α. (α = 001, 002, ---)
TRNS $\alpha.0$ 0.B 0.0	To be used in connection with the LOOP instruction, Subt's. 3 and tests. See page 7 and Appendix.
TRNS $\alpha.0$ 0.0 $\gamma.0$	Jump to instruction number α if $[\gamma] \geq 0$, otherwise go on to next instruction. γ = X, Y or Z address. $[\gamma]$ remains unchanged.

* $[\alpha]$ is used to designate the number stored in address α.

READ	α.0	0.B	γ.0	Copy the contents of DRUM position α, modified by B-line B, to electronic page position γ. α = 001, 002, . . . 00n. γ = X00 or Y00. See Page 7.
WRTE	α.0	0.B	γ.0	Copy the contents of electronic page position γ to DRUM position α. α, β, and γ as for READ. See page 7.
HALT	0.0	0.0	0.0	Insert a stop. (///G) into the programme.
VOID	0.0	0.0	0.0	Insert a dummy instruction into the programme.
COKE	0.0	0.0	0.0	Refresh material permanently required in the electronic store. Must be inserted after every 5 or 10 minutes of calculation.
QUIT	0.0	0.0	0.0	Make final preparations for obeying the programme. Must be the last instruction in any set, although not last on the tape.
PRNT	α.B_α	β.0	γ.0	Print and/or punch floating decimal numbers. α numbers, selected consecutively. B_α numbers per paper line. β digits per amplitude. ($\beta \leq 10$, and not rounded-off). γ-address of first number (X,Y or Z). Two digits per exponent. One space after amplitude, sign preceding. Two spaces after exponent, sign preceding. Extra line-feed at end. Assumes $-75 <$ exponent ≤ 99, otherwise incorrect number output or closed loop occurs. At most 64 digits, signs, etc. in a paper line. Alters B2 and B3.
BSET	0.B	β.0	0.0	Put integer β in B-line B.
	0.B	0.0	γ.0	Put $[\gamma]$ in B-line B. γ = X,Y,Z or C address. See page 16.
JOTB	0.B	0.0	γ.0	Plant [B-line B] in γ. γ = X, Y or Z address, see page 16.
INCB	0.B	β.0	0.0	Add the integer β to [B-line B].
NEGB	0.B	β.0	0.0	Subtract integer β from [B-line B].
	0.B	0.0	γ.0	Subtract $[\gamma]$ from B-line B. γ = X,Y,Z or C address. See page 16.
FNTN	α.0	β.B_β	γ.B_γ	Place the function numbered α, of $[\beta]$, in γ. α = 001, 002, 003, . . as assigned for the programme. β=X,Y,Z or C address; γ=X,Y or Z address. β,γ(B mod) Can be used for functions with more than one argument or answer. Then set β = 000, γ = Z14, and consult separate description.

In order to cut down on the total number of instructions required to carry out a repetitive process, the B-modification facility should be used.

The LOOP instruction is intended to be used in conjunction with the B-dependent, TRNS instruction to cycle through an intermediate set of instructions a given number of times. Any X, Y or Z addresses occurring in the intermediate instructions are then modified progressively. The same B line should be used consistently, and the X, Y or Z addresses of the last set of numbers to be operated on should be written into the programme.

For example: Place zero in each of the locations X01, X02 --- X21, beginning with X01.

```
001 | LOOP  021.0  000.3  000.0 |
002 | ZERO  X21.3  000.0  000.0 |
003 | TRNS  002.0  000.3  000.0 |
```

These 3 instructions can therefore be used in place of the following 21:

```
| ZERO  X01.0  000.0  000.0 |
| ZERO  X02.0  000.0  000.0 |
|         :      :          |
| ZERO  X21.0  000.0  000.0 |
```

Instructions referring to drum storage locations can also be modified progressively, but this must be done by direct setting and modification of some B-line.

For example: Write the X page on drum position 001 + [B4], where B4 is to contain n.

```
| BSET  000.4  00n.0  000.0 |
| WRTE  001.0  000.4  X00.0 |
```

If n is to take on the values 0, 1, 2 -- 15 in succession, these instructions should be imbedded into a loop as follows:

```
001 | BSET  000.4  000.0  000.0 |
002 | LOOP  016.0  000.3  000.0 |
003 | WRTE  001.0  000.4  X00.0 |
004 | INCB  000.4  001.0  000.0 |
005 | TRNS  003.0  000.3  000.0 |
```

Note In terms of the actual machine, when an instruction is B-modified, the contents of the B-line mentioned are added to the instruction just before it is obeyed, although the instruction itself remains unaltered in the electronic store.

In the development stages of testing a programme on the computer, it may be found useful to print out intermediate values occurring in the calculation which would not be desired in the production runs. This can readily be achieved by inserting extra PRNT instructions in the programme, later altering these to VOID instructions by hand-punching and re-perforating the test programme tapes.

Speed

Reading-in of INST'ns., 5 or 6 instructions per Sec.

* Reading-in of NUMB or CNST, 27 msec.[w] per decimal digit. Translation into real code, 4 or 5 instructions per Sec.

	QUIT	1 to 2 Sec.
**	ADDN	73 to 89 msec.
	SUBT	73 to 89 msec.
	MULT	63 msec.
	DIVN	(126 + 15n) msec., n = 1,2,...6.
	1/2 QRT	(155 + 26n) msec., n = 1,2,...6.
	READ	75 msec.
	WRTE	108 msec.
	TRNS	50 msec. (4 msec. if back to same track)
	All B-line instructions	.96 m Sec.
	ZERO	5 m Sec.
	COKE	100 m Sec.
*	Printing	$6\frac{2}{3}$ characters /Sec.
*	Punching	24 " "

---

* There is a slight delay in the conversion of numbers with decimal exponents larger than about 10, this being proportional to the magnitude of the exponent. For exponents of order 1000, the delay is about 2 Sec., for 2000 it is about 5 Sec.

** The results of all numerical operations are automatically normalized and tested for being within range. If the result is too large, a dynamic stop occurs (DS/0); if too small, it is replaced by zero. If all significant figures are lost, a stop occurs (SF/G).

[w] 1 msec. = 1/1000 sec.

$+\frac{1}{4} \leq |\text{normalized number's amplitude}| < +\frac{1}{2}$.

1. The generalized floating decimal number is punched as abb...b±n±.
If n = 0, it is sufficient to punch
abb...b±+ .
Whilst "a" must be ≠ 0 and therefore punched, only the significant digits, b, need be punched. Any number of b-digits may be punched, but only the first 10 will be meaningful.

2. To aid in the preparation of numerical tapes, two facilities are available.

(i) Any number of non-significant b digits (Say b = 0's) may be punched after the significant ones to permit splicing of tapes in this area.
(ii) If an error is detected before the final sign has been punched for any one number, a ∯ may be punched and the number repunched correctly. The ∯ causes the part of the number currently being read-in to be ignored.

3. The number zero must be arbitrarily defined in floating form. It should be punched as 1 + 100 -.

4. All the tape control words and all the operational instruction words consist of 4 letters. However, only the first of these letters is interpreted, so that spelling of the word is immaterial. It may therefore be found convenient to punch SUBT, for example, as SSSS, or VOID as VVVV and so on. Both punching and tape-recognition are then simplified.

5. Characters in the addresses of the instructions which are prescribed as the decimal digit "zero" should be punched as "zero" or "stroke" on the keyboard. Care should be taken not to use the key for "capital letter O". There are times when these zeros need not necessarily be zeros, but this should never be assumed except in connection with the instructions VOID, HALT, QUIT and COKE. Hence it is easy to alter PRNT to VOID without hand-punching the entire tape.

6. Tapes should be punched once on each keyboard and compared for accuracy.

7. The tape controls ENTR and REEN are generally punched at the end of any complete TRANSCODE tape. If these are punched as EEEE //// RRRR, an easily recognizable pattern results which helps one to guard against placing the wrong end of the tape in the FERUT reader.

8. The symbols ∯ and ⁂ can be used interchangeably. They each represent five holes across on the tape.

1. Form the scalar product of the Vectors

$a_1 \times 10^{-b_1}$, $a_2 \times 10^{-b_2}$, ---, $a_{20} \times 10^{-b_{20}}$ and

$-c_1 \times 10^{d_1}$, $-c_2 \times 10^{d_2}$, --, $-c_{20} \times 10^{d_{20}}$ and add it to the constant $ex10^{f}$.

A tape to solve this problem should be punched as follows:

```
Spaces
CNST   e + f + "
Spaces
NUMB   a1 + b2 -
       a2 + b2 -

       a20 + b20 - "
Spaces
DRUM   001
Spaces
NUMB   c1 - d1 +
       c2 - d2 +

       c20 - d20 + "
Spaces
DRUM   002
Spaces
INST 010
READ    0020    0000    Y000
READ    0010    0000    X000
LOOP    0200    0006    0000
OVER    C010    0000    Z010
MULT    X206    Y206    Z020
ADDN    Z020    Z010    Z010
TRNS    0050    0006    0000
PRNT    0011    0100    Z010
HALT    0000    0000    0000
QUIT    0000    0000    0000
Spaces
ENTR
Spaces
REEN
Spaces
```

Spaces should not be punched except where expressly indicated, or between the a_i and the + sign or the c_i and the - sign.

n!
decimal places, 4 entries to a line, and in blocks of 4 lines.

A programme to do this could be written as follows:

```
      INST   011
001   OVER   C01.0     -       Z01.0
002   OVER   Z01.0     -       Z02.0
003   OVER   Z01.0     -       Z03.0
004   LOOP   016.0     000.5   -
005   DIVD   Z02.0     Z03.0   Z02.0
006   OVER   Z02.0     -       X16.5
007   ADDN   Z03.0     Z01.0   Z03.0
008   TRNS   006.0     000.5   -
009   PRNT   016.4     005.0   X01.0
010   TRNS   005.0     -       -
011   QUIT   -         -       -

      CNST   1 + + "
      STOP
      ENTR
```

The method used above is quite direct. Answers are stored in sets of 16 on page X, and printed out when each set is complete. Z01 is used to retain +1. At any instant, Z02 contains $\frac{1}{n!}$ and Z03 contains $(n + 1)$.

$\frac{1}{(n+1)!}$ is then calculated as $\left(\frac{1}{n!}\right) \div (n + 1)$.

Read-in Procedure:

All tape controls are interpreted on the warning character principle, so that the tape is input by a manual entry in the usual way with H = ZA@/. Set write-current on, print, punch, print-and-punch switch as desired, /G Stop as desired, /L Stop off. All working-track switches should be open.

ENTR should follow the input of all data and operational instructions, and hence all INST, CNST, NUMB, DRUM or FNTN controls on the tape. STOP leads to an I//G stop. The reading-in of each FNTN is terminated by an FF/G Stop. ENTR initiates the translation of the information thus far read in, into the form required by the machine. The translation of the QUIT instruction completes the internal preparation of the material, and this step is terminated by Stop (//L). A prepulse then initiates the actual calculation, leading automatically to any programmed output. The programmed HALT leads to a ///G Stop.

All magnetic transfers of data are checked, any failure giving rise to a "motor-boat" hoot Stop.

Restart Procedures:

1. Before the programme be suspected at fault, a test or development case should always be run twice, completely, from the beginning, and the two results agree exactly.

2. A production run should always be preceded by a test run for which the answers are known, to ensure that input and translation of the programme has been performed correctly by the machine.

3. Should the input and translation be known to have taken place correctly, re-start of the calculation is possible by manual entry at REEN.

4. Should the input be known to have taken place correctly, re-start of the translation and calculation is possible by manual entry at ENTR.

5. Should the reading-in process break down, re-start is possible at any of the tape controls except DRUM. DRUM is excepted only because it assumes data present in the electronic store.

6. Should additional or alternate data be required to be read-in after the calculation has reached a certain stage, this can be achieved by direct input of the new data tape. The calculation may then be re-started by the REEN control, provided the machine has not been used for another problem in the intervening time, and provided the same CNST's pertain.

The order KOPY initiates the output punching of all translated instructions in a form suitable for re-input by means of the FNTN and REEN orders. The exact details of this output are given in the appendix. This facility is intended to be used in either of two ways:

1. To compose sub-routines: If a programme contains say more than 30 or 40 operational instructions, it may be found convenient to write portions of it in the form of sub-sections, which would be called in by the main part of the programme as FNTN's, with suitably assigned numbers, n, just as if they were library FNTN's. There are the restrictions that each sub-section prepared in this way should not be more than one FNTN in length, and should terminate with the last instruction to be obeyed before exit. The tape thus output should have the initial FNTN 101* and the final RRRR chopped off, and be incorporated into the main problem tape preceded by FNTN oon. (See appendix page iv.)

2. To prepare final copies of programmed INST's: In the event of programmes to be used repeatedly for production runs over a long period of time, it may be worthwhile to prepare a "write-taped" version of the INST part of the tape for future input. This should be attached, exactly as it comes from the machine, to the end of all sections of the tape containing data and FNTN input sequences, and replacing all INST, CNST and ENTR sequences. It has the advantage of eliminating the translation process and providing input check-sum facilities. It has the disadvantage that the write-tape is, on the average, 3 times as long as the original tape.

---

* Provision is made in the input section of TRANSCODE to store FNTN's numbered 101, 102, 103, ... in the consecutive drum locations normally used to house the translated instructions. The true FNTN's should be numbered 001, 002, ... and are stored on a separate set of drum locations. Also, one of the duties of the QUIT instruction is to arrange for the machine order NS/P to be placed immediately after the last machine order in the programme just translated.

A stop (DS/Q) terminates the entire writetape procedure. Care should be taken that this point is reached to ensure that all the translated instructions have been output.

Restrictions on a TRANSCODE Programme

A. Imposed by the Coding:

1. At most 21 CNST's.
2. At most 21 NUMB's stored in each DRUM location, and therefore to each X or Y page.
3. At most 13 Z-type working locations.
4. Only B-lines 2,3,4,5, and 6 available.
   If no B-line modification is required, B0 is automatically applied, and must therefore be clear.
   Note that PRNT alters the contents of B2 and B3.
5. At most 64 forward transfers of control.

6. Numbers input should lie between 10^{-1000} and 10^{+1000}.
   For numbers outside this range, read-in is slow or impossible.
   Numbers generated in the course of a calculation must lie between 2^{-215} and 2^{+215}, which is equal approximately to $10^{\pm 10,000}$. If a calculated number should be too small, it is automatically replaced by "zero"; if too large, a stop occurs (DS/0).
   Numbers output must lie between 10^{-75} and 10^{+100}.
   For numbers outside this range, either an incorrect answer will be output or a closed loop will occur.

7. The first instruction obeyed must have the instruction number 001. However, this can always be an unconditional TRNS.

8. Available DRUM locations are divided equally into two sets, called the lower half and the upper half of the range. Any set of DRUM locations selected by B-modification techniques must lie entirely within one of these two sets.

9. At most 100 FNTN locations per programme, with addresses 001, 002, 003, ... 100.

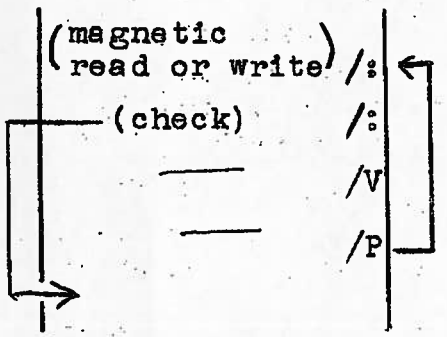
* B. Imposed by the Normal Track Assignments:

1. At most 256 INST's.
   For more than 100 INST's, however, a check should be made to see that the translated instructions do not over-run the 16 tracks assigned. This can be done by watching S3 during translation. See Appendix,p.iv.

2. At most 15 FNTN's, with addresses 001, 002, ... 015.

3. At most 32 DRUM locations, allowing 672 NUMB.
   These are divided into two sets in accordance with restriction A-8 above. The lower half set is numbered 001, 002, ... 016, and the upper half 017, 018, ... 032.

.11..	.....	.....	.1.11	- I//G	- STOP punched on tape.
.11.	1.11.	.....	.1.11	- FF/G	- at end of reading in a FNTN.
...1.	1.1..	.....	111.1	- DS/Q	- If in N/, a neg. non-warning character - If in FA, a pos. non-warning character } on input tape
.1111	11111	.....	.1.11	- $$/L	- at end of translation after ENTR.
..11.	...1.	.....	.1.11	- $N^{\frac{1}{2}}$/G	- if argument for $2^{\frac{1}{2}}$RT negative.
....	.....	.....	.1.11	- ///G	- HALT in executing a programme.
..1..	1.11.	.....	.1.11	- SF/G	- if normalizing factor $\geq 2^{39}$.
...1.	1.1..	.....	...11	- DS/O	- if a calculated number $> 2^{2^{16}}$.
...1.	1.1..	.....	111.1	- DS/Q	- end of write-taping via KOPY.
.1.1.	1.1..	.....	.1..1	- JS/L	- in RCS/T



All transfers of numerical data are checked in this way, leading to a "motor-boat" hoot for a consistent magnetic transfer failure. Transfers of instructions are not checked.

Steady Hoot Stop - Check-sum failure on reading in FNTN.
Closed loop in machine operation - this could be due to the programme, or because a number outside the permitted ranges is being input or output.

Note 1. /L and /G are stops which correspond to switches on the console, and are only operative when the corresponding switch is "on". They can each be surmounted by a prepulse, when the operator is satisfied that the progress of the calculation should proceed.

Note 2. DS/Q and DS/O are dynamic stops, which cannot be surmounted. They should never occur in a correctly designed programme or correctly prepared tape, except at the end of a KOPY run.

Floating Decimal Representation

The machine is constructed to operate on numbers in fixed-point, binary form. Hence, in order that numbers punched in floating-point, decimal form may be read in, a conversion must take place. Within the machine, each floating-point, decimal number is, in fact, represented in its equivalent floating-point binary form, one long line of machine storage being used to represent the amplitude, and one short line to represent twice the binary exponent. (The factor of two in the exponent is merely for convenience in coding).

Integers as well as fractions punched in floating-decimal form may be read in by means of the tape controls NUMB or CNST. However, owing to the conversion, they are not represented with absolute accuracy when finally in the machine. Integers can be in error by an amount lying between 10^{-9} and 10^{-12}. Arithmetical operations involving floating decimal integers and the KOMP instruction may not therefore give the desired results, unless compensation is made for these discrepancies, ie. positive integers should be corrected by adding a small quantity to them, negative integers by subtracting a small quantity from them.

B-Line Representation

Numbers in B-lines are not in floating-point form, but in the fixed form of the computer. Thus integers can be represented in B-lines with absolute accuracy.

Integers are normally intended to be picked up by B-lines via the operand β of the BSET, INCB or NEGB instructions. However, it may be desirable to set a B-line with a variable integer in the manner of a pre-set parameter, rather than have it imbedded in the programme. This can be achieved by planting it in the exponent line of a floating-decimal address (X, Y, Z or C type). In this case, it must be read in as an equivalemt floating-decimal number, using NUMB or CNST. It may then be picked up via the operand γ of the BSET or NEGB instructions. Such a representation is still exact, provided that authorized floating equivalents are used. Some tested equivalents are given on page 17. The following examples should be studied carefully before using these techniques.

Example 1. Pick up the integer 100 in B-line 3, where 100 is preset as a CNST.

```
CNST3+14+"
INST00x
BSET 0003 0000 C010 etc.
```

Example 2. Place in Z01 the sum of the n numbers in X01, X02 ... X0n, where n may be preset.

```
CNST(floating decimal equivalent of (2n-2))"
INST 009
BSET 0006 0000 C010
BSET 0005 0000 0000
ZERO Z010 0000 0000
→ADDN Z010 X015 Z010
NEGB 0005 0030 0000
INCB 0006 0010 0000
```

Notes
B6 used as counter.
B5 used to modify addresse
Floating numbers stored
in reverse sequence

To input the integer n into the exponent line of a floating number, punch as follows, with NUMB or CNST:

n	Punch a
0	3+1-
2	1+ +
4	2+ +
6	4+ +
8	8+ +
10	1+1+
12	2+1+
14	4+1+
16	7+1+
18	2+2+
20	3+2+
22	6+2+
24	2+3+
26	3+3+
28	5+3+
30	9+3+
32	2+4+
34	4+4+
36	7+4+
38	2+5+
40	3+5+
42	6+5+
44	2+6+
46	3+6+
48	5+6+
50	9+6+
60	3+8+
70	9+9+
80	3+11+
90	9+12+
100	3+14+

Note: Only even integers may be input in this manner. They go into the correct location for pick-up by the BSET γ and NEGB γ instructions.

In general, $a = \frac{1}{4} \text{antilog}_{10}\left[n/(2 \log_2 10) \right]$,

or, approximately, a is any integer satisfying

$$\frac{10^{0.151n}}{4} < \text{integer} < \frac{10^{0.151n}}{2} .$$

The TRANSCODE system of routines consists of several parts:

I One section, involving decimal-binary conversion and arithmetic operations on floating numbers, could actually be used independently of the remainder of the TRANSCODE system. This comprises the following routines, and full details are given under these titles elsewhere in the FERUT Library:- TC: DECINPUT, TC: ARITHMETIC and TC: DECOUT. However, these routines are automatically utilized in their TRANSCODE setting.

II Another section consists of a pair of TC: INPUT routines, which interpret the tape controls, transferring the instructions and numbers, with some modification, to predetermined locations within the machine. CNST and NUMB controls automatically employ TC: DECINPUT.

III A large section of the TRANSCODE system consits of routines which are called into play by ENTR. They translate the instructions as stored by TC: INPUT into the real machine code, and make all other necessary preparations for the final execution of the programme under the automatic control of the machine. It is actually in the translation of the QUIT instruction that all anticipated addresses in control transfers are inserted, and all other final preparations are made for switching over from the translating to the executing stage of the work. A ~~$\beta\beta/L$~~ stop separates the two stages.

The FNTN input facility resorts to INPUT/T as a subroutine, and it is assumed that this is stored on tracks 4 and 5. All other necessary routines are peculiar to TRANSCODE.

The reader may be interested to know that, in the final analysis, the standard Toronto Scheme for changing routines and allocating data is employed. During the execution of the calculation, the electronic store is utilized as follows:-

S0 and S1 contain the current problem routines, and these assume that the FLOATER routines of TC: ARITHMETIC are on S_4 and S_5. PERM/T is available on S_2, and column I of S_3 is used for the RCS/T link-list as well as for working space. The remainder of S_3 is used for the Z-type numerical working space, S_6 for the X-type, and S_7 for the Y-type pages of numerical data.

The TRANSCODE routines occupy 17 tracks. During its operation, TRANSCODE also requires 4 sets of consecutive working tracks. The working tracks are designated as follows:

t_1 as the first of n_1 consecutive tracks
t_2 " " n_2 "
t_3 " " n_3 "
t_4 " " n_4 "

where (t_1) L, (t_1+1)L, --- (t_1+n_1-1)L store the modified TC instructions to be translated,

t_1R stores the CNST's,

(t_1+1)R stores the untranslated addresses,

(t_1+2)R stores the final INDEX page,

(t_1+3)R, (t_1+4)R, -- (t_1+n_1-1)R store the TC directory of instruction numbers,

t_2, (t_2+1), --- (t_2+n_2-1) store the translated inst'ns. and also the FNTN101, FNTN102, ...

(t_3)L, (t_3+1)L, --- $(t_3+\frac{n_3}{2}-1)$L store DRUM consignments 1,2,3 --- $\frac{n_3}{2}$,

(t_3)R, (t_3+1)R, --- $(t_3+\frac{n_3}{2}-1)$R store DRUM consignments $(\frac{n_3}{2}+1)$, --- n_3.

t_4 stores FNTN 000, which is a TC routine required if KOPY is to be used.

t_4+1, (t_4+2), ---(t_4+n_4-1) store FNTN consignments 1,2,3 ---(n_4-1).

Normal Track Assignments

Normally, all the TC routines except FNTN 000 will be assumed already stored on tracks TA to ØA inclusive. A Toronto directory tape is preserved by the librarian for this purpose. If KOPY is required, the FNTN 000 tape should also be read in, and it will automatically go to track t_4.

Also, it will normally be assumed that

t_1 = T/ , n_1 = 16,

t_2 = /E , n_2 = 16,

t_3 = TE , n_3 = 16,

t_4 = T@ , n_4 = 16.

These working tracks are all specified in a page of INDICES, which is treated like one of the TC routines.

Available during TC: INPUT on S5,
" " TC: Translation on S2.

H:		S: J
////	/	↓
////	E	
////	@	$n_3 \div 2$
K///	A	♯R //
$\frac{1}{2}$A//	:	////
/@//	S	@J //
(t_1)/N	I	TJ //
(t_2)R/	U	(t_4)@/
(t_1+3)DI	$\frac{1}{2}$	AJ //
(t_1+3)EI	D	EJ //
(t_3-1)//	R	AR //
(t_1+1)DC	J	LJ //
///♯	N	YJ //
(t_1+1)EC	F	ER //
////	C	ZJ //
////	K	MR //
↑	T	GJ //
	Z	KJ //
	L	OJ //
Working	W	♯D //
	H	CR //
	Y	(t_1+2)D:
Space	P	@D //
	Q	UJ //
	O	CJ //
	B	WJ //
	G	♯♯♯♯
	"	E///
	M	/J //
	X	(t_1)EC

Tape Contents of FNTN Tape ≡ KOPY output

Spaces FNTN OOn Spaces
K/½/
contents of first column as in Toronto system
K/D/ " second " " "
K/R/ " third " " "
K/J/ " fourth " " "

KLU@ (check sum for cols. 1 and 2)C/½ KLU@(Check sum for cols. 3 and 4) C/R EQS

(In write-taping,
RRRR is also punched at the end of the last FNTN,
and the FNTN's are automatically numbered as 101,
102,103 ---.)

Real Code Equivalents of TC addresses

XOn ≡ XF-3(n-1)
YOn ≡ XK-3(n-1)
ZOn ≡ XU-3(n-1)
COn ≡ XK-3(n-1) and this is marked for special consideration during translation.

NUMB and CNST read in numbers, consecutively, to S_4. Storage is backwards, beginning with $[XD]_{19}^{39}$ for amplitude, and $[\$D]_{6}^{19}$ for 2×(exponent). Translation from floating decimal to floating binary takes place.

INST, with certain modifications, are read in to consecutive tracks via S_4, 4 short lines per instruction.

A special transcode directory is built up in S_3 during translation, and stored on consecutive ½-tracks. An entry in the form

$$[(\text{track})(\text{machine address of first real instruction})]_{0}^{19}$$

is made in this directory for each TC instruction. QUIT avails itself of this directory for supplying the untranslated addresses occurring in forward TRNS instructions.

The last track to be used for the machine instructions can quickly be determined by inspecting the last entry made on S3 when the $/$/L stop is encountered. This should not exceed the last track assigned for the purpose.

OVER $\alpha.B_\alpha$ 0.0 $\gamma.B_\gamma$

```
(α)     (T+Bα)   ½
(γ)     (T+Bγ)   A
(α+1)   (T+Bα)   ½
(γ+1)   (T+Bγ)   A
```

KOMP $\alpha.B_\alpha$ $\beta.B_\beta$ $\gamma.B_\gamma$

```
(α+1)(T+Bα)      ½
E ½    T         A
(α)    (T+Bα)    ½
E:     /    M
(α)    (T+Bα)    F
/½     TA
(β+1)  (T+Bβ)    ½
: ½    T  A
(β)    (T+Bβ)    F
r-1   A ½    T A
r     (r)    Q 0
r+1   HJ / P
      U½     T ½
      (γ+1)  (T+Bγ)  A
      I ½    T ½
      (γ)    (T+Bγ)  A
```

{READ / WRTE} α.0 0.B γ.0

```
(ab)  (T+B)  ½
P I   (T+B)  B
Q I   (T+B)  B
P I   T      C
P I   T      A
P I   /      :
Q I   /      :
/ /   / V
J J   / Q
```

where $[ab]_0^{39}$ is set = (transf. and check) during translation and $[JJ]_0^{19}$ = M$$$ in FLOATER/B.

HALT ///G

VOID //T$

Real Code Equivalents of TC Instructions

ADDN $\alpha.B_\alpha$ $\beta.B_\beta$ $\gamma.B_\gamma$	$(\alpha+1)$	$(T+B_\alpha)$	$\frac{1}{2}$	
	$E\frac{1}{2}$	T	A	
	(α)	$(T+B_\alpha)$	$\frac{1}{2}$	
	$/\frac{1}{2}$	T	A	
	$(\beta+1)$	$(T+B_\beta)$	$\frac{1}{2}$	
	$:\frac{1}{2}$	T	A	
Real address = r-2	(β)	$(T+B_\beta)$	$\frac{1}{2}$	
r-1	$A\frac{1}{2}$	T	A	
r	(r)	Q	O	
	HJ	/	P	
	$U\frac{1}{2}$	T	$\frac{1}{2}$	
	$(\gamma+1)$	$(T+B_\gamma)$	A	
	$I\frac{1}{2}$	T	$\frac{1}{2}$	
	(γ)	$(T+B_\gamma)$	A	

SUBT Same as ADDN but with $[r-2] = (\beta)(T+B_\beta)N$.

MULT " " " $[r+1] =$ Y J / P .

DIVN " " " $[r+1] =$ P J / P .

$\frac{1}{2}$QRT $\alpha.B_\alpha$ 0.0 $\gamma.B_\gamma$	$(\alpha+1)$	$(T+B_\alpha)$	$\frac{1}{2}$	
	$E\frac{1}{2}$	T	A	
	(α)	$(T+B_\alpha)$	$\frac{1}{2}$	
	$/\frac{1}{2}$	T	A	
r	(r)	Q	O	
	WJ	/	P	
	$U\frac{1}{2}$	T	$\frac{1}{2}$	
	$(\gamma+1)$	$(T+B_\gamma)$	A	
	$I\frac{1}{2}$	T	$\frac{1}{2}$	
	(γ)	$(T+B_\gamma)$	A	

ZERO $\alpha.B$ 0.0 0.0	N D	T	/	where $[ND]_0^{39} =$ //// ///$\frac{1}{2}$
	(α)	$(T+B)$	A	
	FD	T	/	$[CD]_0^{19} =$ ///0 Ln
	$(\alpha+1)$	$(T+B)$	A	FLOATER/A .

COKE

r	(r+3)/:	
r+1	(r+4)/:	
r+2	A: /@	
r+3	ZN //	false line for FLOATER/A
r+4	LN //	false line for FLOATER/B
r+5	(r+7)/:	
r+6	E: /@	
r+7	S//:	

	Backwards, but to same two pages	Backwards to another two pages, or forwards
TRNS α.0 0.0 0.0	r //T$\not$ r+1 E:/Q r+2 S:/Q r+3 //T$\not$ r+4 (r+5)/P r+5 ($\bar{\alpha}$) // r+6 (same track) @/	r //T$\not$ r+1 E:/Q r+2 S:/Q r+3 (r+3)$\not$0 r+4 GS/P r+5 r+6 (Link corresponding to α)
TRNS α.0 0.B 0.0	r DJ(T+B)G r+1 E:/0 r+2 S:/Q r+3 //T$\not$ r+4 (r+5) /P r+5 ($\bar{\alpha}$)// r+6 (same track) @/	r DJ(T+B)G r+1 E:/0 r+2 S:/Q r+3 (r+3)$\not$0 r+4 GS/P r+5 (Link corresponding to α)
TRNS α.0 0.0 γ.0	r (γ)T$\frac{1}{2}$ r+1 E:/M r+2 S:/Q r+3 //T$\not$ r+4 (r+5)/P r+5 ($\bar{\alpha}$)// r+6 (same track)@/	r (γ)T$\frac{1}{2}$ r+1 E:/M r+2 S:/Q r+3 (r+3)$\not$0 r+4 GS/P r+5 (Link corresponding to α)

where $[DJ]$ = A/// in FLOATER/B and
$\bar{\alpha}$ = control entry corresponding to α.
A test is made that r $\neq$ GE. If r = GE, //T$\not$ is made to precede the instructions, so as not to split the link.

LOOP α.0 0.B 0.0 (ab)(T+B)0 where $[ab]_0^{19} = 3(\alpha-1)$

Note 1. LOOP essentially sets B-line B to count α times, as well as to modify X,Y or Z addresses. The B-conditional TRNS instruction mentions B-line B by subtracting 3 from it, thus enabling it, in the next cycle, to pick up the next floating number 3 machine lines back in an electronic page.

Note 2. The two instructions

INCB	000.B	003.0	000.0
TRNS	00α.0	000.B	000.0

have the effect of a normal B-conditional control transfer.

Note 3. Arithmetical operations may be carried out on integers by using the various B-line instructions, but care must be taken not to confuse these with the arithmetic instructions, proper (ADDN, COMP, etc.)

BSET 0.B β.0 0.0 | (ab)(T+B)0 | where $[ab]_0^{--} = \beta$
0.B 0.0 γ.0 | (γ+2)(T+B)0 |

INCB 0.B β.0 0.0 (ab) (T+B)G where $[ab]_0^{19} = -\beta$

JOTB 0.B 0.0 γ.0 (γ+2)(T+B)B

NEGB 0.B β.0 0.0 | (ab) (T+B)G | where $[ab]_0^{19} = \beta$
0.B 0.0 γ.0 | (γ+2)(T+B)G |

PRNT α.B$_\alpha$ β.0 γ.0

r	(r+11)T/
r+1	PI TA
r+2	(r+10)LO
r+3	(r+9)WO
r+4	(r+4)@O
r+5	GS/P
r+6	XA//
r+7	TN//
r+8	S:/@
r+9	(α-1) //
r+10	(γ)//
r+11	- (E//(β-1)
r+12	(B$_\alpha$-1)//
r+13	UJ/J
r+14	ØD/:

false line for TC:DECOUT

where $[UJ]_0^{39}$ = ...W
and $[ØD]_0^{19}$ = ..T.

FNTN α.0 β.B$_\beta$ γ.B$_\gamma$

	(β+1)(T+B$_\beta$)$\frac{1}{2}$
	E $\frac{1}{2}$ T A
	(β) (T+B$_\beta$)$\frac{1}{2}$
r-1	/ $\frac{1}{2}$ T A
r	(r) Q O
r+1	G S / P
	Ø Ø / /
	(track)@/
	E $\frac{1}{2}$ T $\frac{1}{2}$
	(γ+1) (T+B$_\gamma$) A
	/ $\frac{1}{2}$ T $\frac{1}{2}$
	(γ) (T+B$_\gamma$) A

Note: The FNTN routine must operate in S_0 and S_1, with control entry Ø Ø, and leave the answer in $[/\frac{1}{2}]_0^{39}$, $[@\frac{1}{2}]_0^{19}$.

Note In the TRNS, PRNT and FNTN instructions a linking sequence to the RCS/T occurs, and it is important that the first short line of the link should not be placed in ØE. During translation, a test is made for this contingency, and an extra dummy instruction inserted prior to the machine orders quoted above, should it arise.

TRANSCODE LIBRARY FUNCTIONS

Routine	Page
ARCTAN	2
ARCOT	3
CARDIN	3
CARDOUT: I	4
CARDOUT: II	5
COMPLEX ROOTS	5
CONVERSION	6
COSINE	6
EIG/SYM	7
EXP	6
LINALG	9
LOG	11
MATRIX	12
RESCUE	11
RUNGEKUTTA	11
SIN	13
SPECIAL	13

LIBRARY FUNCTIONS

This supplement has been prepared to supersede Chapter 5 of the Transcode Manual, and includes details of library tapes available to Transcode users.

They are described under the following headings:-

1. Title. The library tape will be labelled with this title.

2. "Tape Control - Standard"

This means that the tape for the function, which is to be used as FNTN oof, is to be reperforated into the program exactly as it comes from the library cabinet, immediately following

FFFF oof Spaces

If the Tape Control is not standard, alternative notes are given.

3. "Program Entry - Standard"

This means that the function oof is to be called into action as required by the program by means of the instruction

FNTN oof,O aaa,A ddd,D

Where the argument is to be found in address aaa (modified by B-line A) and the function of this argument is to be placed in address ddd (modified by B-line D). Notation as on Summary Sheet, Appendix V.

If the Program Entry is not Standard, alternative notes are given.

Title TC:ARCTAN

Purpose To evaluate the principal value ($-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}$) in radians of the inverse tangent.

Tape Control Standard

Program Entry Standard

Notes 1. A separate tape is available for ARCOT.

2. If both arctan and arcot are required in the same program, use TC:ARCTAN for arctan N and evaluate arcot N as ($\frac{\pi}{2}$ - arctan ($\frac{\pi}{2}$ = 1,570,796,326,795).

3. $\text{Sin}^{-1}\ N = \tan^{-1}\ (N/\sqrt{1-N^2})$
   $\text{Cos}^{-1}\ N = \tan^{-1}\ (\sqrt{1-N^2}/N)$
   $\text{Sec}^{-1}\ N = \tan^{-1}\ (\sqrt{N^2-1})$
   $\text{Cosec}^{-1}\ N = \frac{\pi}{2} - \tan^{-1}(\sqrt{N^2-1})$.

Title TC:ARCOT

Purpose To evaluate the principal value ($0 \leq \theta \leq \pi$) in radians of the inverse cotangent.

Tape Control The Library tape is prepared for input as FNTN 005 and FNTN 006. These tape controls must not therefore be punched separately into the program tape. Also, these FNTN numbers cannot be used elsewhere in the program.

Program Entry FNTN 005.0 aaa.A ddd.D.

Error Between 1 in 10^{10} and 4 in 10^{7}.

Title TC:CARDIN

Purpose To read data from IBM punched cards.

Tape Control Standard - Requires a guide as described below.

Program Entry FNTN oof.0 000.0 Z14.0

Time

Notes 1. A guide should be added just before the EQS at the end of the library tape. This should be punched as follows:-

aaa.0 $n+d_1+d_2+\ldots,d_n+"$

where aaa.0 is the transcode address in which to place the first number from the card.

n is the number of numbers to be read per card. These are placed in sequential transcode addresses starting at aaa.0.

d_k is the number of digits in the k^{th} number. The guide must be terminated by punching " which is then followed by EQS as on the library tape.

The following restrictions apply:-

$1 \leq n \leq 20 \quad 1 \leq d_k \leq 12 \quad \Sigma d \leq 63$

2. Each entry to TC:CARDIN in effect reads one card. In fact 11 cards are read on the first entry and each 11th entry thereafter. Drum positions 054 to 064 are used by TC:CARDIN and must therefore not be used by the program.

3. The special instruction JOTB 000.0 000.0 208.0 must be included in the transcode program so that it is obeyed at some time prior to the first use of TC:CARDIN and is not obeyed thereafter.

4. Numbers are interpreted as decimal fractions and must be multiplied by a suitable power of 10 after input.

5. An "X" or "Y" punching over any digit will cause the number to be treated as negative.

6. At least 14 blank cards should be placed on top of the cards in the input hopper.

Title TC:CARDOUT:I

Purpose To prepare and store one IBM card of data for subsequent punching by TC:CARDOUT:II.

Tape Control Standard - Requires a guide as described below.

Program Entry FNTN oof.0 000.0 Z14.0

Time 200 + 35n msec.

Notes 1. A guide should be added just before the EQS at the end of the library tape. This should be punched as follows:-

aaa.0 n-d_1+ d_2+ d_3+ . . . . d_n+"

where aaa.0 is the transcode address of the first number to appear on the card.

n is the number of numbers to be punched per card. These are taken from sequential transcode addresses starting at aaa.0.

d_k is the number of digits in the k^{th} number.

The guide must be terminated by punching " which is then followed by EQS as on the library tape.

If no guide is used, the routine will prepare 7 numbers each to 9 digits starting at Z01.

The following restrictions apply:-

$1 \leq n \leq 20$ $1 \leq d_k \leq 12$ $\Sigma d \leq 63$ $n+\Sigma d \leq 72$

2. Each entry to TC:CARDOUT:I prepares one card for output. The prepared data are stored in Drum locations 033 to 064 whic must therefore not be used by the program. The number of cards prepared by TC:CARDOUT:I before using TC:CARDOUT:II to punch them must not be greater than 32 times the largest integer contained in 21/n (21 times if TC:CARDIN is used in the same program).

3. The special instruction JOTB 000.0 000.0 207.0 must be included in the transcode program so that it is obeyed at some time prior to the first use of TC:CARDOUT:I and is not obeyed thereafter.

4. Numbers will be punched in fixed decimal point notation assuming that their absolute value is less than unity. Numbers to be output must be multiplied by a suitable power of 10 to meet this condition before using TC:CARDOUT:I, otherwise only the fractional part will be punched.

5. Negative numbers are marked by an "X" punching in the most significant digit column.

Title TC:CARDOUT:II

Purpose To punch IBM cards with data prepared by TC:CARDOUT:I.

Tape Control Standard

Program Entry FNTN oof,0 000,0 Z14,0

Time 100 cards per minute.

Notes 1. One entry to TC:CARDOUT:II punches all the cards prepared by TC;CARDOUT:I since the last use of TC:CARDOUT:II.

2. Since the IBM punch must be allowed to coast to a stop before restarting, TC:CARDOUT:II must not be used at intervals closer than 3 seconds.

Title TC: COMPLEX ROOTS

Purpose Solves polynomial equations of degree less than 36 with complex coefficients.

Tape Preparation The coefficients are entered as NUMB (in order of decreasing power) with the real part preceding the imaginary one to DRUMS 017 to 020, each full drum containing 10 coefficients. At the end the sum of coeffs is entered.

Program Entry

Prepare short tape

INST	004		
BSET	000,6	00n,0	000,0
FNTN	001,0	000,0	Z14,0
HALT	000,0	000,0	000,0
QUIT	000,0	000,0	000,0
ENTR			

n = degree of polynomial

Roots are printed in complex form.

Title TC: CONVERSION/S
TC: CONVERSION/L

Purpose To convert Transcode floating numbers to fixed point binary numbers, enabling Transcode numbers to be printed in fixed point decimal form if desired.

These routines required some knowledge of machine code, and detailed information must be obtained from the librarian.

Title TC:COSINE

Purpose To evaluate the cosine of an angle expressed in radians.

Tape Control Standard

Program Entry Standard

Restrictions 1. Error is less than 5 in 10^9.

2. Routine reaches a dynamic stop DS/0 if the angle has modulus greater than about 5×10^7.

Title TC:EXP

Purpose To evaluate the exponential of a number N.

Tape Control Standard

Program Entry Standard

Error At most 1 in 10^{10}.

Restrictions If N is greater than 2^{44}, an IF/G stop occurs. A prepulse will cuase this to be replaced by a Transcode infinity.

TRANSCODE LIBRARY FUNCTION.

Title: TC:EIG/SYM

Purpose: To determine the eigenvalues and eigenvectors of a real symmetric matrix A of order $n \leq 21$.

Method: This routine is a modification to the Real code routine Eig: Jacobi: General, and fuller information is contained in the library sheets for the latter routine. The method used is due to Jacobi, and consists of repeated iterations, each iteration reducing to zero the off-diagonal element of greatest modulus. A group of 20 iterations forms a cycle, upon completion of which a residual R is printed, defined by

$$R = \sum R_i = \sum_i \left| \sum_j a_{ij} \bar{u}_{jl} - \bar{\lambda}_l \bar{u}_{il} \right|$$

where $\bar{\lambda}_l$ and $\bar{u}_{il}$ are the intermediate states of the eigenvalues and eigenvectors respectively.

Tape Preparation and Program Entry: All elements of the matrix are to be scaled down by a factor 10^l

where $10^l \geq 10n|a_{pq}| > 10^{l-1}$
and a_{pq} is the element of A of maximum modulus.

The matrix elements are read by columns, followed by a check column (whose elements are the sums of the several columns of A), to successive Drum locations beginning at Drum 001.

The program is entered by the instructions:

INST 007

BSET	000.6	00n.0	000,0
BSET	000.5	00(2n)0	000.0
BSET	000.4	00l.0	000,0
BSET	000.3	00τ.0	000.0
FNTN	012.0	000.0	000,0
HALT	000.0	000,0	000.0
QUIT	000,0	000,0	000.0
ENTR			

n = order of matrix
l = index of 10 in the scaling factor 10^l used to scale down the matrix elements.
τ = no. of decimal places required in the eigenvalues

The results are given with a correctly-positioned decimal point.

Console Procedure:

1. Read in the tape TC:EIG/SYM, no insertions are necessary to this tape.

2. Read in matrix elements, followed by instructions to initiate the program.

3. The matrix elements are first checked for correct read-in by computing check-sums for the various columns and comparing with the values already in the machine. If TALLY:OK is printed, a

prepulse should be given and actual matrix reduction begins. Otherwise the columns which are incorrect are given together with computed check sum.

4. Once the matrix reduction has begun the following /G stops require attention:-

a) /G stop preceded by the printing of an X. Normally occurs only after the first cycle, when the handswitches should b set to //// and a prepulse given. Otherwise set handswitc to ///½ (depress P18 only) and give a prepulse.

b) /G stop preceded by the printing of a residual R (This gives an indication of the accuracy of the eigenvalues available at this stage).

The following choices are then available:-

Set H = //// - Continue with another cycle (R decreasing and consistent with previous R).

H = ///½ (P18 depressed) - Repeat previous cycle (R questionable).

H = ///T (P19 depressed) - Print the results. ($R < 10^{-7}$ should give 6 correct significant figures in the results).

The results are automatically printed out if R is sufficiently small to give 8 or 9 figure accuracy.

5. Should control go astray in the middle of a cycle, operate K.E set handswitches to :/// (P2 depressed), and read in a tape punched - spaces -- @@K@/ - spaces ---.

Results:

The eigenvalues are printed in a two-column layout, the first column being the eigenvalues and the second column the associated residuals.

The eigenvectors are printed in a four-column layout.

Title TC:LINALG

Purpose A single routine which performs the following:-

(i) calculates the determinant D of the n x n matrix A.
(ii) calculates the inverse matrix of A.
(iii) solves m sets of n simultaneous linear equations with A as common matrix of coefficients but with m different sets of right hand sides, $(n + m \leq 20)$.

Tape Preparation and Console Procedure

Library tape is first read in followed by a short tape to set up the size of the matrix:-

```
INST   004
BSET   000.2      00n.0      000.0   }
BSET   000.3   00(n+m).0      000.0   }  Setting up size
FNTN   012.0      000.0      Z14.0   }  of matrix.
QUIT   000.0      000.0      000.0   }
ENTR
```

Data tape, prepared as below, is then read in:-

```
NUMB   A11, A12, A13, .... A1n, C11, C12, .... C1m, S1"   DRUM 022
       FNTN 000 RECHECK
NUMB   A21, A22, - - - - - - - - -
       FNTN 000 RECHECK
                - - - - - - - - -
NUMB   An1, An2, - - - Ann, Cn1, - - - Cnm, Sn"           DRUM 022
       FNTN 000 RECHECK
```

where the n x n matrix A has elements A_{ij} and the right hand sides form an m x n matrix C.
The check sums $S_1, S_2, \ldots, S_n$, are computed using
$S_1 = A_{11} + A_{12} + \ldots + A_{1n} + C_{11} + \ldots + C_{1m} + 1$, etc.

A hoot after read-in of any row indicates that the check has failed, when the read-in of the row in question should be repeated. Persistent failure indicates an incorrect check-sum or a punch error.

The matrix inversion is called in by the following:-

```
INST   008
FNTN   009.0   000.0   Z14.0
BSET   000.6   000.0   000.0   }
LOOP   00n.0   000.5   000.0   }
READ   001.0   000.6   X00.0   }  Output instructions
PRNT     ?       ?       ?     }
INCB   000.6   001.0   000.0   }
TRNS   004.0   000.5   000.0   }
QUIT   000.0   000.0   000.0   }
ENTR
```

The print instruction is set to give desired layout and accuracy.

$X_i^{(1)}$, $X_i^{(2)}$, . . . $X_i^{(n)}$, A_{i1}^{-1}, A_{i2}^{-1}, . . A_{in}^{-1}, S_i, $(i = 1, . . .$ are in successive storage locations beginning at X01, where X's a solutions of the equations.

If the program proceeds satisfactorily, a sequence - - S : A @ E , will be printed. The console switch should initially be set to Print and when the character E is printed set machine to single prepulse and change output to Punch. The value of the determinant is automatically output.

Notes 1. The printing of an M or P indicates a probable machine fault. After operating KEC a short tape punched as follows should be read in:-

```
INST   002
FNTN   011,0   000,0   Z14,0
QUIT   000,0   000.0   000.0
ENTR
```

The same tape will normally rescue the program after a derail.

If the matrix is larger than 12 x 12 requiring over 15 minutes machine time, and difficulty is encountered please obtain additional information from the librarian as the matrix may be ill-conditioned,

2, If a sequence of numbers, e,g, 11,0, 8,0, 5.0, 0,0 is output before the value of the determinant, obtain information regarding interpretation of results.

3, The program TC:LINALG may be used as a sub-routine in a larger problem. It is assumed that the larger problem calculates the matrix elements and the right hand sides of any equations and writes these elements by rows to DRUMS 001 to 00n,

Provision must be made for presetting n and (n+m) as above, and for initiating the sub-routine TC:LINALG via FNTN 010 (FNTN 009 if check sums already calculated),

Output instructions must be arranged as previously from DRUMS 001 to 00n,

Title TC:LOG

Purpose To evaluate the natural log of a number N.

Tape Control Standard.

Program Entry Standard.

Stops An NL/G stop occurs for negative N. If followed by a prepulse, log (-N) is calculated.

Title TC:RESCUE

Purpose To re-enter a Transcode problem after a derail.

Tape Control Prepared for input as FNTN 064.

Program Entry FNTN 064.0 000.0 ddd.D.
The contents of the B-lines, the X,Y and Z pages and the link list are stored. ddd.D is made negative.

Entry after Derail

Set machine to single prepulses. Operate K.E.C. Set handswitches to T:@/. Switch on continuous prepulses.

This replaces the data stored by the last use of the FNTN entry, refreshes Perm and the Floaters and re-enters the calculation at the position in which FNTN 064 was last obeyed. This returns, as far as possible, to the position at the last FNTN entry. If drum positions have been altered they will not be reset, though it may be possible to use the sign of ddd.D to enter a sequence to look after changes in Drum positions.

Title TC:RUNGEKUTTA

Purpose To perform one step of a modified Rungekutta integration for a set of simultaneous first-order ordinary differential equations of the form

$$\frac{dy_i}{dx} = f_i(x, y_1, y_2, \ldots, y_n)$$

Detailed information sheets obtainable from the librarian.

Title: TC:MATRIX

Purpose: To form the product P = AB of two given matrices A and B where A is an rxn matrix and B an nxm matrix, $m \leq 20$, $n \leq 21$, $r \leq 31$.

Tape Control: The library tape is prepared for input as FNTNS 004, 005, 006 and 007. These FNTN numbers must not therefore be used elsewhere in the program.

Data Preparation: The matrix A is read by rows, preceded by a check row (whose elements are the sums of the several columns of A), to successive Drum locations beginning at DRUM 003.

The matrix B is read by columns, preceded by a check column (whose elements are the sums of the several rows of B) to successive Drum locations beginning at DRUM 001.

Program Entry: The following sequence of Transcode instructions will initiate the TC: MATRIX program.

BSET	000.2	00(m-1).0	000.0
BSET	000.3	00(n-1).0	000.0
BSET	000.4	00 r ,0	000.0
FNTN	004.0	000 ,0	Z14.0

The rows of AB, preceded by a check row (whose elements are the sums of the several columns of the matrix AB), are contained in successive drum locations 001-00(r+1), and may be printed from there as desired.

If it is desired to invert the product matrix (assumed square) using TC:LINALG, the following sequence of instructions will place the rows of AB and the required check sums for TC:LINALG in the correct drum locations:-

(S)	BSET	000.6	000.0	000.0
(S+1)	LOOP	00r.0	000.5	000.0
(S+2)	READ	002.0	000.6	X00.0
(S+3)	ADD	X0(m+1).0	*	X0(m+1).0
(S+4)	WRITE	001.0	000.6	X00.0
(S+5)	INCB	000.6	001.0	000.0
(S+6)	TRNS	00(S+2).0	000.5	000.0

* storage location containing unity.

Stores Altered: (by TC:MATRIX)
Z01, Z02, B6.

Title TC:SIN

Purpose To evaluate the sine of an angle given in radians.

Tape Control Standard.

Program Entry Standard.

Error Less than 10^{-10}.

Title TC:SPECIAL

Purpose To perform any of 7 functions listed below.

Tape Control Standard.

1. Program Entry FNTN 00f.0 000.0 ddd.0.
Numbers are read from tape and placed in successive storage locations starting at ddd.0. The numbers are punched in the usual transcode notation and terminated by a " but without a NUMB tape control. ddd.0 may not be B-modified.

2. Program Entry FNTN 00f.0 00b.0 00c.0.
To check that data present in an X or Y page remains unaltered over a period of time.
To check X page, use 00b = X01.
To check Y page, use 00b = Y01.
When called in as a FNTN, one run through this routine forms a 4-character check sum out of the last 63 lines present on the X or Y page (X01 - X21, Y01 - Y21 respectively). If there is agreement (00c) is made negative. If there is not agreement, (00c) is made positive and the attached check-sum is replaced by the newly-formed one.
In each case control is then returned to the next instruction.

This routine should be used in two separate contexts in a given program.

(1) It should be used at the start of a calculation to form the "attached check-sum" for each page of data stored on the drum.
For example, to attach the correct check-sum to the data stored in drum position 00d, use:

```
READ   00d.0   000.0   X00.0
FNTN   00f.0   X01.0   Z14.0
WRTE   00d.0   000.0   X00.0
```

The (Z14) are of no consequence in this context.

(2) The routine should again be used to perform the check whenever data has been left on an X or Y page for any length of time. The sign of (00c) can now be used to indicate whether or not the check has been successful,

and a repeat procedure may be programmed to operate in case of failure.

For example, to check that data on Page Y has remained unchanged during a series of operations, use:

```
00m         READ   00d.0   000.0   Y00.0
 .           .       .       .       .
            (Series of operations)
 .           .       .       .       .
00n         FNTN   00f.0   Y01.0   Z01.0
00(n+1)     TRNS   00m.0   000.0   Z01.0
            (Next part of program)
```

Here it is assumed that a check-sum has already been attached to DRUM position 00d, that this data remains unchanged on the drum, and that Z01 is available as work space. The "series of operations" is repeated in the event of check-sum failure.

3. Program Entry FNTN 00f.0 00n.1 Z14.0
n line feeds are output. This is useful to divide output into blocks or pages.

4. Program Entry FNTN 00f.0 000.2 Z14.0
The printer is disabled so that future output is punched, provided:
a) Console switch set to 'Print and Punch'
b) P9 of the handswitches is not a 1.

5. Program Entry FNTN 00f.0 000.3 Z14.0
The effect of 4. above is cancelled.

6. Program Entry FNTN 00f.0 000.4 ddd.D
The contents of the handswitches (20 bits) are placed in the exponent line of ddd.D.

7. Program Entry FNTN 00f.0 00n.5 Z14.0
A hoot of about n seconds will occur.