(Version 1, 3.6.55)

The Standard Account of the 'Simple Machine'

By means described elsewhere the electronic computer can be made to accept programmes written in a simplified form, described below. In this form a programme of calculation consists of an ordered sequence of instructions arranged in a single column. These instructions are chosen from a permissible set and employ only the symbols given in Table I. Ultimately the programme is presented to the machine in the form of a length of perforated paper tape which is scanned by a photo-electric tape reader, the input unit of the machine. The machine gives its results on a page printer.

The instructions mostly take the form of equations giving the new value of a computed quantity in terms of one or two previously calculated quantities or parameters. Instructions are also necessary however for selecting between alternative courses of action, for printing results, and for the input of further data. The instructions involve the following kinds of quantities.

(a) Variables The quantities or intermediate quantities which it is necessary to compute. These are denoted by v1, v2, v3, etc.. The number of variables which can be introduced is for practical purposes unlimited: it is in fact about 5,000. The range of magnitudes of a variable is virtually infinite; 2^p, where $2^{18} > p \geqslant - 2^{18}$; the precision is limited to 11 decimal figures.

(b) Constants, or variables of which the value is known is advance. The same restrictions of magnitude and precision apply as for variables. The numerical values are written in the form:

integral part, decimal point, fractional part.

Absolute standardisation of form is not necessary, thus to six significant figures π may be written 3.14159 03.14159 03.141590 +03.141590. All these and similar variations are accepted by the machine. Of the number 003.14159265358979 however only the first 11 significant figures would be recorded inside the machine, i.e., the final 8979 would be omitted. Negative numbers must be preceeded by sign.

(c) Indices In order to take advantage of the repetitive nature of calculation it is desirable to have some means of specifying any of the elements of a sequence, e.g., the components of a vector. For this purpose the notation vn1, vn2, vn3, etc., is introduced where n1, n2, n3, etc., are indices restricted to integral values but otherwise computable like variables. The number of indices which can be introduced is limited to 18. The possible range of values of an index quantity is $2^{18} > n \geqslant - 2^{18}$, which of course far exceeds its usefulness as an index proper. In writing down the numerical value of an index the decimal point may be omitted.

Those instructions which take the form of equations are given below (i) - (iii). Permissible instructions are obtained from these basic forms by replacing x, y, and z by the group of symbols denoting a variable (v), combination (vn), or, except in the case of (iii), index (n). In addition x and y may also be replaced by constants.

(i)	z = x	
(ii)	z = x θ y	θ is replaced by one of the symbols + - $\otimes$ /
(iii)	z = F m (x)	The integer m refers to the function table 2

Instructions are normally obeyed in the order in which they are written down until a jump instruction is encountered. This may be conditional or unconditional and takes the form (iv) or (v) as follows.

(iv)	j m	means 'jump to instruction labelled m'.
(v)	j m, x ϕ y	means 'jump to instruction labelled m if x ϕ y' where ϕ is replaced by one of the symbols $\geqslant$, $>$ $=$ $\neq$.

The label is a positive whole number which is written immediately before (i.e., to the left of) the instruction concerned.

The remaining instructions are

(vi) The symbol * inserted before any of the instructions (i, ii, iii, & viii causes the computed value of z to be printed on a new line in the style described earlier, namely, integral part, decimal point, fractional part, the latter to 10 decimal places. Insignificant zeros are suppressed. In the case of a number outside the range $2^{\pm 38}$, the machine prints on the same line both the integral part and the fractional part of the number in the form a, 2^p, where $\frac{1}{2} > |a| \geqslant \frac{1}{4}$, $2^{18} > p \geqslant - 2^{18}$.

(vii) The letter H is a dummy stop instruction: the machine halts: it can be made to resume operation by pressing a key on the console.

Table I

letters	figures
FS	FS
A	1
B	2
C	$\ast$
D	4
E	(
F	)
G	7
H	8
I	$\neq$
J	=
K	-
L	v
M	LF
N	SP
O	,
P	0
Q	>
R	$\geqslant$
S	3
T	j
U	5
V	6
W	/
X	$\otimes$
Y	9
Z	+
LS	LS
.	.
?	n
£	CR
●	●

Table II

1	$\sqrt{x}$
2	$\cos 2\pi x$
3	$\exp x$
4	$\log_e x$
5	$\frac{1}{\pi}\arctan x.$
6	$\lvert x\rvert$

As a simple exercise in coding write down instructions for evaluating the sum of squares of v1, v2, ..., v100. A suitable sequence is:-

$$\begin{aligned} n1 &= 1 \\ v101 &= 0 \\ 2v102 &= vn1 \otimes vn1 \\ v101 &= v101 + v102 \\ n1 &= n1 + 1 \\ j2,\ 100 &\geqslant n1. \end{aligned}$$

Such a group of instructions may form part of a larger programme involving the variables in question.

A complete programme of instructions and numbers is normally recorded <u>inside</u> the machine before being executed: it is therefore necessary to explain the <u>input</u> procedure.

The programme tape is prepared on a keyboard perforator on which are engraved the standard symbols. The symbols are 'punched' in the conventional sequence, namely from left to right and down the column. Each instruction is followed by the symbols 'CR' (carriage return) and 'LF' (line feed). The keyboard is normally on 'figures' which means that capital letters such as F, H have to be preceeded by 'LS' (letter shift) and followed by 'FS' (figure shift). Figure shift corresponds to blank tape and any number of blanks may separate the figure symbols proper provided the order is preserved. About six inches of blank should be left at the head of the tape. Associated with the keyboard is a printer which gives a printed copy of whatever is punched: this should agree with the original manuscript.

As the programme tape is scanned the instructions are normally recorded in the store where ultimately they will be obeyed. The rate of scanning is approximately two seconds per instruction and the rate of execution about 8 per second. In the case of instructions included between brackets each instruction is executed immediately after it has been read and is not recorded in the programme proper. This facility is used to start the programme simply by including an unconditional jump instruction between brackets, e.g., (j1) which means "stop reading the tape and start obeying the programme at the instruction labelled 1".

Having got the programme started it may be necessary to call on the input medium for further numerical data. This may be done in two ways for which it is necessary to introduce two further instructions. These are:-

(viii) z = I, which means 'replace z (which has the same significance as in (i) - (ii)) by the number formed by the next group of symbols on the tape';

(ix) the letter T which causes the machine to start reading further instructions from the tape, adding them to those already recorded in the store.

Instruction (viii) may be used to read a tape bearing numbers only.

The T instruction, combined with the 'bracket' facility, allows data to be input in the form (z = constant). This is a convenient method of altering parameters in between different runs. Thus, e.g., if the supplementary instructions take the form

$$\begin{aligned} (v23 &= 0.012 \\ v24 &= 0.965 \\ n3 &= 12 \\ &j1\ ) \end{aligned}$$

then the effect will be to repeat the run with these new values of the quantities v23, v24, and n3.

It is hoped to include further facilities, e.g., table II will almost certainly be extended. Supplementary notes may therefore be circulated from time to time.

To illustrate the coding scheme described above, the following calculation is programmed.

It is required to compute

$$\alpha = \frac{1 - (P2/P1)^{2/7}}{1 - (P2/P_3)^{2/7}}, \quad \text{where } p_2 = 30, \; p_3 = 40$$

$$r_e = 1.3 + 2 \left\{ 1 - \frac{\alpha^{3/2} (P2/P_3)^{2/7}}{(P2/P_1)^{2/7}} \right\}$$

$$r_m = 1.3 + 2 \left\{ 1 - \frac{\alpha^{1/2} (P2/P_3)^{2/7}}{(P2/P_1)^{2/7}} \right\}$$

for values of p_1 of the form $40 - \frac{10n}{156.4}$, where $n = 0(1)156$.

The programme is given below

2v1 = 40	$(P_1 = 40)$
v2 = 30	$(P_2 = 30)$
v3 = 40	$(P_3 = 40)$
v4 = 10/156.4	
v5 = v2/v3	(P2 / P3)
v6 = F4(v5)	
v7 = 2/7	
v8 = v7⊗v6	
v9 = F3(v8)	$(P2/P_3)^{2/7}$
1v10 = v2/v1	
v11 = F4(v10)	
v12 = v7⊗v11	
v13 = F3(v12)	$(P2/P_1)^{2/7}$
v14 = 1 - v13	
v15 = 1 - v9	
*v16 = v14/v15	forms and prints α
v17 = F1(v16)	
v18 = v17⊗v9	
v19 = v18/v13	
v20 = v16⊗v19	
v21 = 1 - v20	
v22 = 2⊗v21	
*v23 = 1.3 + v22	forms and prints r_e
v24 = 1 - v19	
v25 = 2⊗v24	
*v26 = 1.3 + v25	forms and prints r_m
v1 = v1 - v4	adjusts P_1
j1, v1 > 29	tests for last cycle
H	halt
(j2)	starts programme.

The following notes are of interest

(1) No attempt has been made to economise on the use of variables. Thus e.g., instead of v9 = F3(v8)one could write v8 = F3(v8)since the argument is no longer needed. However nothing is gained by so doing since space is virtually infinite for problems of this kind.

(2) A further example of laziness is the means adopted for evaluating 2/7. Instead of bothering to evaluate the fractions we have simply included an instruction for doing so, namely v7 = 2/7.

(3) The value of any intermediate quantity can be printed simply by inserting an * before the appropriate instruction. This may be useful in locating mistakes: the * can afterwards be 'erased' by turning it into a [illegible] . x

(4) The maximum number of instructions allowed cannot be given very precisely but a safe estimate is 500, which, in view of their comprehensive nature, should be adequate for the class of problems envisaged.