ELLIOTT 902 COMPUTER

FACTS

902 FACTS

GENERAL INFORMATION

The Elliott 902 is a low-cost, high performance third generation digital computer with a 12-bit word length. It is particularly suited to the solution of industrial and scientific instrumentation and process control problems.

The 902 will operate in ambient temperatures within the range 0°C to 40°C. (The upper limit can be extended by a suitable cooling system). Humidity limits are 20-90% RH with no condensation.

Configuration

A standard 902 system consists of the following 19 in. rack-mounting units:—

- (a) Processor unit
- (b) Store, 4096 or 8192 words (1μs or 2μs)
- (c) Control panel
- (d) Power supply unit
- (e) Teleprinter with control unit

Power Supplies

AC input, single phase, 200-250 volts ±10% at 50 c/s ±1/cs. Power consumption varies from 450VA-1300VA according to speed and capacity of store.

Control Unit

The Control Unit provides either manual or automatic control over the computer. It can also be used to test its operation independently of the peripheral system.

A master switch, operated by a Yale-type key, has two positions, AUTO and MANUAL. When in the AUTO position, only the ON/OFF switch is operative. When in the MANUAL position all control switches are operative.

The ON/OFF switch controls the power switching for the 902 and associated equipment. When switched ON the power supplies are sequenced on in such a manner as to retain store contents; the computer then either

(i) if the master switch is in MANUAL, enters the RESET (quiescent) state, or

(ii) if the master switch is in AUTO, starts obeying program, on interrupt level, at location 256.

The Control Unit also provides the following:

12 two-position Number Generator keys which may be set to represent the bits of a computer word; RESET, STOP and RESTART push-button switches; a three-position, normally neutral, JUMP switch; and INTERRUPT controls consisting of a two-position switch and a MANUAL INTERRUPT push-button.

A loudspeaker gives audible indication of program operation and a teleprinter ON-LINE switch controls the availability of the teleprinter for program operation or off-line use.

Paper Tape Equipment

A 250 ch/sec tape reader with necessary control circuits can be added to the system. Operator's controls for its use are mounted on the 902 control panel.
A tape punch, operating at a maximum speed of 110 ch/sec, with necessary control circuits, can also be added. The control circuits and power supplies for both punch and reader are housed in a 19 in. rack mounting unit.

On-line control option

For on-line control applications the tape reader, punch and teleprinter can be supplied with a modified control unit.

Autonomous Transfer Unit

Provision has been made for an Autonomous Transfer Unit, allowing peripherals having the 900 series interface to operate autonomously, transferring data to or from the store on a 'cycle stealing' basis.

FACTS FOR PROGRAMMERS

The 902 is a parallel computer with a 12-bit word length and a separate memory of 4096 or 8192 words. The store may be extended to a total of 32,768 words by adding multiples of 8192 words. The range of integers allowed by single length working is —65,536 to +65,535.

Word length and format

Numbers and instructions in the 902 are each 12 bits in length.
Numbers are represented in fractional form, negative numbers being held in two's complement form. They will thus be in the range —1 to $1\text{-}2^{-11}$.
The significance of individual bits in a number are shown below:

Bit No.	12	11	10	9	8	7	6	5	4	3	2	1
Significance	-1	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}	2^{-10}	2^{-11}

Instructions are of the single address type, one instruction being represented by a 12-bit word. An instruction word consists of 3 parts.

F	M	N
Function	Mode	Address

4 bits —'F'— Specifies the operation to be carried out.
1 bit —'M'— Specifies the manner in which the Address (see below) is interpreted.
7 bits —'N'— Specifies, generally, the store address of one operand.

The instruction format is as shown below:—

12	11	10	9	8	7	6	5	4	3	2	1
F (value 0-15)				M	N (value 0-127)						

Store

The store of a standard 902 system holds 4096 or 8192 words, each of 12 bits; these words are referred to by addresses in the ranges 0 to 4095 and 0 to 8191 respectively. The nominal cycle time of the store is 2μs, the access time being 0·7μs. A 1μs store is also available.

Registers

The following registers are used by instructions:

S. register (13 bits extendable to 15)

The S register automatically controls the extraction of instructions from the store; it holds a number which is the absolute address in the store of the next instruction to be obeyed. This number is automatically incremented by one as the instruction is extracted so that instructions are obeyed from sequential store locations. For a 4096 word store only 12 bits are necessary in the S register (to represent addresses 0 to 4095): the 13th bit allows extension to a 8192 word store and the extension to 15 bits allows for a 32,768 word store.

A register (12 bits)

The A register, referred to as the Accumulator, holds the operands and results for most arithmetic operations.

E register (11 bits)

The E register, referred to as the Extension Accumulator, allows the use of double length operands and results by certain instructions. Its contents then have significances as below:

Bit No.	11	10	9	8	7	6	5	4	3	2	1
Significance	2^{-12}	2^{-13}	2^{-14}	2^{-15}	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}	2^{-21}	2^{-22}

D register (6 bits extendable to 8)

The D register, referred to as the Pointer, is used together with the N bits of an instruction, for certain instructions, to define an address anywhere in the store as follows:—

Register	D								N						
Bit No.	8	7	6	5	4	3	2	1	7	6	5	4	3	2	1
Significance	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

Thus any address in the range 0 to 8191 (or 32,767) can be generated.

B register (12 bits)

The B register, referred to as the Modifier register, is only used for the modification of instruction addresses

Instruction repertoire and operation times

In the tables that follow, the effects of instructions are defined in terms of the initial and final contents of registers and store locations as follows:—

S, A, E, D, B,	refer to the contents of the various registers.
AE	refers to the contents of registers A and E regarded as a double length number.
F, M, N	refer to the parts of the instruction word being obeyed.

Numbers in square brackets indicate particular bits of registers, e.g. A [1-8] means bits 1-8 of register A.

V refers to the 13 bit address formed from D and N as described above.

Primes indicate the contents of store locations specified e.g. N′ indicates the contents of store location N.

: = means 'is made equal to'

The instruction times quoted are for the 2μsec store, with 1μsec figures in brackets.

Unmodified Instructions

Name	Operation	Other effects	Time μs	Instruction F	M	N	Notes
Copy E to A	A [1-11] : = E A [12] : = O		3·7 (2·7)	14	0	32	
Copy A to E	E : = A [1-11]		3·7 (2·7)	14	1	32	
Read pointer	A [1-8] : = D	[A 9-11] : = S [13-15] [A 12] undefined	3·7 (2·7)	14	0	64	
Load pointer	D : = A [1-8]		4·5 (3·5)	14	1	64	
Shift left	AE : = AE × 2^{N}		4·5 (3·5) + 0·8N	14	0	0-31	AE shifted left N places
Shift right	AE : = AE × 2^{N-128}		4·5 (3·5) + 0·8 (128-N)	14	0	96-127	AE shifted right 128-N places arithmetically (i.e. sign digit regenerated)

Name	Operation	Other effects	Time μs	Instruction F	M	Notes
Read	A : = N′		4·4 (2·4)	4	0	
Write	N′ : = A		5·2 (3·2)	5	0	
Load E	E : = N′ [1-11]	A : = N′ - A	5·9 (3·9)	2	0	
Store E	(N′ [1-11] : = E) (N′ [12] : = O)		5·2 (3·2)	3	0	i.e. E stored as positive number.
Modify	B : = N′	E altered	4·4 (2·4)	0	0	Next instruction modified.

Read	A : = V		4·4 (2·4)	4	1	
Write	V′ : = A		5·2 (3·2)	5	1	
Load E	E : = V′ [1-11]	A : = V′ - A	5·9 (3·9)	2	1	
Store E	(V′ [1-11] : = E) (V′ [12] : = O)		5·2 (3·2)	3	1	i.e. E stored as positive number.
Load A	A : = (S + 1)′	S : = S + 2	5·2 (3·2)	14	0	N = 65
Load D	D : = (S + 1)′ [1-8]	S : = S + 2	5·9 (3·9)	14	1	N = 65
Modify	B : = V′	E altered	4·4 (2·4)	0	1	Next instruction modified.

Add	A : = A + N′		4·4 (2·4)	1	0	
Negate & Add	A : = N′ - A.	E : = N′ [1-11]	5·9 (3·9)	2	0	
Multiply	AE : = A × N′		13·4 (11·4)	12	0	
Divide	A : = AE ÷ N	E affected	14·2 (12·2)	13	0	
Collate	A : = A *and* N′		4·4 (2·4)	6	0	
Count	A : = N′ : = N′ + 1		6·6 (3·6)	10	0	

Name	Operation	Other effects	Time µs	Instruction F	Instruction M	Notes
Add	A : = A + V′		4·4 (2·4)	1	1	
Negate & Add	A : = V′ - A	E : = V′ [1-11]	5·9 (3·9)	2	1	
Multiply	AE : A × V′		13·4 (11·4)	12	1	
Divide	A : = AE ÷ V′	E affected	14·2 (12·2)	13	1	
Collate	A : = A *and* V′		4·4 (2·4)	6	1	
Count	A : = V′ : = V′ + 1		6·6 (3·6)	10	1	

Jump forwards	S : = S + N		3·0 (2·0)	8	0	Only S [1-12] are affected by these instructions; hence these jumps are restricted within 4096 word areas of store. The contents of S are not incremented when a jump takes place.
Jump backwards	S : = S - N		3·0 (2·0)	8	1	
Jump forwards if negative	*if* A < O *then* S : = S + N		3·0 (2·0)	9	0	
Jump backwards if negative	*if* A < O *then* S : = S - N		3·0 (2·0)	9	1	
Jump forwards if zero	*if* A = O *then* S : = S + N		(3·0 (2·0) (A < O) (4·5 (3·5) (A ≥ O)	7	0	
Jump backwards if zero	*if* A = O *then* S : = S - N		3·0 (2·0)(A < O) (4·5 (3·5)(A ≥ O)	7	1	
Jump indirect (Exit).	(S [1-12] : = (N + 1)′ (S [13-15] : = N′ [9-11]		6·6 (3·6)	11	0	
	(S [1-12] : = (V + 1) (S [13-15] : = V′ [9-11]		6·6 (3·6)	11	1	

Name	Operation	Other effects	Time µs	Instruction F	Instruction M	Instruction N	Notes
General input	A : = input word		6·3 (5·3 min)	15	0	0-126	External input source defined by N.
Control panel input	A : = switch input		6·0 (5·0 min)	15	0	127	Input from control panel switches.
General output	Output word : = A		6·0 (5·0 min)	15	1	0-126	Output to destination defined by N.
Interface status read	A : = status word	External action defined by N [1-5]	6·0 (5·0)	14	1	0-31	Available only in extended systems
Interface control		External action defined by N [1-5]	6·0 (5·0)	14	1	96-127	
Interrupt Terminate			12·7 (9·7)	15	1	127	

Modified instructions

The effect of an instruction which is immediately preceded by a function 0 instruction is 'modified' in that the address of the instruction is altered by the addition to it of (generally) the contents of the B register, as loaded by the function 0 instruction.

The exact effects of all useful modified instructions are listed below; in this list:

X is used to mean B+N (12 bits only)

Y is used to mean B+V (V [13-15] unaltered)

Name	Operation	Other effects	Time µs	Instruction F	Instruction M	Notes
Read	A : = X′	E altered	4·4 (2·4)	4	0	
Write	X′ : = A	E altered	5·2 (3·2)	5	0	
Load E	E : = X′ [1-11]	A : = X′ - A	5·9 (3·9)	2	0	
Index	B : = X′	E altered	4·4 (2·4)	0	0	Any number of 0 instructions may be used in sequence.
Read	A : = Y′	E altered	4·4 (2·4)	4	1	
Write	Y′ : = A	E altered	5·2 (3·2)	5	1	
Load E	E : = Y′ [1-11]	A : = Y′ - A	5·9 (3·9)	2	1	
Index	B : = Y′	E altered	4·4 (2·4)	0	1	Any number of 0 instructions can be used in sequence.
Add	A : = A + X′	E altered	4·4 (2·4)	1	0	
Negate & add	A : = X′ - A	E altered	5·9 (3·9)	2	0	
Multiply	AE : = A × X′		13·4 (11·4)	12	0	
Collate	A : = A *and* X′	E altered	4·4 (2·4)	6	0	
Count	A : = X′ : = X′ + 1		6·6 (3·6)	10	0	

Name	Operation	Other effects	Time µs	Instruction F	Instruction M	Notes
Add	A : = A + Y′	E altered	4·4 (2·4)	1	1	
Negate & add	A : = Y′ - A	E altered	5·9 (3·9)	2	1	
Multiply	AE : = A × Y′		13·4 (11·4)	12	1	
Collate	A : = A *and* Y′	E altered	4·4 (2·4)	6	1	
Count	A : = Y′ : = Y′ + 1		6·6 (3·6)	10	1	
Jump	(S : = S + B + N	E altered	3·0 (2·0)	8	0	
Jump	(S : = S - (B + N)	E altered	3·0 (2·0)	8	1	
negative	(*if* A < O *then* (S : = S + B + N	E altered	3·0 (2·0)	9	0	Only S [1-12] can be altered by these instructions, hence these jumps are restricted within 4096 word areas of store. The contents of S are not automatically incremented by 1 when a jump takes place.
	(*if* A < O *then* (S : = S - (B + N)	E altered	3·0 (2·0)	9	1	
Jump	(*if* A = O *then* (S : = S + B + N	E altered	3·0 (2·0) (A < O) 4·5 (3·5) (A ⩾ O)	7	0	
zero	(*if* A = O *then* (S : = S - (B + N)	E altered	(3·0 (2·0) (A < O) (4·5 (3·5) (A ⩾ O)	7	1	
Enter	(N + 1)′ : = S [1-12] + 1 N′ [9-11] : = S [13-15] S [1-12] : = (B + 1)′ S [13-15] : = B′ [9-11]	E altered N′ [1-8] : = D N′[12] undefined	11·8 (6·8)	11	0	
	(V + 1)′ : = S [1-12] + 1 V′ [9-11] : = S [13-15] S [1-12] : = (B + 1)′ S [13-15] : = B′ [9-11]	E altered V′ [1-8] : = D V′ [12] undefined	11·8 (6·8)	11	1	
Input	A : = input word	E altered	6·3 (5·3 min)	15	0	Input source selected by X [1-7]
Output	Output word : = A	E altered	6·0 (5·0 min)	15	1	Output destination selected by X [1-7]

PROGRAM INTERRUPT

The obeying of instructions in sequence can be halted and control transferred to a second program as a result of signals received via the interface unit or of operator action. The sequence of events on receipt of such a signal is:—

(a) Instruction being obeyed is completed (in the case of a modify instruction the following (modified) instruction must be completed).

(b) Registers S and D are automatically transferred to store locations as follows:—

0′ [1-8] : = D
0′ [9-11] : = S [13-15]
1′ : = S [1-12]

(c) Registers S and D are automatically loaded from store locations as follows:—

D : = 128′ [1-8]
S [1-12] : = 129′
S [13-15] : = 128′ [9-11]

The time required for (b) and (c) is 9·6μs. (2μs store), 5·6μs (1μs store).

After a program interrupt has occurred the computer is said to be in 'interrupt level' and no further interrupts can occur under these conditions. When the INTERRUPT TERMINATE instruction (F = 15, M = 1, N = 127) is obeyed the S and D registers are re-loaded from store locations 0 and 1 (the reverse of the operation described in (b) above) and operation on the interrupted program is resumed (the computer is then said to be in 'base level').

The locations from which the S and D registers are loaded on interrupt may be varied by interface signals in an extended system, hence different instruction sequences may be entered as a result of different sources. The sequence of events on receipt of an interrupt signal is:

(a) As (a) above.

(b) As (b) above.

(c) Registers S and D are automatically loaded from reserved store locations as follows:—

D : = (128 + L)′ [1-8]
S[1-12] : = (129 + L)′
S[13-15] : = (128 + L)′ [9-11]

The quantity L above is a number in the range 0-15 defined by the interrupt source (except in the case of non-expandable systems, where L is restricted to the values 0 and 1).

902 INITIAL INSTRUCTIONS

902 Initial Instructions

These instructions form a fixed program which is permanently available as a means of loading program into the store using paper tape as the input medium.

The instructions are brought into use by the JUMP control; when in use they occupy locations 5 to 15 of the store. They operate on interrupt level; once a program has been loaded and an INTERRUPT TERMINATE instruction obeyed the locations become available for normal use. The contents of these locations may alter by use of the initial instructions. Details of the initial instructions and their use are given below.

In the standard system the program tape will be read in via the tape reader which forms part of the teleprinter; if in an extended system a high speed tape reader is included, this will be used instead.

	Location	*F*	*M*	*N*	
	5	0	0	1	constant (+1)
Entry →	6	5	0	17	write modifier
Entry →	7	15	0	0	read 1st character
Entry →	8	7	1	2	test for blank
	9	14	0	122	shift into E
	10	15	0	0	read 2nd character
	11	14	0	6	form word
	12	0	0	17	write into location
	13	5	0	16	defined by modifier
	14	4	0	17	Increment
	15	1	0	5	modifier
	(16	8	1	10)	

Words to be read in are punched as two eight-bit tape rows as follows:

	Tracks							
	8	7	6	5	4	3	2	1
1st row	X	X	Bits 1-6					
2nd row	X	X	Bits 7-12					

The program generally ignores tracks 7 and 8, but if bits 1-6 of the word are zero then a one must be present in one of tracks 7 or 8 of the first row. The effect of blank tape being read is to clear the modifier held in location 17. The first word punched on the tape will then be written into location 16; this must be a 8.1 10 instruction to complete the program. The second word read is written into the modifier location (17) and this determines where subsequent words are to be written.

When blank tape is subsequently read, this clears the modifier again; thereafter the modifier can be re-loaded for another block. Alternatively a dynamic stop or trigger can be placed in location 16.

INSTRUCTIONS FOR PAPER TAPE STATION

Instruction	Effect	Status Bits (5)
15 0 0	Input one character from paper tape reader to $A^{2,3}$	1
15 0 1 15 0 0^{1}	Input one character from teleprinter to $A^{2,3}$	3
15 1 0	Output one character (A_{1-8}) to paper tape punch[2]	2
15 1 1 15 1 0	Output one character (A_{1-8}) to teleprinter[2]	4
15 1 8	Set paper tape station [4] "on-line" when bit 1 = "1" or "off-line" when bit 1 = "0"	
15 0 8	Input paper station status word [4]	

Notes

1 Applies to basic teleprinter controller MC2/67 only.

2 "AUTO" conditions of paper tape controller.

3 The effect of these instructions if C_{1-8} is the character input is defined as: $A_{1-8} := C_{1-8}, A_{9-12} := 0$

4 Available only if on-line program adaptor fitted.

5 If the indicated bit of the status word is one, then the appropriate device is available. If not available then the corresponding input/output instruction will be held up until the device is available, unless the paper tape station is in "on-line" mode, when the hold up will be less than 2μsec.

900 SERIES PAPER TAPE AND INTERNAL CODES

ISO Code Value	Value with Parity	Telecode Character	Binary Pattern	SIR Internal code		ISO Code Value	Value with Parity	Telecode Character	Binary Pattern	SIR Internal code	
				Octal	Decimal					Octal	Decimal
0	0	blank	00000·000			64	192	ʻ(grave)	11000·000	40	32
1	129		10000·001			65	65	A	01000·001	41	33
2	130		10000·010			66	66	B	01000·010	42	34
3	3		00000·011			67	195	C	11000·011	43	35
4	132		10000·100			68	68	D	01000·100	44	36
5	5		00000·101			69	197	E	11000·101	45	37
6	6		00000·110			70	198	F	11000·110	46	38
7	135	Bell[3],	10000·111			71	71	G	01000·111	47	39
8	136		10001·000			72	72	H	01001·000	50	40
9	9	Hor. Tab[1]	00001·001			73	201	I	11001·001	51	41
10	10	Line Feed[2]	00001·010	01	1	74	202	J	11001·010	52	42
11	139		10001·011			75	75	K	01001·011	53	43
12	12		00001·100			76	204	L	11001·100	54	44
13	141	Car. Ret.[3],	10001·101			77	77	M	01001·101	55	45
14	142		10001·110			78	78	N	01001·110	56	46
15	15		00001·111			79	207	O	11001·111	57	47
16	144		10010·000			80	80	P	01010·000	60	48
17	17		00010·001			81	209	Q	11010·001	61	49
18	18		00010·010			82	210	R	11010·010	62	50
19	147		10010·011			83	83	S	01010·011	63	51
20	20	Halt[1],	00010·100			84	212	T	11010·100	64	52
21	149		10010·101			85	85	U	01010·101	65	53
22	150		10010·110			86	86	V	01010·110	66	54
23	23		00100·111			87	215	W	11010·111	67	55
24	24		00011·000			88	216	X	11011·000	70	56
25	153		10011·001			89	89	Y	01011·011	71	57
26	154		10011·010			90	90	Z	01011·010	72	58
27	27		00011·011			91	219	[	11011·011	73	59
28	156		10011·100			92	92	£	01011·100	74	60
29	29		00011·101			93	221	]	11011·101	75	61
30	30		00011·110			94	222	↑	11011·110	76	62
31	159		10011·111			95	95	_ →[3],	01011·111	77	63

32	160	Space	10100·000	00	0	96	96	@[1]	01100·000	
33	33	![3]	00100·001			97	225	a	11100·001	41	33
34	34	"	00100·010	02	2	98	226	b	11100·010	42	34
35	163	½	10100·011	03	3	99	99	c	01100·011	43	35
36	36	$	00100·100	04	4	100	228	d	11100·100	44	36
37	165	%	10100·10?	05	5	101	101	e	01100·101	45	37
38	166	&	10100·110	06	6	102	102	f	01100·110	46	38
39	39	′ (acute)	00100·111	07	7	103	231	g	11100·111	47	39
40	40	(	00101·000	10	8	104	232	h	11101·000	50	40
41	169	)	10101·001	11	9	105	105	i	01101·001	51	41
42	170	*	10101·010	12	10	106	106	j	01101·010	52	42
43	43	+	00101·011	13	11	107	235	k	11101·011	53	43
44	172	, (comma)	10101·100	14	12	108	108	l	01101·100	54	44
45	45	—	00101·101	15	13	109	237	m } ,[5]	11101·101	55	45
46	46	.	00101·110	16	14	110	238	n	11101·110	56	46
47	175	/	10101·111	17	15	111	111	o	01101·111	57	47
48	48	0	00110·000	20	16	112	240	p	11110·000	60	48
49	177	1	10110·001	21	17	113	113	q	01110·001	61	49
50	178	2	10110·010	22	18	114	114	r	01110·010	62	50
51	51	3	00110·011	23	19	115	243	s	11110·011	63	51
52	180	4	10110·100	24	20	116	116	t	01110·100	64	52
53	53	5	00110·101	35	21	117	245	u	11110·101	65	53
54	54	6	00110·110	26	22	118	246	v	11110·110	66	54
55	183	7	10110·111	27	23	119	119	w	01110·111	67	55
56	184	8	10111·000	30	24	120	120	x	01111·000	70	56
57	57	9	00111·001	31	25	121	249	y	11111·001	71	57
58	58	:	00111·010	32	26	122	250	z	11111·010	72	58
59	187	;	10111·011	33	27	123	123		01111·011		
60	60	<	00111·100	34	28	124	252		11111·100		
61	189	=	10111·101	35	29	125	125		01111·101		
62	190	>	10111·110	36	30	126	126		01111·110		
63	63	10	00111·111	37	31	127	255	erase	11111·111		

[1] Ignored by Teleprinter
[2] New line on Flexowriter
[3] Ignored by Flexowriter
[5] Upper case on Teleprinter

INSTRUCTIONS FOR MULTIPLEX INTERRUPT UNIT

Instruction	Effect
15 0 2	Input states of "ready" lines 1-12
15 0 4	Input number of highest priority "ready".
15 1 4	Set "ready" enable mask (Accumulator bits set to "ones" corresponds to enable.)
15 1 5	Set "ready" inhibit mask. (Accumulator bits set to ones corresponds to inhibit.)

Instructions for Real Time Clock

Instruction	Effect
15 0 15	Input contents of count register
15 1 15	Load unit register with A_{1-6} and rate register with A_{7-10}; clear count register
15 1 14	Acknowledge Interrupt

Instructions and Store Locations for Autonomous Transfer Unit

Instructions

Instruction			Effect
14	1	95 + n	Initiate sequence on channel n (n = 1 to 4)
14	1	99 + n	Terminate sequence on channel n (n = 1 to 4)
15	0	7 + n	Read channel n Address Register
15	0	11 + n	Read channel n Count Register

Store Locations

Function	Channel No.			
	1	2	3	4
Data Pointer	128	132	136	140
Data Count	129	133	137	141
Indicators	130	134	138	142
Control Pointer/Status Word	131	135	139	143

Indicator Word

Bit Allocation	Function
1- 3	Store Zone Indicator
4- 5	Spare
6	Output Indicator
7	Cyclic Indicator
8- 9	Control Count
10	Status Word Indicator
11-12	Spare

ANALOGUE TO DIGITAL CONVERTER INSTRUCTIONS

Instruction			Effect
15	1	41	Control Output
15	0	40	Data Input
15	0	41	Status Input

Control Word Format

12	11	10	9	8	7	6	5	4	3	2	1
	X	X	X	X	X	X	X	N	N	N	N

Set Mode: Random "0" / Sequential "1" (bit 12)

Set Multiplexer Address (bits 4–1)

Data Word Format

12	11	10	9	8	7	6	5	4	3	2	1
Ds	D	D	D	D	D	D	D	D	D	0	0

Sign (bit 12)

Status Word Format

12	11	10	9	8	7	6	5	4	3	2	1
	0	0	0	0	0	0	0	N	N	N	N

Read Mode: Random "0" / Sequential "1" (bit 12)

Read Multiplexer Address (bits 4–1)

DIGITAL TO ANALOGUE CONVERTER INSTRUCTIONS

Instruction			Effect
15	1	45	Control Output
15	1	44	Data Output
15	0	45	Status Input

Control Word Format

12	11	10	9	8	7	6	5	4	3	2	1
	X	X	X	X	X	X	X	X	N	N	N

Set Mode: Random "0" / Sequential "1" (bit 12)

Set Multiplexer Address (bits 3–1)

Data Word Format

12	11	10	9	8	7	6	5	4	3	2	1
Ds	D	D	D	D	D	D	D	D	D	X	X

Sign (bit 12)

Status Word Format

12	11	10	9	8	7	6	5	4	3	2	1
	0	0	0	0	0	0	0	0	N	N	N

Read Mode: Random "0" / Sequential "1" (bit 12)

Read Multiplexer Address (bits 3–1)

SYMBOLIC ASSEMBLER PROGRAM—(SAP)

S.A.P. Symbols

i) Identifier. A group of up to 5 letters or numbers commencing with a letter.

ii) Global Identifier List [ ] e.g. [CAT DOG]

iii) Words

a) Integers + 2047 to − 2047

b) Fractions + ·999 to − ·9999

c) Octal Groups & e.g. 0367

d) Specials

e.g. £AF where W [1-12] : = AF [1-12]
£AF/ where W [1-7] : = 0
and W [8-12] : = AF [8-12]

<AF1 AF2> where
W [1-8] : = AF2 [8-15]
W [9-11] : = AF1 [13-15]
W [12] : = 0

Note AF represents any address form.

iv) Address forms

a) Absolute A;B where $0 \leqslant A \leqslant 4095$
and $0 \leqslant B \leqslant 7$

b) Identified address { Identifier
Identifier + Integer

v) Literals

a) Any word form as above

b) Long jumps e.g. 0 £AF=
8 = AF

vi) Directives

a)	Start	*START
b)	Skip	> N
c)	Program Pointer	*PROG = AF
d)	Data Pointer	*DATA = AF
e)	PATCH	↑AF

vii) Comment	(THIS IS A COMMENT)
viii) Titles	(*THIS IS A TITLE)
ix) Program Trigger	% AF
x) End of program	% %
xi) Halt Code	Ⓗ

ERROR INDICATIONS

ERROR NO.		Meaning
0		Unlocated Identifier
1		General contextual error
2		Parity error on source tape
3		Label declared twice
4		Violation of one of the following interlocks,
	(a)	Elements other than comments (& stopcodes) before *START directive
	(b)	No ↑ *PROG or *DATA before the first word or skip
	(c)	No globals list before first word
	(d)	Two *START directives in one program. Tapes read differently on second pass to first pass
5	(a)	Different *START directive
	(b)	More blocks on second pass than first
	(c)	Label address different
	(d)	Identifier not in dictionary on second pass
6		'PROGPTR' or 'DATAPTR' incorrectly located
7		Address error
8		Impermissible character
9		Address form which must be located on first pass is not
10		Number outside permitted range
11		Dictionary overflow
12		More than 95 characters to a line
13		Data page full. (Data and literals clash)
14		Attempt to overwrite binary loader (Loc's 16; 0 to 50; 0)
15		Program spills over 4096 word block boundary
16		Address form greater than size of store permitted
17		No linefeed or new line at start of tape
18		Warning that a skip straddles a page.

TABLES OF BINARY EQUIVALENTS

The purpose of these tables is to assist in the setting of binary addresses on the word generator

1. Select the highest multiple of 64 less than (or equal to) the required address, and work out the difference (if any).
2. Set the first (left-hand) 6 keys to the binary equivalent of the multiple, working from Table A.
3. Set the last (right-hand) 6 keys to the binary equivalent of the difference, working from Table B.

TABLE A

Multiple of 64	*Binary equivalent*	*Multiple of 64*	*Binary equivalent*
0	0000000	2048	0100000
64	0000001	2112	0100001
128	0000010	2176	0100010
192	0000011	2240	0100011
256	0000100	2304	0100100
320	0000101	2368	0100101
384	0000110	2432	0100110
448	0000111	2496	0100111
512	0001000	2560	0101000
576	0001001	2624	0101001
640	0001010	2688	0101010
704	0001011	2752	0101011
768	0001100	2816	0101100
832	0001101	2880	0101101
896	0001110	2944	0101110
960	0001111	3008	0101111
1024	0010000	3072	0110000
1088	0010001	3136	0110001
1152	0010010	3200	0110010
1216	0010011	3264	0110011
1280	0010100	3328	0110100
1344	0010101	3392	0110101
1408	0010110	3456	0110110
1472	0010111	3520	0110111
1536	0011000	3584	0111000
1600	0011001	3648	0111001
1664	0011010	3712	0111010
1728	0011011	3776	0111011
1792	0011100	3840	0111100
1856	0011101	3904	0111101
1920	0011110	3968	0111110
1984	0011111	4032	0111111

TABLE B

Difference	*Binary equivalent*	*Difference*	*Binary equivalent*
0	000000	32	100000
1	000001	33	100001
2	000010	34	100010
3	000011	35	100011
4	000100	36	100100
5	000101	37	100101
6	000110	38	100110
7	000111	39	100111
8	001000	40	101000
9	001001	41	101001
10	001010	42	101010
11	001011	43	101011
12	001100	44	101100
13	001101	45	101101
14	001110	46	101110
15	001111	47	101111
16	010000	48	110000
17	010001	49	110001
18	010010	50	110010
19	010011	51	110011
20	010100	52	110100
21	010101	53	110101
22	010110	54	110110
23	010111	55	110111
24	011000	56	111000
25	011001	57	111001
26	011010	58	111010
27	011011	59	111011
28	011100	60	111100
29	011101	61	111101
30	011110	62	111110
31	011111	63	111111

POWERS OF 2 IN DECIMAL

2^n	n	2^{-n}
2	1	·5
4	2	·25
8	3	·125
16	4	·062 5
32	5	·031 25
64	6	·015 625
128	7	·007 812 5
256	8	·003 906 25
512	9	·001 953 125
1 024	10	·000 976 562 5
2 048	11	·000 488 281 25
4 096	12	·000 244 140 625
8 192	13	·000 122 070 312 5
16 384	14	·000 061 035 156 25
32 768	15	·000 030 517 578 125
65 536	16	·000 015 258 789 062 5
131 072	17	·000 007 629 394 531 25
262 144	18	·000 003 814 697 265 625
524 288	19	·000 001 907 348 632 812 5
1 048 576	20	·000 000 953 674 316 406 25
2 097 152	21	·000 000 476 837 158 203 125
4 194 304	22	·000 000 238 418 579 101 562 5
8 388 608	23	·000 000 119 209 289 550 781 25
16 777 216	24	·000 000 059 604 644 775 390 625
33 554 432	25	·000 000 029 802 322 387 695 313
67 108 864	26	·000 000 014 901 161 193 847 656
134 217 728	27	·000 000 007 450 580 596 923 828
268 435 456	28	·000 000 003 725 290 298 461 914
536 870 912	29	·000 000 001 862 645 149 230 957
1 073 741 824	30	·000 000 000 931 322 574 615 479
2 147 483 648	31	·000 000 000 465 661 287 307 739
4 294 967 296	32	·000 000 000 232 830 643 653 870
8 589 934 592	33	·000 000 000 116 415 321 826 935
17 179 869 184	34	·000 000 000 058 207 660 913 467
34 359 738 368	35	·000 000 000 029 103 830 456 734
68 719 476 736	36	·000 000 000 014 551 915 228 367
137 438 953 472	37	·000 000 000 007 275 957 614 183
274 877 906 944	38	·000 000 000 003 637 978 807 092
549 755 813 888	39	·000 000 000 001 818 989 403 546
1 099 511 627 776	40	·000 000 000 000 909 494 701 773

SOME USEFUL CONSTANTS

	= 3·141 592 653 590	1/	= 0·318 309 886 184
$\log_{10}e$	= 0·301 294 481 903	$\log_e 10$	= 2·302 585 092 994
$\log_{10}2$	= 0·301 029 995 664	e	= 2·718 281 828 459
√2	= 1·414 213 562 373	√3	= 1·732 050 807 569
1 radian	= 57·295 779 513 082°	1°	= 0·017 453 292 520 radian

TM/902/5.69/2

The information in this booklet is accurate at the time of going to press, but Elliott Brothers (London) Limited reserve the right to make amendments as necessary without notice.

ELLIOTT

MOBILE COMPUTING DIVISION
ELLIOTT BROTHERS (LONDON) LIMITED
ELSTREE WAY . BOREHAMWOOD . HERTS
Telephone: 01-953 2030

Issue 2, May 1969