ELLIOTT-AUTOMATION COMPUTERS LIMITED

4100 COMPUTER SYSTEM

Technical Information

FACTS

Contents **Page**

1. GENERAL INFORMATION

The 4100 is a fast, general purpose data processing system designed for a wide range of scientific, technical and commercial applications. It is available with a choice of central processors: the 4120 and the more powerful 4130.

The units of the system are fully self-contained and transportable, and installation of a basic system requires only plugging into suitable mains outlets. To this end the logic is fully transistorised, using silicon components, and is capable of tolerating a wide range of environmental conditions.

The central processors comprise arithmetic and control units together with the necessary power supplies. Magnetic core storage is available in a variety of sizes and speeds to suit specific requirements. The capacity of 6 microsecond store is between 8,192 and 32,768 words; for 2 microsecond store the minimum is 16,384 words while the maximum is 65,536 words for a 4120 and 262,144 for a 4130. An autonomous transfer unit can be fitted to enable data transfers to take place in parallel with normal computing.

All the above units are housed in a series of matching cabinets, further details of which are included in the list of individual items. An operator's desk carries the directly connected control typewriter or teleprinter and the control box on which are mounted the manual controls for the processor.

A feature of the 4100 is the standard peripheral interface, which has been designed to minimise the buffering and control logic needed for any peripheral device, and yet to permit extremely high data transfer rates; there is also a standard store interface, which allows either of the processors to be connected to any store module.

The 4120 and 4130 are parallel, binary processors. The word length is 24 bits and this can represent either one long or two short instructions or an integer I. Integers are in the range $-2^{23} \leqslant I \leqslant 2^{23}-1$, and negative integers are held in the 2's complement form. For certain purposes words can conveniently be thought of as made up from four six-bit syllables or characters. Floating-point arithmetic is performed by extracode functions. The floating-point accumulator contains 48 bits of mantissa and 12 bits of exponent. In store floating-point numbers are normally rounded and packed into two words containing 39 bits of mantissa and 9 bits of exponent.

Instructions are of the single address form, and are specified by a single mnemonic code when using the Elliott Assembly Technique, NEAT. Short instructions occupy 12 bits, 6 for specifying the function and 6 the address, which can therefore only refer to the first 64 store locations. Long instructions may be either of the normal or extracode form. On the 4120 extracodes are performed by software subroutines; on the 4130 certain extracodes are performed by hardware. The address phrase of a long instruction occupies 15 bits and may, in general, be treated as literal, direct, modified or indirect, thus providing powerful addressing facilities. Modified and indirect forms provide the means of addressing up to 262,144 store locations. To designate what type of instruction or addressing is intended, the function mnemonic is followed by a specifying letter, for example:

ADD	direct addressing
ADD :S	short instruction
ADD :L	literal addressing
ADD :M	modified addressing
ADD :I	indirect addressing

There are several registers in the processors which are accessible by program. These are:

M	Main accumulator	24 bits
R	Reserve accumulator	24 bits
S	Sequence control register	17 bits
K	Count register	12 bits
C	Conditions register	24 bits (bits 17–7 not used)
Normal interrupt word		12 bits
Attention interrupt word		12 bits

On the 4130, where floating-point arithmetic is performed by hardware, there are also the following:

A	Mantissa accumulator (most significant)	24 bits
B	Mantissa accumulator (least significant)	24 bits
E	Exponent accumulator	12 bits

These can be accessed collectively by the WUF and FLU extracodes

Dimensions, Weights, Power Consumption, Speeds

Model No.	Description	Speed	Width, depth and height in inches	Weight in pounds	Peak Power consumption in kVA
4120	Central Processor		56×26×63	1000 (8K store)	1·3 (no store) 2·0 (16K store)
4130	Central Processor		70×26×63	1040	2·0
4124	Store Control Unit } A.T.U.		4120: 70×26×63	400	0·9
4125	Packed Transfer Unit } A.T.U.		4130: housed in C.P.	50	0·55
4126	Unpacked Transfer Unit } A.T.U.				
4210	Paper Tape Station		82×25×48	700	1·0
4222	Control Teleprinter on Desk		42×26×57	220	0·3
4227	Teleprinter Controller		42×26×36	300	0·5
4228	Teleprinter Console (for multi-access)				
4229	Real Time Clock				
4241	Card Reader	400 c.p.m.	42×22×50	270	0·3
4246	Card Punch	100 c.p.m.	38×33×40	440	2·5
4255	Line Printer (buffered)	300 l.p.m.	56×30×56	1650	1·5
4259	Line Printer (buffered)	1200 l.p.m.	56×30×56	1650	3·0
4262	Disc Pack Controller		42×26×36		0·35
4263	Disc pack handler	208 kc/s	24×36×41	480	1·5
4274	Master magnetic tape unit	12/33 kc/s	24×19×63	460	0·8
4275	Slave magnetic tape unit	33 kc/s	24×19×63	400	1·1
4276	Master magnetic tape unit	12 kc/s	24×19×63	460	0·8
4277	Slave magnetic tape unit	12 kc/s	24×19×63	400	1·1
4280	Graphical Display Unit		28×41×49	270	0·8
4281	Slave display unit				
4296	Digital Plotter (11 in. 0·005 in. step)	200 steps/sec	42×26×44	82	0·5
4297	Digital Plotter (28 cm. 0·1 mm. step)	300 steps/sec.	42×26×44	82	0·5
4299	Digital Plotter (29·5 in. 0·005 in. step)	300 steps/sec.	52×13×39	90	0·2
4311	Store cabinet for 6 μs. store for 4130		56×26×63	860	2·0 (max.)
4320	Store module, 8192 words	6 μs.	housed in		
4325	Store module 16,384 words	6 μs.	4120 or 4311		
4330	Store module, 16,384 words, in cabinet	2 μs.	70×26×63	1168	2·6
4340	Store module, 32,768 words, in cabinet	2 μs.	70×26×63	1190	3·1
4345	Store module, 65,536 words, in cabinet	2 μs.	70×26×63	1270	3·8

2. NEAT ASSEMBLY LANGUAGE

NEAT is the basic mnemonic code for use on the 4100. The following table lists the available instructions with their execution times for the direct form; where there is no direct form the time for the literal variant is given. The variants available are listed in the variants column; their times can be found by adding the appropriate constants given below:—

Time in microseconds to be added		*Variant*				
		:S	*:L*	*:M*	*:I*	*End around*
4120	6 μs.	+0	−4·9	+1·1	+6·0	+8·2
4120	2 μs.	+0	−1·7	+1·1	+2·8	+5·0
4130	6 μs.	+0	−6·0	+0·75	+6·0	+7·3
4130	2 μs.	+0	−1·5	+0·75	+2·25	+3·5

There are a few exceptions to this, but the above will serve as a general guide. For exact details reference must be made to the Technical Manual.

Notation Used

symbol	*meaning*
m	the contents of M
r	the contents of R
k	the contents of K
fpa	the contents of FPA, the floating-point accumulator
N	value of address part of instruction
n	number held in location with address N
n'	number held in location with address n
(N+r)'	number held in location with address N+r
Q	either N, n, n' or (N+r)'
dQ	double-length operand held in Q, Q+1
tQ	triple-length operand held in Q, Q+1, and Q+2
fQ	floating-point operand held in Q, Q+1
C*j*	bit *j* of number held in register C
n*j*	bit *j* of number held in location with address N
aff	affected in unspecified manner
a, b, c, d	the 4 6-bit characters which make up one word, stored with a at the most significant end.
—	condition remains unaltered.
y	represents the 3rd octal digit of long instructions when this can vary:

y = 0 or 1 literal
y = 2 or 3 direct
y = 4 or 5 modified
y = 6 or 7 indirect
} even for normal functions, odd for extracode.

Note that the C register is not necessarily set according to the final value of the quantities given in the table. The negative marker (C_{24}) is set according to the true sign of the answer. Thus when overflow occurs ($C_{25}:=1$) C_{24} and bit 24 of the answer are different.

Function Code (Octal) short	long	Mnemonic	Effect	Variants	Conditions C_{24} Neg	C_{23} St	C_{22} NZ	C_{21} Ca	C_{20} Of	Times in Microseconds 4120 6 μs.	4120 2 μs.	4130 6 μs.	4130 2 μs.
00	40y	ADD	m:=m+Q	:S:L:M:I	m	m	m	m	m	12·0	5·6	12·0	4·5
01	41y	SUB	m:=m−Q	:S:L:M:I	m	m	m	m	m	12·0	5·6	12·0	4·5
02	42y	NADD	m:=Q−m	:S:L:M:I	m	m	m	m	m	12·0	5·6	12·0	4·5
03	43y	LD	m:=Q	:S:L:M:I	m	m	m	—	—	12·0	5·6	12·0	4·5
04	44y	LDR	r:=Q	:S:L:M:I	r	r	r	—	—	12·0	5·6	12·0	4·5
05		JIR	s:=n;C_{24-18}:=n_{24-18}†		n_{24}	n_{23}	n_{22}	n_{21}	n_{20}	12·0	5·6	12·0	4·5
	450	J	s:=N		—	—	—	—	—	7·1	3·9	6·0	3·0
	45y	JI	s:=Q	:M:I	—	—	—	—	—	12·0	5·6	12·0	4·5
06	46y	AND	m:=m and Q	:S:L:M:I	m	m	m	—	—	12·0	5·6	12·0	4·5
07	47y	ANDN	m:=m and not Q	:S:L:M:I	m	m	m	—	—	12·0	5·6	12·0	4·5

† JIR is a short instruction only, although :S is not used. Note therefore that N⩽63.

Function Code (Octal) short	long	Mnemonic	Effect	Variants	Conditions C_{24} Neg	C_{23} St	C_{22} NZ	C_{21} Ca	C_{20} Of	Times in Microseconds 4120 6 µs.	4120 2 µs.	4130 6 µs.	4130 2 µs.
10	50y	ADDR	r:=r+Q	:S:L:M:I	r	r	r	r	r	12·0	5·6	12·0	4·5
11	51y	SUBR	r:=r—Q	:S:L:M:I	r	r	r	r	r	12·0	5·6	12·0	4·5
12	52y	NADR	r:=Q—r	:S:L:M:I	r	r	r	r	r	12·0	5·6	12·0	4·5
	530	JFL*	0′:=s:C_{24-18}+s;		—	—	—	—	—	14·2	7·8	14·0	4·8
13	53y	JIL	0′:=C_{24-18}+s;***s:=Q	:S :M:I	—	—	—	—	—	18·0	8·4	18·0	6·25
14	54y	LDK	k:=Q	:S:L:M:I†	aff	aff	k	—	—	12·0	5·6	12·0	4·5
	55y	COMP	m-Q; set C_{24-21} from(m-Q)	:L:M:I	m—Q	m—Q	m—Q	m—Q	—	12·0	5·6	12·0	4·5
16	560	JF*	s:=s+N	:S	—	—	—	—	—	7·1	3·9	6·0	3·0
16	56y	JA	s:=s+Q	:S:L:M:I†	—	—	—	—	—	12·0	5·6	12·0	4·5
17	570	JB*	s:=s—N	:S	—	—	—	—	—	7·1	3·9	6·0	3·0
17	57y	JS	s:=s—Q	:S:L:M:I†	—	—	—	—	—	12·0	5·6	12·0	4·5
20	600	JN*	s:=s+N if C_{24}=1	:S ††	—	—	—	—	—	7·1	3·9	6·0	3·0
21	610	JNN*	s:=s+N if C_{24}=0	:S ††	—	—	—	—	—	7·1	3·9	6·0	3·0
22	620	JZ*	s:=s+N if C_{22}=0	:S ††	—	—	—	—	—	7·1	3·9	6·0	3·0
23	630	JNZ*	s:=s+N if C_{22}=1	:S ††	—	—	—	—	—	7·1	3·9	6·0	3·0
24	640	JST*	s:=s+N if C_{23}=1	:S ††	—	—	—	—	—	7·1	3·9	6·0	3·0
25	650	JOF*	s:=s+N if C_{20}=1; then C_{20}:=0	:S ††	—	—	—	—	—	7·1	3·9	6·0	3·0
27	670	DKJN*	**	:S ††	—	—	—	—	—	7·1	3·9	6·0	3·0

* When using NEAT, address phrase is a label.

† The short forms of LDK, JA and JS are literal.

** If k_{12}=0 then k: =k—1
If k_{12}=1 then s: =s+N
The conditions in the above two statements are *not* mutually exclusive;
If initially k=0 both conditions will be true.

†† Both the long and short forms of JN, JNN, JZ, JNZ, JST JOF, DKJN are literal.

*** JIL 0 sets S from location 0 before storing the former value of C_{24-18}+s in location 0.

Function Code (Octal) short	long	Mnemonic	Effect	Variants	Conditions C_{24} Neg	C_{23} St	C_{22} NZ	C_{21} Ca	C_{20} Of	Times in Microseconds 4120 6 μs.	4120 2 μs.	4130 6 μs.	4130 2 μs.
30	60y	ST	Q:=m	:S :M :I	Q	Q	Q	—	—	13·1	6·7	12·75	5·25
31	61y	STR	Q:=r	:S :M :I	Q	Q	Q	—	—	13·1	6·7	12·75	5·25
32	62y	NEGS	Q:=—Q	:S :M :I	Q	Q	Q	Q	Q	13·1	6·7	12·75	5·25
33	63y	SUBS	Q:=Q—m	:S :M :I	Q	Q	Q	Q	Q	13·1	6·7	12·75	5·25
34	64y	ADDS	Q:=Q+m	:S :M :I	Q	Q	Q	Q	Q	13·1	6·7	12·75	5·25
35	65y	CLS	Q:=0	:S :M :I	0	0	0	—	—	13·1	6·7	12·75	5·25
36	66y	INCS	Q:=Q+1	:S :M :I	Q	Q	Q	Q	Q	13·1	6·7	12·75	5·25
37	67y	DECS	Q:=Q—1	:S :M :I	Q	Q	Q	Q	Q	13·1	6·7	12·75	5·25
	70y	GET	Q:=Q (bcda); m:=m(abc)Q(a)	:M :I	Q†	Q	Q	—	—	13·1	6·7	12·75	5·25
	71y	PUT	Q:=Q(bcd)m(d)	:M :I	Q†	Q	Q	—	—	13·1	6·7	12·75	5·25
	72y	DIVM	m:=(r, m)/Q; r:= positive remainder	:L :M :I**	m	m	m	—	m	68·1	61·7	29·0	24·0
	73y	MULM	(r, m) :=r+m×Q (Q is treated as an unsigned integer i.e. $0 \leqslant Q \leqslant 2^{24}-1$)	:L :M :I**	r	r	r	—	—	67·0	60·6	22·0	15·0
	74y	MVE	m:=Q;r:=r—1;r':=m	:M :I	m	m	m	—	r	19·1	9·5	18·0	6·8
	75y	MVB	***Q:=m:=r':r:=r+1	:M :I	m	m	m	—	r	20·1	10·6	18·5	7·3
	76y	EXC	Q⇌m	:M :I	Q	Q	Q	—	—	14·2	7·8	12·75	5·25
	77y	EXCR	Q⇌r	:M :I	Q	Q	Q	—	—	14·2	7·8	12·75	5·25

** DIVM and MULM destroy the contents of K.

*** Here, the meaning of Q depends on the original value of r, not on the new value, r+1

† C_{24} is set by the initial not the final value of Q.

Shift instructions (function 15)

Function Address Phrase (Octal)	Mnemonic	Effect	Conditions					Times in Microseconds			
								4120		4130	
			C_{24} Neg	C_{23} St	C_{22} NZ	C_{21} Ce	C_{20} Of	6 µs.	2 µs.	6 µs.	2 µs.
15 0	SRL	Shift r left k places	r	r	r	—	r	8·2+1·1k	5·0+1·1k	6·5+ $0.75\left\{\frac{k}{4}\right\}$	5·5+ $0.75\left\{\frac{k}{4}\right\}$
15 1	SRLA	Shift r left around k places	r	r	r	—	—				
15 2	SRR	Shift r right k places	r	r	r	—	—				
15 3	SRLC	Shift r k 6-bit characters left around	r	r	r	—	—				
15 4	SML	Shift m left k places	m	m	m	—	m				
15 5	SMLA	Shift m left around k places	m	m	m	—	—				
15 6	SMR	Shift m right k places	m	m	m	—	—				
15 7	SMLC	Shift m k 6-bit characters left around	m	m	m	—	—				
15 12	SRRL	Shift r right logical k places	r	r	r	—	—				
15 16	SMRL	Shift m right logical k places	m	m	m	—	—				
15 20	SRST	Shift r until standardised, or k places, whichever is less	r	r	r	—	—				
15 24	SMST	Shift m until standardised, or k places, whichever is less	m	m	m	—	—				
15 40	SBL	Shift both left k places	r	r	r	—	r	8·2+2·2k	5·0+2·2k	6·5+ $0.75\left\{\frac{k}{4}\right\}$	5·5+ $0.75\left\{\frac{k}{4}\right\}$
15 42	SBR	Shift both right k places	r	r	r	—	—				
15 52	SBRL	Shift both right logical k places	r	r	r	—	—				
15 60	SBST	Shift both until standardised or k places, whichever is less	r	r	r	—	—				

The above shift instructions all exist in short form only. They are written without :S and have no address part.

C_{24} is set according to the value of bit 24 immediately before the final bit(or six bits in the case of character shift) has been shifted.

If k is negative or zero, then the effect on C is undefined. If k is greater than 63 the result of the shift instruction is undefined.

If m or r, is standardised then C is undefined.

$\left\{\frac{k}{4}\right\}$ means the next integer larger than $\frac{k}{4}$. Shift times on the 4130 calculated by the above formula will in any case be only approximate.

Register moves (function 700)

Function/ Address Phrase (Octal)	Mnemonic	Effect	Conditions					Times in Microseconds			
								4120		4130	
			C_{24} Neg	C_{23} St	C_{22} NZ	C_{21} Ca	C_{20} Of	6 μs.	2 μs.	6 μs.	2 μs.
700 00020	KTOR	r: =k	0	0	aff	0	—	7·1	3·9	6·0	3·0
700 00402	MTOR	r: =m	r	r	r	0	—				
700 00404	STOR	r: =s	0	0	r	0	—				
700 00441	CAIR	r: =r+1 if carry set	r	r	r	r	r				
700 00541	CADR	r: =r—1 if carry set	r	r	r	r	r				
700 01001	RTOM	m: =r	m	m	m	0	—				
700 01003	MORR	m: =m or r	m	m	m	0	—				
700 01010	CTOM	m: =c	m	m	m	0	—				
700 02002	RTOS	s: =r	—	—	—	—	—				
700 04002	MTOS	s: =m	—	—	—	—	—				
700 10001	MTOC	c: =m	m_{24}	m_{23}	m_{22}	m_{21}	m_{20}				
700 10002	RTOK	k: =r	r	r	r	0	—				
700 10201	MTOK	k: =m	m	m	m	0	—				
700 12001	RNTK	k: =—r	–r	–r	–r	0	—				
700 21000	ITOM	m: =interrupt word	m	m	m	0	—				
700 41000	ATOM	m: =attention word	m	m	m	0	—				

In NEAT no address phase is used with its register move mnemonics.

Extracode Functions

Function (Octal)	Mnemonic	Effect	Variants	Conditions			Registers affected in unspecified manner	Times in Microseconds			
								4120		4130	
				$C_{24\text{-}22}$	C_{21} Ca	C_{20} Of		6 μs.	2 μs.	6 μs.	2 μs.
40y	FL	fpa: =fQ	:M:I	aff	aff	—	M,R,K	124	64	19	7·5
41y	WF	fQ: =fpa	:M:I	aff	aff	—	M,R,K	181	92	20	8·5
42y	FA	fpa: =fpa+fQ	:M:I	aff	aff	—	M,R,K	365	199	25	15
43y	FS	fpa: =fpa−fQ	:M:I	aff	aff	—	M,R,K	387	211	25	15
44y	FM	fpa: =fpa×fQ	:M:I	aff	aff	—	M,R,K	619	400	50	40
45y	FD	fpa: =fpa/fQ	:M:I	aff	aff	—	M,R,K	630	411	81	70
46y	FCP	Set $C_{24\text{-}22}$ from (fpa−fQ)	:M:I	C_{23} aff $C_{24\text{-}22}$ (fpa−fQ)	—	—	M,R,K	249–356	125–181	18	10
50y	MULS	m: =m×Q	:L:M:I	m unless C_{20}=1	aff	m	K	262	161	36	20
51y	DIV	m: =m/Q; r: = remainder (identical to ALGOL DIV function)	:L:M:I	m unless C_{20}=1	aff	m	K	216	138	33	27·5
52y	BL	(r,m): =dQ	:M:I	aff	aff	—		76	39	19	7·5
53y	WB	dQ: =(r,m)	:M:I	aff	aff	—		90	45	20	8·5
401/ 0	FN	fpa: =−fpa		aff	aff	—	M,R,K	97	49	6·25	5·25
401/ 10	FCF	fpa: =integer m in floating point form		aff	aff	—	M,R,K	133–159	70–95	13	12
401/ 100	FMOD	fpa: =modulus (fpa)		aff	aff	aff	M,R,K	69–116	35–59	6	5/2·75
401/ 110	FENT	m: =entier (fpa)		aff	aff	m	R,K	125–150	65–90	13	12
401/1000	FSIG	if fpa < 0, m: =−1 if fpa = 0, m: =0 if fpa > 0, m: =1		m	aff	—	R,K	84	42	6	3·5
60y	FLU	fpa: =tQ	:M:I	—	—	—	R	172	86	26	11·5
61y	WUF	tQ: =fpa	:M:I	—	—	—	R	165	82	26	11·5
771	TR	Nth letter of alphabet displayed		—	—	—		teleprinter limited			
77y	CH	*Q displayed (octal)	:M:I	—	—	—					
54y	JIRX	Jump indirect and restore link	:M:I	$(N+1)_{24\text{-}22}$	$(N+1)_{21}$	$(N+1)_{20}$		164	85	40	20
55y	JIX	Jump indirect	:M:I	—	—	—		222	113	40	20
56y	JILX	Jump indirect setting link	:M:I	—	—	—		273	139	66	30
57y	INDEX	Access chapter item with index Q placing its address in R	:L:M:I	aff	aff	aff	M	282	153	41	21
411/ 0	CTLA	Copy to lower address		aff	aff	aff		115+52k	58+27k	8·25+12k	6·25+4·5k
411/4096	CTHA	Copy to higher address		aff	aff	aff		121+53k	60+28k	8·25+12k	6·25+4·5k

Function Address Phase (Octal)	Mnemonic	Effect	Conditions					Times in Microseconds			
								4120		4130	
			C_{24} Neg	C_{2} St	C_{22} NZ	C_{21} Ca	C_{20} Of	6 μs.	2 μs.	6 μs.	2 μs.
740 000 NN	IDPR	Input data packed repetitive	—	—	—	—	—	10·4+ (12·3+ 4D)W	7·2+ (9·6+4D) W	13·6+ (12+4D) W	6+ (8+4D) W
740 100 NN	ODPR	Output data packed repetitive	—	—	—	—	—				
740 200 NN	IDUR	Input data unpacked repetitive	—	—	—	—	—	10·4+ (6+D)W	7·2+ (3+D)W	13·6+ (6+D)W	6+(2·5 +D)W
740 300 NN	ODUR	Output data unpacked repetitive	—	—	—	—	—				
750 000 NN	ISPR	Input status word packed repetitive	—	—	—	—	—	10·4+ (12·3+ 4D)W	7·2+ (9·6+4D) W	13·6+ (12+4D) W	6+ (8+4D) W
750 100 NN	OCPR	Output control word packed repetitive	—	—	—	—	—				
750 200 NN	ISUR	Input status word unpacked repetitive	—	—	—	—	—	10·4+ (6+D)W	7·2+ (3+D)W	13·6+ (6+D)W	6+(2·5 +D)W
750 300 NN	OCUR	Output control word unpacked repetitive	—	—	—	—	—				
760 200 NN	IDUM	Input data unpacked single to m	m	m	m	—	—	10·4+D	7·2+D	10+D	6+D
760 300 NN	ODUM	Output data unpacked single from m	m	m	m	—	—				
770 200 NN	ISUM	Input status word unpacked single to m	m	m	m	—	—				
770 300 NN	OCUM	Output control word unpacked single from m	m	m	m	—	—				

D is the device response time.
W is the number of words input or output.
NN is channel number as two octal digits.

NEAT Error Messages
Basic Assembler

No.	*Description*
1	Odd parity character input (on reader or typewriter).
3	Symbol contains impermissible character.
4	Non-existent mnemonic in field 4.
5	Impermissible symbol.
6	either (a) Field 1, 2 or 3 not terminated by a $, tabulate, or newline or (b) % not at the beginning of a line or (c) P: in the wrong field.
7	Floating point number has overflowed.
58	A constant in main chapter is due to overwrite part of the dictionary. This also applies to located constants and is not given until after the END director has been read.
59	Too many digits in an integer.
60	Too few digits in an integer.
61	either (a) Non numeric character in an integer (b) 8 or 9 in octal string or (c) Digit as first character of an identifier.
62	Overflow during the assembly of an integer.
63	Impermissible variant of a function mnemonic.
64	Store full. No further compilation of the program possible.
65	Address phrase greater than the permissible limit.
66	Data or constant name introduced the same as one already introduced.
67	Block or chapter name introduced the same as one already introduced.
68	5th character after C: not a $, tabulate, or newline; or C: is followed by a character which is not in the IN-shift of the standard 4100 internal code.
69	Subscripted data name introduced the same as a data or constant name already introduced.
70	Unintroduced identifier in an address phrase.
71	Integer greater than 24 after B:
72	Separator after B: not: or /
73	: as two consecutive separators after B:
74	Data or constant name as the address phrase of a relative jump under CODE.
76	Integer in field 5 not terminated by a $, tabulate, or newline.
80	either (a) Label name introduced the same as a data or constant name or as another label name already introduced, or (b) Label introduced, the destination of an inter-chapter jump already declared using P:, but for which the appropriate label declaration has not been made.

No.	*Description*
81	Forward relative jump to a non-existent label (this error is given at the end of the chapter in which it occurs).
83	An inter-chapter jump, declared using P:, to a chapter, already introduced, which does not contain the appropriate label declaration.
90	Backward relative jump to a non-existent label.
91	Forward relative jump to a label already introduced.
92	Label under CODE as the address phrase of a relative jump under CONST.
93	Destination of a relative jump using the short variant at a distance greater than 63 half-locations.

3. ALGOL

ALGOL Dynamic Routines
DR0 Basic system
DR1 Paper tape dump
DR10 Disc
DR20 Plotter Procedures
DR25 BINPUT
DR30 Magnetic Tape Procedures
DR31 MTDUMP

Summary of Standard Procedures

In the parameters of these procedures

X	represents	a real expression
Z	represents	a real variable
I & J	represent	integer expressions
M	represents	an integer variable
A	represents	a real array
B	represents	an integer array
C	represents	a real or an integer array
P	represents	a boolean variable
Q	represents	a boolean expression
S	represents	a string
H	represents	a handler number

Real result

ABS (X)	SIN (X)	CHECKR (X)
EXP (X)	COS (X)	
LN (X)	ARCTAN (X)	
SQRT (X)	TAN (X)	
	ARCSIN (X)	
	ARCCOS (X)	

Integer result

ENTIER (X)	SIZE (A)	CHECKI (I)
SIGN (X)	RANGE (A, I)	
	LOWBOUND (A, I)	
	STOREMAX	

Boolean result

BUFFER (I, S)		CHECKB (Q)

Input

"READ" Z, M, P, ...	ADVANCE (I)	READER (I)
INSTRING (B, M)	DECODE (I)	SPECIAL (4)

Output

"PRINT" X, I, S, Q, ...
OUTSTRING (B, M)

DIGITS (I)
SCALED (I)
FREEPOINT (I)
ALIGNED (I, J)
SAMELINE
PREFIX (S)
LEADZERO (S)
GROUPING (I)

PUNCH (I)
CHECKS (S)
SPECIAL (I)
CHECKI (I)
CHECKR (X)

Control

WAIT
RESTART
STOP
PTDUMP

Plotter

SETORIGIN (I, J)
DRAWLINE (I, J)
MOVEPEN (I, J)
CENCHARACTER (I)
WAY (I, J)

Magnetic Tape

MTREAD (H, C, I, J)
MTWRITE (H, C, I, J)
MTMARK (H)
MTBACK (H)
MTREWIND (H)
MTSEEK (H)
MTCOND (H)

MTSOURCE (H, S)
MTDEST (H, S, Q)
MTCLOSE (H, I)

SEGMENTS OF NEAT CODE

Any segment of NEAT code must be written as a basic statement introduced by the symbol "CODE". The lines of code have the form:

"CODE"...

	% function $ address-phrase
or	$$$ function $ address-phrase
or	$$ label $ function $ address-phrase
or	$$ label

and the segment is terminated by the following "END", "ELSE" or ;.

ALGOL Punching Conventions

ALGOL symbol	4100 representation
A–Z	A–Z
a–z	a–z
0–9	0–9
begin end true etc.	"BEGIN" "END" "TRUE" etc.
+ − / ↑	+ − / ↑
×	*
÷	"DIV"
< > =	< > =
≤	"LE"
≥	"GE"
≠	"NE"
∧ ∨	"AND" "OR"
⊃ ≡	"IMPL" "EQUIV"
¬	"NOT"
. , : ; :=	. , : ; :=
( ) [ ]	( ) [ ]
⌜ ⌝	^
⌞	space
10	10

Error Messages

No.	*Description*
1	Number of impermissible form.
2	Error in basic word.
3	Impermissible beginning to a statement.
4	Procedure declaration not terminated by a semicolon.
5	Name in declaration not terminated by semicolon or comma.
6	Names in call of "LIBRARY" not present in Library. (This error is followed by a list of the names not found.)
7	Name declared twice in same blockhead.
8	Label occurring twice in same block.
9	Item in a declaration inadmissible.
10	First item in switch declaration not followed by :-

In a procedure declaration (11 to 21):

11	Item following a procedure name not semicolon or open bracket.
12	No semicolon or close bracket after formal parameter part. N.B. If the item following the close bracket is not a semicolon the compiler will ignore a letter string until : .
13	List in value or specification part has impermissible form.
14	Specification part occurs before value part.
15	More than 10 (ten) parameters are specified as procedures.
16	Too many parameters (more than 24).
17	Parameter not specified.
18	Recursive procedure encountered with real, integer, label or Boolean name parameter.

No.	*Description*
19	Name in value part not a formal parameter.
20	Name in specification part not a formal parameter.
21	Parameter specified twice.
22	Program name of impermissible form. (After this error, the program name is changed to ANON and compilation continues normally.)

In array declarations (23 to 26):

23	No comma or open bracket after identifier.
24	No colon between upper and lower bounds.
25	No comma or closed bracket after bound pair.
26	Array with too many dimensions (more than 63).
27	No "END" or semicolon after statement in compound tail.
28	No colon after a label.
29	Number of words required for variables or for constants exceed 4095.
30	Left part variable in assignment statement not followed by :=
31	Value assigned to a procedure identifier outside procedure body, or type procedure name used within a NEAT segment.
32	Identifier not declared, or used outside scope of declaration.
33	Inadmissible complex primary in arithmetic expression.
34	Missing arithmetic operand at start of arithmetic expression.
35	Missing operand in arithmetic expression.
36	No close bracket after subscript list.
37	Unmatched brackets.
38	Wrong number of subscripts in subscripted variable.
39	Missing "ELSE".
40	Missing "THEN".
41	Conditional statement or expression after "THEN".
42	Missing or inadmissible operator in arithmetic expression.
43	Non arithmetic operand in arithmetic expression.
44	Impermissible use of label name.
46	Inadmissible identifier as controlled variable in "FOR" statement.
47	Missing operand at start of Boolean expression.
48	Missing relational operator.
49	Missing operand in Boolean expression.
50	Inadmissible complex Boolean primary.
51	Inadmissible operator in Boolean expression.
52	Inadmissible symbol at start of an expression.

During procedure call (53 to 56):

53	No open bracket following name of procedure with parameters.
54	Actual parameter not followed by comma or close bracket (cf. error 12).
55	Error in parameter delimiter of form)<letter string>:(
56	No actual switch, procedure or string parameter, where one expected.
58	Controlled variable in "FOR" statement not followed by :=
60	Incorrect designational expression.
61	Arithmetic expression in "FOR" list element not followed by "STEP" "WHILE" "DO" or a comma.
62	Missing "UNTIL".

No.	*Description*
63	Program too large or complex to be compiled (e.g. too many declared identifiers).
64	Occurrence of a ' within inner string.
65	"COMMENT" occurs otherwise than after a semicolon or a "BEGIN".
66	Conditional arithmetic expression on right-hand side of relation.
67	"GO TO" into a "FOR" loop from outside.
68	Amount of code produced for current ALGOL block exceeds 16,383 words.
69	Wrong number of parameters in procedure call.
70	Formal array parameter not replaced by array name.
71	Name output parameter replaced by a variable of a different type, a complex expression, or a constant.
72	Error in segment of NEAT code.
73	Jump to one or more non-existent labels in NEAT code segment.
74	Error in switch declaration.
75	Own array with variable bounds.
76	Symbol following "OWN" not "REAL", "INTEGER" or "BOOLEAN".
77	(i) In the call of a formal procedure: The type and class of an actual parameter do not agree with that expected. (ii) In the call of a procedure having parameters which are specified as procedures: An actual procedure possesses parameters which are not of the same type or class as those of the corresponding formal procedure.

In the call of a procedure having parameters which are specified as procedures (78 to 80):

78	An actual procedure possesses name output parameters, and these do not correspond to simple variables in all the calls of the corresponding formal procedure (cf. error number 71).
79	An actual procedure does not have the same number of parameters as the corresponding formal procedure.
80	An actual procedure is not of the same type as the corresponding formal procedure, or is a function designator when the corresponding formal procedure is not, or vice-versa.

Any number greater than 1000:
This indicates that the compiler cannot continue translation. It usually occurs after a number of other errors.

Other Errors during Translation

(1) The display of the message:

PARITY

can only occur when compiling from paper tape and indicates either:
(i) An odd parity character has been read.
or:

(ii) A character not acceptable to 4100 ALGOL has been read. This includes all those characters which are not in the 4100 extended character set, and ← and ↑

The wrong character will be the last character input by the tape reader. To continue compilation, ignoring the wrong character, type

. (full stop)

If any other character(s) are typed, the compilation is abandoned.

(2) The message

CARE

is displayed if the comment between "END" and the following "END". "ELSE" or semi-colon does not have the form of a single legal ALGOL identifier. The message will never be output more than once for any "END" in the program. It is only a warning and compilation continues normally.

(3) If one or more references to non-existent labels or to labels outside the scope of their blocks occur then the message

NOLABEL

is displayed followed by a list of the offending label names, printed in upper case. If any of the names are in lower case, then a * is printed before each such name. It is possible to run the program, but if any of the labels are used, the program is terminated.

(4) If the program is punched on a paper tape, the absence of a halt code at the end of the tape will cause it to shoot through the reader, instead of stopping at the end.

(5) The following errors may cause the ALGOL program to be completely read, without finishing compilation (for example on the B20 or T20 ALGOL systems, the message RELOAD is displayed):
(i) Insufficient "END"s to match all the "BEGIN"s in the program.
(ii) No semi-colon after the final "END".
(iii) Missing \ at the end of a string, causing the program statements that follow to be treated as part of the string. (This error will be detected (ERROR 64) if an inner string occurs in any subsequent statement.)

(6) The following errors will cause the end of an ALGOL program to be found prematurely:
(i) Missing "BEGIN".
(ii) "END" or the comment following "END" not followed by "END", "ELSE" or semi-colon, causing a "BEGIN" to be treated as part of a comment.

These errors lead to breakdown of the block structure of ALGOL and will usually cause spurious error messages to be displayed.

(7) If an editing error occurs, the message output and the action taken should be the same as for EDIT 41 (see Section 2.14.2 of the 4100 Technical Manual).

(8) When compiling to paper tape, the message

LOW

means that the tape in the paper tape punch is running out. This is a systems message, and no action should be taken until the display

of the message.

PTAPELOW

which is a compiler message, indicates that more tape is required. After reloading the tape punch, continuation is effected by typing a full stop.

The resulting IN-code tapes are quite suitable for input provided the last one is input first, and the first one is input last.

(9) When compiling to store, the message

MCNOTACC

means that the main chapter of the compiled program is placed so that it is not accessible by direct orders. This will happen only on stores greater than 32K and if there are a number of other programs in the store. To rectify the condition, cancel one or more of the other programs and recompile.

RUN TIME MESSAGES

The following errors may occur during the course of a run. On configurations not having immediate continuation, to continue, type

CONT:

unless otherwise stated.

Message	*Description*	*Continuation*
PARITY	An odd parity character has been read from data punched into paper tape, or has occurred on the control typewriter.	Continues when a full-stop is typed, ignoring the wrong character.
READERR	(i) decimal point or exponent separator read in an integer;	Continues with zero as input value.
	(ii) two decimal points read in a number;	Continues with zero as input value.
	(iii) two exponent separators read in a number;	Continues with zero as input value.
	(iv) decimal point read following an exponent separator;	Continues with zero as input value.
	(v) no digits or sign read following an exponent separator;	Continues with zero as input value.
	(vi) no digits, decimal point or exponent separator read after a sign;	Continues with zero as input value.

Message	*Description*	*Continuation*
	(vii) no digits read following a decimal point;	Continues with zero as input value.
	(viii) numeric character found before the first ' when obeying INSTRING;	String terminated in store and next instruction obeyed.
	(ix) occurrence of a within an inner string when obeying INSTRING.	String terminated in store and next instruction obeyed.
	(x) ' or digit read when reading Booleans.	Continues with "FALSE" as input value.
INTOFLO	The result of some integer operation (e.g. division by zero) is outside the range -2^{23} to $2^{23}-1$. Note that the integer overflow is only detected on an integer multiplication or on "DIV," even though it may have been caused by some previous operation (for example, in a NEAT segment).	No continuation.
SWITOFLO	The value of the subscript in a switch designator is outside the range of the switch list.	The program immediately proceeds with the next statement, except in the case where the switch designator is used as an actual parameter, when SUBOFLO is displayed at the time this parameter is used.
SUBOFLO	(i) the subscript of a subscripted variable is outside the declared range;	No continuation.
	(ii) in array declarations: the lower bound is greater than the upper bound, or the lower	No continuation.

Message	*Description*	*Continuation*
	bound is outside the range —256 to +255.	
	(iii) in a call of LOW-BOUND or RANGE: the second parameter is greater than the number of sub-scripts of the array;	No continuation.
	(iv) a switch designator used as an actual parameter: see SWITOFLO above.	No continuation.
NOROOM	The program requires more space than is available in the computer.	No continuation.
SQRT ERR	$X < 0$	Continues with SQRT set to zero. See note (1).
SIN ERR	$\|X\| > 8.5 \times 10^{11}$ (approximately)	Continues with SIN set to zero. See note (1).
COS ERR	$\|X\| > 8.5 \times 10^{11}$ (approximately).	Continues with COS se to zero. See note (1).
TAN ERR	$\|X\| > 4.2 \times 10^{11}$ (approximately).	Continues with TAN set to the largest positive number. See note (1).
LOG ERR	$X < 0$ or $p \dagger q$ with $p < 0$ and q real.	Continues with LOG set to zero. See note (1).
EXP ERR	$X > 254 \log_e 2$ (i.e., 176.0)	Continues with EXP set to the largest positive numbei. See note (1).
ARCS ERR	$\|X\| > 1$	Continues with ARCSIN set to zero. See note (1).
ARCC ERR	$\|X\| > 1$	Continues with ARCCOS set to zero. See note (1).
FPOFLO	Floating point overflow.	Continues with largest number of correct sign as the value used. See note (1).

Message	*Description*	*Continuation*
OUTSERR	The string being output by PREFIX or LEAD-ZERO or OUTSTRING contains an inadmissible character.	Ignores the rest of the string and continues. See note (1).
PRNTERR	Non-zero real number to be output has a mantissa less in absolute value than ‡; the most likely cause of this is that the programmer has failed to assign a value to a variable before printing it.	Outputs zero and continues. See note (1).
BUFF ERR	In a call of BUFFER (I, 's'): (i) More than one character specified in the string. e.g. BUFFER (1, 'AB'); (ii) No character in the string. e.g. BUFFER (1, ''); (iii) Inner string used. e.g. BUFFER (1, ''L');	Continues with the value, 'FALSE'. See note (1).
TOOBIG	An attempt is made to allocate an array in which one or more rows contains more than 32,765 integers or 16,382 real numbers.	No continuation.
PARAM ERR	An attempt is made to enter a procedure with the wrong number of parameters.	No continuation.
RECN ERR	An attempt is made to enter recursively a procedure having "REAL", "INTEGER", "BOOLEAN" or "LABEL" parameters called by name.	No continuation.

Message	*Description*	*Continuation*
NODEST <H>	Attempt made by an ALGOL program to write to handler H when that handler has not been opened by means of MTDEST.	No continuation.
MT FAIL <H>	Entry made to a magnetic tape operation when the last operation on the same handler was not MTCOND but MTCOND would have taken the value —3 or —4.	No continuation.
MTSUBOFL	MTREAD or MTWRITE has been used: (i) to transfer elements outside the sub-script bounds of an array. (ii) to transfer more than 255 real numbers or 511 integers.	No continuation.
PLOT N/A	Operation referring to digital plotter when this device is not available.	Continues when device made available by operator.
LP N/A	Operation referring to lineprinter when this device is not available.	Continues when device made available by operator.
SUM FAIL **D. TO REDUMP	The sum check on the dumped paper tape has failed, or the tape reader has become invalid.	To redump the program type. D. To continue without redumping, type fullstop. This continues with the next statement in the program.
PTLOW **D. TO REDUMP	The punch is running short of tape during a dump operation, and must be reloaded.	As for SUMFAIL, above.
RELOAD	The character just input by the tape reader is halt code.	To continue, type fullstop.

4. FORTRAN

4100 FORTRAN implements full ASA FORTRAN IV except that double precision, complex and data initialisation cannot be used on 4100 systems without backing stores.

Additional features of 4100 FORTRAN include:

Mixed mode arithmetic expressions,
Subscripts may be any integer expression,
Subprograms may be written in NEAT.

Summary of Standard Functions

In the parameters of the following functions:

X & Y	represent real expressions
I & J	represent integer expressions
D & E	represent double precision expressions
C	represents a complex expression

Integer	*Real*	*Double Precision*	*Complex*
	EXP (X).	DEXP (D).	CEXP (C).
	ALOG (X).	DLOG (D).	CLOG (C).
	ALOG10 (X).	DLOG10 (D).	
	SIN (X).	DSIN (D).	CSIN (C).
	COS (X).	DCOS (D).	CCOS (C).
	TANH (X).		
	SQRT (X).	DSQRT (D).	CSQRT (C).
	ATAN (X).	DATAN (D).	
	ATAN2 (X, Y).	DATAN2 (D, E).	
IABS (I).	ABS (X).	DABS (D).	
	CABS (C).		
INT (X).	AINT (X).		
IFIX (X).	FLOAT (I).		
IDINT (D).	SNGL (D).	DBLE (X).	
	REAL (C).		
	AIMAG (C).		CMPLX (X, Y).
MOD (I, J).	AMOD (X, Y).	DMOD (D, E).	CONJG (C).
ISIGN (I, J).	SIGN (X,Y).	DSIGN (D, E).	
IDIM (I, J).	DIM (X, Y).		
MAX0 (I, J...).	AMAX0 (I, J...).		
MAX1 (X, Y...).	AMAX1 (X, Y...).	DMAX1 (D, E...).	
MIN0 (I, J...).	AMIN0 (I, J...).		
MIN1 (X, Y...).	AMIN1 (X, Y...).	DMIN1 (D, E...).	

FORTRAN COMPILE TIME ERROR NUMBERS

1. Illegal character or parity failure.
2. Statement label on continuation line.
3. Non-alphanumeric character following <.>.
4. Unidentified operator between two <.> s.
5. Format error in real constant.
6. Integer constant outside range.
7. Decimal exponent exceeding two digits.
8. Floating point overflow in real constant.
9. <. identifier> not terminated by <.>.
10. Identifier of more than six characters.
11. Two main programs.
12. (i) Impermissible statement following logical IF.
    (ii) Impermissible statement in BLOCK DATA subprogram.
13. (i) Nested subprograms.
    (ii) BLOCK DATA not preceded by SUBROUTINE or FUNCTION statement.
14. Specification statement following statement function, executable statement, or data initialisation statement.
15. (i) No executable statement in (non-BLOCK DATA) subprogram
    (ii) No GOTO, RETURN etc. before END.
16. Error in a label.
17. Statement type cannot be identified. Usually this error is caused by a typing error in a statement-identifying word, or by = not occurring on the first line of an arithmetic statement.
18. Impermissible terminator in DIMENSION statement.
19. Impermissible terminator in EXTERNAL statement.
20. Adjustable subscript in array element name has not appeared in dummy argument list of sub-program.
21. Element other than array in DIMENSION statement.
22. Impermissible terminator in TYPE statement.
23. Impermissible element in EXTERNAL statement.
24. Array subscript not terminated by <,>, or <)>.
25. Array with more than three dimensions.
26. Array subscript not integer variable or integer constant.
27. No terminator after identifier or array.
28. No identifier where one expected.
29. Two Class IV items with the same name.
30. Illegal terminator in EQUIVALENCE statement.
31. Inconsistency in EQUIVALENCE statement.
32. Two items in COMMON have been equivalenced.
33. Inconsistency between dimension of array in EQUIVALENCE statement and any other specification statement.
34. Dummy argument in EQUIVALENCE statement.

35. Element other than identifier or array in EQUIVALENCE statement.
36. No <'> <)> or <(> where one expected in EQUIVALENCE statement.
37. Block name error in COMMON statement.
38. (i) Dummy argument in COMMON statement.
    (ii) Element other than identifier or array in COMMON statement.
39. (i) Identifier appearing twice in COMMON blocks.
    (ii) Identifier appearing twice in same COMMON block.
40. Label defined twice, or DO statement terminated by label already defined.
41. Impermissible statement terminating a DO loop.
42. DO loops nested incorrectly.
43. Input/output statements referring to a non-FORMAT statement.
44. DO loops nested too deeply.
45 Impermissible expression in IF statement.
46. Erroneous use of array in format specification.
47. Non-octal string in PAUSE or STOP statement.
48. Non-existent input/output device specified in input/output statement.
49. Impermissible redefinition of DO loop variable.
50. No label where one expected.
51. Missing delimiter in GOTO, DO or IF statement.
52. Variables in DO or GOTO statements not integer.
53. Impermissible LHS of arithmetic statement.
54. Statement function occurring after executable statement or data initialisation statement.
55. Missing operand in arithmetic expression.
56. Logical operator (i.e. . AND. . OR. or . NOT.) without logical operands.
57. Badly formed complex constant, or superfluous comma after a left bracket.
58. Operator on level zero or left side of assignment statement; or, left bracket or constant first in statement.
59. Missing operator in arithmetic expression.
60. Missing right bracket in arithmetic expression.
61. Missing left bracket after function or array name not in a parameter list.
62. Unexpected comma in arithmetic expression, e.g. function or array with too many arguments.
63. Function or array with too few arguments.
64. Right bracket or comma found on level zero of assignment statement.
65. Assignment to a variable of the wrong type. e.g. <complex> = <logical>.
    However, <complex> = <integer> say, is allowed.
66. Array subscript not of type 'integer'.

67. Exponentiation with an impermissible combination of operands.
68. Arithmetic operator with wrong type of operands, e.g. <logical> + ... However <complex> + <integer> say, is accepted.
69. Impermissable operands for one of the comparison operators. <complex> . EQ. <real> . <complex> . EQ. <complex> is not allowed, but <real> . EQ. <integer> would be.
70. Intrinsic or basic external function with operands or impermissable type. Conversions between integer, real and double precision will be generated rather than giving errors.
71. (i) Impermissible sub-program name.
(ii) Impermissible symbol after sub-program name or
(iii) No <)> after dummy argument list.
72. Subroutine name in CALL statement has already appeared as a variable name.
73. Dummy variable appears twice in dummy argument list.
74. Impermissible symbol in argument list.
75. Error in input/output list.
76. Undefined label.
77. Impermissible formal parameter list in statement function definition.
78. Equals not on level zero of assignment statement.
79. Impermissible use of basic external function, subroutine or function sub-program name in an arithmetic expression.
80. Array subscript in input/output list not of permissible form.
81. No RETURN statement in sub-program.
82. Badly formed variable list in a DATA statement.
83. Badly formed constant list in a DATA statement.
84. A name occurring on a DATA statement is not an array or simple variable.
85. The number of subscripts of an array element in a DATA statement is wrong.
86. Corresponding variables and constants in a DATA statement have differing types.
87. Corresponding variable and constant lists of a DATA statement have differing numbers of elements.
88. Attempt to extend common block backwards by an EQUIVALENCE statement.
89. Error in in signtax of FETCH statement.
98. Arithmetic expression too deeply nested (No continuation).
99. Program too large or difficult to compile (No continuation).
200. Jump to higher level DO loop (Does not effect compilation).
201. Re-definition of array size (Does not effect compilation).
226. A formal parameter used as an array dimension specification is apparently real. It will be considered integer.
281. RETURN statement in main program; considered as STOP.

FORTRAN RUN TIME ERROR MESSAGES

Error Numbers — *Description*

Input/Output Errors

1. Inadequate FORMAT specification found on trying to output a real number.

   The number will be output in FORMAT E 18. 11 (or D32. 25 in the case of double precision numbers) in the same record as the offending FORMAT descriptor. The program will continue running and any further output occurs in such a way that the page layout is not disturbed. e.g.

   17.385 18.204
   G1. 3+0. 123456789E+10
   +19. 204 etc.

   the second record contains the alarm output and the next output is placed in the third record in the same position as if the offending FORMAT descriptor had been correct.

2. Inadequate FORMAT specification found on trying to print an Integer in I FORMAT.

   The integer will be output in FORMAT 18 and the program continues. Further output will be done according to the convention described in RUN ERROR 1.

3. An attempt has been made to print, according to F, E, G or D FORMAT, a real element which is not in standard floating point form.

4. The FORMAT descriptor is not compatible with the type of element to be output.

5. (i) $d \geq w$.
   (ii) decimal point, D or E discovered in integer.
   (iii) two decimal points in number.
   (iv) Decimal point after D or E.
   (v) Two E's or D's in number.
   (vi) No decimal point, E or D in real or double precision number.
   (vii) Null integer strings before and after decimal point.
   (viii) Illegal character found in numeric field.
   (ix) Wrong character found on trying to read a logical constant.

   d=number of digits after decimal point as specified by FORMAT descriptor.
   w=total number of characters as specified by FORMAT descriptor.

6. Incorrect device specified for an operation, e.g. attempt to read from lineprinter or rewind the card reader.

FORMAT Syntax Errors

Since FORMAT information may be input at run time via the A descriptor no attempt is made to check the syntax of the statement until this stage. No continuation is possible.

n=

7. Erroneous character following I, L or A.
8. Errors in D, E, F or G descriptor.
9. Erroneous character following descriptor.
10. Erroneous character following separator.
11. Erroneous character following minus sign.
12. Erroneous character following negative integer
13. Erroneous character following P.
14. Erroneous character following integer.
15. Erroneous character following).
16. Repeat count is zero.
17. There is not a right-hand bracket at end of format statement.
18. Unacceptable character following FORMAT.
19. Descriptor used with storage location of wrong type.
20. There are no descriptors other than nH and nX but there is an I/O list.

Other Run Errors

21. Array will not fit into available store.
22. Illegal extra code – resulting from a program with excessive data declaration.
23. Control variable, of a computed GOTO, out of range.
24. The control variable. of an original GOTO, has not been set up by an ASSIGN statement.

4100 Internal Code

Decimal	*Octal*	*Standard (in-shift)*	*Extended (out-shift)*	*Decimal*	*Octal*	*Standard (in-shift)*	*Extended (out-shift)*
0	00	Space	Null	32	40	@	` (grave)
1	01	Horizontal Tabulate		33	41	A	a
2	02	Linefeed		34	42	B	b
3	03	‡		35	43	C	c
4	04	$		36	44	D	d
5	05	%		37	45	E	e
6	06	&		38	46	F	f
7	07	'(acute)		39	47	G	g
8	10	(		40	50	H	h
9	11	)		41	51	I	i
10	12	*		42	52	J	j
11	13	+	Vertical Tabulate	43	53	K	k
12	14	,(comma)	Formfeed	44	54	L	l
13	15	—	Carriage return	45	55	M	m
14	16	.		46	56	N	n
15	17	/		47	57	O	o
16	20	0		48	60	P	p
17	21	1		49	61	Q	q
18	22	2		50	62	R	r
19	23	3		51	63	S	s
20	24	4	Stop	52	64	T	t
21	25	5		53	65	U	u
22	26	6		54	66	V	v
23	27	7		55	67	W	w
24	30	8		56	70	X	x
25	31	9		57	71	Y	y
26	32	:		58	72	Z	z
27	33	;		59	73	[	"
28	34	<		60	74	£	↑
29	35	=		61	75	]	
30	36	>		62	76	Shift Out	Shift Out
31	37	10		63	77	Shift In	Shift In

Line Printer Codes

Decimal	Octal	Models 4258	Models 4259	Decimal	Octal	Models 4258	Models 4259
		Code A	Code B				
0	00	Space	Space	32	40	@	@
1	01	Horizontal Tabulate	Horizontal Tabulate	33	41	A	A
2	02		"	34	42	B	B
3	03	½	½	35	43	C	C
4	04	$	$	36	44	D	D
5	05	%	%	37	45	E	E
6	06	&	&	38	46	F	F
7	07	' (apostrophe)	´(acute)	39	47	G	G
8	10	(	(	40	50	H	H
9	11	)	)	41	51	I	I
10	12	*	*	42	52	J	J
11	13	+	+	43	53	K	K
12	14	, (comma)	, (comma)	44	54	L	L
13	15	—	—	45	55	M	M
14	16	.	.	46	56	N	N
15	17	/	/	47	57	O	O
16	20	0	0	48	60	P	P
17	21	1	1	49	61	Q	Q
18	22	2	2	50	62	R	R
19	23	3	3	51	63	S	S
20	24	4	4	52	64	T	T
21	25	5	5	53	65	U	U
22	26	6	6	54	66	V	V
23	27	7	7	55	67	W	W
24	30	8	8	56	70	X	X
25	31	9	9	57	71	Y	Y
26	32	:	:	58	72	Z	Z
27	33	;	;	59	73		[
28	34	<	<	60	74	£	£
29	35	=	=	61	75		]
30	36	>	>	62	76		↑
31	37			63	77		`(grave)

Paper Tape Codes

Binary	Decimal	Flexowriter Model T	Teletype Model 33	Binary	Decimal	Flexowriter Model T	Teletype Model 33
00000·000	0	Runout	Runout	11000·000	192	` (grave)	` (grave)
10000·001	129			01000·001	65	A	A
10000·010	130			01000·010	66	B	B
00000·011	3			11000·011	195	C	C
10000·100	132			01000·100	68	D	D
00000·101	5			11000·101	197	E	E
00000·110	6			11000·110	198	F	F
10000·111	135			01000·111	71	G	G
10001·000	136			01001·000	72	H	H
00001·001	9	Horizontal Tabulate		11001·001	201	I	I
00001·010	10	Newline	Linefeed	11001·010	202	J	J
10001·011	139	(Vertical Tabulate)		01001·011	75	K	K
00001·100	12			11001·100	204	L	L
10001·101	141		Carriage return	01001·101	77	M	M
10001·110	142			01001·110	78	N	N
00001·111	15			11001·111	207	O	O
10010·000	144			01010·000	80	P	P
00010·001	17			11010·001	209	Q	Q
00010·010	18			11010·010	210	R	R
10010·011	147			01010·011	83	S	S
00010·100	20	Stop		11010·100	212	T	T
10010·101	149			01010·101	85	U	U
10010·110	150			01010·110	86	V	V
00010·111	23			11010·111	215	W	W
00011·000	24			11011·000	216	X	X
10011·001	153			01011·001	89	Y	Y
10011·010	154			01011·010	90	Z	Z
00011·011	27			11011·011	219	[	[
10011·100	156			01011·100	92	£	£
00011·101	29			11011·101	221	]	]
00011·110	30			11011·110	222	↑	↑
10011·111	159			01011·111	95		←
10100·000	160	Space	Space	01100·000	96	@	
00100·001	33	!	!	11100·001	225	a	
00100·010	34	"	"	11100·010	226	b	
10100·011	163	½	½	01100·011	99	c	
00100·100	36	$	$	11100·100	228	d	
10100·101	165	%	%	01100·101	101	e	
10100·110	166	&	&	01100·110	102	f	
00100·111	39	' (acute)	' (acute)	11100·111	231	g	
00101·000	40	(	(	11101·000	232	h	
10101·001	169	)	)	01101·001	105	i	
10101·010	170	*	*	01101·010	106	j	
00101·011	43	+	+	11101·011	235	k	
10101·100	172	, (comma)	, (comma)	01101·100	108	l	
00101·101	45	—	—	11101·101	237	m	
00101·110	46	.	.	11101·110	238	n	
10101·111	175	/	/	01101·111	111	o	
00110·000	48	0	0	11110·000	240	p	
10110·001	177	1	1	01110·001	113	q	
10110·010	178	2	2	01110·010	114	r	
00110·011	51	3	3	11110·011	243	s	
10110·100	180	4	4	01110·100	116	t	
00110·101	53	5	5	11110·101	245	u	
00110·110	54	6	6	11110·110	246	v	
10110·111	183	7	7	01110·111	119	w	
10111·000	184	8	8	01111·000	120	x	
00111·001	57	9	9	11111·001	249	y	
00111·010	58	:	:	11111·010	250	z	
10111·011	187	;	;	01111·011	123		
00111·100	60	<	<	11111·100	252		
10111·101	189	=	=	01111·101	125		
10111·110	190	>	>	01111·110	126		
00111·111	63	↑	↑	11111·111	255	Delete	Delete

Card Code for T30C

Character	*Code*	*Character*	*Code*
sp	null	@	8-4
`	12-8-7	A	12-1
"	8-7	B	12-2
£	8-3	C	12-3
$	11-8-3	D	12-4
%	0-8-4	E	12-5
&	12	F	12-6
'	8-5	G	12-7
(	12-8-5	H	12-8
)	11-8-5	I	12-9
*	11-8-4	J	11-1
†	12-8-6	K	11-2
'	0-8-3	L	11-3
–	11	M	11-4
°	12-8-3	N	11-5
/	0-1	O	11-6
0	0	P	11-7
1	1	Q	11-8
2	2	R	11-9
3	3	S	0-2
4	4	T	0-3
5	5	U	0-4
6	6	V	0-5
7	7	W	0-6
8	8	X	0-7
9	9	Y	0-8
:	8-2	Z	0-9
;	11-8-6	[	12-8-2
<	12-8-4	‡	0-8-2
=	8-6	]	11-8-2
>	0-8-6	↑	11-8-7
10	0-8-7		

6. PERIPHERAL DEVICES

Tape Reader

5, 6, 7 or 8 channel tape is read in by the Model 4213 reader at up to 1,000 characters/second by the appropriate input instruction to the reader.

Tape Punch

5, 6, 7 or 8 channel tape is punched on the Model 4214 punch at up to 110 characters/second by the appropriate output instruction to the punch.

Typewriter/Teleprinter

Messages to and from the central processor may be typed on the control typewriter or teleprinter.

Card Reader

The Model 4241 card reader with a maximum speed of 400 c.p.m. ma be attached to the 4100. The input magazine will hold 1,200 cards, ar the output stacker 1,800 cards. Cards are read column by column a they move past a single read station consisting of 12 photo-transistors.

Card Punch

The Model 4246 is a card punch with a maximum speed of 100 cards/ minute. Punching occurs column by column, and full check-punching facilities are provided. The input hopper and output magazine can each hold up to 1,000 cards.

Line Printers

The line printers contain the necessary control logic and buffer mounted within the console. Data characters can be transferred into the line printer buffer at a rate of one character every ten micro-seconds. The device response time for the status and control words is approximately five micro-seconds. Two versions of line printer are available for use with 4100 series computers and these print information at speeds of up to 333 and 1250 lines per minute. They have, respectively, 80 and 120 printing positions to the line; the character spacing is ten characters per inch.

Disc Pack

A maximum of eight handlers may be attached to one standard interface channel via a single controller Digital data is stored on 10 disc surfaces, each of which has 100 tracks split into 16 sectors of 64 words. A disc pack, therefore, has an on-line capacity of 1,024,000 words. The maximum time to traverse 33 tracks is 100 milliseconds. Within a track the mean access time is 12.5 millseconds, and a sector is transferred in 1.5 milliseconds.

Magnetic Tape

A maximum of eight handlers may be attached to one standard interface channel via a single controller. Alternative transfer rates of 33,000 and 12,000 characters/second are offered. ECMA compatible, variable length blocks can be written or read. The maximum block length is 2047 characters by direct transfer, or 32,767 characters using autonomous transfer. The inter-block gap is $\frac{3}{4}$ inch. Data parity is checked (when reading) in two ways: laterally on each character and longitudinally on each block. A read-after-write parity check is also made.

The system uses $\frac{1}{2}$ inch tape with 7 tracks—one for the lateral parity check, and the remaining six for data.

Digital Incremental Plotter

The plotter will draw continuous two-dimensional plots as a sequence of linear incremental movements of pen over paper. Basic movements of pen relative to paper can be made along three mutually perpendicular axes. Movement of a spindle holding the paper beneath the pen, and movement of the pen carriage along tracks parallel to the spindle axis give the two-dimensional plot, while the pen can be raised from or lowered on to the paper to move from one trace to another.

Spindle and carriage each give rise to steps by means of geared stepping motors which may be stepped in either direction. The spindle and carriage steps may be called for either separately or together, so that from any one point it is possible to move to one of eight others by a single move. The following models are available:

Model No.	*Step size*	*Nominal chart width*	*Speed steps/sec.*
4296	0·005 in.	12 in.	200
4297	0·1 mm.	34 cm.	300
4299	0·005 in.	30 in.	300

Autonomous Transfer Unit

The Autonomous Transfer Unit is an indirect means of transferring information between the core store and a peripheral device, as distinct from a direct transfer via the central processor and provides an independent extension of the transfer facilities of the 4100 peripheral control scheme. Its packed and unpacked transfer units can be addressed by instructions specific to the unit, and will then effect the transfer without using the central processor's arithmetic unit. The Store Control Unit of the A.T.U shares the core store cycles between the central processor and the transfer units, thus peripheral transfers are interleaved with normal computing. Up to three packed transfer units (PTU) and one unpacked transfer unit (UTU) may be attached to the store control unit.

The Real Time Clock

The real time clock is designed to give an accurate measure of program running time. The clock is housed in the Autonomous Transfer Unit cabinet on 4120 and in processor cabinet in 4130 and is usually connected via a UTU socket.

The timing signals of the clock are generated by a crystal controlled source which has an accuracy of ± ·02% (i.e. less than ± 1 second in 1 hour). The clock produces an interrupt every second. These interrupts may be counted by program or the clock may be set to "ring" after "N" seconds by priming the UTU to transfer "N" characters. At any time the UTU count location in main store may be inspected to determine how much time has elapsed since the last priming of the clock and hence the actual time. The maximum interval for which the UTU can be primed is 8,191 seconds (approximately 2¼ hours).

Multiple Teleprinter Controller

The multiple teleprinter controller provides a means of attaching up to eight teleprinters to a 4100 peripheral channel. The connections from the controller to the teleprinters carry only low frequency signals and may be extended to any length.

The controller allows the transfer of characters between each teleprinter and the central processor. In input mode when the teleprinter buffer is full or in output mode when the buffer is empty the controller selects that teleprinter channel and then interrupts the central processor. The program then inputs a status word to discover which teleprinter interrupted and then transfers a data character from or to the buffer The controller then selects the next waiting teleprinter.

Graphical Display

The Graphical Display has a 10 in. x 10 in. viewing area with 1,024 x 1,024 addressable positions, so that each spot has a resolution of one hundredth of an inch. The Model 4280 Display consists of the display unit itself together with a vector generator for drawing straight lines, a character generator, which displays the characters of the 4100 internal code in three sizes (5/64 in., 5/32 in. and 5/16 in.) and a light pen which enables the operator to indicate points on the display. Up to three model 4281 slave viewing units can be connected.

The picture on the screen is generated from a "display file" in the store of the 4100. The structure of the file is powerful and flexible, and includes subroutine facilities.

The file is output automatically ten times a second to give a flicker free picture, in parallel with central processor operation.

Function times:

Display a spot	100 μsecs. (max.)
Display a character	25 μsecs.
Draw a curve or line	100 μsecs./inch.

Disc Pack — Programmers' Statistics

Handlers per controller . . . . . Up to 8
Maximum transfer rate . . . . . 208 kc/s
Average transfer rate . . . . . 164 kc/s
Word capacity 1,024,000
Character capacity 4,096,000
Words per sector 64
Sectors per track 16
Tracks per cylinder 10
Cylinders per pack 100
Revolution time . . . . . 25 ms 2400 rpm
Head movement times: 1 track . . . . . 30 ms
20 tracks . . . . . 90 ms
50 tracks . . . . . 125 ms
100 tracks . . . . . 150 ms

Magnetic Tape — Programmers' Statistics

Tape length (useful) . . . . . 2,400 ft.
Tape width . . . . . 0·5 in.
Tape speed . . . . . 60 in./sec.
Rewind time . . . . . 2 min. (max.)
Inter-block gap length . . . . . 0·75 in.
Inter-block gap time: full speed . . . . . 12·5 ms.
stop and restart . . . . . 20 ms.
Maximum block-length without P.T.U. . . . 2,047 characters (511 complete words)
Maximum block-length with P.T.U. . . . . 32,767 characters (8191 complete words)

	12 kc/s			33 kc/s		
Transfer rate	12,000 char/sec. 3,000 words/sec.			33,360 char/sec. 8,340 words/sec.		
Packing density	200 char/in. 50 words/in.			556 char/in. 139 words/in.		
Time to transfer 2047 chars.	171 ms.			61 ms.		
Time to transfer 32,767 chars.	2732 ms.			984 ms.		
Average block size	64 words	511 words	8191 words	64 words	511 words	8191 words
Length of block	1·3 in.	10·2 in.	164 in.	0·5 in.	3.7 in.	59·2 in.
Time for one block and gap (at full speed)	34 ms.	183 ms.	2750 ms.	20 ms.	74 ms.	1010 ms.
Blocks per second (max.)	29	5½	⅜	49	13½	1
Effective transfer rate (words/sec.)	1,900	2,800	2,978	3,200	6,900	8,190
No. of blocks per reel	14,200	2,620	175	23,800	6,500	482
No. of words per reel (in thousands)	909	1,340	1,439	1,520	3,330	3,950
No. of characters per reel (in thousands)	3,640	5,360	5,756	6,080	13,320	15,800

CHANNEL NUMBERS

The standard sockets and device numbers of peripheral equipment are:

	Peripheral Socket	*ALGOL Device Number*	*FORTRAN Device Number*
Typewriter/Teleprinter	1	3	1
Reader 1	2	1	3
Punch 1	3	1	5
Reader 2	4	2	4
Punch 2	5	2	6
Digital plotter	6	5	9
Lineprinter	7	4	2
Card reader	8	6	7
Card punch	9	6	8
Multi-access teleprinter controller	10	–	–
Disc Pack	11 (21 via P.T.U.)	7	–
Magnetic tape	12 (16 via P.T.U.)	8	10
Real time clock	14	–	–
Graphical Display	19	9	12

SUMMARY OF STATUS AND CONTROL WORD BITS

Device	*Control*	*Status*
Paper Tape Reader	1. Interrupt Inhibited 2. Interrupt Permitted	1. Busy 2. Manual 3. Unloaded 4. Creep 5. Interrupt Inhibited
Paper Tape Punch	1. Normal Interrupt Inhibited 2. Normal Interrupt Permitted	1. Busy 2. Manual 3. Out of Tape 4. — 5. Interrupt Inhibited
Teleprinter	1. Normal Interrupt Inhibited 2. Normal Interrupt Permitted 3. Input Mode 4. Output Mode	1. Busy 2. Manual 3. Message 4. Error 5. Interrupt Inhibited 6. Input Mode
Line Printer	1. Interrupts and Attentions Inhibited 2. Interrupts and Attentions Permitted 3. Set Manual 4.–8. Paper Movement Control for Unbuffered Printer	1. Busy 2. Manual 3. Interrupt sand Attentions Inhibited 4. — 5. Paper Low 6. Error
Magnetic Tape	*Control (A)* 1. Interrupts and Attentions Inhibited 2. Interrupts and Attentions Permitted 3. Set Handler in Manual 4.–6. Handler Number 7.–8. Binary Value 0=No Off-Line 1=Rewind 2=Back up 3=Erase *Control (B)* 1. Set Even Parity Mode 2. Set Odd Parity Mode 3. Clear check bits	*Status (A)* 1. Controller Busy 2. Manual 3. Interrupts and Attentions Inhibited 4.–6. Handler Number 7. Parity 8. Handler in Attention Interrupt *Status (B)* 1. Handler Busy 2. Write Permit on 3. Short Record 4. Long Record 5. Under 4 Characters 6. Beginning-of-Information Marker 7. End-of-Information Marker 8. Handler in Even Parity Mode

Device	*Control*	*Status*
Multi Access Teleprinter Controller	1.–3. Select channel number 4. Set input mode on channel 5. Set output mode on channel 6. — 7. Inhibit Controller Interrupts 8. Permit Controller Interrupts	1.–3. Currently selected channel number 4. Channel in input mode 5. Channel in output mode 6. Error on channel 7. Message on channel 8. Controller Interrupts Inhibited
Digital Plotter	1. Inhibit Interrupts 2. Permit Interrupts	No status word available
Discpack	*Control (A)* 1.–3. Read as integer 0. No operation 1. Prime for reading 2. Prime for writing 3. Prime for checking 4. Seek forwards 5. Seek backwards 6. Return to cylinder 0 7. Set status 4.–6. Handler number 7. Inhibit Interrupts 8. Permit Interrupts *Control (B)* For read, write or check 1–8. Sector Number For Seek 1–8. Number of Cylinders Movement	*Status (A)* 1.–3. Read as an integer 0. Controller idle 1. Read/write busy 2. Read/write/check successful 3. Read/write/check failed 4. Seek in progress 5. Seek successful 6. Seek unsuccessful 7. Handler free 4.–6. Handler number 7. Interrupts Inhibited 8. Special mode *Status (B)* 1–8. Sector Number *Status (C)* 1–8. Cylinder Number *Status (D)* 1. Handler not available 2. Handler Error 3. Data Error 4. Cyclic Redundancy Check Error 5. Check Error 6. Address Error 7. Seek Out-of-range Cylinder Overflow 8. Writing Not Permitted

Device	*Control*	*Status*
Card Reader	1. Inhibit Interrupts and Attentions 2. Permit Interrupts and Attentions 3. Set Manual 4. — 5. — 6. — 7. Feed a Card	1. Busy 2. Manual 3. Interrupt and Attention Inhibited 4. Missed Transfer 5. End of Pack 6. Error 7. Card in Flight
Card Punch	1. Inhibit Interrupts and Attentions 2. Permit Interrupts and Attentions 3. Set Manual 4. Skip Card	1. Busy 2. Manual 3. Interrupts and Attentions Inhibited 4. — 5. Read Check Error

7. MISCELLANEOUS

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
2 147 483 684	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

Useful Constants

π =	3·141 592 653 590	$1/\pi$ =	0·318 309 886 184
$\log_{10}e$ =	0·434 294 481 903	$\log_e 10$ =	2·302 585 092 994
$\log_{10}2$ =	0·301 029 995 664	e =	2·718 281 828 459
$\sqrt{2}$ =	1·414 213 562 373	$\sqrt{3}$ =	1·732 050 807 569
1 radian =	57·295 779 513 082°	1° =	0·017 453 292 520 radian

ELLIOTT-AUTOMATION COMPUTERS LIMITED
ELSTREE WAY . BOREHAMWOOD HERTS
Telephone: 01-953 2030

CATALOGUE No. 440 – ISSUE 4 OCTOBER 1967