000	LDX	$x' = n + c$	V
001	ADX	$x' = x + n + c$	V
002	NGX	$x' = - n - c$	V
003	SBX	$x' = x - n - c$	V
004	LDXC	$x' = n + c$	C
005	ADXC	$x' = x + n + c$	C
006	NGXC	$x' = - n - c$	C
007	SBXC	$x' = x - n - c$	C

010	STO	$n' = x + c$	V
011	ADS	$n' = n + x + c$	V
012	NGS	$n' = - x - c$	V
013	SBS	$n' = n - x - c$	V
014	STOC	$n' = x + c$	C
015	ADSC	$n' = n + x + c$	C
016	NGSC	$n' = - x - c$	C
017	SBSC	$n' = n - x - c$	C

020	ANDX	$x' = x \& n$
021	ORX	$x' = x \vee n$
022	ERX	$x' = x \not\equiv n$
023	OBEY	Obey the instruction in N
024	LDCH	$x' = n_j$
025	LDEX	$x' = n_e$
026	TXU	Set C if $n \neq x$ or $c = 1$
027	TXL	Set C if $n + c > x$

030	ANDS	$n' = n \& x$
031	ORS	$n' = n \vee x$
032	ERS	$n' = n \not\equiv x$
033	STOZ	$n' = 0$
034	DCH	$n_j' = x_3$
035	DEX	$n_e' = x_e$
036	DSA	$n_a' = x_a$
037	DLA	$n_m' = x_m$

⊙ 040	MPY	$x{:}' = n.x$	V
⊙ 041	MPR	$x' = n.x$ rounded, x^* spoiled	V
⊙ 042	MPA	$x{:}' = n.x + x^*$	V
⊙ 043	CDB	$x{:}' = 10.x{:} + n_j$	V
⊙ 044	DVD	$x^{*\prime} = x{:}/n$, $x' =$ Remainder	V
⊙ 045	DVR	$x^{*\prime} = x{:}/n$ rounded, $x' =$ Remainder	V
⊙ 046	DVS	$x^{*\prime} = x^*/n$, $x' =$ Remainder	V
⊙ 047	CBD	$x{:}' = 10.x{:}$, $n_j' =$ Character	

050	BZE	Branch to N if $x = 0$	
052	BNZ	Branch to N if $x \neq 0$	
054	BPZ	Branch to N if $x \geqslant 0$	
056	BNG	Branch to N if $x < 0$	
060	BUX	Single word modify: $x_m' = x_m + 1$	$x_c' = x_c - 1$ Branch to N if $x_c' \neq 0$
062	BDX	Double word modify: $x_m' = x_m + 2$	
064	BCHX	Character modify: $x_m' = x_m + .1$	
‡ 066	BCT	Count least significant 15 bits of X.	$x_m' = x_m - 1$ Branch to N if $x_m' \neq 0$

I·C·T

INTERNATIONAL COMPUTERS AND TABULATORS LTD.

I.C.T. 1900 SERIES

PLAN - SUMMARISED PROGRAMMING INFORMATION

Form No. 11/129

DATE OF PUBLICATION - MARCH 1965

	070	CALL	Subroutine Entry Link in X	
	072	EXIT	Subroutine Exit Link in X	V
	074		Conditional Branch to N:-	
$X = 0$		BRN	Branch unconditionally	
$X = 1$		BVS	Branch if V is set	
$X = 2$		BVSR	Branch if V is set and clear V	
$X = 3$		BVC	Branch if V is clear	
$X = 4$		BVCR	Branch if V is clear or clear V	
$X = 5$		BCS	Branch if C is set	
$X = 6$		BCC	Branch if C is clear	
$X = 7$		BVCI	Branch if V is clear and/or invert V	V
	‡076		Test floating point accumulator	

100	LDN	$x' = N + c$	
101	ADN	$x' = x + N + c$	V
102	NGN	$x' = - N - c$	
103	SBN	$x' = x - N - c$	V
104	LDNC	$x' = N + c$	
105	ADNC	$x' = x + N + c$	C
106	NGNC	$x' = - N - c$	C
107	SBNC	$x' = x - N - c$	C

$N_t = 0$	110	SLC	Shift x left N_s places. Circular		Single length
$N_t = 1$		SLL	Shift x left N_s places. Logical		
$N_t = 2, 3$		SLA	Shift x left N_s places. Arithmetic	V	
$N_t = 0$	112	SRC	Shift x right N_s places. Circular		
$N_t = 1$		SRL	Shift x right N_s places. Logical		
$N_t = 2$		SRA	Shift x right N_s places. Arithmetic		
$N_t = 3$		SRAV	Shift x right N_s places. Special		
	⊙ 114	NORM	Normalize x	V	
	‡ 116	MVCH	Transfer N characters		

$N_t = 0$	⊙ 111	SLC	Shift $x{:}$ left N_s places. Circular		Double length
$N_t = 1$		SLL	Shift $x{:}$ left N_s places. Logical		
$N_t = 2, 3$		SLA	Shift $x{:}$ left N_s places. Arithmetic	V	
$N_t = 0$	⊙ 113	SRC	Shift $x{:}$ right N_s places, Circular		
$N_t = 1$		SRL	Shift $x{:}$ right N_s places. Logical		
$N_t = 2$		SRA	Shift $x{:}$ right N_s places. Arithmetic		
$N_t = 3$		SRAV	Shift $x{:}$ right N_s places. Special		
	⊙ 115	NORM	Normalize $x{:}$	V	
	‡ 117	SMO	Supplementary modifier to next instruction		

120	ANDN	$x' = x \& N$
121	ORN	$x' = x \vee N$
122	ERN	$x' = x \not\equiv N$
123	NULL	No operation
124	LDCT	$x_c' = N$, $x_m' = 0$
125	MODE	Set mode N
⊙ 126	MOVE	Transfer N words from address x to address x^*
⊙ 127	SUM	$x' =$ Sum of N words from address x^*

TECHNICAL INFORMATION GROUP,
BRIDGE HOUSE NORTH,
PUTNEY BRIDGE APPROACH,
LONDON, S.W.6.

* 130	FLOAT	Convert $n{:}$ from fixed to floating		
* 131	FIX	Convert a from floating to fixed		V
** 132	FAD	$a' = a + n{:}$	If $X = 1$, Unrounded $X = 2$, Not normalized $X = 4$, Interchange a and n	
** 133	FSB	$a' = a - n{:}$		
** 134	FMPY	$a' = a.n{:}$		
** 135	FDVD	$a' = a/n{:}$		
** 136	LFP	$a' = n{:}$	If $X = 1$, $a' = 0$	
** 137	SFP	$n{:}' = a$	If $X = 1$, $n{:}' = a$, $a' = 0$	

* 150 X N(M)	SUSBY	Suspend if peripheral N(M), unit X, is active
* 151 X N(M)	REL	Release peripheral N(M), unit X
* 152 X N(M)	DIS	Disengage peripheral N(M), unit X
* 153 X N(M)		Unassigned
* 154 X N(M)	CONT	Read more program from peripheral N(M), unit X
* 155 X N(M)	SUSDP	Suspend and dump program on peripheral N(M), unit X
* 156 X N(M)	ALLOT	Allocate peripheral N(M), unit X, to the program
* 157 X N(M)	PERI	Initiate peripheral transfer according to control area N(M), unit X

* 160 0 N(M)	SUSTY	Suspend and type message on console typewriter
* 160 1 N(M)	DISTY	Type message on console typewriter without suspension
* 160 2 N(M)	DELTY	Delete program and treat message as console directive
* 161 0 NN(M)	SUSWT	Suspend and type HALTED NN on the console typewriter
* 161 1 NN(M)	DISP	Type DISPLAY NN on the console typewriter without suspension
* 161 2 NN(M)	DEL	Delete program and type DELETED NN on the console typewriter
†* 162 X 0	SUSMA	Suspend if subprogram X is active
†* 163 X N(M)	AUTO	Activate and enter subprogram X at N(M)
†* 164 0 0	SUSAR	De-activate the current subprogram
* 165 X N(M)	GIVE	If N(M) = 0, X will contain date in binary If N(M) = 1, XX* will contain date in character form If N(M) = 2, XX* will contain time in character form If N(M) = 3, X will contain core store allocated to this program

Notes

The function codes 140 to 147 are undefined.

C These instructions may set the carry register but cannot cause overflow.

The carry register C is left clear by any order except 023 and 123, unless that order sets C.

V These instructions may cause overflow.

⊙ These instructions are performed on 1902, 1903 by extracode and by hardware on the other machines.

** These instructions are performed on 1902, 1903, 1904 by extracode and by hardware on the other machines.

* These instructions are performed by extracode on all machines.

† These facilities are not available on 1902, 1903 processors with less than 16 K store.

‡ These instructions are available on 1906, 1907 processors only.

NOTATION

N is a core store address or a 12 bit number.

X is an accumulator (registers 0-7).

M is a modifier register (registers 1-3).

F is a function.

C is the carry register.

c is the content of C (0 or 1).

V is the overflow register.

A is the floating point accumulator.

a is the content of A.

x, m are the contents of X, M respectively.

n is the content of N after modification by m if necessary.

X^* is the accumulator $X + 1$ ($X7^* = X0$)

x^* is the content of X^*.

x', n', a' are the contents of X, N, A after an instruction has been obeyed.

$x:$, $n:$ are double length numbers in X, $X + 1$, and N, $N + 1$ respectively.

S is the sign bit (bit 0).

The most significant bit of the second word of a double length number is always zero.

Subscripts

In general these are applicable to x or n.

x_e is the least significant 9 bits of x. The exponent of a floating point number occupies this portion of the second word.

x_a is the least significant 12 bits of x (the N address of an instruction).

x_c is a 9 bit counter at the most significant end of x.

x_m is the least significant 15 bits of x (the modifier part of an index register).

x_k is the most significant 2 bits of x, used in character modifying with end-around - carry to x_m.

x_d is the least significant 7 bits of x_c.

x_j is any one of x_0, x_1, x_2, x_3, the four 6-bit characters of x.

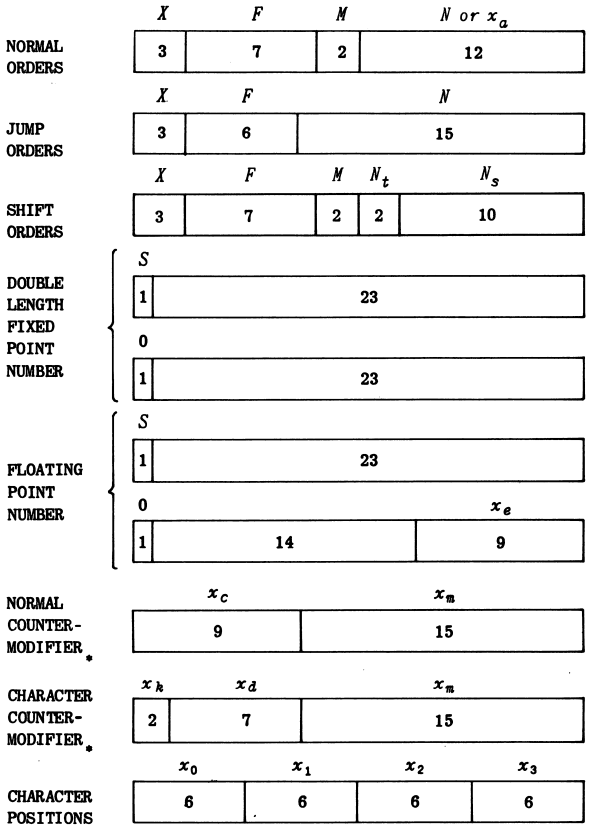
N_t is the most significant 2 bits of the 12 bit N address.

N_s is the least significant 10 bits of the 12 bit N address.

Note:-

* When in extended mode (1906 and 1907 only) the modifier extends to 22 bits, the count being held separately.

24-bit I.C.T. 1900 word



MACRO INSTRUCTIONS (PLAN 3 ONLY)

INSTRUCTION			EFFECT	NO. OF BASIC INSTRUCTIONS
LDX	XX*	N(M)	$x:' = n:$	2
ADX	XX*	N(M)	$x:' = x: + n:$	2
NGX	XX*	N(M)	$x:' = - n:$	2
SBX	XX*	N(M)	$x:' = x: - n:$	2
STO	XX*	N(M)	$n:' = x:$	2
ADS	XX*	N(M)	$n:' = n: + x:$	2
NGS	XX*	N(M)	$n:' = -x:$	2
SBS	XX*	N(M)	$n:' = n: - x:$	2
BXU	X	N_1(M),N_2	If $x \neq n_1$ jump to N_2	2
BXU	XX*	N_1(M),N_2	If $x: \neq n_1:$ jump to N_2	3
BXE	X	N_1(M),N_2	If $x = n_1$ jump to N_2	2
BXE	XX*	N_1(M),N_2	If $x: = n_1:$ jump to N_2	3
BXL	X	N_1(M),N_2	If $x < n_1$ jump to N_2	2
BXL	XX*	N_1(M),N_2	If $x: < n_1:$ jump to N_2	3
BXGE	X	N_1(M),N_2	If $x \geqslant n_1$ jump to N_2	2
BXGE	XX*	N_1(M),N_2	If $x: \geqslant n_1:$ jump to N_2	3
LDSA	X	N(M)	$x' = n_a$	2
LDLA	X	N(M)	$x' = n_m$	2
LDPL	X	N	$x' = N$(15 bits)	1

TAPE MACROS	WTM	X	Write tape mark on MTX	1
	REW	X	Rewind MTX	1
	BSP	X	Backspace MTX	1
	BTM	X	Move back past tape mark on MTX	1
	FTM	X	Move forward past tape mark on MTX	1
	CLOSE	X	Close MTX	1
	SCR	X	OPEN MTX and leave scratch	1
	UNL	X	Close file and unload	

MAJOR DIRECTIVES

The appearance of any directive in this group cancels the effect of any previous directive in the group.

PROGRAM – introduces a section of program instructions

LOWER – introduces lower data (below location 4096)

UPPER – introduces upper data (not Plan 1)

PERIPHERAL – is followed by specification of peripherals (other than magnetic tapes)

MACRO – indicates that a description of a private macro follows (Plan 3 only)

END – the last statement of a segment; ends compilation

FINISH – indicates that this is the last segment to be compiled

PLAN 1 only

COMPLETE – indicates that the program is to be output in consolidated form.

PROGRAM AREA DIRECTIVES

These directives appear in PROGRAM area only.

CUE – gives a label to the following instruction for use by all segments

ENTRY – makes the following instruction entry point N, where N is written in the operand field

MONITOR – introduces specification of monitor printing

GENERAL PURPOSE DIRECTIVES

The directives may appear anywhere in the program

SET – used to define a name (may be reset)

DEFINE – used to define a name (may not be redefined)

– used for writing comments

PAGE – causes paper throw on printer.