DCL-83

INTERPRETED INSTRUCTION CODE OF THE MIT CS II COMPUTER

(see definition of symbols in Table I)

Instr.##	Decimal	Meaning	Definition	Alarm #
	0	(undefined instruction)		D
its al+c	1	(cycle) transfer N(MRA) into (al+2i, al+2i+1)	N(MRA)→N(al+2i)	B, F
iex al+c	2	(cycle) exchange	N(MRA)↔N(al+2i)	B, F
ica al+c	3	(cycle) clear MRA; add N(al+2i)	N(al+2i)→N(MRA)	F
ics al+c	4	(cycle) clear MRA; subtract N(al+2i)	-N(al+2i)→N(MRA)	F
iad al+c	5	(cycle) add	N(MRA)+N(al+2i)→N(MRA)	C', F
isu al+c	6	(cycle) subtract	N(MRA)-N(al+2i)→N(MRA)	C', F
imr al+c	7	(cycle) multiply and roundoff	N(MRA)xN(al+2i)→N(MRA)	C', F, K
idv al+c	8	(cycle) divide	N(MRA)÷N(al+2i)→N(MRA)	C, E, F, K
isp al+c	9	(cycle) transfer of control	Take the next instruction from reg.(al+i) and continue from there	D, F
isc j**	10	select counter	Select cycle count line j	A
icr m**	11	cycle reset	Set i = +0, n = m	
ict al	12	cycle count	Increase i by 1; if $\|i_k\| \geq \|n\|$, reset i=+0 and take next instruction in sequence; if $\|i_k\| < \|n\|$ take next instruction from register al	D, J
iat al	13	add and transfer	Add C(index reg.) to the C(al) and store the result in index register and register al	I
iti al	14	transfer index digits	Transfer the right 11 digits of the index reg. into the right 11 digits of register al	I
sp al	15	transfer of control	Take the next instr. from reg. al and continue from there in uninterpretive mode (sp0 stops computer)	D
ici m**	16	cycle increase	Increase contents of index register by m	G
icd m**	17	cycle decrease	Decrease contents of index register by m	H
iex al	18	cycle exchange	Exchange C(index reg.) with C(al) and exchange C(criterion register) with C(al+1)	
ita al	19	transfer address	Replace the address section of the instruction in register al with the address that is one more than the address of the register containing the last isp (or icp with N(MRA) neg.)	
icp al	20	conditionally transfer control (conditional program)	Take the next instruction from reg. al and continue from there, if N(MRA) is neg.; if N(MRA) is pos. take next instruction in sequence	D
*its al	21	transfer N(MRA) into (al, al+1)	N(MRA)→N(al)	B
*iex al	22	exchange N(MRA) with N(al)	N(MRA)↔N(al)	B
*ica al	23	clear MRA; add N(al)	N(al)→N(MRA)	
*ics al	24	clear MRA; subtract N(al)	-N(al)→N(MRA)	
*iad al	25	add	N(MRA)+N(al)→N(MRA)	C, L
*isu al	26	subtract	N(MRA)-N(al)→N(MRA)	C, L
*imr al	27	multiply and roundoff	N(MRA)xN(al)→N(MRA)	C, K
*idv al	28	divide	N(MRA)÷N(al)→N(MRA)	C, E, K
isp	29	transfer control	Take the next instr. from reg. al and continue from there	D
	30-31	(undefined instructions)		D

* The buffer letter "b" may be used with these instructions only.
**m and j are positive integers less than 2,048.
Consult Alarm Table in this memo.
##For Output Instructions see Table III.

Table I - DEFINITION OF SYMBOLS

Symbol	Meaning
MRA	multiple register accumulator
al	let al represent any floating address, absolute address or buffer address
N(MRA)	the number in the MRA before the instruction is obeyed
N(al)	the number stored in registers al and al+1 before the instruction is obeyed
C(...)	contents of ...
i	C(index register)
i_k	new C(index register)
n	C(criterion register)
N(al+2i)	the number stored in registers al+2i and al+2i+1 before the instruction is obeyed
$\longrightarrow$	replaces
clear...	set the contents of ... to zero
Buffer	block of three registers containing numbers in same form as in MRA

Table II - ALARMS (C', D' are same as C, D except that al+c replaces al)

Check Order Alarms

(A) Counter not provided for by the PA is selected (this can occur only if the "j" in isc j has been modified by the program so that it has become greater than the largest j in the isc j instructions before the program was performed).

(B) Exponent of N(MRA) $\geq 2^j$ where j refers to the (30-j,j) notation (provided al is not a buffer). See * above.

(C) $0 < |C(al)| < 1/2$

(D) When control is transferred to an undefined instruction, an alarm occurs on the undefined instruction.

Divide Error Alarm

(E) C(al) = 0

Arithmetic Overflow Alarms

(F) The contents of the index register could be large enough to cause an alarm; i.e., when al+c > 32,767.

(G) C(index register) +m > 32,767

(H) C(index register) -m < -32,767

(I) $|C(\text{index register}) + C(al)| > 32{,}767$

(J) i = 32,767 before the iet is executed

(K) $|\text{Result}| > 7.0 \times 10^{9863}$ or $|\text{Result}| < 7.1 \times 10^{-9864}$

(L) If al is a buffer, then alarm K could occur.

Table III - OUTPUT INSTRUCTIONS

A. Specifications using either iTOA, iMOA or iSOA

iTOA abcdefg Record N(MRA) on direct printer
iMOA abcdefg Record N(MRA) on delayed printer
iSOA abcdefg Record N(MRA) on scope(film)

} See below for description of a, b, c, d, e, f, g

Ex: iTOA + i 12.345 x 2^{-3} x 10^{2} e
(a)(b)(c)(d)(e) (f) (g)

(a) Sign

	Meaning
+	all numbers will be preceded by sign
-	only negative numbers will be preceded by sign
nothing	numbers will not be preceded by sign

(b) Initial Zeros

	Meaning
i	initial zeros will be skipped
p	initial zeros will be replaced by spaces
n	all numbers will be printed in a normalized form
nothing	initial zeros will be printed as zeros

(c) Digits Left

The programmer indicates the number of digits he wishes to have printed to the left of the decimal point by actually writing a sample number containing the same number of digits to the left of the decimal point as he wishes printed. Thus

iTOA + 123.4567

specifies that the programmer wishes to have his numbers printed with 3 digits to the left of the decimal point. (The magnitude of the digits one writes in the sample number has no effect whatsoever on the program.) If the number actually contains more than three digits to the left of the decimal point, all these digits will automatically be printed out.

(d) Decimal Point

	Meaning
.	A decimal point will appear in all numbers
nothing	No decimal point will be printed. (Used when programmer desires only integral part of number.)
r	No decimal point will be printed. (Used when programmer expects all results to be less than one; if the number in the MRA should unexpectedly exceed unity, then the integral part of the number will also be printed out but no decimal point will separate the integral from the fractional part of the number.

(e) Digits Right

If a programmer were to use the sample number illustrated in (c), he would get four digits printed out to the right of the decimal point (or to the right of where the decimal point should be).

(f) Scale Factors

Powers of 2 and 10 may be used as scale factors to multiply the number in the MRA before it is printed out:

(1) Every factor must be preceded by a lower case x.
(2) $|\alpha|, |\beta| \leq 99$

(g) Terminal Characters (character used to terminate a number)	Meaning
s	space
ss	2 spaces
sss	3 spaces
ssss	4 spaces
c	carriage return
t	tab
nothing	carriage of typewriter will remain exactly where it was after the last number was typed
f	format (see section C)

EXAMPLES

	N(MRA)	OUTPUT REQUEST	PRINTED RESULT	
(1)	-7.953261	iTOA + 123.1234s	-007.9532 space	DIRECT
		iTOA + i123.1234	-7.9532	
		iTOA p123.1234c	7.9532 car. ret.	
		iTOA - n123.1234t	-795.3261 -02 tab	
(2)	+795.3261	iMOA 123.123ss	795.326 space space	DELAYED
		iMOA - i1234.5x10^{2}c	79532.6 car. ret.	
		iMOA + p12r34	+79532	
		iMOA n1.234t	7.953 +02 tab	

B. Special Characters To print out a single special character (such as a decimal point, space, tab, sign, or carriage return) the programmer follows the iTOA or iMOA by the single symbol representing the desired character (as indicated in sections (a), (d), and (g) of table III).

e.g. iTOAc

will cause a carriage return to be typed on the "direct" typewriter.

C. Format Specification (This facility provides the programmer with an automatic device for obtaining a suitable layout of his output data.)

If "f" is used as a terminal character in an output request (see Table III-section (A)-g) then the instruction and 3 program parameters

iFOR
α
β
γ

must appear somewhere in the program before the first output instruction containing the "f". (This will furnish the CS output section with the necessary layout information before a number is printed out.)

α represents the number of words/line (maximum number of characters per line is 155)

β represents the number of spaces between words (A tab is obtained by setting $\beta = 0$)

γ represents the number of words per block (The maximum γ is 32,767. Since the block counter is automatically reset after each block is completed, the upper limit, 32,767, for γ is not a significant limitation.)

ex. 1 Supposing the programmer wishes to have 2500 words typed out and uses

```
iFOR
+12
+2
+400
```

The output request iTOA+12.345f will then give 12 words per line, 2 spaces between words and 400 words per block. The blocks are separated by 2 carriage returns. In this example there will be six blocks of 400 words and one block of 100 words. (The programmer should provide carriage returns at the beginning and at the end of his print-out if the latter doesn't coincide with the end of a block.)

Table IV - AUXILIARY EQUIPMENT INSTRUCTIONS

Magnetic Drum

The drum may be used with the instructions iDIB, iDOB where D designates Drum, I designates Input (drum to CM), O designates Output (CM to drum), and B designates Binary. Each of these forms must be followed by 3 program parameters as described below and in the order listed:

(1) Initial address of Core Memory (CM)
(2) Initial address of Drum Memory (DM)
    (Drum registers available are 2,048 - 22,527)
(3) Length of block being transferred to or from CM.

Example:

```
iDIB
+900
+12280
+42
```

This will transfer the contents of registers 12280-12321 of DM to registers 900 - 941 of CM (interpreted return control). If registers 900-941 are double-length numbers, then 21 numbers will be transferred.

TABLE V - OPERATING TIME, IN μ SEC., OF CS II INSTRUCTIONS

	Neither "C" nor "b" blocks used	"C" block used	"b" block used	"b" & "C" blocks used	Neither "C" nor "b" blocks used minimum*
ica	654	694	718	758	
ics	678	718	742	782	
iad	2007#	2047#	2089#	2129#	850
isu	2030#	2070#	2110#	2150#	880
imr	1441#	1481#	1521#	1561#	
idv	2203#	2227#	2267#	2307#	
isp	465	505	529	569	
iex	1275	1315	1339	1379	
its	901	941	965	1005	
icp	281† 472	321† 512	345† 572	385† 612	
ita	360	384	408	448	
icx	x**	554	x	618	
icd	x	384	x	448	
ici	x	361	x	425	
iti	x	385	x	449	
iat	x	440	x	504	
icr	x	360	x	424	
isp c	x	673	x	673	
idv c	x	2331#	x	2331#	
imr c	x	1580#	x	1580#	
isu c	x	2183#	x	2183#	1016
iad c	x	2159#	x	2159#	990
ics c	x	815	x	815	
ica c	x	791	x	791	
iex c	x	1411	x	1411	
its c	x	1045	x	1045	
IN-OUT	361	401	425	465	
IN-spx	361	401	425	465	
ict	x	434	x	498	
isc	x	851	x	915	
ica b	x	x	974	1014	

TABLE V - (Cont.)

	Neither "C" nor "b" blocks used	"C" block used	"b" block used	"b" & "C" blocks used	Neither "C" nor "b" blocks used minimum*
ics b	x	x	1017	1057	
its b	x	x	983	1023	
iex b	x	x	1001	1041	
iad b	x	x	2068#	2108#	
isu b	x	x	2184#	2224#	
imr b	x	x	1716#	1756#	
idv b	x	x	2464#	2504#	

\# average for operating on positive and negative numbers

* in addition the addend and augend, and in subtraction the mimuend and subtrahend do not affect each other because of the great disparity in magnitude

** x indicates that the instruction doesn't have any meaning in that column