MICRO-PROGRAMMING MANUAL

8 Jan 1971

TABLE OF CONTENTS

DEFINITIONS

GENERAL DEFINITIONS

COMPUTER- IN THE FOLLOWING DEFINITIONS, WE SHALL USE THE TERM "COMPUTER", MEANING "THE PHYSICAL UNIT, HARDWARE", ONLY.

COMPUTER PROGRAM- A COMPLETE AND SEQUENCED GROUP OF INSTRUCTIONS, NECESSARY TO SOLVE A PROBLEM BY COMPUTER. REFERRED TO AS A "PROGRAM".

B1500 DEFINITIONS

S-SOURCE PROGRAM- A COMPUTER PROGRAM, CONSISTING OF A SET OF INSTRUCTIONS, FROM WHICH A TRANSLATOR (GENERATOR, ASSEMBLER, OR COMPILER) GENERATES AN S-OBJECT PROGRAM.

S-OBJECT PROGRAM- A COMPUTER PROGRAM, CONSISTING OF A SET OF INSTRUCTIONS. THE INSTRUCTIONS ARE MADE ENTIRELY OF CHARACTERS, CAPABLE OF REPRESENTING THE "PLUS-MINUS" LOGIC OF A COMPUTER WITHOUT TRANSLATION, USUALLY THE BINARY NUMBERS 0 AND 1.

PRE-B1500 PROCEDURE- IN THE FIELD, AN APPLICATIONS PROGRAMMER WOULD CODE AN S-SOURCE PROGRAM, A SEQUENCED GROUP OF INSTRUCTIONS WRITTEN IN SOME PRE-DEFINED LANGUAGE (FORTRAN, COBOL, ETC.), TO SOLVE A SPECIFIC PROBLEM. BY MEANS OF THE APPROPRIATE TRANSLATOR AN S-OBJECT PROGRAM WOULD BE GENERATED. THE S-OBJECT PROGRAM, WOULD THEN BE DIRECTLY INTERPRETED AND USED BY THE COMPUTER (HARDWARE).

M- ACRONYM FOR MICRO.

M-SOURCE PROGRAM- A COMPUTER PROGRAM, CONSISTING OF A SET OF M-INSTRUCTIONS, FROM WHICH THE MIL COMPILER GENERATES AN M-OBJECT PROGRAM.

M-OBJECT PROGRAM- A COMPUTER PROGRAM, CONSISTING OF A SET OF INSTRUCTIONS. THE INSTRUCTIONS ARE MADE ENTIRELY OF THE BINARY NUMBERS 0 AND 1. THE M-OBJECT PROGRAM IS CAPABLE OF BEING USED DIRECTLY, WITHOUT TRANSLATION, BY THE B1500 COMPUTER (HARDWARE).

B1500 PROCEDURE- WITH THE B1500 COMPUTER AN ADDITIONAL LEVEL OF PROGRAMMING HAS BEEN INTRODUCED, THE "M-LEVEL". THE TASK OF ALL M-PROGRAMS IS TO INTERPRET S-OBJECT PROGRAMS, THUS PLACING PREVIOUS HARDWARE TASKS IN THE HANDS OF THE SYSTEMS PROGRAMMER. WITH THE B1500, AN APPLICATIONS PROGRAMMER WOULD CODE AN S-SOURCE PROGRAM; BY MEANS OF THE APPROPRIATE TRANSLATOR AN S-OBJECT PROGRAM WOULD BE GENERATED, AND THEN INTERPRETED BY THE APPROPRIATE "M-OBJECT PROGRAMS". THUS THE "M-LEVEL" ACTS AS A LINK BETWEEN S-OBJECT PROGRAMS AND THE COMPUTER HARDWARE.

B1500 DEFINITIONS(CONTINUED)

M-SOURCE INSTRUCTION- AN M-OPERATOR, KEYWORDS, OPERAND OR OPERANDS, AND CONSTANTS ARRANGED IN A PRE-DEFINED PATTERN.

M-OBJECT INSTRUCTION- A PRE-DEFINED PATTERN OF BINARY NUMBERS 0 AND 1, DIRECTLY USED BY B1500 HARDWARE WITHOUT TRANSLATION. USUALLY GENERATED BY THE MIL COMPILER FROM M-SOURCE INSTRUCTIONS.

M-OPERATOR- THE ACTION WORDS OR VERBS OF AN M-INSTRUCTION.

KEYWORDS- PREPOSITIONS (FROM, TO, BY, WITH) USED FOR CLARIFICATION IN M-SOURCE INSTRUCTIONS.

OPERAND- THAT PORTION OF AN INSTRUCTION THAT IS ACTED UPON BY THE M-OPERATORS. REGISTERS, SCRATCHPADS, A-STACK, M-MEMORY.

REGISTERS

REGISTER GROUP AND SELECT

	SELECT 0	SELECT 1	SELECT 2	SELECT 3
GROUP 0	TA	FU	X	SUM
GROUP 1	TB	FLB	Y	CMPX
GROUP 2	TC	FLC	T	CMPY
GROUP 3	TD	FLD	L	XANY
GROUP 4	TE	FLE	A	XEOY
GROUP 5	TF	FLF	M	MSKX
GROUP 6	CA	BICN	BR	MSKY
GROUP 7	CB	FLCN	LR	XORY
GROUP 8	LA	MC	FA	DIFF
GROUP 9	LB	MD	FB	MAXS
GROUP 10	LC	ME	FL	MAXM
GROUP 11	LD	MF	TAS	U
GROUP 12	LE	XYCN	CP	
GROUP 13	LF	XYST	MSMA	
GROUP 14	CC	INCN		
GROUP 15	CD	CPU		NULL

NOTES:

1) BICN, FLCN, XYCN, INCN AND ALL SELECT 3 REGISTERS ARE SOURCE REGISTERS ONLY. THEREFORE, THEY MAY NOT BE USED AS SINK (DESTINATION) REGISTERS.

2) CPU IS A SINK REGISTER ONLY.

3) NULL ALWAYS CONTAINS A VALUE OF ZERO. ANY REGISTER OR SCRATCHPAD WORD TO WHICH IT IS MOVED WILL BE CLEARED TO ZERO.

ALPHABETICAL LIST OF REGISTERS

NAME	LENGTH (BITS)	NOTE
A	20	MICRO-MEMORY ADDRESS
BICN	4	BIT CONDITIONS
BR	24	BASE REGISTER
CA	4	-HIGH SUBFIELD OF C
CB	4	- SUBFIELD OF C
CC	4	- SUBFIELD OF C, INTERRUPTS AND FLAGS
CD	4	- SUBFIELD OF C, INTERRUPTS AND FLAGS
CMPX	24	RESULT: COMPLEMENT OF X; MASKED
CMPY	24	RESULT: COMPLEMENT OF Y; MASKED
CP	8	- SUBFIELD OF C
CPU	2	- SUBFIELD OF CP; SINK ONLY
DIFF	24	RESULT: X MINUS (Y + CYF); MASKED
FA	24	S-MEMORY ADDRESS
FB	24	S-MEMORY UNIT AND LENGTH
FL	20	- SUBFIELD OF FB; S-MEMORY LENGTH
FLB	4	-HIGH SUBFIELD OF FL
FLC	4	- SUBFIELD OF FL
FLD	4	- SUBFIELD OF FL
FLE	4	- SUBFIELD OF FL
FLF	4	-LOW SUBFIELD OF FL
FLCN	4	FL CONDITIONS
FU	4	- SUBFIELD OF FB; S-MEMORY UNIT
INCN	4	DISPATCH INTERRUPT CONDITIONS
L	24	IMPLICIT USE IN DISPATCH AND OVERLAY
LA	4	-HIGH SUBFIELD OF L
LB	4	- SUBFIELD OF L
LC	4	- SUBFIELD OF L
LD	4	- SUBFIELD OF L
LE	4	- SUBFIELD OF L
LF	4	-LOW SUBFIELD OF L

ALPHABETICAL LIST OF REGISTERS(CONTINUED)

NAME	LENGTH (BITS)	NOTE
LR	24	LIMIT REGISTER
M	16	CONTAINS CURRENT MICRO-INSTRUCTION
MAXM	24	CONSTANT=NUMBER OF M-MEMORY WORDS AVAILABLE
MAXS	24	CONSTANT=NUMBER OF S-MEMORY BITS AVAILABLE
MC	4	-HIGH SUBFIELD OF M
MD	4	- SUBFIELD OF M
ME	4	- SUBFIELD OF M
MF	4	-LOW SUBFIELD OF M
MSKX	24	RESULT: MASK X; MASKED
MSKY	24	RESULT: MASK Y; MASKED
MSMA	16	M-MEMORY WORD POINTED TO BY A; MTR MODE ONLY
NULL	24	ALWAYS ZERO.
SUM	24	RESULT: X + Y + CYF; MASKED
T	24	ROTATE, SHIFT, EXTRACT BITS, ETC.
TAS	24	TOP OF ASTACK
TA	4	-HIGH SUBFIELD OF T
TB	4	- SUBFIELD OF T
TC	4	- SUBFIELD OF T
TD	4	- SUBFIELD OF T
TE	4	- SUBFIELD OF T
TF	4	- SUBFIELD OF T
U	16	CASSETTE INPUT ONLY
X	24	INPUT TO FUNCTION BOX
XANY	24	RESULT: X AND Y ; MASKED
XEOY	24	RESULT: X EXCLUSIVE OR Y; MASKED
XORY	24	RESULT: X OR Y; MASKED
XYCN	4	XY CONDITIONS
XYST	4	XY STATUS
Y	24	INPUT TO FUNCTION BOX

NOTE: THE MOST SIGNIFICANT(LEFT MOST) BIT IN ANY REGISTER IS IDENTIFIED IN THE MIL SYNTAX AS BIT 0(ZERO), THE NEXT MOST SIGNIFICANT AS BIT 1, ETC. THIS IS PARTICULARLY ADVANTAGEOUS IN A BIT ADDRESSABLE MACHINE, SINCE, FOR SOFTWARE PURPOSES, IT IS OFTEN DESIRABLE TO THINK OF A REGISTER AS BEING AN EXTENSION OF MAIN MEMORY. PLEASE NOTE THAT THIS CONVENTION IS AT VARIANCE WITH THE HARDWARE BIT NUMBERING CONVENTION WHERE, GENERALLY, ALL BITS ARE NUMBERED RIGHT TO LEFT, 0 THRU 23. THIS DIFFERENCE HAS PARTICULAR SIGNIFICANCE WHEN ANY PORTION OF A MICRO-OPERATOR IS TO BE "OR-ED" INTO THE M REGISTER AT RUN TIME.

SCRATCHPAD WORDS

THE M-PROCESSOR MAKES USE OF THE SCRATCH PAD FOR TEMPORARY STORAGE OF ACTIVE REGISTERS. THE SCRATCH PAD WORDS MAY BE ADDRESSED AS 16 FORTY-EIGHT BIT WORDS OR 32 TWENTY-FOUR BIT WORDS.

PAD-WORDS - 24 BITS EACH

S0A	S4A	S8A	S12A
S0B	S4B	S8B	S12B
S1A	S5A	S9A	S13A
S1B	S5B	S9B	S13B
S2A	S6A	S10A	S14A
S2B	S6B	S10B	S14B
S3A	S7A	S11A	S15A
S3B	S7B	S11B	S15B

DOUBLE PAD-WORDS - 48 BITS EACH

(S@ = S@A AND S@B CONCATENATED, WHERE @ = 0 THRU 15)

S0	S4	S8	S12
S1	S5	S9	S13
S2	S6	S10	S14
S3	S7	S11	S15

SPECIAL REGISTER TERMS

TERM	BITS	NOTE
F	48	FA AND FB CONCATENATED
XY	48	X AND Y CONCATENATED
SO	48	SOA AND SOB, DOUBLE SCRATCHPAD WORD
CP	8	CYF,CPU,CPL COLLECTIVELY
CYF	1	CARRY FLIP-FLOP, PART OF CP
CPU	2	INDICATES UNIT, PART OF CP
CPL	5	LENGTH OF X AND Y, PART OF CP
FU	4	INDICATES UNIT SIZE
SU	4	UNIT, PART OF SOB CORR. TO FU IN FB
CYL	1	CARRY LEVEL
CYD	1	CARRY DIFFERENCE
FL	20	S-MEMORY ACCESS LENGTH
SFL	20	PART OF SOB CORRESPONDING TO FL IN FB

ORGANIZATION OF REGISTER FIELDS AND SUBFIELDS

FIELD	SUBFIELDS
C	CA CB CC CD CP
CP	CPU
F	FA FB
FB	FU FL
FL	FLB FLC FLD FLE FLF
L	LA LB LC LD LE LF
M	MC MD ME MF
T	TA TB TC TD TE TF

NOTE: C DOES NOT EXIST AS COMPOSITE; ONLY AS SUBFIELDS.

CONTROL REGISTERS

MICRO-INSTRUCTION CONTROLS

A
M
TAS

S-MEMORY CONTROLS

BR
LR
FA
CP

INTERRUPT CONTROLS

CC
CD

VARIABLES

A
BR
CA CB CC CD CP CPU
FA
FB
FL
FLB FLC FLD FLE FLF
FU
L
LA LB LC LD LE LF
LR
M
MC MD ME MF
MSMA
T
TA TB TC TD TE TF
X
Y
U

ACTIVE REGISTERS

X-Y REGISTERS

THE X AND Y REGISTERS ARE EACH 24 BITS WIDE AND ARE USED AS THE BUFFERS INTO THE X-Y FUNCTION BOX. ALL FUNCTIONS ARE PERFORMED UNDER CONTROL OF THE C REGISTER WHICH CONTAINS THE LENGTH OF OPERATION, THE TYPE OF UNIT, AND LEAST SIGNIFICANT CARRY INPUT. IN ADDITION THE X AND Y REGISTERS ARE CAPABLE OF BEING SHIFTED OR ROTATED AND MAY RECEIVE OR TRANSMIT DATA TO THE MAIN MEMORY SYSTEM.

F REGISTER

THE F REGISTER IS DIVIDED INTO FA AND FB, EACH 24 BITS WIDE. THE FA PORTION IS USED TO ADDRESS MAIN MEMORY. FB IS FURTHER DIVIDED INTO FU, 4 BITS INDICATING UNIT SIZE, AND FL, 20 BITS GENERALLY USED TO INDICATE LENGTH OF FIELDS IN MAIN MEMORY. FL IS FURTHER SUBDIVIDED INTO FLB, FLC, FLD, FLE, AND FLF, EACH BEING 4 BITS LONG.

L REGISTER

THE L REGISTER IS 24 BITS WIDE AND SUBDIVIDED INTO LA, LB, LC, LD, LE, AND LF, EACH 4 BITS IN LENGTH. L AND ITS SUBDIVISIONS ARE GENERALLY USED TO HOLD TEMPORARILY THE CONTENTS OF OTHER M-PROCESSOR REGISTERS. IT IS ALSO USED AS A SOURCE AND SINK FOR MEMORY ACCESS AND HAS IMPLICIT USE IN DISPATCH AND OVERLAY.

T REGISTER

THE T REGISTER IS A 24 BIT TRANSFORMATION REGISTER USED EXTENSIVELY FOR INTERPRETATION OF OPERATORS. IT IS SUBDIVIDED INTO TA, TB, TC, TD, TE, TF, EACH OF 4 BITS. T HAS STRONG SHIFT AND EXTRACT LOGICS ASSOCIATED WITH IT AND IS THE PRINCIPAL FORMATTING REGISTER OF THE M-PROCESSOR. THE REGISTER ALSO HAS THE CAPABILITY OF RECEIVING OR TRANSMITTING DATA WITH MAIN MEMORY.

M REGISTER

THE M REGISTER IS A 16 BIT REGISTER WHICH HOLDS THE MICRO- OPERATOR. AS SUCH, IT IS THE MACHINE INSTRUCTION REGISTER. IT IS FURTHER DIVIDED INTO MC, MD, ME, AND MF, EACH OF 4 BITS.

ACTIVE REGISTERS(CONTINUED)

BR AND LR REGISTERS

THE BR AND LR REGISTERS ARE EACH 24 BITS WIDE AND USED TO HOLD THE BASE AND LIMIT ADDRESS OF THE ACTIVE PROCESS WORK SPACE. THE HARDWARE USES THESE REGISTERS TO DETERMINE IF ADDRESSES IN THE FA REGISTER ARE WITHIN THE BASE/LIMIT BOUNDRIES OF MAIN MEMORY.

A REGISTER

THE A REGISTER IS A 20 BIT ADDRESS REGISTER USED TO FETCH M INSTRUCTIONS TO THE M REGISTER FOR EXECUTION. THE ADDRESS IN A IS A WORD ADDRESS WHERE A WORD, IN THIS CONTEXT ONLY, IS DEFINED TO BE 16 BITS IN LENGTH. AUTOMATIC INCREMENTATION IS PROVIDED.

C REGISTER

THE C REGISTER IS A 24 BIT CONTROL REGISTER FOR THE PROCESSOR. IT CONTAINS THE 24 BIT FUNCTION BOX CONTROLS AND CARRY INPUT AND ALSO SOME PROCESSOR INTERRUPTS AND FLAGS. IT IS SUBDIVIDED INTO CA, CB, CC, CD EACH 4 BITS, AND CP, 8 BITS. CA AND CB (8 BITS) ARE TEMPORARILY OPEN. CC AND CD (8 BITS) REPRESENT M-PROCESSOR INTERRUPTS AND FLAGS. CP (8 BITS) CONTAINS FUNCTION BOX CONTROLS; CYF (0 BIT OF CP), CPU (1 AND 2 BITS OF CP), AND CPL (3,4,5,6,7 BITS OF CP). CYF NOTIFIES THE FUNCTION BOX THAT A PREVIOUS UNIT CARRY MUST BE ADDED TO ITS SUMMARY RESULTS. CPL NOTIFIES THE FUNCTION BOX OF THE LENGTH IN BITS OF THE ITEMS IN X AND Y. CPU NOTIFIES THE FUNCTION BOX OF THE TYPE OF UNITS CONTAINED IN X AND Y; 00 - BINARY, 01 - 4 BIT DECIMAL, 10 - 6 BIT DECIMAL, AND 11 - 8 BIT DECIMAL.

RESULT REGISTERS

XORY	XANY	XEOY
CMPX	CMPY	MSKX
MSKY	SUM	DIFF

THE RESULT REGISTERS ARE 24 BIT OUTPUTS OF THE 24 BIT FUNCTION BOX. THEIR CONTENTS ARE PRODUCED IMMEDIATELY AND AUTOMATICALLY FROM THE INPUTS TO THE FUNCTION BOX(X, Y AND CYF) AND CANNOT BE CHANGED EXCEPT BY CHANGING THESE INPUTS OR BY CHANGING CPU OR CPL. IF THE VALUE OF CPL IS LESS THAN 24, THEN THE 24 MINUS CPL MOST SIGNIFICANT BIT POSITIONS OF ALL RESULT REGISTERS ARE SET TO ZERO, E.G. IF CPL = 16, THEN THE 8 MOST SIGNIFICANT BITS OF ALL RESULT REGISTERS WILL BE ZERO. THESE REGISTERS ARE SOURCE REGISTERS ONLY AND THEREFORE MUST NOT BE USED AS THE SINK (DESTINATION) REGISTER IN A "MOVE" OR IN ANY OTHER INSTRUCTION.

XORY

"OR" X INCLUSIVELY WITH Y. THIS IS A BIT BY BIT OPERATION, WITH CORRESPONDING PAIRS OF BITS BEING TREATED INDEPENDENTLY.

XANY

"AND" X WITH Y. THIS IS THE LOGICAL PRODUCT OF X AND Y. CORRESPONDING PAIRS OF BITS ARE TREATED INDEPENDENTLY.

XEOY

"OR" X EXCLUSIVELY WITH Y, MEANING THAT THE SUM OF X, Y, AND XEOY IS MODULO 2.

CMPX

THE ONES COMPLEMENT OF X.

CMPY

THE ONES COMPLEMENT OF Y.

RESULT REGISTERS(CONTINUED)

MSKX

"MASKED X" IS THE LOW ORDER CPL BITS OF X. IF CPL IS EQUAL TO 24, THEN MASKED X EQUALS X.

MSKY

"MASKED Y" IS THE LOW ORDER CPL BITS OF Y. IF CPL IS EQUAL TO 24, THEN MASKED Y EQUALS Y.

SUM

"SUM". THE DECIMAL OR BINARY ADDITION (DETERMINED BY CPU) OF X PLUS Y PLUS CYF. CORRESPONDING PAIRS OF BITS ARE GROUPED BY CPU. GROUPING MAY BE BINARY, DECIMAL FOUR BIT, SIX BIT, OR EIGHT BIT. IF THE SUM OF X+Y+CYF IS LARGER THAN THE SIZE SPECIFIED BY CPL, THEN "CYL" WILL BE ON (=1). THE ZONE BITS ARE "OR-ED" TOGETHER FOR SIX AND EIGHT BIT DECIMAL.

DIFF

RESULT OF Y + CYF SUBTRACTED FROM X, EITHER DECIMAL OR BINARY SUBTRACTION, ACCORDING TO CPU. CORRESPONDING PAIRS OF BITS ARE GROUPED BY CPU. IF THE DIFFERENCE IS NEGATIVE, THAT IS IF X IS LESS THAN Y + CYF, THEN DIFF WILL BE IN TWOS COMPLEMENT FORM OR TENS COMPLEMENT FORM DEPENDING UPON THE MODE,EITHER BINARY OR DECIMAL; AND CYD WILL BE ON(=1). THE ZONE BITS ARE "OR-ED" TOGETHER FOR SIX AND EIGHT BIT DECIMAL. NOTE THAT CYD IS NOT CONDITIONED BY CPL; IT IS ALWAYS BASED ON A 24 BIT COMPARE.

CONDITION REGISTERS

BICN FLCN INCN XYCN XYST

THERE ARE FIVE CONDITION REGISTERS, EACH CONTAINING FOUR BITS. THE BITS ARE IDENTIFIED FROM LEFT TO RIGHT AND ARE ASSIGNED THE POSITION NUMBERS 0 THRU 3, WITH BIT 0 BEING THE 8 OR MOST SIGNIFICANT BIT.

ALL CONDITION REGISTERS ARE SOURCE REGISTERS ONLY. THEY MAY BE MOVED TO ANOTHER REGISTER OR TESTED, USING THE "IF" AND "SKIP" INSTRUCTIONS, FOR THEIR CURRENT CONTENTS; BUT THEY MAY NOT BE THE SINK(DESTINATION) REGISTER OF ANY INSTRUCTION.

BIT	B1CN	XYCN	XYST	FLCN	INCN
0	LSUY	MSBX	LSUX	FL=SFL	NO-DEVICE
1	CYF	X=Y	[illegible]	FL>SFL	HI-PRIORITY
2	CYD	X<Y	Y≠0	FL<SFL	INTERRUPT
3	CYL	X>Y	X≠0	FL≠0	LOCKOUT

BICN - BIT CONDITIONS

LSUY IS TRUE IF THE LEAST SIGNIFICANT UNIT OF Y IS 1 FOR BINARY(CPU=0) OR 9 FOR DECIMAL(CPU NOT = 0).

CYF IS TRUE IF CYF, THE CARRY FLIP-FLOP, IN THE CP REGISTER IS TRUE. CYF MAY BE MANIPULATED AS PART OF THE CP REGISTER AND BY THE "CARRY" INSTRUCTION.

CYD, CARRY DIFFERENCE, IS TRUE IF X IS LESS THAN Y PLUS CYF. THIS CONDITION IS NOT AFFECTED BY CPL, I.E. A 24 BIT COMPARE IS ALWAYS MADE.

CYL, CARRY LEVEL, IS TRUE IF THE SUM OF X + Y + CYF, LIMITED BY CPL, OVERFLOWS.

CONDITION REGISTERS(CONTINUED)

XYCN - XY CONDITIONS

MSBX IS TRUE IF THE MOST SIGNIFICANT BIT OF X, AS DETERMINED BY CPL, IS A "1".

THE COMPARISONS OF X TO Y ARE NOT AFFECTED BY CPL; THEY ARE ALWAYS 24 BIT COMPARES.

XYST - XY STATES

LSUX IS TRUE IF THE LEAST SIGNIFICANT UNIT OF X IS 1 FOR BINARY(CPU=0) OR 9 FOR DECIMAL(CPU NOT = 0).

THE COMPARISONS OF X AND/OR Y TO ZERO ARE NOT AFFECTED BY CPL; ALL 24 BITS OF X AND/OR Y ARE USED IN THE COMPARISONS.

FLCN - FIELD LENGTH CONDITIONS

ALL CONDITIONS ARE BASED ON COMPARISONS BETWEEN THE 20 BITS OF THE FL REGISTER AND ZERO OR THE CORRESPONDING LOW ORDER 20 BITS(SFL) OF THE FIRST WORD IN SCRATCHPAD, SOB.

INCN - INTERRUPT CONDITIONS

NO-DEVICE IS TRUE IF AN INTERRUPT MESSAGE IS PRESENT IN THE DISPATCH BUFFER FOR A PORT OR CHANNEL WHICH DOES NOT HAVE A DEVICE ATTACHED TO IT. THIS CONDITION IS NORMALLY CLEARED BY THE PROCESSOR WITH A "DISPATCH READ AND CLEAR".

HI-PRIORITY IS TRUE IF THERE IS A HIGH PRIORITY MESSAGE PRESENT IN THE DISPATCH BUFFER. THE MESSAGE IS NOT NECESSARILY FOR THE PROCESSOR.

INTERRUPT IS TRUE IF THERE IS A MESSAGE PRESENT IN THE DISPATCH BUFFER FOR THE PROCESSOR. THIS CONDITION IS NORMALLY CLEARED BY THE PROCESSOR WITH A "DISPATCH READ AND CLEAR".

LOCKOUT IS TRUE IF THE INTERRUPT SYSTEM IS LOCKED(MARKED AS IN USE).

CC AND CD INTERRUPTS AND FLAGS

THE CC AND CD REGISTERS ARE BOTH 4 BIT SOURCE AND SINK REGISTERS. THE BITS IN EACH ARE NUMBERED 0 THRU 3 WITH BIT 0 BEING THE MOST SIGNIFICANT OR 8 BIT. THEY HAVE BEEN ASSIGNED THE FOLLOWING USES AND MEANINGS:

CC(0) TEMPORARILY OPEN.

CC(1) TEMPORARILY OPEN.

CC(2) TEMPORARILY OPEN.

CC(3) HALT INTERRUPT - SET WHEN THE HALT BUTTON ON THE CONSOLE IS PUSHED.

CD(0) MEMORY PARITY ERROR INTERRUPT - SET WHEN AN S-MEMORY PARITY ERROR IS DETECTED DURING A READ OR SWAP OPERATION OR WHEN AN ATTEMPT IS MADE TO ACCESS NON-EXISTENT MEMORY.

CD(1) MEMORY ADDRESS OUT OF BOUNDS OVERRIDE - IF THE ADDRESS IN FA IS LESS THAN THE BASE REGISTER(BR) SETTING OR GREATER THAN THE LIMIT REGISTER(LR) SETTING, THEN READ, WRITE AND SWAP OPERATIONS WILL BE INHIBITED UNLESS THIS BIT IS SET(=1). THE STATE OF THIS BIT DOES NOT AFFECT THE SETTING OF CD(2) OR CD(3).

CD(2) READ ADDRESS OUT OF BOUNDS INTERRUPT - SET WHEN A READ OPERATION IS ATTEMPTED AND THE ADDRESS IN FA IS EITHER LESS THAN THE BASE REGISTER OR GREATER THAN THE LIMIT REGISTER.

CD(3) WRITE/SWAP ADDRESS OUT OF BOUNDS INTERRUPT - SET WHEN A WRITE OR SWAP OPERATION IS ATTEMPTED AND THE ADDRESS IN FA IS EITHER LESS THAN THE BASE REGISTER OR GREATER THAN THE LIMIT REGISTER.

ALL BITS IN THE CC AND CD REGISTERS, ONCE SET, WILL REMAIN SET EVEN THOUGH THE CONDITIONS THAT CAUSED THEM TO BE SET MAY NO LONGER EXIST. THEREFORE, IF IT IS DESIRED TO CLEAR ANY OF THESE BITS TO ZERO, THIS MUST BE DONE EXPLICITLY.

THE MICRO IMPLEMENTATION LANGUAGE

CONCURRENT EXECUTION OF MICRO-OPERATORS

CONCURRENT OR OVERLAPPING EXECUTION OF MICRO-OPERATORS CAN BE ACHIEVED IF A "READ", "WRITE" OR "SWAP" INSTRUCTION IS FOLLOWED BY ANY OF THE FOLLOWING INSTRUCTIONS:

BIAS
COUNT
GO TO
JUMP
LOAD
NOP
READ
STORE
SWAP
WRITE
XCH

RESERVED WORDS

THERE ARE NO RESERVED WORDS AS FAR AS LABELS ARE CONCERNED; HOWEVER, NO WORD USED IN THE MIL SYNTAX MAY BE USED AS A DEFINE OR MACRO INDENTIFIER OR AS A MODULE OPTION TOGGLE.

LITERALS

WHEREVER LITERAL-1 APPEARS IN THE TEXT OF THIS MANUAL, IT IS TO BE REPLACED BY EITHER A DECIMAL INTEGER, A HEXADECIMAL NUMBER WHOSE FIRST CHARACTER IS AN IDENTIFYING "H", OR BY A BINARY NUMBER WHOSE FIRST CHARACTER IS AN IDENTIFYING "B".

CHARACTERS WHICH ARE VALID FOR USE IN A LITERAL ARE DEPENDENT UPON THE TYPE OF THE LITERAL.

TYPE	VALID CHARACTERS
DECIMAL	NUMERIC 0 THRU 9
HEXADECIMAL	NUMERIC 0 THRU 9 AND ALPHA A, B, C, D, E AND F, REPRESENTING RESPECTIVELY 10, 11, 12, 13, 14 AND 15
BINARY	NUMERIC 0 AND 1

IN GENERAL, LITERALS USED IN THE MIL SYNTAX HAVE A MAXIMUM RANGE OF DECIMAL 0 THRU 16777215 WHICH IS EQUAL TO HEXADECIMAL H0 THRU HFFFFFF. A BINARY LITERAL, HOWEVER, MAY NOT CONTAIN MORE THAN 21 CHARACTERS. THEREFORE, B0 THRU B111111111111111111111 IS THE MAXIMUM RANGE OF A BINARY LITERAL.

LEADING ZEROS ARE NEVER REQUIRED FOR LITERALS UNLESS AN ACTUAL VALUE OF ZERO IS USED, THEN EITHER "0", "H0" OR "B0" MUST BE USED.

NO CHARACTER MAY BE EMBEDDED BETWEEN THE IDENTIFYING "H" OR "B" OF A HEXADECIMAL OR BINARY LITERAL AND THE NUMBER STRING THAT FOLLOWS. NO CHARACTERS, SUCH AS COMMAS, MAY BE EMBEDDED IN ANY LITERAL.

CONDITIONS

WHEREVER CONDITION-1 APPEARS IN THE TEXT, IT IS TO BE REPLACED BY ANY CONDITION WHOSE TRUTH OR FALSITY CAN BE DETERMINED BY TESTING ONE OR MORE BITS IN ONE OF THE CONDITION REGISTERS. THEREFORE, ANY OF THE FOLLOWING MAY BE USED IN THE "IF" AND "SKIP" STATEMENTS:

```
X =/</>/≤/≥/≠/EQL/LSS/GTR/LEQ/GEQ/NEQ Y
X =/≠/EQL/NEQ 0
Y =/</>/≤/≥/≠/EQL/LSS/GTR/LEQ/GEQ/NEQ X
Y =/≠/EQL/NEQ 0
XY =/≠/EQL/NEQ 0
FL =/</>/≤/≥/≠/EQL/LSS/GTR/LEQ/GEQ/NEQ SFL
FL =/≠/EQL/NEQ 0
CYL
CYD
LSUY
MSBX
LSUX
LOCKOUT
INTERRUPT
HI-PRIORITY
NO-DEVICE
```

IN ADDITION, ANY COMBINATION OF CONDITIONS THAT ARE ALL IN ONE CONDITION REGISTER CAN BE TESTED USING AND/OR LOGIC IF ALL BITS TESTED CAN BE TESTED FOR "TRUE" OR ALL BITS TESTED CAN BE TESTED FOR "FALSE". FOR EXAMPLE, THE FOLLOWING ARE ALL VALID CONDITIONS:

```
CYL AND LSUY
CYL OR CYD
X LSS Y AND MSBX
X NEQ 0 OR Y NEQ 0
FL NEQ 0 AND FL EQL SFL
INTERRUPT OR HI-PRIORITY OR NO-DEVICE
```

HOWEVER, NONE OF THE FOLLOWING ARE VALID CONDITIONS:

```
X GEQ Y AND MSBX          (THE COMPILER MUST GENERATE A TEST ON
                           X LSS Y FALSE)
X NEQ 0 OR Y EQL 0        (Y NEQ 0 MUST BE TESTED FOR FALSE)
FL EQL 0 AND FL EQL SFL   (FL NEQ 0 MUST BE TESTED FOR FALSE)
```

AT LEAST FOR THE TIME BEING, CONDITIONS IN WHICH MORE THAN ONE BIT MUST BE TESTED FOR "FALSE", SUCH AS "X EQL 0 AND Y EQL 0" AND "FL EQL 0 AND FL NEQ SFL" ARE NOT VALID. REST ASSURED THERE MUST BE SOME GOOD REASON FOR THIS.

NOTE: TO FACILITATE RUNNING ON OTHER MACHINES IT IS ADVISED TO USE THE MNEMONIC FORM OF THE RELATIONAL OPERATOR.

SOURCE CARD LAYOUT

THE SOURCE PROGRAM CARDS HAVE THE FOLLOWING FORMAT:

CARD COLUMNS

1 THRU 5	RESERVED FOR LABEL DECLARATIONS WHICH, IF USED, MUST BEGIN ANYWHERE WITHIN THIS FIELD.
1 THRU 72	AN ASTERISK(*) ANYWHERE WITHIN THIS FIELD INDICATES THAT EVERYTHING IN THIS FIELD TO THE RIGHT OF THE ASTERISK IS A COMMENT.
6 THRU 72	STATEMENTS MAY BE USED ANYWHERE WITHIN THIS FIELD. AT LEAST ONE BLANK MUST BE USED BETWEEN WORDS EXCEPT IN THOSE CASES WHERE A SPECIAL CHARACTER, E.G. A PARENTHESIS OR A RELATIONAL OPERATOR, IS REQUIRED, IN WHICH CASE BLANKS ARE OPTIONAL, E.G. "EXTRACT 7 BITS FROM T(11) TO Y" AND "SKIP WHEN X>Y".
73 THRU 80	THIS FIELD IS RESERVED FOR SEQUENCE NUMBERS.

NOTE: ONLY ONE MIL SOURCE LANGUAGE CONSTRUCT IS ALLOWED PER CARD.

DECLARATIONS

DEFINES

FORMAT: DEFINE DEFINE-IDENTIFIER-1 = DEFINE-STRING-1#

DESCRIPTION:

THIS DECLARATION IS USED TO ASSIGN A NAME(DEFINE-IDENTIFIER-1) TO A STRING OF CHARACTERS(DEFINE-STRING-1). ANY LATER REFERENCE TO DEFINE-IDENTIFIER-1 IS REPLACED BY DEFINE-STRING-1.

DEFINE-IDENTIFIER-1 MAY BE MADE UP OF ALPHA(A THRU Z) OR NUMERIC(0 THRU 9) CHARACTERS IN ANY COMBINATION, EXCEPT THAT NO MICRO NAME("WRITE", MOVE", ETC.) OR NOISE WORD("TO", "BITS", ETC.) THAT IS USED ELSEWHERE IN THE MIL SYNTAX MAY BE USED AS A DEFINE IDENTIFIER. THE SPECIAL CHARACTER DASH(-) IS ALSO ACCEPTABLE, BUT IT MAY NOT BE USED AS THE FIRST CHARACTER. NO LENGTH RESTRICTION IS IMPOSED, HOWEVER, EVERYTHING AFTER THE TWENTY-FIRST CHARACTER IS CONSIDERED TO BE DOCUMENTATION ONLY.

DEFINE-STRING-1 MAY BE A SCRATCHPAD NAME(24 OR 48 BIT); A REGISTER NAME; A LITERAL; A PART OF ONE INSTRUCTION; AN ENTIRE INSTRUCTION, PART OF WHICH MAY HAVE BEEN PREVIOUSLY "DEFINED"; OR IT MAY BE EMPTY(BLANK). ACTUALLY, THE ONLY RESTRICTIONS PLACED ON A DEFINE STRING ARE THAT IT MUST BE TERMINATED BY A POUND SIGN(#) AND THAT IT MUST NEITHER BEGIN WITH A POUND SIGN NOR CONTAIN ANY EMBEDDED POUND SIGNS.

THE ENTIRE DEFINE DECLARATION MUST BE CONTAINED ON ONE CARD, AND ALL DEFINES MUST BE DECLARED PRIOR TO ANY EXECUTABLE INSTRUCTIONS.

NESTED DEFINES ARE ALLOWED UP TO 13 DEEP.

EXAMPLES:

DECLARATION	REFERENCE
DEFINE SOURCE-POINTER = S3#	LOAD F FROM SOURCE-POINTER
DEFINE OP-REG = L#	CLEAR OP-REG
DEFINE TEST-OP = H800000#	MOVE TEST-OP TO OP-REG
DEFINE HINT = CC(3)#	RESET HINT
DEFINE IGNORE-HALT = RESET HINT#	IGNORE-HALT

MACROS

FORMAT: MACRO MACRO-IDENTIFIER-1 [(FP-1[,FP-2][,FP-N])]=
         STATEMENT-1
         STATEMENT-2
             :
             :
         STATEMENT-N#

DESCRIPTION:

THIS DECLARATION IS USED TO ASSIGN A NAME(MACRO-IDENTIFIER-1) TO A SERIES OF STATEMENTS(THE "MACRO DEFINITION") AND TO DECLARE ANY FORMAL PARAMETERS WHICH MAY BE USED IN THE MACRO DEFINITION. ANY LATER REFERENCE TO MACRO-IDENTIFIER-1 IS REPLACED IN LINE BY THIS SERIES OF STATEMENTS, AND ANY FORMAL PARAMETERS USED IN THESE STATEMENTS ARE REPLACED BY THE ACTUAL PARAMETERS IN THE REFERENCE.

MACRO-IDENTIFIER-1 MAY BE MADE UP OF ALPHA(A THRU Z) OR NUMERIC(0 THRU 9) CHARACTERS IN ANY COMBINATION, EXCEPT THAT NO MICRO NAME("WRITE", MOVE", ETC.) OR NOISE WORD("TO", "BITS", ETC.) THAT IS USED ELSEWHERE IN THE MIL SYNTAX MAY BE USED AS A MACRO IDENTIFIER. THE SPECIAL CHARACTER DASH(-) IS ALSO ACCEPTABLE, BUT IT MAY NOT BE USED AS THE FIRST CHARACTER. NO LENGTH RESTRICTION IS IMPOSED, HOWEVER, EVERYTHING AFTER THE TWENTY-FIRST CHARACTER IS CONSIDERED TO BE DOCUMENTATION ONLY.

FORMAL PARAMETERS(FP-N) ARE OPTIONAL. IF USED, THEY MUST BE ENCLOSED IN PARENTHESES AND EACH HAS THE FORMAT MACRO-IDENTIFIER-1@, WHERE @ IS ORIGINALLY 1 AND IS INCREMENTED BY 1 FOR EACH FORMAL PARAMETER(SEE EXAMPLE ON NEXT PAGE). IF MORE THAN ONE FORMAL PARAMETER IS USED IN THE DECLARATION, THEY MUST BE SEPARATED BY COMMAS.

THE ACTUAL PARAMETERS USED IN THE REFERENCE TO A MACRO MUST BE SINGLE IDENTIFIERS. THIS MEANS THAT THEY MUST NOT CONTAIN ANY EMBEDDED BLANKS OR ANY SPECIAL CHARACTERS. THE ONE EXCEPTION IS THAT AN ACTUAL PARAMETER COULD BE A DEFINE IDENTIFIER AND THEREFORE COULD CONTAIN AN EMBEDDED DASH; HOWEVER, THE DEFINE IDENTIFIER ITSELF WOULD THEN HAVE TO DEFINE A VALID ACTUAL PARAMETER. FOR EXAMPLE, "X", "3", "H801F" AND "TO" ARE VALID ACTUAL PARAMETERS; BUT "3 TO X" AND "(" ARE NOT. NO ACTUAL PARAMETER MAY BE OMITTED; AND, AS WITH FORMAL PARAMETERS, THEY MUST BE ENCLOSED IN PARENTHESES AND SEPARATED BY COMMAS.

MACROS(CONTINUED)

THE MACRO DECLARATION MUST BE ENTIRELY ON ONE LINE AND MUST BE TERMINATED WITH AN EQUAL SIGN(=).

THE MACRO DEFINITION THEN FOLLOWS, ONE STATEMENT PER LINE, AND THE LAST STATEMENT MUST BE TERMINATED BY A POUND SIGN(#). FOR THIS REASON, A MACRO DEFINITION MUST NOT ITSELF CONTAIN A POUND SIGN. A MACRO DEFINITION MAY REFERENCE ANOTHER MACRO OR A DEFINE WHICH HAS BEEN PREVIOUSLY DECLARED, BUT IT MUST NOT BE RECURSIVE.

ALL MACROS MUST BE DECLARED PRIOR TO ANY EXECUTABLE INSTRUCTIONS.

EXAMPLE:

THE DECLARATION--

```
     MACRO WRITEM(WRITEM1, WRITEM2, WRITEM3)=
                  XCH WRITEM1 F WRITEM1
                  WRITE 24 BITS FROM WRITEM2 WRITEM3 FA AND DEC FL
                  XCH WRITEM1 F WRITEM1#
```

WHEN REFERENCED AS--

```
     WRITEM(SO,X,INC)
```

WOULD RESULT IN THE REFERENCE BEING REPLACED BY THE FOLLOWING IN LINE CODE:

```
     XCH SO F SO
     WRITE 24 BITS FROM X INC FA AND DEC FL
     XCH SO F SO
```

LABELS

LABELS ARE DECLARED AS THE FIRST ITEM ON A CARD AND MUST BEGIN ANYWHERE IN COLUMNS 1 THRU 5. ANY NUMBER OF LABELS MAY BE DECLARED FOR THE SAME ADDRESS WITH ONE CARD REQUIRED FOR EACH DECLARATION.

ACCEPTABLE LABEL CHARACTERS ARE ALPHA A THRU Z, NUMERIC 0 THRU 9, AND THE SPECIAL CHARACTER DASH(-). A DASH IS NOT ACCEPTABLE AS THE FIRST CHARACTER OF A LABEL, AND ALL LABELS MUST BE FOLLOWED BY A BLANK OR AN ASTERISK(*).

THERE IS NO LENGTH RESTRICTION EXCEPT THAT THEY MUST NOT EXTEND BEYOND COLUMN 72, AND ANY PART OF A LABEL BEYOND THE TWENTY-FIRST CHARACTER IS CONSIDERED DOCUMENTATION ONLY.

THERE ARE TWO TYPES OF LABELS WHICH ARE ACCEPTABLE TO THE COMPILER - UNIQUE LABELS AND POINT LABELS. UNIQUE LABELS ARE NOT REUSABLE AND THEREFORE MUST BE MADE UNIQUE WITHIN THE FIRST TWENTY-ONE CHARACTERS. POINT LABELS ARE REUSABLE LABELS WHICH ARE USUALLY, BUT NOT NECESSARILY, USED FOR SHORT DISTANCE BRANCHING OF 15 WORDS OR LESS. THEY ARE DECLARED WITH A FIRST CHARACTER OF "." AND ARE REFERENCED EITHER + OR - FROM THE LOCATION OF THE PRESENT INSTRUCTION. THE FOLLOWING IS AN EXAMPLE OF POINT LABEL USAGE:

```
1)    .ABC      SKIP WHEN X NEQ Y
2)        .ABC      GO TO -ABC    * TRAP
3)    .ABC      MOVE SUM TO Y
```

INSTRUCTION 2) IS A BRANCH TO ITSELF; BUT HAD IT BEEN WRITTEN AS "GO TO +ABC", IT WOULD HAVE BEEN A BRANCH TO INSTRUCTION 3).

MICRO-OPERATORS

ADD (BASE RELATE)

FORMAT: ADD BR TO FA

DESCRIPTION:

THIS INSTRUCTION IS USED TO ADD THE BASE REGISTER(BR) TO FA. THE RESULT IS PLACED IN FA AND THE CONTENTS OF THE BASE REGISTER REMAIN UNCHANGED.

NOTE: ALSO SEE "SUBTRACT"

TIME: ONE CLOCK

AND

FORMAT: AND REGISTER-1 WITH [LITERAL-1]
[REGISTER-2]

DESCRIPTION:

THIS INSTRUCTION IS USED TO LOGICALLY BIT "AND" REGISTER-1 WITH LITERAL-1 OR REGISTER-2 AND PLACE THE RESULT IN REGISTER-1.

REGISTER-1 MAY BE ANY 4 BIT REGISTER(SELECT 0 OR 1) EXCEPT A CONDITION REGISTER AND THE CPU, MC, MD, ME AND MF REGISTERS.

REGISTER-2 MAY BE ANY 4 BIT REGISTER(SELECT 0 OR 1) EXCEPT CPU, MC, MD, ME AND MF. THE CONTENTS OF THIS REGISTER ARE NOT CHANGED BY THIS INSTRUCTION.

LITERAL-1 HAS A RANGE OF 0 THRU 15.

"AND" TRUTH TABLE:

```
0 0 = 0
0 1 = 0
1 0 = 0
1 1 = 1
```

EXAMPLE: AND TB WITH 3

```
BEFORE---T = 0AF312
AFTER----T = 02F312
```

TIME: ONE CLOCK UNLESS THE "REGISTER-2" OPTION IS USED, THEN TWO CLOCKS

BIAS

FORMAT: BIAS BY [UNIT] [REGISTER-1] [AND REG-2] [AND REG-3] [TEST]

DESCRIPTION:

SETS CPL AND CPU TO VALUES CALCULATED FROM THE OPERANDS.

IF ONLY REGISTER-1 IS USED IT MUST BE F OR S. IF ONLY REGISTER-1 AND REGISTER-2 ARE USED THEY MAY NOT BE S AND CP, I.E., ONE OF THEM MUST BE F.

REGISTER REFERS TO F, S, OR CP. IDENTICAL INSTRUCTIONS ARE PRODUCED IF THE POSITIONS OF ANY TWO OR THREE REGISTERS ARE INTERCHANGED WITHIN THE INSTRUCTION.

CPU IS SET TO 1, 2, 3, OR 0 DEPENDING ON WHETHER FU IS 4, 6, 8, OR SOME OTHER VALUE, 0-15. THIS IS DONE FOR ALL TEN VARIATIONS OF "BIAS" EXCEPT "BIAS BY S", WHICH SETS CPU FROM SU INSTEAD OF FROM FU.

"BIAS BY ...." SETS CPL EQUAL TO THE VALUE OF THE SMALLER OF 24 AND THE SPECIFIED OPERANDS. "BIAS BY UNIT" SETS CPL EQUAL TO FU.

IF "TEST" IS USED, THE ABOVE ACTIONS ARE PERFORMED AND THE NEXT MICRO-INSTRUCTION IS SKIPPED IF CPL HAS NOT BEEN SET TO ZERO AS THE RESULT OF THE ABOVE ACTIONS. TEST MAY BE USED WITH ALL VARIATIONS OF "BIAS".

EXAMPLE: "BIAS BY S" OR "BIAS BY F"

THIS WILL SET CPL EQUAL TO THE SMALLER OF 24 AND SFL OR FL RESPECTIVELY. CPU IS SET DEPENDING ON THE VALUE OF SFU OR FU RESPECTIVELY.

TIME: ONE CLOCK; TEST OPTION WITH A SKIP: TWO CLOCKS

BRANCH

FORMAT: BRANCH

DESCRIPTION:

THIS INSTRUCTION IS USED TO CAUSE THE COMPILER TO GENERATE A "GO TO" WITH AN ADDRESS OF ZERO. THIS IS TO FACILITATE "OR-ING" THE ACTUAL ADDRESS INTO THE M REGISTER PRIOR TO THE EXECUTION OF A "BRANCH"("GO TO") INSTRUCTION.

TIME: TWO CLOCKS

CALL

FORMAT: CALL LABEL-1

DESCRIPTION:

DOES A STACK PUSH.

SAVES THE ADDRESS OF THE NEXT M-INSTRUCTION IN THE A-STACK (TAS), THEN BRANCHES TO LABEL-1 IN THE M-PROGRAM.

THE "EXIT" INSTRUCTION CAUSES RETURN.

TIME: TWO CLOCKS

CARRY

FORMAT:

```
                [0]
                [1]
CARRY           [SUM]
                [DIFFERENCE]
```

DESCRIPTION:

SETS THE CARRY FLIP-FLOP (CYF) TO EITHER "0" (ZERO) OR "1" (ONE).

"CARRY 0" OR "CARRY 1" SET CYF TO "0" OR "1" RESPECTIVELY.

"CARRY SUM" SETS CYF TO THE VALUE OF CYL.

"CARRY DIFFERENCE" SETS CYF TO THE VALUE OF CYD, I.E. "0" IF X IS GREATER THAN Y OR IF X = Y AND CYF = 0; "1" IF X IS LESS THAN Y OR IF X = Y AND CYF = 1. NOTE THAT CYD, UNLIKE CYL, IS NOT CONDITIONED BY CPL.

TIME: ONE CLOCK

CASSETTE

FORMAT: CASSETTE [START] [STOP [WHEN X NEQ Y]]

DESCRIPTION:

THIS INSTRUCTION IS USED TO CAUSE THE SYSTEM CASSETTE TAPE TO START A READ OPERATION OR TO STOP(EITHER UNCONDITIONALLY OR WHEN THE X REGISTER IS NOT EQUAL TO THE Y REGISTER) AT THE NEXT INTER-RECORD GAP.

THE INFORMATION READ FROM THE CASSETTE IS LOADED INTO THE U REGISTER AND REMAINS THERE FOR A MAXIMUM OF 2 CLOCKS BEFORE U IS CLEARED.

A "MOVE U TO REGISTER=1" WILL CAUSE THE PROCESSOR TO WAIT UNTIL U IS FILLED, AT WHICH TIME THE MOVE IS PERFORMED AND PROCESSING CONTINUES.

CLEAR

FORMAT: CLEAR REGISTER-1 [REGISTER-2] [REGISTER-N]

DESCRIPTION:

SETS THE SPECIFIED REGISTERS TO ZERO.

IF MORE THAN ONE REGISTER IS TO BE CLEARED IN ONE CLOCK TIME, THEY MUST BE ENTIRELY FROM GROUP "A" OR GROUP "B"; BUT THEY MUST NOT BE MIXED AND MUST BE SEPARATED BY BLANKS OR COMMAS.

GROUP A	GROUP B
FA	ASTACK (STACK POINTER RESET)
FB	SO (CLEARS ALL 48 BITS)
FL	BR (ALSO CLEARS LR)
FU	LR (ALSO CLEARS BR)
L	
T	
X	
Y	
CP	

TIME: ONE CLOCK

ANY OTHER SELECT 0, 1 AND 2 REGISTERS(EXCEPT CONDITION REGISTERS AND THE M, MC, MD, ME AND MF REGISTERS)AND ANY 24 BIT SCRATCHPAD WORDS CAN ALSO BE CLEARED; HOWEVER, ONE CLOCK IS USED FOR EACH OF THESE OTHER REGISTERS AND SCRATCHPAD WORDS.

EXAMPLES:	TIME
CLEAR FL, FA, FU, CP	1 CLOCK
CLEAR X,Y,T,L BR	2 CLOCKS
CLEAR CP, SO,S1A,S1B	4 CLOCKS
CLEAR TA TB TC,LA LB LC,CA CB	8 CLOCKS

COMPLEMENT

FORMAT: COMPLEMENT REG-1(LIT-1) [AND REG-1(LIT-2) [AND REG-1(LIT-N)]]

DESCRIPTION:

THIS INSTRUCTION IS USED TO COMPLEMENT(SWITCH THE STATE OF) THE BIT IN REGISTER-1 WHICH IS SPECIFIED BY LITERAL-1. MORE THAN ONE BIT IN ANY ONE REGISTER CAN BE COMPLEMENTED WITH THE SAME INSTRUCTION IF ALL THE BITS ARE IN THE SAME 4 BIT REGISTER.

REGISTER-1 MAY BE ANY 4 BIT(SELECT 0 OR 1) REGISTER EXCEPT A CONDITION REGISTER AND THE CPU, MC, MD, ME AND MF REGISTERS; OR IT MAY BE FL, FB, L OR T. IF MORE THAN ONE BIT IS TO BE COMPLEMENTED IN FL, FB, L OR T, THEN ALL THE BITS MUST BE IN THE SAME 4 BIT SUBFIELD.

LITERAL-1(LITERAL-N) HAS A RANGE OF 0 THRU 3 FOR A 4 BIT REGISTER, 0 THRU 19 FOR FL AND 0 THRU 23 FOR FB, L AND T. PARENTHESES ARE REQUIRED AROUND LITERAL-1.

NOTE: ALSO SEE THE "SET" AND "RESET" INSTRUCTIONS.

EXAMPLE: COMPLEMENT LD(0) AND L(13)
BEFORE---L = 123856
AFTER----L = 123456

TIME: ONE CLOCK

CONSTANT

FORMAT: CONSTANT LITERAL-1

DESCRIPTION:

THIS INSTRUCTION IS USED TO PLACE AN IN-LINE, 16 BIT, M-STRING CONSTANT. IT IS THE RESPONSIBILITY OF THE PROGRAMMER TO PROVIDE ANY PROTECTION THAT MAY BE NEEDED TO PREVENT A "CONSTANT" FROM EXECUTING.

LITERAL-1 HAS A RANGE OF 0 THRU 65535(HEXADECIMAL H0 THRU HFFFF).

COUNT

FORMAT:

```
                [FA] [UP]        [FL] [DOWN]     [CPL]
      COUNT [FL] [DOWN] AND [FA] [UP]    BY [LITERAL-1]
```

DESCRIPTION:

INCREMENTS OR DECREMENTS THE DESIGNATED REGISTER BY THE AMOUNT OF THE LITERAL OR CPL. IF THE VALUE OF THE LITERAL IS ZERO, CPL IS USED.

ALL COMBINATIONS ARE ACCEPTABLE BUT "COUNT FA UP AND FL UP", WHICH IS NOT ACCEPTABLE.

IF FA IS COUNTED DOWN, IT MAY GO THROUGH ZERO (I.E., IF FA=0 AND IS COUNTED DOWN BY ONE, IT IS SET TO ALL ONES). IF FL IS COUNTED DOWN, IT WILL NOT BECOME LESS THAN ZERO.

IF EITHER FA OR FL OVERFLOWS, IT WRAPS AROUND TO ZERO, E.G. IF EITHER IS EQUAL TO THE MAXIMUM VALUE IT CAN CONTAIN AND IS COUNTED UP BY 1, IT BECOMES EQUAL TO ZERO.

LITERAL-1 IS 5 BITS, VALUE 0 TO 31.

EXAMPLE: INSTRUCTION- COUNT FA UP AND FL DOWN BY 10

	REGISTER FA	REGISTER FL
INITIAL	09A7FB	00008
RESULT	09A805	00000

FA IS COUNTED UP BY DECIMAL 10, HEXADECIMAL A, WHILE FL IS COUNTED DOWN 8 TO ITS MINIMUM VALUE.

TIME: ONE CLOCK

DEC

FORMAT: DEC REGISTER-1 BY [LITERAL-1] [TEST]
[REGISTER-2]

DESCRIPTION:

THIS INSTRUCTION IS USED TO DECREMENT REGISTER-1 BY THE VALUE OF LITERAL-1 OR REGISTER-2 AND PLACE THE RESULT IN REGISTER-1.

REGISTER-1 MAY BE ANY 4 BIT REGISTER(SELECT 0 OR 1) EXCEPT A CONDITION REGISTER AND THE CPU, MC, MD, ME AND MF REGISTERS.

REGISTER-2 MAY BE ANY 4 BIT REGISTER(SELECT 0 OR 1) EXCEPT CPU, MC, MD, ME AND MF. THE CONTENTS OF THIS REGISTER ARE NOT CHANGED BY THIS INSTRUCTION.

LITERAL-1 HAS A RANGE OF 0 THRU 15.

IF THE "TEST" OPTION IS USED, THEN IF REGISTER-1 UNDERFLOWS(IS DECREMENTED BEYOND 0, THE SMALLEST VALUE IT CAN CONTAIN) AS A RESULT OF THIS INSTRUCTION, THEN THE NEXT MICRO-INSTRUCTION WILL BE SKIPPED(NOT EXECUTED). IF NO UNDERFLOW OCCURS WITH THE "TEST" OPTION OR IF THE "TEST" OPTION IS NOT USED, THEN THE NEXT MICRO-INSTRUCTION WILL BE EXECUTED.

NOTE THAT ALL 4 BIT REGISTERS COUNT MODULO 16, E.G. IF A REGISTER CONTAINS A VALUE OF 0 AND IS DECREMENTED BY 2, IT UNDERFLOWS TO A VALUE OF 14.

TIME: ONE CLOCK FOR THE "LITERAL-1" OPTION.
TWO CLOCKS IF THE "LITERAL-1 TEST" OPTION IS USED AND UNDERFLOW OCCURS OR IF THE "REGISTER-2" OPTION IS USED.
THREE CLOCKS IF THE "REGISTER-2 TEST" OPTION IS USED AND UNDERFLOW OCCURS.

DISPATCH

FORMAT: DISPATCH [LOCK] [WRITE] [READ [AND CLEAR]] [SKIP WHEN UNLOCKED]

DESCRIPTION:

THE DISPATCH OPERATION IS USED TO SEND A MESSAGE TO ANOTHER PORT AND TO RECEIVE INTERRUPT INFORMATION FROM OTHER PORTS. AN EXAMPLE OF A MESSAGE SENT TO ANOTHER PORT IS THE ADDRESS OF AN I/O DESCRIPTOR. AN EXAMPLE OF INTERRUPT INFORMATION RECEIVED FORM ANOTHER PORT IS THE ADDRESS + 48 OF THE RESULT INFORMATION STORED BY AN I/O CHANNEL AFTER THE COMPLETION OF AN I/O OPERATION.

BEFORE SENDING A MESSAGE TO ANOTHER PORT, THE PROCESSOR SHOULD FIRST ATTEMPT TO GAIN CONTROL OF THE INTERRUPT SYSTEM WITH A "DISPATCH LOCK". THIS IS NECESSARY BECAUSE THE INTERRUPT SYSTEM IS SHARED BY ALL PORTS.

THE "DISPATCH LOCK" OPTION WILL LOCK(MARK AS IN USE) THE INTERRUPT SYSTEM AND THEN SKIP(NOT EXECUTE) THE NEXT MICRO-INSTRUCTION IF THE INTERRUPT SYSTEM WAS ALREADY LOCKED.

THE "DISPATCH LOCK SKIP WHEN UNLOCKED" OPTION WILL LOCK THE INTERRUPT SYSTEM AND THEN SKIP THE NEXT MICRO-INSTRUCTION IF THE INTERRUPT SYSTEM WAS NOT ALREADY LOCKED.

THE "DISPATCH WRITE" OPTION IS USED TO SEND A 24 BIT MESSAGE TO ANOTHER PORT. BEFORE A "DISPATCH WRITE" IS EXECUTED, THE L REGISTER MUST CONTAIN THE 24 BIT MESSAGE, AND THE 7 LEAST SIGNIFICANT BITS OF THE T REGISTER MUST CONTAIN THE DESTINATION PORT AND CHANNEL NUMBERS WITH T (17), (18) AND (19) = THE PORT NUMBER AND T(20), (21) AND (22) = THE CHANNEL NUMBER. THE CONTENTS OF THE L REGISTER ARE THEN STORED IN THE DISPATCH BUFFER(S-MEMORY LOCATIONS 0 THRU 23), AND THE PORT AND CHANNEL NUMBERS ARE TRANSFERED TO THE PORT INTERCHANGE.

THE "DISPATCH READ" OPTION IS USED TO TRANSFER A 24 BIT MESSAGE FROM THE DISPATCH BUFFER TO THE L REGISTER AND THE SOURCE PORT AND CHANNEL NUMBERS TO THE 7 LEAST SIGNIFICANT BITS OF THE T REGISTER. NOTE: T(23), WHEN FOUND SET AFTER A DISPATCH READ AND WHEN THE SOURCE PORT IS AN I/O MULTIPLEXOR, INDICATES THAT AN S MEMORY PARITY ERROR WAS ENCOUNTERED DURING THE FETCH OF AN I/O DESCRIPTOR ADDRESS OR AN I/O DESCRIPTOR OR DURING A RESULT SWAP OPERATION. IN THIS CASE, THE MESSAGE TRANSFERRED TO THE L REGISTER WILL BE THE ADDRESS + 24 OF THE PARITY ERROR.

DISPATCH(CONTINUED)

THE "DISPATCH READ AND CLEAR" OPTION WILL DO EVERYTHING A "DISPATCH READ" WILL DO AND ADDITIONALLY WILL CLEAR THE INCN(INTERRUPT CONDITION) REGISTER.

THE CONTENTS OF THE L AND T REGISTERS WILL BE UNCHANGED AFTER A "DISPATCH WRITE" OPERATION, AND ONLY THE LEAST SIGNIFICANT 7 BITS OF THE T REGISTER ARE INVOLVED IN ANY DISPATCH OPERATION.

IF THE "SKIP WHEN UNLOCKED" OPTION IS USED WITH ANY VARIANT OTHER THAN A "DISPATCH LOCK", THE NEXT MICRO-INSTRUCTION WILL ALWAYS BE SKIPPED.

EOR

FORMAT: EOR REGISTER-1 WITH [LITERAL-1]
[REGISTER-2]

DESCRIPTION:

THIS INSTRUCTION IS USED TO LOGICALLY BIT "EXCLUSIVE OR" REGISTER-1 WITH LITERAL-1 OR REGISTER-2 AND PLACE THE RESULT IN REGISTER-1.

REGISTER-1 MAY BE ANY 4 BIT REGISTER(SELECT 0 OR 1) EXCEPT A CONDITION REGISTER AND THE CPU, MC, MD, ME AND MF REGISTERS.

REGISTER-2 MAY BE ANY 4 BIT REGISTER(SELECT 0 OR 1) EXCEPT CPU, MC, MD, ME AND MF. THE CONTENTS OF THIS REGISTER ARE NOT CHANGED BY THIS INSTRUCTION.

LITERAL-1 HAS A RANGE OF 0 THRU 15.

"EOR" TRUTH TABLE:

```
0 0 = 0
0 1 = 1
1 0 = 1
1 1 = 0
```

EXAMPLE: EOR TB WITH 3

```
BEFORE---T = 05F312
AFTER----T = 06F312
```

TIME: ONE CLOCK UNLESS THE "REGISTER-2" OPTION IS USED, THEN TWO CLOCKS

EXIT

FORMAT: EXIT

DESCRIPTION:

ALLOWS RETURN TO THE CALLING PROGRAM BY CAUSING THE COMPILER TO GENERATE A "MOVE TAS TO A" OPERATION.

THE TOP OF THE A-STACK (TAS) WILL BE MOVED TO A, WHICH IS USED BY THE HARDWARE LOGIC AS THE ADDRESS OF THE NEXT INSTRUCTION TO BE FETCHED. A STACK "POP" IS DONE AUTOMATICALLY BY THE HARDWARE, AFTER THE MOVE.

THEREFORE, "MOVE TAS TO A" MAY BE USED INSTEAD OF "EXIT" WITH THE SAME RESULT.

TIME: TWO CLOCKS

EXTRACT

FORMAT: EXTRACT LITERAL-1 BITS FROM T(LITERAL-2) [TO REGISTER-1]

DESCRIPTION:

ISOLATES THE SPECIFIED BITS FROM T AND MOVES THEM TO REGISTER-1.

ANY NUMBER OF BITS MAY BE MOVED FROM 1 THROUGH 24. THESE MAY BE TAKEN FROM ANY BITS OF T. LITERAL-1, THE NUMBER OF BITS TO BE EXTRACTED, MAY BE FROM 0-24. IF IT IS ZERO, NO OPERATION IS PERFORMED. LITERAL-2 INDICATES THE LEFT-MOST BIT TO BE MOVED AND MAY BE FROM 0 TO 23.

THE SELECTED BITS ARE RIGHT JUSTIFIED IN THE RECEIVING FIELD, AND LEFT ZEROES ARE INSERTED IF THE NUMBER OF BITS MOVED IS LESS THAN THE LENGTH OF THE RECEIVING REGISTER.

SPACES BEFORE AND AFTER THE PARENTHESES ARE OPTIONAL, PARENTHESES ARE REQUIRED.

REGISTER-1 MAY BE T, X, Y, OR L. T REMAINS UNCHANGED, UNLESS IT IS SPECIFIED AS THE DESTINATION REGISTER.

EXAMPLE: INSTRUCTION- EXTRACT 4 BITS FROM T(20) TO L

	REGISTER T	REGISTER L
INITIAL	0138E4	1E39FC
RESULT	0138E4	000004

T REMAINS UNCHANGED, WHILE THE FOUR EXTRACTED BITS FROM T ARE PLACED IN L. THE BITS ARE RIGHT JUSTIFIED AND LEADING ZEROES ADDED.

TIME: ONE CLOCK

NOTE: CAUTION MUST BE USED WHEN "ORING" INTO THE M REGISTER BEFORE AN EXTRACT. THE ACTUAL MACHINE INSTRUCTION REQUIRES THE RIGHT BIT POINTER FOR THE EXTRACTION FIELD, NOT THE LEFT. THE HARDWARE ALSO INDEXES THE T REGISTER FROM 1 TO 24, LEFT TO RIGHT, NOT 0 TO 23. EXTRACT (LITERAL-1) BITS FROM T TO REGISTER-1 WILL CAUSE THE BIT POINTER TO BE SET TO ZERO, E.G. "EXTRACT 8 BITS FROM T TO X".

GO TO

FORMAT: GO TO LABEL-1

DESCRIPTION:

CAUSES A BRANCH TO THE ABSOLUTE ADDRESS SPECIFIED BY LABEL-1. LABEL-1 MUST BE AT AN ADDRESS EQUAL TO OR LESS THAN 8191 OR (HEXADECIMAL 1FFF).

IF AN ADDRESS FIELD EQUAL TO ZERO IS DESIRED, PLEASE SEE THE "BRANCH" INSTRUCTION.

TIME: TWO CLOCKS

IF

1of6

```
                [REGISTER-1(LITERAL-1)]  [TRUE]            [CALL]
FORMAT 1:   IF [CONDITION-1]             [FALSE]  [THEN]   [GO TO] LABEL-1
```

```
                [REGISTER-1(LITERAL-1)]  [TRUE]
FORMAT 2:   IF [CONDITION-1]             [FALSE]  [THEN]

            BEGIN
               ANY-STATEMENTS
            END [ELSE
            BEGIN
               ANY-STATEMENTS
            END]
```

```
                                         [TRUE]
FORMAT 3:    IF MODULE-OPTION-1          [FALSE]  [THEN]  INCLUDE
             BEGIN
                ANY-STATEMENTS
             END [ELSE
             BEGIN
                ANY-STATEMENTS
             END]
```

DESCRIPTION:

FORMAT 1:

THIS FORMAT IS USED TO TEST A BIT OR BITS FOR TRUE(ON OR = 1) OR FALSE(OFF OR = 0) AND THEN TO EITHER CALL OR GO TO LABEL-1 IF THE TEST CONDITION IS MET. BY USING AND/OR LOGIC, MORE THAN ONE BIT CAN BE TESTED AT THE SAME TIME, BUT ONLY IF ALL THE BITS ARE IN THE SAME 4 BIT REGISTER.

REGISTER-1 MAY BE ANY 4 BIT(SELECT 0 OR 1) REGISTER EXCEPT CPU, MC, MD, ME AND MF; OR IT MAY BE FL, FB, L OR T.

IF(CONTINUED) 2 of 6

LITERAL-1 HAS A RANGE OF 0 THRU 3 FOR A 4 BIT REGISTER, 0 THRU 19 FOR FL AND 0 THRU 23 FOR FB, L AND T. PARENTHESES ARE REQUIRED AROUND LITERAL-1.

CONDITION-1 MAY BE ANY CONDITION WHICH IS AVAILABLE FROM THE CONDITION REGISTERS(SEE "CONDITIONS").

IF NEITHER "TRUE" NOR "FALSE" IS SPECIFIED, THEN TRUE IS ASSUMED.

IF THE "GO TO" OPTION IS USED, THEN LABEL-1 MUST BE ASSOCIATED WITH AN ADDRESS WHICH IS NOT MORE THAN 15 M-MEMORY LOCATIONS AWAY.

FORMAT 2:

THIS FORMAT IS THE SAME AS FORMAT 1 EXCEPT THAT BEGIN-END PAIRS REPLACE THE "CALL" AND "GO TO" OPTIONS.

IF THE TEST CONDITION IS MET THEN ALL STATEMENTS BETWEEN THE FIRST BEGIN-END PAIR ARE PERFORMED AND THEN A BRANCH IS TAKEN AROUND ANY SECOND(OPTIONAL) BEGIN-END PAIR.

IF THE TEST CONDITION IS NOT MET, THEN A BRANCH AROUND THE FIRST BEGIN-END PAIR IS TAKEN.

THE FIRST "BEGIN" CARD MUST IMMEDIATELY FOLLOW THE "IF" CARD, EXCEPT THAT COMMENT OR BLANK CARDS MAY BE USED BETWEEN THEM. THE SECOND BEGIN-END PAIR IS OPTIONAL; BUT, IF USED, THEN THE FIRST "BEGIN" CARD MUST BE PAIRED WITH AN "END ELSE" CARD. AGAIN, NO OTHER CARDS OTHER THAN COMMENT OR BLANK CARDS MAY BE USED BETWEEN THE "END ELSE" AND FOLLOWING "BEGIN" CARDS. THERE ARE NO RESTRICTIONS ON THE NUMBER OR TYPE OF STATEMENTS WHICH MAY BE USED BETWEEN ANY BEGIN-END PAIR OF CARDS. NOTE: IF ONLY ONE STATEMENT APPEARS BETWEEN ANY BEGIN-END PAIR, THEN BOTH THE "BEGIN" AND "END" MAY BE OMITTED UNLESS THAT ONE STATEMENT IS ANOTHER "IF" STATEMENT.

FORMAT 3:

THIS FORMAT IS INTENDED TO BE USED FOR CONDITIONAL INCLUSION OF CODE, DEPENDING UPON THE SETTING OF A USER DEFINED MODULE-OPTION TOGGLE. THIS MODULE-OPTION TOGGLE IS DECLARED AND "SET" OR "RESET" ON A MODULE OPTION $ CARD(SEE APPENDIX A).

MORE THAN ONE MODULE-OPTION-TOGGLE CAN BE TESTED WITH THE SAME "IF" STATEMENT BY USING AND/OR LOGIC. IF "NOT" IS USED IN FRONT OF ANY MODULE-OPTION-TOGGLE, THEN THAT MODULE-OPTION-TOGGLE IS CHECKED FOR THE "RESET" STATE; AND THE "TRUE" OR "FALSE" OPTION APPLIES TO THE WHOLE CONDITION, E.G. "IF NOT A AND B FALSE ..." PRODUCES THE SAME RESULT AS "IF A OR NOT B TRUE ...". AGAIN, IF BOTH "TRUE" AND "FALSE" ARE OMITTED, THEN TRUE IS ASSUMED.

THE BEGIN-END PAIRS ARE AS EXPLAINED FOR FORMAT 2 ABOVE.

IF(CONTINUED)
3 of 6

TIME: FOR THE "IF" STATEMENT ONLY IS TWO CLOCKS IF THE BRANCH IS TAKEN, OTHERWISE ONE CLOCK.

EXAMPLE OF FORMAT 1: IF TD(2) TRUE GO TO LABL7

REGISTER TD	BRANCH
B0101	NO
B1101	NO
B0111	YES
B0011	YES

IN CASES THREE AND FOUR THE BRANCH TO "LABL7" WOULD BE TAKEN SINCE BIT POSITION TWO OF TD IS ON. NOTE THAT TD(2) COULD HAVE BEEN REFERRED TO AS T(14).

IN THE FOLLOWING EXAMPLES THE STATEMENTS ON THE LEFT WILL GENERATE CODE EQUIVALENT TO THE CODE GENERATED BY THE STATEMENTS ON THE RIGHT.

```
     IF X=Y THEN GO TO +A                    IF X EQL Y GO TO +A
          * IN BOTH CASES AN ERROR WILL BE GENERATED IF THE RELATIVE
          *    DISPLACEMENT TO .A IS GREATER THAN 15
*
     IF CYL AND CYD THEN GO TO B             SKIP WHEN BICN ALL B0011 FALSE
                                                  GO TO B
*
     IF TA(0) THEN CALL SUBA                 SKIP WHEN TA ANY B1000 FALSE
                                                  CALL SUBA
*
     IF TB(1) OR TB(3) THEN EXIT             SKIP WHEN TB ANY B0101 FALSE
                                                  EXIT
*
     IF CA(0) AND CA(2) FALSE EXIT           SKIP WHEN CA ALL B1010
                                                  EXIT
*
     IF LF(2) THEN                           IF LF(2) FALSE THEN GO TO +C
         MOVE X TO Y                         MOVE X TO Y
     SET TA(1)                        .C     OR TA WITH B0100
*
     IF FU(1) FALSE THEN                     IF FU(1) GO TO +D
         COMPLEMENT T(10)                    EOR TC WITH B0010
     ELSE                                    GO TO +E
         RESET FL(5)                  .D     AND FLC WITH B1011
     SET L(6) AND L(7)                .E     OR LB WITH B0011
```

IF(CONTINUED) 4 of 6

```
*
    IF FLF(3) FALSE THEN                        IF FLF(3) GO TO +A
    BEGIN                                       AND FU WITH B1010
        RESET FB(1) AND FB(3)                   MOVE NULL TO S14A
        CLEAR S14A                        .A    XCH S14 F S14
    END
    XCH S14 F S14
```

IN THE ABOVE EXAMPLE AND IN ALL FOLLOWING EXAMPLES WHERE THE TEST IN THE IF STATEMENT IS ON A SINGLE BIT, IF MORE THAN 15 M-CODE INSTRUCTIONS ARE GENERATED BETWEEN THE FIRST BEGIN-END PAIR THEN AN ERROR MESSAGE WILL RESULT INDICATING A BRANCH ADDRESS OUT OF RANGE.

```
    IF LA(0) THEN                               IF LA(0) FALSE GO TO +B
    BEGIN                                       MOVE TAS TO T
        MOVE TAS TO T                           STORE F INTO S0
        STORE F INTO S0                         GO TO +C
    END ELSE                              .B    MOVE FA TO T
        MOVE FA TO T                      .C    MOVE TE TO LF
    MOVE TE TO LF
*
    IF TD(3) THEN                               IF TD(3) FALSE GO TO +D
        MOVE L TO X                             MOVE L TO X
    ELSE                                        GO TO +E
    BEGIN                                 .D    MOVE T TO X
        MOVE T TO X                             MOVE SUM TO X
        MOVE SUM TO X                     .E    MOVE SUM TO FA
    END
    MOVE SUM TO FA
*
    IF LA = 14 THEN                             SKIP WHEN LA EQL B1110
    BEGIN                                       GO TO +A
        MOVE 512 TO X                           MOVE 512 TO X
        MOVE X TO S15A                          MOVE X TO S15A
    END                                   .A    EOR FU WITH B1001
    COMPLEMENT FU(0) AND FU(3)
*
    IF LE(2) OR LE(3) FALSE THEN                SKIP WHEN LE ANY B0011 FALSE
    BEGIN                                       GO TO +B
        MOVE L TO X                             MOVE L TO X
        MOVE T TO Y                             MOVE T TO Y
        MOVE SUM TO FA                          MOVE SUM TO FA
    END ELSE                                    GO TO +C
    BEGIN                                 .B    MOVE FA TO X
        MOVE FA TO X                            MOVE FL TO Y
        MOVE FL TO Y                            MOVE DIFF TO FA
        MOVE DIFF TO FA                   .C    READ 8 BITS TO T
    END
    READ 8 BITS TO T
```

IF(CONTINUED) 5of6

FOLLOWING ARE EXAMPLES OF CONDITIONAL INCLUSION OF CODE

```
        $ SET DEBUG, RESET TRACE
        $ SET TRACE, RESET B1501
```

AFTER PROCESSING THESE $ CARDS, THE MODULE OPTIONS WILL BE SET TRUE OR FALSE AS FOLLOWS:

```
        DEBUG = TRUE
        TRACE = TRUE
        B1501 = FALSE
*
    IF DEBUG THEN INCLUDE                   CALL DEBUG-ROUTINE
        CALL DEBUG-ROUTINE
*
    IF TRACE THEN INCLUDE                   CALL SAVE-REGISTERS
    BEGIN                                   CALL TRACE-ROUTINE
        CALL SAVE-REGISTERS
        CALL TRACE-ROUTINE
    END
*
    IF B1501 THEN INCLUDE                   NORMALIZE
    BEGIN
.LOOP  IF MSBX FALSE THEN
       BEGIN
         IF FL NEQ 0 THEN
         BEGIN
           SHIFT XY LEFT BY 1 BIT
           COUNT FL DOWN BY 1
           GO TO -LOOP
         END
       END
    END ELSE
      NORMALIZE
*
    IF DEBUG AND NOT B1501 INCLUDE          MOVE T TO X
    BEGIN
        MOVE T TO X
    END ELSE
    BEGIN
        MOVE L TO X
        MOVE T TO SOA
    END
```

```
*
    IF NOT TRACE OR B1501 INCLUDE       CALL TRACE-ROUTINE
    BEGIN                               MOVE T TO X
        MOVE L TO X
        MOVE T TO S1A
    END ELSE
    BEGIN
        CALL TRACE-ROUTINE
        MOVE T TO X
    END
```

ANY OF THE PRECEDING EXAMPLES MAY BE NESTED WITHIN ANY OF THE ABOVE BEGIN-END PAIRS UP TO A MAXIMUM OF 31 LEVELS. THAT IS, AT ANY GIVEN TIME DURING A COMPILATION THERE MAY BE AT MOST 31 BEGINS THAT HAVE NOT BEEN PAIRED UP WITH THEIR RESPECTIVE ENDS.

INC

FORMAT: INC REGISTER-1 BY [LITERAL-1] [TEST]
[REGISTER-2]

DESCRIPTION:

THIS INSTRUCTION IS USED TO INCREMENT REGISTER-1 BY THE VALUE OF LITERAL-1 OR REGISTER-2 AND PLACE THE RESULT IN REGISTER-1.

REGISTER-1 MAY BE ANY 4 BIT REGISTER(SELECT 0 OR 1) EXCEPT A CONDITION REGISTER AND THE CPU, MC, MD, ME AND MF REGISTERS.

REGISTER-2 MAY BE ANY 4 BIT REGISTER(SELECT 0 OR 1) EXCEPT CPU, MC, MD, ME AND MF. THE CONTENTS OF THIS REGISTER ARE NOT CHANGED BY THIS INSTRUCTION.

LITERAL-1 HAS A RANGE OF 0 THRU 15.

IF THE "TEST" OPTION IS USED, THEN IF REGISTER-1 OVERFLOWS(IS INCREMENTED BEYOND 15, THE LARGEST VALUE IT CAN CONTAIN) AS A RESULT OF THIS INSTRUCTION, THEN THE NEXT MICRO-INSTRUCTION WILL BE SKIPPED(NOT EXECUTED). IF NO OVERFLOW OCCURS WITH THE "TEST" OPTION OR IF THE "TEST" OPTION IS NOT USED, THEN THE NEXT MICRO-INSTRUCTION WILL BE EXECUTED.

NOTE THAT ALL 4 BIT REGISTERS COUNT MODULO 16, E.G. IF A REGISTER CONTAINS A VALUE OF 15 AND IS INCREMENTED BY 2, IT OVERFLOWS TO A VALUE OF 1.

TIME: ONE CLOCK FOR THE "LITERAL-1" OPTION.
TWO CLOCKS IF THE "LITERAL-1 TEST" OPTION IS USED AND OVERFLOW OCCURS OR IF THE "REGISTER-2" OPTION IS USED.
THREE CLOCKS IF THE "REGISTER-2 TEST" OPTION IS USED AND OVERFLOW OCCURS.

JUMP

FORMAT:

```
          [FORWARD]
JUMP      [TO LABEL-1]
          [BACKWARD]
```

DESCRIPTION:

CAUSES TRANSFER OF CONTROL TO THE DESIGNATED LOCATION. THE ADDRESS OF LABEL-1 IS LIMITED TO A RELATIVE DISPLACEMENT OF PLUS OR MINUS 127.

THE "FORWARD" OR "BACKWARD" OPTIONS ARE USED TO CAUSE THE COMPILER TO GENERATE A "JUMP" INSTRUCTION WITH A DISPLACEMENT OF PLUS OR MINUS ZERO, RESPECTIVELY. THIS IS TO FACILITATE "OR-ING" THE ACTUAL DISPLACEMENT INTO THE M REGISTER PRIOR TO THE EXECUTION OF A "JUMP" INSTRUCTION.

TIME: TWO CLOCKS

LIT

FORMAT: [MOVE] [LIT] LITERAL-1 TO REGISTER-1

DESCRIPTION:

THIS INSTRUCTION IS USED TO MOVE A LITERAL TO ANY SELECT 0, 1 OR 2 REGISTER EXCEPT A CONDITION REGISTER OR THE M, MC, MD, ME AND MF REGISTERS.

LITERAL-1 MAY BE A DECIMAL INTEGER "0" THRU "16777215", A HEXADECIMAL NUMBER "H0" THRU "HFFFFFF", OR A BINARY NUMBER "B0" THRU "B111111111111111111111"(MAXIMUM LENGTH OF 21 CHARACTERS). LEADING ZEROS ARE NOT REQUIRED UNLESS THE ACTUAL VALUE OF THE LITERAL IS ZERO. IF THE VALUE OF THE LITERAL IS GREATER THAN THE MAXIMUM VALUE THAT REGISTER-1 CAN CONTAIN, LEFT TRUNCATION OCCURS; IF IT IS LESS, LEFT ZERO FILL OCCURS.

NOTE: "LIT .... TO TAS" AUTOMATICALLY CAUSES A STACK "PUSH" BEFORE THE MOVE.

EXAMPLES:

```
     MOVE 12 TO L
          BEFORE---L = 309A13
          AFTER----L = 00000C

     MOVE HF TO LB
          BEFORE---L = 309A13
          AFTER----L = 3F9A13

     MOVE B1111 TO CPU
          BEFORE---CPU = 0
          AFTER----CPU = 3
```

TIME: ONE CLOCK UNLESS THE VALUE OF LITERAL-1 IS GREATER THAN 255, THEN 3 CLOCKS.

LOAD

FORMAT: LOAD F FROM DOUBLE-PAD-WORD-1

DESCRIPTION:

MOVES A PAIR OF SCRATCHPAD WORDS (48 BITS TOTAL) TO F. THE SCRATCHPAD IS UNCHANGED.

DOUBLE-PAD-WORDS:

S0	S4	S8	S12
S1	S5	S9	S13
S2	S6	S10	S14
S3	S7	S11	S15

TIME: ONE CLOCK

MOVE

FORMAT 1: MOVE REGISTER-1 TO REGISTER-2

FORMAT 2: MOVE REGISTER-1 TO SCRATCHPAD-WORD-1

FORMAT 3: MOVE SCRATCHPAD-WORD-1 TO REGISTER-2

FORMAT 4: MOVE ADDRESS(LABEL-1) TO REGISTER-2

FORMAT 5: MOVE SEGMENT-COUNT TO REGISTER-1

FORMAT 6: [MOVE] LITERAL-1 TO REGISTER-2
[LIT]

FORMATS 1, 2 AND 3:

REGISTER-1, THE SOURCE REGISTER, MAY BE ANY REGISTER EXCEPT CPU AND MSMA. IF IT IS M, MC, MD, ME OR MF, THEN, FOR FORMAT 1, THE "MOVE" MICRO ITSELF IS MOVED TO THE DESTINATION REGISTER; FOR FORMAT 2, THE SCRATCHPAD WORD IS CLEARED TO ZEROS.

REGISTER-2, THE DESTINATION REGISTER, MAY BE ANY REGISTER EXCEPT A CONDITION REGISTER OR A RESULT REGISTER. IF IT IS M, MC, MD, ME OR MF, THEN THE SOURCE REGISTER OR SCRATCHPAD WORD IS BIT "OR-ED" INTO IT.

SCRATCHPAD-WORD-1 MAY BE ANY 24 BIT SCRATCHPAD WORD. NOTE THAT A SCRATCHPAD WORD MAY NOT BE MOVED DIRECTLY TO ANOTHER SCRATCHPAD WORD.

THE SOURCE FIELD IS UNCHANGED BY ANY MOVE UNLESS "MOVE TAS TO ..." IS USED, THEN A STACK "POP" IS AUTOMATICALLY DONE AFTER THE MOVE. FOR "MOVE ... TO TAS", A STACK "PUSH" IS AUTOMATICALLY DONE BEFORE THE MOVE.

MOVE(CONTINUED)

FORMAT 4:
THIS FORMAT IS USED TO MOVE THE COMPILER GENERATED M MEMORY ADDRESS OF LABEL-1 TO REGISTER-2. REGISTER-2 MAY BE ANY SELECT 2 REGISTER EXCEPT M. PARENTHESES ARE REQUIRED AROUND LABEL-1, WHICH MAY BE EITHER A UNIQUE LABEL, A POINT LABEL OR THE NAME OF A SEGMENT(SEE THE PSEUDO OPERATOR "SEGMENT"). IF IT IS A POINT LABEL, IT MUST BE REFERENCED EITHER + OR - , E.G. "MOVE ADDRESS(+PL) TO TAS".

FORMAT 5:
IF THE "SEGMENT" PSEUDO-INSTRUCTION IS USED IN THE PROGRAM, THEN THIS FORMAT CAN BE USED TO MOVE, AT RUN TIME, AN 8 BIT LITERAL COUNT OF THE NUMBER OF TIMES THE "SEGMENT" STATEMENT APPEARS IN THE PROGRAM PREVIOUS TO THE OCCURRENCE OF THIS INSTRUCTION.

REGISTER-1 MAY BE ANY SELECT 2 REGISTER EXCEPT M.

FORMAT 6:
PLEASE SEE THE "LIT" INSTRUCTION FOR AN EXPLANATION OF THIS FORMAT.

ALL MOVES TO UNEQUAL LENGTHS WILL JUSTIFY RIGHT WITH LEFT ZERO FILL OR TRUNCATE FROM THE LEFT.

EXAMPLE: MOVE X TO S7B

	REGISTER X	SCRATCHPAD-WORD S7B
INITIAL	39FED0	333E05
RESULT	39FED0	39FED0

THE CONTENTS OF REGISTER X HAVE BEEN MOVED TO THE SCRATCHPAD-WORD S7B, X REMAINS UNCHANGED.

TIME: ONE CLOCK EXCEPT THAT FORMAT 4 ALWAYS TAKES 3 CLOCKS AND FORMAT 6 TAKES 3 CLOCKS WHEN LITERAL-1 HAS A VALUE GREATER THAN 255.

NOP

FORMAT: NOP

DESCRIPTION:

THIS IS THE NO OPERATION INSTRUCTION. IT WILL DO NOTHING EXCEPT USE ONE CLOCK AND TAKE UP ONE WORD OF M-MEMORY.

TIME: ONE CLOCK

NORMALIZE

FORMAT: NORMALIZE

DESCRIPTION:

SHIFTS XY (X CONCATENATE Y) LEFT WHILE COUNTING FL DOWN, ANY NUMBER OF BITS, UNTIL FL=0, OR UNTIL THE MOST SIGNIFICANT BIT OF X (DETERMINED BY CPL)=1.

TIME: ONE CLOCK PER BIT SHIFTED, MINIMUM ONE CLOCK

OR

FORMAT: OR REGISTER-1 WITH [LITERAL-1]
[REGISTER-2]

DESCRIPTION:

THIS INSTRUCTION IS USED TO LOGICALLY BIT "OR" REGISTER-1 WITH LITERAL-1 OR REGISTER-2 AND PLACE THE RESULT IN REGISTER-1.

REGISTER-1 MAY BE ANY 4 BIT REGISTER(SELECT 0 OR 1) EXCEPT A CONDITION REGISTER AND THE CPU, MC, MD, ME AND MF REGISTERS.

REGISTER-2 MAY BE ANY 4 BIT REGISTER(SELECT 0 OR 1) EXCEPT CPU, MC, MD, ME AND MF. THE CONTENTS OF THIS REGISTER ARE NOT CHANGED BY THIS INSTRUCTION.

LITERAL-1 HAS A RANGE OF 0 THRU 15.

"OR" TRUTH TABLE:

```
0 0 = 0
0 1 = 1
1 0 = 1
1 1 = 1
```

EXAMPLE: OR TB WITH 3

```
BEFORE---T = 05F312
AFTER----T = 07F312
```

TIME: ONE CLOCK UNLESS THE "REGISTER-2" OPTION IS USED, THEN TWO CLOCKS

OVERLAY

FORMAT: OVERLAY

DESCRIPTION:

OVERLAY M-STRING MEMORY FROM S-MEMORY. PREVIOUS TO INITIATING AN OVERLAY THE L REGISTER MUST CONTAIN THE FIRST M-STRING OVERLAY ADDRESS, FA THE BEGINNING S-MEMORY ADDRESS, AND FL THE LENGTH IN BITS TO BE OVERLAID. OVERLAY WILL CONTINUE UNTIL FL =0 OR A IS OUT OF BOUNDS. THE INTERNAL PROCEDURE IS: MOVE A TO TAS; MOVE L TO A; READ 16 BITS TO L INC FA AND DEC FL; MOVE L TO MSMA; INC A; TEST FL=0 AND A OUT OF BOUNDS; LOOP BACK AND READ OR EXIT.

READ

FORMAT: READ [LIT-1 BITS] [REVERSE] TO [T] / [X] [INC] [FA] / [Y] [DEC] [FL] / [L] AND [INC] [FA] [DEC] [FL]

DESCRIPTION:

INITIATES A MAIN MEMORY READ CYCLE TO THE SPECIFIED REGISTER (T,X,Y,L). FA MUST HAVE BEEN SET TO THE APPROPRIATE S-MEMORY ADDRESS PREVIOUSLY.

LITERAL-1, THE NUMBER OF BITS TO READ AND THE NUMBER TO INC OR DEC, IS OPTIONAL. LITERAL-1 MUST BE AN INTEGER 0 TO 24. IF LITERAL-1 IS OMITTED OR IF IT EQUALS ZERO, THEN CPL INDICATES THE NUMBER OF BITS. READ "CPL BITS" IS NOT PERMITTED.

IF THE NUMBER OF BITS TO READ IS ZERO, THE DESTINATION REGISTER WILL BE SET TO ALL ZEROES, OTHERWISE THE SELECTED BITS ARE RIGHT JUSTIFIED IN THE DESTINATION REGISTER, AND LEFT ZEROES ARE INSERTED IF THE NUMBER OF BITS READ IS LESS THAN THE LENGTH OF THE DESTINATION REGISTER.

IF THE "REVERSE" OPTION IS USED, THEN FA POINTS AT THE LAST BIT + 1 OF THE FIELD TO BE READ FROM MEMORY. THE DESTINATION REGISTER STILL RECEIVES THE CONTENTS OF THIS FIELD AS THOUGH IT HAD BEEN READ IN A FORWARD DIRECTION. FOR EXAMPLE, "READ 24 BITS TO T" WITH FA = ZERO AND "READ 24 BITS REVERSE TO T" WITH FA = 24, WILL BOTH RESULT IN T BEING SET TO THE SAME VALUE, I.E. T(0) CORRESPONDS TO ADDRESS ZERO, AND T(23) CORRESPONDS TO ADDRESS 23.

INCREMENTING OR DECREMENTING OCCURS FOLLOWING THE READ AND IS OPTIONAL. FOR DETAILS SEE THE "COUNT" VERB.

TIME: FIVE CLOCKS

RESET

FORMAT: RESET REG-1(LIT-1) [AND REG-1(LIT-2) [AND REG-1(LIT-N)]]

DESCRIPTION:

THIS INSTRUCTION IS USED TO RESET(= 0) THE BIT IN REGISTER-1 WHICH IS SPECIFIED BY LITERAL-1. MORE THAN ONE BIT IN ANY ONE REGISTER CAN BE RESET WITH THE SAME INSTRUCTION IF ALL THE BITS ARE IN THE SAME 4 BIT REGISTER.

REGISTER-1 MAY BE ANY 4 BIT(SELECT 0 OR 1) REGISTER EXCEPT A CONDITION REGISTER AND THE CPU, MC, MD, ME AND MF REGISTERS; OR IT MAY BE FL, FB, L OR T. IF MORE THAN ONE BIT IS TO BE RESET IN FL, FB, L OR T, THEN ALL THE BITS MUST BE IN THE SAME 4 BIT SUBFIELD.

LITERAL-1(LITERAL-N) HAS A RANGE OF 0 THRU 3 FOR A 4 BIT REGISTER, 0 THRU 19 FOR FL AND 0 THRU 23 FOR FB, L AND T. PARENTHESES ARE REQUIRED AROUND LITERAL-1.

NOTE: ALSO SEE THE "SET" AND "COMPLEMENT" INSTRUCTIONS.

EXAMPLE: RESET T(0) AND TA(3)
BEFORE---T = FACE99
AFTER----T = 6ACE99

TIME: ONE CLOCK

ROTATE OR SHIFT T

```
                [SHIFT]                 [LITERAL-1 BITS]
FORMAT 1:       [ROTATE] T LEFT BY      [CPL]               [TO REGISTER-1]


FORMAT 2:       ROTATE T RIGHT BY LITERAL-1 BITS [TO REGISTER-1]


                                                            [X]
FORMAT 3:       SHIFT T RIGHT BY LITERAL-1 BITS [TO         [Y] ]
                                                            [T]
                                                            [L]
```

DESCRIPTION:

THIS INSTRUCTION IS USED TO ROTATE OR SHIFT THE CONTENTS OF THE T REGISTER AND PLACE THE RESULT EITHER IN T OR IN SOME OTHER REGISTER. IF THE RESULT IS NOT PLACED IN T, THEN T IS UNCHANGED BY THIS INSTRUCTION.

FORMAT 1:

LITERAL-1 HAS A RANGE OF 0 THRU 23. WHEN LITERAL-1 EQUALS ZERO OR WHEN "CPL" IS USED, A SHIFT OR ROTATE BY THE VALUE OF CPL WILL OCCUR. IF CPL IS GREATER THAN 24, THEN 24 IS USED.

IF THE "TO REGISTER-1" OPTION IS USED, THE ROTATED OR SHIFTED RESULT WILL BE PLACED IN REGISTER-1, AND T IS UNCHANGED. OTHERWISE THE RESULT IS PLACED IN T. REGISTER-1 MAY BE ANY SELECT 0, 1 OR 2 REGISTER EXCEPT A CONDITION REGISTER. IF IT IS M, MC, MD, ME OR MF, THEN THE RESULT OF THE ROTATE OR SHIFT OPERATION IS "BIT OR-ED" INTO M AND MODIFIES THE NEXT MICRO-INSTRUCTION.

FORMAT 2:

BECAUSE THE HARDWARE CAN ONLY ROTATE T LEFT, THE COMPILER CONVERTS THIS FORMAT TO THE PROPER LEFT ROTATE TO ACCOMPLISH THE SAME THING. THE "TO REGISTER-1" OPTION IS AS EXPLAINED FOR FORMAT 1 ABOVE.

ROTATE OR SHIFT T(CONTINUED)

FORMAT 3:

IT IS RECOMMENDED THAT THE "EXTRACT" INSTRUCTION ITSELF BE USED, RATHER THAN THIS FORMAT.

BECAUSE THE HARDWARE CAN ONLY SHIFT T LEFT, THE COMPILER GENERATES AN "EXTRACT" TO ACCOMPLISH WHAT IS ASKED FOR HERE. THEREFORE, T MAY BE "SHIFTED" RIGHT ONLY TO X, Y, T OR L. IF THE "TO...." OPTION IS NOT USED, THE RESULT IS PLACED IN T, OTHERWISE T IS UNCHANGED UNLESS "TO T" IS USED.

LITERAL-1 HAS A RANGE OF 1 THRU 23.

TIME: ONE CLOCK

ROTATE OR SHIFT X, Y AND XY

```
                [SHIFT]  [X]  [LEFT]      [UNIT]
FORMAT:         [ROTATE] [Y]  [RIGHT] BY  [LITERAL-1 BITS]
                         [XY]
```

DESCRIPTION:

ROTATES OR SHIFTS (X, Y, OR XY) A SPECIFIED NUMBER OF BITS TO THE RIGHT OR LEFT. ZERO FILL OCCURS WITH SHIFT.

WHEN "UNIT" IS USED, THE OPERAND IS ROTATED OR SHIFTED BY 1, 4, 6, OR 8 BITS ACCORDING TO THE VALUE OF CPU (TYPE OF UNITS CONTAINED IN X AND Y MAY BE DETERMINED BY CPU, EITHER BINARY, 4 BIT, 6 BIT OR 8 BIT DECIMAL). FOR XY (X CONCATENATE Y) LITERAL-1 MAY BE 0-47. FOR X AND Y LITERAL-1 MAY BE 0-23.

TIME: ONE CLOCK PER BIT SHIFTED

SET

FORMAT 1: SET REGISTER-1 TO LITERAL-1

FORMAT 2: SET REG-1(LIT-1) [AND REG-1(LIT-2) [AND REG-1(LIT-N)]]

DESCRIPTION:

FORMAT 1:
IS USED TO SET REGISTER-1 TO THE VALUE OF LITERAL-1. REGISTER-1 MAY BE ANY 4 BIT(SELECT 0 OR 1) REGISTER EXCEPT A CONDITION REGISTER AND THE MC, MD, ME AND MF REGISTERS.

LITERAL-1 HAS A RANGE OF 0 THRU 15, EXCEPT WHEN REGISTER-1 IS CPU, IN WHICH CASE LITERAL-1 HAS A RANGE OF 0 THRU 3.

NOTE: ALSO SEE THE "MOVE" OR "LIT" INSTRUCTION.

FORMAT 2:
IS USED TO SET(= 1) THE BIT SPECIFIED BY LITERAL-1 IN REGISTER-1. MORE THAN ONE BIT IN ANY ONE REGISTER CAN BE SET WITH THE SAME INSTRUCTION IF ALL THE BITS ARE IN THE SAME 4 BIT REGISTER.

REGISTER-1 MAY BE ANY 4 BIT(SELECT 0 OR 1) REGISTER EXCEPT A CONDITION REGISTER AND THE CPU, MC, MD, ME AND MF REGISTERS; OR IT MAY BE FL, FB, L OR T. IF MORE THAN ONE BIT IS TO BE SET IN FL, FB, L OR T, THEN ALL THE BITS MUST BE IN THE SAME 4 BIT SUBFIELD.

LITERAL-1(LITERAL-N) HAS A RANGE OF 0 THRU 3 FOR A 4 BIT REGISTER, 0 THRU 19 FOR FL AND 0 THRU 23 FOR FB, L AND T. PARENTHESES ARE REQUIRED AROUND LITERAL-1.

NOTE: ALSO SEE THE "RESET" AND "COMPLEMENT" INSTRUCTIONS.

EXAMPLE OF FORMAT 1: SET TA TO 3
BEFORE---T = F45678
AFTER----T = 345678

EXAMPLE OF FORMAT 2: SET TC(2) AND T(11)
BEFORE---T = 120456
AFTER----T = 123456

TIME: ONE CLOCK

SKIP

```
                                    [ALL [CLEAR]]
                        [REGISTER-1] [ANY]          [LITERAL-1]
                                    [EQL]
FORMAT:     SKIP WHEN                                          [FALSE]
                        [CONDITION-1]
```

DESCRIPTION:

SKIPS ONE M-INSTRUCTION IF THE DESIGNATED CONDITION IS SATISFIED.

LITERAL-1 IS A 4 BIT MASK. IT MAY BE DECIMAL, BINARY OR HEXADECIMAL.

"ALL" MEANS THAT ALL BITS IN REGISTER-1 CORRESPONDING TO ONE BITS IN MASK MUST BE ONE. "ANY" IS TRUE IF AT LEAST ONE BIT IN REGISTER-1 CORRESPONDING TO A ONE BIT IN MASK IS ONE. "EQL" MEANS THAT ALL REGISTER-1 BITS MUST EQUAL CORRESPONDING BITS IN MASK, OR REGISTER-1 MUST EQUAL THE MASK.

"CLEAR", OPTIONAL AND USED ONLY WITH "ALL", CAUSES THE MASKED BITS OF THE REGISTER TO BE SET TO ZEROES AFTER TESTING THE "ALL" CONDITION. ALL OF THE BITS ARE NOT CLEARED; ONLY BITS TESTED ARE CLEARED. IF "ALL CLEAR" IS USED, THE CLEAR ALWAYS OCCURS WHETHER THE SKIP IS TAKEN OR NOT. IF "ALL" IS USED WITH A MASK OF 0000 THE RESULT IS ALWAYS TRUE. IF "ANY" IS USED WITH A MASK OF 0000 THE RESULT IS ALWAYS FALSE.

"FALSE" IS OPTIONAL AND CAUSES A SKIP WHEN THE WHOLE CONDITION IS FALSE.

CONDITION-1 MAY BE ANY CONDITION AVAILABLE FROM THE CONDITION REGISTERS (SEE "CONDITIONS").

REGISTER-1 MAY BE:

FU	TA	LA	CA	BICN
FLB	TB	LB	CB	FLCN
FLC	TC	LC	CC	INCN
FLD	TD	LD	CD	XYCN
FLE	TE	LE		XYST
FLF	TF	LF		

TIME: TWO CLOCKS ON THE SKIP; ONE CLOCK OTHERWISE

STORE

FORMAT: STORE F INTO DOUBLE-PAD-WORD-1

DESCRIPTION:

MOVES F (48 BITS) INTO A PAIR OF SCRATCHPAD WORDS. F IS UNCHANGED.

DOUBLEPAD WORDS:

S0	S4	S8	S12
S1	S5	S9	S13
S2	S6	S10	S14
S3	S7	S11	S15

TIME: ONE CLOCK

SUBTRACT (BASE RELATE)

FORMAT: SUBTRACT BR FROM FA

DESCRIPTION:

THIS INSTRUCTION IS USED TO SUBTRACT THE BASE REGISTER(BR) FROM FA. THE RESULT IS PLACED IN FA AND THE CONTENTS OF THE BASE REGISTER REMAIN UNCHANGED.

NOTE: ALSO SEE "ADD"

TIME: ONE CLOCK

SWAP

FORMAT: SWAP [LITERAL-1] BITS [REVERSE] WITH [REGISTER-1]

DESCRIPTION:

SWAPS THE SPECIFIED NUMBER OF BITS BETWEEN S-MEMORY AND REGISTER-1. REGISTER-1 MAY BE T, X, Y, L.

FA MUST HAVE BEEN SET TO THE S-MEMORY ADDRESS PREVIOUSLY.

LITERAL-1 MUST BE AN INTEGER 0 THROUGH 24. IT DETERMINES THE NUMBER OF BITS TO SWAP BETWEEN S-MEMORY AND REGISTER-1. IF THE VALUE OF LITERAL-1 IS ZERO, THEN CPL IS USED. IF CPL IS ALSO ZERO, THEN REGISTER-1 IS CLEARED TO ALL ZEROS.

IF LESS THAN 24 BITS ARE SWAPPED, THE LEADING BITS OF THE REGISTER WILL BE ZERO.

FOR AN EXPLANATION OF THE "REVERSE" OPTION, PLEASE SEE THE "READ" AND "WRITE" INSTRUCTIONS.

TIME: AT LEAST 5 CLOCKS IF FREE STANDING

WRITE

FORMAT: WRITE [LIT=1 BITS] [REVERSE] FROM [T] [X] [INC] [FA] [DEC] [FL] [Y] [DEC] [FL] AND [INC] [FA] [L]

DESCRIPTION:

INITIATES A MAIN MEMORY WRITE CYCLE FROM THE SPECIFIED REGISTER (T, X, Y, L). FA MUST HAVE BEEN SET TO THE APPROPRIATE S-MEMORY ADDRESS PREVIOUSLY.

LITERAL=1, THE NUMBER OF BITS TO WRITE AND THE NUMBER TO INC OR DEC, IS OPTIONAL. LITERAL=1 MUST BE INTEGER, 0 THROUGH 24. IF LITERAL=1 IS OMITTED OR IF IT EQUALS ZERO, THEN CPL INDICATES THE NUMBER OF BITS TO WRITE AND THE NUMBER TO INC OR DEC. IF CPL ALSO EQUALS ZERO, THEN NO OPERATION TAKES PLACE. WRITE "CPL BITS" IS NOT PERMITTED.

IF LESS THAN 24 BITS ARE WRITTEN, THE REGISTER IS TRUNCATED FROM THE LEFT. THE CONTENTS OF THE REGISTER ARE UNCHANGED BY THIS INSTRUCTION.

IF THE "REVERSE" OPTION IS USED, THEN FA POINTS AT THE LAST BIT + 1 OF THE S MEMORY FIELD INVOLVED. THE CONTENTS OF THE SOURCE REGISTER ARE STILL PLACED IN MEMORY AS THOUGH THEY HAD BEEN WRITTEN IN A FORWARD DIRECTION. FOR EXAMPLE, "WRITE 24 BITS FROM T" WITH FA = ZERO AND "WRITE 24 BITS REVERSE FROM T" WITH FA = 24, WILL BOTH RESULT IN THE MEMORY FIELD BEING SET TO THE SAME VALUE, I.E. T(0) CORRESPONDS TO ADDRESS ZERO, AND T(23) CORRESPONDS TO ADDRESS 23.

INCREMENTING OR DECREMENTING OCCURS FOLLOWING THE READ AND IS OPTIONAL. FOR DETAILS SEE THE "COUNT" VERB.

TIME: AT LEAST 4 CLOCKS IF FREE STANDING

XCH

<from> <to>

FORMAT: XCH [DOUBLE-WORD-1] F [DOUBLE-WORD-2]

DESCRIPTION:

MOVES F TO DOUBLE-SCRATCHPAD-WORD-2; AND DOUBLE-SCRATCHPAD-WORD-1 IS MOVED TO F. THE TWO WORDS MAY BE THE SAME.

THE DOUBLE-SCRATCHPAD-WORDS:

S0	S4	S8	S12
S1	S5	S9	S13
S2	S6	S10	S14
S3	S7	S11	S15

TIME: ONE CLOCK

PSEUDO OPERATORS AFFECTING COMPILER OPERATION

ADJUST

```
                                        [LITERAL-1]
FORMAT 1:     ADJUST LOCATION TO        [ADDRESS(LABEL-1)]
                                        [MOD LITERAL-2]


                                                [PLUS]
FORMAT 2:     ADJUST LOCATION TO LOCATION       [+]      LITERAL-3
                                                [MINUS]
                                                [-]
```

DESCRIPTION:

THIS PSEUDO-OPERATOR IS USED TO ADJUST THE COMPILERS LOCATION COUNTER. THE VALUE OF THE LOCATION COUNTER SPECIFIES THE LOCATION(M MEMORY ADDRESS) INTO WHICH THE NEXT GENERATED MICRO INSTRUCTION IS TO BE PLACED

THE "LITERAL-1" OPTION SETS THE LOCATION COUNTER TO LITERAL-1.

THE "ADDRESS(LABEL-1)" OPTION SETS THE LOCATION COUNTER TO THE ADDRESS ASSOCIATED WITH LABEL-1. LABEL-1 MUST HAVE BEEN DECLARED PREVIOUS TO THE EXECUTION OF THIS PSEUDO-INSTRUCTION.

THE "MOD LITERAL-2" OPTION CAUSES THE LOCATION COUNTER TO BE INCREMENTED (IF NECESSARY) UNTIL ITS VALUE, MODULO LITERAL-2, IS EQUAL TO ZERO. THIS FEATURE IS USEFUL WHEN IT IS DESIRED TO GENERATE TABLES WHICH CONTAIN MORE THAN 128 ELEMENTS.

THE "LOCATION PLUS LITERAL-3" AND "LOCATION MINUS LITERAL-3" OPTIONS CAUSE THE LOCATION COUNTER TO BE INCREMENTED OR DECREMENTED, RESPECTIVELY, BY LITERAL-3.

THE VALUE OF THE LOCATION COUNTER MUST NOT EXCEED 8191(OR H1FFF).

PROGID

FORMAT: PROGID = IDENTIFIER-1

DESCRIPTION:

IF THE "PUNCH" OPTION IS USED ON A COMPILER OPTION $ CARD(SEE APPENDIX A), THEN IDENTIFIER-1 WILL BE PUNCHED INTO COLUMNS 72 THRU 80 ON ALL OF THE OBJECT DECK CARDS. IF A PROGID STATEMENT IS USED, IT SHOULD BE THE FIRST STATEMENT OF THE PROGRAM.

IDENTIFIER-1 MAY BE MADE UP OF ALPHA(A THRU Z) OR NUMERIC(0 THRU 9) CHARACTERS IN ANY COMBINATION. THE SPECIAL CHARACTER DASH(-) MAY ALSO BE USED, BUT IT MAY NOT BE THE FIRST CHARACTER. IF IDENTIFIER-1 IS NOT 9 CHARACTERS IN LENGTH, IT WILL BE TRUNCATED FROM THE RIGHT OR RIGHT JUSTIFIED WITH LEADING BLANK FILL.

PAGE

FORMAT: PAGE

DESCRIPTION:

THIS PSEUDO-OPERATOR CAUSES THE COMPILER TO SKIP THE LINE-PRINTER LISTING TO THE TOP OF THE NEXT PAGE.

SEGMENT

FORMAT:

```
                    [NEWSEGMENT]      [LITERAL-1]
      SEGMENT [LABEL-1]     AT  [ADDRESS(LABEL-2)]
```

DESCRIPTION:

THIS PSEUDO OPERATOR IS USEFUL WHEN A SECTION OR SEGMENT OF MICRO-CODE IS TO BE PHYSICALLY PLACED AT THE END OF A MAINLINE INTERPRETER OR EMULATOR AT COMPILE TIME, BUT AT RUN TIME WILL BE OVERLAYED INTO THE MAINLINE ROUTINE IN M MEMORY AT THE LOCATION SPECIFIED BY LITERAL-1 OR "ADDRESS(LABEL-2)".

LABEL-1, IF USED, MUST BE A UNIQUE LABEL, I.E. IT MUST NOT BE A POINT LABEL. THE ADDRESS ASSOCIATED WITH LABEL-1 WILL BE THE PHYSICAL LOCATION OF THE FIRST MICRO-INSTRUCTION FOLLOWING THE SEGMENT STATEMENT.

THE "NEWSEGMENT" OPTION MAY BE USED IN LIEU OF LABEL-1 IF IT IS NOT NECESSARY TO NAME THE SEGMENT, E.G. A TEST ROUTINE WHICH IS TO BE EXECUTED FROM THE CASSETTE.

LITERAL-1 OR "ADDRESS(LABEL-2)" MUST SPECIFY AN ADDRESS WHICH IS LESS THAN THE ADDRESS ASSOCIATED WITH LABEL-1. LABEL-2, IF USED, MUST HAVE BEEN DECLARED PREVIOUS TO THE OCCURRENCE OF THE SEGMENT STATEMENT AND MUST BE ENCLOSED IN PARENTHESES.

ALL MICRO-CODE APPEARING AFTER A SEGMENT STATEMENT IS ASSUMED TO BE IN THAT SEGMENT UNTIL ANOTHER SEGMENT STATEMENT IS FOUND.

THERE IS NO LIMIT TO THE NUMBER OF SEGMENT STATEMENTS THAT MAY APPEAR IN ONE PROGRAM.

LABELS OUTSIDE A SEGMENT MAY BE REFERENCED FROM WITHIN THE SEGMENT, AND VICE-VERSA, WITH THE CORRECT RUN TIME BRANCH ADDRESSES BEING GENERATED.

PSEUDO OPERATORS AFFECTING THE B1500 SIMULATOR

DUMP

FORMAT: DUMP [M] [S] MEMORY

DESCRIPTION:

THIS PSEUDO-OPERATOR CAUSES THE B1500 SIMULATOR(SEE APPENDIX B) TO DUMP AN IMAGE OF THE CONTENTS OF THE SIMULATORS M OR S MEMORY ARRAYS TO THE LINE PRINTER. THIS ACTION OCCURS WHETHER OR NOT THE SIMULATORS TRACE OPTION IS SET.

THE CONTENTS OF THE SIMULATORS M OR S MEMORY ARRAYS ARE NOT CHANGED BY THIS PSEUDO-OPERATOR AND THE SIMULATOR WILL CONTINUE TO RUN.

TRACE

FORMAT: TRACE [ON] / [OFF]

DESCRIPTION:

THIS PSEUDO-OPERATOR IS USED TO TURN THE B1500 SIMULATORS TRACE OPTION ON OR OFF(SEE APPENDIX B).

WHEN THE SIMULATORS TRACE OPTION IS SET(ON), A PRINTED LISTING, SHOWING THE RESULTS OF EACH M-OPERATOR AS IT IS "EXECUTED", IS PRODUCED UNTIL THE TRACE OPTION IS RESET(OFF) OR UNTIL A "HALT" OPERATOR IS "EXECUTED".

"TRACE" OR "TRACE ON" WILL TURN THE TRACE OPTION ON, WHILE "TRACE OFF" WILL TURN IT OFF.

APPENDIX A - MIL COMPILER OPERATION

PROGRAM NAME

MIL/DISK(RUN ON B5500).

FILE NAMES

CARD	INPUT SOURCE CARD READER FILE.
TAPE	INPUT SOURCE TAPE FILE.
DISK	INPUT SOURCE DISK FILE.
NEWTAPE	OUTPUT SOURCE TAPE FILE.
NEWDISK	OUTPUT SOURCE DISK FILE.
LINE	OUTPUT LISTING LINE PRINTER FILE.
PUNCH	OUTPUT OBJECT CODE CARD PUNCH FILE.

PURPOSE

THE MIL COMPILER PRODUCES M-STRING OBJECT CODE FROM A SOURCE PROGRAM WHICH HAS BEEN WRITTEN IN THE MICRO IMPLEMENTATION LANGUAGE(MIL). IT ALSO CAN PRODUCE A PRINTED LISTING OF THE SOURCE INPUT AND THE GENERATED OBJECT CODE.

OPERATING INSTRUCTIONS

THE MIL COMPILER CAN BE CALLED FROM THE CARD READER WITH THE FOLLOWING SET OF CONTROL AND DATA CARDS:

```
? COMPILE [PROG-NAME]/[USER-NAME] WITH MIL LIBRARY #[CC] [PROJ #].
? FILE TAPE = [FILE-NAME]/[USER-NAME]
? FILE NEWTAPE = [FILE-NAME]/[USER-NAME]
? DATA CARD
(OPTIONAL $ CARDS)
(SOURCE PROGRAM OR UPDATES IF $ CARD SPECIFIES TAPE OR DISK)
? END
```

ALL CARDS(INCLUDING SOURCE CARDS) MUST BE PUNCHED IN BCL IF THEY ARE TO BE READ ON THE B5500.

$ CARDS

THERE ARE THREE TYPES OF $ CARDS ACCEPTABLE TO THE COMPILER. THESE ARE THE COMPILER OPTION, MODULE OPTION, AND VOID $ CARDS. THESE $ CARDS ARE ALL FREE FORM EXCEPT THAT A $ MUST BE PLACED IN COLUMN 1, ALL ITEMS MUST BE SEPARATED BY AT LEAST ONE BLANK, COLUMNS 73 THRU 80 ARE RESERVED FOR AN OPTIONAL SEQUENCE NUMBER, AND EXCEPT AS NOTED IN THE FOLLOWING DESCRIPTIONS OF EACH TYPE.

THE COMPILER OPTION $ CARD

THIS CARD SPECIFIES WHICH COMPILER OPTIONS ARE TO BE USED. THE FIRST OPTION FOLLOWING THE $ MUST BE CARD, TAPE, OR DISK. THE FOLLOWING OPTIONS ARE AVAILABLE:

CARD	SOURCE INPUT IS ENTIRELY FROM CARDS.
TAPE	PRIMARY SOURCE INPUT IS FROM MAGNETIC TAPE.
DISK	PRIMARY SOURCE INPUT IS FROM DISK.
NEWTAPE	PRODUCES UPDATED SOURCE OUTPUT ON TAPE.
NEWDISK	PRODUCES UPDATED SOURCE OUTPUT ON DISK.
LIST	PRODUCES DOUBLE SPACED LINE PRINTER LISTING OF SOURCE IMAGES AND THE FIRST ADDRESS AND FIRST WORD OF OBJECT CODE ASSOCIATED WITH A SOURCE IMAGE. IF NO LISTING IS ASKED FOR, THEN ONLY ERROR AND WARNING MESSAGES, TOGETHER WITH THEIR ASSOCIATED SOURCE IMAGES, WILL BE LISTED.
LIST1	SAME AS "LIST", BUT WITH SINGLE SPACING.
DEBUG	CAUSES ALL GENERATED OBJECT CODE AND ASSOCIATED ADDRESSES AND ALL SEGMENT NAMES AND ABSOLUTE ADDRESSES TO BE LISTED. ALSO CAUSES "LIST" TO BE ASSUMED UNLESS "LIST1" HAS BEEN SPECIFIED.
EXPAND	CAUSES ALL SOURCE STATEMENTS MAKING UP A MACRO TO BE LISTED EACH TIME A MACRO IS REFERENCED. ALSO CAUSES THE "LIST" OPTION TO BE ASSUMED UNLESS "LIST1" HAS BEEN SPECIFIED.
PRINTNOW	CAUSES LISTING TO BE PRINTED IMMEDIATELY DURING A COMPILATION WHENEVER THERE ARE NO LABELS WHOSE ADDRESSES ARE STILL UNRESOLVED. WITHOUT THIS OPTION,

THE ENTIRE LISTING WILL BE PRINTED AT THE END OF THE COMPILATION.

SEQ — CAUSES UPDATED SOURCE OUTPUT AND/OR LISTING TO BE RESQUENCED. IF THE ITEM FOLLOWING "SEQ" IS AN INTEGER, THEN THIS INTEGER IS USED AS THE RESEQUENCE STARTING VALUE; OTHERWISE 1000 IS USED. IF THE ITEM FOLLOWING THE ABOVE INTEGER IS ANOTHER INTEGER WHOSE FIRST CHARACTER IS A PLUS SIGN(+), THEN THIS SECOND INTEGER IS USED AS THE RESEQUENCE INCREMENT VALUE; OTHERWISE 1000 IS USED.

NOCHECK — NO SEQUENCE CHECKING WILL BE PERFORMED ON THE INPUT SOURCE FILE(S). IF THIS OPTION IS NOT SPECIFIED, A WARNING MESSAGE WILL BE PRINTED IF A SOURCE IMAGE WITH A SEQUENCE NUMBER WHICH IS LESS THAN THE PREVIOUS SOURCE IMAGES SEQUENCE NUMBER IS ENCOUNTERED. THIS IS NOT, HOWEVER, CONSIDERED TO BE AN ERROR.

SUPPRESS — CAUSES ALL WARNING MESSAGES, EXCEPT "OUT OF SEQUENCE", TO BE SUPPRESSED ON THE OUTPUT LISTING.

PUNCH — CAUSES OBJECT CODE TO BE PUNCHED ON CARDS. THE CARDS HAVE THE FOLLOWING FORMAT, WITH ALL FIELDS, EXCEPT THE PROGRAM IDENTIFIER, IN HEXADECIMAL FORMAT(0-9,A-F):

CARD COLUMNS	
1 THRU 6	24 BIT M-MEMORY ADDRESS.
8 AND 9	8 BIT COUNT OF THE NUMBER OF BITS OF DATA.
11 THRU 70	UP TO 240 BITS OF DATA, LEFT JUSTIFIED.
72 THRU 80	PROGRAM IDENTIFIER FROM "PROGID" STATEMENT, RIGHT JUSTIFIED AND FOR DOCUMENTATION ONLY.

IF MORE THAN ONE COMPILER OPTION $ CARD IS USED DIRECTLY FOLLOWING THE "? DATA CARD" CARD, THEN ONLY THE LAST OF THESE WILL BE VISIBLE TO THE COMPILER.

ALL COMPILER OPTION $ CARDS ENCOUNTERED AFTER THE FIRST NON $ CARD ARE ALL VISIBLE TO THE COMPILER. A SOURCE INPUT MEDIA OPTION ("CARD", "TAPE" OR "DISK") MUST AGAIN BE THE FIRST ITEM FOLLOWING THE $; HOWEVER, ALL INPUT AND OUTPUT("NEWTAPE" AND "NEWDISK") SOURCE MEDIA OPTIONS ARE OTHERWISE IGNORED ON THESE LATER COMPILER OPTION $ CARDS. ANY OTHER OPTION DESIRED ON THESE LATER COMPILER OPTION $ CARDS, EXCEPT "PUNCH" WHICH WILL REMAIN IN EFFECT ONCE IT IS SET, MUST BE EXPLICITLY DECLARED OR IT WILL BE IMPLICITLY RESET.

THE DEFAULT COMPILER OPTION $ CARD IS "$ CARD LIST".

EXAMPLES:

```
$ CARD LIST1
$ CARD NEWDISK LIST SEQ 100 +100
$ DISK NOCHECK SUPPRESS
$ TAPE NEWTAPE EXPAND DEBUG PUNCH
```

THE MODULE OPTION $ CARD

THE MODULE OPTION $ CARD IS USED TO SET OR RESET USER DEFINED TOGGLES TO BE USED IN CONJUNCTION WITH THE "IF" STATEMENT FOR CONDITIONAL INCLUSION OF SOURCE STATEMENTS. IT MAY BE USED ANYWHERE WITHIN THE SOURCE DECK AND EACH MODULE OPTION $ CARD EFFECTS ONLY THOSE USER TOGGLES WHICH ARE REFERENCED ON THAT CARD. BEFORE ANY TOGGLE CAN BE REFERENCED BY AN "IF" STATEMENT, IT MUST BE DECLARED ("SET" OR "RESET") ON A MODULE OPTION $ CARD.

EXAMPLES:

```
$ SET SYSTEM1
$ RESET SYSTEM2, RESET SYSTEM3, RESET SYSTEM4
$ SET SW1, RESET SW2, RESET SW3, SET SW4, SET SW5
```

THE VOID $ CARD

THE SOURCE IMAGE(TAPE OR DISK ONLY) WITH A SEQUENCE NUMBER EQUAL TO THE SEQUENCE NUMBER OF A VOID $ CARD WILL BE IGNORED BY THE COMPILER AND WILL NOT BE COPIED OUT TO ANY UPDATED SOURCE FILE. IF AN 8 DIGIT INTEGER(LEADING ZEROS REQUIRED) FOLLOWS "VOID", THEN THE ABOVE ACTION WILL BE CONTINUED UNTIL A SOURCE IMAGE WITH A SEQUENCE NUMBER GREATER THAN THIS NUMBER IS ENCOUNTERED. THIS CARD MAY BE USED ANYWHERE WITHIN THE SOURCE INPUT, BUT IT MUST BE USED IN SEQUENCE.

EXAMPLES:

```
$ VOID                          00120000
$ VOID   00139900               00120000
```

APPENDIX B - B1500 SIMULATOR OPERATION

PROGRAM NAME

B1500/SIM (RUN ON B5500).

FILE NAMES

M	M-MEMORY INPUT DISK FILE, NORMALLY THE OBJECT CODE OUTPUT FROM THE MIL COMPILER.
S	S-MEMORY INPUT DISK FILE, NORMALLY THE OBJECT CODE OUTPUT FROM THE HIL, COBOL AND FORTRAN COMPILERS.
LINE	OUTPUT TRACE LISTING LINE PRINTER FILE.
DISCARD	INPUT DISK FILE FOR SIMULATED EBCDIC CARD READ.
BINCARD	INPUT DISK FILE FOR SIMULATED BINARY CARD READ.

PURPOSE

B1500/SIM IS USED TO SIMULATE THE OPERATION OF THE B1500 MICRO PROCESSOR ON THE B5500. IT ALSO PRODUCES A PRINTED TRACE LISTING OF THE RESULTS OF EACH INSTRUCTION AS IT IS SIMULATED.

OPERATING INSTRUCTIONS

B1500/SIM CAN BE CALLED FROM THE CARD READER WITH THE FOLLOWING SET OF CONTROL CARDS:

```
? EXECUTE B1500/SIM #[COST-CENTER] [PROJECT-NUMBER]
? FILE M = [PROGRAM-NAME]/[USER-NAME]
? FILE S = S/[USER-NAME]
? END
```

NOTE: BOTH THE M AND S FILES ARE REQUIRED FILES.

IF A "? COMMON = 1" CARD IS USED FOLLOWING THE EXECUTE CARD, TRACE WILL BE TURNED ON; AND ALL INSTRUCTIONS WILL BE TRACED UNTIL A "TRACE OFF" PSUEDO INSTRUCTION IS ENCOUNTERED. ALTERNATIVELY, TRACE CAN BE TURNED ON WITH THE "TRACE" PSEUDO; HOWEVER, THESE PSEUDOS MUST BE ASSEMBLED INTO THE M FILE BY THE COMPILER.

ALL CARDS MUST BE PUNCHED IN BCL IF THEY ARE TO BE READ ON THE B5500.

FYI:
THE TRACE TOGGLE IS LOCATED AT PRT 25.

THE DUMP TOGGLE IS LOCATED AT PRT 26
WHERE 1 = DUMP M MEMORY AND 2 = DUMP S MEMORY.

THE B1500 CLOCK IS LOCATED AT PRT 27.