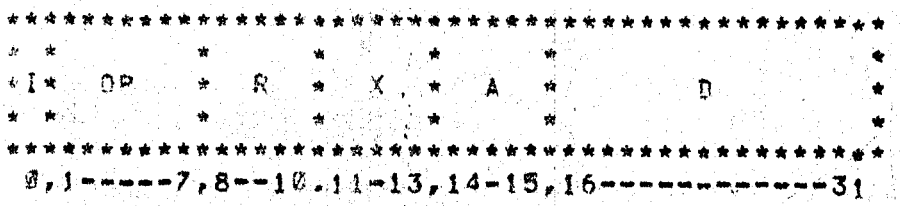
HP OMEGA INSTRUCTION SET

SEPTEMBER 19, 1968

FORMATS:

INSTRUCTION FORMAT



I = INDIRECT BIT

OP = OPERATION CODE (0-127)

R = REGISTER REFERENCE (FIRST OPERAND)

X = REGISTER REFERENCE (INDEX REGISTER OR SECOND OPERAND)

A = ADDRESSING CONVENTION

D = DISPLACEMENT FIELD

MEMORY ADDRESS (M) COMPUTATION:

IF A = 0, SECOND OPERAND IS REGISTER X
INDIRECT BIT IS IGNORED
NO VALIDITY CHECKING.

IF A = 1, ADDRESS RELATIVE TO PROGRAM COUNTER (P).
INDIRECT ADDRESSING PERMITTED
X SPECIFIES AN INDEX REGISTER
M = (P+X+D) MODULO 65K
VALIDITY CHECK: B<=M<=MPR

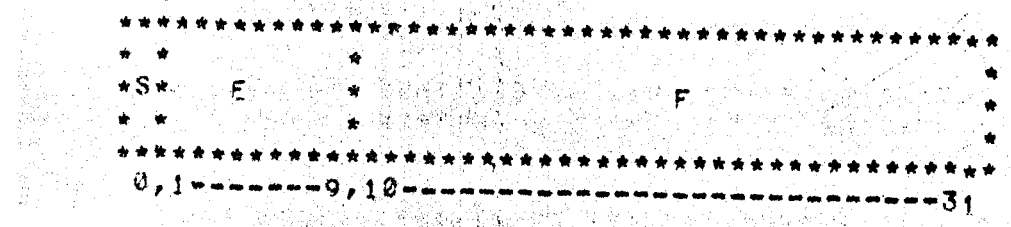
IF A = 2, ADDRESS RELATIVE TO STACK MARKER (Q)
INDIRECT ADDRESSING PERMITTED
X SPECIFIES AN INDEX REGISTER
M = (Q+X+D) MODULO 65K
VALIDITY CHECK: B<=M<=Z

IF A = 3, ADDRESS RELATIVE TO BASE REGISTER (B)
INDIRECT ADDRESSING PERMITTED
X SPECIFIES AN INDEX REGISTER
M = (B+X+D) MODULO 65K
VALIDITY CHECK: B<=M<=MPR

(NOTE: ADDRESS ARITHMETIC MODULO 65K PERMITS POSITIVE OR NEGATIVE DISPLACEMENT FROM P, Q, OR B.)

IF INDIRECT BIT = 1 AND A = 1, 2, OR 3:
FETCH THE WORD ADDRESSED BY D + (P, Q, OR B).
IF THE LEFTMOST BIT OF THIS WORD IS 1,
ADD B TO OTHER 31 BITS, AND FETCH THE
WORD ADDRESSED BY THE RESULTING SUM.
REPEAT THIS CYCLE UNTL A WORD IS OBTAINED
WHOSE LEFTMOST BIT IS Ø. ADD B+X TO THIS
WORD. THE RESULT IS M, THE OPERAND ADDRESS.

FLOATING POINT NUMBER FORMAT



$1.15*10^{77}$
$8.6*10^{-78}$

S = FRACTION SIGN BIT (Ø FOR POSITIVE, 1 FOR NEGATIVE)

E = EXPONENT, EXCESS 256 (RANGE Ø-511)

F = FRACTION (RANGE Ø TO (2↑22) - 1)

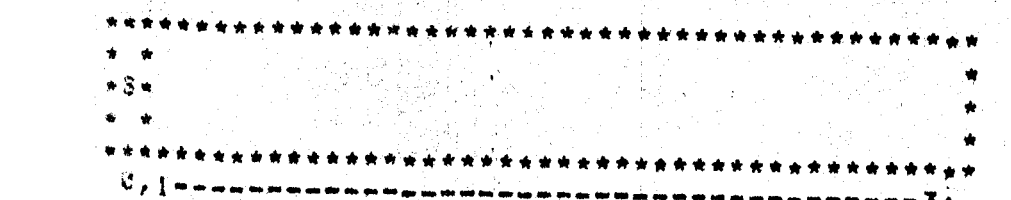
DECIMAL VALUE = (-1) ↑ S * 2 ↑ (E-256) * (1+F*2↑(-22))

DECIMAL VALUE = Ø WHEN E = F = Ø

NOTE THAT THE FRACTION FIELD HAS AN IMPLIED LEADING 1
TO THE LEFT OF THE BINARY POINT. THUS ALL FLOATING
POINT NUMBERS ARE STORED IN NORMALIZED FORM, BUT NO BIT
IS WASTED ON THE LEADING 1, SO ALL FRACTION BITS ARE
SIGNIFICANT.

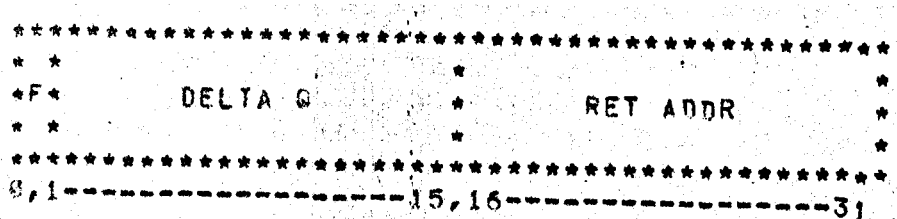
(FOR DOUBLE-PRECISION FLOATING POINT FORMAT 32 FRACTION
BITS ARE ADDED ON THE RIGHT.)

FIXED POINT FORMAT: 32 BIT, TWO'S COMPLEMENT



(FOR DOUBLE-WORD FIXED POINT FORMAT, 32 MAGNITUDE BITS
ARE ADDED ON THE RIGHT.)

STACK MARKER FORMAT



F = FLAG: 0 FOR PROCEDURE MARKERS
1 FOR SYSTEM CALL STACK MARKERS OR INTERRUPT STACK MARKERS.

DELTA Q: CONTAINS THE NUMBER BY WHICH THE Q REGISTER MUST BE DECREMENTED TO ARRIVE AT THE NEXT LOWER STACK MARKER.

RET ADDR: POINTS TO RETURN ADDRESS OF THE PROCEDURE OR SYSTEM CALL RELATIVE TO BASE ADDRESS

CONDITION CODE PATTERNS:

AT ALL TIMES, EXACTLY ONE CONDITION CODE BIT IS ON. THE BITS ARE DESIGNATED 0, 1, AND 2.

PATTERN A:

CC = 0 IF RESULT = 0
CC = 1 IF RESULT > 0
CC = 2 IF RESULT < 0

PATTERN B:

CC = 0 IF ALL RESULT BITS ARE 0
CC = 1 IF ANY RESULT BITS ARE 1

PATTERN C:

CC = 0 IF OPERANDS ARE EQUAL
CC = 1 IF OPERAND 1 > OPERAND 2
CC = 2 IF OPERAND 1 < OPERAND 2

ISTERS:

```
**********************************
*        R0, NO INDEXING         *
**********************************
*            R1, X1              *
**********************************
*            R2, X2              *
**********************************
*            R3, X3              *
**********************************
*            R4, X4              *
**********************************
*            R5, X5              *
**********************************
*            R6, X6              *
**********************************
*            R7, X7              *
**********************************
```

THE EIGHT GENERAL-PURPOSE 32-BIT REGISTERS MAY BE USED AS OPERANDS OR AS INDEX REGISTERS. IF INDEX REGISTER 0 IS SPECIFIED, NO INDEXING OCCURS.

B: BASE REGISTER; 16 BITS
CONTAINS THE ABSOLUTE ADDRESS OF THE FIRST LOCATION OF THE PROGRAM BEING EXECUTED.

S: STACK POINTER REGISTER; 16 BITS
CONTAINS THE ABSOLUTE ADDRESS OF THE LAST USED CORE LOCATION OF THE STACK ("TOP" OF THE STACK).

C: STACK MARKER POINTER; 16 BITS
CONTAINS THE ABSOLUTE ADDRESS OF THE LAST STACK MARKER USED WITHIN THE STACK.

Z: STACK TOP POINTER; 16 BITS
CONTAINS THE ABSOLUTE ADDRESS OF THE LAST POSSIBLE STORAGE LOCATION OF THE STACK.

P: PROGRAM COUNTER; 16 BITS
CONTAINS THE ABSOLUTE ADDRESS OF THE INSTRUCTION BEING EXECUTED.

MPR: MEMORY PROTECT REGISTER; 16 BITS
CONTAINS THE LAST ABSOLUTE ADDRESS OF UNPROTECTED CORE.

WOULD LIKE
UPPER & LOWER
BOUNDS

CHINE INSTRUCTIONS:

THE MACHINE HAS TWO MODES: EXECUTIVE MODE, AND USER MODE. PRIVILEGED INSTRUCTIONS MAY BE EXECUTED ONLY IN EXECUTIVE MODE. IF A PRIVILEGED INSTRUCTION IS ENCOUNTERED IN USER MODE, AN INTERRUPT OCCURS.

THE FOLLOWING EXPLANATIONS,

R = THE REGISTER SPECIFIED IN THE R FIELD.

X = THE REGISTER SPECIFIED IN THE X FIELD.

M = THE RESULT OF NORMAL ADDRESS COMPUTATION, EXCEPT WHERE SPECIFIED OTHERWISE. NOTE THAT M MAY BE A REGISTER OR A MEMORY ADDRESS.

(A) = REGISTER P, Q, OR 8, DEPENDING ON WHETHER A = 1, 2, OR 3.

D = CONTENTS OF D FIELD, BITS 16-31. USED FOR IMMEDIATE OPERANDS.

D1 = CONTENTS OF D FIELD, BITS 16-23.

D2 = CONTENTS OF D FIELD, BITS 24-31.

LOAD R,M (LOAD)

REGISTER R IS LOADED WITH THE CONTENT OF M.
CC = PATTERN A.

LLM R,M (LOAD LEFT HALF)

BITS 0-15 OF THE CONTENT OF M IS LOADED INTO BITS 16-31 OF R.
BITS 0-15 OF R ARE SET TO ZERO.
CC = PATTERN A ON THE RESULTING REGISTER R.

LRM R,M (LOAD RIGHT HALF)

BITS 15-31 OF THE CONTENT OF M IS LOADED INTO BITS 16-31 OF R.
BITS 0-15 OF R ARE SET TO ZERO.
CC = PATTERN A ON THE RESULTING REGISTER R.

STOR M,M (STORE)

THE CONTENT OF R IS STORED IN M.
CC = PATTERN A.

SLM R,M (STORE LEFT HALF)

BITS 16-31 OF R ARE STORED IN BITS 0-15 OF M.
BITS 16-31 OF M ARE UNCHANGED.
CC = PATTERN A ON THE RESULTING WORD M.

5.

STH R,M (STORE RIGHT HALF)

BITS 16-31 OF R ARE STORED IN BITS 16-31 OF M.
BITS 0-15 OF M ARE UNCHANGED.
CC = PATTERN A ON THE RESULTING WORD M.

6.

XCH R,M (EXCHANGE)

THE CONTENT OF R AND THE CONTENT OF M ARE EXCHANGED.
CC = PATTERN A ON THE WORD LOADED INTO R FROM M.

METIC AND LOGIC INSTRUCTIONS:

7.

ADD R,M (ADD)

THE CONTENT OF M IS ADDED TO REGISTER R IN FIXED POINT FORM.
CC = PATTERN A.

8.

SUB R,M (SUBTRACT)

THE CONTENT OF M IS SUBTRACTED FROM REGISTER R IN FIXED POINT FORM.
CC = PATTERN A.

9.

MPY R,M (MULTIPLY)

THE CONTENT OF R IS MULTIPLIED BY THE CONTENT OF M IN FIXED POINT FORM, AND THE DOUBLE-LENGTH PRODUCT IS LEFT IN R AND R+1.
CC = PATTERN A.

12.

DIV R,M (DIVIDE)

THE DOUBLE-LENGTH INTEGER IN REGISTERS R AND R+1 IS DIVIDED BY THE CONTENT OF M. THE QUOTIENT INTEGER IS LEFT IN R, AND THE REMAINDER IN R+1.
CC = PATTERN A ON THE QUOTIENT.

N R,M (AND)

THE CONTENT OF R IS REPLACED BY THE LOGICAL "AND" OF R AND M.
CC = PATTERN B.

12.

OR R,M (INCLUSIVE OR)

THE CONTENT OF R IS REPLACED BY THE "INCLUSIVE OR" OF R AND M.
CC = PATTERN B.

13.

XOR R,M (EXCLUSIVE OR)

THE CONTENT OF R IS REPLACED BY THE "EXCLUSIVE OR" OF R AND M.
CC = PATTERN B.

14.

A R,M (ADD TO MEMORY)

THE CONTENT OF R IS ADDED TO THE CONTENT OF M IN FIXED POINT FORM.
CC = PATTERN A.

15.

S R,M (SUBTRACT FROM MEMORY)

THE CONTENT OF R IS SUBTRACTED FROM THE CONTENT OF M IN FIXED POINT FORM.
CC = PATTERN A.

16.

N R,M (AND MEMORY)

THE CONTENT OF M IS REPLACED BY THE LOGICAL "AND" OF M AND R.
CC = PATTERN B.

17.

O R,M (INCLUSIVE OR MEMORY)

THE CONTENT OF M IS REPLACED BY THE "INCLUSIVE OR" OF R AND M.
CC = PATTERN B.

18.

XORM R,M (EXCLUSIVE OR MEMORY)

THE CONTENT OF M IS REPLACED BY THE "EXCLUSIVE OR" OF R AND M.
CC = PATTERN B.

FLOATING POINT INSTRUCTIONS:

18.

FADD R,M (FLOATING ADD)

THE CONTENT OF M IS ADDED TO REGISTER R IN FLOATING POINT.
CC = PATTERN A.

19.

FSUB R,M (FLOATING SUBTRACT)

THE CONTENT OF M IS SUBTRACTED FROM REGISTER R IN FLOATING POINT.
CC = PATTERN A.

20.

FMPY R,M (FLOATING MULTIPLY)

THE CONTENT OF R IS MULTIPLIED BY THE CONTENT OF M IN FLOATING POINT.
CC = PATTERN A.

21.

FDV R,M (FLOATING DIVIDE)

THE CONTENT OF R IS DIVIDED BY THE CONTENT OF M IN FLOATING POINT.
CC = PATTERN A.

22.

FLT R,M (FLOAT)

THE INTEGER IN M IS FLOATED AND PLACED IN R.
CC = PATTERN A.

23.

FIX R,M (FIX)

THE FLOATING POINT NUMBER IN M IS CONVERTED TO FIXED POINT AND PLACED IN R.
CC = PATTERN A.

ARY OPERATORS:

25.

INC M (INCREMENT)

THE CONTENT OF M IS INCREMENTED BY 1.
CC = PATTERN A.

26.

DEC M (DECREMENT)

THE CONTENT OF M IS DECREMENTED BY 1.
CC = PATTERN A.

27.

(NEGATE)

THE CONTENT OF M IS NEGATED.
CC = PATTERN A.

28.

(ABSOLUTE VALUE)

THE CONTENT OF M IS REPLACED BY ITS ABSOLUTE VALUE.
CC = PATTERN A.

29.

(COMPLEMENT)

THE CONTENT OF M IS REPLACED BY ITS ONE'S COMPLEMENT.
CC = PATTERN B.

30.

ZERO M (ZERO)

THE CONTENT OF M IS SET TO ZERO.
CC = UNAFFECTED.

31.

TEST M (TEST)

THE CONDITION CODE IS SET TO PATTERN A DEPENDING ON THE CONTENT OF M.

DOUBLEWORD INSTRUCTIONS:

32.

DLOD R,M (DOUBLE LOAD)

IN ADDRESS COMPUTATION, THE INDEX REGISTER CONTAINS A DOUBLE-WORD COUNT RATHER THAN A WORD COUNT. THE WORDS IN M AND M+1 ARE LOADED INTO R AND R+1.
CC = PATTERN A ON THE DOUBLE-WORD RESULT.

33.

DSTO R,M (DOUBLE STORE)

IN ADDRESS COMPUTATION, THE INDEX REGISTER CONTAINS A DOUBLE-WORD COUNT RATHER THAN A WORD COUNT. THE WORDS IN R AND R+1 ARE STORED IN M AND M+1.
CC = UNAFFECTED.

34.

DADD R,M (DOUBLE ADD)

IN ADDRESS COMPUTATION, THE INDEX REGISTER CONTAINS A DOUBLE-WORD COUNT RATHER THAN A WORD COUNT. THE DOUBLE-WORD TWO'S COMPLEMENT INTEGERS IN R, R+1 AND M, M+1 ARE ADDED, AND THE RESULT IS LEFT IN R, R+1.
CC = PATTERN A ON THE DOUBLE-WORD RESULT.

35.

DSUB R,M (DOUBLE SUBTRACT)

IN ADDRESS COMPUTATION, THE IN
WORD COUNT RATHER THAN A WORD
COMPLEMENT INTEGER IN M, M+1 I
AND THE RESULT IS LEFT IN R, R
CC = PATTERN A ON THE DOUBLE-

36.

DCMA R,M (DOUBLE COMPARE)

IN ADDRESS COMPUTATION, THE INDEX REGISTER CO INS A DOUBLE-WORD COUNT RATHER THAN A WORD COUNT. THE DOU -WORD TWO'S COMPLEMENT INTEGERS IN R, R+1 AND M, M+1 ARE C PARED NUMERICALLY.
CC = PATTERN C.

E INSTRUCTIONS:

37.

LDB R,M (LOAD BYTE)

IN ADDRESS COMPUTATION FOR THIS INSTRUCTION, THE INDEX REGISTER IS ASSUMED TO CONTAIN A NUMBER OF BYTES RATHER THAN A NUMBER OF WORDS. THE D FIELD OF THE INSTRUCTION CONTAINS A WORD DISPLACEMENT AS USUAL. THE RESULT OF THE ADDRESS COMPUTATION IS A BYTE ADDRESS M. THE BYTE AT M IS LOADED INTO THE LOW-ORDER 8 BITS OF R, AND THE HIGH-ORDER 24 BITS OF R ARE SET TO ZERO. IF A=0, THE SECOND OPERAND IS THE RIGHTMOST BYTE OF REGISTER X.
CC = PATTERN A ON THE RESULTING REGISTER R.

38.

STB R,M (STORE BYTE)

IN ADDRESS COMPUTATION FOR THIS INSTRUCTION, THE INDEX REGISTER IS ASSUMED TO CONTAIN A NUMBER OF BYTES RATHER THAN A NUMBER OF WORDS. THE D FIELD OF THE INSTRUCTION CONTAINS A WORD DISPLACEMENT AS USUAL. THE RESULT OF THE ADDRESS COMPUTATION IS A BYTE ADDRESS M. THE LOW-ORDER 8 BITS OF REGISTER R ARE STORED INTO M. IF A=0, THE SECOND OPERAND IS THE RIGHTMOST BYTE OF REGISTER X.
CC = UNCHANGED.

39.

MBS R,X (MOVE BYTES)

THE FIELD BEGINNING AT THE RELATIVE BYTE ADDRESS IN REGISTER R IS MOVED INTO THE FIELD BEGINNING AT THE RELATIVE BYTE ADDRESS IN REGISTER R+1. THE BYTE COUNT TO BE MOVED IS IN REGISTER X. IN R AND R+1, THE WORD ADDRESS IS IN BITS 14-29, AND THE BYTE ADDRESS (0-3) IS IN BITS 30-31. REGISTERS R, R+1, AND X ARE STEPPED ALONG AS THE INSTRUCTION IS EXECUTED BYTE-BY-BYTE. AT THE END OF THE INSTRUCTION, R AND R+1 POINT TO THE NEXT BYTE PAST THE END OF THEIR RESPECTIVE FIELDS, AND X CONTAINS ZERO.
CC = UNAFFECTED.

48.

CMB R,X (COMPARE BYTES)

THE FIELD BEGINNING AT THE RELATIVE ADDRESS IN R IS COMPARED BYTE-BY-BYTE WITH THE FIELD BEGINNING AT THE RELATIVE ADDRESS IN R+1. THE BYTE COUNT TO BE COMPARED IS IN REGISTER X. CC=0 IF THE FIELDS COMPARE PERFECTLY; CC=PATTERN C IF ANY INEQUALITY IS FOUND. DURING EXECUTION, R AND R+1 ARE STEPPED ALONG, AND X IS DECREMENTED TO ZERO. AT THE CONCLUSION OF THE INSTRUCTION, R AND X POINT TO THE FIRST BYTES IN THE RESPECTIVE FIELDS WHICH DID NOT COMPARE EQUAL, OR JUST PAST THE RIGHTMOST BYTE OF EACH FIELD IN THE CASE OF EQUAL COMPARISON. IN R AND X, THE WORD ADDRESS IS IN BITS 14-29, AND THE BYTE ADDRESS (0-3) IS IN BITS 30-31.

FULLWORD COMPARES:

41.

NCM R,M (NUMERIC COMPARE)

THE CONTENT OF R IS COMPARED NUMERICALLY WITH THE CONTENT OF M IN FIXED POINT FORM.
CC = PATTERN C.

42.

LCM R,M (LOGICAL COMPARE)

THE CONTENT OF R IS COMPARED LOGICALLY WITH THE CONTENT OF M, AS AN ABSOLUTE 32-BIT MAGNITUDE.
CC = PATTERN C.

43.

FCM R,M (FLOATING COMPARE)

THE CONTENT OF R IS COMPARED WITH THE CONTENT OF M AS FLOATING POINT NUMBERS.
CC = PATTERN C.

BRANCHES:

44.

BCC R,M (BRANCH ON CONDITION CODE)

A BRANCH TO LOCATION M IS EXECUTED UNDER THE FOLLOWING CONDITIONS:

IF R = 0, NEVER BRANCH
R = 1, BRANCH IF CC = 0
R = 2, BRANCH IF CC = 1
R = 3, BRANCH IF CC = 0 OR 1
R = 4, BRANCH IF CC = 2
R = 5, BRANCH IF CC = 0 OR 2
R = 6, BRANCH IF CC = 1 OR 2
R = 7, ALWAYS BRANCH

IF A=0, M IS COMPUTED BY ADDING REGISTER B TO THE LOW-ORDER HALFWORD OF REGISTER X.
CC = UNAFFECTED.

45.

BBO R,M (BRANCH IF REGISTER ODD)

A BRANCH TO LOCATION M IS EXECUTED IF THE NUMBER IN REGISTER R IS ODD.
IF A=0, M IS COMPUTED BY ADDING REGISTER B TO THE LOW-ORDER HALFWORD OF REGISTER X.
CC = UNAFFECTED.

46.

BBE R,M (BRANCH IF REGISTER EVEN)

A BRANCH TO LOCATION M IS EXECUTED IF THE NUMBER IN REGISTER R IS EVEN.
IF A=0, M IS COMPUTED BY ADDING REGISTER B TO THE LOW-ORDER HALFWORD OF REGISTER X.
CC = UNAFFECTED.

BBP R,M (BRANCH IF REGISTER POSITIVE)

A BRANCH TO LOCATION M IS EXECUTED IF THE NUMBER IN REGISTER R IS POSITIVE.
IF A=0, M IS COMPUTED BY ADDING REGISTER B TO THE LOW-ORDER HALFWORD OF REGISTER X.
CC = UNAFFECTED.

49.

BNP R,M (BRANCH IF REGISTER NON-POSITIVE)

A BRANCH TO LOCATION M IS EXECUTED IF THE NUMBER IN REGISTER R IS LESS THAN OR EQUAL TO ZERO.
IF A=0, M IS COMPUTED BY ADDING REGISTER B TO THE LOW-ORDER HALFWORD OF REGISTER X.
CC = UNAFFECTED.

4A.

BNM R,M (BRANCH IF REGISTER NEGATIVE)

A BRANCH TO LOCATION M IS EXECUTED IF THE NUMBER IN REGISTER R IS NEGATIVE.
IF A=0, M IS COMPUTED BY ADDING REGISTER B TO THE LOW-ORDER HALFWORD OF REGISTER X.
CC = UNAFFECTED.

4B.

BNN R,M (BRANCH IF REGISTER NON-NEGATIVE)

A BRANCH TO LOCATION M IS EXECUTED IF THE NUMBER IN REGISTER R IS GREATER THAN OR EQUAL TO ZERO.
IF A=0, M IS COMPUTED BY ADDING REGISTER B TO THE LOW-ORDER HALFWORD OF REGISTER X.
CC = UNAFFECTED.

4C.

BZ R,M (BRANCH IF REGISTER ZERO)

A BRANCH TO LOCATION M IS EXECUTED IF THE NUMBER IN REGISTER R IS ZERO.
IF A=0, M IS COMPUTED BY ADDING REGISTER B TO THE LOW-ORDER HALFWORD OF REGISTER X.
CC = UNAFFECTED.

4D. BNZ R,M (BRANCH IF REGISTER NON-ZERO)

A BRANCH TO LOCATION M IS EXECUTED IF THE NUMBER IN REGISTER R IS NON-ZERO.
IF A=0, M IS COMPUTED BY ADDING REGISTER B TO THE LOW-ORDER HALFWORD OF REGISTER X.
CC = UNAFFECTED.

33.

BOF M (BRANCH ON OVERFLOW)

IF THE OVERFLOW FLIP-FLOP IS ON, THE MACHINE BRANCHES TO M. IF A=0, M IS COMPUTED BY ADDING REGISTER B TO THE LOW-ORDER HALFWORD OF REGISTER X. THE OVERFLOW FLIP-FLOP IS CLEARED. CC = UNAFFECTED.

SLL R,X,D2 (LEFT SHIFT LOGICAL)

SRL R,X,D2 (RIGHT SHIFT LOGICAL)

THE CONTENTS OF REGISTERS R AND X ARE SHIFTED LEFT (RIGHT) TOGETHER D2 BITS (R IS ASSUMED TO BE ON THE LEFT) AND THE VACATED BITS ARE SET TO ZEROS. IF R = X, ONLY ONE REGISTER IS SHIFTED.
CC = UNAFFECTED.

SLA R,X,D2 (LEFT SHIFT ARITHMETIC)

SRA R,X,D2 (RIGHT SHIFT ARITHMETIC)

THE CONTENTS OF REGISTERS R AND X ARE SHIFTED LEFT (RIGHT) TOGETHER D2 BITS (R IS ASSUMED TO BE ON THE LEFT), EXCEPT THAT THE LEFTMOST (SIGN) BIT OF R IS UNCHANGED. VACATED LOW-ORDER BITS ARE SET TO ZERO; VACATED HIGH-ORDER BITS ARE SET TO THE SIGN BIT. IF R = X, ONLY ONE REGISTER IS SHIFTED, AND ITS SIGN BIT IS PRESERVED.
ASL OVERFLOWS IF BIT 1 IS DIFFERENT FROM BIT Ø BEFORE SHIFTING.
CC = UNAFFECTED.

SLDL R,X,D2 (LEFT SHIFT CIRCULAR)

SRDL R,X,D2 (RIGHT SHIFT CIRCULAR)

THE CONTENTS OF REGISTERS R AND X ARE ROTATED LEFT (RIGHT) TOGETHER D2 BITS (R IS ASSUMED TO BE ON THE LEFT).
IF R = X, ONLY ONE REGISTER IS SHIFTED.
CC = UNAFFECTED.

IATE:

.

LDI R,D (LOAD IMMEDIATE)

D IS PLACED IN THE RIGHT HALF OF R.
THE LEFT HALF OF R IS SET TO ZERO.
CC = UNAFFECTED.

1.

LNI R,D (LOAD NEGATIVE IMMEDIATE)

D IS NEGATED AND PLACED IN R.
CC = UNAFFECTED.

2.

ADI R,D (ADD IMMEDIATE)

D IS ADDED TO THE CONTENT OF R.
D IS ASSUMED TO BE A 16-BIT POSITIVE INTEGER.
CC = PATTERN A.

3.

SUBI R,D (SUBTRACT IMMEDIATE)

D IS SUBTRACTED FROM THE CONTENT OF R.
D IS ASSUMED TO BE A 16-BIT POSITIVE INTEGER.
CC = PATTERN A.

4.

MPYI R,D (MULTIPLY IMMEDIATE)

THE CONTENT OF R IS MULTIPLIED BY D. THE DOUBLE-WORD RESULT
IS LEFT IN REGISTERS R AND R+1.
D IS ASSUMED TO BE A 16-BIT POSITIVE INTEGER.
CC = PATTERN A ON THE DOUBLE-WORD RESULT.

5.

IVI R,D (DIVIDE IMMEDIATE)

THE DOUBLE-LENGTH INTEGER IN R, R+1 IS DIVIDED BY D. THE QUOTIENT IS LEFT IN R, AND THE REMAINDER IN R+1.
D IS ASSUMED TO BE A 16-BIT POSITIVE INTEGER.
CC = PATTERN A ON THE QUOTIENT.

66.

LI R,D (AND LEFT IMMEDIATE)

BITS 0-15 OF R ARE REPLACED BY THE LOGICAL "AND" OF D AND BITS 0-15 OF R.
BITS 16-31 OF R ARE UNCHANGED.
CC = PATTERN B ON THE RESULTING REGISTER R.

67.

RI R,D (AND RIGHT IMMEDIATE)

BITS 16-31 OF R ARE REPLACED BY THE LOGICAL "AND" OF D AND BITS 16-31 OF R.
BITS 0-15 OF R ARE UNCHANGED.
CC = PATTERN B ON THE RESULTING REGISTER R.

LI R,D (INCLUSIVE OR LEFT IMMEDIATE)

BITS 0-15 OF R ARE REPLACED BY THE "INCLUSIVE OR" OF D AND BITS 0-15 OF R.
BITS 16-31 OF R ARE UNCHANGED.
CC = PATTERN B ON THE RESULTING REGISTER R.

RI R,D (INCLUSIVE OR RIGHT IMMEDIATE)

BITS 16-31 OF R ARE REPLACED BY THE "INCLUSIVE OR" OF D AND BITS 16-31 OF R.
BITS 0-15 OF R ARE UNCHANGED.
CC = PATTERN B ON THE RESULTING REGISTER R.

70.

XOLI R,D (EXCLUSIVE OR LEFT IMMEDIATE)

BITS 0-15 OF R ARE REPLACED BY THE "EXCLUSIVE OR" OF D AND BITS 0-15 OF R.
BITS 16-31 OF R ARE UNCHANGED.
CC = PATTERN B ON THE RESULTING REGISTER R.

71.

XORI R,D (EXCLUSIVE OR RIGHT IMMEDIATE)

BITS 16-31 OF R ARE REPLACED BY THE "EXCLUSIVE OR" OF D AND BITS 16-31 OF R.
BITS 0-15 OF R ARE UNCHANGED.
CC = PATTERN B ON THE RESULTING REGISTER R.

72.

FLTI R,D (FLOAT IMMEDIATE)

THE INTEGER D IS FLOATED AND PLACED IN R.
CC = PATTERN A.

73.

CI R,D (COMPARE IMMEDIATE)

THE INTEGER IN R IS COMPARED [illegible]RICALLY THE [illegible]-BIT POSITIVE INTEGER IN D.
CC = PATTERN C.

74.

CNI R,D (COMPARE NEGATIVE IMM[illegible]E)

THE INTEGER IN R IS COMPARED NU[illegible]CALL[illegible] H[illegible] [illegible]0'S-COMPLEMENT OF THE 16-BIT INTEGE[illegible]D.
CC = PATTERN C.

75.

ADSI D (ADD TO S IMMEDIATE)

THE POSITIVE INTEGER D IS ADDED TO REGISTER S.
CC = UNAFFECTED.

76.

OSTI D (OR STATUS IMMEDIATE)

THE STATUS HALFWORD IS REPLACED BY ITS INCLUSIVE OR WITH THE D FIELD. IF THE MACHINE IS IN THE USER MODE, ONLY THOSE BITS OF STATUS WHICH ARE CONTROLLED BY THE USER WILL BE AFFECTED.
CC = CHANGED AS PART OF STATUS HALFWORD.

77.

ANDI D (AND STATUS IMMEDIATE)

THE STATUS HALFWORD IS REPLACED BY ITS LOGICAL AND WITH THE D FIELD. IF THE MACHINE IS IN THE USER MODE, ONLY THOSE BITS OF STATUS WHICH ARE CONTROLLED BY THE USER ARE AFFECTED.
CC = CHANGED AS PART OF STATUS HALFWORD.

SPECIAL PURPOSE:

78.

EXF R,X,D1,D2 (EXTRACT FIELD)

BITS D1, D1+1, ..., D1+D2-1 OF REGISTER X ARE PLACED IN THE LEAST SIGNIFICANT END OF R. THE REMAINING BITS OF R ARE SET TO ZERO.
CC = UNAFFECTED.

79.

DEF R,X,D1,D2 (DEPOSIT FIELD)

THE D2 LEAST SIGNIFICANT BITS OF X ARE PLACED IN BITS D1, D1+1, ..., D1+D2-1 OF R. THE REMAINING BITS OF R ARE UNCHANGED.
CC = UNAFFECTED.

80.

MVW R,X,D (MOVE WORDS)

THE FIELD BEGINNING AT THE RELATIVE ADDRESS IN R IS MOVED INTO THE FIELD BEGINNING AT THE RELATIVE ADDRESS IN R+1. THE WORD COUNT TO BE MOVED IS IN X. DURING EXECUTION, R AND R+1 ARE INCREMENTED AND X IS DECREMENTED, UNTIL R AND R+1 POINT TO THE NEXT WORD AFTER THEIR RESPECTIVE FIELDS AND X CONTAINS ZERO.
CC = UNAFFECTED.

81.

CVB R,M (INTEGER TO BINARY-CODED-DECIMAL)

THE POSITIVE INTEGER IN M IS CONVERTED TO BINARY-CODED-DECIMAL FORM (EIGHT 4-BIT DIGITS) AND PLACED IN R. THIS INSTRUCTION MAY CAUSE AN OVERFLOW CONDITION IF THE NUMBER IN M IS TOO LARGE.
CC = UNAFFECTED.

82.

BCDI R,M (BINARY-CODED-DECIMAL TO INTEGER)

THE CONTENT OF M IS INTERPRETED AS EIGHT 4-BIT BINARY-CODED-DECIMAL DIGITS. THIS NUMBER IS CONVERTED TO A BINARY INTEGER AND PLACED IN R. INVALID BCD CODES ARE TREATED AS ZEROS; HOWEVER, THE OVERFLOW FLIP-FLOP IS SET IF AN INVALID BCD CODE IS FOUND.
CC = UNAFFECTED.

83.

XEQ R,M (EXECUTE)

THE INSTRUCTION IN M IS FETCHED, "INCLUSIVE-ORED" WITH THE CONTENT OF R, AND EXECUTED. IF R = 0, THE INSTRUCTION IS NOT MODIFIED. LOOPS CAUSED BY AN XEQ INSTRUCTION EXECUTING ITSELF ARE INTERRUPTABLE.
CC = SET BY THE EXECUTED INSTRUCTION.

54.

.DDS M (LOAD Q AND S)

REGISTER B IS ADDED TO THE CONTENT OF BITS 0-15 OF M, AND THE RESULT IS LOADED INTO Q. REGISTER B IS ADDED TO THE CONTENT OF BITS 16-31 OF M, AND THE RESULT IS LOADED INTO S.
CC = UNAFFECTED.

55.

[illegible] M (STORE Q AND S)

REGISTER B IS SUBTRACTED FROM REGISTERS Q AND S. Q-B IS STORED IN BITS 0-15 OF M; S-B IS STORED IN BITS 16-31 OF M. REGISTERS B, Q, AND S ARE UNCHANGED.
CC = UNAFFECTED.

56.

[illegible] M (LOAD STATUS)

THE STATUS HALFWORD IS LOADED FROM THE RIGHT HALFWORD OF M. IF THE MACHINE IS IN THE USER MODE, ONLY THOSE BITS WHICH ARE CONTROLLED BY THE USER ARE AFFECTED.

57.

[illegible] M (STORE STATUS)

THE STATUS HALFWORD IS STORED IN THE RIGHT HALFWORD OF M. THE LEFT HALFWORD OF M IS UNAFFECTED.
CC = UNAFFECTED.

AWS R (AVAILABLE STACK)

THE QUANTITY Z-S IS PLACED IN REGISTER R. THIS QUANTITY REPRESENTS THE NUMBER OF WORDS REMAINING IN THE AVAILABLE STACK AREA.
CC = PATTERN A.

LAR R,M (LOAD ADDRESS INTO REGISTER)

THE RELATIVE ADDRESS M IS COMPUTED AND LOADED INTO R. THE LEFT HALFWORD OF R IS SET TO ZERO.
IF A=0, M = (RIGHT HALFWORD OF REGISTER X)
IF A=1, 2, OR 3, M = (A) + D + X - B
CC = UNAFFECTED.

LAS M (LOAD ADDRESS ONTO STACK)

THE STACK POINTER S IS INCREMENTED BY 1. THE RELATIVE ADDRESS M IS COMPUTED AND PLACED IN THE CORE WORD POINTED TO BY S.
IF A=0, M = (RIGHT HALFWORD OF REGISTER X).
IF A=1, 2, OR 3, M = (A) + D + X - B.
CC = UNAFFECTED.

PSH R,X (PUSH)

```
TEMP←R;
WHILE TEMP IS LESS THAN OR EQUAL TO X DO
     BEGIN
          S←S+1;
          (S)←(TEMP);
          TEMP←TEMP+1 MODULO 8;
     END;
```

ALL THE WORDS FROM REGISTER R TO REGISTER X ARE PUSHED ONTO THE STACK.
CC = UNAFFECTED.

2.

P R,X (POP)

```
TEMP←X;
WHILE TEMP IS EQUAL TO OR GREATER THAN R DO
     BEGIN
          (TEMP)←(S);
          S←S-1;
          TEMP←TEMP-1 MODULO 8;
     END;
```

THE STACK IS POPPED AND ITS DATA IS STORED IN REGISTERS FROM X DOWN TO R.
CC = UNAFFECTED.

3.

AL M (PROCEDURE CALL)

```
S←S+1;
(S[16]-S[31])←P+1-B;
(S[1]-S[15])←S-Q;
(S[0])←0;
Q←S;
P←M;
```

A NEW STACK MARKER IS CREATED AND PLACED ON THE STACK, AND A BRANCH TO M IS EXECUTED.
IF A=0, M IS COMPUTED BY ADDING REGISTER B TO THE LOW-ORDER HALF-ORD OF REGISTER X.
CC = UNAFFECTED.

4.

AL D (EXTERNAL CALL)

```
S←S+1;
(S[16]-S[31])←P+1-B;
(S[1]-S[15])←S-Q;
(S[0])←1;
Q←S;
P←(D+(U[16]-U[30]))INDIRECT;
IF D>(U[0]-U[15]) THEN INTERRUPT;
```

U IS A FIXED MEMORY LOCATION IN THE SYSTEM AREA. ITS RIGHT HALF POINTS TO A TABLE OF EXTERNAL PROCEDURES, AND ITS LEFT HALF CONTAINS THE LENGTH OF THE TABLE. D IS USED TO CHOOSE THE TABLE ENTRY. A STACK MARKER IS CREATED, AND D IS CHECKED TO MAKE SURE IT IS NOT GREATER THAN THE LENGTH OF THE TABLE. IF D IS VALID, THE MACHINE BRANCHES THROUGH U AND THROUGH THE TABLE TO THE EXTERNAL PROCEDURE.
CC = UNAFFECTED.

```
55.

SCAL D          (SYSTEM CALL)

     S←S+1;
     (S[15]←S[31])←B;
     (S[0]←S[15])←STATUS;
     S←S+1;
     (S[0])←1;
     (S[1]←S[15])←S-Q;
     (S[16]←S[31])←P+1←B;
     Q←S;
     S←0;
     STATUS←KNOWN STATE (EXECUTIVE MODE);
     P←(D+(V[16]←V[30])) INDIRECT;
     IF D>(V[0]←V[15]) THEN INTERRUPT;
```

A NEW (TWO-WORD) STACK MARKER IS CREATED AND PLACED ON THE STACK. B IS SET TO ZERO, AND THE MACHINE IS PLACED IN EXECUTIVE MODE. V IS A FIXED MEMORY LOCATION IN THE SYSTEM AREA, WHICH POINTS TO A TABLE OF SYSTEM ROUTINES. IF D IS VALID, THE MACHINE BRANCHES THROUGH V AND THROUGH THE TABLE TO THE SYSTEM ROUTINE, AS EXPLAINED IN THE ECL INSTRUCTION. CC = UNAFFECTED.

56.

```
XAX  0          (STACK EXIT)

     IF Q[0]=1 THEN
          BEGIN
               B←((Q-1)[16]←(Q-1)[31]);
               STATUS←((Q-1)[0]←(Q-1[15]));
          END;
     P←(Q[16]←Q[31])+B;
     S←Q-1;
     Q←Q←DELTA Q;
```

A BRANCH IS EXECUTED TO THE RETURN ADDRESS, AND STACK DATA PERTAINING TO THE OLD PROCEDURE OR SYSTEM ROUTINE IS THROWN AWAY. THIS INSTRUCTION IS USED TO RETURN FROM ROUTINES CALLED BY PCL, ECL, SCL, OR INTERRUPTS.
CC = UNAFFECTED.

```
PUSH R,X,D      (PUSH DATA STACK)

    M←D+(P, Q, OR B);
    IF X[16-31] = X[0-15] THEN
         BEGIN
              CC←2;
              EXECUTE (M);
         END;
    X[16-31] ← X[16-31]+1;
    (M+X[15-31])←(R);
    CC←0;
```

THE ADDRESS M=D+(P, Q, OR B) POINTS TO THE BEGINNING OF A DATA STACK. X[0-15] CONTAINS THE NUMBER OF WORDS ALLOCATED FOR THE DATA STACK, AND X[16-31] CONTAINS THE NUMBER OF ELEMENTS CURRENTLY ON THE DATA STACK. THE CONTENT OF REGISTER R IS PUSHED ONTO THE DATA STACK. IF OVERFLOW OCCURS, THE CONDITION CODE IS SET TO 2, AND THE PROCEDURE CALL IN M IS EXECUTED. IF THE PSDS WAS SUCCESSFUL, CC=0.

```
POPS R,X,D      (POP DATA STACK)

    M←D+(P, Q, OR B);
    IF X[16-31]=0 THEN
         BEGIN
              CC←1;
              EXECUTE(M);  (M CONTAINS A PCL INSTRUCTION)
         END;
    R←(M+X[16-31]);
    X[16-31] ← X[16-31]-1;
    CC←0;
```

THE ADDRESS M=D+(P, Q, OR B) POINTS TO THE BEGINNING OF A DATA STACK, AND X[16-31] CONTAINS THE NUMBER OF ELEMENTS CURRENTLY ON THE DATA STACK. THE TOPMOST ELEMENT OF THE DATA STACK IS POPPED INTO REGISTER R. IF UNDERFLOW OCCURS, THE CONDITION CODE IS SET TO 1, AND THE PROCEDURE CALL IN M IS EXECUTED. IF THE PPDS WAS SUCCESSFUL, CC=0.

(LOAD Z AND MPR)

REGISTER B IS ADDED TO THE CONTENT OF BITS 0-15 OF M, AND THE RESULT IS LOADED INTO Z. REGISTER B IS ADDED TO THE CONTENT OF BITS 16-31 OF M, AND THE RESULT IS LOADED INTO MPR.
THIS IS A PRIVILEGED INSTRUCTION.
CC = UNAFFECTED.

(STORE Z AND MPR)

REGISTER B IS SUBTRACTED FROM REGISTERS Z AND MPR. Z-B IS STORED IN BITS 0-15 OF M; MPR-B IS STORED IN BITS 16-31 OF M. REGISTERS B, Z, AND MPR ARE UNCHANGED.
THIS IS A PRIVILEGED INSTRUCTION.
CC = UNAFFECTED.

(INTERROGATE)

THE CONTENT OF THE INTERRUPT REGISTER IS PLACED IN R. REGISTER INR IS CLEARED. THE "EXTERNAL INTERRUPT PENDING" BIT IN THE STATUS HALFWORD IS CLEARED.
THIS IS A PRIVILEGED INSTRUCTION.
CC = UNAFFECTED.

12.

PAUS (PAUSE)

THE MACHINE HALTS UNTIL AN INTERRUPT OCCURS OR THE START BUTTON IS PUSHED.
CC = UNAFFECTED.

13.

LD R,M (START I/O)

THE LEFT HALF OF REGISTER R CONTAINS THE I/O PROCESSOR NUMBER (4 BITS) AND THE DEVICE NUMBER (8 BITS). M IS THE ADDRESS OF AN I/O PROGRAM WHICH IS TO BE SENT TO THE I/O PROCESSOR TO BE EXECUTED. IF THE SIO COMMAND IS NOT SUCCESSFUL, AN I/O STATUS HALFWORD IS RETURNED TO THE RIGHT HALF OF REGISTER R BY THE I/O PROCESSOR. IF A = 0, THE RIGHT HALF OF REGISTER X CONTAINS THE ADDRESS OF THE I/O PROGRAM.
THIS IS A PRIVILEGED INSTRUCTION.
CC = 0 IF SIO WAS SUCCESSFUL.
CC = 1 IF SIO WAS NOT SUCCESSFUL (STATUS IN REGISTER R)

14.

TD R (TEST I/O)

THE LEFT HALF OF REGISTER R CONTAINS THE I/O PROCESSOR NUMBER (4 BITS) AND THE DEVICE NUMBER (8 BITS). A HALFWORD CONTAINING THE STATUS OF THE I/O PROCESSOR AND DEVICE IS RETURNED TO THE RIGHT HALF OF REGISTER R.
THIS IS A PRIVILEGED INSTRUCTION.
CC = 0 IF THE I/O PROCESSOR IS AVAILABLE.
CC = 1 IF THE I/O PROCESSOR IS BUSY.
CC = 2 IF THE I/O PROCESSOR IS NOT OPERATIONAL.

15.

H (HALT I/O)

THE LEFT HALF OF REGISTER R CONTAINS THE I/O PROCESSOR NUMBER (4 BITS) AND THE DEVICE NUMBER (8 BITS). THIS DEVICE IS HALTED, AND ITS STATUS HALFWORD IS RETURNED TO THE RIGHT HALFWORD OF REGISTER R.
THIS IS A PRIVILEGED INSTRUCTION.
CC = UNAFFECTED.

6.

RDD R,M (READ DIRECT)

THE LEFT HALF OF REGISTER R CONTAINS THE I/O PROCESSOR NUMBER (4 BITS) AND THE DEVICE NUMBER (8 BITS). A WORD IS READ DIRECTLY FROM THIS DEVICE INTO M. PROCESSING DOES NOT CONTINUE UNTIL THE READ IS COMPLETED. IF THE READ WAS UNSUCCESSFUL, THE DEVICE STATUS HALFWORD IS RETURNED TO THE RIGHT HALF OF REGISTER R.
THIS IS A PRIVILEGED INSTRUCTION.
CC = 0 IF RDD WAS SUCCESSFUL.
CC = 1 IF RDD WAS NOT SUCCESSFUL (STATUS IN REGISTER R).

WRD R,M (WRITE DIRECT)

THE LEFT HALF OF REGISTER R CONTAINS THE I/O PROCESSOR NUMBER (4 BITS) AND THE DEVICE NUMBER (8 BITS). A WORD IS WRITTEN DIRECTLY FROM M ONTO THE DEVICE. PROCESSING DOES NOT CONTINUE UNTIL THE WRITE IS COMPLETED. IF THE WRITE WAS UNSUCCESSFUL, THE DEVICE STATUS HALFWORD IS RETURNED TO THE RIGHT HALF OF REGISTER R.
THIS IS A PRIVILEGED INSTRUCTION.
CC = 0 IF WRD WAS SUCCESSFUL.
CC = 1 IF WRD WAS NOT SUCCESSFUL (STATUS IN REGISTER R).

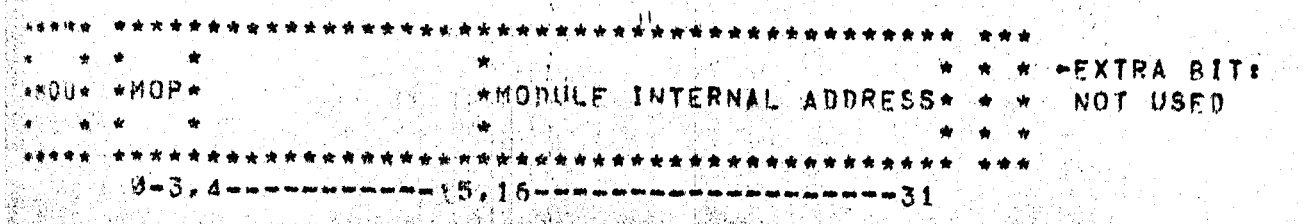
PROGRAMMING:

TO DO I/O, THE PROGRAMMER MUST SET UP A PROGRAM IN CORE CONSISTING OF I/O COMMAND WORDS (IOCW'S) AND I/O DATA WORDS (IODW'S). EACH IOCW HAS BIT 0 = 1, AND CONTAINS AN OPCODE AND CERTAIN MODIFIER BITS. EACH IOCW MAY BE FOLLOWED BY ONE OR MORE IODW'S, WHICH SPECIFY THE AREA(S) IN MEMORY ON WHICH IT IS TO OPERATE. EACH IODW HAS BIT 0 = 0, AND CONTAINS A BYTE ADDRESS (18 BITS) AND A BYTE COUNT (13 BITS). WHEN AN SIO COMMAND IS ISSUED, THE I/O PROCESSOR EXECUTES THE FIRST IOCW SUCCESSIVELY ON EACH OF THE FOLLOWING IODW'S, THEN EXECUTES THE NEXT IOCW ON EACH OF THE IODW'S FOLLOWING IT, ETC., UNTIL A SPECIAL "END" IOCW IS ENCOUNTERED.

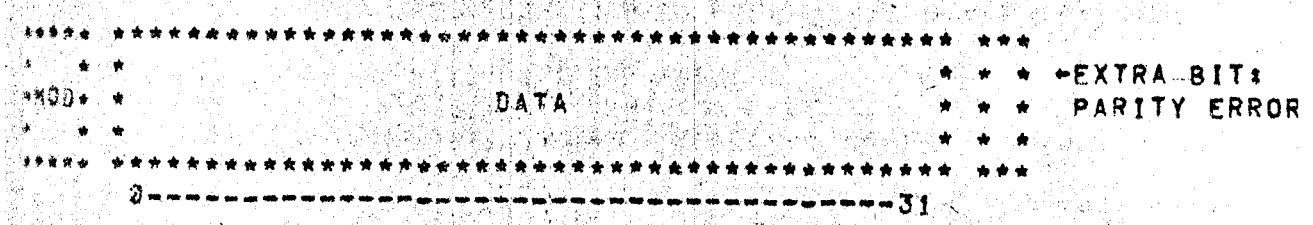
EACH I/O DEVICE HAS A DEVICE STATUS DOUBLEWORD (DSD) IN A DEDICATED POSITION IN LOWER CORE STORAGE. THE FIRST WORD CONTAINS A POINTER TO THE IOCW CURRENTLY BEING EXECUTED BY THE DEVICE, IF ANY, AND OTHER STATUS INFORMATION ABOUT THE DEVICE (BUSY, NOT OPERATIONAL, ETC.) THE SECOND WORD CONTAINS THE CURRENT BYTE ADDRESS AND BYTE COUNT FOR THE DATA TRANSFER BEING MADE BY THE DEVICE. AS THE DEVICE EXECUTES THE TRANSFER SPECIFIED BY AN IODW, THE BYTE ADDRESS IS INCREMENTED AND THE BYTE COUNT IS DECREMENTED UNTIL IT REACHES ZERO; THEN THE DEVICE PROCEEDS TO THE NEXT IODW.

FORMATS:

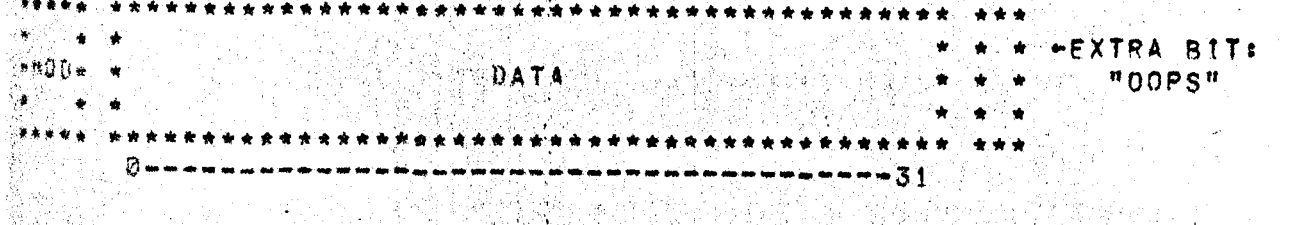
CPU - MEMORY



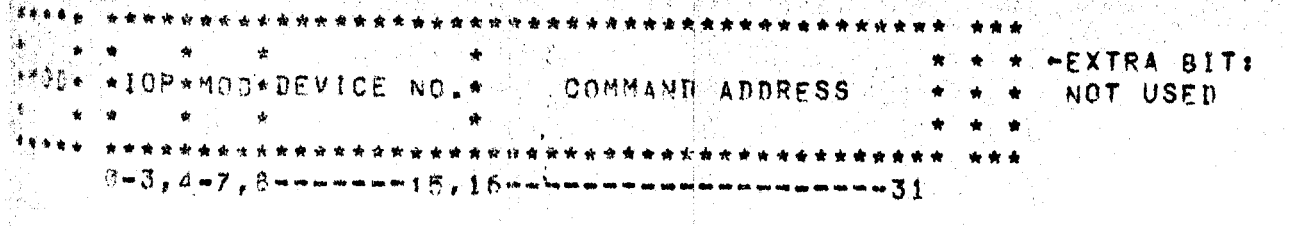
MEMORY - CPU



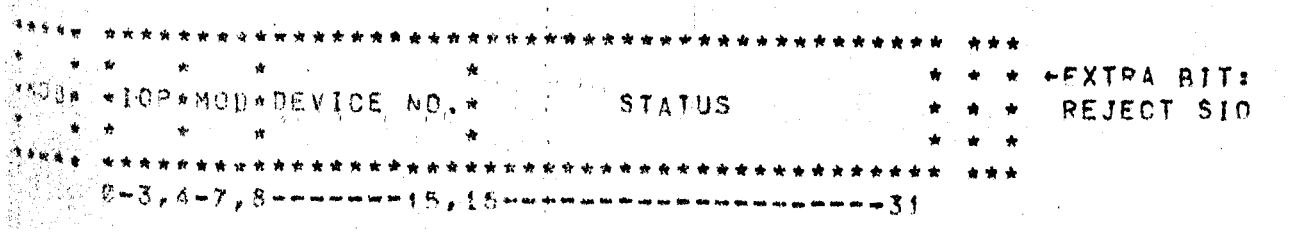
CPU - MEMORY



CPU - I/O



I/O - CPU



ERNAL INTERRUPT PROCESSING

EXTERNAL INTERRUPTS ARE DEFINED AS THOSE INPUTS TO THE CPU FROM OTHER MODULES, (I.E., I/O MULTIPLEXORS, I/O SELECTORS, OPERATOR CONSOLES, ADDITIONAL CPU'S AND UNIQUE MODULES), EXCEPTING THOSE INPUTS THAT THE CPU IS CURRENTLY REQUESTING AND WAITING FOR (I.E., MEMORY DATA, I/O RESPONSES FROM INSTRUCTIONS).

EXTERNAL INTERRUPT INPUTS WILL CONTAIN THE INTERRUPTING MODULE NUMBER AND 28 BITS OF DATA.

THE CPU HAS ONE 32-BIT INTERRUPT REGISTER (INR). WHEN THIS REGISTER IS CLEAR THE CPU CAN ACCEPT ONE EXTERNAL INTERRUPT (ASYNCHRONOUSLY). THE INTERRUPTING MODULE SENDS ITS MODULE NUMBER, MODULE TYPE, AND OTHER PERTINENT DATA TO THE INTERRUPT REGISTER.

ADDITIONAL INTERRUPTS THAT ARRIVE WHEN INR IS SET WILL BE REJECTED, I.E., MODULE BUSS TRANSFER WILL NOT BE ACKNOWLEDGED. NORMALLY THE INTERRUPTING MODULE WILL CONTINUE TO REQUEST ACCESS TO THE CPU UNTIL THIS TRANSFER IS ACKNOWLEDGED.

THE INTERRUPT REGISTER (INR) IS CLEARED VIA THE INTERROGATE INSTRUCTION (INT).

THE EXTERNAL INTERRUPT ROUTINE (EIR) MAY BEGIN AT THE COMPLETION OF A CPU INSTRUCTION (AND AT SELECTED INTERVALS IN LONG INSTRUCTIONS) WHEN THE STATUS MASK BIT FOR EXTERNAL INTERRUPTS IS CLEAR.

THE EXTERNAL INTERRUPT ROUTINE (EIR) IS FUNCTIONALLY SIMILAR TO THE SYSTEM CALL INSTRUCTION (SCL). THE RETURN

ADDRESS, STACK MARKER, BASE REGISTER, AND STATUS ARE PLACED ON THE STACK, THE BASE REGISTER IS SET TO ZERO, AND CONTROL IS TRANSFERRED TO THE ABSOLUTE ADDRESS CONTAINED IN A SPECIFIC LOCATION. THE EIR AND SCL BOTH SET BIT 0 OF THE STACK MARKER WORD TO 1.

INTERRUPT ROUTINE EXITS MAY BE HANDLED IN THE SAME MANNER AS A SYSTEM CALL EXIT UTILIZING THE STACK EXIT INSTRUCTION (SEX).

EIR EXTERNAL INTERRUPT ROUTINE

```
S ← S + 1
(S[16] - S[31]) ← B
(S[0] - S[15]) ← STATUS
STATUS←KNOWN STATE (EXECUTIVE MODE);
S ← S + 1
(S[0]) ← 1
(S[16] - S[31]) ← P-B   ←  ( P IS THE ADDRESS OF THE
(S[1] - S[15]) ← S-Q          INSTRUCTION TO BE RETURNED TO.)
Q ← S
B ← 0
P ← CONTENT OF K (*)
```

*K TO BE DETERMINED WHEN FIXED MEMORY LOCATIONS ARE ASSIGNED.

FOLLOWING EXECUTION OF THE EIR, AT LEAST ONE MACHINE INSTRUCTION WILL BE EXECUTED BEFORE FURTHER INTERRUPTS OF ANY KIND ARE PERMITTED. THE EIR SETS THE STATUS TO INHIBIT FURTHER INTERRUPTS OF THE SAME TYPE, UNTIL AN INSTRUCTION IS EXECUTED TO ENABLE THIS TYPE OF INTERRUPT AGAIN.

INTERNAL INTERRUPT PROCESSING

INTERNAL INTERRUPT PROCESSING IS INITIATED WHEN THE CPU DETECTS THOSE INTERNAL CONDITIONS THAT WOULD INHIBIT NORMAL PROGRAM EXECUTION AND/OR WARRANT IMMEDIATE ACTION BY THE PROCESSOR. THE TYPES OF INTERNAL INTERRUPTS ARE LISTED BELOW. INTERNAL INTERRUPT PROCESSING WILL NORMALLY BEGIN IMMEDIATELY UPON DETECTION.

THE INTERNAL INTERRUPT ROUTINE (IIR) IS SIMILAR TO THE EXTERNAL INTERRUPT ROUTINE. CONTROL IS TRANSFERRED TO PREDETERMINED MEMORY LOCATIONS FOR EACH TYPE OF INTERNAL INTERRUPT LISTED BELOW. INTERNAL INTERRUPTS DO NOT LEAVE DATA IN A REGISTER; HOWEVER, PERTINENT DATA CAN BE RECOVERED BECAUSE THE PROGRAM ADDRESS AT WHICH THE INTERRUPT OCCURED IS PLACED ON THE STACK.

INTERNAL INTERRUPT TYPE		MAY BE MASKED OFF?
1	POWER FAIL	NO
2	MEMORY PARITY	NO
3	UNIMPLEMENTED INSTRUCTION	NO
4	PRIVILEGED INSTRUCTION CALLED IN USER MODE	NO
5	STACK OVERFLOW	NO
6	STACK UNDERFLOW	NO
7	MEMORY PROTECT VIOLATION	NO
8	ARITHMETIC OVERFLOW	YES
9	CLOCK	YES

IIR INTERNAL INTERRUPT ROUTINE

```
S ← S + 1
(S[16] - S[31]) ← B
(S[0] - S[15]) ← STATUS
S ← S + 1
(S[0]) ← 1
(S[16] - S[31]) ← P-B   ← (P IS ADDRESS OF INSTRUCTION
                            IN PROCESS AT TIME OF INTERRUPT)
(S[1] - S[15]) ← B-Q
Q ← S
B ← 0
P ← L + INTERRUPT TYPE # (*)
```

* L TO BE DETERMINED WHEN FIXED MEMORY LOCATIONS ARE ASSIGNED.

CPU STATUS HALFWORD

BIT

```
 0        CONDITION CODE
 1        CONDITION CODE
 2        CONDITION CODE
 3        MODE  USER/PRIVILEGED
 4        OVERFLOW
 5        CONDITION CODE FROZEN
 6        EXTERNAL INTERRUPT PENDING
 7
 8
 9
10        MASK MEMORY OUT OF BOUNDS  READ
11        MASK          INT. ARITH. OVF
12        MASK          FLT. PT. ARITH. OVF
13        MASK          FLT. PT. ARITH. UNDERFLOW
14        MASK          CLOCK INTERRUPT
15        MASK          EXTERNAL INTERRUPT
```

HP OMEGA

MEMORY MODULE SPECIFICATIONS

GENERAL DESCRIPTION

EACH MEMORY MODULE ON THE HP OMEGA COMPUTER IS A SELF CONTAINED UNIT (EXCEPT FOR POWER SUPPLIES) CONSISTING OF 8192 WORDS OF MAGNETIC CORE STORAGE PLUS PARITY AND THE NECESSARY CONTROL LOGIC. EACH UNIT MAY COMMUNICATE WITH ANY CPU OR I/O MODULE VIA THE DATA BUSS, SUBJECT TO ACCESS SIGNALS FROM THE MODULE CONTROL UNIT.

STRUCTURE

FIGURE 1 SHOWS THE BLOCK DIAGRAM OF A MEMORY MODULE. THIS STRUCTURE IS GENERAL IN NATURE AND IS INDEPENDENT OF THE CYCLE TIME OF THE MAGNETIC CORE STACK.

MIO: MEMORY INPUT OUTPUT CONTROL

CONTAINS ALL LOGIC FUNCTIONS NECESSARY FOR TALKING TO THE MODULE CONTROL UNIT AND FOR USING THE DATA BUSS. DECODES MEMORY OP CODES AND HANDLES INTERRUPT CONDITIONS.

MTG: MEMORY TIMING GENERATOR

GENERATES ALL TIMING SIGNALS NECESSARY TO DRIVE A MAGNETIC CORE STACK. THESE SIGNALS HAVE A FIXED RELATIONSHIP TO THE MIO'S CLOCK, BUT ARE AT A FREQUENCY DETERMINED BY THE PARTICULAR TYPE OF CORE STACK BEING DRIVEN.

MAR: MEMORY ADDRESS REGISTER

HOLDS THE MEMORY ADDRESS BEING ACCESSED. ALL ADDRESSES (13 BITS) LIE BETWEEN 0 AND 8191.

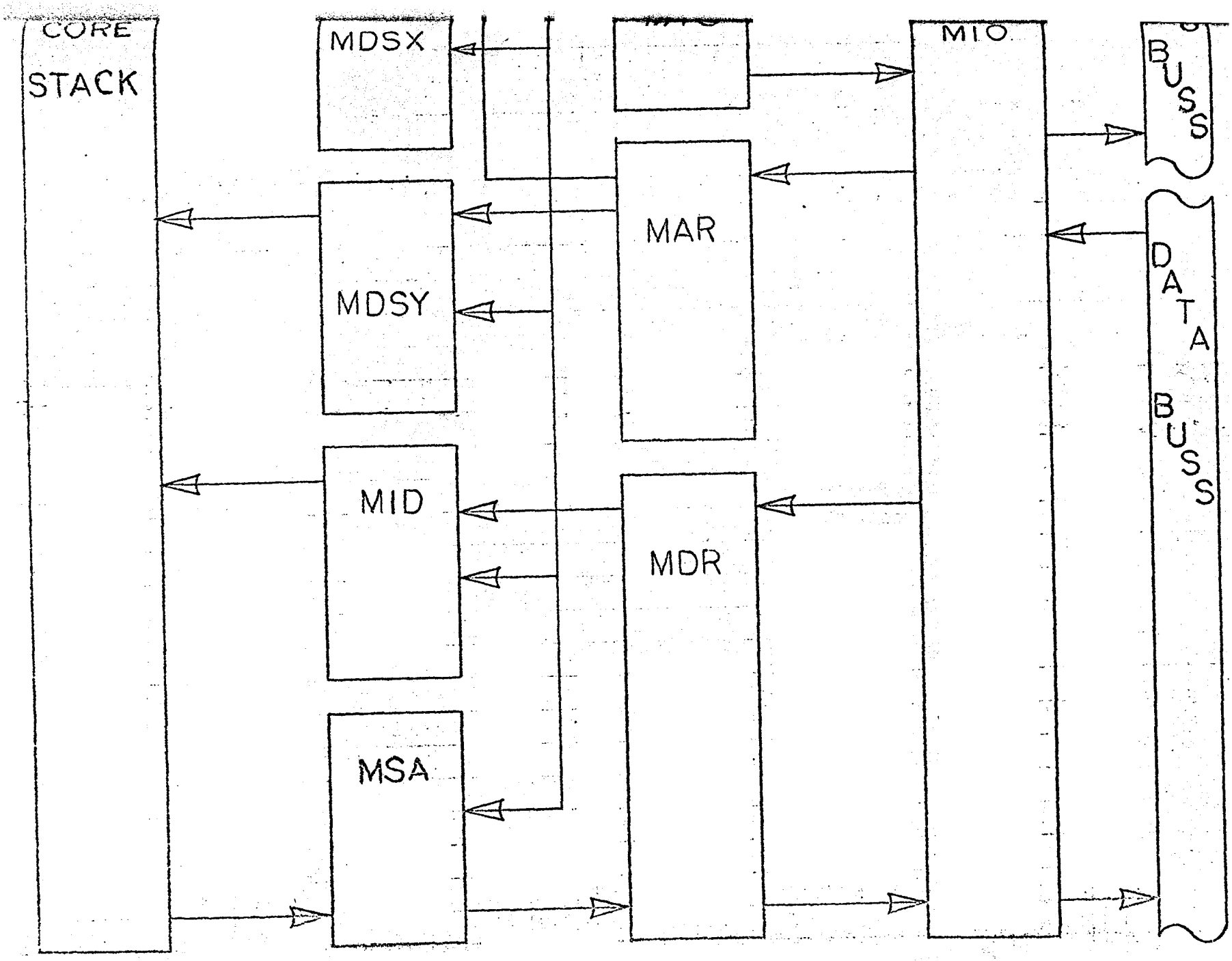


FIGURE 1. MEMORY MODULE DIAGRAM

MDR: MEMORY DATA REGISTER

HOLDS THE WORD (32 BITS + PARITY) READ OUT OF CORE AND TO BE PUT ON THE DATA BUSS, OR THE WORD RECEIVED ON THE DATA BUSS TO BE STORED IN CORE. ODD PARITY IS USED.

MSA: MEMORY SENSE AMPLIFIERS

DETECT THE OUTPUTS OF THE MAGNETIC CORES AND ENTER THE INFORMATION INTO THE MDR VIA THE DIRECT SET INPUTS.

MID: MEMORY INHIBIT DRIVERS

INHIBIT THE WRITING OF ONE'S INTO THE CORE MEMORY FOR BITS IN THE MDR THAT ARE ZEROS.

MDSX: MEMORY DRIVER-SWITCHES X

PROVIDE HALF THE READ AND WRITE WORD CURRENTS TO THE CORE STACK.

MDSY: MEMORY DRIVERS-SWITCHES Y

PROVIDE HALF THE READ AND WRITE WORD CURRENTS TO THE CORE STACK. THE X AND Y DRIVERS TOGETHER SELECT 1 OF 8192 WORDS (32 BITS + PARITY) IN MEMORY.

OPERATION

OP CODES

THE MEMORY MODULE RESPONDS TO 3 MAJOR TYPES OF COMMANDS:

READ-WRITE (RW)

THE CONTENTS OF THE ADDRESSED LOCATION IN MEMORY ARE READ, TRANSMITTED TO THE REQUESTING MODULE VIA THE DATA BUSS, AND WRITTEN BACK INTO MEMORY. THIS COMMAND WOULD TYPICALLY BE USED FOR INSTRUCTION

FETCHES OR LOAD REGISTER INSTRUCTIONS.

CLEAR-WRITE (CW)

THE CONTENTS OF THE ADDRESSED LOCATION IN MEMORY ARE READ AND CLEARED. A DATA WORD IS RECEIVED ON THE BUSS AND WRITTEN INTO MEMORY. THIS COMMAND WOULD TYPICALLY BE USED FOR STORE REGISTER INSTRUCTIONS.

READ-MODIFY-WRITE (RMW)

THE CONTENTS OF THE ADDRESSED LOCATION IN MEMORY ARE READ, TRANSMITTED TO THE REQUESTING MODULE. NEW DATA IS RECEIVED FROM THE SAME MODULE AND IS WRITTEN INTO MEMORY. THIS COMMAND WOULD TYPICALLY BE USED FOR INCREMENT MEMORY INSTRUCTIONS.

THE RW AND CW COMMANDS MAY OPERATE ON EITHER 32 BIT WORDS OR 1 OF 4 8-BIT BYTES. THE BYTE COMMANDS ARE READ BYTE-WRITE (RBWN) AND CLEAR BYTE-WRITE(CBWN), WHERE N IS THE BYTE NUMBER (SEE FIGURE 2). CBWN AFFECTS ONLY THE SPECIFIED BYTE, THE OTHER 3 ARE LEFT UNCHANGED. A BYTE OF DATA IS ALWAYS TRANSMITTED ON THE LOW ORDER POSITION ON THE DATA BUSS (BITS 24-31).

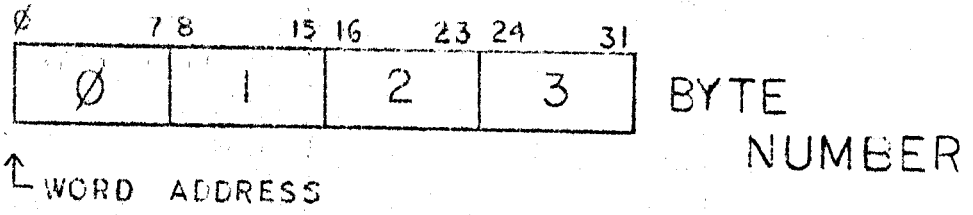


FIGURE 2.

DATA TRANSMISSION

MCU BUSS

THESE 19 LINES CONCERN THE MEMORY MODULE'S ABILITY TO ACCESS THE DATA BUSS AS DETERMINED BY THE MODULE CONTROL UNIT.

CLK: THIS IS THE MASTER CLOCK FOR THE SYSTEM. ALL COMMUNICATION WITH THE MCU AND ALL DATA TRANSMISSION ON THE DATA BUSS ARE IN SYNCHRONISM WITH THIS CLOCK.

RYM: READY MODULE

THIS LINE BEING TRUE INDICATES THAT THE MEMORY MODULE IS READY TO ACCEPT EITHER COMMANDS OR DATA FROM THE DATA BUSS.

RQBN: REQUEST BUSS TO MODULE N

THESE 16 LINES ARE MUTUALLY EXCLUSIVE AND THE ONE THAT IS TRUE IS A REQUEST TO THE MCU FOR THE MEMORY MODULE TO USE THE DATA BUSS TO SEND DATA TO MODULE N. NOTE THAT ONLY 15 OF THE LINES ARE USED, AS THE 16TH WOULD BE THE MEMORY MODULE'S OWN ADDRESS.

SLM: SELECT MODULE

THIS LINE ORIGINATES IN THE MCU AND IS USED BY THE BY THE MEMORY MODULE TO GATE DATA ONTO THE DATA BUSS. IT IS THE MCU'S RESPONSIBILITY TO SEE THAT ONLY ONE MODULE IS SELECTED IN ANY GIVEN CLOCK PERIOD.

DATA BUSS

THESE 42 LINES CONNECT TO ALL MODULES ON A SYSTEM AND SERVE TO TRANSMIT DATA AMONG THEM.

TO0-3: TO MODULE N

THESE 4 LINES CARRY THE BINARY ADDRESS (N) OF THE MODULE THE DATA IS BEING SENT TO.

FR0-3: FROM MODULE N

THESE 4 LINES CARRY THE BINARY ADDRESS (N) OF THE MODULE SENDING THE DATA.

DT0-31: DATA (32 BITS)

THESE LINES CARRY THE COMMANDS AND DATA. BIT 0 IS THE MOST SIGNIFICANT BIT (MSB). ALL BYTE TRANSMISSIONS TO AND FROM THE MEMORY MODULE USE ONLY BITS 24 TO 31. ALL OTHER BITS ARE IGNORED ON RECEIVED DATA, AND ARE SET TO 0 ON TRANSMISSION.

DT32: DISASTEROUS TROUBLE

THIS IS AN INTERRUPT LINE FOR BOTH INPUT AND OUTPUT.

INPUT:

SINCE A DATA FETCH MAY ALREADY BE IN PROGRESS BEFORE THE MEMORY PROTECT BOUNDS ARE CHECKED BY THE CPU, THIS LINE IS USED TO INHIBIT MODIFICATION OF THE ADDRESSED LOCATION. IT WOULD BE TRUE WHEN A MEMORY PROTECT VIOLATION IS DETECTED BY THE CPU. IT IS TRANSMITTED DURING THE TIME THE DATA WOULD ORDINARILY BE BE ON THE BUSS TO THE MEMORY MODULE.

OUTPUT:

THIS LINE IS HELD TRUE WHEN THE MEMORY MODULE HAS ACCESS TO THE BUSS IF THE DATA JUST READ BY A RW, RBW, OR RMW COMMAND HAD INCORRECT PARITY. IN THIS CASE THE ADDRESS OF THE WORD IS PLACED ON THE DATA BUSS, INSTEAD OF THE ERRONEOUS DATA. THE WORD IS REWRITTEN INTO MEMORY WITH CORRECT PARITY.

EMORY OPERATION CODES

0	0000	NOP	NO OPERATION
1	0001	RMW	READ MODIFY WRITE
2	0010	RBW0	READ BYTE WRITE 0
3	0011	CBW0	CLEAR BYTE WRITE 0
4	0100		
5	0101	CW	CLEAR WRITE
6	0110	RBW1	READ BYTE WRITE 1
7	0111	CBW1	CLEAR BYTE WRITE 1
8	1000		
9	1001		
10	1010	RBW2	READ BYTE WRITE 2
11	1011	CBW2	CLEAR BYTE WRITE 2
12	1100	RW	READ WRITE
13	1101		
14	1110	RBW3	READ BYTE WRITE 3
15	1111	CBW3	CLEAR BYTE WRITE 3

HP OMEGA

MODULE CONTROL UNIT SPECIFICATIONS

GENERAL DESCRIPTION

THE MODULE CONTROL UNIT (MCU) GRANTS PERMISSION TO ONE AND ONLY ONE OF THE 16 MODULES AT ANY ONE TIME TO USE THE DATA BUSS. THE MCU TAKES INTO CONSIDERATION THE PRIORITY OF THE REQUESTING MODULE. IF THE RECEIVING MODULE IS READY TO RECEIVE DATA, AND IF A REQUESTING MODULE HAS BEEN WAITING FOR THE BUSS. NO MODULE WILL BE DENIED PERMISSION TO USE THE BUSS FOR ANY UNREASONABLE LENGTH OF TIME.

FUNCTION

A. TO MONITOR ALL MODULE REQUESTS FOR DATA TRANSMISSION TO ANOTHER MODULE VIA THE DATA BUSS.

B. TO MONITOR THE STATUS OF ALL MODULES AND DETERIMINE IF A RECEIVING MODULE IS READY TO ACCEPT DATA.

C. TO SELECT ONE AND ONLY ONE MODULE REQUESTING THE BUSS AND ENABLE THAT MODULE'S SELECT LINE.

D. TO GENERATE THE BASIC CLOCK (CLK) AND OUTPUT THIS CLOCK TO EACH OF THE MODULES IN THE SYSTEM.

MODULE TO MCU CONTROLS LINES

A. TRANSMISSION REQUEST LINES - UNIDIRECTIONAL LINES FROM EACH MODULE IN THE SYSTEM TO THE MCU. THEY CONTAIN THE FOLLOWING INFORMATION:

1. WHICH MODULE IS REQUESTING.
2. WHICH UNIQUE MODULE IS TO RECEIVE THE DATA.

EACH MODULE HAS ONE REQUEST LINE FOR EACH MODULE IN THE SYSTEM EXCLUDING ITSELF, THEREFORE A MAXIMUM OF 15 REQUEST LINES PER MODULE FOR A 16 MODULE SYSTEM.

B. MODULE READY LINES - UNIDIRECTIONAL LINES, ONE FROM EACH MODULE TO THE MCU INDICATING THAT IT IS READY TO RECEIVE DATA TRANSMISSIONS. THIS DOES NOT IMPLY THAT AN ACKNOWLEDGEMENT WILL BE GIVEN FOR ANY TRANSMISSION.

C. MODULE SELECT LINES - UNIDIRECTIONAL LINES, ONE TO EACH MODULE FROM THE MCU TO BE USED AS AN "ENABLE" LINE TO GATE INFORMATION WAITING TO BE TRANSMITTED ONTO THE BUSS, THEREBY GRANTING PERMISSION TO TRANSMIT TO A REQUESTING MODULE.

D. CLOCK - A UNIDIRECTIONAL LINE FROM THE MCU TO EACH MODULE.

LIMITATIONS:

A. ALL LINES COMMUNICATING WITH THE MCU MUST BE RAISED AND LOWERED WITH THE CLOCK.

B. THE MINIMUM TIME TO SELECT WILL BE ON THE CLOCK CYCLE FOLLOWING THE REQUEST.

C. A REQUEST LINE MUST ONLY BE DROPPED ON THE CLOCK CYCLE AFTER BEING SELECTED. THIS DOES NOT IMPLY THAT THE REQUEST LINE CANNOT REMAIN HIGH IF MULTIPLE TRANSMISSIONS ARE DESIRED (SEE SECTION H). IN ALL CASES, IF A REQUEST LINE IS LEFT HIGH ONE CLOCK CYCLE AFTER SELECT, THE MCU INTERPRETS THAT AS A NEW REQUEST FOR THE BUSS. IF ACKNOWLEDGEMENT OF TRANSMITTED DATA IS NOT RECEIVED ON THE CYCLE AFTER SELECT, IT IS THE RESPONSIBILITY OF THE SENDING MODULE TO REQUEST THE BUSS AGAIN AND WAIT TO BE SELECTED TO TRANSMIT THE DATA.

D. A REQUEST LINE MAY BE RAISED ON ANY CLOCK CYCLE.

E. ONLY ONE REQUEST LINE MAY BE RAISED AT A TIME FROM ANY ONE MODULE.

F. THE BUSS MUST BE ENABLED BY THE SELECT LINE, THEREFORE THE DELAY WITHIN THE SELECTED MODULE MUST BE KEPT TO A MINIMUM TO ALLOW MAXIMUM BUSS TRANSFER TIME.

G. THE RECEIVING MODULE'S ACKNOWLEDGEMENT OF TRANSMISSION AND ACCEPTANCE OF DATA MUST BE PLACED ON THE BUSS WITH THE CLOCK DURING THE CYCLE IMMEDIATELY FOLLOWING THE DATA TRANSMISSION.

H. MULTIPLE TRANSMISSIONS - WHEN IT IS DESIRABLE TO MAKE MULTIPLE TRANSMISSIONS ALONG THE BUSS TO THE SAME MODULE, THE REQUEST LINE MAY STAY UP AT ALL TIMES, UNTIL THE LAST PIECE OF DATA HAS BEEN TRANSMITTED. THE REQUEST LINE MUST BE DROPPED IMMEDIATELY UPON THE CYCLE FOLLOWING THE LAST SELECT. IF IT IS NOT ACKNOWLEDGED, THE REQUEST LINE MUST THEN BE RAISED AND LOWERED AGAIN IN THE NORMAL FASHION.

WHEN IT IS DESIRABLE TO MAKE MULTIPLE TRANSMISSIONS ALONG THE BUSS TO SEPARATE MODULES, THE REQUEST LINE MUST BE DROPPED WITH THE SELECT LINE AND THE CLOCK. IF NO ACKNOWLEDGEMENT IS RECEIVED, THE SAME REQUEST LINE MAY BE RAISED ON THE NEXT CLOCK CYCLE.

I. MODULE READY LINES SHOULD BE ENABLED ONE CYCLE BEFORE THEY ARE READY TO RECEIVE DATA SINCE THE EARLIEST THE MODULE MAY RECEIVE DATA IS ONE CYCLE AFTER READY IS ENABLED.

CIRCUIT OPERATION

LINES ARE GATED ONTO THE BUSS IN THE FOLLOWING FASHION:

A. NO MODULE MAY TRANSMIT ON THE BUSS DURING TWO SUCCESSIVE CYCLE.

B. A PRIORITY FLAG IS SET IF EVER 2 OR MORE MODULES REQUEST THE BUSS AT THE SAME TIME. IN THIS STATE, ALL REQUESTING MODULES ARE IN A CLOSED GROUP WHICH IS COMPLETELY SERVICED AND THEN THE PRIORITY FLAG IS CLEARED.

C. MODULES IN THE CLOSED GROUP DESCRIBED ABOVE ARE SERVICED BY ORDER OF A PRIORITY STRING DETERMINED BY PHYSICAL LOCATION OF THE MODULE IN THE SYSTEM.

D. ALL REQUESTS MADE SINCE THE PRIORITY FLAG WAS SET AND ANY PRESENT REQUESTS ARE THEN CONSIDERED AS ANOTHER CLOSED GROUP OF REQUESTS, SERVICED IN THE SAME FASHION. THE PRIORITY FLAG IS THEN SET UNTIL THIS SECOND SET IS SERVICED.

E. IF NO REQUESTS WERE MADE DURING THE SERVICING OF THE FIRST GROUP OF REQUESTS, THE PRIORITY FLAG REMAINS CLEARED UNTIL TWO OR MORE MODULES REQUEST SIMULTANEOUSLY.

F. A MODULE WILL NOT BE SELECTED TO TRANSMIT IF THE RECEIVING MODULE IS NOT READY.

HP OMEGA

I/O MODULE SPECIFICATIONS

TABLE OF CONTENTS

1. GENERAL DESCRIPTION

I/O MODULES IN THE HP OMEGA SYSTEM PROVIDE MEANS FOR THE TRANSMISSIONS OF DATA, CONTROL ORDERS, AND STATUS INFORMATION AMONG THE CPU, MEMORY MODULES AND PERIPHERAL DEVICES. EACH I/O MODULE IS CAPABLE OF ACCEPTING AND INTERPRETING I/O INSTRUCTIONS ISSUED BY THE CPU, EXECUTING STORED I/O PROGRAMS WHICH CONSIST OF I/O COMMANDS, AND PROCESSING OR TRANSFERRING TO THE CPU INTERRUPTION REQUESTS FROM PERIPHERAL DEVICES.

AN I/O MODULE CONSISTS OF AN I/O PROCESSOR WITH A SET OF DEVICE CONTROLLERS ATTACHED TO IT THROUGH AN I/O INTERFACE. EACH DEVICE CONTROLLER IS IN TURN LINKED TO A SET OF PERIPHERAL DEVICES OF THE SAME TYPE. THE RELATIONSHIP BETWEEN AN I/O MODULE AND OTHER ELEMENTS OF THE HP OMEGA SYSTEM IS SHOWN IN FIG. 1.

2. TYPES OF I/O PROCESSORS

THERE ARE TWO TYPES OF I/O PROCESSORS: MULTIPLEXOR AND SELECTOR. THERE MAY BE UP TO 256 DEVICE CONTROLLERS ATTACHED TO EACH I/O PROCESSOR. AN I/O MULTIPLEXOR SERVICES PERIPHERAL DEVICES IN THE MULTIPLEX MODE, I.E., IT CAN SUSTAIN CONCURRENTLY ONE I/O OPERATION PER DEVICE CONTROLLER SO LONG AS IT CAN TOLERATE THE TOTAL LOAD IMPOSED BY THE SIMULTANEOUS I/O OPERATIONS. AN I/O SELECTOR OPERATES IN THE BURST MODE, I.E., ONLY ONE DATA TRANSFER MAY BE IN PROGRESS THROUGH THE SELECTOR AT ANY TIME. THE FACILITIES OF A MULTIPLEXOR ARE TIME-SHARED AMONG DEVICE CONTROLLERS. THE FACILITIES OF A SELECTOR IS MONOPOLIZED BY A SINGLE DEVICE CONTROLLER AT A TIME. AN I/O MULTIPLEXOR MAY APPEAR AVAILABLE TO THE CPU FOR ACCEPTING

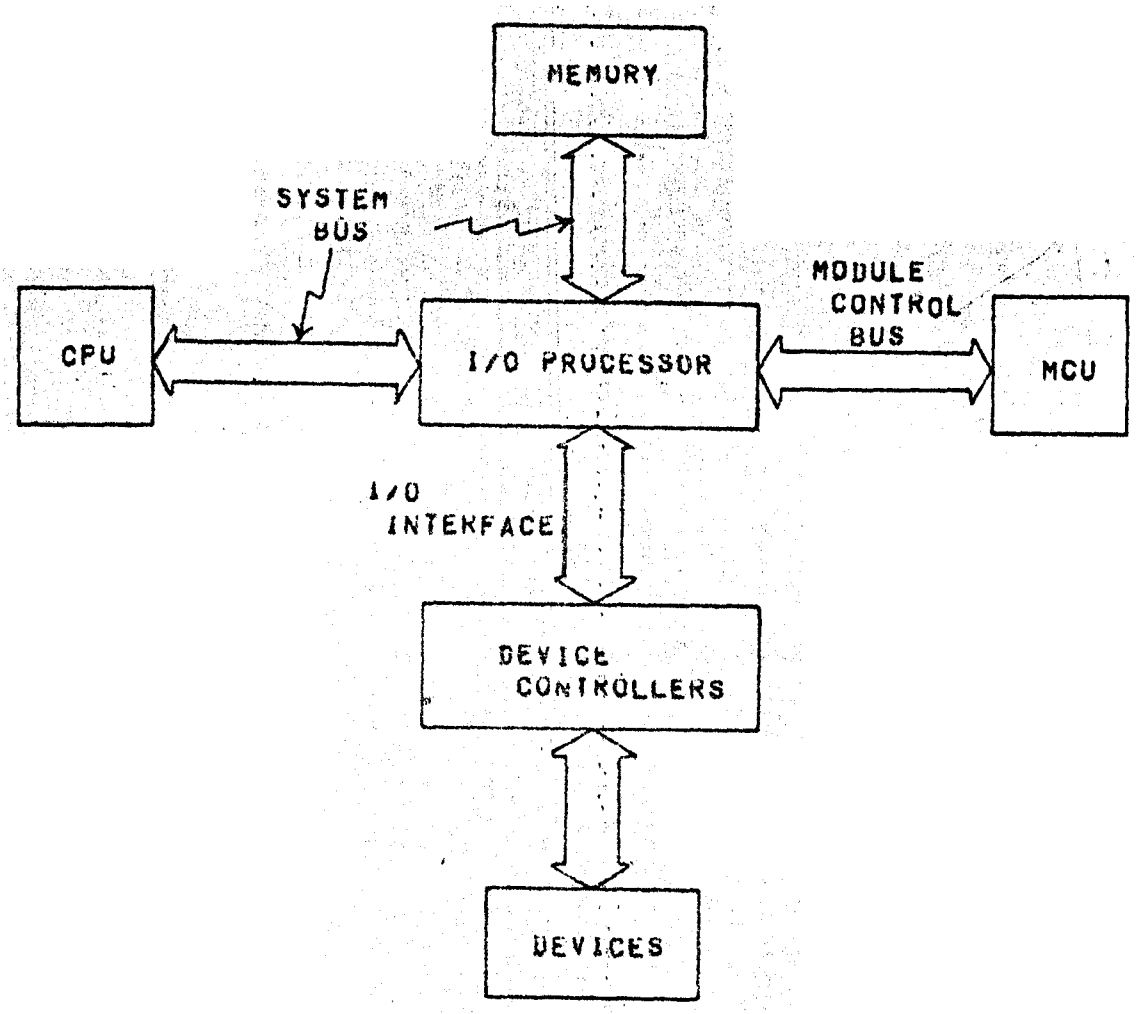


FIG. 1 AN I/O MODULE IN THE HP OMEGA SYSTEM

NEW I/O INSTRUCTIONS WHILE PREVIOUSLY INITIATED I/O OPERATIONS ARE STILL IN PROGRESS, WHEREAS AN I/O SELECTOR APPEARS BUSY TO NEW I/O INSTRUCTIONS.

3. I/O SYSTEM OPERATION

ALL I/O OPERATIONS ARE INITIATED BY I/O INSTRUCTIONS ISSUED TO I/O PROCESSORS FROM THE CPU. AN I/O PROCESSOR IS RESPONSIBLE, WHEN AVAILABLE, FOR ACCEPTING AND INTERPRETING AN I/O INSTRUCTION. PROCESSING OF AN I/O INSTRUCTION MAY ENABLE THE CPU TO TRANSFER DATA DIRECTLY TO OR FROM A SPECIFIC PERIPHERAL DEVICE, MAY CAUSE AN I/O OPERATION TO HALT, MAY PRESENT I/O STATUS INFORMATION TO THE CPU, OR MAY INITIATE THE EXECUTION OF A SEQUENCE OF I/O COMMANDS AS PREVIOUSLY SET UP IN CORE. TO ACCOMPLISH THE TASKS REQUIRED BY I/O INSTRUCTIONS, AN I/O PROCESSOR ISSUES I/O COMMANDS TO ITS DEVICE CONTROLLERS, EACH DEVICE CONTROLLER IN TURN INTERPRETS THE COMMANDS AND ISSUES CONTROL ORDERS TO ITS DEVICES.

4. I/O INSTRUCTIONS

THERE ARE FIVE I/O INSTRUCTIONS AS DESCRIBED IN THE CPU INSTRUCTION SET:

SIO (START I/O)

INITIATES THE EXECUTION OF AN I/O PROGRAM.

TIO (TEST I/O)

FETCHES THE STATUS INFORMATION OF AN I/O DEVICE.

HIO (HALT I/O)

TERMINATES OPERATIONS AT AN I/O DEVICE.

RDD (READ DIRECT)

PROVIDES A DIRECT INPUT DATA PATH FOR AN I/O DEVICE.

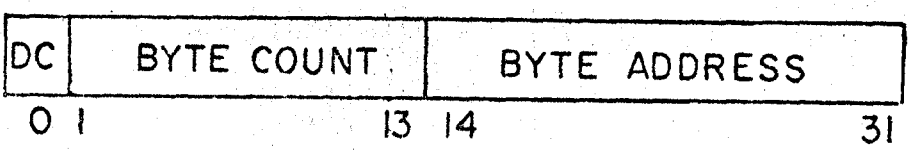
WRD (WRITE DIRECT)

PROVIDES A DIRECT OUTPUT DATA PATH FOR AN I/O DEVICE.

5. I/O PROGRAM

AN I/O PROGRAM SHOULD BE SET UP IN CORE PRIOR TO THE ISSUING OF START IO INSTRUCTION TO AN I/O PROCESSOR. THE CONTENT OF THE I/O PROGRAM MAY NOT BE ALTERED BY THE I/O PROCESSOR. TWO TYPES OF INFORMATION MAY APPEAR IN AN I/O PROGRAM. THEY ARE I/O COMMAND WORDS (CW) AND DATA BLOCK PARAMETER WORDS (BPW). A BPW MAY ONLY FOLLOW A READ OR WRITE COMMAND, OR IT MAY FOLLOW ANOTHER BPW IF SO SPECIFIED BY THE LEFTMOST BIT OF THE PRECEDING BPW. THEREFORE, THE FIRST WORD IN AN I/O PROGRAM IS ALWAYS A COMMAND WORD. A READ OF WRITE COMMAND IS ALWAYS FOLLOWED BY ONE OR MORE BLOCK PARAMETER WORDS. THE LEFTMOST BIT OF A BPW IS USED TO SPECICY WHETHER THE NEXT PROGRAM WORD, IF ANY, IS A CW OR ANOTHER BPW.

A BLOCK PARAMETER WORD HAS THE FOLLOWING FORMAT:



A BLOCK PARAMETER SPECIFIES THE LOCATION AND SIZE OF A BLOCK OF DATA TO BE TRANSFERRED. BIT 0 IS THE DATA-CHAINING (DC) BIT WHICH IS USED TO SPECIFY WHETHER OR NOT ANOTHER BPW IS IMMEDIATELY FOLLOWING. BITS 1-13 ARE USED TO SPECIFY THE

NUMBER OF BYTES IN THE DATA BLOCK TO BE TRANSFERRED. BITS 14-31 ARE USED TO SPECIFY THE STARTING ADDRESS FROM OR TO WHICH THE FIRST BYTE OF DATA IN THE BLOCK IS TO BE TRANSFERRED.

THERE ARE FIVE COMMANDS WHICH MAY APPEAR IN AN I/O PROGRAM:

READ

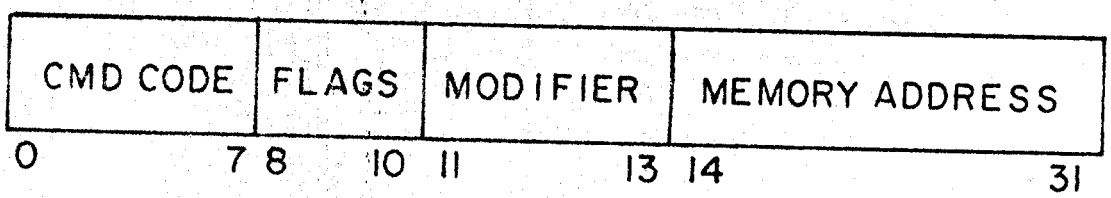
WRITE

SENSE

CONTROL

JUMP

A COMMAND WORD (CW) HAS THE FOLLOWING FORMAT:



THE FIELDS ARE DEDICATED TO THE FOLLOWING PURPOSES:

COMMAND CODE: BITS 0-7 SPECIFY THE OPERATION TO BE PERFORMED.

FLAGS: BITS 8-10 ARE USED TO SPECIFY COMMAND CHAINING, INTERRUPT ON COMPLETION, AND HALT ON TRANSMISSION ERROR. IF THE COMMAND CHAINING (CC) BIT IS ONE, THE PRESENT COMMAND WORD TOGETHER WITH ITS BLOCK PARAMETERS IF ANY, IS FOLLOWED BY ANOTHER COMMAND WORD IN THE I/O PROGRAM. THUS, ONLY THE LAST COMMAND WORD IN AN I/O PROGRAM HAS THE COMMAND CHAINING BIT ZERO. IF THE INTERRUPT ON COMPLETION BIT IS ONE, AN INTERRUPTION WILL BE REQUESTED UPON THE COMPLETION OF THE COMMAND EXECUTION. IF THE HALT ON TRANSMISSION ERROR BIT IS ONE, THE DATA TRANSFER OPERATION AND THE DEVICE WILL BE HALTED AS SOON AS A

TRANSMISSION ERROR IS DETECTED; OTHERWISE, DATA TRANSFER WILL CONTINUE AND THE I/O DEVICE MAY REQUEST TO PRESENT THE ERROR STATUS AFTER COMPLETION OF THE CURRENT DATA TRANSFER.

MODIFIER: BITS 11-13, IF USED, FURNISHES MORE INFORMATION ABOUT THE OPERATION TO BE PERFORMED.

MEMORY ADDRESS: BITS 14-31 SPECIFY A MEMORY LOCATION. THIS FIELD ISSUED ONLY FOR SOME OF THE COMMANDS: JUMP, SENSE, AND CONTROL.

ANY FIELD WHICH IS NOT USED FOR A PARTICULAR COMMAND IS IGNORED WHEN THE COMMAND IS EXECUTED.

THE EFFECTS OF THE I/O COMMANDS ARE EXPLAINED IN THE FOLLOWING:

(1) READ

ONE OR MORE BLOCKS OF DATA AS SPECIFIED IN SUBSEQUENT BPW'S ARE TO BE TRANSFERRED FROM THE I/O DEVICE TO MEMORY CORE. DATA WITHIN EACH BLOCK IS TO BE STORED IN AN ASCENDING ORDER OF ADDRESSES, STARTING WITH THE ADDRESS SPECIFIED IN THE BPW.

(2) WRITE

ONE OR MORE BLOCKS OF DATA AS SPECIFIED IN SUBSEQUENT BPW'S ARE TO BE TRANSFERRED TO THE I/O DEVICE FROM MEMORY CORE. DATA WITHIN EACH BLOCK IS TO BE FETCHED FROM MEMORY IN AN ASCENDING ORDER OF ADDRESSES, STARTING WITH THE ADDRESS SPECIFIED IN THE BPW.

(3) SENSE

ONE OR MORE BYTES OF DATA CONCERNING BOTH UNUSUAL CONDITIONS DETECTED IN THE LAST OPERATION AND

THE STATUS OF THE DEVICE ARE TRANSFERRED TO MEMORY CORE. THE DATA IS STORED IN MEMORY IN AN ASCENDING ORDER OF ADDRESSES, STARTING WITH THE ADDRESS SPECIFIED IN THE ADDRESS FIELD OF THE COMMAND WORD. THE NUMBER AND THE MEANING OF BYTES TO BE TRANSFERRED ARE PECULIAR TO THE TYPE OF THE I/O DEVICE.

(4) CONTROL

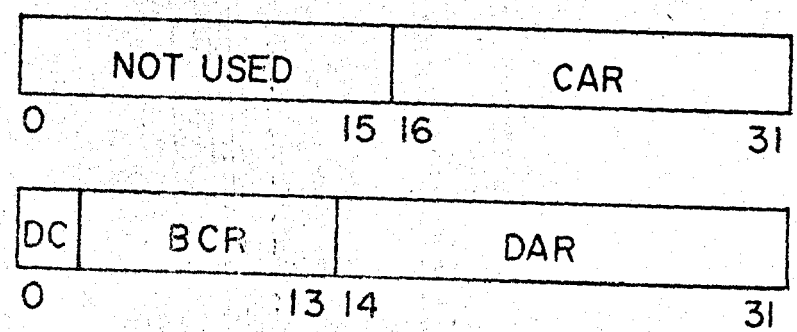
AN OPERATION NOT INVOLVING DATA TRANSFER IS INITIATED AT THE I/O DEVICE. EITHER THE LEFT HALF OR THE WHOLE OF THE CW MAY BE SUFFICIENT TO SPECIFY THE OPERATIONS TO BE PERFORMED. IF MORE CONTROL INFORMATION IS NEEDED, THE ADDRESS FIELD IN THE CW MAY BE USED TO SPECIFY THE LOCATION CONTAINING THE ADDITIONAL INFORMATION. THE CONTROL INFORMATION IN CORE, IF ANY, IS TRANSFERRED TO THE DEVICE FROM CORE IN AN ASCENDING ORDER OF ADDRESSES, STARTING WITH THE ADDRESS SPECIFIED IN THE CW.

(5) JUMP

THE JUMP COMMAND DOES NOT INITIATE ANY I/O OPERATION OVER THE I/O INTERFACE, AND THE I/O DEVICE IS NOT AWARE OF THE PRESENCE OF THIS COMMAND. THE EFFECT OF THIS COMMAND IS FOR THE I/O PROCESSOR TO FETCH THE NEXT COMMAND WORD FROM THE LOCATION SPECIFIED IN THE ADDRESS FIELD OF THE JUMP COMMAND WORD.

6. DEVICE REFERENCE TABLE

SINCE DATA IS TO BE TRANSFERRED IN BLOCKS AND THE I/O PROGRAM IS NOT TO BE ALTERED BY THE I/O PROCESSOR, THE I/O SYSTEM MUST BE PROVIDED WITH SOME SORT OF WORKING REGISTERS, SOFT OR HARD, TO KEEP TRACK OF THE PROGRESS OF DATA TRANSFER. A SET OF CONTIGUOUS LOCATIONS IN THE CORE MEMORY IS ASSIGNED TO BE USED AS THE DEVICE REFERENCE TABLE (DRT). FOR THE I/O MULTIPLEXOR, TWO CONTIGUOUS LOCATIONS ARE ALLOCATED FOR EACH DEVICE CONTROLLER. THE CONTENTS OF EACH ENTRY IN THE DRT HAVE THE FOLLOWING FORMAT:



WHERE:

CAR: COMMAND ADDRESS REGISTER HOLDS THE ADDRESS OF THE NEXT COMMAND TO BE EXECUTED.

DC: DATA CHAINING BIT SPECIFIES DATA CHAINING.

BCR: BYTE COUNT REGISTER HOLDS THE NUMBER OF BYTES LEFT IN THE CURRENT BLOCK TO BE TRANSFERRED.

DAR: DATA ADDRESS REGISTER SPECIFIES THE ADDRESS TO OR FROM WHICH THE NEXT BYTE IS TO BE TRANSFERRED.

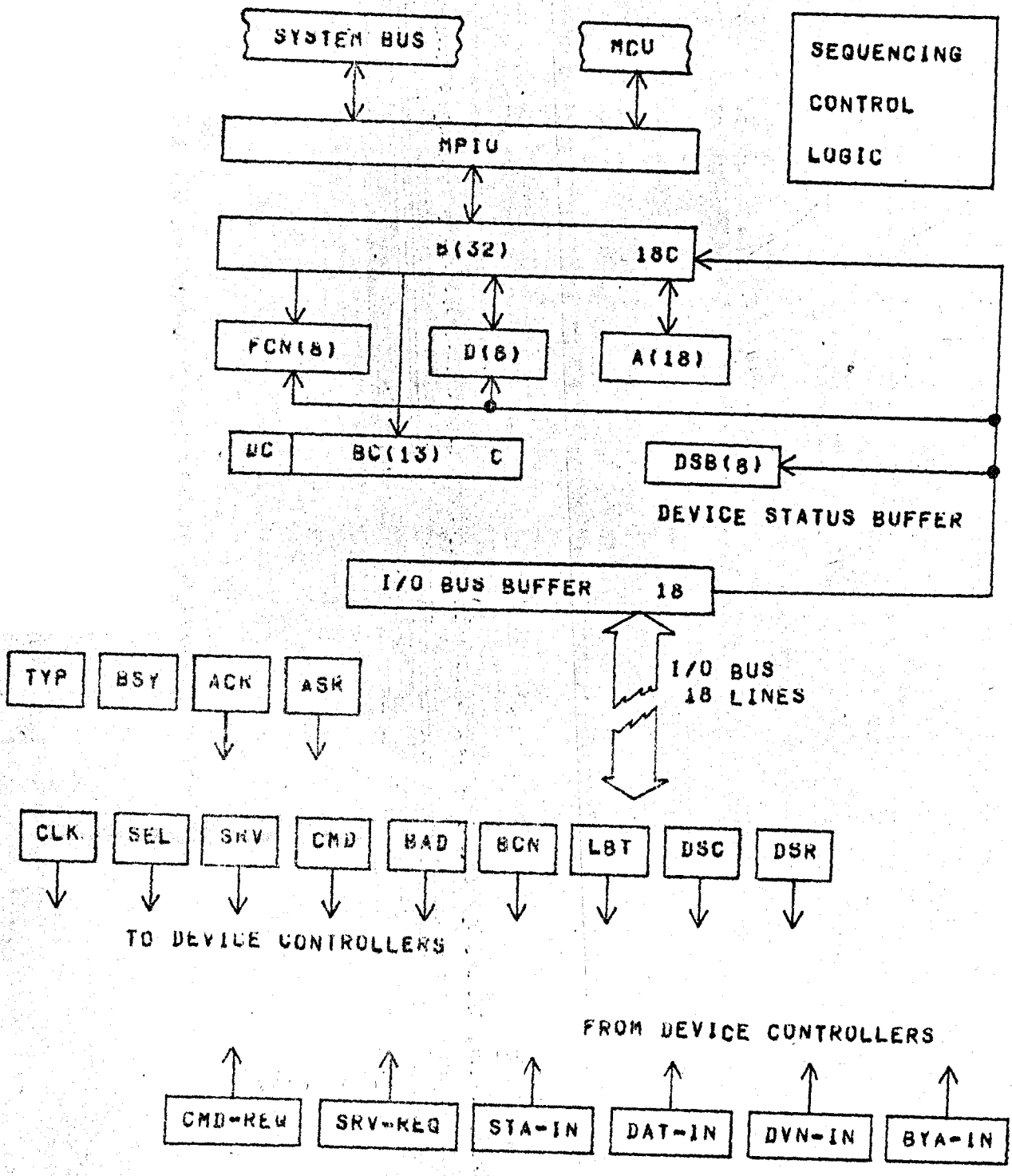
IT IS OPTIONAL FOR A DEVICE CONTROLLER TO HAVE AN ACTIVE REGISTER HOLDING THE DATA-CHAINING BIT, BYTE COUNT, AND DATA ADDRESS. FOR CONVENIENCE OF REFERENCE, A DEVICE CONTROLLER WITH THE OPTIONAL REGISTER IS REFFERED TO AS OF TYPE A; DEVICE

CONTROLLERS WITHOUT THIS OPTION ARE REFERRED TO AS OF TYPE B. FOR A TYPE A DEVICE CONTROLLER, THE SECOND WORD ASSIGNED TO IT IN THE DEVICE REFERENCE TABLE IS NOT USED.

DEVICE CONTROLLERS ATTACHED TO AN I/O SELECTOR ARE ALL OF TYPE B. THE DEVICE REFERENCE TABLE FOR EACH I/O SELECTOR HAS ONLY ONE ALLOCATED LOCATION FOR EACH ATTACHED DEVICE CONTROLLER. ONLY COMMAND ADDRESSES ARE ENTERED INTO THIS TABLE.

7. BASIC STRUCTURE OF THE I/O MULTIPLEXOR (IOMP)

FIG. 7.1 BLOCK DIAGRAM OF THE I/O MULTIPLEXOR.



NOTES ABOUT FIG. 7.1.

MPIO: LOGIC CONTROLLING THE COMMUNICATION BETWEEN THE IOMP AND OTHER MODULES IN THE HP OMEGA SYSTEM.

B: SYSTEM BUFFER REGISTER, B(14-31) IS CAPABLE OF COUNTING.

A: ADDRESS REGISTER.

D: DEVICE NUMBER REGISTER.

FCN: FUNCTION REGISTER CONTROLS I/O SEQUENCING.

DC: DATA CHAINING FLIP-FLOP.

BC: BYTE COUNTER (13 BITS)

DSB: DEVICE STATUS BUFFER PROVIDES TEMPORARY STORAGE FOR STATUS PRESENTED FROM THE DEVICE.

TYP: FLIP-FLOP HOLDS THE TYPE OF THE DEVICE CONTROLLER CURRENTLY COMMUNICATING WITH THE IOMP.

BSY: BUSY FLIP-FLOP.

ACR: ACKNOWLEDGE COMMAND REQUEST FLIP-FLOP.

ASR: ACKNOWLEDGE SERVICE REQUEST FLIP-FLOP.

CLK: CLOCK.

SEL: SELECT-OUT.

SRV: SERVICE-OUT.

CMD: COMMAND-OUT

BAD: BYTE-ADDRESS-OUT.

BCN: BYTE-COUNT-OUT (INCLUDING DATA-CHAINING BIT).

LBT: LAST BYTE.

DSC: DISCONNECT

DSR: DISCONNECT AND RAISE COMMAND REQUEST.

CMD-REQ: COMMAND REQUEST.

SRV-REQ: SERVICE REQUEST.

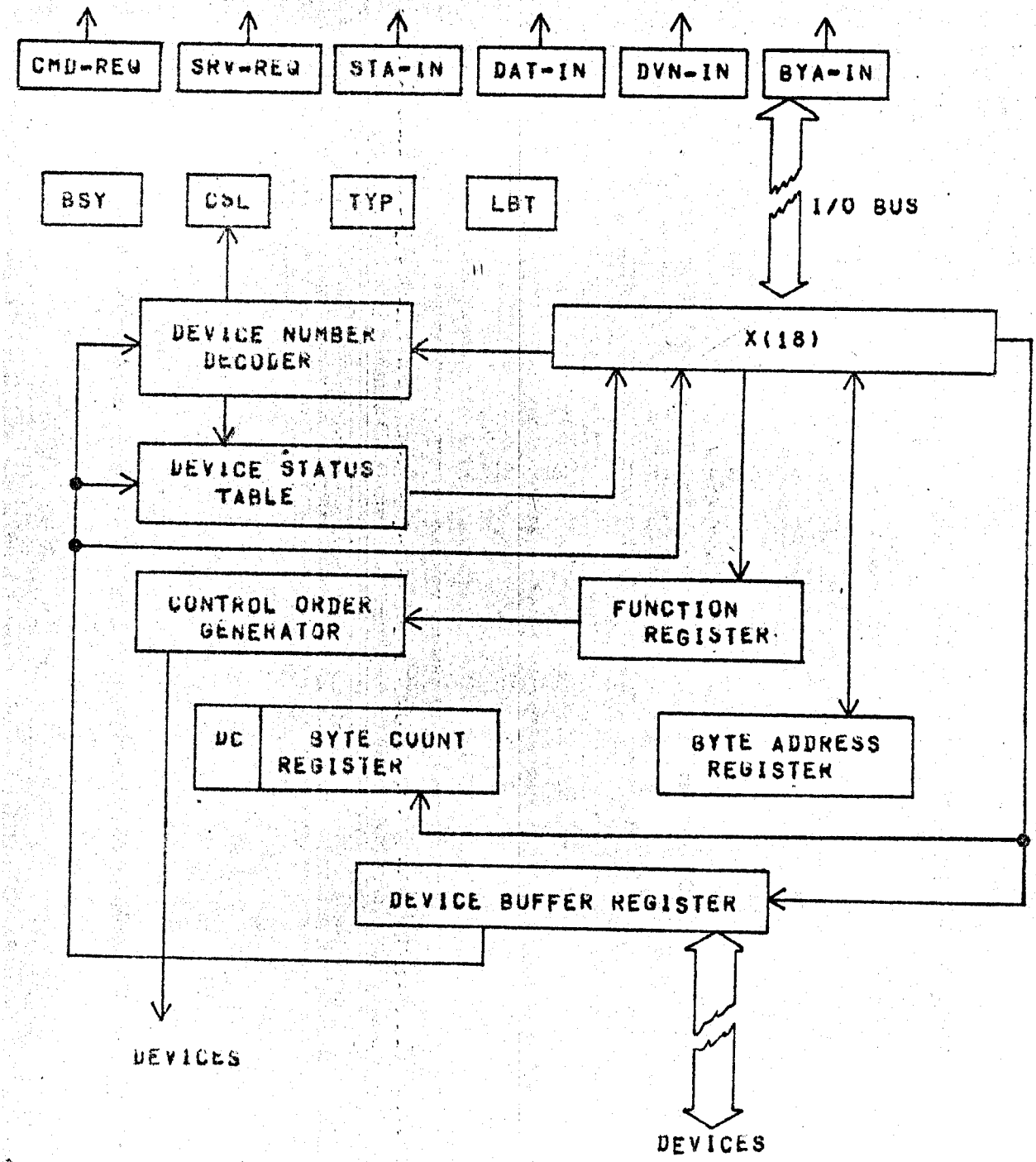
STA-IN: STATUS-IN

DAT-IN: DATA-IN.

DVN-IN: DAVICE-NUMBER-IN.

BYA-IN: BYTE-ADDRESS-IN.

IG. 7.2 TYPICAL STRUCTURE OF A TYPE A DEVICE CONTROLLER.



OTES:

X: 16-BIT I/O INTERFACE BUFFER REGISTER.

CSL: CONTROLLER-SELECT FLIP-FLOP. THE CONTROLLER IS LOGICALLY CONNECTED TO THE I/O INTERFACE IF CSL IS ON.

DST: DEVICE STATUS TABLE CONTAINS ONE ENTRY FOR EACH ATTACHED DEVICE TO KEEP ITS STATUS BITS.

COG: CONTROL ORDER GENERATOR TRANSFORMS I/O COMMANDS INTO SEQUENCES OF OPERATION ORDERS TO ATTACHED I/O DEVICES. THE LOGIC OF COG IS DEPENDENT ON THE TYPE OF I/O DEVICES BEING ATTACHED TO THE DEVICE CONTROLLER.

CONNECTION AND DISCONNECTION OF DEVICE CONTROLLERS TO THE I/O INTERFACE BUS

CONNECTION MAY BE INITIATED BY THE IOMP AND ACCEPTED BY THE SPECIFIED DEVICE CONTROLLER, OR MAY BE REQUESTED BY THE DEVICE CONTROLLER AND THEN GRANTED BY THE IOMP.

THE IOMP ATTEMPTS TO INITIATE A CONNECTION UPON RECEIVING AN I/O INSTRUCTION FROM THE CPU. THIS IS DONE BY PRESENTING THE SPECIFIED DEVICE NUMBER ONTO THE I/O BUS AND RAISING THE SEL-LINE. EACH ATTACHED DEVICE CONTROLLER SEES THE RAISE OF SEL-LINE AND ATTEMPTS TO DECODE THE DEVICE NUMBER PRESENTED ON THE I/O BUS. NORMALLY, ONE AND ONLY ONE DEVICE CONTROLLER DECODES THE DEVICE NUMBER AS ONE OF ITS ATTACHED DEVICE AND THUS TURNS ON ITS CSL (CONTROLLER-SELECTED) FLIP-FLOP. THE "ON" OF CSL WILL AUTOMATICALLY CONNECT THE DEVICE CONTROLLER TO THE I/O INTERFACE BUS LOGICALLY. AFTER RAISING THE SEL-LINE, THE IOMP IS EXPECTING A STATUS RESPONSE FROM THE SELECTED DEVICE CONTROLLER. THE IOMP ASSUMES THAT THE SPECIFIED DEVICE CONTROLLER IS EITHER PHYSICALLY NON-EXISTANT OR NOT OPERATIONAL IF NO STATUS RESPONSE IS RECEIVED WITHIN A CERTAIN AMOUNT OF TIME.

TWO OF THE IN-BOUND LINES IN THE I/O INTERFACE ARE COMMAND-REQUEST AND SERVICE-REQUEST. EACH DEVICE CONTROLLER HAS A CMD-REQ FLIP-FLOP AND A SRV-REQ FLIP-FLOP. THE CMD-REQ LINE OR THE SRV-REQ LINE IS RAISED IF A CMD-REQ FLIP-FLOP OR A SRV-REQ FLIP-FLOP OF ANY ATTACHED DEVICE CONTROLLER IS SET, RESPECTIVELY. TO BE HONORED BY THE IOMP, A SERVICE REQUEST HAS PRECEDENCE OVER A COMMAND REQUEST. THE IOMP MAY HONOR A

REQUEST BY SETTING THE ACR (ACKNOWLEDGE COMMAND REQUEST) FLIP-FLOP OR THE ASR (ACKNOWLEDGE SERVICE REQUEST) FLIP-FLOP. ASR IS SET IF THE IOMP IS AVAILABLE AND THE IN-BOUND LINE SRV-REQ IS UP. ACR IS SET IF THE IOMP IS AVAILABLE AND THE IN-BOUND LINE SRV-REQ IS DOWN AND CMD-REQ IS UP. THE IOMP BECOMES UNAVAILABLE IF EITHER ACR OR ASR IS SET. THE "UP" SIGNALS OF ACR OR ASR WILL BE PROPAGATED THROUGH DEVICE CONTROLLERS AND STOPPED AT THE FIRST DEVICE CONTROLLER WHICH HAS THE APPROPRIATE REQUEST FLIP-FLOP (CMD-REQ OR SRV-REQ) ON. THE DEVICE CONTROLLER WHICH STOPS THE PROPAGATION OF THE ACKNOWLEDGE REQUEST SIGNAL CLAIMS ITSELF AS BEING CHOSEN BY TURNING ON ITS CSL FLIP-FLOP. THE PROPAGATION OF THE ACKNOWLEDGEMENT SIGNALS COULD BE SPEEDED UP BY GROUPING DEVICE CONTROLLERS AND USING CERTAIN LOOK-AHEAD CIRCUITRY.

A DEVICE CONTROLLER IS LOGICALLY DISCONNECTED FROM THE I/O INTERFACE IF ITS CSL FLIP-FLOP IS CLEARED. THE RESET OF CSL IS INITIATED BY SIGNALS DSC OR DSR COMING FROM THE IOMP.

SEQUENCES OF I/O OPERATIONS

ALL I/O OPERATIONS ARE ACCOMPLISHED IN THREE SEQUENCES: INSTRUCTION SEQUENCE, COMMAND REQUEST SEQUENCE, AND SERVICE REQUEST SEQUENCE. THE FIRST ONE IS INITIATED BY THE I/O PROCESSOR; THE LAST TWO ARE INITIATED BY DEVICE CONTROLLERS. NORMALLY, EACH SEQUENCE STARTS WITH THE ESTABLISHMENT OF COMMUNICATION BETWEEN THE I/O PROCESSOR AND ONE OF ITS ATTACHED DEVICE CONTROLLERS AND ENDS WITH THE DISCONNECTION OF THE DEVICE CONTROLLER FROM THE I/O INTERFACE.

9.1. INSTRUCTION SEQUENCE

THE INSTRUCTION SEQUENCE BEGINS WHEN THE IOMP RECEIVES AN INSTRUCTION FROM THE CPU. THE IOMP GATES THE SPECIFIED DEVICE NUMBER OUT TO THE I/O BUS TO SELECT ONE OF THE DEVICE CONTROLLERS. THE SELECTED CONTROLLER WILL RESPOND BY SENDING ITS STATUS INFORMATION OVER THE I/O BUS WITH THE STA-IN TAG UP. EVENTS FOLLOWING THE SELECTION OF A DEVICE CONTROLLER DIFFER FOR DIFFERENT INSTRUCTIONS.

(1) START IO

IF THE DEVICE IS NOT AVAILABLE, THE IOMP SENDS THE I/O STATUS AND INSTRUCTION-REJECT INFORMATION TO THE CPU. BOTH THE CPU AND THE DEVICE CONTROLLER, IF ANY, ARE THEN DISCONNECTED FROM THE IOMP.

IF THE DEVICE IS AVAILABLE, THE IOMP NOTIFIES THE CPU OF THE ACCEPTANCE OF SIO AND RELEASES THE CPU. THE STARTING COMMAND ADDRESS IS STORED IN THE DEVICE REFERENCE TABLE IN CORE, AND THE DEVICE IS SIGNALED TO DISCONNECT AND THEN SET THE CMD-REQ FLIP-FLOP ATTEMPTING TO INITIATE A COMMAND-REQUEST-SEQUENCE.

(2) TEST IO

IN ANY CASE, THE I/O STATUS IS SENT TO THE CPU, AND BOTH THE CPU AND THE DEVICE CONTROLLER ARE RELEASED.

(3) HALT IO

IF THE SPECIFIED I/O DEVICE IS NOT RECOGNIZED, THE STATUS IS SENT TO THE CPU. IF THE SPECIFIED DEVICE IS RECOGNIZED, A COMMAND IS ISSUED TO THE DEVICE CONTROLLER TO TERMINATE ANY OPERATION BEING

CONDUCTED BY THE SPECIFIED DEVICE. BOTH THE CPU AND THE DEVICE CONTROLLER ARE THEN RELEASED.

(4) READ DIRECT AND WRITE DIRECT

IF THE DEVICE IS NOT AVAILABLE, THE CPU IS NOTIFIED OF INSTRUCTION-REJECT WITH THE STATUS INFORMATION. IF THE DEVICE IS AVAILABLE, THE COMMAND CODE OF READ DIRECT OR WRITE DIRECT IS SENT TO THE DEVICE CONTROLLER, AND DATA IS TRANSFERRED TO OR FROM THE DEVICE WHEN IT IS READY. THE CPU IS RELEASED AFTER THE REQUIRED DATA IS TRANSFERRED TO OR FROM THE CPU. IF THE DATA IS TO BE TRANSFERRED TO OR FROM A MEMORY LOCATION AS SPECIFIED IN THE INSTRUCTION, IT IS THE RESPONSIBILITY OF THE CPU TO FETCH FROM OR STORE INTO THE MEMORY THE DATA.

COMMAND-REQUEST SEQUENCE

THIS SEQUENCE IS TO INITIATE A NEW I/O OPERATION AT THE I/O DEVICE. ESSENTIALLY, A COMMAND CODE TOGETHER WITH SOME NECESSARY INFORMATION IS SENT TO THE DEVICE CONTROLLER, AND THE DEVICE CONTROLLER SHOULD KNOW WHAT TO DO AFTER BEING DISCONNECTED.

THE SEQUENCE IS INITIATED IF THE CMD-REQ LINE IS UP, SRV-REQ LINE IS DOWN, AND THE IOMP IS AVAILABLE. THE DEVICE CONTROLLER, AFTER BEING SELECTED, SENDS THE DEVICE NUMBER AND IDENTIFIES ITSELF AS OF TYPE A OR B OVER THE I/O INTERFACE. THE IOMP FETCHES COMMAND ADDRESS FROM THE DEVICE REFERENCE TABLE, FETCHES THE COMMAND WORD FROM CORE AND LOADS IT INTO THE B REGISTER, UPDATES THE COMMAND ADDRESS IN THE DRT.

IF THE COMMAND WORD IN B IS "JUMP", THE NEW COMMAND ADDRESS SPECIFIED IN THE JUMP COMMAND IS STORED IN THE DRT, AND THE DEVICE CONTROLLER IS SIGNALED TO DISCONNECT AND ATTEMPT ANOTHER COMMAND-REQUEST. IF THE COMMAND WORD IN B IS OTHER THAN JUMP, THE IOMP PUTS THE LEFT 16 BITS OF THE B REGISTER ON THE I/O BUS AND RAISES CMD-OUT. AFTER THE COMMAND CODE IS RECEIVED BY THE DEVICE CONTROLLER, DIFFERENT EVENTS MAY FOLLOW DEPENDING ON THE TYPE OF THE COMMAND AND THE TYPE OF THE DEVICE CONTROLLER. THE DEVICE CONTROLLER WILL BE DISCONNECTED AFTER RECEIVING ALL THE NECESSARY INFORMATION FOR EXECUTING THE COMMAND.

.3. SERVICE REQUEST SEQUENCE

THIS SEQUENCE IS REQUESTED BY THE DEVICE CONTROLLER WHEN IT IS READY TO PRESENT OR RECEIVE A SET OF DATA, STATUS, OR CONTROL INFORMATION. AFTER BEING SELECTED, THE DEVICE CONTROLLER SENDS OVER THE I/O BUS TO THE IOMP THE DEVICE NUMBER AND THE TYPE OF SERVICE FOR WHICH IT IS REQUESTING. THE IOMP THEN MODIFIES ITS FCN REGISTER DEPENDING UPON THE TYPE OF SERVICE REQUESTED. DIFFERENT EVENTS WILL FOLLOW FOR DIFFERENT TYPES OF SERVICE REQUESTS.

THE SERVICE TYPES INCLUDE:

(1) TO PRESENT DATA IN A READ COMMAND FOR A TYPE A DEVICE CONTROLLER WITHOUT DEMANDING DATA-CHAINING. *

(2) TO PRESENT DATA IN A READ COMMAND FOR A TYPE A DEVICE CONTROLLER DEMANDING DATA-CHAINING. *

(3) TO REQUEST DATA IN A WRITE COMMAND FOR A TYPE A DEVICE CONTROLLER NOT DEMANDING DATA-CHAINING.

(4) TO REQUEST DATA IN A WRITE COMMAND FOR A TYPE A DEVICE CONTROLLER DEMANDING DATA-CHAINING.

(5) TO PRESENT DATA IN A READ COMMAND FOR A TYPE B DEVICE CONTROLLER.

(6) TO REQUEST DATA IN A WRITE COMMAND FOR A TYPE B DEVICE CONTROLLER.

(7) TO PRESENT DATA IN A SENSE COMMAND.

(8) TO REQUEST DATA IN A CONTROL COMMAND.

(9) TO PRESENT STATUS INFORMATION.

* DEMANDING DATA-CHAINING MEANS DATA-CHAINING IS REQUIRED IMMEDIATELY FOLLOWING THE TRANSFER OF CURRENT BYTE.

I/O STATUS HALFWORD

BIT	STATUS
00	
01	DEVICE AVAILABLE
02	DEVICE BUSY
03	DEVICE INTERRUPTION PENDING
04	DEVICE NOT RECOGNIZED
05	DEVICE CONTROLLER BUSY
06	DEVICE CONTROLLER INTERRUPTION PENDING
07	DEVICE CONTROLLER NOT RECOGNIZED
10	
11	INSTRUCTION REJECT
12	COMMAND REJECT
13	INTERVENTION REQUIRED
14	OVERRUN
15	TRANSMISSION ERROR
16	PROCESSOR DONE
17	FILE PROTECT