MICROASSEMBLER

Blocks 61 to 66 are the Microassembler source texts.
Blocks 67 to 68 are simple examples according to Chilmondeley.
Block 69 is the load block of three AMD Handbook examples.
Block 75 is the load block of the SUPER-16 computer discussed by Mick and Brick.
Block 120 is the load block of a Disk controller by Paul Chu, et. al, p. 6-65 in the Bipolar Microprocessor Logic and Interface, AMD 1983 Data Book.

VARIABLES

WIDTH	Length of microcode in bytes. It must be even.
MEMORY	Record number of microcode in the program file.
ADDRESS	Microcode address in the microprogram ROM.
ORIGIN	Starting block of the microcode program file.
MASK	A array storing the default microcode.
MAX#	Return the maximum number of microcode which can be fitted into one disk block.
INIT	Initialize WIDTH with the stack number. Clear MEMORY ADDRESS, and MASK. Set ORIGIN to Block 180.

CODE MAINTAINENCE

CODE-BUFFER	Return the virtual memory address of the next microcode to be added to the microprogram file.
DEFAULT	Copy the default microcode from MASK to the next microcode recode.
$	Terminate the current microcode assembly by updating MEMORY, ADDRESS, record number, and move default microcode into the next microcode.
ORG	Update ADDRESS with the stack number. Assemble the next microcode into this address.

```
   60 LIST

( MICROASSEMBLER LOAD BLOACK, CHT, 5-JUN-83)
EMPTY
61 LOAD ( VARIABLES)
62 LOAD ( ADDRESS)
63 LOAD ( NUMBERS)
64 LOAD ( BIT FILED)
65 LOAD ( FIELD)
66 LOAD ( MDUMP)

75 CONSTANT SUPER-16
120 CONSTANT DISK
```

```
   61 LIST

(   VARIABLES, CHT, 5-JUN-83)
VARIABLE WIDTH    ( bytes in the microcode, must be even. )
VARIABLE MEMORY   ( microcode sequence number )
VARIABLE ADDRESS  ( microcode address in ROM )
VARIABLE ORIGIN   ( microcode starting block )
VARIABLE MASK 68 ALLOT    ( default microcode )

: MAX#   ( --- n , maximum number of microcodes in one block)
         1024 WIDTH @    2 + /    ;
: INIT   ( width --- )
         WIDTH !    0 MEMORY !    0 ADDRESS !    180 ORIGIN !
         MASK 68 ERASE    ;
```

```
   62 LIST

(    ADDRESS MAINTAINENCE, CHT, 5-JUN-83 )
: CODE-BUFFER    ( --- addr , buffer memory address for next
                   microcode.)
         MEMORY @ MAX# /MOD     ORIGIN @ + BLOCK
         SWAP    WIDTH @ 2 +    * +    UPDATE    ;
: DEFAULT ( Fill code with defualt value.)
         CODE-BUFFER    ADDRESS @ OVER !
         2 +    MASK SWAP     WIDTH @ MOVE    ;
: $      ( --- , terminate a microinstruction)
         1 MEMORY +!    1 ADDRESS +!
         MEMORY @ ORIGIN @ BLOCK !    DEFAULT    UPDATE    ;
: ORG    ( n --- )
         ADDRESS !    DEFAULT    ;
```

EXTENDED PRECISION NUMBER OPERATORS

EOR	OR the extended precision number in PAD with the extended precision number in addr. Result is put back in addr.
EAND	AND the extended precision numbers in PAD and in addr. Result ends in addr.
ENOT	Bitwise complementing the extended precision number in PAD.
SHIFT	Left shift 'value' by 'n' bits and put the resulting extended precision number in PAD.
PATTERN	Return a integer with rightmost 'width' bits set to 1.

BIT FIELD DEFINITION

BIT	Create a bit field with a default bit value. When the created bit field instruction is later executed, toggle this default bit in the current microcode.

FIELD DEFINITIONS

FIELD	Create a field in the microcode, starting at the 'start' bit with width of 'width' bits. Set this field in the default microcode in MASK according to 'default'. When the created field operator is later executed, insert the pattern on stack into the appropriate field in the current microcode.
LABEL	Name the current microcode address so that it can be later referred to by other microcodes.
FILLER	Leave a gap in the microcode field definition.
PROGRAM	Initialize the microprogram file. Copy the following string into the 0th record as the program name.

```
   63 LIST

(    EXTENDED NUMBER OPERATORS, CHT, 5-JUN-83)
: EOR    ( addr --- , OR the extended number in PAD with that
                      in addr, and store result in addr .)
         WIDTH @ 0 DO    I OVER + DUP @
         PAD I + @    OR    SWAP !    2 +LOOP    DROP    ;
: EAND   ( addr --- )    WIDTH @ 0 DO    I OVER + DUP @
         PAD I + @    AND   SWAP !    2 +LOOP    DROP    ;
: ENOT   ( ---, complement PAD)    WIDTH @ 0 DO    I PAD + DUP @
         -1 XOR    SWAP !    2 +LOOP    ;
: SHIFT  ( value n --- , shift value left n binary digits and
                         put resulting extended number in PAD.)
         PAD WIDTH @    ERASE    16 /MOD >R
         DUP IF    0 SWAP 0 DO    2DUP D+    LOOP    THEN
         WIDTH @ 4 -    R> 2* - PAD +    2!    ;
: PATTERN ( width --- n , pattern cover width with 1's )
         0 SWAP 0 DO    2* 1+    LOOP    ;


   64 LIST

(      BIT FIELD DEFINITION, CHT, 7-JUN-83)
: BIT    CREATE    ( start defalut --- next )
         OVER SHIFT    MASK EOR    DUP ,    1+
         DOES>     ( --- , toggle the designated bit field)
         @    16 /MOD >R    ?DUP
         IF 1 SWAP 0    DO 2* LOOP    ELSE 1 THEN
         WIDTH @    R> 2* -    CODE-BUFFER +    DUP @
         ROT XOR SWAP !    ;


   65 LIST

(    FIELD DEFINITION, CHT, 5-JUN-83)
: FIELD    CREATE     ( start width default --- next )
           >R 2DUP , ,    OVER R> SWAP SHIFT    MASK EOR    +
           DOES>    2@ ( value start width --- )
           PATTERN OVER SHIFT    ENOT    CODE-BUFFER 2 + DUP EAND
           >R SHIFT R>    EOR    ;
: LABEL    ( --- , name the current microcode address)
           ADDRESS @ CONSTANT ;
: FILLER   ( code-position width --- next-positon )
           +    ;
: PROGRAM  ( Name 0th record.)
           ORIGIN @ BLOCK >R    I WIDTH @ 2 + ERASE
           32 WORD DUP C@    SWAP 1+ R> 2 +    ROT MOVE
           1 MEMORY !    ;
```

PRINTING AND DUMPING THE MICROCODES

DBASE	A two integer array keeping the base and width to format the output codes.
.BINARY	Print microcodes in binary format. Each 16 bit code is printed in a 17 column field.
.DECIMAL	Print decimal numbers in 6 column fields.
.HEX	Print hexadecimal numbers in 5 column fields.
.OCTAL	Print octal numbers in 7 column field.
.NUMBER	Change base and print the stack number according to the format specified by DBASE.
DELAY	Delay loop for the slow printer to catch up.
MDUMP	Dump the microprogram file beginning at the block determined by ORIGIN. The microcode format is specified by one of the above DBASE instructions.

MICROASSEMBLY EXAMPLES

The examples are quite straightforward and do not need extensive commentary. Generally, it starts with the initial declaration as to the microcode width, program origin and program name. The second phase is the field definitions where fields in the microcode are declared and default values assigned. Multiple and overlapping fields can be declared, but care must be exercised in assigning the default values. The lastly declared default value will be the one gets into MASK. Then microcode address is assigned and assembly process can begin. As many field operators as needed are executed to insert values into their respective fields until the closing command '$', which initializes the next microcode.

MACRO FACILITIES

Many microcodes share identical field command phrases. These recuring phrases can be defined as colon definitions which improve the program clarity and also reduce coding efforts. Many examples are shown in the last two large programs.

```
   66 LIST

(    MICROCODE DUMP, CHT, 5-JUN-83)
VARIABLE DBASE 2 ALLOT    ( base, field)
: .BINARY    17 2 DBASE 2!    ;
: .DECIMAL   6 10 DBASE 2!   ;
: .HEX       5 16 DBASE 2!   ;
: .OCTAL     7 8 DBASE 2!    ;
: .NUMBER    ( n --- )    BASE @ >R    DBASE 2@
          BASE !    U.R    R> BASE !    ;
: DELAY    32000 0 DO LOOP ;
: MDUMP    ( --- , dump the microcodes from block ORIGIN.)
          CR CR ORIGIN @ BLOCK DUP @    SWAP 2 + WIDTH @ TYPE CR
          1 DO    CR    I MEMORY !    CODE-BUFFER
          DUP @ 5 U.R 2 SPACES    2 +    WIDTH @ 0 DO
          I OVER + @    .NUMBER   2 +LOOP    DROP DELAY    LOOP
          1 MEMORY +!    ;
.BINARY


   67 LIST

(    EXAMPLES PER G. E. CHOLMONDELEY, FD III/4, P. 126)
4 INIT    181 ORIGIN !

PROGRAM 1EXAMPLE

0 8 255 FIELD HH    16 65535 FIELD BB    8 255 FIELD AA    DROP
0 8 FILLER    12 4095 FIELD DD    4 15 FIELD CC    DROP
0 8 FILLER     2 3 FIELD FF    10 1023 FIELD EE    DROP
0 8  FILLER     16 65535 FIELD GG    DROP

512 ORG

  0 AA $    0 BB $    0 CC $    0 DD $
  0 EE $    0 FF $    0 GG $    0 HH $



   68 LIST

(    BIT FIELD TESTS, CHT, 7-JUN-83)
4 INIT    182 ORIGIN !
PROGRAN TEST
0 1 BIT 1ST    0 BIT 2ND
  16 FILLER
  1 BIT XND    0 BIT YND    1 BIT ZND
  DROP

0 ORG
$    1ST $    2ND $    XND $    YND $    ZND $
1ST 2ND XND YND ZND $
```

lEXA

512	11111111	1111111111111111
513	1111111100000000	11111111
514	1111111100001111	1111111111111111
515	1111111111110000	11111111
516	1111111111110000	1111111111
517	1111111111111111	1111110011111111
518	1111111100000000	11111111
519	1111111111111111	1111111100000000

TEST

0	10100	1
1	10100	0
2	10100	11
3	10000	1
4	11100	1
5	100	1
6	1000	10

AMD-EXMP

240	0	0	1111100000	100111111111001
241	0	0	1111111110	1000111111111111
242	0	0	1111100000	1101111111111111
244	0	0	100000	1111000111110000
248	0	0	100000	1111001011110000
252	0	0	100000	1111001111110000
504	0	0	111011	1001000111111000
512	0	0	100001	1100111111110001
513	0	0	111001	1101111111111111
514	0	0	111110	1001001000000010
515	0	0	111011	11001000000101
516	0	0	100000	10111111111111
517	0	0	100000	10111111111111

4. Microcodes of Simple Examples

```
   69 LIST

( AMD EXAMPLES LOAD BLOCK, CHT, 6-JUN-83)

70 LOAD    ( field definitions)
71 LOAD    ( 29811 & 29803 instruction mnemonics)
72 LOAD    ( example 1)
73 LOAD    ( example 2)
74 LOAD    ( example 3)

MDUMP      ( display microcodes)
PRINT MDUMP  ( print microcodes)
```

```
   70 LIST

(    EXAMPLE PER AMD HANDBOOK, CHT, 6-JUN-83)
8 INIT    ( word length 64 bits )      183 ORIGIN !
HEX
0 C  FFF FIELD NUMB    ( address or counter value)
  4  0   FIELD NACU    ( next address control unit instruction)
  1  0   FIELD POLARITY
  4  0   FIELD TEST    ( conditional test select)
  1  1   FIELD DB      ( data bus read-in)
  4  0   FIELD BCU     ( branch control unit 29803)
  DROP
DECIMAL

15 CONSTANT CNTR
 1 CONSTANT INV
 0 CONSTANT IN
```

```
   71 LIST

: JZ     0 NACU ;          : NOT     0 BCU ;
: CJS    1 NACU ;          : T0      1 BCU ;
: JMAP   2 NACU ;          : T1      2 BCU ;
: CJP    3 NACU ;          : T10     3 BCU ;
: PUSH   4 NACU ;          : T2      4 BCU ;
: JSRP   5 NACU ;          : T02     5 BCU ;
: CJV    6 NACU ;          : T12     6 BCU ;
: JRP    7 NACU ;          : T012    7 BCU ;
: RFCT   8 NACU ;          : T3      8 BCU ;
: RPCT   9 NACU ;          : T03     9 BCU ;
: CRTN  10 NACU ;          : T13    10 BCU ;
: CJPP  11 NACU ;          : T013   11 BCU ;
: LDCT  12 NACU ;          : T23    12 BCU ;
: LOOP  13 NACU ;          : T023   13 BCU ;
: CONT  14 NACU ;          : T123   14 BCU ;
: JP    15 NACU ;          : T0123  15 BCU ;
```

```
   72 LIST

(    AMD EXAMPLE 1, CHT, 6-JUN-83)
PROGRAM AMD-EXMPL
HEX
0F0 ORG   LABEL SWAP
   FF9 NUMB   0 TEST   PUSH   T0123  $
   RFCT   CNTR TEST   T0123          $
   CJV   0 TEST   T0123              $
0F4 ORG   LABEL ORTEST2
   0 TEST   JP   1F0 NUMB   $
0F8 ORG   LABEL ORTEST
   0 TEST   JP   2F0 NUMB   $
0FC ORG   LABEL ORTEST23
   0 TEST   JP   3F0 NUMB   $
DECIMAL
```

```
   73 LIST

(    AMD EXAMPLE 2, CHT, 15-MAY-83)
OCTAL
770 ORG
LABEL ALIGN
   770 NUMB   15 TEST   INV POLARITY   RPCT   $
DECIMAL
```

```
   74 LIST

(    AMD EXAMPLE 3, CHT, 15-MAY-83)
OCTAL
1000 ORG
LABEL DIVIDE
   LDCT   0 TEST   INV POLARITY   7761 NUMB   $
   14 TEST   INV POLARITY   CJV   $
   RPCT   CNTR TEST   ADDRESS @ NUMB   $
   15 TEST   INV POLARITY   CJP   ADDRESS @ 2 + NUMB   $
   0 TEST   JMAP   $
   0 TEST   JMAP   $
DECIMAL
```

```
   75 LIST

( SUPER-16 COMPUTER, MICK-BRICK, CHT, 7-JUN-83)
: THRU   1+ SWAP DO I I . LOAD LOOP ;
12 INIT   PROGRAM SUPERSIXTEEN
76 83 THRU
EXIT
```

```
   76 LIST

( AMD SUPER SIXTEEN, FIELD DEFINITIONS, CHT, 9-JUN-83)  HEX
0 10 0 FIELD ADDR   4 0 FIELD NAC
  6  0 FIELD TEST   2 0 FIELD CARRY
  1 BIT CEU   1 BIT CEM   1 BIT EOVR   1 BIT ES   1 BIT EC
  1 BIT EZ    1 BIT OECT
  8  0 FIELD CNTLB   1 BIT SHIFTEN   1 BIT INTRIEN  1 BIT INTDIS
  1 BIT IOEN   0 BIT ROM   0 BIT IMM   0 BIT MWORD   0 BIT WRITE
  1 BIT HREQ   0 BIT MREQ   0 BIT REQB
  3 0 FIELD B-RAM   3 0 FIELD A-RAM   3 1 FIELD SOURCE
  0 BIT SUBPCU   1 BIT QEU   1 BIT BRIEN   1 BIT SHTCNTEN
  1 BIT PSW   1 BIT ENZ0   0 BIT ZII   0 BIT LDD   0 BIT LDMAR
  2 0 FIELD TRANS   0 BIT INC   1 BIT ENCTR   0 BIT LDTREG
  1 BIT ENTREG   0 BIT I0   4 8 FIELD FUNCT   4 C FIELD DEST
  1 BIT OEB   1 BIT OEY   1 BIT EA   1 BIT WORD
  0 BIT CCEN   0 BIT ZZI   2 FILLER   1 BIT RTB   DROP
DECIMAL
```

```
   77 LIST

(    2910 SEQUENCER DEFINITIONS, CHT, 4-JUN-83)
: JZ   0 NAC   ;              : CJS   1 NAC ADDR  ;
: JSB   CCEN 1 NAC ADDR ;     : JMAP   2 NAC   ;
: CJP   3 NAC ADDR ;          : JMP   CCEN 3 NAC ADDR ;
: PUSH  4 NAC ADDR ;          : PHLC  CCEN 4 NAC ADDR ;
: JSRP  5 NAC ADDR ;          : CJV   6 NAC ;
: JMPV  CCEN 6 NAC ;          : JRP   7 NAC ADDR ;
: RFCT  8 NAC ADDR ;          : RPCT  CCEN 8 NAC ADDR ;
: CRTN  10 NAC ;              : RTN   CCEN 10 NAC ;
: CJPP  11 NAC ADDR ;         : LDCT  12 NAC ADDR ;
: LOOP  13 NAC ;              : CONT  14 NAC   ;
: TWB   15 NAC ADDR ;
```

```
   78 LIST

(    ALU OPERATIONS, CHT, 3-JUN-83)
( ALU functions and destination controls are separated.)
79 4 0 FIELD FUNC   4 0 FIELD DEST   DROP
: ADR 0 DEST ;      : LDR 1 DEST ;     : ADRQ 2 DEST ;
: LDRQ 3 DEST ;     : RPT 4 DEST ;     : LDQP 5 DEST ;
: QPT 6 DEST ;      : RQPT 7 DEST ;    : AUR 8 DEST ;
: LUR 9 DEST ;      : AURQ 10 DEST ;   : LURQ 11 DEST ;
: YBUS 12 DEST ;    : LUQ 13 DEST ;    : SINEX 14 DEST ;
: REG 15 DEST ;
: HIGH 0 FUNC ;      : SURS 1 FUNC ;      : SUSR 2 FUNC ;
: ADD 3 FUNC ;       : PASS 4 FUNC ;      : COMPLS 5 FUNC ;
: PASSR 6 FUNC ;     : COMPR 7 FUNC ;     : LOW 8 FUNC ;
: NOTRS 9 FUNC ;     : EXNOR 10 FUNC ;    : EXOR 11 FUNC ;
: AND 12 FUNC ;      : NOR 13 FUNC ;      : NAND 14 FUNC ;
: OR 15 FUNC ;
```

```
   79 LIST

(    ALU OPERAND SOURCES, CHT, 4-JUN-83)
: AB OEB EA ;       : ADB EA ;       : AQ I0 EA ;
: DAB OEB ;         : DADB  ;        : DAQ I0 ;
HEX
( SPECIAL FUNCTIONS)
: USMUL 0 DEST ;       : TCMUL 2 DEST ;
: INCTWO 4 DEST ;      : SMTC  5 DEST ;
: TCMULS 6 DEST ;      : SLN  8 DEST ;
: DLN   A DEST ;       : TCDIV C DEST ;
: TCDC  E DEST ;
DECIMAL
```

```
   80 LIST

(    2901 PCU DEFINITIONS, CHT, 7-JUN-83)      OCTAL
: AQPCU 0 SOURCE ;    : ABPCU 1 SOURCE ;    : ZQPCU 2 SOURCE ;
: ZBPCU 3 SOURCE ;    : ZAPCU 4 SOURCE ;    : DAPCU 5 SOURCE ;
: DBPCU 6 SOURCE ;    : DZPCU 7 SOURCE ;

: A0 0 A-RAM ;   : A1 1 A-RAM ;   : A2 2 A-RAM ;   : A3 3 A-RAM ;
: A4 4 A-RAM ;   : A5 5 A-RAM ;   : A6 6 A-RAM ;   : A7 7 A-RAM ;
: B0 0 B-RAM ;   : B1 1 B-RAM ;   : A2 2 B-RAM ;   : B3 3 B-RAM ;
: B4 4 B-RAM ;   : B5 5 B-RAM ;   : B6 6 B-RAM ;   : B7 7 B-RAM ;

: NEXTPCU ABPCU A4 B0 ;          : PUSHPCU SUBPCU ABPCU A4 B1 ;
: POPPCU  ABPCU A4 B1 ;          : JUMPPCU DZPCU ;
: TR2PCU  DAPCU A4 ;             : NOPPCU  ZBPCU ;
: SPPCU   ZBPCU A1 B1 ;          : DEC4 SUBPCU DAPCU A5 ;
DECIMAL
```

```
   81 LIST

(    2904 STATUS BITS, CHT, 9-JUN-83)
: EQ0 0 CARRY ;
: EQ1 1 CARRY ;
: EQCI  2 CARRY ;
: EQST  3 CARRY ;
```

```
   82 LIST

(    DATA PATH BIT DEFINITIONS, CHT, 7-JUN-83)

: YMAR 0 TRANS ;
: PCUY 1 TRANS ;
: PCUMAR 3 TRANS ;
```

```
   83 LIST

(    STROBE AND MEMORY CONTROL, CHT, 7-JUN-83)
: IMMD   ( value --- )   IMM ADDR ;
```

```
   84 LIST

( SUPER-16 MICROPROGRAM LOADER, CHT, 10-JUN-83)
85 99 THRU
103 106 THRU
100 102 THRU
107 LOAD
```

```
   85 LIST

(    RESET SEQUENCE, CHT, 9-JUN-83)
: (START)   YBUS PASS WORD OEY   LDMAR   REQB MWORD   ;
HEX
0 ORG   LABEL RESET
  WORD   IOEN INTDIS   127 CNTLB   REQB MREQ MWORD
  NOPPCU   180 ( INIT) JMP   $
LABEL START
  AB (START)   NOPPCU   CONT   $
LABEL 1START
  AB (START)   NEXTPCU   CONT   $
LABEL 2START
  DAB (START)   ZII   MREQ   NEXTPCU   JMAP   $
LABEL RNI
  DAB (START)   ZZI ZII MREG   NEXTPCU   JMAP   $
DECIMAL
```

```
   86 LIST

(    RR INSTRUCTIONS, CHT, 8-JUN-83)
: LOAD$   REG AB OEY WORD   ZZI LDMAR ZII   REQB MREQ MWORD
        NEXTPCU   JMAP   $   ;
: FLAGS   EZ EC ES EOVR CEM   ;

LABEL LR
  PASSR        LOAD$
LABEL AR
  ADD FLAGS    LOAD$
LABEL SR
  SURS FLAGS   LOAD$
LABEL NR
  AND FLAGS    LOAD$
```

```
   87 LIST

(    RR INSTRUCTIONS, CHT, 9-JUN-83)

LABEL ORR
  OR FLAGS      LOAD$
LABEL CLR
  SURS FLAGS EQ1    LOAD$
LABEL XORR
  EXOR FLAGS    LOAD$
```

```
   88 LIST

(    RX INSTRUCTIONS, CHT, 8-JUN-83)
HEX
: LDX     YBUS ADD DAB OEY WORD    RTB    YMAR LDMAR ENZ0
          REQB MREQ MWORD    ;
: LDX$    NOPPCU    1B ( FETCHOP) JSB    $   ;
: RXOP$   REG   DAB OEY WORD    LDMAR ZII ENZ0    REQB MREQ MWORD
          NEXTPCU    JMAP    $    ;
DECIMAL
LABEL LD
  LDX$
  PASSR RXOP$
LABEL ST
  LDX    NOPPCU    CONT    $
  YBUS PASS AB OEY WORD    ZZI LDD    REQB MWORD    NOPPCU CONT $
  AB (START)    MREQ WRITE    NEXTPCU    RNI JMP    $
```

```
   89 LIST

(    RX INSTRUCTION 1, CHT, 8-JUN-83)
LABEL ADDX
  LDX$
  ADD RXOP$
LABEL SUBXX
  LDX$
  SURS RXOP$
LABEL N
  LDX$
  AND RXOP$
```

```
   90 LIST

(     RX INSTRUCTION 2, CHT, 8-JUN-83)    HEX
LABEL O
  LDX$
  OR RXOP$
LABEL CMP
  LDX$
  SURS EQ1 RXOP$
LABEL FETCHOP
  YBUS PASSR DAB OEY WORD    ZZI LDMAR    REQB MREQ MWORD
  NEXTPCU    RTN    $
```

```
   91 LIST

(     IMMEDIATE INSTRUCTIONS, CHT, 8-JUN-83)
: IM$    REG DAB OEY WORD    LDMAR ENZ0    REQB MREQ MWORD
         NEXTPCU    2START JMP    $    ;
LABEL LI
  PASSR IM$
LABEL NI    ( AND IMMEDIATE)
  AND FLAGS IM$
LABEL OI    ( OR IMMEDIATE)
  OR FLAGS IM$
LABEL XI    ( XOR IMMEDIATE)
  EXOR FLAGS IM$
```

```
   92 LIST

(     IMMEDIATE INSTRUCTIONS 1, CHT, 8-JUN-83)
LABEL AI    ( ADD IMMEDIATE)
  ADD FLAGS IM$
LABEL SI    ( SUBTRACT IMMEDIATE)
  SURS EQ1 FLAGS IM$
LABEL CI    ( COMPARE IMMEDIATE)
  YBUS SURS DAB EQ1 OEY WORD    LDMAR ENZ0    REQB MREQ MWORD
  EZ EC ES EOVR CEM    NEXTPCU    2START JMP    $
```

```
   93 LIST

(    BRACH INSTRUCTIONS, CHT, 10-JUN-83)
: 1ALU    YBUS PASS AB OEY WORD    ;
: 2ALU    YBUS PASSR DAB OEY WORD    ;
: 3ALU    REG PASSR DAB OEY WORD    ;
: 4ALU    YBUS ADD DAB OEY WORD    RTB    REQB MREQ MWORD    ;
: 1MEM    REQB MREQ MWORD    ;

LABEL BX
  4ALU    ENTREG LDMAR    TR2PCU 2START JMP    $
LABEL BX1
  1ALU    ENTREG LDMAR    1MEM    TR2PCU 2START JMP    $
LABEL BC
  4ALU    LDTREG YMAR LDMAR    ENZ0 BRIEN    OCTAL 57 TEST DECIMAL
  OECT    NOPPCU    BX1 CJP    $
```

```
   94 LIST

(    BRANCH INSTRUCTIONS 1, CHT, 10-JUN-83)
LABEL BC1
  1ALU    ZZI LDMAR    REQB MWORD    NEXTPCU    RNI JMP    $
LABEL BAL
  YBUS ADD DAB OEY WORD    RTB    LDTREG YMAR LDMAR ENZ0
  MWORD    NOPPCU    CONT    $
LABEL BL1
  REG PASS DAB WORD    PCUY    REQB MWORD    NOPPCU    BX1 JMP $
LABEL BLR
  2ALU    LDTREG YMAR LDMAR ENZ0    MWORD    NOPPCU    BL1 JMP $
LABEL BR
  YBUS PASS DAB OEY WORD    LDTREG YMAR LDMAR    REQB MWORD
  NOPPCU    BX1 JMP    $
```

```
   95 LIST

(    SHIFT INSTRUCTIONS, CHT, 10-JUN-83)
HEX
: SH$    1ALU    ZZI SHTCNTEN    MWORD    NOPPCU  FFF LDCT    $    ;
: SHOP$    PASS AB OEY WORD    ( N -- ) CNTLB    MWORD
        SHIFTEN FLAGS    NOPPCU    ADDRESS @ RPCT    $    ;
: SHEND$    1ALU    LDMAR ZII    1MEM    NEXTPCU    JMAP    $    ;
LABEL SLA
  SH$    F0 AUR SHOP$    SHEND$
LABEL SLL
  SH$    F0 LUR SHOP$    SHEND$
LABEL SRA
  SH$    E0 ADR SHOP$    SHEND$
LABEL SRL
  SH$    E0 LDR SHOP$    SHEND$
DECIMAL
```

```
   96 LIST

(    SHIFT INSTRUCTIONS 1, CHT, 10-JUN-83)
HEX
LABEL RRL
  SH$   0EA LDR SHOP$   SHEND$
LABEL RRC
  SH$   E9 LDR SHOP$   SHEND$
LABEL RLC
  SH$   F9 LUR SHOP$   SHEND$
DECIMAL
```

```
   97 LIST

(    I/O INSTRUCTIOONS, CHT, 10-JUN-83)
HEX
: IN1   WORD OEY IOEN   ( N ---) CNTLB   HREQ MWORD NOPPCU   ;
LABEL IN
  LDX   1 LDCT   $
  FD IN1   ADDRESS @ RPCT   $
  FF IN1   CONT   $
  3ALU   LDMAR ENZ0   REQB MWORD   NOPPCU   1START JMP   $
LABEL OUT
  LDX   CONT   $
  1ALU   IOEN   FB CNTLB   LDD   HREQ MWORD   NOPPCU   1 LDCT $
  OEY WORD   HREQ MWORD   NOPPCU   ADDRESS @ RPCT   $
  OEY WORD   IOEN   FF CNTLB   HREQ MWORD   NOPPCU
      START JMP   $
DECIMAL
```

```
   98 LIST

(    STACK INSTRUCTIONS, CHT, 10-JUN-83)
OCTAL
: SPOP   1ALU   1MEM   ;
LABEL PUSH
  SPOP   ZZI   NOPPCU   CONT   $
  SPOP   LDMAR LDD   PUSHPCU   CONT   $
  SPOP   ENCTR INC   WRITE   70 TEST   NOPPCU   PUSH 1+ CJP   $
  SPOP   LDMAR   NOPPCU   RNI JMP   $
LABEL POP
  SPOP   ZZI   NOPPCU   CONT   $
  REG PASS AB OEY WORD   LDMAR ENZ0   REQB MWORD   SPPCU CONT $
  SPOP   ENCTR   70 TEST   POPPCU   POP 1+   CJP   $
  SPOP   LDMAR   NOPPCU   RNI JMP   $
DECIMAL
```

```
   99 LIST

(    STACK INSTRUCTIONS 1, CHT, 10-JUN-83)
LABEL CALL
  4ALU   LDTREG LDMAR ENZ0   PUSHPCU   CONT   $
  SPOP   PCUY LDD   OEY ( !)   NOPPCU   CONT   $
  SPOP   ENTREG LDMAR   WRITE   JUMPPCU   1START JMP   $
LABEL RETURN
  SPOP   LDMAR   MREQ   SPPCU   CONT   $
  SPOP   LDMAR   SPPCU   CONT   $
  2ALU   LDTREG   YMAR LDMAR ENZ0   1MEM   POPPCU   CONT   $
  1ALU   ENTREG   MWORD   JUMPPCU   1START JMP   $
```

```
  100 LIST

(    IO WRITE INSTRUCTION, CHT, 10-JUN-83)
HEX
LABEL IOW
  1ALU   ROM   10 CNTLB   YMAR LDMAR   1MEM   NOPPCU   CONT   $
  1ALU   ROM   20 CNTLB   LDD   1MEM   NOPPCU   CONT   $
  WORD IOEN   FB CNTLB   1MEM   NOPPCU   1 LDCT   $
  WORD   HREQ MWORD   NOPPCU   ADDRESS @ RPCT   $
  WORD IOEN   FF CNTLB   1MEM   NOPPCU   RTN   $
DECIMAL
```

```
  101 LIST

(    VECTOR JUMP INSTRUCTIONS, CHT, 10-JUN-83)
HEX
1D0 ORG
: INT$   MWORD   DZPCU A0 B0   10 IMMD   CONT   $   ;
: INTOP   WORD   INTDIS INTRIEN   FD CNTLB   ;
: INTEND$   WORD IOEN   FF CNTLB   LDMAR
        REQB MWORD   NOPPCU   1START JMP   $   ;
LABEL 0INT
  WORD IOEN   F7 CNTLB   10 INT$
  INTOP   NOPPCU   CONT   $
  INTEND$
LABEL 1INT
  INTOP 14 INT$   INTEND$
DECIMAL
```

```
  102 LIST

(     VECTOR JUMP 1, CHT, 10-JUN-83)
HEX
LABEL 2INT
  INTOP 18 INT$    INTEND$
LABEL 3INT
  INTOP 1C INT$    INTEND$
LABEL 4INT
  INTOP 20 INT$    INTEND$
LABEL 5INT
  INTOP 24 INT$    INTEND$
LABEL 6INT
  INTOP 28 INT$    INTEND$
LABEL 7INT
  INTOP 2C INT$    INTEND$
DECIMAL


  103 LIST

(     INTERRUPT INSTRUCTIONS, CHT, 10-JUN-83)
HEX
LABEL LIM
  2ALU    INTRIEN    FE CNTLB    LDMAR ENZ0    1MEM
     NEXTPCU    2START JMP    $
LABEL EI
  1ALU    INTRIEN    FF CNTLB    ZZI LDMAR ZII    1MEM
     NEXTPCU    JMAP    $
LABEL DI
  1ALU    INTRIEN    FD CNTLB    ZZI LDMAR ZII    1MEM
     NEXTPCU    JMAP    $
DECIMAL


  104 LIST

(     INTERRUPT INSTRUCTIONS 1, CHT, 10-JUN-83)
HEX
LABEL RTI
  1ALU    LDMAR    REQB MWORD    SPPCU    CONT    $
  2ALU    LDTREG ENZ0    1MEM    POPPCU    CONT    $
  1ALU    ENTREG    MWORD    JUMPPCU    CONT    $
  1ALU    INTRIEN    F9 CNTLB    LDMAR REQB MWORD    SPPCU CONT    $
  1ALU    INTRIEN    FF CNTLB    0 TEST
     ENZ0    1MEM    FLAGS    POPPCU    START JMP    $
DECIMAL
```

```
  105 LIST

(    INSTALLATION, CHT, 10-JUN-83)
HEX
180 ORG
LABEL INITIAL
  3ALU   HREQ MWORD   ROM   0 CNTLB   0 IMMD   CONT   $
  WORD INTRIEN   F0 CNTLB   HREQ MWORD
     DZPCU A0 B0   0 IMMD   CONT   $
  WORD INTRIEN   F8 CNTLB   HREQ MWORD
     DZPCU A1 B1   4000 IMMD   CONT   $
  WORD   DZPCU A4 B4   HREQ MWORD   2 IMMD   CONT   $
  WORD   DZPCU A5 B5   HREQ MWORD   4 IMMD   CONT   $
  WORD   IOEN   FF CNTLB   HREQ MWORD   NOPPCU   CONT   $
DECIMAL
```

```
  106 LIST

(    INSTALLATION 1, CHT, 10-JUN-83)
HEX
LABEL CONSOLE
  3ALU   ROM   10 CNTLB   FFFB IMMD   CONT   $
  3ALU   ROM   20 CNTLB   CE IMMD     CONT   $
  WORD   18C ( IOW ) JSB   $
  3ALU   ROM   20 CNTLB   35 IMMD   CONT   $
  WORD  18C JSB   $
  NOPPCU   START JMP   $
DECIMAL
```

```
  107 LIST

(    INTERRUPT HANDLERS, CHT, 10-JUN-83)
HEX
1F0 ORG
LABLE INTERRUPT
  WORD   1MEM WRITE   DEC4   CONT   $
  WORD   LDMAR MWORD   PUSHPCU   CONT   $
  WORD   PCUY LDD   REQB MWORD   NOPPCU   CONT   $
  WORD   INTRIEN   F5 CNTLB   1MEM WRITE   NOPPCU   JMPV   $
1FF ORG
LABEL ENTRY
  WORD   LDMAR LDD PSW   1MEM   PUSHPCU   INTERRUPT JMP   $
DECIMAL
```

```
 0  1000101111100100   101000000011111   110000000111010    1100100111111 1111000000000011        110000000
 1  1000000001100010   101000100111111   110000000101010  1111100000000111 1111000000001110              0
 2  1000000001100010   101000100111111    1100000101010  1111100000000111 1111000000001110              0
 3  1000001001100010   101000101111111    1100000111010  1111100000000111 1111000000000010              0
 4  1001001001100010   101000101111111    1100000111010  1111100000000111 1111000000000010              0
 5  1001000001111011   101000101111111    1100000111010  1111100000000111 1111000000000010              0
 6  1001000001111001 101010001011111     1100000111010  1111100000000100    1000000000010              0
 7  1001000001111000 101010001011111     1100000111010  1111100000000100    1000000000010              0
 8  1001000001111110   101000101111111    1100000111010  1111100000000100    1000000000010              0
 9  1001000001111111 101010001011111     1100000111010  1111100000000100    1000000000010              0
10  1001000001111000 101010001011111     1100000111010  1111100000000100    1010000000010              0
11  1001000001111101 101010001011111     1100000111010  1111100000000100    1000000000010              0
12  1000111111100100   101000000011111  110000000001000  1111100000000111 1111000000000001          11011
13  1000001001111011   101000101011111    1100000111010  1111100000000111 1111000000000010              0
14        1001100001 101010001000111     1100000111010  1111100000000111 1111000000001110              0
15  1001000001100010   101000010111111  110000000101010  1111100000000111 1111000000001110              0
16  1000100001100010   101000100111111    1100000111110  1111100000000111 1111000000000011            100
17  1000111111100100   101000000011111  110000000001000  1111100000000111 1111000000000001          11011
18  1000001001111001 101010001010111     1100000111010  1111100000000111 1111000000000010              0
19  1000111111100100   101000000011111  110000000001000  1111100000000111 1111000000000001          11011
20  1000001001111000 101010001010111     1100000111010  1111100000000111 1111000000000010              0
21  1000111111100100   101000000011111  110000000001000  1111100000000111 1111000000000001          11011
22  1000001001111110   101000101011111    1100000111010  1111100000000111 1111000000000010              0
23  1000111111100100   101000000011111  110000000001000  1111100000000111 1111000000000001          11011
24  1000001001111111 101010001010111     1100000111010  1111100000000111 1111000000000010              0
25  1000111111100100   101000000011111  110000000001000  1111100000000111 1111000000000001          11011
26  1000001001111000 101010001010111     1100000111010  1111100000000111 1111010000000010              0
27  1001101001100011   101000100111111    1100000111010  1111100000000111 1111000000001010              0
28  1000101001111011   101000100011111    1100000111010  1111100000000111 1111000000000011             11
29  1000101001111110   101000100011111    1100000111010  1111100000000100    1000000000011             11
30  1000101001111111 101010001000111     1100000111010  1111100000000100    1000000000011             11
31  1000101001111101 101010001000111     1100000111010  1111100000000100    1000000000011             11
32  1000101001111001 101010001000111     1100000111010  1111100000000100    1000000000011             11
33  1000101001111000 101010001000111     1100000111010  1111100000000100    1010000000011             11
34  1000101001100000 101010001000111     1100000111010  1111100000000100    1010000000011             11
35      1010011000001 100010001011111  101100000111010  1111100000000111 1111000000000011             11
36  1000100001100010     100010011111  101100000111010  1111100000000111 1111000000000011             11
37        1001100001 101110001000110     1100000111010  1111100000000011 1111001011110011         100100
38  1001100001100010   101000100111111    1100000101010  1111100000000111 1111000000000011            100
39        1001100001 101110001000111     1100000001010  1111100000000111 1111000000001110              0
40  1000101101111010   101001000111111  110000000101010  1111100000000111 1111000000000011         100100
```

5. Partial List of Microcodes of Supersixteen

```
117 LIST
```

```
118 LIST
```

```
119 LIST

(   REVERSED FIELD DEFINITIONS, CHT, 12-JUN-83)
0 3 0 FIELD XLAT   1 BIT WRGA   1 BIT SAST   1 BIT RFIF
  1 BIT RDGA   1 BIT PREQ   1 BIT PL03   1 BIT PF03
  1 BIT PFPM   1 BIT PENB   1 BIT OUTP   1 BIT MWRT
  1 BIT MREA   1 BIT MADR   2 0 FIELD AM116   1 BIT INPT
  1 BIT CREQ   1 BIT CP20   1 BIT CE20   1 BIT CEL2
  1 BIT BT20   1 BIT BTU2   1 BIT BTL2   1 BIT BT16
  1 BIT BT03   1 BIT BOUT   1 BIT BFU2   1 BIT BFL2
  1 BIT BF16   1 BIT BF03   1 BIT BFTB   1 BIT BFCB
  1 BIT ADMC   1 BIT IF   5 0 FIELD NAC  10 1023 FIELD ADDRES
  4 0 FIELD AM10   0 BIT DLE   1 BIT IEN   1 BIT OEY
  0 BIT SRE   4 0 FIELD CT   5 0 FIELD DIST   4 0 FIELD SOURCE
  4 0 FIELD OPCODE   2 0 FIELD QUAD   1 BIT BYTE
DROP
```

```
  120 LIST

( AMD DISK CONTROLLER, CHT, 11-JUN-83)
: THRU 1+ SWAP DO   I . I LOAD   LOOP ;
10 INIT   PROGRAM DISK
119 LOAD
122 145 THRU
```

```
  121 LIST

(    FIELD DEFINITIONS, CHT, 11-JUN-83)
0 1 BIT BYTE   2 0 FIELD QUAD   4 0 FIELD OPCODE
  4 0 FIELD SOURCE   5 0 FIELD DIST   4 0 FIELD CT
  0 BIT SRE   1 BIT OEY   1 BIT IEN   0 BIT DLE
  4 0 FIELD AM10   10 1023 FIELD ADDRES   5 0 FIELD NAC
  1 BIT IF   1 BIT ADMC   1 BIT BFCB   1 BIT BFTB
  1 BIT BF03   1 BIT BF16   1 BIT BFL2   1 BIT BFU2
  1 BIT BOUT   1 BIT BT03   1 BIT BT16   1 BIT BTL2
  1 BIT BTU2   1 BIT BT20   1 BIT CEL2   1 BIT CE20
  1 BIT CP20   1 BIT CREQ   1 BIT INPT   2 0 FIELD AM116
  1 BIT MADR   1 BIT MREA   1 BIT MWRT   1 BIT OUPT
  1 BIT PENB   1 BIT PFPM   1 BIT PF03   1 BIT PL03
  1 BIT PREQ   1 BIT RDGA   1 BIT RFIF   1 BIT SAST
  1 BIT WRGA   3 0 FIELD XLAT
DROP
```

```
  122 LIST

(    SINGLE OPERAND INSTRUCTIONS, CHT, 11-JUN-83)
: MOVE 2 QUAD 12 OPCODE ;
: COMP 2 QUAD 13 OPCODE ;
: INC  2 QUAD 14 OPCODE ;
: NEG  2 QUAD 15 OPCODE ;
: RASO 0 SOURCE ;   : RYSO 2 SOURCE ;   : RSSO 3 SOURCE ;
: ARSO 4 SOURCE ;   : DRSO 6 SOURCE ;   : IRSO 7 SOURCE ;
: ZRSO 8 SOURCE ;   : ZERSO 9 SOURCE ;  : SERSO 10 SOURCE ;
: RRSO 11 SOURCE ;

: Z 3 QUAD ;  ( SINGLE OPERAND NON-RAM INSTRUCTIONS)
: ASO 4 SOURCE Z ;   : DSO 6 SOURCE Z ;   : ISO 7 SOURCE Z ;
: ZSO 8 SOURCE Z ;   : ZESO 9 SOURCE Z ;  : SESO 10 SOURCE Z ;
: NRY 0 DIST Z ;   : NRA 1 DIST Z ;   : NRS 4 DIST Z ;
: NRAS 5 DIST Z ;
```

```
  123 LIST

(    RAM REGISTERS, CHT, 11-JUN-83)
: R0 0 DIST ;    : R1 1 DIST ;    : R2 2 DIST ;    : R3 3 DIST ;
: R4 4 DIST ;    : R5 5 DIST ;    : R6 6 DIST ;    : R7 7 DIST ;
: R8 8 DIST ;    : R9 9 DIST ;    : R10 10 DIST ; : R11 11 DIST ;
: R12 12 DIST ; : R13 13 DIST ; : R14 14 DIST ; : R15 15 DIST ;
: R16 16 DIST ; : R17 17 DIST ; : R18 18 DIST ; : R19 19 DIST ;
: R20 20 DIST ; : R21 21 DIST ; : R22 22 DIST ; : R23 23 DIST ;
: R24 24 DIST ; : R25 25 DIST ; : R26 26 DIST ; : R27 27 DIST ;
: R28 28 DIST ; : R29 29 DIST ; : R30 30 DIST ; : R31 31 DIST ;
: N0 0 OPCODE ;    : N1 1 OPCODE ;    : N2 2 OPCODE ;
: N3 3 OPCODE ;    : N4 4 OPCODE ;    : N5 5 OPCODE ;
: N6 6 OPCODE ;    : N7 7 OPCODE ;    : N8 8 OPCODE ;
: N9 9 OPCODE ;    : N10 10 OPCODE ; : N11 11 OPCODE ;
: N12 12 OPCODE ; : N13 13 OPCODE ; : N14 14 OPCODE ;
: N15 15 OPCODE ;


  124 LIST

(    TWO OPERAND INSTRUCTIONS, CHT, 11-JUN-83)
: SBR 0 SOURCE ;    : SBRC 1 SOURCE ;    : SBS 2 SOURCE ;
: SBSC 3 SOURCE ;  : ADD 4 SOURCE ;     : ADC 5 SOURCE ;
: AND 6 SOURCE ;    : NAND 7 SOURCE ;    : EXOR 8 SOURCE ;
: NOR 9 SOURCE ;    : OR 10 SOURCE ;     : EXNOR 11 SOURCE ;
: RAATO 0 OPCODE ;    : RIATO 2 OPCODE ;    : DRATO 3 OPCODE ;
: RAYTO 8 OPCODE ;    : RIYTO 10 OPCODE ;  : DRYTO 11 OPCODE ;
: RARTO 12 OPCODE ;  : RIRTO 14 OPCODE ;  : DRRTO 15 OPCODE ;
( TWO OPERAND RAM-2)
: DARTO 2 QUAR 1 OPCODE ;    : AIRTO 2 QUAD 1 OPCODE ;
: DIRTO 2 QUAD 5 OPCODE ;
( TWO OPERAND NON-RAM)
: DATO 3 QUAD 1 OPCODE ;    : AITO 3 QUAD 2 OPCODE ;
: DITO 3 QUAD 5 OPCODE ;


  125 LIST

(    SHIFT INSTRUCTIONS, CHT, 11-JUN-83)
( SHIFT RAM )
: RRS 2 QUAD 6 OPCODE ;    : DRS 2 QUAD 7 OPCODE ;
: UPZSH 0 SOURCE ;    : UPISH 1 SOURCE ;    : UPLSH 2 SOURCE ;
: DNZSH 4 SOURCE ;    : DNISH 5 SOURCE ;    : DNLSH 6 SOURCE ;
: DNCSH 7 SOURCE ;    : DNOVSH 8 SOURCE ;

( SHIFT NON-RAM )
: ASH 3 QUAD 6 OPCODE ;    : DSH 3 QUAD 7 OPCODE ;
```

```
  126 LIST

(    ROTATE INSTRUCTIONS, CHT, 11-JUN-83)
( ROTATE RAM 1 )
: ARTR 12 SOURCE ;    : YRTR 14 SOURCE ;    : RRTR 15 SOURCE ;
( ROTATE RAM 2 )
: ARRT 1 QUAD 0 SOURCE ;    : DRRT 1 QUAD 1 SOURCE ;
( ROTATE NON RAM )
: DYRT 3 QUAD 12 SOURCE 24 DIST ;
: DART 3 QUAD 12 SOURCE 25 DIST ;
: AYRT 3 QUAD 12 SOURCE 28 DIST ;
: AART 3 QUAD 12 SOURCE 29 DIST ;
```

```
  127 LIST

(    BIT INSTRUCTIONS, CHT, 11-JUN-83)
( BIT INSTRUCTION RAM 1 )
: NRSET 3 QUAD 13 SOURCE ;    : NRRST 3 QUAD 14 SOURCE ;
: NRTST 3 QUAD 15 SOURCE ;
( BIT INSTRUCTION RAM 2 )
: NRLD2 2 QUAD 12 SOURCE ;    : NRLDC2 2 QUAD 13 SOURCE ;
: NRA2  2 QUAD 14 SOURCE ;    : NRS2   2 QUAD 15 SOURCE ;
( BIT INSTRUCTION NON RAM )
: Z 3 QUAD 12 SOURCE ;
: NATST Z 0 DIST ;    : NARST Z 1 DIST ;    : NASET Z 2 DIST ;
: NAA2  Z 4 DIST ;    : NAS2  Z 5 DIST ;    : NALD2 Z 6 DIST ;
: NALDC2 Z 7 DIST ;   : NDTST Z 16 DIST ;   : NDRST Z 17 DIST ;
: NDSET Z 18 DIST ;   : DYA2N Z 20 DIST ;   : DYS2N Z 21 DIST ;
: NYLD2 Z 22 DIST ;   : NYLDC2 Z 23 DIST ;
```

```
  128 LIST

(    ROTATE MERGE, CHT, 11-JUN-83)
( ROTATE AND MERGE )
: Z 1 QUAD ;
: DAIRM Z 7 SOURCE ;     : DARRM Z 8 SOURCE ;
: DRIRM Z 9 SOURCE ;     : DRARM Z 10 SOURCE ;
: ARIRM Z 12 SOURCE ;    : RAIRM Z 14 SOURCE ;
( ROTATE AND COMPARE)
: DAIC Z 2 SOURCE ;      : DRIC Z 3 SOURCE ;
: DRAC Z 4 SOURCE ;      : RAIC Z 5 SOURCE ;
```

```
  129 LIST

(    PRIORITIZE INSTRUCTIONS, CHT, 11-JUN-83)
( RAM ADDRESS MASKS )
: Z 2 QUAD ;
: 1APRT Z 7 SOURCE ;     : 1DPR Z 9 SOURCE ;
: 1APR  Z 8 OPCODE ;     : 1YPR Z 10 OPCODE ;
: 1RPR  Z 11 OPCODE ;
( PRIORITIZE RAM )
: 2APR  Z 0 SOURCE ;     : 2YPR Z 2 SOURCE ;
: APR   Z 8 OPCODE ;     : ZPR  Z 10 OPCODE ;
: IPR   Z 11 OPCODE ;
: 3RPR  Z 3 SOURCE ;     : 3APR Z 4 SOURCE ;
: 3DPR  Z 6 SOURCE ;
( PRIORITIZE NON RAM )
: APRT 3 QUAD 4 SOURCE ;    : DPRT 3 QUAD 6 SOURCE ;



  130 LIST

(    CYCLIC REDUNDANCY CHECK, CHT, 11-JUN-83)
: FCRC    ( N --- ) 2 QUAD 6 OPCODE 3 SOURCE ;
: RCRC    ( N --- ) 2 QUAD 6 OPCODE 9 SOURCE ;
: NOOP    BYTE 3 QUAD 8 OPCODE 10 SOURCE 0 DIST ;
          ( OCTAL 7140 DIST )









  131 LIST

(    SET STATUS, CHT, 11-JUN-83)
: Z BYTE 11 OPCODE 10 SOURCE 3 QUAD ;
: SONZC Z 3 DIST ;    : SL   Z 5 DIST ;    : SF1   Z 6 DIST ;
: SF2   Z 9 DIST ;    : SF3  Z 10 DIST ;

( RESET STATUS )
: Z BYTE 10 OPCODE 10 SOURCE 3 QUAD ;
: RONZC Z 3 DIST ;    : RL   Z 5 DIST ;    : RF1   Z 6 DIST ;

( SAVE STATUS RAM )
: SVSTR   2 QUAD 7 OPCODE 10 SOURCE ;

( SAVE STATUS NON RAM )
: SVSTNR  3 QUAD 7 OPCODE 10 SOURCE ;
: RF2   Z 9 DIST ;    : RF3  Z 10 DIST ;
```

```
  132 LIST

(    TEST STATUS, CHT, 11-JUN-83)
: Z   BYTE 3 QUAD 9 OPCODE 10 SOURCE ;
: TNOZ Z 0 DIST ;     : TNO  Z 2 DIST ;     : TZ   Z 4 DIST ;
: TOVR Z 6 DIST ;     : TLOW Z 8 DIST ;     : TC   Z 10 DIST ;
: TZC  Z 12 DIST ;    : TN   Z 14 DIST ;    : TL   Z 16 DIST ;
: TF1  Z 18 DIST ;    : TF2  Z 20 DIST ;    : TF3  Z 22 DIST ;

( IMMEDIATE OPERANDS )
64 16 0 FIELD IMME

( CT MULTIPLEXER CONTROL )
: NOZ  0 CT ;    : NO   1 CT ;    : Z    2 CT ;    : OVR  3 CT ;
: LOW  4 CT ;    : C    5 CT ;    : ZC   6 CT ;    : N    7 CT ;
: L    8 CT ;    : F1   9 CT ;    : F2  10 CT ;    : F3  11 CT ;



  133 LIST

(    AM2910 COMMANDS, CHT, 11-JUN-83 )
: AM1 AM10 ADDRES ;
: JZ  0 AM10 ;     : CJS  1 AM1 ;    : JS   2 AM1 15 NAC ;
: JMAP 2 AM10 ;    : CJP  3 AM1 ;    : JP   3 AM1 15 NAC ;
: PUSH 4 AM10 ;    : JSRP 5 AM1 ;    : CJV  6 AM1 ;
: JRP  7 AM1 ;    : RFCT 8 AM1 ;    : RPCT 9 AM1 ;
: CRTN 10 AM10 ;  : RTN  10 AM10 15 NAC ;   : CJPP 11 AM1 ;
: LDCT 12 AM1 ;   : LOOP 13 AM1 ;   : CONT 14 AM10 ;
: TWB  15 AM1 ;
( CONDITION CODE SELECTION)
: AE20  8 NAC ;    : CT16  9 NAC ;    : EP20 10 NAC ;
: ER20 11 NAC ;    : FAIL 12 NAC ;    : IRDY 13 NAC ;
: ORDY 14 NAC ;    : SUCC 15 NAC ;    : ATTN 16 NAC ;
: BACK 17 NAC ;    : BUSY 18 NAC ;    : INDX 19 NAC ;
: SAMD 20 NAC ;    : PM2  21 NAC ;    : PM3  22 NAC ;
: PM4  23 NAC ;


  134 LIST

(    MISCELLANEOUS, CHT, 11-JUN-83)
: JMPI   1 AM116 ;
: NOJMPI   2 AM116 ;
```

```
  135 LIST

( AMD DISK MICROCODES, CHT, 11-JUN-83)
```

```
  136 LIST

(    ADDITIONAL MNEMONICS, CHT, 11-JUN-83)
HEX
8005 CONSTANT MSKCRC    10 CONSTANT NITCRC
  40 CONSTANT NSPASS    41 CONSTANT RDITCT
  16 CONSTANT PF1       0D CONSTANT PF2
  59 CONSTANT PF3       17 CONSTANT PF4
E723 CONSTANT A1LSW      6 CONSTANT A1MSW
BFA8 CONSTANT A2LSW   D530 CONSTANT A3LSW
A928 CONSTANT A4LSW   7100 CONSTANT KL128
 477 CONSTANT KM128   EEE2 CONSTANT KLSW
   8 CONSTANT KMSW
DECIMAL
```

```
  137 LIST
```

```
  138 LIST

( SECTOR I/O, CHT, 11-JUN-83)
HEX    0 ORG
LABEL SECTIO
  N0 NRS2 R0    Z    CT16    1A ( CFCODE) CJP    $
  MOVE ISO NRY    OEY    NOJMPI    CONT    $
  0 IMME    OEY    BT20   NOJMPI    CONT    $
  MOVE ISO NRY    OEY    NOJMPI    CONT    $
  10 IMME    OEY    BT20    NOJMPI    7F LDCT    $
  MOVE RYSO R3    DLE OEY    NOJMPI    CONT    $
  MOVE DRSO R4    IEN    NOJMPI    CONT    $
DECIMAL
```

```
  139 LIST

(     CHECK BITS PRECALCULATION LOOP, CHT, 11-JUN-83)
HEX
LABEL PCPREL
  MOVE RYSO R4    OEY    MADR    NOJMPI    CONT    $
  NOOP    BTL2    NOJMPI    CONT    $
  NOOP    CP20    NOJMPI    CONT    $
  NOOP    NOJMPI    CONT    $
  NOOP    BTU2    NOJMPI    CONT    $
  INC RRSO R4    IEN    CP20    NOJMPI    PCPREL RPCT    $
  MOVE ISO NRY    OEY    NOJMPI    CONT    $
  11 IMME    OEY    BT20    NOJMPI    2 LDCT    $
DECIMAL
```

```
  140 LIST

(     STORE CHECK BITS IN BUFFER, CHT, 11-JUN-83)
HEX
LABEL SCBIBL
  MOVE RYSO R4    OEY    CP20    MADR    NOJMPI    CONT    $
  NOOP    CEL2    NOJMPI    CONT    $
  NOOP    NOJMPI    CONT    $
  NOOP    CP20    NOJMPI    CONT    $
  NOOP    NOJMPI    CONT    $
  INC RRSO R4    IEN    BFL2 BFU2 MWRT    NOJMPI    SCBIBL RPCT    $
  MOVE ZRSO R5    IEN    CP20    NOJMPI    CONT    $
  NOOP    CEL2    NOJMPI    CONT    $
  MOVE DRSO R5    IEN    BFL2 BT16    NOJMPI    CONT    $
  MOVE RYSO R4    OEY    MADR    NOJMPI    CONT    $
  MOVE RYSO R5    OEY    BF16 MWRT    NOJMPI    CONT    $
DECIMAL
```

```
  141 LIST

(    CONVERT CODE IN R0 TO MICROCODE ADDRESS, CHT, 11-JUN-83)
HEX
LABEL CFCODE
  MOVE ISO NRA    IEN    NOJMPI    CONT    $
  36 ( BRTABL)   IMME    IEN    NOJMPI    CONT    $
  ADD RAATO R0    IEN    NOJMPI    CONT    $
  MOVE ISO NRA    IEN    NOJMPI    CONT    $
  MSKCRC IMME    IEN    NOJMPI    CONT    $
  MOVE ZRSO R4    IEN    NOJMPI    CONT    $
  MOVE RYSO R1    IEN    NOJMPI    NITCRC LDCT    $
  MOVE DRSO R5    IEN    NOJMPI   CONT    $
DECIMAL
```

```
  142 LIST

(    SHIFT R5 AND SET QLINK, CHT, 11-JUN-83)
HEX
LABEL FL1CRC
  UPZSH RRSH R5    IEN    NOJMPI    CONT    $
  FCRC R4    IEN    NOJMPI    FL1CRC RPCT    $
  MOVE RYSO R2    IEN    NOJMPI    NITCRC LDCT    $
  MOVE DRSO R5    IEN    NOJMPI    CONT    $
LABEL FL2CRC
  UPZSH RRSH R5    IEN    NOJMPI    CONT    $
  FCRC R4    IEN    NOJMPI    FL2CRC RPCT    $
  NOOP    INPT NOJMPI RDGA    NSPASS LDCT    $
DECIMAL
```

```
  143 LIST

(    SECTOR PASS, CHT, 11-JUN-83)
HEX
LABEL 1SECTL
  NOOP    ADMC INPT NOJMPI RDGA RFIF    CONT    $
  NOOP    ADMC INPT NOJMPI RDGA    SAMD    ADDRESS @ CJP    $
  NOOP    INPT NOJMPI RDGA    IRDY    ADDRESS @ CJP    $
  MOVE DRSO R5    DLE IEN    BF03 BT16 INPT NOJMPI RDGA
     IRDY    ADDRESS @ CJP    $
  EXOR DRRTO R1    SRE Z    INPT NOJMPI RDGA    IF CT16
     33 ( 2SECTL)  CJP    $
  NOOP    PF03 INPT NOJMPI RDGA    CONT    $
  MOVE DRSO R5    DLE IEN    BF03 BT16 INPT NOJMPI RDGA
     IRDY    ADDRESS @ CJP    $
DECIMAL
```

```
  144 LIST

(     SECTOR PASS 1, CHT, 11-JUN-83)
HEX
  EXOR DRRTO R2    SRE Z    PF03 INPT NOJMPI RDGA
     CT16    33 ( 2SECTL) CJP    $
  MOVE DRSO R5    DLE IEN    PF03 BT16 INPT NOJMPI RDGA
     CONT    $
  EXOR DRRTO R4    SRE Z    INPT NOJMPI RDGA
     CT16    35 ( 1MATCH) CJP    $
LABEL 2SECTL
  MOVE RYSO R0    NOJMPI    1SECTL RPCT    $
  NRLD2 N0 R0    IEN    NOJMPI    RTN    $
DECIMAL
```

```
  145 LIST

(     SUCCESSFUL MATCH, CHT, 11-JUN-83)
HEX
LABEL 1MATCH
  MOVE RYSO R0    OEY    JMPI    CONT    $
LABEL BRTABL
  NEG RASO R3    IEN    NOJMPI RFIF    38 ( 1RDSEC) JP    $
  NEG RASO R3    IEN    NOJMPI RFIF    ED ( 1WRSEC) JP    $
DECIMAL
```

```
  146 LIST
```

0	1100000111100000	10011000110000	1101001001111111	1111111111111100	1111111111111000
1	1111100011100000	1011101111	1111110000011111	1111111111111110	1111111111111000
2	0	1011101111	1111110000011111	1111111101111110	1111111111111000
3	1111100011100000	1011101111	1111110000011111	1111111111111110	1111111111111000
4	10000	1011000001	1111110000011111	1111111101111110	1111111111111000
5	1101100001000011	1111101111	1111110000011111	1111111111111110	1111111111111000
6	1101100011000100	10011101111	1111110000011111	1111111111111110	1111111111111000
7	1101100001000100	1011101111	1111110000011111	1111111111111110	1111111111111000
8	1110001010000000	11011101111	1111110000011111	1111110111111110	1111111111111000
9	1110001010000000	11011101111	1111110000011111	1111111111101110	1111111111111000
10	1110001010000000	11011101111	1111110000011111	1111111111111110	1111111111111000
11	1110001010000000	11011101111	1111110000011111	1111111011111110	1111111111111000
12	1101110101100100	10010010000	1110000011111	1111111111101110	1111111111111000
13	1111100011100000	1011101111	1111110000011111	1111111111111110	1111111111111000
14	10001	1011000000	1000000011111	1111111101111110	1111111111111000
15	1101100001000100	1011101111	1111110000011111	1111111111101110	1111111111111000
16	1110001010000000	11011101111	1111110000011111	1111111110111110	1111111111111000
17	1110001010000000	11011101111	1111110000011111	1111111111111110	1111111111111000
18	1110001010000000	11011101111	1111110000011111	1111111111101110	1111111111111000
19	1110001010000000	11011101111	1111110000011111	1111111111111110	1111111111111000
20	1101110101100100	10010010000	1111000011111	1001111111111110	1101111111111000
21	1101100100000101	10011101111	1111110000011111	1111111111101110	1111111111111000
22	1110001010000000	11011101111	1111110000011111	1111111110111110	1111111111111000
23	1101100011000101	10011101111	1111110000011111	1011101111111110	1111111111111000
24	1101100001000100	1011101111	1111110000011111	1111111111111110	1111111111111000
25	1101100001000101	1011101111	1111110000011111	1111111111111110	1101111111111000
26	1111100011100001	10011101111	1111110000011111	1111111111111110	1111111111111000
27	110110	10011101111	1111110000011111	1111111111111110	1111111111111000
28	1000000010000000	10011101111	1111110000011111	1111111111111110	1111111111111000
29	1111100011100001	10011101111	1111110000011111	1111111111111110	1111111111111000
30	1000000000000101	10011101111	1111110000011111	1111111111111110	1111111111111000
31	1101100100000100	10011101111	1111110000011111	1111111111111110	1111111111111000
32	1101100001000001	10011000000	1000000000011111	1111111111111110	1111111111111000
33	1101100011000101	10011101111	1111110000011111	1111111111111110	1111111111111000
34	1000000101100101	10011101111	1111110000011111	1111111111111110	1111111111111000
35	1100110001100100	10010010000	1000100000011111	1111111111111110	1111111111111000
36	1101100001000010	10011000000	1000000000011111	1111111111111110	1111111111111000
37	1101100011000101	10011101111	1111110000011111	1111111111111110	1111111111111000
38	1000000101100101	10011101111	1111110000011111	1111111111111110	1111111111111000
39	1100110001100100	10010010000	1001100000011111	1111111111111110	1111111111111000
40	1110001010000000	11011000001	11111	1111111111111010	1111111110111000

6. Partial List of Microcodes of Disk Controller