```
10 PRINT "     DISASSEMBLER FOR THE 8X300 INTERPRETER"
20 PRINT : PRINT "THIS PROGRAM DISASSEMBLES 8X300 CODE THAT RESIDES"
30 PRINT "BETWEEN ADRESSES 0 AND 511 ON THE 8X300 SYSTEM. THE"
40 PRINT "LOW BYTE FIELD OF THE CODE MUST BE FROM F000 HEX TO"
50 PRINT "F1FF HEX IN THE MEMORY OF THIS 8080 COMPUTER, AND"
60 PRINT "THE HIGH BYTE FIELD MUST BE FROM F200 HEX TO F3FF"
70 PRINT "HEX."
80 INPUT "ENTER THE BEGINNING 8X300 ADDRESS: ",B
90 INPUT "ENTER THE ENDING 8X300 ADRESS: ",E
100 LET A1=61439+B: REM.F000H+B-1
110 LET A2=A1+512
120 FOR A0=B TO E: REM.MAIN LOOP
130   LET A1=A1+1: LET A2=A2+1
140   LET D=A0
150   IF D<256 THEN  PRINT "00"; ELSE  PRINT "01";: LET D=D-256
160   GOSUB 1170
170   PRINT H$;": ";
180   LET H=PEEK(A2): LET D=H: GOSUB 1170
190   PRINT H$;
200   LET L=PEEK(A1): LET D=L: GOSUB 1170
210   PRINT H$;" ";
220   LET O=INT(H/32)+1
230   LET S=H-(O-1)*32
240   ON O GOSUB 270,560,580,600,620,790,950,1110
250 NEXT A0
260 END
270 PRINT "MOVE ";
280 REM. PROCESS ARITHMETIC/ LOGICAL COMMAND
290 LET R=INT(L/32): LET D=L-R*32
300 IF S>15 THEN  GOTO 520
310 IF S=0 THEN  PRINT "AUX";: GOTO 360
320 IF S=8 THEN  PRINT "OVF";: GOTO 360
330 IF S=7 OR S=15 THEN  PRINT "?"; ELSE  PRINT "R";
340 IF S<8 THEN  PRINT "0"; ELSE  PRINT "1";: LET S=S-8
350 PRINT STR(S);
360 REM.
370 REM. FIND DESTINATION
380 IF D>15 THEN  GOTO 470
390 PRINT " RR";R;"-->";
400 IF D=7 THEN  PRINT "LB.IV.AL": RETURN
410 IF D=15 THEN  PRINT "RB.IV.AL": RETURN
420 IF D=0 THEN  PRINT "AUX": RETURN
430 IF D=8 THEN  PRINT "?"; ELSE  PRINT "R";
440 IF D<8 THEN  PRINT "0"; ELSE  PRINT "1";: LET D=D-8
450 PRINT STR(D)
460 RETURN
470 LET D=D-16
480 PRINT "-->";
490 IF D<8 THEN  PRINT "L"; ELSE  PRINT "R";: LET D=D-8
500 PRINT "B.IV SL";7-D;" M";R
510 RETURN
520 LET S=S-16
530 IF S<8 THEN  PRINT "L"; ELSE  PRINT "R";: LET S=S-8
540 PRINT "B.IV RR";7-S;" L";R;"-->";
550 IF D<16 THEN  GOTO 400 ELSE  LET D=D-16: GOTO 490
560 PRINT "ADD  AUX + ";
570 GOTO 280
580 PRINT "AND  AUX AND ";
590 GOTO 280
600 PRINT "XOR  AUX XOR ";
610 GOTO 280
620 PRINT "XEC  ";
630 IF S>16 THEN 710
640 IF S=8 THEN  PRINT "OVF";: GOTO 690
650 IF S=0 THEN  PRINT "AUX";: GOTO 690
```

```
650 IF S=0 THEN  PRINT "AUX";: GOTO 690
660 IF S=7 OR S=15 THEN  PRINT "?"; ELSE  PRINT "R";
670 IF S<8 THEN  PRINT "0"; ELSE  PRINT "1";: LET S=S-8
680 PRINT STR(S);
690 IF A0<256 THEN  PRINT " + 00";H$ ELSE  PRINT " + 01";H$
700 RETURN
710 IF A0>255 THEN  LET A3=A0-256 ELSE  LET A3=A0
720 LET A3=32*INT(A3/32)
730 LET R=INT(L/32): LET D=A3+L-R*32: GOSUB 1170
740 LET S=S-16
750 IF S<8 THEN  PRINT "L"; ELSE  PRINT "R";: LET S=S-8
760 PRINT "B.IV RR";7-S;" M";R;" + ";
770 IF A0<256 THEN  PRINT "00";H$ ELSE  PRINT "01";H$
780 RETURN
790 PRINT "NZT  ";
800 IF S>15 THEN 880
810 IF S=8 THEN  PRINT "OVF";: GOTO 860
820 IF S=0 THEN  PRINT "AUX";: GOTO 860
830 IF S=7 OR S=15 THEN  PRINT "?"; ELSE  PRINT "R";
840 IF S<8 THEN  PRINT "0"; ELSE  PRINT "1";: LET S=S-8
850 PRINT STR(S);
860 IF A0<256 THEN  PRINT ",00";H$ ELSE  PRINT ",01";H$
870 RETURN
880 IF A0<256 THEN  LET A3=A0 ELSE  LET A3=A0-256
890 LET S=S-16: LET R=INT(L/32): LET D=32*INT(A3/32)+L-R*32
900 GOSUB 1170
910 IF S<8 THEN  PRINT "L"; ELSE  PRINT "R";: LET S=S-8
920 PRINT "B.IV RR";7-S;" L";R;",";
930 IF A0<256 THEN  PRINT "00";H$ ELSE  PRINT "01";H$
940 RETURN
950 PRINT "XMIT ";
960 IF S>15 THEN 1050
970 PRINT "#";H$;"-->";
980 IF S=7 THEN  PRINT "LB.IV.AL": RETURN
990 IF S=15 THEN  PRINT "RB.IV.AL": RETURN
1000 IF S=0 THEN  PRINT "AUX": RETURN
1010 IF S=8 THEN  PRINT "?"; ELSE  PRINT "R";
1020 IF S<8 THEN  PRINT "0"; ELSE  PRINT "1";: LET S=S-8
1030 PRINT STR(S)
1040 RETURN
1050 LET S=S-16
1060 LET R=INT(L/32): LET D=L-R*32: GOSUB 1170
1070 PRINT "#";H$;"-->";
1080 IF S<8 THEN  PRINT "L"; ELSE  PRINT "R";: LET S=S-8
1090 PRINT "B.IV SL";7-S;" M";R
1100 RETURN
1110 PRINT "JMP  ";
1120 LET L$=H$
1130 LET D=S
1140 GOSUB 1170
1150 PRINT H$;L$
1160 RETURN
1170 REM.THIS SUBROUTINE CONVERTS A DECIMAL NUMBER IN D"
1180 REM.TO A HEX NUMBER IN HS. IT ALSO USES N.
1190 LET H$="": LET N=INT(D/16)
1200 GOSUB 1220
1210 LET N=D-N*16: REM.LOWER NIBBLE
1220 IF N<10 THEN  LET H$=H$+STR(N) ELSE  LET HS=HS+CHR(55+N)
1230 RETURN
RUN
     DISASSEMBLER FOR THE 8X300 INTERPRETER

 THIS PROGRAM DISASSEMBLES 8X300 CODE THAT RESIDES
BETWEEN ADRESSES 0 AND 511 ON THE 8X300 SYSTEM. THE
LOW BYTE FIELD OF THE CODE MUST BE FROM F000 HEX TO
F1FF HEX IN THE MEMORY OF THIS 8080 COMPUTER, AND
```

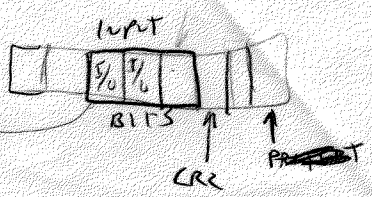
```
 1230 RETURN
RUN
     DISASSEMBLER FOR THE 8X300 INTERPRETER

 THIS PROGRAM DISASSEMBLES 8X300 CODE THAT RESIDES
BETWEEN ADRESSES 0 AND 511 ON THE 8X300 SYSTEM. THE
LOW BYTE FIELD OF THE CODE MUST BE FROM F000 HEX TO
F1FF HEX IN THE MEMORY OF THIS 8080 COMPUTER, AND
THE HIGH BYTE FIELD MUST BE FROM F200 HEX TO F3FF
HEX.
 ENTER THE BEGINNING 8X300 ADDRESS: 0
 ENTER THE ENDING 8X300 ADRESS: 511
 0000: C1FF XMIT #FF-->R01
 0001: C713 XMIT #13-->LB.IV.AL
 0002: 0117 MOVE R01-->LB.IV SL 0 M 0
 0003: C712 XMIT #12-->LB.IV.AL
 0004: 0117 MOVE R01-->LB.IV SL 0 M 0
 0005: C711 XMIT #11-->LB.IV.AL
 0006: 0117 MOVE R01-->LB.IV SL 0 M 0
 0007: D720 XMIT #00-->LB.IV SL 0 M 1
 0008: C712 XMIT #12-->LB.IV.AL
 0009: D420 XMIT #00-->LB.IV SL 3 M 1   } CYL. RESTORE
 000A: D520 XMIT #00-->LB.IV SL 2 M 1   }
 000B: D543 XMIT #03-->LB.IV SL 2 M 2   }
 000C: C701 XMIT #01-->LB.IV.AL
 000D: C015 XMIT #15-->AUX
 000E: 0017 MOVE AUX-->LB.IV SL 0 M 0
 000F: C704 XMIT #04-->LB.IV.AL
 0010: 0117 MOVE R01-->LB.IV SL 0 M 0
 0011: C715 XMIT #15-->LB.IV.AL
 0012: B792 NZT  LB.IV RR 0 L 4,0012  WAIT 'TILL HOME SEEKING
>0013: C704 XMIT #04-->LB.IV.AL       HAND SHAKING
 0014: D720 XMIT #00-->LB.IV SL 0 M 1   } <A1 Ready for New command
 0015: D721 XMIT #01-->LB.IV SL 0 M 1   }
 0016: C705 XMIT #05-->LB.IV.AL
 0017: B637 NZT  LB.IV RR 1 L 1,0017  WAI FOR LB2 TO GO LOW ; WAIT FOR LINK COMMAND TO BE WRITTEN
 0018: C703 XMIT #03-->LB.IV.AL
 0019: 1260 MOVE LB.IV RR 5 L 3-->AUX   } FETCH INSTRUCTION (COMMAND)
 001A: 17A1 MOVE LB.IV RR 0 L 5-->R01  Rest of INST.
 001B: C707 XMIT #07-->LB.IV.AL
 001C: 1702 MOVE LB.IV RR 0 L 0-->R02  Low half of command
 001D: C704 XMIT #04-->LB.IV.AL
 001E: D620 XMIT #00-->LB.IV SL 1 M 1 (A1 LO)      ACKNOWLEDGE COMMAND
 001F: D621 XMIT #01-->LB.IV SL 1 M 1 (A1 HI)
 0020: C702 XMIT #02-->LB.IV.AL  CLEAR A2
 0021: C600 XMIT #00-->R06
 0022: 0617 MOVE R06-->LB.IV SL 0 M 0   0->161
 0023: C901 XMIT #01-->R11
 0024: 8025 XEC  AUX + 0025
 0025: E02D JMP 0 002D  SEEK
 0026: E042 JMP 1 0042  WRITE SECTOR / READ SECTOR
 0027: E04D JMP 2 004D  WRITE BUFFER / READ BUFFER
 0028: E072 JMP 3 0072  READ STATUS
 0029: E07E JMP 4 007E  SET BYTE
 002A: E088 JMP 5 0088
 002B: E091 JMP 6 0091
 002C: E000 JMP 7 0000  RESET CONTROLLER (USE FOR HOME)
 002D: C0FF XMIT #FF-->AUX
 002E: C713 XMIT #13-->LB.IV.AL
 002F: 6217 XOR  AUX XOR R02-->LB.IV SL 0 M 0   } INVERT CYL. ADDRESS
 0030: C712 XMIT #12-->LB.IV.AL
 0031: 6137 XOR  AUX XOR R01-->LB.IV SL 0 M 1   } PUT CYL. ADDR. OUT
 0032: C200 XMIT #00-->R02
 0033: E1B0 JMP  01B0   )
 0034: C001 XMIT #01-->AUX  ARRIVE HERE IF READY
 0035: C716 XMIT #16-->LB.IV.AL
```

```
0034: C001 XMIT #01-->AUX  DRIVE HERE IF READY
0035: C716 XMIT #16-->LB.IV.AL
0036: 37A2 ADD  AUX + LB.IV RR 0 L 5-->R02   1+ SECTOR COUNT 6-41
0037: C018 XMIT #18-->AUX
0038: 6206 XOR  AUX XOR R06 RR 0-->AUX
0039: A03B NZT  AUX,003B
003A: C200 XMIT #00-->R02
003B: C712 XMIT #12-->LB.IV.AL
003C: D520 XMIT #00-->LB.IV SL 2 M 1   CYL STROBE
003D: E03E JMP  003E STUPID? DELAY?
003E: D521 XMIT #01-->LB.IV SL 2 M 1
003F: C715 XMIT #15-->LB.IV.AL
0040: B780 NZT  LB.IV RR 0 L 4,0040  WAIT 'TILL NOT BUSY SEEKING
0041: E0CA JMP  00CA
0042: C902 XMIT #02-->R11  Read/write status
0043: C701 XMIT #01-->LB.IV.AL
0044: 0151 MOVE R01-->LB.IV SL 6 M 2   BUFFER address
0045: E1B0 JMP  01B0
0046: 0106 MOVE R01 RR 0-->R06  WAIT # 06
0047: E0CA JMP  00CA
0048: C001 XMIT #01-->AUX
0049: 4641 AND  AUX AND R06 RR 2-->R01  Read or WRITE??
004A: 814B XEC  R01 + 004B
004B: E182 JMP  0182  write sect
004C: E11F JMP  011F  READ SECT
004D: C701 XMIT #01-->LB.IV.AL  Read/ WRITE BUFF
004E: 0151 MOVE R01-->LB.IV SL 6 M 2  BUFFER #
004F: C0FF XMIT #FF-->AUX
0050: 6202 XOR  AUX XOR R02 RR 0-->R02   TRANSFER length
0051: C001 XMIT #01-->AUX
0052: 2202 ADD  AUX + R02 RR 0-->R02
0053: 4181 AND  AUX AND R01 RR 4-->R01  Read/write? ~~WAIT TEST~~
0054: C300 XMIT #00-->R03
0055: A164 NZT  R01,0064
0056: C704 XMIT #04-->LB.IV.AL  WRITE
0057: D020 XMIT #00-->LB.IV SL 7 M 1  ready for   <20 instructions
0058: D021 XMIT #01-->LB.IV SL 7 M 1  WRITE DATA  < 5msec =
0059: 030F MOVE R03 RR 0-->RB.IV.AL  ADDRESS
005A: C705 XMIT #05-->LB.IV.AL
005B: B03B NZT  LB.IV RR 7 L 1,005B  DATA?
005C: C707 XMIT #07-->LB.IV.AL
005D: 171F MOVE LB.IV RR 0 L 0-->RB.IV SL 0 M 0  STASH IT
005E: 2303 ADD  AUX + R03 RR 0-->R03   BUMP
005F: 2202 ADD  AUX + R02 RR 0-->R02
0060: C705 XMIT #05-->LB.IV.AL
0061: 9021 XEC  ~~R18 + 0021~~  LB.IV.AL + 21 R01  WAIT FOR OVERWRITING DATA
0062: A259 NZT  R02,0059  DONE?
0063: E013 JMP  0013
0064: C704 XMIT #04-->LB.IV.AL  Read BUFFER
0065: D120 XMIT #00-->LB.IV SL 6 M 1   ready for Read data
0066: D121 XMIT #01-->LB.IV SL 6 M 1
0067: 030F MOVE R03 RR 0-->RB.IV.AL
0068: C706 XMIT #06-->LB.IV.AL  read data part   4 instr. times = 1msec.
0069: 1F17 MOVE RB.IV RR 0 L 0-->LB.IV SL 0 M 0
006A: 2303 ADD  AUX + R03 RR 0-->R03   BUMP POINTERS
006B: 2202 ADD  AUX + R02 RR 0-->R02
006C: C705 XMIT #05-->LB.IV.AL
006D: 912D XEC  LB.IV RR 6 M 1 + 006D  WAIT TILL'  WAIT TILL done reading
006E: B12E NZT  LB.IV RR 6 L 1,006E  WAIT TILL 0
006F: 912F XEC  LB.IV RR 6 M 1 + 006F  WAIT TILL 2
0070: A267 NZT  R02,0067  LOOP
0071: E013 JMP  0013
0072: C905 XMIT #05-->R11  Read status
0073: 0203 MOVE R02 RR 0-->R03
0074: C200 XMIT #00-->R02
0075: E1B0 JMP  01B0
```

```
0075: E1B0 JMP  01B0
0076: 0307 MOVE R03 RR 0-->LB.IV.AL
0077: 1702 MOVE LB.IV RR 0 L 0-->R02
0078: C706 XMIT #06-->LB.IV.AL
0079: 0217 MOVE R02-->LB.IV SL 0 M 0
007A: C704 XMIT #04-->LB.IV.AL
007B: D120 XMIT #00-->LB.IV SL 6 M 1
007C: D121 XMIT #01-->LB.IV SL 6 M 1
007D: E013 JMP  0013
007E: C704 XMIT #04-->LB.IV.AL
007F: D020 XMIT #00-->LB.IV SL 7 M 1
0080: D021 XMIT #01-->LB.IV SL 7 M 1
0081: C705 XMIT #05-->LB.IV.AL
0082: B022 NZT  LB.IV RR 7 L 1,0082
0083: C707 XMIT #07-->LB.IV.AL
0084: 1703 MOVE LB.IV RR 0 L 0-->R03
0085: 0207 MOVE R02 RR 0-->LB.IV.AL
0086: 0317 MOVE R03-->LB.IV SL 0 M 0
0087: E013 JMP  0013
0088: C904 XMIT #04-->R11  WRONG command #6
0089: E043 JMP  0043
008A: C418 XMIT #18-->R04
008B: C308 XMIT #08-->R03
008C: C001 XMIT #01-->AUX
008D: E1CC JMP  01CC
008E: E1EA JMP  01EA
008F: C150 XMIT #50-->R01
0090: E125 JMP  0125
0091: C903 XMIT #03-->R11
0092: E1B0 JMP  01B0
0093: C902 XMIT #02-->R11
0094: C0FF XMIT #FF-->AUX
0095: C713 XMIT #13-->LB.IV.AL
0096: 0017 MOVE AUX-->LB.IV SL 0 M 0   } seek track 0
0097: C712 XMIT #12-->LB.IV.AL
0098: 0017 MOVE AUX-->LB.IV SL 0 M 0   }
0099: D420 XMIT #00-->LB.IV SL 3 M 1   } cyl restore
009A: D520 XMIT #00-->LB.IV SL 2 M 1   }
009B: E09C JMP  009C
009C: D543 XMIT #03-->LB.IV SL 2 M 2   }
009D: C715 XMIT #15-->LB.IV.AL
009E: B79E NZT  LB.IV RR 0 L 4,009E  WAIT TILL DONE SEEKING
009F: C600 XMIT #00-->R06
00A0: 0602 MOVE R06 RR 0-->R02
00A1: 89A0 XEC  R11 + 00A0
00A2: E148 JMP  0148
00A3: E0CA JMP  00CA
00A4: C702 XMIT #02-->LB.IV.AL
00A5: C007 XMIT #07-->AUX
00A6: 5460 AND  AUX AND LB.IV RR 3 L 3-->AUX
00A7: A0C7 NZT  AUX,00C7  QUIT IF SO
00A8: C001 XMIT #01-->AUX
00A9: 2606 ADD  AUX + R06 RR 0-->R06  NEXT SECT
00AA: C018 XMIT #18-->AUX
00AB: 6600 XOR  AUX XOR R06 RR 0-->AUX
00AC: A0A0 NZT  AUX,00A0
00AD: C713 XMIT #13-->LB.IV.AL
00AE: C0FF XMIT #FF-->AUX
00AF: 7702 XOR  AUX XOR LB.IV RR 0 L 0-->R02
00B0: C712 XMIT #12-->LB.IV.AL
00B1: B736 NZT  LB.IV RR 0 L 1,00B6
00B2: C095 XMIT #95-->AUX
00B3: 6200 XOR  AUX XOR R02 RR 0-->AUX
00B4: A0B6 NZT  AUX,00B6
00B5: E0C4 JMP  00C4
00B6: C001 XMIT #01-->AUX
```

CRC OR CYL. ADDR. ERROR on seek?

IN FROM THIS PORT



NEXT TRACK

COULD BE FORMAT...

```
00B6: C001 XMIT #01-->AUX
00B7: 2202 ADD  AUX + R02 RR 0-->R02
00B8: A8BA NZT  OVF,00BA
00B9: E0BC JMP  00BC
00BA: C712 XMIT #12-->LB.IV.AL
00BB: D720 XMIT #00-->LB.IV SL 0 M 1
00BC: C713 XMIT #13-->LB.IV.AL
00BD: C0FF XMIT #FF-->AUX
00BE: 6217 XOR  AUX XOR R02-->LB.IV SL 0 M 0
00BF: C712 XMIT #12-->LB.IV.AL
00C0: D520 XMIT #00-->LB.IV SL 2 M 1
00C1: E0C2 JMP  00C2
00C2: D521 XMIT #01-->LB.IV SL 2 M 1
00C3: E09D JMP  009D
00C4: C003 XMIT #03-->AUX
00C5: 6900 XOR  AUX XOR R11 RR 0-->AUX
00C6: A0C8 NZT  AUX,00C8
00C7: E013 JMP  0013
00C8: C903 XMIT #03-->R11
00C9: E094 JMP  0094
→00CA: 0905 MOVE R11 RR 0-->R05
00CB: C430 XMIT #30-->R04
00CC: C300 XMIT #00-->R03
00CD: C903 XMIT #03-->R11
00CE: C001 XMIT #01-->AUX
00CF: E1CC JMP  01CC
00D0: E1EA JMP  01EA
00D1: C150 XMIT #50-->R01
00D2: C0FF XMIT #FF-->AUX
00D3: 2101 ADD  AUX + R01 RR 0-->R01
00D4: A1D3 NZT  R01,00D3
00D5: C725 XMIT #25-->LB.IV.AL
00D6: D121 XMIT #01-->LB.IV SL 6 M 1
00D7: C724 XMIT #24-->LB.IV.AL
00D8: 9738 XEC  LB.IV RR 0 M 1 + 00D8
00D9: C725 XMIT #25-->LB.IV.AL
00DA: D621 XMIT #01-->LB.IV SL 1 M 1
00DB: D620 XMIT #00-->LB.IV SL 1 M 1
00DC: C724 XMIT #24-->LB.IV.AL
00DD: 973D XEC  LB.IV RR 0 M 1 + 00DD
00DE: C721 XMIT #21-->LB.IV.AL
00DF: 1721 MOVE LB.IV RR 0 L 1-->R01
00E0: C001 XMIT #01-->AUX
00E1: C712 XMIT #12-->LB.IV.AL
00E2: 7720 XOR  AUX XOR LB.IV RR 0 L 1-->AUX
00E3: 6100 XOR  AUX XOR R01 RR 0-->AUX
00E4: C702 XMIT #02-->LB.IV.AL
00E5: E0E6 JMP  00E6
00E6: 0032 MOVE AUX-->LB.IV SL 5 M 1
00E7: C725 XMIT #25-->LB.IV.AL
00E8: D621 XMIT #01-->LB.IV SL 1 M 1
00E9: D620 XMIT #00-->LB.IV SL 1 M 1
00EA: C724 XMIT #24-->LB.IV.AL
00EB: 972B XEC  LB.IV RR 0 M 1 + 00EB
00EC: C721 XMIT #21-->LB.IV.AL
00ED: 1701 MOVE LB.IV RR 0 L 0-->R01
00EE: C0FF XMIT #FF-->AUX
00EF: C713 XMIT #13-->LB.IV.AL
00F0: 7700 XOR  AUX XOR LB.IV RR 0 L 0-->AUX
00F1: 6100 XOR  AUX XOR R01 RR 0-->AUX
00F2: A0F4 NZT  AUX,00F4
00F3: E0F6 JMP  00F6
00F4: C702 XMIT #02-->LB.IV.AL
00F5: D221 XMIT #01-->LB.IV SL 5 M 1
00F6: C725 XMIT #25-->LB.IV.AL
00F7: D621 XMIT #01-->LB.IV SL 1 M 1
```

FORMAT...

TIMER

START TRANSFER (STOP AT LINE 116)

WAIT FOR X-fer REQ

TRANSFER ACK

WAIT FOR X-fer REQ

fifo DATA

CHECK CYL ADDR AGAINST DISK DATA

SO: ~~FIRST~~ LAST STUFF ON EACH SECTOR IS DISK ADDRESS. IF THIS DOES NOT MATCH WHAT DRIVE SAYS ADDRESS IS, THEN ERROR

GET MSB OF CYL. ADDR

WAIT FOR TRANSFER REQUEST

fifo DATA

CYL. ADDR

COMPLEMENT IT

CYL. ADDR = FIFO DATA?

SET BIT 5 HIGH IF FIFO DATA ≠ FIFO DATA

JUST PUT 1 OUT

```
00F7: D621 XMIT #01-->LB.IV SL 1 M 1   TRANSFER ACK
00F8: D620 XMIT #00-->LB.IV SL 1 M 1   STROBE 1+
00F9: C724 XMIT #24-->LB.IV.AL
00FA: 973A XEC  LB.IV RR 0 M 1 + 00FA  WAIT FOR TRANSFER REQUEST
00FB: C01F XMIT #1F-->AUX
00FC: 4202 AND  AUX AND R02 RR 0-->R02
00FD: C721 XMIT #21-->LB.IV.AL  FIFO DATA
00FE: 5700 AND  AUX AND LB.IV RR 0 L 0-->AUX
00FF: 6200 XOR  AUX XOR R02 RR 0-->AUX
0100: A002 NZT  AUX,0102                    ARE TOP 3 BITS OF R02 SAME AS FIFO DATA?
0101: E104 JMP  0104
0102: C702 XMIT #02-->LB.IV.AL
0103: D321 XMIT #01-->LB.IV SL 4 M 1
0104: C007 XMIT #07-->AUX
0105: C711 XMIT #11-->LB.IV.AL
0106: 5361 AND  AUX AND LB.IV RR 4 L 3-->R01  GET HEADER
0107: C721 XMIT #21-->LB.IV.AL                FIFO DATA
0108: 5260 AND  AUX AND LB.IV RR 5 L 3-->AUX
0109: 6101 XOR  AUX XOR R01 RR 0-->R01        FIFO DATA = HEADER?
010A: A10C NZT  R01,010C
010B: E10B JMP  010B
010C: C702 XMIT #02-->LB.IV.AL  NOPE.
010D: D121 XMIT #01-->LB.IV SL 6 M 1
010E: C724 XMIT #24-->LB.IV.AL
010F: 962F XEC  LB.IV RR 1 M 1 + 010F  WAIT FOR TRANSFER REQ.
0110: C114 XMIT #14-->R01
0111: C0FF XMIT #FF-->AUX
0112: 2101 ADD  AUX + R01 RR 0-->R01    TIMER
0113: A112 NZT  R01,0112
0114: C725 XMIT #25-->LB.IV.AL
0115: D120 XMIT #00-->LB.IV SL 6 M 1    STOP TRANSFER
0116: C724 XMIT #24-->LB.IV.AL
0117: 1522 MOVE LB.IV RR 2 L 1-->R02    CHECK CRC
0118: C702 XMIT #02-->LB.IV.AL
0119: 0234 MOVE R02-->LB.IV SL 3 M 1
011A: 0509 MOVE R05 RR 0-->R11
011B: 891B XEC  R11 + 011B
011C: E013 JMP  0013
011D: E048 JMP  0048  reads
011E: E0A4 JMP  00A4  SCAN ALL DRIVES...
011F: C418 XMIT #18-->R04  Read sector
0120: C308 XMIT #08-->R03
0121: C901 XMIT #01-->R11
0122: E1CC JMP  01CC
0123: D221 XMIT #01-->LB.IV SL 5 M 1
0124: C114 XMIT #14-->R01
0125: C0FF XMIT #FF-->AUX
0126: 2101 ADD  AUX + R01 RR 0-->R01    TIMER
0127: A126 NZT  R01,0126
0128: C725 XMIT #25-->LB.IV.AL
0129: D121 XMIT #01-->LB.IV SL 6 M 1    set Read mode
012A: C724 XMIT #24-->LB.IV.AL
012B: 972B XEC  LB.IV RR 0 M 1 + 012B
012C: C725 XMIT #25-->LB.IV.AL
012D: D621 XMIT #01-->LB.IV SL 1 M 1    TRANSFER ACK
012E: D620 XMIT #00-->LB.IV SL 1 M 1
012F: C001 XMIT #01-->AUX
0130: C724 XMIT #24-->LB.IV.AL
0131: 9751 XEC  LB.IV RR 0 M 2 + 0131   LOOP
0132: E135 JMP  0135
0133: E13D JMP  013D
0134: E135 JMP  0135
0135: 010F MOVE R01 RR 0-->RB.IV.AL DATA
0136: C721 XMIT #21-->LB.IV.AL
0137: 171F MOVE LB.IV RR 0 L 0-->RB.IV SL 0 M 0   READ BUFFER NOW
0138: 2101 ADD  AUX + R01 RR 0-->R01
```

```
0138: 2101 ADD  AUX + R01 RR 0-->R01
0139: C725 XMIT #25-->LB.IV.AL
013A: D621 XMIT #01-->LB.IV SL 1 M 1
013B: D620 XMIT #00-->LB.IV SL 1 M 1
013C: A130 NZT  R01,0130
013D: C114 XMIT #14-->R01
013E: C0FF XMIT #FF-->AUX
013F: 2101 ADD  AUX + R01 RR 0-->R01   delay
0140: A13F NZT  R01,013F
0141: C725 XMIT #25-->LB.IV.AL          } STOP DISK TRANSFER
0142: D120 XMIT #00-->LB.IV SL 6 M 1
0143: C724 XMIT #24-->LB.IV.AL          } CRC CHECK TO BIT2 PORT 02
0144: 1522 MOVE LB.IV RR 2 L 1-->R02
0145: C702 XMIT #02-->LB.IV.AL
0146: 0235 MOVE R02-->LB.IV SL 2 M 1
0147: E013 JMP  0013<-
0148: C420 XMIT #20-->R04
0149: C300 XMIT #00-->R03
014A: C905 XMIT #05-->R11
014B: C000 XMIT #00-->AUX
014C: E1CC JMP  01CC
014D: E1EA JMP  01EA
014E: C712 XMIT #12-->LB.IV.AL
014F: D320 XMIT #00-->LB.IV SL 4 M 1    ENABLE WRITE
0150: C0FF XMIT #FF-->AUX
0151: C722 XMIT #22-->LB.IV.AL
0152: 0017 MOVE AUX-->LB.IV SL 0 M 0    WRITE TO FIFO
0153: C725 XMIT #25-->LB.IV.AL
0154: D621 XMIT #01-->LB.IV SL 1 M 1
0155: D620 XMIT #00-->LB.IV SL 1 M 1
0156: C4A0 XMIT #A0-->R04
0157: 2404 ADD  AUX + R04 RR 0-->R04    } TIMER
0158: A457 NZT  R04,0157
0159: C725 XMIT #25-->LB.IV.AL
015A: D121 XMIT #01-->LB.IV SL 6 M 1
015B: C001 XMIT #01-->AUX
015C: C712 XMIT #12-->LB.IV.AL
015D: 5701 AND  AUX AND LB.IV RR 0 L 0-->R01   TRACK MSB
015E: 6101 XOR  AUX XOR R01 RR 0-->R01
015F: C722 XMIT #22-->LB.IV.AL
0160: 0117 MOVE R01-->LB.IV SL 0 M 0    WRITE TO FIFO
0161: C725 XMIT #25-->LB.IV.AL
0162: D621 XMIT #01-->LB.IV SL 1 M 1
0163: D620 XMIT #00-->LB.IV SL 1 M 1
0164: C0FF XMIT #FF-->AUX
0165: C713 XMIT #13-->LB.IV.AL          TRACK LSB
0166: 7701 XOR  AUX XOR LB.IV RR 0 L 0-->R01
0167: C722 XMIT #22-->LB.IV.AL
0168: 0117 MOVE R01-->LB.IV SL 0 M 0
0169: C725 XMIT #25-->LB.IV.AL          } SEND SYNC
016A: D621 XMIT #01-->LB.IV SL 1 M 1
016B: D620 XMIT #00-->LB.IV SL 1 M 1
016C: C007 XMIT #07-->AUX
016D: C711 XMIT #11-->LB.IV.AL
016E: 5360 AND  AUX AND LB.IV RR 4 L 3-->AUX   <-SECT HEAD#
016F: 0060 MOVE AUX RR 3-->AUX
0170: 2601 ADD  AUX + R06 RR 0-->R01
0171: C722 XMIT #22-->LB.IV.AL          PUT THAT
0172: 0117 MOVE R01-->LB.IV SL 0 M 0
0173: C725 XMIT #25-->LB.IV.AL
0174: D621 XMIT #01-->LB.IV SL 1 M 1
0175: D620 XMIT #00-->LB.IV SL 1 M 1
0176: C724 XMIT #24-->LB.IV.AL
0177: 9637 XEC  LB.IV RR 1 M 1 + 0177   WAIT TILL DONE
0178: C0FF XMIT #FF-->AUX
0179: C450 XMIT #50-->R04               } TIMER
```

```
0179: C450 XMIT #50-->R04                      } timer
017A: 2404 ADD  AUX + R04 RR 0-->R04           }
017B: A47A NZT  R04,017A
017C: C712 XMIT #12-->LB.IV.AL
017D: D321 XMIT #01-->LB.IV SL 4 M 1     ; disable write
017E: C725 XMIT #25-->LB.IV.AL
017F: D120 XMIT #00-->LB.IV SL 6 M 1
0180: C902 XMIT #02-->R11
0181: E0A8 JMP  00A8
0182: C702 XMIT #02-->LB.IV.AL
0183: C01B XMIT #1B-->AUX
0184: 5480 AND  AUX AND LB.IV RR 3 L 4-->AUX
0185: A08B NZT  AUX,018B                 ; no write if disk error
0186: C408 XMIT #08-->R04
0187: C308 XMIT #08-->R03
0188: C902 XMIT #02-->R11
0189: C000 XMIT #00-->AUX
018A: E1CC JMP  01CC
018B: E013 JMP  0013                     ; Don't write if error
018C: C712 XMIT #12-->LB.IV.AL
018D: D320 XMIT #00-->LB.IV SL 4 M 1
018E: C1FF XMIT #FF-->R01
018F: C722 XMIT #22-->LB.IV.AL
0190: 0117 MOVE R01-->LB.IV SL 0 M 0
0191: C725 XMIT #25-->LB.IV.AL
0192: D621 XMIT #01-->LB.IV SL 1 M 1
0193: D620 XMIT #00-->LB.IV SL 1 M 1
0194: C0FF XMIT #FF-->AUX
0195: C421 XMIT #21-->R04
0196: 2404 ADD  AUX + R04 RR 0-->R04
0197: A496 NZT  R04,0196
0198: C001 XMIT #01-->AUX
0199: C725 XMIT #25-->LB.IV.AL
019A: D121 XMIT #01-->LB.IV SL 6 M 1
019B: 2101 ADD  AUX + R01 RR 0-->R01
019C: 010F MOVE R01 RR 0-->RB.IV.AL
019D: C722 XMIT #22-->LB.IV.AL
019E: 1F17 MOVE RB.IV RR 0 L 0-->LB.IV SL 0 M 0   ; write to sector
019F: C725 XMIT #25-->LB.IV.AL
01A0: D621 XMIT #01-->LB.IV SL 1 M 1
01A1: D620 XMIT #00-->LB.IV SL 1 M 1
01A2: C724 XMIT #24-->LB.IV.AL
01A3: 9743 XEC  LB.IV RR 0 M 2 + 01A3
01A4: E19B JMP  019B
01A5: E1A7 JMP  01A7
01A6: E1A7 JMP  01A7
01A7: C0FF XMIT #FF-->AUX
01A8: C428 XMIT #28-->R04                      } timer
01A9: 2404 ADD  AUX + R04 RR 0-->R04           }
01AA: A4A9 NZT  R04,01A9
01AB: C712 XMIT #12-->LB.IV.AL
01AC: D321 XMIT #01-->LB.IV SL 4 M 1     ; disable write
01AD: C725 XMIT #25-->LB.IV.AL
01AE: D120 XMIT #00-->LB.IV SL 6 M 1     ; end transfer
01AF: E013 JMP  0013                     ; Done
01B0: 0141 MOVE R01 RR 2-->R01           ; unit #
01B1: C003 XMIT #03-->AUX
01B2: 4100 AND  AUX AND R01 RR 0-->AUX   ; unit #
01B3: C711 XMIT #11-->LB.IV.AL           ; select drive
01B4: 80C5 XEC  AUX + 01C5               ; Select
01B5: C004 XMIT #04-->AUX                ; Head select 0
01B6: 62A0 XOR  AUX XOR R02 RR 5-->AUX
01B7: 0073 MOVE AUX-->LB.IV SL 4 M 3     ; Get file protected status
01B8: C715 XMIT #15-->LB.IV.AL
01B9: 1220 MOVE LB.IV RR 5 L 1-->AUX     ; ... Port 161 / FILE PROTECT STATUS IS WRITTEN TO PORT 02 ... SEEK. (top bit)
01BA: C702 XMIT #02-->LB.IV.AL
```

```
01B7: 0073 MOVE AUX-->LB.IV SL 4 M 3
01B8: C715 XMIT #15-->LB.IV.AL
01B9: 1220 MOVE LB.IV RR 5 L 1-->AUX
01BA: C702 XMIT #02-->LB.IV.AL
01BB: 0030 MOVE AUX-->LB.IV SL 7 M 1
01BC: C715 XMIT #15-->LB.IV.AL
01BD: B03F NZT  LB.IV RR 7 L 1,01BF
01BE: E1C9 JMP  01C9
01BF: 89BF XEC  R11 + 01BF
01C0: E034 JMP  0034
01C1: E046 JMP  0046
01C2: E093 JMP  0093
01C3: E08A JMP  008A
01C4: E076 JMP  0076
01C5: D78E XMIT #0E-->LB.IV SL 0 M 4
01C6: D78D XMIT #0D-->LB.IV SL 0 M 4
01C7: D78B XMIT #0B-->LB.IV SL 0 M 4
01C8: D787 XMIT #07-->LB.IV SL 0 M 4
01C9: C702 XMIT #02-->LB.IV.AL
01CA: D721 XMIT #01-->LB.IV SL 0 M 1
01CB: E013 JMP  0013
01CC: C725 XMIT #25-->LB.IV.AL
01CD: 0032 MOVE AUX-->LB.IV SL 5 M 1
01CE: D142 XMIT #02-->LB.IV SL 6 M 2
01CF: D780 XMIT #00-->LB.IV SL 0 M 4
01D0: D421 XMIT #01-->LB.IV SL 3 M 1
01D1: C723 XMIT #23-->LB.IV.AL
01D2: D38F XMIT #0F-->LB.IV SL 4 M 4
01D3: 0497 MOVE R04-->LB.IV SL 0 M 4
01D4: D320 XMIT #00-->LB.IV SL 4 M 1
01D5: D321 XMIT #01-->LB.IV SL 4 M 1
01D6: 0484 MOVE R04 RR 4-->R04
01D7: 0497 MOVE R04-->LB.IV SL 0 M 4
01D8: D220 XMIT #00-->LB.IV SL 5 M 1
01D9: D221 XMIT #01-->LB.IV SL 5 M 1
01DA: 0397 MOVE R03-->LB.IV SL 0 M 4
01DB: D120 XMIT #00-->LB.IV SL 6 M 1
01DC: D121 XMIT #01-->LB.IV SL 6 M 1
01DD: 0383 MOVE R03 RR 4-->R03
01DE: 0397 MOVE R03-->LB.IV SL 0 M 4
01DF: D020 XMIT #00-->LB.IV SL 7 M 1
01E0: D021 XMIT #01-->LB.IV SL 7 M 1
01E1: C725 XMIT #25-->LB.IV.AL
01E2: D420 XMIT #00-->LB.IV SL 3 M 1
01E3: D421 XMIT #01-->LB.IV SL 3 M 1
01E4: 89E4 XEC  R11 + 01E4
01E5: E123 JMP  0123
01E6: E18C JMP  018C
01E7: E0D0 JMP  00D0
01E8: E08E JMP  008E
01E9: E14D JMP  014D
01EA: C318 XMIT #18-->R03
01EB: C716 XMIT #16-->LB.IV.AL
01EC: B02C NZT  LB.IV RR 7 L 1,01EC
01ED: 902D XEC  R18 + 012D
01EE: C01F XMIT #1F-->AUX
01EF: 4200 AND  AUX AND R02 RR 0-->AUX
01F0: 77A0 XOR  AUX XOR LB.IV RR 0 L 5-->AUX
01F1: A0F6 NZT  AUX,01F6
01F2: 89F0 XEC  R11 + 01F0
01F3: E0D1 JMP  00D1
01F4: E08F JMP  008F
01F5: E14E JMP  014E
01F6: C0FF XMIT #FF-->AUX
01F7: 2303 ADD  AUX + R03 RR 0-->R03
01F8: A3EC NZT  R03,01EC
```

29 instr's

Get File Status

To Port 161

File protect status is written to port 02 upon seek. (top bit)

drive select commands

Read mode

Disk trans. start

Clear

Stop clearing

Load pulses

Read servo shift

Wait for Sector Pulse to go low

LB.IV RR0 L1 + LED wait for sector pulse to go HI

Count tries to find right sector

01C9 Not ready

R11 has code telling where it came from

From status check

From Read Sector

110

Command echo 110-

01BF Ready;

01C9: C702 XMIT #02-->LB.IV.AL — SET BIT 0 PORT 02 HIGH IF DISK NOT READY

01E7: E0D0 JMP 00D0 Read Sect

```
01D8: D220 XMIT #00-->LB.IV SL 5 M 1
01D9: D221 XMIT #01-->LB.IV SL 5 M 1
01DA: 0397 MOVE R03-->LB.IV SL 0 M 4
01DB: D120 XMIT #00-->LB.IV SL 6 M 1
01DC: D121 XMIT #01-->LB.IV SL 6 M 1
01DD: 0383 MOVE R03 RR 4-->R03
01DE: 0397 MOVE R03-->LB.IV SL 0 M 4
01DF: D020 XMIT #00-->LB.IV SL 7 M 1
01E0: D021 XMIT #01-->LB.IV SL 7 M 1
01E1: C725 XMIT #25-->LB.IV.AL
01E2: D420 XMIT #00-->LB.IV SL 3 M 1
01E3: D421 XMIT #01-->LB.IV SL 3 M 1
01E4: 89E4 XEC  R11 + 01E4
01E5: E123 JMP  0123
01E6: E18C JMP  018C
01E7: E0D0 JMP  00D0 Read Sect
01E8: E08E JMP  008E 110
01E9: E14D JMP  014D 111
01EA: C318 XMIT #18-->R03
01EB: C716 XMIT #16-->LB.IV.AL
01EC: B02C NZT  LB.IV RR 7 L 1,01EC
01ED: 902D XEC  R18 + 012D LB.IV RR0 L1 +1ED
01EE: C01F XMIT #1F-->AUX
01EF: 4200 AND  AUX AND R02 RR 0-->AUX
01F0: 77A0 XOR  AUX XOR LB.IV RR 0 L 5-->AUX
01F1: A0F6 NZT  AUX,01F6
01F2: 89F0 XEC  R11 + 01F0
01F3: E0D1 JMP  00D1
01F4: E08F JMP  008F 110
01F5: E14E JMP  014E 111
01F6: C0FF XMIT #FF-->AUX
01F7: 2303 ADD  AUX + R03 RR 0-->R03
01F8: A3EC NZT  R03,01EC
01F9: C702 XMIT #02-->LB.IV.AL
01FA: D821 XMIT #01-->LB.IV SL 1 M 1
01FB: E013 JMP  0013
01FC: 0000 MOVE AUX RR 0-->AUX
01FD: 0000 MOVE AUX RR 0-->AUX
01FE: 0000 MOVE AUX RR 0-->AUX
01FF: FF00 JMP  1F00
READY
```

wait for Sector Pulse to go low

wait for sector pulse to go #0#1

sector requested

Right sector?

COUNT TRIES TO FIND RIGHT SECTOR

CAN'T FIND SECTOR