```
                    (0001) *
                    (0002) *       AD-G, CA-G,U-CODE,MHJ,FEBRUARY 1975
                    (0003) *       PRIME 300 MICRO-CODE
                    (0004) *       PRIME COMPUTER INC.,SRC0768.001
                    (0005) *       COPYRIGHT 1974,PRIME COMPUTER INC.,NATICK MASS.
                    (0006) *
          000C01    (0007) P300   XSET    1
                    (0008) *      MA,SRCLIB,MHJ,FEBRUARY 1975
                    (0009) *      PRIME MICRO-CODE ASSEMBLER
                    (0010) *      PRIME COMPUTER INC.,0702.002
                    (0011) *      COPYRIGHT 1974,PRIME COMPUTER INC.,NATICK MASS.
                    (0012) *
                    (0013) *
                    (0014) *           ************************************
                    (0015) *           *                                  *
                    (0016) *           *            P R I M E             *
                    (0017) *           *                                  *
                    (0018) *           *       MICRO-CODE ASSEMBLER       *
                    (0019) *           *                                  *
                    (0020) *           *           SRC0702.002            *
                    (0021) *           *                                  *
                    (0022) *           ************************************
                    (0023) *
                    (0024) *
                    (0855)             IDNT    (CPU PROM SET AD ),(REV G ),(FEBRUARY 25,1975)
                    (0856) *
                    (0857) *                   FETCH CYCLE
                    (0858) *
                    (0859) *       BEGIN FETCH CYCLE, TEST FOR NEED TO INTERRUPT PROCESSING
                    (0860) *
                    (0861) F1      RR      RP => RY  CLEARFUII ;
                    (0862)                 JUMP ON FETCH1 TO FHALT
000: 0200 780F 0004 A030
                    (0863) *
                    (0864) *       INCREMENT P COUNTER, READ INSTRUCTION
                    (0865) *
                    (0866) F3      CPU     AL   NOP  ALL   INC    1     M     RP ;
                    (0867)                 CLMCLFCLI MREAD ,  ;
```

```
                    (0868)                  JUMP ON REL TO F9
001: 9304 7F84 0004 D004
                    (0869) *
                    (0870) *        SECTORED DECODE, BUILD EA WITH PRE-INDEXING IF NEEDED
                    (0871) *
                    (0872) F4       CPU     BB    RM    NONE  ADD     0       XM      0 ;
                    (0873)                  RY280       NOP   EAF ;
                    (0874)                  S FETCHDONE TO DP200 DECODE
002: EC61 0E0D 000D 3108
                    (0875) *
                    (0876) *        INDIRECTION REQUIRED
                    (0877) *
                    (0878) F5       CPU     0     NOP   ALL   0       0       0       0 ;
                    (0879)                  RMMRDY      MREAD NOP ;
                    (0880)                  GO TO F6
003: 1300 0784 0004 0010
                    (0881) *
                    (0882) *        RELATIVE DECODE, BUILD RELATIVE EA
                    (0883) *
                    (0884) F9       CPU     BB    RM    NONE  ADD     0       M       RP ;
                    (0885)                  RY280       NOP   EAF ;
                    (0886)                  S FETCHDONE TO DP200 RM07
004: EC60 7E0D 000D 3048
                    (0887) *
                    (0888) *        TEST FOR DOUBLE WORD OR STACK OPERATION
                    (0889) *
                    (0890) F10      RR      ,,    NOP ;
                    (0891)                  JUMP ON RMLTM240 TO F13
005: 0200 0004 0006 2008
                    (0892) *
                    (0893) *        PRE-INDEX FOR RELATIVE MODE
                    (0894) *
                    (0895) F11      CPU     BB    RY    ALL   ADD     0       XM      0 ;
                    (0896)                  RY240       NOP   EAF ;
                    (0897)                  S FETCHDONE TO POP F11
006: EB61 030D 000D 3104
                    (0898) *
                    (0899) *        INDIRECTION REQUIRED
```

```
                    (0900) *
                    (0901) F12     CPU      0     NOP   ALL   0       0      0      0 ;
                    (0902)                  RMMRDY      MREAD NOP ;
                    (0903)                  GO TO F6
007: 1300 0784 0004 0010
                    (0904) *
                    (0905) F13     RR       RP => RY   NOP TR= ALL ;
                    (0906)                  JUMP ON STACKOP TO F17
008: 0300 7804 0005 000D
                    (0907) *
                    (0908) *       SECOND OPERAND FETCH
                    (0909) *
                    (0910) F14     CPU      AL    NOP   ALL    INC     1      M      RP ;
                    (0911)                  RMRFMRDY   MREAD EAF ;
                    (0912)                  EAC   SETFUII
009: 9304 758D 000A 8000
                    (0913) *
                    (0914) *       DOUBLE STACK INDEX, IF REQUIRED
                    (0915) *
                    (0916) F16     CPU      BB    RM    ALL    ADD     0      M      RS ;
                    (0917)                  RY240       NOP    EAF ;
                    (0918)                  S RELDONE TO POP F16
00A: EF60 330D 000D 1024
                    (0919) *
                    (0920) *       DOUBLE INDEX ON RX, IF REQUIRED
                    (0921) *
                    (0922) F16A    CPU      BB    RY    ALL    ADD     0      XM     0 ;
                    (0923)                  RY240       NOP    EAF ;
                    (0924)                  S FETCHDONE TO POP F16A
00B: E861 030D 000D 3014
                    (0925) *
                    (0926) *       DOUBLE INDIRECT
                    (0927) *
                    (0928) F16B    CPU      0     NOP   ALL    0       0      0      0 ;
                    (0929)                  RMMRDY      MREAD NOP ;
                    (0930)                  GO TO F6
00C: 1300 0784 0004 0010
                    (0931) *
```

```
                    (0932) *       STACK OPERATION PROCESSING
                    (0933) *
                    (0934) F17     RR       RS => RY  NOP ;
                    (0935)                  JUMP ON PUSH TO F19
00D: 0200 3804 0005 2014
                    (0936) *
                    (0937) *       PRE-DECREMENT OR POP PROCESSING
                    (0938) *
                    (0939) F18     CPU      AL   NOP  ALL   INC 1  M      RS ;
                    (0940)                  RF200     NOP   EAF ;
                    (0941)                  S ON RELDONE, POP
00E: 9304 3A0D 000D 1004
                    (0942) F18A    CPU      ,,ALL,,,,,RMMRDY   MREAD
00F: 1300 0784 0000 0000
                    (0943) *
                    (0944) *       INDIRECTION AND POST INDEXING
                    (0945) *
                    (0946) F6      CPU      RF   RCM  NONE  0      0      M      RA ;
                    (0947)                  RF160     ,LOADRSC  DATA  -8
010: 1000 190E 0003 FEF8
                    (0948) F6A      CPU      BB   RM   NX    ADD    0      XM     0 ;
                    (0949)                  RY240     NOP   EAF ;
                    (0950)                  S INDIRECTEND TO POP INDIRECT
011: EE61 030D 000C E084
                    (0951) *
                    (0952) *       MULTIPLE LEVELS OF INDIRECTION
                    (0953) *
                    (0954) F7      CPU      0    NOP  ALL   0      0      0      0 ;
                    (0955)                  RMMRDY    MREAD INCRSC ;
                    (0956)                  JUMP ON RSCNEM1 TO F6A
012: 1300 0785 0004 9011
                    (0957) F7A     RR       ,,REST    GO TO F6A
013: 0200 0006 0004 0011
                    (0958) *
                    (0959) *       POST-INCREMENT STACK OP---PUSH
                    (0960) *
                    (0961) F19     CPU      AL   NOP  ALL   DEC    0      M      RS ;
                    (0962)                  RYRF240 NOP  EAF ;
```

```
                    (0963)                  S ON RELDONE TO POP
014: 93F0 3D0D 000D 1004
                    (0964) *
                    (0965) *        INDIRECT PUSH
                    (0966) *
                    (0967) F20      CPU     0    NOP  ALL   0      0      0      0 ;
                    (0968)                  RMMRDY    MREAD NOP ;
                    (0969)                  GO TO F6
015: 1300 0784 0004 0010
                    (0970)        EJCT
```

```
                    (0971) *
                    (0972) *      EXTERNAL INTERUPT
                    (0973) *
                    (0974) INT3   RR       ,,CLEARFUII JUMP ON FUII TO INT3
016: 0200 000F 0005 5016
                    (0975)        CPU      BB   BPA  NX    0      0      0      0 ;
                    (0976)                 RY280     NOP   NOP ;
                    (0977)                 JUMP ON BPSP1 TO MI1
017: F600 0E04 0004 308F
                    (0978)        ALU      CON ZERO => RM  TR= NONE ,JUMP ON FVIM TO NVECT
018: 903C 0204 0007 001A
                    (0979)        CPU      BB   RCM  NONE  0      0      RY     ICAI ;
                    (0980)                 RY200     ,,    DATA '63
019: F002 8804 0002 0133
                    (0981) NVECT  CPU      ,,ALL,,,RY,ICPN,RMMRDY  AREAD,, ;
                    (0982)                 MODAL  CLRNVKEYS  (TRIGNVKEYS,EINTM)
01A: 1302 47C4 000A 0410
                    (0983)        CPU      AL   RM   NONE  BB     L      0      0 ;
                    (0984)                 RY240 ,   SETCC  GO TO AVECT+2
01B: 8C5C 0307 0004 0079
                    (0985) *
                    (0986) *      MEMORY INCREMENT.  THE LOCATION SPECIFIED BY THE BPA IS
                    (0987) * READ AND THEN INCREMENTED.  AFTER THIS , IT IS WRITTEN
                    (0988) * BACK.  IF IT INCREMENTED TO 0, END OF BLOCK IS GENERATED
                    (0989) * ALONG WITH ICPN.  IF NOT, END OF BLOCK (BPCEOB) IS
                    (0990) * GENERATED AS ZERO.  BPCEOB = CARRY OUT OF BIT 1 OF THE ALU.
                    (0991) *
                    (0992) MI2    ALU      INC RM => RM    SETCC
01C: 8E65 1207 0000 0000
                    (0993)        RR       ,,C= AWRITE , JUMP ON NE TO *+2
01D: 0200 00F4 0004 401F
                    (0994)        CPU      0    RM   1     MINUS1 1     RY     ICPN ;
                    (0995)                 280  ,    ,     GO TO CP8
01E: 0DC6 4604 0004 0020
                    (0996)        CPU      0    RM   1     MINUS1 0     RY     ICPN ;
                    (0997)                 280  ,    ,     GO TO CP8
01F: 0DC2 4604 0004 0020
                    (0998)       EJCT
```

```
                    (0999) *       BEGIN PROCESSING THE FUNCTIONS
                    (1000) *
                    (1001) *
                    (1002)         ORG     $20
                    (1003) *               1    STOP/STEP  (OTHERS FOLLOW IN SEQUENCE)
                    (1004) *
                    (1005) CP8     RR      RP => RY  NOP  TR= ALL ;
                    (1006)                 GO TO F3
020: 0300 7804 0004 0001
                    (1007) CP9     RR      YSAVE => RY     NOP ;
                    (1008)                 GO TO CP1
021: 0200 C804 0004 0035
                    (1009) CP10    ALU     INC YSAVE => YSAVE     , ;
                    (1010)                 JUMP ON LT TO CP9
022: 9204 CA04 0006 6021
                    (1011) CP11    RR      YSAVE => RY     , GO TO CP19
023: 0200 C804 0004 002C
                    (1012) CP12    RR      RA => RM  , GO TO BOOT
024: 0200 1204 0004 0091
                    (1013) *
                    (1014) *       CLEAR THE ACTIVE REGISTER.  BIT 1 CLEARS THE DATA.
                    (1015) *
                    (1016) CP13    CPU     AL   NOP  NX    ZERO   L     RSC   0 ;
                    (1017)                 280  NOP  SETCC ;
                    (1018)                 JUMP ON LT TO CP15
025: 923F 0607 0006 6027
                    (1019) *
                    (1020) *       ADDRESS CHANGES THE RSC ADDRESSED REGISTER FROM
                    (1021) * MSAVE TO YSAVE.
                    (1022) *
                    (1023) CP14    CPU     BB   RCM  NX    0      0     RSC   0 ;
                    (1024)                 200  NOP  LOADRSC ;
                    (1025)                 DATA $C
026: F203 000E 0002 010C
                    (1026) *
                    (1027) *       THE SAVED REGISTER SELECTED IS OUTPUT TO THE LIGHTS,
                    (1028) * THEN THE DATA SWITCHES ARE ORED INTO IT.
                    (1029) *
```

```
                    (1030) CP15    CPU      RF    NOP  NX     ,,       RSC    DATA  RM200
027: 1203 2204 0000 0000
                    (1031) CP16    CPU      0     NOP  NX     0        0      RSC   (DATA,STROBE) ;
                    (1032)                  280
028: 1203 A604 0000 0000
                    (1033)         CPU      BB    RCM  NX     0        0      RSC   0 ;
                    (1034)                  RY200      ,,DATA *131720
029: F203 0804 0001 66D0
                    (1035) CP17    CPU      AL    BPD  NX     OR       L      RSC 0 RF280 , ;
                    (1036)                  ,JUMP ON NE TO CP3
02A: 9A4F 0C04 0004 4037
                    (1037) CP18    CPU      AL    NOP  NX     ZERO     L      RSC   0 ;
                    (1038)                  RF280      ,, GO TO CP3
02B: 923F 0C04 0004 0037
                    (1039) CP19    CPU      RF    NOP  NX     0        0      M     MSAVE ;
                    (1040)                  RM280      AWRITE          ,GO TO CP3
02C: 1200 D4F4 0004 0037
                    (1041)         EJCT
```

```
                    (1042) *
                    (1043) *       TEST FOR CONTROL PANEL REQUEST, IF NOT, DO EXTERNAL INT.
                    (1044) *
                    (1045) FHALT3 RR       ,, TR= ALL  ,JUMP ON CP TO CP1
02D: 0300 0004 0004 C035
                    (1046) *
                    (1047) *       EXTERNAL INTERRUPT PROCESSING.
                    (1048) * IENB IS VALID FOR THE THIRD STEP.
                    (1049) * IENB BEGINS 480 NS EARLIER AT THE START
                    (1050) * OF THE SECOND CYCLE.  BPSP1 IS TESTED, IF TRUE
                    (1051) * A MEMORY INCREMENT OPERATION IS PERFORMED.  IF FALSE, AN
                    (1052) * EXTERNAL INTERRUPT IS TAKEN.  THE VECTOR ADDRESS IS TAKEN
                    (1053) * FROM BPA IN VECTORED MODE, AND GENERATED AS '63
                    (1054) * IF IN COMPATIBLE MODE.
                    (1055) *
                    (1056) INTEX  CPU      0    NOP  NX    0      0     RY    IEN ;
                    (1057)                 200  NOP  NOP ;
                    (1058)                 EAC SETFUII
02E: 1202 2004 000A 8000
                    (1059)         CPU     0    NOP  NX    0      0     RY    IEN ;
                    (1060)                 280 NOP NOP GO TO INT3
02F: 1202 2604 0004 0016
                    (1061)       EJCT
```

```
                    (1062) *
                    (1063) *
                    (1064) *       HALTED FETCH CYCLE, FIND WHAT NEEDS PROCESSING
                    (1065) *
                    (1066) *
                    (1067) *       MASTER CLEAR INVERSE TEST
                    (1068) *
                    (1069) FHALT   RR       ,,, TR= ALL  JUMP ON SYSCLRNOT TO FHALT2
030: 0300 0004 0004 B074
                    (107C) *
                    (1071) *       MASTER CLEAR ROUTINE CLEARS CARRY, MODALS, AND REGISTERS
                    (1072) *       0 TO '37. '1000 IS PUT INTO THE P COUNTER, AND THE CONTROL
                    (1073) *       PANEL ROUTINE WHICH FOLLOWS READS '1000 INTO THE LIGHTS.
                    (1074) *
                    (1075) *
                    (1076) MC1     RR       RCM = '1000 => RY       C= BD01       LOADRSC
031: F200 083E 0000 0500
                    (1077) MC2     CPU      AL   NOP  NX    ZERO   L      RSC    0 ;
                    (1078)                  RF280     NOP   INCRSC ;
                    (1079)                  JUMP ON RSCNEM1 TO *
032: 923F 0C05 0004 9032
                    (1080) MC3     CPU      AL   NOP  NX    ZERO   L      0      0 ;
                    (1081)                  RM200     NOP   CLEARFUII ;
                    (1082)                  JUMP ON VERIFY TO VIRY1
033: 923C 020F 0007 30C4
                    (1083) MC4     CPU      BB   RY   NX    0      0      M      RP ;
                    (1084)                  RF280     NOP   NOP ;
                    (1085)                  EAC CLRNVKEYS (54,56,58,59,60,61,62,63)
034: EA00 7C04 000A 057E
                    (1086)         EJCT
```

```
                    (1087) *
                    (1088) *
                    (1089) *       CONTROL PANEL ROUTINE.  THIS PROGRAM FIRST READS THE
                    (1090) * FUNCTION SWITCHES (INA 1520).  THE INFORMATION IS
                    (1091) * ENCODED IN THE LOW ORDER FOUR BITS OF THE WORD AND THE
                    (1092) * SIGN BIT.  A SIXTEEN WAY BRANCH IS THEN DONE WITH THE
                    (1093) * FIRST 8 BRANCHES DEFINED AS FOLLOWS
                    (1094) *                1     RUN OR STOP STEP
                    (1095) *                2     FETCH THIS
                    (1096) *                3     FETCH OR STORE NEXT (BIT 1 IS SET FOR FETCH NEXT)
                    (1097) *                4     STORE THIS
                    (1098) *                5     AUTO LOAD
                    (1099) *                6     CLEAR ADDRESS OR DATA (BIT 1 IS SET FOR DATA)
                    (1100) *                7     ADDRESS (NO SPECIAL FUNCTION--LIGHTS =ADDRESS)
                    (1101) *                8     DATA (NO SPECIAL FUNCTION--LIGHTS = ADDRESS)
                    (1102) *
                    (1103) *
                    (1104) *       REGISTERS YSAVE AND MSAVE HOLD THE WORKING
                    (1105) * COPIES OF THE ADDRESS AND DATA LIGHTS, RESPECTIVELY.  THESE
                    (1106) * ARE UPDATED EACH TIME THROUGH THE LOOP AND DISPLAYED ON THE
                    (1107) * NEXT PASS.  INA 1720 IS USED TO READ THE DATA SWITCHES AND
                    (1108) * OR THEM INTO THE SAVED REGISTERS.  OTA 1720 IS THEN USED TO
                    (1109) * DUMP OUT THE OLD IMAGE.
                    (1110) *
                    (1111) *
                    (1112) *
                    (1113) CP1     CPU      BB    RY    NX     0       0      M      YSAVE ;
                    (1114)                  RMRFMRDY   AREAD NOP
035: EA00 C5C4 0000 0000
                    (1115) CP2     RR       RM => MSAVE     TR= NX
036: EE00 DA04 0000 0000
                    (1116) CP3     RR       RCM = '131520 => RY                    INA FUNCTION
037: F200 0804 0001 6750
                    (1117) CP4     CPU      BB    RCM   NX     0       0      RY     PIO ;
                    (1118)                  280   NOP   LOADRSC ;
                    (1119)                  DATA $D
038: F202 160E 0002 000D
                    (1120) CP5     CPU      BB    BPD   NX     0       0      RSC    STROBE ;
```

```
                    (1121)                  RM280                   FUNCTION SWITCH TO RM
039: FA03 8404 0000 0000
                    (1122) CP6      CPU     BB    RCM   NONE  0       0       RSC     0 ;
                    (1123)                  RY200       ,,DATA '171720
03A: F003 0804 0003 E6D0
                    (1124) CP7      CPU     AL    RM    NONE  BB      L       RSC     0 ;
                    (1125)                  RM280       NOP   SETCC ;
                    (1126)                  S ON TRUE 16WAYS TO CP8
03B: 8C5F 0407 000C 0022
                    (1127) *
                    (1128) *        REGISTER SAVE FOR DMX ROUTINE.
                    (1129) *
                    (1130) DMX2     RR      RY  =>  YSAVE
03C: E800 C904 0000 0000
                    (1131)          CPU     AL    RSC   NX      BB      L       M       RSCSAVE ;
                    (1132)                  RF200       ,,JUMP ON DMTNOTVALID TO DMA
03D: 865C EA04 0005 F04B
                    (1133)          EJCT
```

```
                    (1134) *
                    (1135) *       DMT.  HIGHEST SPEED OF ALL TRANSFERS, THE PERIPHERAL
                    (1136) * ADDRESS LINES ARE READ INTO RY AND DIRECTLY SELECT THE
                    (1137) * MEMORY LOCATION TO BE USED.  INPUT OR OUTPUT TRANSFER
                    (1138) * IS DETERMINED BY TESTING THE INPUT/OUTPUT LINE AS IS
                    (1139) * TRUE OF ALL DMX.  THE SIGNALS GENERATED ARE THE SAME
                    (1140) * AS FOR DMA EXCEPT THAT THE END OF RANGE SIGNAL IS UN=
                    (1141) * DEFINED FOR THIS TRANSFER TYPE.  THIS MEANS THE CONTROL-
                    (1142) * ER DETERMINES IF THE TRANSFER ENDS THE BLOCK.
                    (1143) *
                    (1144) *
                    (1145) DMT     CPU      BB   BPA  NX    0      0      RSC   (CPN,STROBE) ;
                    (1146)                  RY200     0     LOAD256K ;
                    (1147)                  JUMP ON OUTPUT TO TOUT1
03E: F603 C808 0006 5046
                    (1148)         CPU      ,,,,,RSC STROBE ,,,GO TO TIN1
03F: 1203 8004 0004 0042
                    (1149) *
                    (1150) *       CYCLE TO CYCLE FULLY OVERLAPPED DMT INPUT LOOP.
                    (1151) * NOTE THAT THE STROBE HAS ALREADY BEGUN BEFORE
                    (1152) * THE CYCLE CAN EXIT.  THIS FORCES ALL OF THE OTHER
                    (1153) * ALGORITHMS TO BE WRITTEN WITH STROBE ALWAYS EN-
                    (1154) * ABLED.
                    (1155) *
                    (1156) TIN     CPU      ,,,,,RSC CPN
040: 1203 4004 0000 0000
                    (1157)         CPU      BB   BPA  NX    0      0      RSC   STROBE ;
                    (1158)                  RY200     NOP   LOAD256K ;
                    (1159)                  JUMP ON DMTINNVAL TO TEXIT
041: F603 8808 0007 604A
                    (1160) TIN1    CPU      BB   BPD  NX    0      0      RSC   (ENB,STROBE) ;
                    (1161)                  RM280     AWRITE       NOP ;
                    (1162)                  GO TO TIN
042: FA03 94F4 0004 0040
                    (1163) *
                    (1164) *       DMT OUTPUT LOOP.
                    (1165) *
                    (1166) TOUT    CPU      0    NOP  NX    0      0      RSC   (STROBE,DATA) ;
```

```
                    (1167)                  RMMRDY
043: 1203 A704 0000 0000
                    (1168)          CPU     BB   BPA  NX    0       0      RSC   (CPN,DATA) ;
                    (1169)                  RY240     NOP   LOAD256K ;
                    (1170)                  JUMP ON DMTOUTNVAL TO TEXIT
044: F603 6308 0007 504A
                    (1171)          CPU     RF   NOP  NONE  0       0      RSC   STROBE ;
                    (1172)                  RF160
045: 1003 8904 0000 0000
                    (1173) TOUT1    CPU     0    NOP  NX    0       0      RSC   (ENB,STROBE) ;
                    (1174)                  280  AREAD      NOP ;
                    (1175)                  GO TO TOUT
046: 1203 96C4 0004 0043
                    (1176)          EJCT
```

```
                    (1177) *       THIS CODE TRANSFERS CONTROL FROM ONE MODE OF DMX
                    (1178) * TRANSFER TO ANOTHER OR RESUMES NORMAL USER LEVEL CODE
                    (1179) * EXECUTION.
                    (1180) *
                    (1181) DMA1    CPU     RF   NOP  NX    0      0     RSC   (DATA,CPN) ;
                    (1182)                 RF240     NOP   NOP ;
                    (1183)                 JUMP ON DMXSTATREQ TO REST
047: 1203 6B04 0005 D059
                    (1184) *
                    (1185) *       START OF DMC LOOP
                    (1186) *
                    (1187) DMC     CPU     BB   BPA  NX    0      0     RSC   STROBE ;
                    (1188)                 RY200     NOP   NOP ;
                    (1189)                 JUMP ON DMCVALID TO DMC1
048: F603 8804 0006 805C
                    (1190)         CPU     ,,,,,RSC STROBE ,,,JUMP ON DMANOTVALID TO DMT
049: 1203 8004 0005 E03E
                    (1191) TEXIT   CPU     ,,,,,RSC STROBE ,,,JUMP ON DMXSTATREQ TO REST
04A: 1203 8004 0005 D059
                    (1192) *
                    (1193) *
                    (1194) *       DMA.  THE ADDRESS LINES ARE READ INTO THE SHIFT COUNTER.
                    (1195) * THIS DIRECTLY SELECTS THE FIRST DMA REGISTER.  IT IS INCRE-
                    (1196) * MENTED BY $10 AND TESTED TO SEE IF ZERO IS CROSSED BY THE
                    (1197) * END OF BLOCK SIGNAL.  THE SECOND REGISTER IS ACCESSED AND AN
                    (1198) * INPUT OR OUTPUT TRANSFER IS BEGUN TO THE LOCATION READ.  A
                    (1199) * BPCENB IS GENERATED BEFORE THE COMPLETION OF THE TRANSFER TO
                    (1200) * DETECT CONSECUTIVE TRANSFERS.  IF CONSECUTIVE, LOOP BACK,
                    (1201) * IF NOT, EXIT.
                    (1202) *
                    (1203) *
                    (1204) DMA     CPU     0    BPA  NX    0      0     RSC   STROBE ;
                    (1205)                 200  NOP  LOADRSC ;
                    (1206)                 JUMP ON DMANOTVALID TO DMC
04B: 1603 800E 0005 E048
                    (1207)         CPU     AL   RCM  NX    ADD    0     RSC   (STROBE,CPN) ;
                    (1208)                 RYRF240   NOP   INCRSC ;
                    (1209)                 DATA $10
```

```
04C: 9263 CD05 0002 001C
                    (1210)          CPU     RF   NOP  NX    0      0     RSC   STROBE ;
                    (1211)                  RY200     NOP   LOAD256K ;
                    (1212)                  JUMP ON OUTPUT TO DOUT1
04D: 1203 8808 0006 5058
                    (1213)          CPU     BB   BPD  NONE  0      0     RSC   (ENB,STROBE) ;
                    (1214)                  RM280     AWRITE       NOP ;
                    (1215)                  GO TO DIN
04E: F803 94F4 0004 004F
                    (1216) *
                    (1217) *        CYCLE TO CYCLE, FULLY OVERLAPPED DMA INPUT ROUTINE.
                    (1218) *
                    (1219) DIN      CPU     AL   NOP  NX    INC    1     RSC   0 ;
                    (1220)                  RF200
04F: 9207 0A04 0000 000C
                    (1221)          RR      BPA =>  ,LOADRSC,  JUMP ON DMANOTVALID TO DMA1
050: F600 000E 0005 E047
                    (1222)          CPU     AL   RCM  NX    ADD    0     RSC   (CPN,STROBE) ;
                    (1223)                  RYRF240   NOP   INCRSC ;
                    (1224)                  DATA $10
051: 9263 CD05 0002 001C
                    (1225)          CPU     RF   NOP  NX    0      0     RSC   STROBE ;
                    (1226)                  RY200     NOP   LOAD256K ;
                    (1227)                  JUMP ON OUTPUT TO DOUT1
052: 1203 8808 0006 5058
                    (1228) DIN1     CPU     BB   BPD  NONE  0      0     RSC   (ENB,STROBE) ;
                    (1229)                  RM280     AWRITE       NOP ;
                    (1230)                  GO TO DIN
053: F803 94F4 0004 004F
                    (1231) *
                    (1232) *        DMA CYCLE-CYCLE OUTPUT ROUTINE.
                    (1233) *
                    (1234) DOUT     CPU     RF   ,NONE,,,RSC       STROBE       RF160
054: 1003 8904 0000 0000
                    (1235)          CPU     0    BPA  NX    0      0     RSC   (STROBE,DATA) ;
                    (1236)                  RMMRDY  NOP  LOADRSC ;
                    (1237)                  JUMP ON DMANOTVALID TO DMA1
055: 1603 A70E 0005 E047
```

```
                    (1238)          CPU     AL   RCM  NX    ADD    0     RSC   (CPN,DATA) ;
                    (1239)                  RYRF240   NOP   INCRSC ;
                    (1240)                  DATA $10
056: 9263 6D05 0002 0010
                    (1241)          CPU     RF   NOP  NX    0      0     RSC   STROBE ;
                    (1242)                  RY200     NOP   LOAD256K ;
                    (1243)                  JUMP ON INPUT TO DIN1
057: 1203 8808 0004 5053
                    (1244) DOUT1    CPU     AL   NOP  NONE  INC    1     RSC   (STROBE,ENB) ;
                    (1245)                  RF280     AREAD NOP ;
                    (1246)                  GO TO DOUT
058: 9007 9CC4 0004 0054
                    (1247)          EJCT
```

```
                    (1248) *
                    (1249) *       RESTORE ROUTINE.  RETURN TO NORMAL PROCESSING.
                    (1250) *
                    (1251) REST    RR       YSAVE  =>  RY
059: 0200 C804 0000 0000
                    (1252)         RR       RSCSAVE => RM
05A: 0000 E104 0000 0000
                    (1253)         CPU      RF   RM   ALL   0      0      M      MSAVE ;
                    (1254)                  RM280     NOP   LOADRSC ;
                    (1255)                  S ON TRUE, POP
05B: 0F00 D40E 000C 0004
                    (1256) *
                    (1257) *       DMC.  SLOWEST OF ALL THE TRANSFERS, DMC HAS AS ITS VIRTUE
                    (1258) * THE CAPABILITY OF USING ANY OF THE FIRST 64K MAIN
                    (1259) * MEMORY LOCATIONS AS A CHANNEL.  IDENTICAL IN FUNCTION
                    (1260) * TO DMA, THE TWO LOCATIONS MAKING UP A CHANNEL CONTAIN
                    (1261) * THE CURRENT AND ENDING MEMORY ADDRESSES, AND MUST BE ACCESSED
                    (1262) * TESTED AND UPDATED EACH CYCLE.
                    (1263) *
                    (1264) *
                    (1265) DMC1    CPU      0    RCM  NX    0      0      RSC    STROBE ;
                    (1266)                  RMMRDY    AREAD LOADRSC ;
                    (1267)                  DATA '17
05C: 1203 87CE 0002 010F
                    (1268)         CPU      BB   RCM  NX    0      0      RSC    STROBE ;
                    (1269)                  RF200     NOP   NOP ;
                    (1270)                  DATA 1
05D: F203 8A04 0002 0001
                    (1271)         CPU      AL   RM   NX    ADD    0      RSC    STROBE ;
                    (1272)                  RM280     AWRITE
05E: 8E63 84F4 0000 0000
                    (1273)         CPU      AL   RY   NX    ADD    0      RSC    STROBE ;
                    (1274)                  RY240     ,,GO TO DMC3
05F: 8A63 8304 0004 0086
                    (1275)         EJCT
```

```
                    (1276)          ORG     $60
                    (1277) *
                    (1278) *
                    (1279) *        TRAP ENTRY POINTS.  EVERY OTHER LOCATION FROM HERE TO
                    (1280) * $7E IS A POTENTIAL TRAP ENTRY POINT.  THOSE LOCATIONS
                    (1281) * WHERE TRAPS ARE NOT IMPLEMENTED ARE AVAILABLE FOR OTHER
                    (1282) * CODING.  PRIOITY IS FROM 6F TO 60, THEN FROM
                    (1283) * 7F TO 70.  THEREFORE MISSING MEMORY MODULE IS HIGHEST
                    (1284) * PRIORITY, AND DMA IS LOWEST.
                    (1285) *
                    (1286) *
                    (1287) *
                    (1288) *        FETCH PAGE TRAP.
                    (1289) *
                    (1290) FPAGE  CPU       BB    RCM   NONE  ,,,,280       ,PUSHBD ;
                    (1291)                  DATA FPAGE3
060: F000 060C 0002 008A
                    (1292)          RR      RY => YSAVE TR= NONE  ,  GO TO PAGE+1
061: E800 CA04 0004 007D
                    (1293) *
                    (1294) *        WRITE ADDRESS TRAP
                    (1295) *
                    (1296) WRITE  CPU       BB   RM    1      ,,RY    0      RF240 ;
                    (1297)                  ,,S ON TRUE, POP
062: ED02 0B04 000C 0004
                    (1298) PAGE3  CPU       AL    RM    NONE  BB      L      M      17 ;
                    (1299)                  RMRFMRDY  AREAD FORCERD ;
                    (1300)                  GO TO PAGE4
063: 8C5C F5C9 0004 0080
                    (1301) *
                    (1302) *        READ ADDRESS TRAP
                    (1303) *
                    (1304) READ   CPU       RF    NOP   1      ,,RY    0      RM280 ;
                    (1305)                  ,,S ON TRUE, POP
064: 1102 0404 000C 0004
                    (1306)          ORG     $66
                    (1307) *
                    (1308) *        FETCH READ ADDRESS TRAP
```

```
                    (1309) *
                    (1310) FREAD  CPU      RF   NOP  1      ,,RY   0      RM200 ;
                    (1311)                 ,,EAC LOADF01F02
066: 1102 0204 000A C000
                    (1312)        CPU      RF,,1,,,M,0,RF240,,,S ON TRUE, POP
067: 1100 0B04 000C 0004
                    (1313) *
                    (1314) *      RESTRICT EXECUTION MODE TRAP.  THOSE INSTRUCTIONS WHICH
                    (1315) *
                    (1316) * ARE RESTRICTED FORCE THIS TRAP IF THE MODE IS ENABLED.
                    (1317) *
                    (1318) RXM    RR       RCM = '62  => RY TR= NONE
068: F000 0804 0002 0032
                    (1319)        CPU      ,,NONE,,,,,RMMRDY,AREAD,,GO TO AVECT1
069: 1000 07C4 0004 0077
                    (1320) *
                    (1321) *      CENTRAL PROCESSOR PARITY .
                    (1322) *
                    (1323) CPPAR  ALU      CON 0 => RM  CLEARFUII TR= NONE  JUMP ON VERIFY TO VIRY1
06A: 903C 020F 0007 30C4
                    (1324)        RR       RP => RY  TR= NONE , GO TO MC4
06B: 0000 7804 0004 0034
                    (1325)        ORG      $6C
                    (1326) *
                    (1327) *      MEMORY PARITY ERROR
                    (1328) *
                    (1329) MEMPAR RR       RCM = '67  => RY  TR= NONE
06C: F000 0804 0002 0037
                    (1330)        ALU      CON 0  => RM  TR= 1    , GO TO MEMP3
06D: 913C 0204 0004 008C
                    (1331) *
                    (1332) *      MISSING MEMORY MODULE
                    (1333) *
                    (1334) MMOD   RR       RCM = '71  => RY  TR= NONE
06E: F000 0804 0002 0139
                    (1335)        RR       ,,TR= NONE       ,GO TO MEMPAR+1
06F: 0000 0004 0004 006D
                    (1336) *
```

```
                    (1337) *       DMX INSTRUCTION INTERRUPT ENTRY POINT
                    (1338) *
                    (1339) DMX1    RR       RM => MSAVE  TR= NX
070: EE00 DA04 0000 0000
                    (1340)         CPU      ,,NONE,,,RSC ENB 280 ,,GO TO  DMX2
071: 1003 1604 0004 003C
                    (1341) *
                    (1342) *       PAGE WRITE PROTECT VIOLATION
                    (1343) *
                    (1344) WRITEP  RR       RCM = '73 => RY  TR= 1
072: F100 0804 0002 003B
                    (1345)         CPU      ,,NONE,,,,,RMMRDY,AREAD,,GO TO AVECT1
073: 1000 07C4 0004 0077
                    (1346) *
                    (1347) *       POWER FAILURE INTERRUPT
                    (1348) *
                    (1349) FHALT2  ALU      CON ZERO => RM ,JUMP ON PFLNOT TO FHALT3
074: 923C 0204 0005 902D
                    (1350) PFL     RR       RCM = '60  => RY  TR= NONE
075: F000 0804 0002 0130
                    (1351)         CPU      AL   NOP  NONE  INC    1     M     RP ;
                    (1352)                  RMRFMRDY  AREAD  NOP ;
                    (1353)                  EAC  CLRPWFL  (TRIGNVKEYS,EINTM)
076: 9004 75C4 000B 0410
                    (1354) *
                    (1355) *       ABSOLUTE VECTOR   ---   THE FIRST ENTRY ALSO BACKS THE
                    (1356) * PROGRAM COUNTER UP BEFORE VECTORING.   THE LOCATION IS
                    (1357) * READ.  IF ZERO, THE PROCESSER HALTS, IF NOT, A JST TO THE
                    (1358) * LOCATION READ IS DONE.  NOTE THAT THE ADDRESS IS
                    (1359) * INTERPRETED AS AN ABSOLUTE 64K POINTER, NO INDEXING OR
                    (1360) * INDIRECTION.
                    (1361) *
           000167   (1362) AVECT   EQU      *
                    (1363) AVECT1  ALU      DEC RP  => RP TR= 1    CLEARFUII ;
                    (1364)                  JUMP ON FUII TO *
077: 91F0 7A0F 0005 5077
                    (1365)         CPU      AL   RM   NONE  BB     L     ,,RY240 ,SETCC ;
                    (1366)                  EAC CLRNVKEYS  TRIGNVKEYS
```

```
078: 8C5C 0307 000A 0400
                    (1367)          RR      RP => RM  TR= ALL  ,   ;
                    (1368)                  JUMP ON EQ TO CP1
079: 0300 7204 0006 4035
                    (1369)          RR      RY => RP  C= AWRITE  TR= ALL    ,GO TO CAS5
07A: EB00 7AF4 0004 01E5
                    (1370) *
                    (1371) *        PAGE TRAP .  PROCESS TO UPDATE CAM IF POSSIBLE
                    (1372) * AND CONTINUE THE INSTRUCTION.  IF NO PAGE IN MEMORY
                    (1373) * BACK UP THE P-COUNTER , AND PAGE FAULT INTERRUPT.
                    (1374) *
                    (1375)          ORG     $7C
                    (1376) PAGE     RR      RY => YSAVE
07C: E800 C904 0000 0000
                    (1377)          ALU     PMAR OR RY  => RY TR= 0   FLIP BYTES  DISABLERHBB ;
                    (1378)                  GO TO PAGE3
07D: 684C 830B 0004 0063
                    (1379) *
                    (1380) *        POST PROCESSING OF CENTRAL PROCESSOR PARITY ERROR
                    (1381) * AFTER MICRO-VERIFY HAS SUCESSFULLY BEEN EXECUTED.
                    (1382) *
                    (1383) CPPAR3   RR      RCM = '70   => RY       TR= NONE
07E: F000 0804 0002 0038
                    (1384)          CPU     ,,NONE,,,,,RMMRDY,AREAD,,GO TO AVECT+1
07F: 1000 07C4 0004 0078
                    (1385) PAGE4    RR      YSAVE => RY TR= NONE ,     ;
                    (1386)                  JUMP ON RMC1NOT TO PAGE7
080: 0000 C804 0006 1083
                    (1387)          CPU     RF   NOP  0     0      0     M     17 ;
                    (1388)                  RM280     ,,EAC LOADCAM
081: 1000 F404 0009 8000
                    (1389)          CPU     ,,ALL     ,,,,RMMRDY   MWRITE ;
                    (1390)                  HSMRESUME  S ON TRUE,POP
082: 1300 07BA 000C 0004
                    (1391) PAGE7    RR      RY => EAS  TR= 1
083: E900 AA04 0000 0000
                    (1392)          RR      RCM = '64  => RY       TR= 1
084: F100 0804 0002 0034
```

```
                    (1393)          CPU     ,,NONE,,,,,RMMRDY,AREAD,,GO TO AVECT1
085: 1000 07C4 0004 0077
                    (1394) *
                    (1395) *        DMC EXECUTION
                    (1396) *
                    (1397) DMC3     CPU     BB    RM    NX    0       0       RSC    STROBE ;
                    (1398)                  RMRFMRDY  AREAD
086: EE03 85C4 0000 0000
                    (1399)          CPU     RF    RM    NX    SUB     0       RSC    (CPN,STROBE) ;
                    (1400)                  RY240     ,,JUMP ON INPUT TO DMC2
087: 0E93 C304 0004 5089
                    (1401)          CPU     AL,,NX,DEC,0,RSC  (STROBE) RY240 ;
                    (1402)                  ,,GO TO DOUT1
088: 92F3 8304 0004 0058
                    (1403) DMC2     CPU     AL    NOP   NX    DEC     0       RSC    STROBE ;
                    (1404)                  RY240   ,,GO TO DIN1
089: 92F3 8304 0004 0053
                    (1405) *
                    (1406) *        FETCH PAGE FINISH OFF.  IT IS NECESSARY TO LOAD F01 AND2
                    (1407) *
                    (1408) FPAGE3   CPU     BB    RM    NONE  ,,,,RM200     ,,EAC LOADF01F02
08A: EC00 0204 000A C000
                    (1409)          CPU     RF,,1,,,M,0,RF240,,,S ON TRUE, POP
08B: 1100 0B04 000C 0004
                    (1410) *
                    (1411) *        FINISH THE MEMORY PARITY ERROR TRAP
                    (1412) *
                    (1413) MEMP3    CPU     AL    NOP   NONE  RF      0       M      RX ;
                    (1414)                  RF200     ,,EAC NOP     (TRIGNVKEYS,MCHK)
08C: 9000 0A04 0008 0404
                    (1415)          CPU     AL    NOP   1     RF      0       M      RA ;
                    (1416)                  RMRFMRDY  AREAD   ,GO TO AVECT+1
08D: 9100 15C4 0004 0078
                    (1417) FOUT1    ALU     RX ADD RCM = $20  => RX   TR= NONE  SETCC
08E: 9060 0A07 0002 0020
                    (1418)          RR      ,,,TR= NONE    JUMP ON NE TO *-1
08F: 0000 0004 0004 408E
                    (1419)          RR      ,,,TR= NONE    GO TO VIRY1
```

```
090: 0000 0004 0004 00C4
                    (1420)        EJCT
```

```
                    (1421) *      CONTROL PANEL BOOT.  THIS PROGRAM READS IN 512 WORDS FROM
                    (1422) * THE CONTROL PANEL AND PUTS THEM INTO LOCATIONS 6
                    (1423) * THROUGH '56 IN VIRTUAL MEMORY.  THE LOCATION POINTED TO
                    (1424) * BY THE P COUNTER IS THEN EXECUTED.
                    (1425) *
                    (1426) *      METHOD OF OPERATION.  AN OTA '1720 SENDS THE CONTROL
                    (1427) * PANEL THE LOCATION TO BE READ FROM ITS MEMORY.  AN INA
                    (1428) * '1420 IS THEN ISSUED TO GET THE CONTENTS OF THE LOCATION
                    (1429) * JUST ACCESSED.  THIS IS WRITTEN INTO MEMORY, THE POINTERS
                    (1430) * ARE INCREMENTED AND THE PROCESS REPETED UNTIL ALL 512 WORDS
                    (1431) * ARE TRANSFERRED.
                    (1432) *
                    (1433) *
                    (1434) *
                    (1435) BCOT   RR      RCM = 0  => RM
091: F000 0104 0002 0100
                    (1436)        RR      RCM = 0  => RB
092: F000 2904 0002 0100
                    (1437)        RR      RCM = 5  => RS
093: F000 3904 0002 0105
                    (1438) BOOT1  CPU     BB   RCM  NX    0      0     RY    PIO ;
                    (1439)                RY200     ,,DATA '171720
094: F202 1804 0003 E6D0
                    (1440)        CPU     BB   RCM  NX    0      0     RSC   (DATA,STROBE) ;
                    (1441)                RY280     ,,  DATA '131420
095: F203 AE04 0001 6610
                    (1442)        CPU     AL   NOP  NX    ZERO   L     RSC      0 ;
                    (1443)                280  NOP  NOP ;
                    (1444)                EAC  PLOADVKEYS TRIGVKEYS
096: 923F 0604 000A 4100
                    (1445)        CPU     AL   BPD  NX    BB     L     RSC   0 ;
                    (1446)                RM280
097: 9A5F 0404 0000 0000
                    (1447)        ALU     INC RS => (RY,RS)
098: 9204 3D04 0000 0000
                    (1448)        ALU     RS MINUS RCM = '56  => NULL  C= MWRITE SETCC
099: 9294 30B7 0002 012E
                    (1449)        ALU     INC RB  => RM
```

```
09A: 9204 2204 0000 0000
                    (1450)        ALU      INC  RB => RB ;
                    (1451)                 JAMF   JUMP ON LT TO BOOT1
09B: 9204 2A00 0006 6094
                    (1452)        EJCT
```

```
                    (1453) DIV27  ALU      DEC RB  => RM TR= ALL   ,JUMP ON GT TO IAB
09C: 93F0 2204 0006 71A2
                    (1454)        RR       RY => RB  ,GO TO LDA+1
09D: EA00 2A04 0004 01DF
                    (1455) DIV17  CPU      ALLS RM   NX    ADD    0     M     RA ;
                    (1456)                 200  DIVER      , ;
                    (1457)                 EAC DIVLOGIC
09E: AE60 1014 0009 0000
                    (1458)        CPU      RFLS 3    ALL   3      0     M     RB ;
                    (1459)                 RF200     LINK  , ;
                    (1460)                 JUMP ON FCBIT TO DIVER
09F: 2F30 2A44 0004 F146
                    (1461)        CPU      ALLS RM   NX    ADD    0     M     RA ;
                    (1462)                 RF280     LINK  ,EAC DIVLOGIC
0A0: AE60 1C44 0009 0000
                    (1463)        CPU      RFLS 3    ALL   3      0     M     RB ;
                    (1464)                 RF200     LINK  INCRSC ;
                    (1465)                 JUMP ON RSCNEM1 TO *-1
0A1: 2F30 2A45 0004 90A0
                    (1466) * SET RSC = 1 SO IAB CODE AT END WORKS
                    (1467)        CPU      ALLS RM   NX    ADD    0     M     RA ;
                    (1468)                 200  LINK INCRSC EAC DIVLOGIC
0A2: AE60 1045 0009 0000
                    (1469)        CPU      AL   RM   NX    ADD    0     M     RA ;
                    (1470)                 RYRF240   ,,EAC DIVLOGIC
0A3: 8E60 1D04 0009 0000
                    (1471)        CPU      RFLS 3    ALL   6      0     M     RB ;
                    (1472)                 RF200
0A4: 2F60 2A04 0000 0000
                    (1473)        CPU      AL   NOP  NX    ANOT   L     M     RB ;
                    (1474)                 RF200
0A5: 92FC 2A04 0000 0000
                    (1475)        ALU      INC RB => RB  TR= ALL   ,JUMP ON GE TO XCB
0A6: 9304 2A04 0004 61A8
                    (1476)        ALU      RA ADD RM  => RY       SETCC   GO TO DIV27
0A7: 8E60 1307 0004 009C
                    (1477) DIV4   CPU      ,,ALL RF   0     M     RA ;
                    (1478)                 200  ,SETCC     JUMP ON LT TO DIV17
```

```
0A8: 1300 1007 0006 609E
                    (1479)          CPU     ALLS RM   NX    SUB    1     M     RA ;
                    (1480)                  200  DIVER      ,EAC DIVLOGIC
0A9: AE94 1014 0009 0000
                    (1481)          CPU     RFLS 3    ALL   3      0     M     RB ;
                    (1482)                  RF200     LINK  ,JUMP ON FCBIT TO DIVER
0AA: 2F30 2A44 0004 F146
                    (1483)          CPU     ALLS RM   NX    SUB    1     M     RA ;
                    (1484)                  RF280     LINK  ,EAC DIVLOGIC
0AB: AE94 1C44 0009 0000
                    (1485)          CPU     RFLS 3    ALL   3      0     M     RB ;
                    (1486)                  RF200     LINK  INCRSC ;
                    (1487)                  JUMP ON RSCNEM1 TO *-1
0AC: 2F30 2A45 0004 90AB
                    (1488)          CPU     ALLS RM   NX    SUB    1     M     RA ;
                    (1489)                  200  LINK INCRSC EAC DIVLOGIC
0AD: AE94 1045 0009 0000
                    (1490)          CPU     AL   RM   NX   SUB   1      M     RA ;
                    (1491)                  RYRF240   ,,EAC DIVLOGIC
0AE: 8E94 1D04 0009 0000
                    (1492)          CPU     RFLS 3    ALL   6      0     M     RB ;
                    (1493)                  RF200     ,,JUMP ON GE TO XCB
0AF: 2F60 2A04 0004 61A8
                    (1494)          ALU     RA SUB RM  => RY SETCC
0B0: 8E94 1307 0000 0000
                    (1495)          CPU     AL   KEYS ALL   RF     1     M     RB ;
                    (1496)                  RM200     AOVFL , ;
                    (1497)                  JUMP ON GT IAB
0B1: 8704 2274 0006 71A2
                    (1498)          CPU     BB   RY   NX    0      0     M     RB ;
                    (1499)                  RF200     ,,GO TO LDA+1
0B2: EA00 2A04 0004 01DF
                    (1500) *
                    (1501) *        UNIMPLEMENTED INSTRUCTION VECTOR
                    (1502) *
                    (1503) UII3     RR      RCM = '66   => RY
0B3: F200 0804 0002 0136
                    (1504)          CPU     BB   RM   NX    0      0     M     11 ;
```

```
                    (1505)                  RMRFMRDY  AREAD   NOP ;
                    (1506)                  GO TO AVECT1
0B4: EE00 95C4 0004 0077
                    (1507) *
                    (1508) *        FINISH ILLEGAL INSTRUCTION VECTOR
                    (1509) *
                    (1510) ILL3     RR      RCM = '72  => RY
0B5: F200 0804 0002 013A
                    (1511)          RR      ,,TR= ALL ,GO TO UII3+1
0B6: 0300 0004 0004 00B4
                    (1512) STA3     ALU     INC RY => RY EAF
0B7: 8A65 130D 0000 0000
                    (1513)          CPU     RF   NOP  ALL   0       0      M      RB ;
                    (1514)                  RM280     MWRITE        JAMF
0B8: 1300 24B0 0000 0000
                    (1515) ADD3     ALU     INC RY => RY  EAF
0B9: 8A65 130D 0000 0000
                    (1516)          CPU     BB   RM   ALL   0       0      M      13 ;
                    (1517)                  RMRFMRDY  MREAD
0BA: EF00 B584 0000 0000
                    (1518)          ALU     RB PLUS RM => RB C= AOVFL
0BB: 8E60 2A74 0000 0000
                    (1519)          ALU     RB AND RCM = $7FFF  => RB  TR= ALL
0BC: 931C 2A04 0000 FFFF
                    (1520)          RR      13 => RM
0BD: 0000 B104 0000 0000
                    (1521)          ALU     RA PLUS RM + CBIT  => RA JAMF  C= AOVFL
0BE: 8E68 1A70 0000 0000
                    (1522) M11      CPU     ,,NX,,,RY,0,RMMRDY,AREAD,,GO TO MI2
0BF: 1202 07C4 0004 001C
                    (1523)          EJCT
```

```
(1524) *
(1525) *
(1526) *
(1527) *
(1528) *       MICRO-VERIFICATION ROUTINES.  THESE ROUTINES ARE DIVIDED
(1529) * UP INTO '13 TESTS.  THE TEST NUMBER IS LOCATED IN RY AT
(1530) * LEAT AT THE END OF THE TEST WHEN CHECKS ARE DONE
(1531) * TO SEE IF THE TEST WAS SUCESSFUL OR NOT.  IF A TEST FAILS, THE
(1532) * TEST NUMBER IS DISPLAYED ON THE CONTROL PANEL LIGHTS VIA THE
(1533) * THE BMA PATH.  THESE ROUTINES ARE EXECUTED FOR THREE DIFFERENT
(1534) * FUNCTIONS:
(1535) *                MASTER CLEAR:
(1536) *
(1537) *       THE TESTS ARE RUN UNTIL THEY ASE ALL PASSED.  UPON COMPLETION,
(1538) *       THE CONTROL PANEL ROUTINE IS ENTERED.  FAILURE OF ANY TEST
(1539) *       CAUSES A DELAY, FOLLOWED BY A RETRY OF ALL THE TESTS.
(1540) *
(1541) *                MACHINE CHECK:
(1542) *
(1543) *       ENTERED ON THE DETECTION OF A CPU PARITY ERROR, THE TESTS ARE
(1544) *       RUN EXACTLY AS IN MASTER CLEAR EXCEPT THAT WHEN THE TESTS ARE
(1545) *       SUCESSFULLY COMPLETED, THE MACHINE CHECK VECTOR IS
(1546) *       EXECUTED.
(1547) *
(1548) *                VERIFY  (VIRY):
(1549) *
(1550) *       THIS IS AN EXPLICIT USER REQUESTED EXECUTION OF THE TEST
(1551) *       ROUTINES.  SUCCESS IS REWARDED BY SKIPPING
(1552) *       THE NEXT SEQUENTIAL INSTRUCTION, FAILURE IS SHOWN
(1553) *       BY PUTTING THE FAILED TEST NUMBER INTO RA AND EXECUTING
(1554) *       THE NEXT SEQUENTIAL INSTRUCTION.
(1555) *
(1556) *
(1557) *       TEST 0:  TESTS ALU =0 ,SUB ,=0 CONDITIONAL BRANCH, RCM =0,
(1558) *                CONTROL UNIT AND SETCC.
(1559) *
(1560) *
(1561)         ORG     $C4
```

```
                    (1562) VIRY1  ALU     CON 0 => RM        TR= NONE
0C4: 903C 0204 0000 0000
                    (1563)        CPU     AL   RM   NONE  ZERO   L  M     RA  RYRF240  ,LOADRSC ;
                    (1564)                MODAL NOP (54,56,57,58,59,60,61,63,64)
0C5: 8C3C 1D0E 0008 05FB
                    (1565)        CPU     0    RCM  NONE  SUB    1     M     RA ;
                    (1566)                200  ,SETCC     DATA 0
0C6: 1094 1007 0002 0100
                    (1567)        CPU     0    RY   NONE  SUB    1     M     RA ;
                    (1568)                200  ,SETCC ;
                    (1569)                JUMP ON NE TO F1
0C7: 0894 1007 0004 4000
                    (1570) *
                    (1571) *      TEST 1:   TESTS RY => BB
                    (1572) *
                    (1573)        ALU     CON 0  => RX  ,TR= NONE  EAC CLRNVKEYS (54,56,62)
0C8: 903C 0A04 000A 0504
                    (1574)        ALU     INC RY => RY  ,TR= NONE  ;
                    (1575)                JUMP ON NE TO FOUT
0C9: 8865 1304 0004 40FC
                    (1576) *
                    (1577) *      TEST 2:   TESTS MACHINE CHECK RESET
                    (1578) *
                    (1579)        ALU     INC RY => RY  TR= NONE , ;
                    (1580)                JUMP ON MC TO FOUT
0CA: 8865 1304 0007 10FC
                    (1581) *
                    (1582) *      TEST 3:   TESTS SHIFT COUNTER INCREMENTING AND
                    (1583) *                THE RSCNOTEQM1 TEST. ALSO RX+1 => RX, AND COUNTS FOR
                    (1584) *                PROPER NUMBER OF INCREMENTS.  (THIS TEST WILL SHOW AS #2
                    (1585) *                IF RSCNEM1 IS ALWAYS TRUE.)
                    (1586) *
                    (1587)        ALU     INC RX => RX  TR= NONE  INCRSC ;
                    (1588)                JUMP ON RSCNEM1 TO *
0CB: 9004 0A05 0004 90CB
                    (1589)        ALU     RX MINUS RCM = $40   => NULL  TR= NONE  SETCC
0CC: 9094 0007 0002 0040
                    (1590)        ALU     INC RY => RY  TR= NONE , ;
```

```
                    (1591)                  JUMP ON NE TO FOUT
0CD: 8865 1304 0004 40FC
                    (1592) *
                    (1593) *        TESTS 4,5,6:   THESE TESTS VERIFY THAT EACH OF THE REGISTERS
                    (1594) *                       CAN DETECT BAD PARITY.  4 TESTS RY, 5 RM, AND 6 RF.  ALSO
                    (1595) *                       CHECKED:  16 WAY BRANCH, MC- TEST.
                    (1596) *
                    (1597)          CPU     BB    RCM  NONE  ,,,,RY200     ,, ;
                    (1598)                  EMIT  $100  $003
0CE: F000 0804 0002 0003
                    (1599) VIRY12   ALU     CON 0 => RM  TR= NX
0CF: 923C 0204 0000 0000
                    (1600)          CPU     AL    NOP  NONE  ZERO   L      ,,RF160
0D0: 903C 0904 0000 0000
                    (1601)          ALU     INC RY => RY  TR= NONE , ;
                    (1602)                  JUMP ON MCNOT TO FOUT
0D1: 8865 1304 0007 20FC
                    (1603)          RR      ,,,TR= NONE  MODAL NOP (TRIGNVKEYS,MCHK)
0D2: 0000 0004 0008 0404
                    (1604)          CPU     BB    RY    NONE  ,,,,280       ,, ;
                    (1605)                  S ON TRUE, 16WAYS TO VIRY12+1
0D3: E800 0604 000C 00D2
                    (1606)          RR      RCM => RM TR= NONE ,GO TO VIRY12
0D4: F000 0204 0004 00CF
                    (1607)          RR      RCM => RX TR= NONE ,GO TO VIRY12
0D5: F000 0A04 0004 00CF
                    (1608) *
                    (1609) *        TEST 7:   PARITY ON LEFT SHIFT,ROTATE
                    (1610) *
                    (1611)          ALU     CON -1  => RA , EAC LOADSERIALINT
0D6: 92CC 1A04 0008 4000
                    (1612) * LOGICAL LEFT SHIFT
                    (1613)          CPU     RFLS 7    NX    7      0      M      RA ;
                    (1614)                  RF200     RF01
0D7: 3E70 1A54 0000 0000
                    (1615) * LOGICAL LEFT ROATATE
                    (1616)          CPU     RFLS 1    NX    8      0      M      RA ;
                    (1617)                  RF200    ,, ;
```

```
                    (1618)                  JUMP ON MC TO VIRY12-5
0D8: 2680 1A04 0007 10CA
                    (1619) * ROTATE 64 TIMES
                    (1620)          RR      RA => RA  INCRSC ;
                    (1621)                  JUMP ON RSCNEM1 TO *-1
0D9: 0200 1A05 0004 90D8
                    (1622) *
                    (1623) *        TEST 7 (CONT):  TEST C BIT FOR SET, LOAD AND RESET
                    (1624) * CBIT SHOULD BE SET SO RA IS TESTED AGAINST -2
                    (1625) *
                    (1626)          ALU     RA MINUS RCM = -2  + CBIT  => RA ;
                    (1627)                  C= BD01  SETCC
0DA: 9298 1A37 0003 FEFE
                    (1628)          CPU     RF   NOP  NX    MINUS1 L     M     RP ;
                    (1629)                  RM200     ,SETCC ;
                    (1630)                  JUMP ON NE TO VIRY12-2
0DB: 12CC 7207 0004 40CD
                    (1631) *
                    (1632) *        TEST '10:   WRITE BITS THROUGH REGISTER FILE.  CHECK
                    (1633) *                PATTERN USING EXCLUSIVE ORS TO VERIFY PROPPER
                    (1634) *                OPERATION.  ZERO OUT FILES.
                    (1635) *
                    (1636)          RR      RCM = $FFC0  => RX  TR= NX
0DC: F200 0A04 0003 FFC0
                    (1637)          CPU     AL   NOP  NONE  RF     1     RSC   0 ;
                    (1638)                  RY240     ,INCRSC ;
                    (1639)                  JUMP ON EQ TO *+2
0DD: 9007 0305 0006 40DF
                    (1640)          CPU     BB   RY   NONE  BB     L     RSC   0 ;
                    (1641)                  RF200     ,SETCC ;
                    (1642)                  GO TO *-1
0DE: E85F 0A07 0004 00DD
                    (1643)          RR      RCM = $FFE1  => RY
0DF: F200 0804 0003 FFE1
                    (1644)          ALU     INC RY  => RY INCRSC ;
                    (1645)                  JUMP ON NE TO *+4
0E0: 8A65 1305 0004 40E4
                    (1646)          CPU     RF   NOP  NX    INC CBIT     RSC   0 ;
```

```
                    (1647)                  200  ,SETCC
0E1: 120B 0007 0000 0000
                    (1648)          CPU     AL   RY   NX    XOR    L     RSC   0 ;
                    (1649)                  RF200     ,SETCC ;
                    (1650)                  JUMP ON NE TO *-2
0E2: 8A6F 0A07 0004 40E0
                    (1651)          CPU     AL   RY   NX    ZERO   L     M     RA ;
                    (1652)                  RF200     ,LOADRSC
0E3: 8A3C 1A0E 0000 0000
                    (1653)          RR      RM  => RP  TR= NX
0E4: EE00 7A04 0000 0000
                    (1654)          RR      RCM = '10  => RY
0E5: F200 0804 0002 0008
                    (1655)          RR      ,,,JUMP ON NE TO FOUT
0E6: 0200 0004 0004 40FC
                    (1656) *
                    (1657) *        TEST '11:   I/O BUS TEST.  ALTENATING ONES AND ZEROS ARE
                    (1658) *            SENT OUT BOTH BPA AND BPD AND READ BACK IN.  THIS
                    (1659) *            CHECKS CPU LOU C IN THE PATH AND 'STUCK AT' FAULTS ON THE
                    (1660) *            CONTROLERS.
                    (1661) *
                    (1662)          CPU     BB   RCM  NX    0      0     RSC   0 ;
                    (1663)                  RYRF240   ,,DATA $A5A5
0E7: F203 0D04 0003 4BA5
                    (1664)          CPU     BB   RY   NX    0      0     RY    PIO ;
                    (1665)                  RM200
0E8: EA02 1204 0000 0000
                    (1666)          CPU     BB   BPA  NX    XOR    L     RSC   DATA ;
                    (1667)                  RY240     ,SETCC
0E9: F66F 2307 0000 0000
                    (1668)          CPU     BB   BPD  NX    XOR    L     RSC   DATA ;
                    (1669)                  RM200     ,SETCC ;
                    (1670)                  JUMP ON NE TO *+4
0EA: FA6F 2207 0004 40EE
                    (1671)          CPU     RFRS 2    NX    2      0     RSC   0 ;
                    (1672)                  RYRF240   ,, ;
                    (1673)                  JUMP ON NE TO *+3
0EB: 4A23 0D04 0004 40EE
```

```
                    (1674)          CPU     BB   BPA  NX    XOR    L     RSC   DATA ;
                    (1675)                  RY240     ,SETCC
0EC: F66F 2307 0000 0000
                    (1676)          CPU     BB   BPD  NX    XOR    L     RSC   DATA ;
                    (1677)                  RM200     ,SETCC ;
                    (1678)                  JUMP ON EQ TO *+3
0ED: FA6F 2207 0006 40FC
                    (1679)          RR      RCM = '11  => RY
0EE: F200 0804 0002 0109
                    (1680)          CPU     AL   NOP  NX    ZERO   L     RSC   0 ;
                    (1681)                  RF200     ,,GO TO FOUT
0EF: 923F 0A04 0004 00FC
                    (1682) *
                    (1683) *        TEST '12:   MEMORY TEST.  LOCATION 5 (UNDER THE REGISTERS)
                    (1684) *            IS TESTED TO SEE IF ONES AND ZEROS CAN BE WRITTEN AND READ.
                    (1685) *            NOTE THAT A MISSING FIRST 8K CAN CAUSE A MISSING MEMORY
                    (1686) *            TRAP.
                    (1687) *
                    (1688)          RR      RCM = 5   => RY
0F0: F200 0804 0002 0105
                    (1689)          CPU     BB   RM   NONE  0     0     M     RX ;
                    (1690)                  RF200     AWRITE
0F1: EC00 0AF4 0000 0000
                    (1691)          CPU     BB   RY   NONE  0     0     0     0 ;
                    (1692)                  RY200
0F2: E800 0804 0000 0000
                    (1693)          CPU     ,,NONE    ,,,,RMMRDY   AREAD
0F3: 1000 07C4 0000 0000
                    (1694)          ALU     RX XOR RM => RX SETCC
0F4: 8E6C 0A07 0000 0000
                    (1695)          RR      RCM = '12  => RY  TR= NONE
0F5: F000 0804 0002 010A
                    (1696)          RR      ,,,JUMP ON NE TO FOUT
0F6: 0200 0004 0004 40FC
                    (1697) *
                    (1698) *        TEST '13:   PARITY TEST.  AFTER VERIFYING PARITY WORKS IN
                    (1699) *            TESTS 4,5, AND 6, THE REST OF THE TESTS ARE RUN WITH
                    (1700) *            PARITY ENABLED.  IF MACHINE CHECK GETS SET DURING THOSE
```

```
                    (1701) *                TESTS, '13 WILL FAIL.
                    (1702) *
                    (1703)          ALU     INC RY => RY,, JUMP ON MC TO FOUT
0F7: 8A65 1304 0007 10FC
                    (1704) *
                    (1705) *        SUCCESSFUL TEST COMPLETION
                    (1706) *
                    (1707)          CPU     BB   RCM  NX    RF     0     M     RP ;
                    (1708)                  RY200     BD01  SETCC  DATA '1000
0F8: F200 7837 0000 0500
                    (1709)          ALU     CON 0 => RM ,JUMP ON EQ TO MC4
0F9: 923C 0204 0006 4034
                    (1710)          RR      ,,CLEARFUII  JUMP ON FUII TO CAS5
0FA: 0200 000F 0005 51E5
                    (1711)          RR      ,,,GO TO CPPAR3
0FB: 0200 0004 0004 007E
                    (1712) *
                    (1713) *        FAILED TEST COMPLETION/RETRY
                    (1714) *
                    (1715) FOUT     CPU     BB   RY   NONE  0     0     M     RA ;
                    (1716)                  RF200    ,, ;
                    (1717)                  JUMP ON FUII TO F1
0FC: E800 1A04 0005 5000
                    (1718)          ALU     CON 0  => RX   TR= NONE ,  GO TO FOUT1
0FD: 903C 0A04 0004 008E
                    (1719)          EJCT
```

```
                    (1720)          ORG     $100
                    (1721) ENTR     ALU     RS MINUS RY  => RY
100: 8A94 3304 0000 0000
                    (1722)          CPU     RF,,ALL,,,M,RS,RM280,MWRITE
101: 1300 34B4 0000 0000
                    (1723)          RR      RY => RS  JAMF
102: E800 3900 0000 0000
                    (1724) FLX1     RR      RM  => RX  ,GO TO FLX2
103: EE00 0A04 0004 012A
                    (1725)          ORG     $104
                    (1726) RTN      RR      RS => RY
104: 0200 3804 0000 0000
                    (1727)          CPU     ,,ALL,,,,,RMMRDY,MREAD
105: 1300 0784 0000 0000
                    (1728)          ALU     INC RM => RY
106: 8E65 1304 0000 0000
                    (1729)          CPU     BB   RM   ALL   BB     L     M     11 ;
                    (1730)                  RMRFMRDY  MREAD SETCC
107: EF5C 9587 0000 0000
                    (1731)          RR      11 => RY   ,JUMP ON EQ TO *+3
108: 0200 9804 0006 410B
                    (1732)          RR      RY => RS
109: E800 3904 0000 000C
                    (1733)          RR      RM => RP  JAMF  TR= NX
10A: EE00 7A00 0000 0000
                    (1734)          RR      RCM = '75  => RY
10B: F200 0804 0002 003D
                    (1735)          RR      RP  => RM  ,GO TO UII3+1
10C: 0200 7204 0004 00B4
                    (1736) FGEN     CPU     RF   NOP  NX    0      0     M        0 ;
                    (1737)                  RF240     0     NOP    GO TO FGEN1
10D: 1200 0B04 0004 0406
                    (1738)          ORG     $110
                    (1739) *
                    (1740) *        INSTRUCTION EXECUTION SPACE
                    (1741) *
          000420    (1742) EVMX     EQU     *
          000420    (1743) ERMX     EQU     *
```

```
          000420    (1744) EVMJ    EQU     *
                    (1745) ERMJ    RR      RP => RY  NOP  JUMP ON RXM TO RXM
110: 0200 7E04 0007 406E
                    (1746)         CPU     BB   RM   ALL   0      0     M     13 ;
                    (1747)                 RMRFMRDY  MREAD NOP ;
                    (1748)                 MODAL     NOP   (TRIGNVKEYS,RXM,EINTM)
111: EF00 B584 0008 0411
                    (1749)         RR      RM => RY  EAF   EAC  SETFUII
112: EE00 080D 000A 8000
                    (1750)         CPU     AL   NOP  ALL   INC    1     M        RP ;
                    (1751)                 RMRFMRDY  MREAD NOP ;
                    (1752)                 JUMP ON RSC12 TO EVMX1
113: 9304 7584 0005 B1F2
                    (1753)         RR      RM => RP  TR= NX
114: EE00 7A04 0000 00CC
                    (1754)         RR      13 => RM  TR= NX   ,EAC CLEARCAM
115: 0200 B204 0009 C000
                    (1755) EPMJ6   RR      ,,JAMF TR= ALL   MODAL NOP (TRIGNVKEYS,PAM)
116: 0300 000C 000E 0420
                    (1756) ERM     RR      ,,RESTJAMF    MODAL NOP (TRIGNVKEYS,RXM,EINTM)
117: 0200 0002 0008 0411
                    (1757) NRM     RR      RCM = 0   =>  RM  C= BD01  LOADRSC
118: F000 013E 0002 0100
                    (1758)         CPU     RF   NOP  ALL   0      0     M     RA ;
                    (1759)                 200  SOVFL
119: 1300 1064 0000 0000
                    (1760)         CPU     RFLS 7    NX    3      0     M        RB ;
                    (1761)                 RF200     LINK  , ;
                    (1762)                 JUMP ON FCBIT TO *+2
11A: 3E30 2A44 0004 F11C
                    (1763)         CPU     RFLS 3    ALL   7      0     M     RA ;
                    (1764)                 RF200     SOVFL INCRSC ;
                    (1765)                 JUMP ON RSCNEM1 TO *-2
11B: 2F70 1A65 0004 9119
                    (1766)         CPU     RFRS 0    NX    6      0     M     RB ;
                    (1767)                 RF200
11C: 4260 2A04 0000 0000
                    (1768)         RR      RSC => 13
```

```
11D: E400 B904 0000 0000
                    (1769)          ALU     13 AND RCM =  $3F  => (RY,13)  TR= TDMX
11E: 911C BD04 0002 013F
                    (1770)          RR      ,,,GO TO OTK+2
11F: 0200 0004 0004 0194
          000440    (1771) EPMX     EQU     *
          000440    (1772) LPMX     EQU     *
          000440    (1773) LPMJ     EQU     *
                    (1774) EPMJ     RR      RP => RY NOP  JUMP ON RXM TO RXM
120: 0200 7804 0007 4068
                    (1775)          CPU     BB   RM   ALL   0      0      M      13 ;
                    (1776)                  RMRFMRDY  MREAD  ,GO TO ERMJ+2
121: EF00 B584 0004 0112
                    (1777) XEC      RR      ,,,JUMP ON FETCH1 TO *+2
122: 0200 0004 0004 A124
                    (1778)          CPU     RF   NOP  NX   0      0      M      RA ;
                    (1779)                  CLMCLFCLI  MREAD  ,GO TO F9
123: 1200 1F84 0004 0004
                    (1780)          ALU     DEC RP => RP CLEARFUII JUMP ON FUII TO *
124: 92F0 7A0F 0005 5124
                    (1781)          RR      ,,JAMF
125: 0200 0000 0000 0000
                    (1782) VIRY     RR      ,,REST    EAC SETFUII
126: 0200 0006 000A 8000
                    (1783)          RR      ,,,GO TO VIRY1
127: 0200 0004 0004 00C4
          000450    (1784) CEA3     EQU     *
                    (1785) EAA      RR      RY => RA  TR= NX  JAMF
128: EA00 1A0C 0000 0000
                    (1786) FLX      CPU     ,,ALL,,,,,RMMRDY MREAD  ,GO TO FLX1
129: 1300 0784 0004 0103
                    (1787) FLX2     CPU     RFLS 7    NX    3      0      M      RX ;
                    (1788)                  RF200     ,JAMF
12A: 3E30 0A0C 0000 000C
                    (1789) JEQ      ALU     INC RA + 0  => NULL SETCC
12B: 9200 1007 0000 0000
                    (1790)          RR      ,,JAMF JUMP ON EQ TO JMP
12C: 0200 000C 0006 410D
```

```
                    (1791) JNE     ALU     INC RA + 0  => NULL SETCC
12D: 9200 1007 0000 0000
                    (1792)         RR      ,,JAMF JUMP ON NE TO JMP
12E: 0200 0000 0004 41D0
                    (1793) JLE     ALU     INC RA + 0  => NULL SETCC
12F: 9200 1007 0000 0000
                    (1794)         RR      ,,JAMF JUMP ON LE TO JMP
130: 0200 0000 0004 71D0
                    (1795) JGT     ALU     INC RA + 0  => NULL SETCC
131: 9200 1007 0000 0000
                    (1796)         RR      ,,JAMF JUMP ON GT TO JMP
132: 0200 0000 0006 71D0
                    (1797) AOA     ALU     INC RA => RA C= AOVFL  JAMF  DATA 0
133: 9204 1A70 0002 0100
                    (1798) JLT     ALU     INC RA + 0  => NULL SETCC
134: 9200 1007 0000 0000
                    (1799)         RR      ,,JAMF JUMP ON LT TO JMP
135: 0200 0000 0006 61D0
                    (1800) JGE     ALU     INC RA + 0  => NULL SETCC
136: 9200 1007 0000 0000
                    (1801)         RR      ,,JAMF JUMP ON GE TO JMP
137: 0200 0000 0004 61D0
                    (1802) JDX     ALU     DEC RX => RX  SETCC ;
                    (1803)                 GO TO JNE+1
138: 92F0 0A07 0004 012E
                    (1804) JIX     ALU     INC RX => RX  SETCC ;
                    (1805)                 GO TO JNE+1
139: 9204 0A07 0004 012E
                    (1806) JSX     RR      RP => RM
13A: 0000 7104 0000 0000
                    (1807)         RR      RM => RX ,GO TO JMP
13B: EE00 0A04 0004 01D0
                    (1808) CREP    RR      RY => 13
13C: E800 B904 0000 0000
                    (1809)         ALU     INC RS => RY
13D: 9204 3304 0000 0000
                    (1810)         CPU     RF   NOP  ALL   0      0     M     RP ;
                    (1811)                 RM280     MWRITE
```

```
13E: 1300 74B4 0000 000C
                    (1812)          RR      13 => RY ,GO TO JMP
13F: 0200 B804 0004 01D0
                    (1813) *
                    (1814) *        UNIMPLEMENTED INSTRUCTION VECTOR (UII)
                    (1815) *
                    (1816) UII      RR      RY => EAS TR= ALL
140: EB00 AA04 0000 0000
                    (1817)          RR      RP => RM  ,GO TO UII3
141: 0200 7204 0004 00B3
                    (1818) *
                    (1819) *        ILLEGAL INSTRUCTION VECTOR (ILL)
                    (1820) *
                    (1821) ILL      RR      RY => EAS TR= NX
142: EA00 AA04 0000 0000
                    (1822)          RR      RP => RM  ,GO TO ILL3
143: 0200 7204 0004 00B5
                    (1823) DIV      CPU     0    RCM  ALL   0      0      0     0 ;
                    (1824)                  RMMRDY    MREAD LOADRSC ;
                    (1825)                  DATA -15
144: 1300 078E 0003 FEF1
                    (1826)          CPU     0    RM   NX    XOR    L      M     RA ;
                    (1827)                  200  ,SETCC     GO TO DIV4
145: 0E6C 1007 0004 00A8
                    (1828) DIVER    CPU     RFRS 0    NX    6      0      M     RB ;
                    (1829)                  RF200     0     JAMF
146: 4260 2A00 0000 0000
                    (1830) LDX      CPU     ,,ALL,,,,,RMMRDY   MREAD
147: 1300 0784 0000 0000
                    (1831)          CPU     BB   RM   NX    0      0      XM    0 ;
                    (1832)                  RF200     ,JAMF
148: EE01 0A00 0000 0000
                    (1833) *
                    (1834) *
                    (1835) *        INSTRUCTION EXECUTION.  IN THIS SECTION, INSTRUCTIONS
                    (1836) * (INDICATED BY THE LABELS) ARE EXECUTED.
                    (1837) *
                    (1838) *
```

```
                    (1839) *      SEVERAL GEN B INSTRUCTIONS DECODE TO THE SAME PLACE. AS
                    (1840) * A RESULT, THEY MUST BE SPECIALLY DECODED BY TESTING FOR DIF-
                    (1841) * FERENT OP CODES IN RM.
                    (1842) *
                    (1843) GENB   RR      ,,,JUMP ON RSC12 TO RMC
149: 0200 0004 0005 B187
                    (1844)        RR      ,,,JUMP ON RS16 TO GENB1
14A: 0200 0004 0005 2183
                    (1845) HLT    RR      RP => RY REST GO TO CP1
14B: 0200 7806 0004 0035
                    (1846) SUB    CPU     ,,ALL,,,,,RMMRDY   MREAD , JUMP ON DP TO SUB3
14C: 1300 0784 0005 C16C
                    (1847)        ALU     RA SUB RM => RA C= AOVFL JAMF
14D: 8E94 1A70 0000 0000
                    (1848) JST    CPU     ,,ALL,,,,,RMMRDY   MREAD
14E: 1300 0784 0000 0000
                    (1849)        CPU     AL   RM   NX    RF     0     M     RP ;
                    (1850)                RM200     ,,EAC JST
14F: 8E00 7204 0008 8000
                    (1851) JST1   CPU     BB   RY   ALL   0      0     M     RP ;
                    (1852)                RF200     MWRITE       ,GO TO CAS5
150: EB00 7AB4 0004 01E5
                    (1853)        EJCT
```

```
                    (1854) *
                    (1855) *       PROGRAMMED INPUT/OUTPUT.  INA,OTA,SKS,AND OCP ARE
                    (1856) * ALL EXECUTED USING THIS COMMON CODE. INSTRUCTION
                    (1857) * SPECIFIC CODE IS EXECUTED AFTER TESTING THE TWO HIGH
                    (1858) * ORDER BITS OF THE OP CODE.  TH  TESTS ON DEVICE
                    (1859) * ADDRESS 20 ARE TO AVOID TESTING READY, CLEAN UP PARITY
                    (1860) * AND NOT SKIP FOR DEVICE 20 DEVICES INCLUDING THE CONTROL
                    (1861) * PANEL THE REAL TIME CLOCK, AND THE MASK CHANGE COMMANDS.
                    (1862) *
                    (1863) INA     CPU      0     RCM  NX    0      0     RY    PIO ;
                    (1864)                  280   NOP  LOADRSC ;
                    (1865)                  DATA 1
151: 1202 160E 0002 0001
                    (1866)         CPU      RF    NOP  NX    0      0     RSC   0 ;
                    (1867)                  RM200       NOP   NOP ;
                    (1868)                  JUMP ON F01NOT TO SKS
152: 1203 0204 0005 3157
                    (1869)         CPU      BB    BPD  NX    0      0     RSC   0 ;
                    (1870)                  RF280       0     NOP ;
                    (1871)                  JUMP ON READYNOTANDNE20 TO INA1
153: FA03 0C04 0005 7156
                    (1872)         CPU      0     NOP  NONE  0      0     RSC   STROBE ;
                    (1873)                  200   NOP  NOP ;
                    (1874)                  JUMP ON BPSP1NOT TO INA2
154: 1003 8004 0006 318A
                    (1875)         CPU      AL    NOP  NONE  RF     0     RSC   0 ;
                    (1876)                  RM200       NOP   NOP ;
                    (1877)                  JUMP ON DANOT20 TO CAS5
155: 9003 0204 0005 61E5
                    (1878) INA1    CPU      AL    RM   NONE  BB     L     RSC   0 ;
                    (1879)                  RF200       NOP   JAMF
156: 8C5F 0A0C 0000 0000
                    (1880) SKS     CPU      ,,NX,,,RSC,0,280,,,JUMP ON READYANDF02 TO SKS1
157: 1203 0604 0005 81EC
                    (1881)         CPU      ,,NX,,,RSC,STROBE,,,,GO TO INA1
158: 1203 8004 0004 0156
         000531     (1882) PIO     EQU      *
                    (1883) OTA     RR       RM => RY  TR= ALL  REST  JUMP ON OTANOT TO INA
```

```
159: EF00 0806 0005 4151
                (1884)          CPU     0    RCM  NX    0      0     RY    PIO ;
                (1885)                  280  0    LOADRSC  DATA 1
15A: 1202 160E 0002 0001
                (1886)          CPU     RF   RCM  NX    0      0     RSC   0 ;
                (1887)                  RM200     NOP   LOADRSC  DATA 7
15B: 1203 020E 0002 0007
                (1888)          CPU     0    NOP  NX    0      0     RSC   DATA ;
                (1889)                  280  NOP  NOP ;
                (1890)                  JUMP ON READYNOTANDNE20 TO F1
15C: 1203 2604 0005 7000
                (1891)          CPU     ,,,,,RSC  (DATA,STROBE)  200 ,,JUMP ON DANOT20 TO OTA1
15D: 1203 A004 0005 615F
                (1892)          CPU     ,,ALL,,,RSC DATA  200 ,JAMF
15E: 1303 2000 0000 000C
                (1893) OTA1     CPU     AL   NOP  ALL   INC    1     RSC   DATA ;
                (1894)                  RF200  NOP  EAF ;
                (1895)                  GO TO F1
15F: 9307 2A0D 0004 0000
                (1896)          EJCT
```

```
                    (1897)          ORG     $160
                    (1898) LDA3     ALU     INC RY => RY EAF
160: 8A65 130D 0000 0000
                    (1899)          CPU     BB   RM   ALL   0      0     M     RA ;
                    (1900)                  RMRFMRDY  MREAD
161: EF00 1584 0000 0000
                    (1901)          RR      RM => RB  TR= NX       JAMF
162: EE00 2A00 0000 000C
                    (1902) FLD      CPU     BB   RCM  ALL   0      0     M         VSC ;
                    (1903)                  RMRFMRDY  MREAD NOP    DATA  $00FF
163: F300 6584 0002 01FF
                    (1904)          CPU     BB   RM   NX    0      0     M         FLTH ;
                    (1905)                  RF200     NOP   NOP    GO TO FLD1
164: EE00 4A04 0004 0456
                    (1906) FAD      CPU     0    NOP  ALL   0      0     0         0 ;
                    (1907)                  RMMRDY    MREAD NOP    GO TO FAD1
165: 1300 0784 0004 040B
                    (1908) FSB      CPU     0    NOP  ALL   0      0     0         0 ;
                    (1909)                  RMMRDY    MREAD NOP    GO TO FSB1
166: 1300 0784 0004 040D
                    (1910) FMP      CPU     0    NOP  ALL   0      0     0         0 ;
                    (1911)                  RMMRDY    MREAD NOP    GO TO FMP1
167: 1300 0784 0004 0417
                    (1912) FDV      CPU     0    NOP  ALL   0      0     0         0 ;
                    (1913)                  RMMRDY    MREAD NOP    GO TO FDV1
168: 1300 0784 0004 042B
                    (1914) FST      CPU     RF   NOP  NX    0      0     M         FLTH ;
                    (1915)                  RM200     NOP   NOP    GO TO FST1
169: 1200 4204 0004 0464
                    (1916) FCS      CPU     BB   RCM  ALL   0      0     M         11 ;
                    (1917)                  RMRFMRDY  MREAD NOP    DATA  $00FF
16A: F300 9584 0002 01FF
                    (1918)          CPU     BB   RM   NX    XOR    L     M         FLTH ;
                    (1919)                  280  BD01 SETCC GO TO FCS1
16B: EE6C 4637 0004 046B
                    (1920) SUB3     ALU     INC RY => RY EAF
16C: 8A65 130D 0000 0000
                    (1921)          CPU     BB   RM   ALL   0      0     M      13 ;
```

```
                    (1922)                  RMRFMRDY  MREAD
16D: EF00 B584 0000 0000
                    (1923)          ALU     RB MINUS RM  => RB SETCC
16E: 8E94 2A07 0000 0000
                    (1924)          RR      13 => RM  TR= NX ,   JUMP ON GE SUB+1
16F: 0200 B204 0004 614D
                    (1925)          ALU     RB AND RCM = $7FFF  => RB  TR= ALL
170: 931C 2A04 0000 FFFF
                    (1926)          ALU     RA SUB RM + 0  => RA  C= AOVFL  JAMF
171: 8E90 1A70 0000 0000
                    (1927) PIM      ALU     RA AND RCM = $8000  => RM
172: 921C 1204 0001 0100
                    (1928)          ALU     RB OR RM  => RM ,GO TO LDA+1
173: 8E4C 2204 0004 01DF
                    (1929) PID      ALU     INC RA => RY + 0 SETCC
174: 9200 1307 0000 0000
                    (1930)          ALU     CON -1 => RA    ,JUMP ON GE TO XCA+1
175: 92CC 1A04 0004 61A3
                    (1931)          RR      RY => RB
176: E800 2904 0000 0000
                    (1932)          ALU     RB AND RCM = $7FFF  => RB    JAMF
177: 921C 2A00 0000 FFFF
                    (1933) IMA      RR      RA => RM
178: 0000 1104 0000 0000
                    (1934)          CPU     BB   RM  ALL   0      0     M     13 ;
                    (1935)                  RMRFMRDY  MREAD
179: EF00 B584 0000 0000
                    (1936)          RR      RM => RA  TR= NX
17A: EE00 1A04 0000 0000
                    (1937)          CPU     RF   NOP  ALL   0      0     M     13 ;
                    (1938)                  RM280     MWRITE       JAMF
17B: 1300 B4B0 0000 0000
                    (1939) MPY      CPU     0    RCM  ALL   0      0     M     0 ;
                    (1940)                  RMMRDY    MREAD LOADRSC ;
                    (1941)                  DATA -15
17C: 1300 078E 0003 FEF1
                    (1942)          CPU     RFRS 5    NX    6      0     M     RA ;
                    (1943)                  RY240     LINK
```

```
17D: 5660 1344 0000 0000
                    (1944)          RR      RY => RB
17E: E800 2904 0000 0000
                    (1945)          RR      RCM = 0  => RA
17F: F000 1904 0002 0100
                    (1946)          CPU     ALRS RM   NX    ADD    0     M     RA ;
                    (1947)                  RF240     LINK  ,EAC MPYLOGIC
180: CE60 1B44 0008 C000
                    (1948)          CPU     RFRS 0    ALL   6      0     M     RB ;
                    (1949)                  RF200     LINK  INCRSC ;
                    (1950)                  JUMP ON RSCNEM1 TO *-1
181: 4360 2A45 0004 9180
                    (1951)          CPU     AL   RM   NX    SUB    1     M     RA ;
                    (1952)                  RF240     AOVFL JAMF   EAC MPYLOGIC
182: 8E94 1B70 0008 C000
                    (1953) GENB1    RR      ,,JAMF    JUMP ON RSC11 TO *+1
183: 0200 0000 0005 A184
                    (1954)          ALU     VSC AND RCM = $FF => RM  TR= ALL
184: 931C 6204 0002 01FF
                    (1955) SCA      RR      RM => RA  JAMF ;
                    (1956)                  JUMP ON RS15 TO *+1
185: EE00 1A0C 0005 0186
                    (1957) INK      ALU     RA OR KEYS => RA       DISABLERHBB GO TO F1
186: 864C 1A0B 0004 0000
                    (1958) RMC      RR      ,,RESTJAMF    MODAL NOP (TRIGNVKEYS,MCHK)
187: 0200 0002 0008 0404
          000610    (1959) DBL      EQU     *
                    (1960) SGL      RR   ,,JAMF     MODAL NOP (TRIGVKEYS,DP)
188: 0200 0000 0008 0140
          000611    (1961) E16S     EQU     *
          000611    (1962) E32S     EQU     *
          000611    (1963) E32R     EQU     *
                    (1964) E64R     RR      ,,JAMF    MODAL NOP (TRIGVKEYS,AM1,AM2)
189: 0200 000C 0008 010C
                    (1965) INA2     CPU     AL   RM   NX    OR     L     RSC   0 ;
                    (1966)                  RF200     0     NOP ;
                    (1967)                  GO TO CAS5
18A: 8E4F 0A04 0004 01E5
```

```
                    (1968) CEA     CPU      BB   RCM  NONE  ,,,,280       ,PUSHBD ;
                    (1969)                  DATA CEA3
18B: F000 060C 0000 0328
                    (1970)         RR       RA => RM , GO TO F6
18C: 0200 1204 0004 001C
          000615    (1971) EMCM    EQU      *
                    (1972) LMCM    RR       ,,,JUMP ON RXM TO RXM
18D: 0200 0004 0007 4068
                    (1973)         RR       ,,JAMF MODAL NOP (TRIGNVKEYS,PARIM)
18E: 0200 0000 0008 0402
                    (1974) IAB2    RR       RY => RB  TR= NX        ,GO TO LDA+1
18F: EA00 2A04 0004 01DF
          000620    (1975) ENB     EQU      *
                    (1976) INH     RR       ,,,JUMP ON RXM TO RXM
190: 0200 0004 0007 4068
                    (1977)         RR       ,,JAMF    MODAL NOP (TRIGNVKEYS,EINTM)
191: 0200 000C 0008 041C
                    (1978) OTK     RR       RA => , TR= NX  C= BD01       , ;
                    (1979)                  EAC PLOADVKEYS TRIGVKEYS
192: 0200 1034 000A 4100
                    (1980)         ALU      RA AND RCM = $FF => RY  TR= ALL
193: 931C 1304 0002 01FF
                    (1981)         ALU      VSC AND RCM = $FF00 => VSC
194: 921C 6A04 0003 FF00
                    (1982)         ALU      VSC OR  RY  => VSC     JAMF
195: 8A4C 6A00 0000 0000
                    (1983) *
                    (1984) *       NON-VISIBLE KEYS ARE TRIGGERED ON CAI BECAUSE THIS
                    (1985) * INHIBITS INTERRUPTS FOR THE FOLLOWING INSTRUCTION.
                    (1986) * (JST CONCATINATION ALSO PERFORMS THE SAME FUNCTION.
                    (1987) *
          000626    (1988) ESIM    EQU      *
                    (1989) EVIM    RR       ,,,JUMP ON RXM TO RXM
196: 0200 0004 0007 4068
                    (1990)         RR       ,,JAMF    MODAL NOP (TRIGNVKEYS,VIM)
197: 0200 0000 0008 0408
                    (1991) SVC     RR       RCM = '65 => RY
198: F200 0804 0002 0135
```

```
                    (1992)          CPU     ,,NONE,,,,,RMMRDY,AREAD,,GO TO AVECT+1
199: 1000 07C4 0004 0078
                    (1993) ISI      RR      SI => RA  TR= NX
19A: E200 1A04 0000 0000
                    (1994)          ALU     RA AND RCM = $F => RA  TR= 0  RESTJAMF
19B: 901C 1A02 0002 010F
                    (1995) OSI      RR      ,,,JUMP ON RXM TO RXM
19C: 0200 0004 0007 4068
                    (1996)          RR      RA => , TR= NX  JAMF   EAC LOADSERIALINT
19D: 0200 100C 0008 4000
                    (1997) CRL      RR      RCM = 0 => RA   TR= NX
19E: F200 1A04 0002 0100
                    (1998) CRB      RR      RCM = 0 => RB   TR= NX JAMF
19F: F200 2A00 0002 0100
                    (1999) CHS      ALU     RA XOR RCM = $8000 => RA      JAMF
1A0: 926C 1A00 0001 0100
                    (2000) SSP      ALU     RA AND RCM = $7FFF => RA      JAMF
1A1: 921C 1A0C 0000 FFFF
          000642    (2001) IAB      EQU     *
                    (2002) XCA      RR      RA => RY  TR= NX  , JUMP ON RS16 TO XCB
1A2: 0200 1804 0005 21A8
                    (2003)          RR      RY => RB  TR= NX ,GO TO CRA
1A3: EA00 2A04 0004 01CE
                    (2004) SOA      ALU     RA MINUS RCM = 1  => RA  C= AOVFL  JAMF
1A4: 9294 1A70 0002 0001
                    (2005) IRX      CPU     AL   ,,INC 1    XM     0     RF200 ,SETCC
1A5: 9205 0A07 0000 0000
                    (2006)          RR      ,,JAMF    JUMP ON EQ TO CAS5
1A6: 0200 0000 0006 41E5
                    (2007) RCB      RR      RCM = 0 ,C= BD01       JAMF
1A7: F200 0030 0002 0100
                    (2008) XCB      RR      RB => RM  TR= NX  , JUMP ON RS16 TO IAB2
1A8: 0200 2204 0005 218F
                    (2009)          RR      RM => RA  TR= NX       ,GO TO CRB
1A9: EE00 1A04 0004 019F
                    (2010) DRX      CPU     AL   ,,DEC 0    XM     0     RF200 ,SETCC ;
                    (2011)                  GO TO IRX+1
1AA: 92F1 0A07 0004 01A6
```

```
                    (2012) CAZ     CPU      RF   ,,RF 0     M      RA     RF200 ;
                    (2013)                   ,SETCC    GO TO CAS5-2
1AB: 1200 1A07 0004 01E3
                    (2014) CSA     ALU      RA AND RCM = $7FFF  => RA     C= RF01 JAMF
1AC: 921C 1A50 0000 FFFF
                    (2015) A2A     CPU      AL   RCM  NX    ADD    0      M        RA ;
                    (2016)                  RF240     AOVFL JAMF   DATA 2
1AD: 9260 1B70 0002 0002
                    (2017) S2A     CPU      AL   RCM  NX    SUB    1      M        RA ;
                    (2018)                  RF240     AOVFL JAMF   DATA 2
1AE: 9294 1B70 0002 0002
                    (2019) CMA     ALU      NOT RA => RA    JAMF
1AF: 92FC 1A0C 0000 0000
                    (202C) TCA     ALU      NOT RA => RA    ,GO TO AOA
1B0: 92FC 1A04 0004 0133
                    (2021) SSM     ALU      RA OR RCM = $8000  => RA     JAMF
1B1: 924C 1A00 0001 0100
                    (2022) SCB     RR       RCM = $8000 => RY      C= BD01     JAMF
1B2: F200 0830 0001 0100
                    (2023) CAR     ALU      RA AND RCM = $FF00  => RA    JAMF
1B3: 921C 1A00 0003 FF00
                    (2024) CAL     ALU      RA AND RCM = $00FF  => RA    JAMF
1B4: 921C 1A00 0002 01FF
                    (2025) ICL     ALU      RA AND RCM = $FF00  => RA    FLIP BYTES JAMF
1B5: 721C 1A00 0003 FF00
                    (2026) ICR     ALU      RA AND RCM = $00FF  => RA    FLIP BYTES JAMF
1B6: 721C 1A00 0002 01FF
                    (2027) STX     CPU      RF   NOP  ALL    0      0      XM     0 ;
                    (2028)                  RM280   MWRITE  JAMF
1B7: 1301 04B0 0000 0000
                    (2029) ACA     ALU      INC RA + CBIT => RA    C= AOVFL    JAMF
1B8: 9208 1A70 0000 0000
                    (2030) ICA     ALU      INC RA + L  => RA      FLIP BYTES  JAMF
1B9: 720C 1A00 0000 0000
                    (2031) *
                    (2032) *       LOGIC COMMON ENTRY.  ALL LOGICIZE INSTRUCTIONS BEGIN HERE.
                    (2033) *
                    (2034) LOGIC   CPU      BB   RM   NX    DEC    1      M      RA ;
```

```
                    (2035)                          280  ,SETCC     S TRUE 16WAYS TO $1C0
1BA: EEF4 1607 000C 01C2
                    (2036) LRL     CPU      RFRS 5     ALL   6       ,M     RA     RF200 ;
                    (2037)                  LINKS
1BB: 5760 1A14 0000 0000
                    (2038)         CPU      RFRS 1     NX    6       ,M     RB     RF200 ;
                    (2039)                  LINKS INCRSCF ;
                    (2040)                  JUMP ON RSCNEM1 TO *-1
1BC: 4660 2A11 0004 91BB
                    (2041) LRS     CPU      RFRS 3     ALL   6       ,M     RA     RF200 ;
                    (2042)                  LINKS
1BD: 4F60 1A14 0000 0000
                    (2043)         CPU      RFRS 0     NX    6       ,M     RB     RF200 ;
                    (2044)                  LINKS INCRSCF ;
                    (2045)                  JUMP ON RSCNEM1 TO *-1
1BE: 4260 2A11 0004 91BD
                    (2046) LRR     CPU      RFRS    1     NX    6      ,M      RB     200 ;
                    (2047)                  LINK
1BF: 4660 2044 0000 0000
                    (2048)         CPU      RFRS 1     ALL   6       ,M     RA     RF200 ;
                    (2049)                  LINKS
1C0: 4760 1A14 0000 0000
                    (2050)         CPU      RFRS 1     NX    6       ,M     RB     RF200 ;
                    (2051)                  LINKS INCRSCF ;
                    (2052)                  JUMP ON RSCNEM1 TO *-2
1C1: 4660 2A11 0004 91BF
                    (2053) ARL     CPU      RFRS    5     ALL   6      ,M      RA     RF200 ;
                    (2054)                  LINKS INCRSCF ;
                    (2055)                  JUMP ON RSCNEM1 TO *
1C2: 5760 1A11 0004 91C2
                    (2056) ARS     CPU      RFRS 3     ALL   6       ,M     RA     RF200 ;
                    (2057)                  LINKS INCRSCF ;
                    (2058)                  JUMP ON RSCNEM1 TO *
1C3: 4F60 1A11 0004 91C3
                    (2059) ARR     CPU      RFRS 2     ALL   6       ,M     RA     RF200 ;
                    (2060)                  LINKS INCRSCF ;
                    (2061)                  JUMP ON RSCNEM1 TO *
1C4: 4B60 1A11 0004 91C4
```

```
                    (2062) LLR      CPU       RFLS 3     NX     7        ,M     RA     200 ;
                    (2063)                    LINK
1C5: 2E70 1044 0000 0000
                    (2064)          CPU       RFLS 3     ALL    7        ,M     RB     RF200 ;
                    (2065)                    RF01
1C6: 2F70 2A54 0000 0000
                    (2066)          CPU       RFLS 3     NX     7        ,M     RA     RF200 ;
                    (2067)                    RF01 INCRSCF ;
                    (2068)                    JUMP ON RSCNEM1 TO *-2
1C7: 2E70 1A51 0004 91C5
                    (2069)          EJCT
```

```
                    (2070)          ORG     $1C8
                    (2071) LLT      ALU     CON ZERO => RA  JAMF ;
                    (2072)                  JUMP ON LT TO LT
1C8: 923C 1A00 0006 61CF
                    (2073) LLE      ALU     CON ZERO => RA  JAMF ;
                    (2074)                  JUMP ON LE TO LT
1C9: 923C 1A00 0004 71CF
                    (2075) LNE      ALU     CON ZERO => RA  JAMF ;
                    (2076)                  JUMP ON NE TO LT
1CA: 923C 1A00 0004 41CF
                    (2077) LEQ      ALU     CON ZERO => RA  JAMF ;
                    (2078)                  JUMP ON EQ TO LT
1CB: 923C 1A00 0006 41CF
                    (2079) LGE      ALU     CON ZERO => RA  JAMF ;
                    (2080)                  JUMP ON GE TO LT
1CC: 923C 1A00 0004 61CF
                    (2081) LGT      ALU     CON ZERO => RA  JAMF ;
                    (2082)                  JUMP ON GT TO LT
1CD: 923C 1A00 0006 71CF
          000716    (2083) LF       EQU     *
                    (2084) CRA      RR      RCM = 0  => RA  TR= NX JAMF
1CE: F200 1A0C 0002 0100
                    (2085) LT       RR      RCM = 1  => RA  TR= NX JAMF
1CF: F200 1A00 0002 0001
                    (2086) JMP      RR      RY => RP  TR= NX   JAMF
1D0: EA00 7A00 0000 0000
                    (2087) LLL      CPU     RFLS 7    ALL   7      ,M    RB    RF200 ;
                    (2088)                  RF01
1D1: 3F70 2A54 0000 0000
                    (2089)          CPU     RFLS 3    NX    7      ,M    RA    RF200 ;
                    (2090)                  RF01 INCRSCF ;
                    (2091)                  JUMP ON RSCNEM1 TO *-1
1D2: 2E70 1A51 0004 91D1
                    (2092) LLS      RR      RCM = 0  => ,C= BD01
1D3: F200 0034 0002 0100
                    (2093)          CPU     RFLS 7    NX    3      ,M    RB    RF200 ;
                    (2094)                  LINK
1D4: 3E30 2A44 0000 0000
```

```
                    (2095)          CPU     RFLS 3    ALL   6       ,M    RA    RF200 ;
                    (2096)                  SOVFL     INCRSCF ;
                    (2097)                  JUMP ON RSCNEM1 TO *-1
1D5: 2F60 1A61 0004 91D4
                    (2098) ALL      CPU     RFLS 7    ALL   7       ,M    RA    RF200 ;
                    (2099)                  RF01 INCRSCF ;
                    (2100)                  JUMP ON RSCNEM1 TO *
1D6: 3F70 1A51 0004 91D6
                    (2101) ALS      RR      RCM = 0   => ,C= BD01
1D7: F200 0034 0002 0100
                    (2102)          CPU     RFLS 7    ALL   7       ,M    RA    RF200 ;
                    (2103)                  SOVFL     INCRSCF ;
                    (2104)                  JUMP ON RSCNEM1 TO *
1D8: 3F70 1A61 0004 91D8
                    (2105) ALR      CPU     RFLS 1    ALL   8       ,M    RA    RF200 ;
                    (2106)                  RF01 INCRSCF ;
                    (2107)                  JUMP ON RSCNEM1 TO *
1D9: 2780 1A51 0004 91D9
                    (2108)          EJCT
```

```
                    (2109) SKP     CPU      BB   RCM  NX     RF      0      M      RA ;
                    (2110)                  RY200     ,SETCC         DATA #131620
1DA: F200 1807 0001 6790
                    (2111) *
                    (2112) *       THE SKIP INSTRUCTION MUST READ THE CONTROL PANEL
                    (2113) * SENSE SWITCHES.  THIS MEANS THE I/O COMMAND MUST BE
                    (2114) * FORMED IN RY AND THE SIGNALS GENERATED.  THE SKIP HARD-
                    (2115) * WARE WILL WORK CORRECTLY ONLY IF RA IS SELECTED FROM THE
                    (2116) * REGISTER FILES, THE CONDITION CODES HAVE BEEN PRESET TO THE
                    (2117) * CONTENTS OF RA, AND BD HAS THE SENSE SWITCHES ON IT.  IF THE
                    (2118) * ABOVE IS TRUE IN ONE CYCLE, THEN FSKIP IS DEFINED TO HAVE
                    (2119) * THE CORRECT SENSE FOR TESTING ON THE NEXT CYCLE.
                    (2120) *
                    (2121) *
                    (2122)         CPU      BB   RCM  NX     ,,RY   PIO    200 ;
                    (2123)                  ,LOADRSC  DATA 1
1DB: F202 100E 0002 0001
                    (2124)         CPU      BB   BPD  NX     RF     0      RSC    0 280
1DC: FA03 0604 0000 0000
                    (2125)         RR       RA => ,   JAMF ;
                    (2126)                  JUMP ON SKIP TO CAS5
1DD: 0200 1000 0004 81E5
                    (2127) LDA     CPU      ,,ALL,,,,,RMMRDY   MREAD , ;
                    (2128)                  JUMP ON DP TO LDA3
1DE: 1300 0784 0005 C160
                    (2129)         RR       RM => RA  TR= NX       JAMF
1DF: EE00 1A00 0000 0000
                    (2130) CAS     CPU      AL   KEYS ALL   BB     L      M      13 ;
                    (2131)                  RMRFMRDY  MREAD
1E0: 875C B584 0000 0000
                    (2132)         ALU      RA MINUS RM => NULL  C= AOVFL  SETCC
1E1: 8E94 1077 0000 0000
                    (2133)         RR       13 => NULL  NOP  C= RF01 ;
                    (2134)                  JUMP ON FCBIT TO CAS6
1E2: 0200 B054 0004 F1F1
                    (2135)         RR       ,,JAMF    JUMP ON LE TO CAS4
1E3: 0200 0000 0004 71E4
                    (2136) CAS4    ALU      INC RP => RP  JAMF ;
```

```
                    (2137)                      JUMP ON NE TO CAS5
1E4: 9204 7A00 0004 41E5
                    (2138) CAS5     ALU         INC RP => RP TR= ALL  NOP   GO TO F1
1E5: 9304 7A04 0004 000C
                    (2139) IRS      CPU         ,,ALL,,,,,RMMRDY MREAD
1E6: 1300 0784 0000 0000
                    (2140)          ALU         INC RM => RM  SETCC
1E7: 8E65 1207 0000 000C
                    (2141)          CPU         ,,ALL,,,,,280   MWRITE JAMF  JUMP ON EQ TO CAS5
1E8: 1300 06B0 0006 41E5
                    (2142) ANA      CPU         ,,ALL,,,,,RMMRDY   MREAD
1E9: 1300 0784 0000 000C
                    (2143)          ALU         RA AND RM => RA JAMF
1EA: 8E1C 1A00 0000 0000
                    (2144) STA      CPU         RF   NOP  ALL   0      0      M      RA ;
                    (2145)                      RM280     MWRITE       JAMF ;
                    (2146)                      JUMP ON DP TO STA3
1EB: 1300 14B0 0005 C0B7
                    (2147) SKS1     CPU         ,,NX,,,RSC STROBE 200 ,,GO TO INA2
1EC: 1203 8004 0004 018A
                    (2148) ERA      CPU         ,,ALL,,,,,RMMRDY   MREAD
1ED: 1300 0784 0000 0000
                    (2149)          ALU         RA XOR RM => RA JAMF
1EE: 8E6C 1A0C 0000 0000
                    (2150) ADD      CPU         ,,ALL,,,,,RMMRDY   MREAD  ,JUMP ON DP TO ADD3
1EF: 1300 0784 0005 C0B9
                    (2151)          ALU         RA ADD RM => RA C= AOVFL     JAMF
1F0: 8E60 1A70 0000 0000
                    (2152) CAS6     RR          ,,JAMF  JUMP ON GT TO CAS4
1F1: 0200 0000 0006 71E4
                    (2153) EVMX1    RR          RM  => 11    TR= NX
1F2: EE00 9A04 0000 0000
                    (2154)          RR          13  => RM  TR= NX  ,EAC CLEARCAM
1F3: 0200 B204 0009 C000
                    (2155)          RR          11  => NULL ,MODAL  NOP  (TRIGNVKEYS,PAM)
1F4: 0200 9004 0008 0420
                    (2156)          CPU         RF   NOP  NX    0      0      M        11 ;
                    (2157)                      280  NOP  NOP   S ON TRUE BD
```

```
1F5: 1200 9604 000C 0000
                    (2158) DFLD    CPU      ,,ALL,,,,,RMMRDY,MREAD,,GO TO DFLD1
1F6: 1300 0784 0004 0500
                    (2159) DFST    CPU      RF,,ALL,,,M,FLTH,RM280,MWRITE,,GO TO DFST1
1F7: 1300 44B4 0004 0507
                    (2160) DFAD    CPU      ,,ALL,,,,,RMMRDY,MREAD,,GO TO DFAD1
1F8: 1300 0784 0004 050D
                    (2161) DFSB    CPU      ,,ALL,,,,,RMMRDY,MREAD,,GO TO DFSB1
1F9: 1300 0784 0004 0516
                    (2162) DFMP    CPU      ,,ALL,,,,,RMMRDY,MREAD,,GO TO DFMP1
1FA: 1300 0784 0004 054B
                    (2163) DFDV    CPU      ,,ALL,,,,,RMMRDY,MREAD,,GO TO DFDV1
1FB: 1300 0784 0004 0570
                    (2164) DFCS    CPU      ,,ALL,,,,,RMMRDY,MREAD,,GO TO DFCS1
1FC: 1300 0784 0004 05B4
                    (2165) CAI     RR       ,,,JUMP ON RXM TO RXM
1FD: 0200 0004 0007 4068
                    (2166)         CPU      ,,,,,RY ICAI,,,JAMF  MODAL NOP TRIGNVKEYS
1FE: 1202 8000 0008 0400
                    (2167)         EJCT
```

```
                    (2168) *      CA-G,U-CODE,MHJ,FEBRUARY 1975
                    (2169) *      FLOATING POINT PROCESSOR - PRIME COMPUTER
                    (2170) *      PRIME COMPUTER INC.,SRC0769-001
                    (2171) *      COPYRIGHT 1974,PRIME COMPUTER INC.,NATICK MASS.
                    (2172) *
                    (2173) *
                    (2174) *
                    (2175) *
          000002    (2176) P300   XSET     2
                    (2177)        CLOCK
                    (2178)        RRCLK
                    (2179)        ALUCLK
                    (2180)         ORG     $400
                    (2181) *
                    (2182) *       FLOATING POINT GENERICS DECODE TO THIS GENERAL
                    (2183) * AREA.  FLOATING SKIPS ARE FIRST.
                    (2184) *
          002000    (2185) FGEN2  EQU     *
                    (2186) FSEQ   RR      ,,JAMF   JUMP ON EQ TO CAS5
400: 0200 060C 0006 41E5
                    (2187) FSNE   RR      ,,JAMF   JUMP ON NE TO CAS5
401: 0200 0600 0004 41E5
                    (2188) FSMI   RR      ,,JAMF  JUMP ON LT TO CAS5
402: 0200 060C 0006 61E5
                    (2189) FSPL   RR      ,,JAMF  JUMP ON GE TO CAS5
403: 0200 0600 0004 61E5
                    (2190) FSLE   RR      ,,JAMF   JUMP ON LE TO CAS5
404: 0200 060C 0004 71E5
                    (2191) FSGT   RR      ,,JAMF   JUMP ON GT TO CAS5
405: 0200 0600 0006 71E5
                    (2192) FGEN1  RR      RCM = $37  => 11
406: F200 9A04 0002 0037
                    (2193)        ALU     11 AND RM  => 11  C= BD01  TR= ALL
407: 8F1C 9A34 0000 0000
                    (2194)        ALU     11 OR     RCM = FGEN2 => RM  TR= ALL
408: 934C 9204 0000 0900
                    (2195)        CPU     BB   RM   NX    0      0     0     0 ;
                    (2196)                200
```

```
409: EE00 0004 0000 0000
                    (2197)          CPU     BB   RM   NX    RF     0     M     FLTH ;
                    (2198)                  280  NOP  SETCC ;
                    (2199)                  S ON TRUE, BD
40A: EE00 4607 000C 0000
                    (2200) FAD1     CPU     BB   RCM  0     0      0     0     0 ;
                    (2201)                  280  NOP  PUSHBD  DATA  FAD4
40B: F000 060C 0000 095A
                    (2202)          ALU     INC  RY => RY  TR= ALL  ,GO TO LOAD
40C: 8B65 1304 0004 0419
                    (2203) FSB1     CPU     BB   RCM  0     0      0     0     0 ;
                    (2204)                  280  NOP  PUSHBD  DATA FSB4
40D: F000 060C 0000 095F
                    (2205)          ALU     INC RY => RY  TR= ALL  ,GO TO LOAD
40E: 8B65 1304 0004 0419
                    (2206) *
                    (2207) *        FLOATING COMPLEMENT INSTRUCTION
                    (2208) *
                    (2209)          ORG     (FGEN2 .OR. $10)
                    (2210) FCM      ALU     NOT FLTL => FLTL  ,JUMP ON FCBIT TO FLEX
410: 92FC 5B04 0004 F4CE
                    (2211)          ALU     FLTL ADD RCM = 1  => FLTL C= COUT  SETCC
411: 9260 5A27 0002 0001
                    (2212)          ALU     NOT FLTH => FLTH  TR= ALL ;
                    (2213)                  ,JUMP ON NE TO NRML
412: 93FC 4B04 0004 444B
                    (2214)          CPU     AL RCM    NX    INC    1     M     FLTH ;
                    (2215)                  RF240     AOVFL SETCC   GO TO NRML
413: 9204 4B77 0004 044B
                    (2216) FRN      ALU     FLTL ADD RCM = $80  => FLTL  C= COUT
414: 9260 5A24 0002 008C
                    (2217)          CPU     AL   RCM  NX    INC    C     M     FLTH ;
                    (2218)                  RF240     AOVFL SETCC GO TO NRML
415: 9208 4B77 0004 044B
                    (2219) FLOT1    RR      RM => FLTL  ,GO TO NRML
416: EE00 5B04 0004 044B
                    (2220) FMP1     CPU     BB   RCM  0     0      0     0     0 ;
                    (2221)                  280  NOP  PUSHBD  DATA FMP4
```

```
417: F000 060C 0000 087A
                    (2222)          ALU     INC RY => RY  TR= ALL ,GO TO LOAD
418: 8B65 1304 0004 0419
                    (2223) *
                    (2224) *
                    (2225) *        LOAD SUBROUTINE.  THIS ROUTINE ASSUMES RM CONTAINS THE
                    (2226) * FIRST WORD FROM MEMORY, RY CONTAINS THE ADDRESS OF THE SECOND
                    (2227) * WORD, AND THE STACK CONTAINS THE RETURN ADDRESS.  THE FIRST
                    (2228) * WORD IS PUT INTO 13 AS IS.  THE SECOND WORD IS BROKEN INTO
                    (2229) * THE REST OF THE FRACTION AND THE EXPONENT AND PLACED INTO
                    (2230) * RY AND 11 FOR THE EXPONENT, AND 12 AND RM FOR THE LOW
                    (2231) * ORDER PART OF THE FRACTION.
                    (2232) *
                    (2233) *
                    (2234) LOAD    CPU     BB   RM   ALL   0      0     M     13 ;
                    (2235)                 RMRFMRDY  MREAD
419: EF00 B584 0000 0000
                    (2236)          RR      RCM = $FF00  => 12
41A: F200 AA04 0003 FF00
                    (2237)          RR      RCM = $00FF  => 11
41B: F200 9A04 0002 01FF
                    (2238)          ALU     11 AND RM => (11,RY)
41C: 8E1C 9D04 0000 0000
                    (2239)          ALU     12 AND RM  => 12  CLEARFUII
41D: 8E1C AA0F 0000 000C
                    (2240)          RR      12 => RM  TR= ALL  ,S ON TRUE, POP
41E: 0300 A404 000C 0004
                    (2241)          ORG     (FGEN2 .OR. $20)
                    (2242) FLOT    RR      RCM = 128+30  => VSC  C= BD01
420: F200 6A34 0002 009E
                    (2243)          CPU     RFLS 7    NX    7      0     M     RB  RM200
421: 3E70 2204 0000 0000
                    (2244)          RR      RA => RY
422: 0200 1804 0000 0000
                    (2245)          RR      RY => FLTH , GO TO FLOT1
423: EA00 4B04 0004 0416
                    (2246) INT     ALU     VSC MINUS RCM = 127  => NULL SETCC  C= AOVFL
424: 9294 6077 0002 007F
```

```
                    (2247)          CPU     RFRS 5    NX    6      0     M     FLTL ;
                    (2248)                  RY240     NOP   NOP ;
                    (2249)                  JUMP ON LE TO CRL
425: 5660 5304 0004 719E
                    (2250)          ALU     VSC MINUS RCM = 128+30 => RM  SETCC  TR= ALL
426: 9394 6207 0002 009E
                    (2251)          CPU     RF   RM   NX    0      0     M     FLTH ;
                    (2252)                  RM200     NOP   LOADRSC  JUMP ON FCBIT TO CRL
427: 0E00 420E 0004 F19E
                    (2253) INT1     RR      RM => RA ,JUMP ON GT TO FLEX9
428: EE00 1B04 0006 74CB
                    (2254)          CPU     AL   RY   ALL   BB     L     M     RB ;
                    (2255)                  RF240     BD01  SETCC ;
                    (2256)                  JUMP ON LT TO INT2
429: 8B5C 2B37 0006 64BF
                    (2257)          RR      ,,,GO TO F1
42A: 0200 0604 0004 0000
                    (2258) FDV1     CPU     BB   RCM  0     0      0     0     0 ;
                    (2259)                  280  NOP  PUSHBD  DATA FDV4
42B: F000 060C 0000 098D
                    (2260)          ALU     INC RY => RY  ,GO TO LOAD  TR= ALL
42C: 8B65 1304 0004 0419
                    (2261) NRM7     CPU     RFRS 1    NX    6      0     M     FLTL ;
                    (2262)                  RF200     NOP   NOP
42D: 4660 5A04 0000 0000
                    (2263)          ALU     11 7 RSC + L  => RY
42E: 867C 9304 0000 0000
                    (2264)          ALU     VSC MINUS RY => VSC  C= AOVFL  ,GO TO FLEX8
42F: 8A94 6B74 0004 04CD
                    (2265)          ORG     (FGEN2 .OR. $30)
                    (2266) FRAC     ALU     VSC MINUS RCM = 128-31  => NULL SETCC C= AOVFL
430: 9294 6077 0002 0061
                    (2267)          CPU     RFRS 5    NX    6      0     M     FLTL ;
                    (2268)                  RY240     ,,JUMP ON LE TO CRL
431: 5660 5304 0004 719E
                    (2269)          ALU     VSC MINUS RCM = 128 => RM SETCC  TR= ALL
432: 9394 6207 0002 0080
                    (2270)          CPU     AL   RM   NX    SUB    1     XM    DISABLE ;
```

```
                    (2271)                  RM280     NOP   LOADRSC ;
                    (2272)                  GO TO FRAC1
433: 8E95 140E 0004 0482
                    (2273) *
                    (2274) * DOUBLE PRECISION FLOATING POINT COMPLEMENT
                    (2275) *
                    (2276) DFCM   ALU       NOT FLTL => FLTL ,JUMP ON FCBIT TO DFLEX
434: 92FC 5B04 0004 F5C8
                    (2277)        ALU       NOT RB => RB
435: 92FC 2A04 0000 000C
                    (2278)        ALU       INC RB => RB C= COUT  SETCC
436: 9204 2A27 0000 0000
                    (2279)        ALU       NOT FLTH => FLTH , JUMP ON NE TO DNRML
437: 92FC 4B04 0004 453A
                    (2280)        ALU       INC FLTL + C => FLTL C= COUT
438: 9208 5A24 0000 000C
                    (2281)        ALU       INC FLTH + C => FLTH C= AOVFL ;
                    (2282)                  SETCC GO TO DNRML
439: 9208 4B77 0004 053A
                    (2283) *
                    (2284) *
                    (2285) *       ADJUST SUBROUTINE.  THIS SUBROUTINE TAKES THE TWO NUMBERS
                    (2286) * FOUND IN THE FLOATING ACCUMULATOR AND THE RESULTS OF THE LOAD
                    (2287) * SUBROUTINE AND MAKES THE EXPONENTS THE SAME SO THEY CAN BE ADDED
                    (2288) * OR SUBTRACTED.  THE TWO NUMBERS ARE LEFT IN THE FLOATING
                    (2289) * ACCUMULATOR AND IN 12,13,AND RM FOR THE LOW ORDER HALF.
                    (2290) * THE VSC CONTAINS THE FINAL ADJUSTED EXPONENT.
                    (2291) *
                    (2292) *
                    (2293) ADJUST ALU       VSC SUB RY => RY ,JUMP ON FCBIT TO ADJ9
43A: 8A94 6304 0004 F443
                    (2294)        RR        RY => NULL  LOADRSC  S ON EQ, POP
43B: EA00 060E 000E 4004
                    (2295)        RR        RCM = 32 => YSAVE
43C: F200 CA04 0002 0020
                    (2296)        CPU       AL   RY   NX    SUB    1     XM    DISABLE ;
                    (2297)                  RY240     NOP   NOP ;
                    (2298)                  JUMP ON LE TO ADJ8
```

```
43D: 8A95 1304 0004 7444
                    (2299) *
                    (2300) * ABOVE BRANCH IS TAKEN IF THE ACCUMULATOR MUST BE RIGHT SHIFTED.
                    (2301) *
                    (2302) * NOW TEST THE SHIFT COUNTER AGAINST 31 TO SEE IF
                    (2303) * THE TWO NUMBERS ARE WITHIN RANGE, AND IF SO LOAD RSC AND
                    (2304) * UNNORMALIZE THE NEW ARGUMENT.
                    (2305) *
                    (2306) *
                    (2307)          ALU     YSAVE ADD RY => NULL  SETCC ;
                    (2308)                  C= AOVFL
43E: 8A60 C077 0000 0000
                    (2309)          RR      RY => NULL LOADRSC
43F: EA00 000E 0000 0000
                    (2310)          CPU     RFRS 3    ALL   6      0     M     13 ;
                    (2311)                  RF240     LINK  NOP ;
                    (2312)                  JUMP ON LE TO NRML
440: 4F60 BB44 0004 744B
                    (2313)          CPU     RFRS 1    NX    6      0     M     12 ;
                    (2314)                  RF240     NOP   INCRSC ;
                    (2315)                  JUMP ON RSCNEM1 TO *-1
441: 4660 AB05 0004 9440
                    (2316)          RR      12 => RM  ,S ON TRUE, POP
442: 0200 A404 000C 0004
                    (2317) *
                    (2318) * NEW ARGUMENT IS GREATER OR EQUAL TO ACC.
                    (2319) *
                    (2320) *
                    (2321) ADJ9     ALU     ZERO MINUS RY => RY ,GO TO MOVE
443: 8A95 1304 0004 0449
                    (2322) ADJ8     ALU     INC 13 + 0 => NULL SETCC
444: 9200 B007 0000 000C
                    (2323)          ALU     YSAVE SUB RY => NULL  SETCC ;
                    (2324)                  S ON EQ POP
445: 8A94 C607 000E 4004
                    (2325)          CPU     RFRS 3    ALL   6      0     M     FLTH ;
                    (2326)                  RF240     LINK  NOP ;
                    (2327)                  JUMP ON LE TO MOVE
```

```
446: 4F60 4B44 0004 7449
                    (2328)          CPU     RFRS 1    NX    6      0     M     FLTL ;
                    (2329)                  RF240     NOP   INCRSC ;
                    (2330)                  JUMP ON RSCNEM1 TO *-1
447: 4660 5B05 0004 9446
                    (2331)          ALU     VSC ADD RY  => VSC  ,S ON TRUE,POP
448: 8A60 6B04 000C 0004
                    (2332) MOVE     CPU     BB   RCM  NX    ZERO   L     M     FLTL ;
                    (2333)                  RF200     ,,DATA 0
449: F23C 5A04 0002 0100
                    (2334)          CPU     AL   NOP  NX    ZERO   L     M     FLTH ;
                    (2335)                  RF240     ,,GO TO MOVE-1
44A: 923C 4B04 0004 0448
                    (2336) *
                    (2337) *
                    (2338) *        NORMALIZE SUBROUTINE.  THIS ROUTINE TAKES THE FLOATING
                    (2339) * ACCUMULATOR AND NORMALIZES THE RESULTS, ADJUSTING THE
                    (2340) * VSC AS NEEDED.  THE ROUTINE EXPECTS THE CARRY BIT TO
                    (2341) * BE SET IF THE ACCUMULATOR IS ALREADY OVER SHIFTED.  THAT
                    (2342) * IS, THE VALUE MUST BE RIGHT SHIFTED ONE PLACE.
                    (2343) * IF THE CBIT CAN BE SET, THEN THE CONDITION CODE MUST ALSO
                    (2344) * BE SET TO REFLECT THE VALUE OF THE FACCUM.
                    (2345) * IF OVERFLOW OCCURRED, THE CONDITION CODE WILL IN FACT SHOW
                    (2346) * THE REVERSE OF THE CORRECT SIGN. NRML DEPENDS ON THIS.
                    (2347) * THE ROUTINE THEN EXITS TO THE FETCH CYCLE.
                    (2348) *
                    (2349) *
                    (2350) NRML     CPU     RFLS 3    ALL   7      0     M     FLTH ;
                    (2351)                  RY240  SOVFL      ,JUMP ON FCBIT TO NRM10
44B: 2F70 4364 0004 F452
                    (2352)          ALU     CON 0 => RY C= BD01 ,JUMP ON FCBIT TO F1
44C: 923C 0334 0004 F000
                    (2353)          RR      RCM = -32 => 11 LOADRSC
44D: F200 9A0E 0003 FEE0
                    (2354) NRM18    CPU     RFLS 7    ALL   7      0     M     FLTL ;
                    (2355)                  RF240     LINK  ,JUMP ON FCBIT TO NRM7
44E: 3F70 5B44 0004 F42D
                    (2356)          CPU     RFLS 3    NX    7      0     M     FLTH ;
```

```
                    (2357)                  RF240     NOP   INCRSC ;
                    (2358)                  JUMP ON RSCNEM1 TO NRM17
44F: 2E70 4B05 0004 9451
                    (2359)          RR      RCM = 0  => VSC  C= BB01  JAMF
450: F200 6A3C 0002 0100
                    (2360) NRM17    CPU     RFLS 3    ALL   7     0     M     FLTH ;
                    (2361)                  RY240     SOVFL ,GO TO NRM18
451: 2F70 4364 0004 044E
                    (2362) NRM10    ALU     VSC PLUS RCM = 1  => VSC C= AOVFL
452: 9260 6A74 0002 0001
                    (2363)          CPU     RFRS 5    ALL   6     0     M     FLTH ;
                    (2364)                  RF240     LINK  NOP ;
                    (2365)                  JUMP ON LT TO *+2
453: 5760 4B44 0006 6455
                    (2366)          ALU     FLTH OR RCM = $8000  => FLTH
454: 924C 4A04 0001 0100
                    (2367)          CPU     RFRS 1    ALL   6     0     M     FLTL ;
                    (2368)                  RF240     NOP   JAMF ;
                    (2369)                  JUMP ON FCBIT TO FLEX
455: 4760 5B00 0004 F4CE
                    (2370) *
                    (2371) *
                    (2372) *        EXECUTION OF THE FLOATING POINT MEMORY REFERENCE
                    (2373) * INSTRUCTIONS.
                    (2374) *
                    (2375) *
                    (2376) FLD1     ALU     INC RY  => RY
456: 8A65 1304 0000 0000
                    (2377)          CPU     BB   RCM  ALL   0     0     M     FLTL ;
                    (2378)                  RMRFMRDY  MREAD ,DATA $FF00
457: F300 5584 0003 FF00
                    (2379)          ALU     FLTL AND RM  => FLTL
458: 8E1C 5A04 0000 0000
                    (2380)          ALU     VSC AND RM => VSC  JAMF
459: 8E1C 6A00 0000 0000
                    (2381) FAD4     CPU     BB   RCM  0     0     0     0     0 ;
                    (2382)                  280  NOP  PUSHBD  DATA FAD6
45A: F000 060C 0000 095C
```

```
                    (2383)       ALU    VSC MINUS RY => NULL SETCC C= AOVFL  GO TO ADJUST
45B: 8A94 6677 0004 043A
                    (2384) FAD6 ALU     FLTL ADD RM => FLTL C= COUT
45C: 8E60 5A24 0000 0000
                    (2385)       RR     13 => RM
45D: 0200 B204 0000 0000
                    (2386)       ALU    FLTH ADD RM + C => FLTH  C= AOVFL SETCC ;
                    (2387)              GO TO NRML
45E: 8E68 4B77 0004 044B
                    (2388) FSB4   CPU      BB   RCM  0     0      0     0     0 ;
                    (2389)                 280  NOP  PUSHBD  DATA FSB6
45F: F000 060C 0000 0861
                    (2390)       ALU    VSC MINUS RY => NULL SETCC C= AOVFL GO TO ADJUST
460: 8A94 6677 0004 043A
                    (2391) FSB6 ALU     FLTL SUB RM => FLTL  C= COUT
461: 8E94 5A24 0000 0000
                    (2392)         RR       13 => RM
462: 0200 B204 0000 0000
                    (2393)       ALU    FLTH SUB RM  + C => FLTH C= AOVFL SETCC ;
                    (2394)              GO TO NRML
463: 8E98 4B77 0004 044B
                    (2395) FST1 ALU     VSC AND RCM = $FF00  => NULL SETCC C= MWRITE TR= ALL
464: 931C 60B7 0003 FF00
                    (2396)         ALU INC RY => RY
465: 8A65 1304 0000 0000
                    (2397)         ALU      VSC AND RCM = $00FF => RM  C= BD01
466: 921C 6234 0002 01FF
                    (2398)         RR       RM => 11
467: EE00 9A04 0000 0000
                    (2399)         ALU      FLTL AND RCM = $FF00 => RM
468: 921C 5204 0003 FF00
                    (2400)         ALU      11 OR RM  => RM  C= MWRITE  TR= ALL ;
                    (2401)             JAMF  JUMP ON NE TO FLEX1
469: 8F4C 94B0 0004 44CF
                    (2402) *
                    (2403) *            BB SELECT CHANGED FROM RM TO RY SO IT WORKS
                    (2404) *
                    (2405) FDV13  CPU      AL   RY   NX    DEC    0     M     VSC ;
```

```
                    (2406)                  RF240     AOVFL ,GO TO FCM
46A: 8AF0 6B74 0004 0410
                    (2407) *
                    (2408) *
                    (2409) *    FLOATING COMPARE.  THIS INSTRUCTION MUST COMPARE:  FIRST,
                    (2410) * SIGN, THEN EXPONENT, FINALLY UPPER THEN LOWER MAGNITUDE.
                    (2411) FCS1   ALU       INC RY => RY ,JUMP ON LT TO FCS2
46B: 8A65 1304 0006 6477
                    (2412) * SIGNS ARE EQUAL
                    (2413)        CPU       BB   RM   ALL   BB     L      M      13 ;
                    (2414)                  RMRFMRDY  MREAD  SETCC
46C: EF5C B587 0000 0000
                    (2415)        ALU       11 AND RM => RY  ,JUMP ON LT TO FCS3
46D: 8E1C 9304 0006 64C8
                    (2416) * EXPONENT TEST IS BACKWARDS FOR NEG #'S.
                    (2417) * TEST FOR ZERO IN EITHER ARGUMENT.
                    (2418)        ALU       INC FLTH + 0  => NULL  SETCC  JUMP ON EQ TO FCS7
46E: 9200 4607 0006 4479
                    (2419)        ALU       VSC MINUS RY  => NULL  SETCC C= AOVFL ;
                    (2420)                  TR= ALL  JUMP ON EQ TO FCS2+1
46F: 8B94 6677 0006 4478
                    (2421) FCS4   ALU       11  7  RM  => 11  ,JUMP ON FCBIT TO FCS2+1
470: 8E7C 9B04 0004 F478
                    (2422)        RR        RX => RX , JUMP ON GT TO F1
471: 0200 0B04 0006 7000
                    (2423) FCS5   RR        13 => RM  NOP  JUMP ON NE TO FCS2+1
472: 0200 B404 0004 4478
                    (2424) * EXPONENTS EQUAL
                    (2425)        ALU       FLTH MINUS RM => NULL SETCC
473: 8E94 4007 0000 0000
                    (2426)        RR        11 => RM  NOP  TR= ALL JUMP ON GT TO F1
474: 0300 9404 0006 7000
                    (2427)        ALU       FLTL MINUS RM => NULL C= COUT  SETCC ;
                    (2428)                  JUMP ON LT TO FCS2+1
475: 8E94 5627 0006 6478
                    (2429) * HIGH ORDER MAGNITUDES ARE EQUAL
                    (2430)        RR        ,,,JUMP ON EQ TO CAS5
476: 0200 0604 0006 41E5
```

```
                    (2431) FCS2    RR       RX => RX , JUMP ON FCBIT TO F1
477: 0200 0B04 0004 F000
                    (2432)         ALU      RP PLUS RCM = 2 => RP JAMF
478: 9260 7A0C 0002 0002
                    (2433) FCS7    RR       RX => RX  JAMF  JUMP ON EQ TO CAS5
479: 0200 0B0C 0006 41E5
                    (2434) *
                    (2435) *
                    (2436) *       FLOATING POINT MULTIPLY.  THE ALGORITHM USED IS BASICALLY THAT
                    (2437) * OF SHIFT AND ADD.  ONLY 24 ITTERATIONS ARE REQUIRED FOR FULL
                    (2438) * ACCURACY.  IN ADDITION, THE FIRST 7 ITTERATIONS ARE SINGLE
                    (2439) * PRECISION ONLY.  THE REMAINING 16 ARE FULL DOUBLE
                    (2440) * PRECISION.
                    (2441) *
                    (2442) *
                    (2443) FMP4    ALU      11 MINUS RCM = 128 => RY
47A: 9294 9304 0002 0080
                    (2444)         ALU      VSC PLUS RY => VSC C= AOVFL
47B: 8A60 6A74 0000 0000
                    (2445)         RR       FLTH => RM
47C: 0200 4204 0000 0000
                    (2446)         RR       FLTL => RY  , JUMP ON FCBIT TO  FLEX
47D: 0200 5304 0004 F4CE
                    (2447)         CPU      AL   RCM  ALL   ZERO   L     M     FLTH ;
                    (2448)                  RF200   NOP   LOADRSC  DATA -7
47E: 933C 4A0E 0003 FFF9
                    (2449)         CPU      ALFB 0    NONE  RF     0     M     12 ;
                    (2450)                  RF200
47F: 6000 AA04 0000 0000
                    (2451)         CPU      RFRS 0    NONE  6      0     M     12 ;
                    (2452)                  RF200  LINK
480: 4060 AA44 0000 0000
                    (2453) *  STD MPY FOR FIRST 7 BITS.  (SORT OF)
                    (2454)         CPU      ALRS RM   NX    ADD    0     M     FLTH ;
                    (2455)                  RF240     NOP   ,EAC MPYLOGIC
481: CE60 4B04 0008 C000
                    (2456)         CPU      RFRS 0    ALL   6      0     M     12 ;
                    (2457)                  RF240     LINKS  INCRSC ;
```

```
                    (2458)                  JUMP ON RSCNEM1 TO *-1
482: 4360 AB15 0004 9481
                    (2459) * CBIT = RF16 = LAST BIT FROM 12 (NEW LOW)
                    (2460)          CPU     AL   RCM  ALL   ZERO   L     M      FLTL ;
                    (2461)                  RF200     NOP   LOADRSC ;
                    (2462)                  DATA -16
483: 933C 5A0E 0003 FFF0
                    (2463) *  SET UP FOR FULL DOUBLE PRECISION MULTIPY FOR THE
                    (2464) * REMAINING PORTION OF THE LOOP.  (16 TIMES + SUB)
                    (2465) *
                    (2466)          RR      ,,,  JUMP ON FCBIT TO FMP5
484: 0200 0604 0004 F487
                    (2467) * SHIFT ONLY
                    (2468)          CPU     RFRS      3     ALL    6     0      M FLTH ;
                    (2469)                  RF240     LINK  NOP    60 TO FMP7
485: 4F60 4B44 0004 0489
                    (2470) FMP6     CPU     RFRS 5    ALL   6      0     M      13 ;
                    (2471)                  RF280     NOP   NOP ;
                    (2472)                  JUMP ON NOTRF16 TO *-1
486: 5760 BC04 0007 7485
                    (2473) FMP5     ALU     FLTL PLUS RY  => FLTL  C= COUT
487: 8A60 5A24 0000 0000
                    (2474)          CPU     ALRS RM   ALL   6      C     M      FLTH ;
                    (2475)                  RF240     LINK  NOP
488: CF68 4B44 0000 0000
                    (2476) FMP7     CPU     RFRS 1    ALL   6      0     M      FLTL ;
                    (2477)                  RF240     NOP   INCRSC ;
                    (2478)                  JUMP ON RSCNEM1 TO FMP6
489: 4760 5B05 0004 9486
                    (2479) *TEST FOR LAST MOVE -- ZERO CBIT
                    (2480)          CPU     RF   NOP  NX    0      0     M      13 ;
                    (2481)                  RF280     BD01  NOP ;
                    (2482)                  JUMP ON NOTRF16 TO NRML
48A: 1200 BC34 0007 744B
                    (2483) * FINAL SUBTRACT
                    (2484)          ALU     FLTL MINUS RY => FLTL    C= COUT
48B: 8A94 5A24 0000 0000
                    (2485)          ALU     FLTH MINUS RM + C => FLTH  C= AOVFL ;
```

```
                    (2486)                  SETCC  GO TO NRML
48C: 8E98 4877 0004 044B
                    (2487) *
                    (2488) *
                    (2489) *       DIVIDE INSTRUCTION.  ROUTINE EXECUTES A NON-PREFORMING
                    (2490) * STYLE DIVIDE.  BEFORE STARTING THE LOOP, THE NUMBERS ARE
                    (2491) * BOTH MADE POSITIVE.  THERE ARE 31 ITTERATIONS OF THE
                    (2492) * DIVIDE LOOP BUT ONLY 17 OF THEM ARE IN FACT FULL DOUBLE
                    (2493) * PRECISION, THE REST BEING SINGLE PRECISION.
                    (2494) *
                    (2495) *
                    (2496) *
                    (2497) FDV4    ALU      INC 13 + 0 => RM  SETCC
48D: 9200 B207 0000 0000
                    (2498)         ALU      FLTH XOR RM => NULL  SETCC ;
                    (2499)                  JUMP ON EQ TO FLEX2
48E: 8E6C 4607 0006 44D1
                    (2500) * DIVISION BY ZERO OVERFLOW EXIT.
                    (2501) * NOW MOVE EXPONENT TO RY
                    (2502)         RR       RCM = -17 => NULL LOADRSC
48F: F200 000E 0003 FEEF
                    (2503) * DO EXPONENT CALCULATION
                    (2504)         ALU      11 MINUS RCM = 129 => RY
490: 9294 9304 0002 0181
                    (2505)         ALU      VSC MINUS RY => VSC C= AOVFL ;
                    (2506)                  , JUMP ON GE TO *+2
491: 8A94 6B74 0004 6493
                    (2507) * SET FUII AS A 'COMPLEMENT RESULTS' FLAG
                    (2508)         RR       ,,,EAC SETFUII
492: 0200 0004 000A 8000
                    (2509) * SKIP ON LIKE SIGNS, SET UP FOR RM COMP TEST.
                    (2510)         ALU      INC RM + 0 => NULL  SETCC ;
                    (2511)                  JUMP ON FCBIT TO FLEX
493: 8E61 1607 0004 F4CE
                    (2512)         RR       12 => RY , JUMP ON GE TO FDV5
494: 0200 A304 0004 6498
                    (2513) * MUST TWO'S COMPLEMENT RM[RY .
                    (2514)         ALU      ZERO MINUS RY => NULL C= COUT
```

```
495: 8A95 1024 0000 0000
                    (2515)          ALU     ZERO MINUS RM + C => RM  SETCC
496: 8E99 1207 0000 0000
                    (2516)          ALU     ZERO MINUS RY => RY ,JUMP ON LT TO FDV13
497: 8A95 1304 0006 646A
                    (2517) * ABOVE EXIT IS FOR -(1/2)**N---COMPLEMENT THE ACCUMULATOR
                    (2518) * AND EXIT
                    (2519) * TEST ACC FOR NEG
                    (2520) FDV5     ALU     INC FLTH + 0 => NULL SETCC
498: 9200 4007 0000 0000
                    (2521)          ALU     CON 0 => 12 , JUMP ON GE TO FDV6
499: 923C AB04 0004 64AA
                    (2522) * COMPLEMENT ACC
                    (2523)          ALU     NOT FLTL => FLTL
49A: 92FC 5A04 0000 0000
                    (2524)          ALU     INC FLTL => FLTL SETCC
49B: 9204 5A07 0000 0000
                    (2525)          ALU     NOT FLTH => FLTH , JUMP ON NE TO FDV6
49C: 92FC 4B04 0004 44AA
                    (2526)          ALU     INC FLTH => FLTH SETCC
49D: 9204 4A07 0000 0000
                    (2527)          RR      RX => RX , JUMP ON GE TO FDV6
49E: 0200 0B04 0004 64AA
                    (2528)          ALU     VSC PLUS RCM = 1 => VSC  C= AOVFL
49F: 9260 6A74 0002 0001
                    (2529)          CPU     RFRS 5    ALL   6     0     M     FLTH ;
                    (2530)                  RF240     ,,JUMP ON FCBIT TO FLEX
4A0: 5760 4B04 0004 F4CE
                    (2531)          RR      RX => RX , GO TO FDV6
4A1: 0200 0B04 0004 04AA
                    (2532) * FINNALLY DOUBLE PRECISION DIVIDE LOOP CAN BE EXECUTED.
                    (2533) *
                    (2534) * TEST FOR SUB OR SHIFT AND SHIFT IN
                    (2535) * A QUOTIENT BIT FROM THE LAST ITTERATION.
                    (2536) *
                    (2537) FDV10    CPU     RFLS 3    NX    3     0     M     13 ;
                    (2538)                  RF240     NOP   INCRSC ;
                    (2539)                  JUMP ON LT TO FDV7
```

```
4A2: 2E30 BB05 0006 64A5
                    (2540) * IF SHIFT, GO TO SHIFT.  THEN, TEST LOW ORDER HALF.
                    (2541) * ABORT LOW TEST IF HIGH TEST WAS CONCLUSIVE.
                    (2542)          ALU     FLTL MINUS RY => NULL  , ;
                    (2543)                  C= COUT  JUMP ON GT TO FDV8
4A3: 8A94 5624 0006 74A7
                    (2544) * LOW TEST REQUIRED - SHIFT ON LT
                    (2545)          RR      ,,,JUMP ON FCBIT TO FDV8
4A4: 0200 0604 0004 F4A7
                    (2546) * SHIFT STEP PROCESSING -- NO SUBTRACT
                    (2547) *
                    (2548) FDV7     CPU     RFLS 7    NX    7     0     M     FLTL ;
                    (2549)                  RF200     LINK  NOP
4A5: 3E70 5A44 0000 0000
                    (2550)          CPU     RFLS 3    ALL   7     0     M     FLTH ;
                    (2551)                  RF240     LINK NOP  GO TO FDV6
4A6: 2F70 4B44 0004 04AA
                    (2552) * PERFORM SUBTRACT AND SHIFT
                    (2553) FDV8     ALU     FLTL MINUS RY => FLTL  C= COUT
4A7: 8A94 5A24 0000 0000
                    (2554)          CPU     RFLS 7    ALL   7     0     M     FLTL ;
                    (2555)                  RF200     LINK  NOP
4A8: 3F70 5A44 0000 0000
                    (2556)          CPU     ALLS RM   ALL   SUB   C     M     FLTH ;
                    (2557)                  RF240     LINK
4A9: AF98 4B44 0000 0000
                    (2558) FDV6     ALU     FLTH MINUS RM => NULL SETCC ;
                    (2559)                  JUMP ON RSCNEM1 TO FDV10
4AA: 8E94 4607 0004 94A2
                    (2560)          CPU     RFLS 3    NX    3     0     M     FLTL ;
                    (2561)                  RF200
4AB: 2E30 5A04 0000 0000
                    (2562) *DOUBLE PRECISION HALF FINISHED.  NOW DO SINGLE PRECISIONPART.
                    (2563) FDV11    RR      RCM = -15  => NULL  LOADRSC
4AC: F200 000E 0003 FEF1
                    (2564)          CPU     AL   RCM  NX    AND   L     M     13 ;
                    (2565)                  RY240     NOP   SETCC DATA $7FFF
4AD: 921C B307 000C FFFF
```

```
                    (2566)          CPU     ALLS RM   ALL   SUB    1     M     FLTH ;
                    (2567)                  RF280     LINK  ,EAC DIVLOGIC
4AE: AF94 4C44 0009 0000
                    (2568)          CPU     RFLS 3    ALL   7      0     M     FLTL ;
                    (2569)                  RF240     LINK  INCRSC ;
                    (2570)                  JUMP ON RSCNEM1 TO *-1
4AF: 2F70 5B45 0004 94AE
                    (2571) *RESULTS OF DIVIDE NOW IN RY AND FLTL
                    (2572)          RR      RY => FLTH C= BD01 , JUMP ON FUII TO FCM
4B0: EA00 4B34 0005 5410
                    (2573)          CPU     RFLS 3    ALL   7      0     M     FLTH ;
                    (2574)                  RY240     SOVFL  ,GO TO NRML+1
4B1: 2F70 4364 0004 044C
                    (2575) * PATCH SPACE
                    (2576) FRAC1    RR      RY => RB  TR= ALL  C= BD01
4B2: EB00 2A34 0000 000C
                    (2577)          RR      FLTH => RY  ,JUMP ON LE TO FRAC3
4B3: 0200 4304 0004 74EC
                    (2578)          RR      RY => RA
4B4: EA00 1A04 000C 000C
                    (2579)          ALU     VSC MINUS RCM = 128+31  => NULL ;
                    (2580)                  SETCC  TR= ALL
4B5: 9394 6007 0002 019F
                    (2581)          RR      RM => NULL LOADRSC  JUMP ON GT TO CRL
4B6: EE00 060E 0006 719E
                    (2582)          CPU     RFLS 7    ALL   3      0     M     RB ;
                    (2583)                  RF200     LINK
4B7: 3F30 2A44 0000 0000
                    (2584)          CPU     RFLS 3    NX    3      0     M     RA ;
                    (2585)                  RF240     NOP   INCRSC ;
                    (2586)                  JUMP ON RSCNEM1 TO *-1
4B8: 2E30 1B05 0004 94B7
                    (2587) FRAC2    ALU     RA XOR RCM = $8000  => NULL  SETCC
4B9: 926C 1007 0001 0100
                    (2588)          ALU     INC RB + 0 => NULL  SETCC ;
                    (2589)                  JUMP ON NE TO F1
4BA: 9200 2607 0004 4000
                    (2590)          RR      RX => RX JAMF  JUMP ON EQ TO CRA
```

```
4BB: 0200 0B0C 0006 41CE
                    (2591) FRAC3  CPU     BB   RY   ALL   ZERO   L     M     RA ;
                    (2592)                RF240      NOP   SETCC ;
                    (2593)                JUMP ON EQ TO FRAC2
4BC: EB3C 1B07 0006 44B9
                    (2594)        CPU     RFRS 3     ALL   6      0     M     RA ;
                    (2595)                RF200      LINK
4BD: 4F60 1A44 0000 0000
                    (2596)        CPU     RFRS 0     NX    6      0     M     RB ;
                    (2597)                RF240      NOP   INCRSCF ;
                    (2598)                JUMP ON RSCNEM1 TO *-1
4BE: 4260 2B01 0004 94BD
                    (2599) INT2   CPU     RFRS 3     ALL   6      0     M     RA ;
                    (2600)                RF240      LINK  ,JUMP ON FCBIT TO INT6
4BF: 4F60 1B44 0004 F4C7
                    (2601)        CPU     RFRS 0     NX    6      0     M     RB ;
                    (2602)                RF240      LINKS  INCRSC ;
                    (2603)                JUMP ON RSCNEM1 TO *-1
4C0: 4260 2B15 0004 94BF
                    (2604)        ALU     CON ZERO  => NULL  C= BD01  ,JUMP ON FCBIT TO INT4
4C1: 923C 0634 0004 F4C6
                    (2605) INT5   RR      ,,JAMF  JUMP ON LT TO *+1
4C2: 0200 0600 0006 64C3
                    (2606)        ALU     RB PLUS RCM = 1  => RB  C= AOVFL
4C3: 9260 2A74 0002 0001
                    (2607)        ALU     INC RA + C => RA
4C4: 9208 1A04 0000 0000
                    (2608)        ALU     RB AND RCM = $7FFF  => RB C= BD01  JAMF
4C5: 921C 2A30 0000 FFFF
                    (2609) INT4   ALU     INC RA + 0 => NULL SETCC  GO TO INT5
4C6: 9200 1607 0004 04C2
                    (2610) INT6   ALU     INC RA + 0 => NULL SETCC GO TO INT2+1
4C7: 9200 1607 0004 04C0
                    (2611) FCS3   ALU     VSC MINUS RY => NULL C= AOVFL SETCC
4C8: 8A94 6077 0000 0000
                    (2612)        ALU     11  7 RM => 11 , JUMP ON FCBIT TO F1
4C9: 8E7C 9B04 0004 F000
                    (2613)        RR      RX => RX JAMF JUMP ON GE TO FCS5
```

```
4CA: 0200 0B00 0004 6472
                    (2614) * FLOATING POINT EXCEPTION VECTOR.  ENTRY POINTS ARE:
                    (2615) *       FLEX    OVERFLOW
                    (2616) *       FLEX1   STORE EXCEPTION
                    (2617) *       FLEX2   DIVIDE BY ZERO
                    (2618) *       FLEX9   INT EXCEPTION
                    (2619) *
                    (2620) * REGISTER 11 IS USED TO SHOW THE TYPE OF EXCEPTION:
                    (2621) *       $100    OVERFLOW
                    (2622) *       $101    DIVIDE BY ZERO
                    (2623) *       $102    STORE EXCEPTION   REGISTER 12 = EFFECTIVE ADDRESS
                    (2624) *       $103    INT EXCEPTION
                    (2625) *
                    (2626) * IF 74 IS ZERO, THE VECTOR ABORTS, EXECUTING THE NEXT
                    (2627) * SEQUENTIAL INSTRUCTION WITH THE CBIT SET.
                    (2628) * . THE ABSOLUTE MAPPED VECTOR HANDLING
                    (2629) * IS IDENTICAL TO THAT OF THE UII.
                    (2630) *
                    (2631) *
                    (2632) FLEX9  RR      RCM = $FFFC C= BD01 TR= ALL => 11
4CB: F300 9A34 0003 FFFC
                    (2633)        RR      ,,,GO TO FLEX4
4CC: 0200 0604 0004 04D2
                    (2634) FLEX8  RR      ,,JAMF JUMP ON FCBIT TO FLEX
4CD: 0200 0600 0004 F4CE
                    (2635) FLEX   ALU     CON  MINUS1 => 11  C= BD01 , GO TO FLEX4
4CE: 92CC 9B34 0004 04D2
                    (2636) FLEX1  RR      RCM = $FFFD  => 11  C= BD01
4CF: F200 9A34 0003 FEFD
                    (2637)        RR      RY => 12 , GO TO FLEX4
4D0: EA00 A804 0004 04D2
                    (2638) FLEX2  RR      RCM = $FFFE => 11 C= BD01
4D1: F200 9A34 0003 FEFE
                    (2639) FLEX4  ALU     11 XOR RCM = $FEFF => 11
4D2: 926C 9A04 0001 FDFF
                    (2640) FLEX5  RR      RCM = '74 => RY
4D3: F200 0804 0002 013C
                    (2641)        CPU     AL   NOP  ALL   DEC    0     M     EAS ;
```

```
                    (2642)                  RMRFMRDY  AREAD ,EAC CLRNVKEYS TRIGNVKEYS
4D4: 93F0 A5C4 000A 0400
                    (2643)          CPU     AL   RM   NX    BB     L     0     0 ;
                    (2644)                  RY240     NOP   SETCC
4D5: 8E5C 0307 000C 0000
                    (2645)          CPU     AL   RY   NX    INC    0     M     RP ;
                    (2646)                  RM280     NOP   JAMF ;
                    (2647)                  JUMP ON NE TO JST1
4D6: 8A00 7400 0004 4150
                    (2648) *
                    (2649) * DOUBLE PRECISION  EXECUTION SPACE.  FOR EASE OF DEBUG,
                    (2650) * EXECUTION BEGINS AT $500.
                    (2651) *
                    (2652)          ORG     $500
                    (2653) DFLD1    ALU     INC RY => RY
500: 8A65 1304 0000 0000
                    (2654)          CPU     BB   RM   ALL   0     0     M     FLTH ;
                    (2655)                  RMRFMRDY  MREAD
501: EF00 4584 0000 0000
                    (2656)          ALU     INC RY => RY
502: 8A65 1304 0000 0000
                    (2657)          CPU     BB   RM   ALL   0     0     M     FLTL ;
                    (2658)                  RMRFMRDY  MREAD
503: EF00 5584 0000 0000
                    (2659)          ALU     INC RY => RY
504: 8A65 1304 0000 0000
                    (2660)          CPU     BB   RM   ALL   0     0     M     RB ;
                    (2661)                  RMRFMRDY  MREAD
505: EF00 2584 0000 000C
                    (2662)          RR      RM => VSC ,  GO TO F1
506: EE00 6B04 0004 0000
                    (2663) DFST1    ALU INC RY => RY
507: 8A65 1304 0000 0000
                    (2664)          CPU     RF,,ALL,,,M,FLTL,RM280,MWRITE
508: 1300 54B4 0000 0000
                    (2665)          ALU     INC RY => RY
509: 8A65 1304 0000 0000
                    (2666)          CPU     RF,,ALL,,,M,RB,RM280,MWRITE
```

```
50A: 1300 24B4 0000 0000
                    (2667)          ALU     INC RY => RY
50B: 8A65 1304 0000 0000
                    (2668)          CPU     RF,,ALL,,,M,VSC,RM280,MWRITE,JAMF
50C: 1300 64B0 0000 0000
                    (2669) DFAD1    CPU     BB   RCM  NONE  0      0      0      0 ;
                    (2670)                  280  ,PUSHBD  DATA DFAD4
50D: F000 060C 0002 0B0F
                    (2671)          ALU     INC RY => RY ,GO TO DGET
50E: 8A65 1304 0004 051F
                    (2672) DFAD4    CPU     BB   RCM  NONE  0      0      0      0 ;
                    (2673)                  280  ,PUSHBD  DATA DFAD6
50F: F000 060C 0002 0B11
                    (2674)          ALU     VSC MINUS RM => RM SETCC C= AOVFL ;
                    (2675)                  GO TO DFIX
510: 8E94 6477 0004 0524
                    (2676) DFAD6    ALU     RB PLUS RY => RB C= COUT
511: 8A60 2A24 0000 0000
                    (2677)          RR      12 => RM
512: 0200 A204 0000 000C
                    (2678)          ALU     FLTL PLUS RM + C  => FLTL  C= COUT  TR= ALL
513: 8F68 5A24 0000 0000
                    (2679)          RR      13 => RM
514: 0200 B204 0000 000C
                    (2680)          ALU     FLTH PLUS RM + C  => FLTH  C= AOVFL ;
                    (2681)                  SETCC  GO TO DNRML
515: 8E68 4B77 0004 053A
                    (2682) DFSB1    CPU     BB   RCM  NONE  0      0      0      0 ;
                    (2683)                  280  ,PUSHBD  DATA DFSB4
516: F000 060C 0002 0B18
                    (2684)          ALU     INC RY => RY ,GO TO DGET
517: 8A65 1304 0004 051F
                    (2685) DFSB4    CPU     BB   RCM  NONE  0      0      0      0 ;
                    (2686)                  280  ,PUSHBD  DATA DFSB6
518: F000 060C 0002 0A1A
                    (2687)          ALU     VSC MINUS RM => RM C= AOVFL ;
                    (2688)                  SETCC     GO TO DFIX
519: 8E94 6477 0004 0524
```

```
                    (2689) DFSB6  ALU     RB MINUS RY => RB C= COUT  TR= ALL
51A: 8B94 2A24 0000 0000
                    (2690)        RR      12 => RM
51B: 0200 A204 0000 0000
                    (2691)        ALU     FLTL MINUS RM + C  => FLTL  C= COUT  TR= ALL
51C: 8F98 5A24 0000 0000
                    (2692)        RR      13 => RM
51D: 0200 B204 0000 0000
                    (2693)        ALU     FLTH MINUS RM + C  => FLTH  C= AOVFL ;
                    (2694)                SETCC  GO TO DNRML
51E: 8E98 4B77 0004 053A
                    (2695) *
                    (2696) *
                    (2697) *  SUBROUTINE DLOAD.  ROUTINE LOADS THE FULL 4 WORD
                    (2698) * FLOATING POINT ARGUMENT INTO 13,12,11,AND RM, RESPECTIVLY.
                    (2699) * THE SUBROUTINE EXPECTS THE FIRST WORD TO BE IN RM.  RY IS
                    (2700) * ASSUMED TO CONTAIN A POINTER TO THE SECOND WORD IN MEMORY.
                    (2701) * NOTE THAT THE EXPONENT IS LEFT IN RM.
                    (2702) *
                    (2703) *
                    (2704) DGET   CPU     BB   RM   ALL  0      0     M     13 ;
                    (2705)                RMRFMRDY  MREAD
51F: EF00 B584 0000 0000
                    (2706)        ALU     INC RY => RY
520: 8A65 1304 0000 0000
                    (2707)        CPU     BB   RM   ALL  0      0     M     12 ;
                    (2708)                RMRFMRDY  MREAD
521: EF00 A584 0000 0000
                    (2709)        ALU     INC RY => RY
522: 8A65 1304 0000 0000
                    (2710)        CPU     BB   RM   ALL  0      0     M     11 ;
                    (2711)                RMRFMRDY  MREAD ,S ON TRUE, POP
523: EF00 9584 000C 0004
                    (2712) *
                    (2713) *
                    (2714) * ADJUST SUBROUTINE.  CALLED DFIX, THIS ROUTINE ADJUSTS TWO
                    (2715) * NUMBERS OF UNEQUAL EXPONENTS BY 'UN-NORMALIZING' THE
                    (2716) * NUMBER WITH SMALLER EXPONENT.  IF THE TWO NUMBERS ARE TOO
```

```
                    (2717) * FAR APART, AND THE NEW ARGUMENT IS SMALLER, THE ROUTINE
                    (2718) * ABORTS, RETURNING TO THE FETCH CYCLE.  IF THE OLD ARGUMENT
                    (2719) * IS THE SMALLER,  THEN IT IS ZEROED,AND THE EXPONENT (VSC)
                    (2720) * IS MADE EQUAL TO THAT OF THE NEW ARGUMENT.
                    (2721) *
                    (2722) * IF THE TWO NUMBERS CAN BE COMBINED, VSC IS SET TO THE
                    (2723) * LARGER EXPONENT, AND THE SMALLER NUMBER IS SHIFTED RIGHT.
                    (2724) *
                    (2725) * RY IS ALWAYS LOADED WITH THE LSW OF THE NEW ARGUMENT.
                    (2726) *
                    (2727) DFIX    ALU      ZERO MINUS RM => RM LOADRSC  TR= ALL ;
                    (2728)                  JUMP ON FCBIT TO DFIX1
524: 8F95 140E 0004 F52F
                    (2729)         RR       11 => RY TR= ALL ,S ON EQ TO POP
525: 0300 9304 000E 4004
                    (2730)         RR       RCM = 48 => YSAVE
526: F200 CA04 0002 0130
                    (2731) DFIX3   ALU      INC 13 + 0 => NULL SETCC ;
                    (2732)                  JUMP ON LT TO DFIX2
527: 9200 B607 0006 6532
                    (2733) * NEW ARGUMENT IS SMALLER, SHIFT IT RIGHT OR ABORT.
                    (2734) *
                    (2735)         ALU      INC FLTH + 0 => NULL SETCC
528: 9200 4007 0000 0000
                    (2736)         ALU      YSAVE PLUS RM => NULL SETCC  TR= ALL ;
                    (2737)                  JUMP ON EQ TO DFIX4-1
529: 8F60 C607 0006 4536
                    (2738)         ALU      ZERO PLUS RM => NULL C= AOVFL ;
                    (2739)                  LOADRSC  JUMP ON LE TO DNRML
52A: 8E61 167E 0004 753A
                    (2740) *
                    (2741) * DE-NORMALIZE CAN BE DONE.
                    (2742) *
                    (2743)         CPU      RFRS 3    ALL   6      0     M     13 ;
                    (2744)                  RF200     LINK
52B: 4F60 BA44 0000 0000
                    (2745)         CPU      RFRS 1    NX    6      0     M     12 ;
                    (2746)                  RF200     LINK
```

```
52C: 4660 AA44 0000 0000
                    (2747)          CPU     RFRS 1    NX    6      0     M     11 ;
                    (2748)                  RF240     NOP   INCRSC ;
                    (2749)                  JUMP ON RSCNEM1 TO *-2
52D: 4660 9B05 0004 952B
                    (2750)          RR      11 => RY , S ON TRUE, POP
52E: 0200 9304 000C 0004
                    (2751) * IF EXPONENT OVERFLOW IS DETECTED HERE, ONE OF THE
                    (2752) * TWO NUMBERS WILL BE PRESERVED UNCHANGED BECAUSE THE
                    (2753) * EXPONENTS ARE TOO FAR APART FOR AN ADD OR SUBTRACT.
                    (2754) * THEREFORE, FORCE THE SIGN THE RIGHT WAY AND ALSO FORCE
                    (2755) * THE MAXIMUN EXPONENT DIFFERENCE ALLOWED TO BE ZERO
                    (2756) * TO ALLOW THE REST OF THE ADJUST ROUTINE TO ACT CORRECTLY.
                    (2757) *
                    (2758) *
                    (2759) DFIX1    RR      RCM = $8000 => YSAVE
52F: F200 CA04 0001 0100
                    (2760)          ALU     YSAVE MINUS RM => NULL SETCC
530: 8E94 C007 0000 0000
                    (2761)          RR      11 => RY  , GO TO DFIX3
531: 0200 9304 0004 0527
                    (2762) *
                    (2763) *
                    (2764) * TEST FOR VALID ALIGN.  IF VALID, UN-NORMALIZE THE
                    (2765) * FACC.  IF NOT VALID, ZERO THE FACC AND RETURN.
                    (2766) * IN BOTH CASES, ADD THE EXPONENT DIFFERENCE TO THE
                    (2767) * VSC.
                    (2768) *
                    (2769) *
                    (2770) DFIX2    ALU     YSAVE MINUS RM => NULL SETCC TR= ALL ;
                    (2771)                  S ON EQ, POP
532: 8F94 C607 000E 4004
                    (2772)          CPU     RFRS 3    ALL   6      0     M     FLTH ;
                    (2773)                  RF240     LINK  NOP ;
                    (2774)                  JUMP ON LE TO DFIX4
533: 4F60 4B44 0004 7537
                    (2775)          CPU     RFRS 1    NX    6      0     M     FLTL ;
                    (2776)                  RF200     LINK
```

```
534: 4660 5A44 0000 0000
                    (2777)          CPU     RFRS 1    NX    6      0     M     RB ;
                    (2778)                  RF240     NOP   INCRSC ;
                    (2779)                  JUMP ON RSCNEM1 TO *-2
535: 4660 2B05 0004 9533
                    (2780)          ALU     VSC PLUS RM => VSC ,S ON TRUE, POP
536: 8E60 6B04 000C 0004
                    (2781) * ZERO OUT THE FACC
                    (2782) *
                    (2783) DFIX4    ALU     CON 0 => FLTH
537: 923C 4A04 0000 0000
                    (2784)          ALU     CON 0 => FLTL
538: 923C 5A04 0000 0000
                    (2785)          ALU     CON 0 => RB , GO TO DFIX4-1
539: 923C 2B04 0004 0536
                    (2786) *
                    (2787) *
                    (2788) * NORMALIZE SUBROUTINE.  THIS DOUBLE PRECSION NORMALIZE
                    (2789) * ROUTINE OPERATES EXACTLY AS THE SINGLE PRECISION ROUTINE
                    (2790) * DOES.  IF THE CBIT IS SET COMMING INTO THE ROUTINE, THE
                    (2791) * VALUE IS TAKEN TO BE OVERSHIFTED, AND THE NUMBER IS
                    (2792) * RIGHT SHIFTED ONE PLACE.  THE SIGN BIT IS RECONSTRUCTED
                    (2793) * FROM THE CONDITION CODE AT ENTRY.  THE CODE IS ASSUMED
                    (2794) * TO BE THE OPPOSITE SIGN OF THE PROPPER NUMBER.
                    (2795) *
                    (2796) * IF THE NUMBER IS ALREADY NORMALIZED, A RETURN TO FETCH IS
                    (2797) * DONE.  OTHERWISE, THE FACC IS SHIFTED LEFT UNTIL
                    (2798) * THE NUMBER IS NORMALIZED, THE EXPONENT IS REDUCED
                    (2799) * APPROPRIATELY, AND A RETURN TO FETCH IS DONE.
                    (2800) *
                    (2801) *
                    (2802) DNRML    RR      RCM = -48 => NULL LOADRSC
53A: F200 000E 0003 FED0
                    (2803)          CPU     RFLS 3    ALL   7      0     M     FLTH ;
                    (2804)                  RY240  SOVFL  , JUMP ON FCBIT TO DNRM1
53B: 2F70 4364 0004 F540
                    (2805)          CPU     RFLS 7    NX    7      0     M     RB ;
                    (2806)                  RF240     LINK  , JUMP ON FCBIT TO DNRM2
```

```
53C: 3E70 2B44 0004 F545
                    (2807)          CPU     RFLS 3    ALL   7     0     M     FLTL ;
                    (2808)                  RF200     LINK
53D: 2F70 5A44 0000 0000
                    (2809)          CPU     RFLS 3    NX    3     0     M     FLTH ;
                    (2810)                  RF240     NOP   INCRSC ;
                    (2811)                  JUMP ON RSCNEM1 TO *-3
53E: 2E30 4B05 0004 953B
                    (2812)          ALU     CON 0 => VSC C= BD01 ,GO TO F1
53F: 923C 6B34 0004 0000
                    (2813) DNRM1    ALU     VSC PLUS RCM = 1  => VSC  C= AOVFL
540: 9260 6A74 0002 0001
                    (2814)          CPU     RFRS 5    ALL   6     0     M     FLTH ;
                    (2815)                  RF240     LINK  NOP ;
                    (2816)                  JUMP ON LT TO *+2
541: 5760 4B44 0006 6543
                    (2817)          ALU     FLTH OR RCM = $8000 => FLTH
542: 924C 4A04 0001 0100
                    (2818)          CPU     RFRS 1    ALL   6     0     M     FLTL ;
                    (2819)                  RF200     LINK
543: 4760 5A44 0000 000C
                    (2820)          CPU     RFRS 1    NX    6     0     M     RB ;
                    (2821)                  RF240     NOP   JAMF ;
                    (2822)                  JUMP ON FCBIT TO DFLEX
544: 4660 2B00 0004 F5C8
                    (2823) DNRM2    RR      RCM = $FFC0 => 11
545: F200 9A04 0003 FFC0
                    (2824)          CPU     RFRS 1    ALL   6     0     M     RB ;
                    (2825)                  RF200     NOP   NOP
546: 4760 2A04 0000 0000
                    (2826)          ALU     11 OR RSC => 11
547: 864C 9A04 0000 000C
                    (2827)          ALU     11 PLUS RCM = 48  => RY
548: 9260 9304 0002 0130
                    (2828)          ALU     VSC MINUS RY => VSC C= AOVFL
549: 8A94 6A74 0000 0000
                    (2829)          RR      ,,JAMF JUMP ON FCBIT TO DFLEX
54A: 0200 0600 0004 F5C8
```

```
                    (2830) *
                    (2831) *
                    (2832) * DOUBLE PRECISION FLOATING POINT MULTIPLY.  TECHNIQUE USED IS
                    (2833) * A SIMPLE EXTENSION OF THE SINGLE PRECISION MULTIPLY.
                    (2834) * THE EXPONENT IS UPDATED FIRST, THEN 16 BITS OF SINGLE
                    (2835) * PRECISION MULTIPLY, THEN 16 BITS OF DOUBLE PRECISION,
                    (2836) * AND FINALLY 15 BITS OF TRIPPLE PRECISION SHIFT AND/OR
                    (2837) * MULTIPLY.  FINALLY, A TRIPLE PRECISION SUBTRACT
                    (2838) * IS DONE, IF NECESSARY, FOLLOWED BY NORMALIZE AS NEEDED
                    (2839) *
                    (2840) *
                    (2841) DFMP1   CPU      BB    RCM   NONE  0       0      0      0 ;
                    (2842)                  280   NOP   PUSHBD  DATA DFMP4
54B: F000 060C 0002 0B4D
                    (2843)         ALU      INC RY => RY , GO TO DGET
54C: 8A65 1304 0004 051F
                    (2844) *
                    (2845) * FIRST, FIX THE EXPONENT
                    (2846) *
                    (2847) DFMP4   ALU      VSC MINUS RCM = 128 => VSC  C= AOVFL
54D: 9294 6A74 0002 0080
                    (2848)         ALU      VSC PLUS RM => VSC C= AOVFL , ;
                    (2849)                  JUMP ON FCBIT TO DFM1
54E: 8E60 6B74 0004 F55C
                    (2850) DFM2    RR       FLTL => RY , JUMP ON FCBIT TO DFLEX
54F: 0200 5304 0004 F5C8
                    (2851)         RR       FLTH => RM
550: 0200 4204 0000 0000
                    (2852)         CPU      AL   RCM  ALL   ZERO   L      M      FLTH ;
                    (2853)                  RF200     ,LOADRSC DATA -16
551: 933C 4A0E 0003 FFF0
                    (2854)         CPU      RFRS 5    NX    6      0      M      11 ;
                    (2855)                  RF200     LINK
552: 5660 9A44 0000 0000
                    (2856)         CPU      ALRS RM   NX    ADD    0      M      FLTH ;
                    (2857)                  RF240     NOP , EAC MPYLOGIC
553: CE60 4B04 0008 C000
                    (2858)         CPU      RFRS 5    ALL   6      0      M      11 ;
```

```
                    (2859)                  RF240     LINKS INCRSC ;
                    (2860)                  JUMP ON RSCNEM1 TO *-1
554: 5760 9B15 0004 9553
                    (2861) *
                    (2862) * SINGLE PRECISION PART DONE,  DO DOUBLE PRECISION.
                    (2863) *
                    (2864)          CPU     AL   RCM  ALL   ZERO   L     M     FLTL ;
                    (2865)                  RF200    NOP   LOADRSC  DATA -16
555: 933C 5A0E 0003 FFF0
                    (2866)          CPU     RFRS 5    NX    6      0     M     12 ;
                    (2867)                  RF280     LINKS NOP ;
                    (2868)                  JUMP ON NOTRF16 TO DFM3
556: 5660 AC14 0007 755B
                    (2869)          ALU     FLTL PLUS RY => FLTL C= COUT
557: 8A60 5A24 0000 0000
                    (2870)          CPU     ALRS RM   ALL   ADD    C     M     FLTH ;
                    (2871)                  RF240     LINK
558: CF68 4B44 0000 0000
                    (2872) DFM5     CPU     RFRS 1    NX    6      0     M     FLTL ;
                    (2873)                  RF240     NOP   INCRSC ;
                    (2874)                  JUMP ON RSCNEM1 TO *-3
559: 4660 5B05 0004 9556
                    (2875)          RR      RY => 11 , GO TO DFM4
55A: EA00 9B04 0004 055E
                    (2876) *
                    (2877) * SHIFT PORTION OF DOUBLE PRECISION PART
                    (2878) *
                    (2879) DFM3     CPU     RFRS 3    ALL   6      0     M     FLTH ;
                    (2880)                  RF240     LINK  NOP    GO TO DFM5
55B: 4F60 4B44 0004 0559
                    (2881) *
                    (2882) * EXPONENT OVERFLOW TESTS
                    (2883) *
                    (2884) DFM1     ALU     CON 0 => NULL C= BD01 , JUMP ON FCBIT TO DFM2
55C: 923C 0634 0004 F54F
                    (2885)          RR      ,,,GO TO DFLEX
55D: 0200 0604 0004 05C8
                    (2886) *
```

```
                    (2887) * FINALLY DO TRIPLE PRECISION MULTIPLY
                    (2888) *
                    (2889) DFM4    RR       RCM = -15 => NULL LOADRSC
55E: F200 000E 0003 FEF1
                    (2890) DFM7    CPU      RFRS 5     NX     6      0      M      13 ;
                    (2891)                  RF280      LINKS NOP ;
                    (2892)                  JUMP ON NOTRF16 TO DFM8
55F: 5660 BC14 0007 756F
                    (2893)         RR       RB => RY
560: 0200 2804 0000 0000
                    (2894)         ALU      12 PLUS RY => 12 C= COUT
561: 8A60 AA24 0000 0000
                    (2895)         RR       11 => RY
562: 0200 9804 0000 0000
                    (2896)         ALU      FLTL PLUS RY + C  => FLTL C= COUT
563: 8A68 5A24 0000 0000
                    (2897)         CPU      ALRS RM    ALL    ADD    C      M      FLTH ;
                    (2898)                  RF240      LINK
564: CF68 4B44 0000 0000
                    (2899) DFM6    CPU      RFRS 1     ALL    6      0      M      FLTL ;
                    (2900)                  RF200      LINK
565: 4760 5A44 0000 0000
                    (2901)         CPU      RFRS       1      NX     6      0      M 12 ;
                    (2902)                  RF240      NOP    INCRSC ;
                    (2903)                  JUMP ON RSCNEM1 TO DFM7
566: 4660 AB05 0004 955F
                    (2904) *
                    (2905) * NOW DO FINAL SUBTRACT
                    (2906) *
                    (2907)         CPU      RF   NOP  ALL    0      0      M      13 ;
                    (2908)                  RF280      BD01   NOP ;
                    (2909)                  JUMP ON NOTRF16 TO DFM9
567: 1300 BC34 0007 756D
                    (2910)         RR       RB => RY
568: 0200 2804 0000 0000
                    (2911)         ALU      12 MINUS RY => 12 C= COUT
569: 8A94 AA24 0000 0000
                    (2912)         RR       11 => RY
```

```
56A: 0200 9804 0000 0000
                    (2913)          ALU     FLTL MINUS RY + C  => FLTL  C= COUT
56B: 8A98 5A24 0000 0000
                    (2914)          ALU     FLTH MINUS RM + C => FLTH C= AOVFL  SETCC
56C: 8E98 4A77 0000 0000
                    (2915) DFM9     RR      12 => RM
56D: 0200 A204 0000 0000
                    (2916)          RR      RM => RB , GO TO DNRML
56E: EE00 2B04 0004 053A
                    (2917) *
                    (2918) * SHIFT PART OF TRIPLE PRECISION MPY LOOP.
                    (2919) *
                    (2920) DFM8     CPU     RFRS 3    ALL   6      0     M     FLTH ;
                    (2921)                  RF240     LINK  NOP GO TO DFM6
56F: 4F60 4B44 0004 0565
                    (2922) *
                    (2923) *
                    (2924) * DIVIDE -- DOUBLE PRECISION FLOATING POINT
                    (2925) *
                    (2926) DFDV1    CPU     BB   RCM  0     0      0     0     0 ;
                    (2927)                  280  NOP  PUSHBD   DATA DDV4
570: F000 060C 0002 0AC7
                    (2928)          ALU     INC RY => RY ,GO TO DGET
571: 8A65 1304 0004 051F
                    (2929) * DIVIDE BY ZERO TEST
                    (2930) DDV30    ALU     FLTH XOR RY => NULL SETCC ;
                    (2931)                  JUMP ON EQ TO DFLEX1
572: 8A6C 4607 0006 45CA
                    (2932)          ALU     VSC PLUS RCM = 129 => VSC C= AOVFL ;
                    (2933)                  CLEARFUII
573: 9260 6A7F 0002 0181
                    (2934)          RR      RX => RX NOP JUMP ON GE TO *+2
574: 0200 0B04 0004 6576
                    (2935) * SET FUII FOR NEGATIVE RESULTS
                    (2936)          RR      ,,,EAC SETFUII
575: 0200 0004 000A 800C
                    (2937) * EXPONENT CALCULATION
                    (2938)          ALU     VSC MINUS RM => VSC ,C= AOVFL TR= ALL ;
```

```
                    (2939)                  JUMP ON FCBIT TO DDV5
576: 8F94 6B74 0004 F5A5
                    (2940) * EXPONENTS DONE EXCEPT FOR OVERFLOW
                    (2941) * TEST NOW TO SEE IF EITHER OF THE TWO ARGUMENTS
                    (2942) * IS NEGATIVE.  IF IT IS, THEN COMPLEMENT IT
                    (2943) *
                    (2944) DDV6    ALU      INC 13 + 0 => RM  SETCC ;
                    (2945)                  JUMP ON FCBIT TO DFLEX
577: 9200 B407 0004 F5C8
                    (2946)         ALU      INC FLTH + 0 => NULL SETCC ;
                    (2947)                  JUMP ON GE TO DDV7
578: 9200 4607 0004 657D
                    (2948) * COMPLEMENT NEW ARGUMENT
                    (2949) *
                    (2950)         ALU      NOT 11 => 11
579: 92FC 9A04 0000 0000
                    (2951)         ALU      NOT 13 => RM
57A: 92FC B204 0000 0000
                    (2952)         ALU      INC 11 => 11 C= COUT
57B: 9204 9A24 0000 0000
                    (2953)         ALU      NOT 12 => 12 , TR= ALL ;
                    (2954)                  JUMP ON FCBIT TO DDV8
57C: 93FC AB04 0004 F5A7
                    (2955) * NOW TEST NEW ARGUMENT
                    (2956) *
                    (2957) DDV7    RR       12 => RY ,JUMP ON GE TO DDV11 TR= ALL
57D: 0300 A304 0004 6582
                    (2958) * COMPLEMEMT ACCUMULATOR
                    (2959) *
                    (2960)         ALU      NOT RB => RB
57E: 92FC 2A04 0000 000C
                    (2961)         ALU      NOT FLTL => FLTL
57F: 92FC 5A04 0000 0000
                    (2962)         ALU      INC RB => RB C= COUT TR= ALL
580: 9304 2A24 0000 0000
                    (2963)         ALU      NOT FLTH => FLTH , JUMP ON FCBIT TO DDV10
581: 92FC 4B04 0004 F5AB
                    (2964) * NOW SET UP AND EXECUTE 16 ITTERATIONS OF
```

```
                    (2965) * THE TRIPPLE PRECISION DIVIDE LOOP.
                    (2966) *
                    (2967) DDV11   ALU      FLTH MINUS RM => NULL SETCC TR= ALL
582: 8F94 4007 0000 0000
                    (2968)         RR       RCM = -17 => NULL LOADRSC
583: F200 000E 0003 FEEF
                    (2969) * RM=NH/RY,12=NM/11=NL/13=POSITIVE
                    (2970) *
                    (2971) *
                    (2972) * COMPARE PART OF NON-PERFORMING DIVIDE
                    (2973) *
                    (2974) DDV15   CPU      RFLS 3     ALL    3       0      M      13 ;
                    (2975)                  RF240      NOP    INCRSC ;
                    (2976)                  JUMP ON LT TO DDV13
584: 2F30 BB05 0006 65B1
                    (2977)         ALU      FLTL MINUS RY => NULL SETCC ;
                    (2978)                  C= COUT JUMP ON GT TO DDV14
585: 8A94 5627 0006 75C6
                    (2979)         RR       11 => RY , JUMP ON FCBIT TO *+2
586: 0200 9304 0004 F588
                    (2980)         RR       12 => RY , GO TO DDV13
587: 0200 A304 0004 05B1
                    (2981)         ALU      RB MINUS RY => NULL C= COUT  , ;
                    (2982)                  JUMP ON NE TO DDV12
588: 8A94 2624 0004 458B
                    (2983)         RR       RX => RX , JUMP ON FCBIT TO DDV12
589: 0200 0B04 0004 F58B
                    (2984)         RR       12 => RY , GO TO DDV13
58A: 0200 A304 0004 05B1
                    (2985) *END OF DIVIDE TEST.
                    (2986) *
                    (2987) * SUBTR ACT -- SHIFT
                    (2988) *
                    (2989) DDV12   ALU      RB MINUS RY => RB C= COUT
58B: 8A94 2A24 0000 0000
                    (2990)         CPU      RF   NOP  ALL   ZERO   L      M      12 ;
                    (2991)                  RY200     NOP   SETCC
58C: 133C A807 0000 000C
```

```
                    (2992)          ALU     FLTL MINUS RY + C => FLTL C= COUT
58D: 8A98 5A24 0000 0000
                    (2993)          CPU     RFLS 7    ALL   7     0     M     RB ;
                    (2994)                  RF200     LINK
58E: 3F70 2A44 0000 0000
                    (2995)          CPU     RFLS 3    NX    7     0     M     FLTL ;
                    (2996)                  RF200     LINK
58F: 2E70 5A44 0000 0000
                    (2997)          CPU     ALLS RM   ALL   SUB   C     M     FLTH ;
                    (2998)                  RF240     LINK
590: AF98 4B44 0000 0000
                    (2999) *
                    (3000) * START OF NEXT ITTERATION
                    (3001) *
                    (3002) DDV9     ALU     FLTH MINUS RM => NULL SETCC ;
                    (3003)                  JUMP ON RSCNEM1 TO DDV15
591: 8E94 4607 0004 9584
                    (3004) * DOUBLE PERECISION DIVIDE.  SIMILAR TO THAT IN FDV.
                    (3005) *
                    (3006) * ON ENTRY: RM=NH,RY=NM,13=HIGH ANSWER,LINK=16TH QUOTIENT BIT.
                    (3007) *
                    (3008)          RR      RCM = -17 => NULL LOADRSC
592: F200 000E 0003 FEEF
                    (3009) *
                    (3010) * HIGH ORDER CONDITION CODE ALREADY TESTED UPON ENTRY
                    (3011) *
                    (3012) DDV23    CPU     RFLS 3    NX    7     0     M     12 ;
                    (3013)                  RF240     NOP   INCRSC ;
                    (3014)                  JUMP ON LT TO DDV21
593: 2E70 AB05 0006 6596
                    (3015)          ALU     FLTL MINUS RY => NULL C= COUT , ;
                    (3016)                  JUMP ON GT TO DDV24
594: 8A94 5624 0006 7598
                    (3017)          RR      RX => RX , JUMP ON FCBIT TO DDV24
595: 0200 0B04 0004 F598
                    (3018) * DOUBLE WORD SHIFT
                    (3019) DDV21    CPU     RFLS 7    NX    7     0     M     FLTL ;
                    (3020)                  RF200     LINK
```

```
596: 3E70 5A44 0000 0000
                    (3021)         CPU      RFLS 3     ALL   7      0      M      FLTH ;
                    (3022)                  RF240      LINK  NOP ;
                    (3023)                  GO TO DDV20
597: 2F70 4B44 0004 059B
                    (3024) DDV24   ALU      FLTL MINUS RY => FLTL C= COUT
598: 8A94 5A24 0000 0000
                    (3025)         CPU      RFLS 7     ALL   7      0      M      FLTL ;
                    (3026)                  RF200      LINK
599: 3F70 5A44 0000 0000
                    (3027)         CPU      ALLS RM    ALL   SUB    C      M      FLTH ;
                    (3028)                  RF240      LINK
59A: AF98 4B44 0000 0000
                    (3029) DDV20   ALU      FLTH MINUS RM => NULL SETCC ;
                    (3030)                  JUMP ON RSCNEM1 TO DDV23
59B: 8E94 4607 0004 9593
                    (3031) *
                    (3032) * FINISHED DOUBLE PART. NOW DO SINGLE PART.
                    (3033) *
                    (3034) *
                    (3035) * LINK CONTAINS 32ND QUOTIENT BIT.  RM=NH,13=Q1,12=Q2
                    (3036) * BUILD Q3 IN RB. PUT 13 IN RY, INITIALIZE AND DIVIDE FLTH/RM
                    (3037) *
                    (3038)         CPU      RF   RCM  ALL   ZERO   L      M      RB ;
                    (3039)                  RF200     NOP   LOADRSC ;
                    (3040)                  DATA -15
59C: 133C 2A0E 0003 FEF1
                    (3041)         CPU      RFLS 3     NX    7      0      M      RB ;
                    (3042)                  RF200      LINK
59D: 2E70 2A44 0000 0000
                    (3043)         CPU      AL   RCM   ALL   AND    L      M      13 ;
                    (3044)                  RY240      NOP   SETCC DATA $7FFF
59E: 931C B307 0000 FFFF
                    (3045) * SINGLE PRECISION DIVIDE LOOP
                    (3046) *
                    (3047)         CPU      ALLS RM    ALL   SUB    1      M      FLTH ;
                    (3048)                  RF280      LINK ,EAC DIVLOGIC
59F: AF94 4C44 0009 0000
```

```
                    (3049)          CPU     RFLS 3    NX    7      0     M     RB ;
                    (3050)                  RF240     LINK  INCRSC ;
                    (3051)                  JUMP ON RSCNEM1 TO *-1
5A0: 2E70 2B45 0004 959F
                    (3052) * DONE WITH DIVIDE
                    (3053) *
                    (3054) * RESULTS IN RY,12,RB
                    (3055) *
                    (3056)          RR      RY => FLTH C= BD01
5A1: EA00 4A34 0000 0000
                    (3057)          RR      12 => RM  TR= ALL  ,JUMP ON FUII TO *+2
5A2: 0300 A404 0005 55A4
                    (3058)          RR      RM => FLTL ,GO TO DNRML
5A3: EE00 5B04 0004 053A
                    (3059)          ALU     NOT RM => FLTL ,GO TO DFCM+1
5A4: 8EAC 5B04 0004 0435
                    (3060) *
                    (3061) * ASSORTED DIVIDE NON-STRAIGHT LINE CODE
                    (3062) *
                    (3063) DDV5     ALU     CON 0 => NULL C= BD01 , ;
                    (3064)                  JUMP ON FCBIT TO DDV6
5A5: 923C 0634 0004 F577
                    (3065)          RR      ,,,GO TO DFLEX
5A6: 0200 0604 0004 05C8
                    (3066) DDV8     ALU     INC 12 + C => 12 C= COUT
5A7: 9208 A[illegible]4 0000 0000
                    (3067)          ALU     INC RM + C => RM SETCC
5A8: 8E69 1207 0000 0000
                    (3068)          ALU     INC FLTH + 0 => NULL SETCC ;
                    (3069)                  JUMP ON GE TO DDV7
5A9: 9200 4607 0004 657D
                    (3070)          CPU     AL   RCM  NX    DEC    0     M     VSC ;
                    (3071)                  RF240     AOVFL , GO TO DFCM
5AA: 92F0 6B74 0004 0434
                    (3072) *
                    (3073) * FINISH COMPLEMENT
                    (3074) *
                    (3075) DDV10    ALU     INC FLTL + C => FLTL C= COUT
```

```
5AB: 9208 5A24 0000 0000
                    (3076)          ALU     INC FLTH + C => FLTH SETCC
5AC: 9208 4A07 0000 0000
                    (3077)          ALU     FLTH MINUS RM => NULL SETCC  TR= ALL ;
                    (3078)                  JUMP ON GE DDV11+1
5AD: 8F94 4607 0004 6583
                    (3079)          ALU     VSC PLUS RCM = 1 => VSC C= AOVFL
5AE: 9260 6A74 0002 0001
                    (3080)          CPU     RFRS 5    ALL   6     0     M     FLTH ;
                    (3081)                  RF240     NOP   NOP ;
                    (3082)                  JUMP ON FCBIT TO DFLEX
5AF: 5760 4B04 0004 F5C8
                    (3083)          ALU     FLTH MINUS RM => NULL  SETCC GO TO DDV11+1
5B0: 8E94 4607 0004 0583
                    (3084) *
                    (3085) * SHIFT STEP -- TRIPPLE SUBTRACT
                    (3086) *
                    (3087) DDV13    CPU     RFLS 7    ALL   7     0     M     RB ;
                    (3088)                  RF200     LINK
5B1: 3F70 2A44 0000 0000
                    (3089)          CPU     RFLS 3    NX    7     0     M     FLTL ;
                    (3090)                  RF200     LINK
5B2: 2E70 5A44 0000 0000
                    (3091)          CPU     RFLS 3    ALL   7     0     M     FLTH ;
                    (3092)                  RF240     LINK  NOP ;
                    (3093)                  GO TO DDV9
5B3: 2F70 4B44 0004 0591
                    (3094) *
                    (3095) * DOUBLE PRECISION COMPARE
                    (3096) *
                    (3097)          ORG     $5B4
                    (3098) DFCS1    CPU     BB   RM   NX    XOR   L     M     FLTH ;
                    (3099)                  200  8D01 SETCC
5B4: EE6C 4037 0000 0000
                    (3100)          ALU     INC RY => RY , JUMP ON LT TO FCS2
5B5: 8A65 1304 0006 6477
                    (3101)          CPU     BB   RM   ALL   BB    L     M     13 ;
                    (3102)                  RMRFMRDY  MREAD SETCC
```

```
5B6: EF5C B587 0000 0000
                    (3103)          RR      RY => 12
5B7: EA00 AA04 0000 0000
                    (3104)          ALU     12 PLUS RCM = 2  => RY
5B8: 9260 A304 0002 0002
                    (3105) * FOR NEG REVERSE EXPONENT TEST
                    (3106)          CPU     BB   RM   ALL   0      0     M     11 ;
                    (3107)                  RMRFMRDY  MREAD  NOP   ;
                    (3108)                  JUMP ON LT TO CS1
5B9: EF00 9584 0006 65C3
                    (3109) *
                    (3110) * EXPONENT TEST
                    (3111) *
                    (3112)          ALU     INC FLTH + 0 => NULL SETCC JUMP ON EQ TO FCS7
5BA: 9200 4607 0006 4479
                    (3113)          ALU     VSC MINUS RM => RM SETCC ;
                    (3114)                  C= AOVFL  JUMP ON EQ TO FCS2+1
5BB: 8E94 6477 0006 4478
                    (3115) * IF OVERFLOW - REVERSE TEST
                    (3116)          RR      13 => RM , JUMP ON FCBIT TO CS3
5BC: 0200 B404 0004 F5C5
                    (3117) *
                    (3118)          ALU     INC 12 => RY , JUMP ON GT TO F1
5BD: 9204 A304 0006 7000
                    (3119) *
                    (3120) * EXPONENTS EQUAL
                    (3121) *
                    (3122) CS4      ALU     FLTH MINUS RM => NULL SETCC ;
                    (3123)                  JUMP ON NE TO FCS2+1
5BE: 8E94 4607 0004 4478
                    (3124)          RR      11 => RM , JUMP ON GT TO F1
5BF: 0200 9404 0006 7000
                    (3125)          ALU     FLTL MINUS RM => NULL SETCC TR= ALL  C= COUT ;
                    (3126)                  JUMP ON LT TO FCS2+1
5C0: 8F94 5627 0006 6478
                    (3127) *
                    (3128) * HIGH ORDER PART EQUAL -- TEST MIDDLE
                    (3129) *
```

```
                    (3130)          CPU     ,,ALL,,,,,RMMRDY MREAD NOP JUMP ON NE TO FCS2
5C1: 1300 0784 0004 4477
                    (3131)          ALU     RB MINUS RM => NULL C= COUT SETCC GO TO FCS2-1
5C2: 8E94 2627 0004 0476
                    (3132) * NEGATIVE EXPONENTS, REVERSE TEST
                    (3133) CS1      ALU     VSC MINUS RM => NULL SETCC C= AOVFL
5C3: 8E94 6077 0000 0000
                    (3134) * IF OVERFLOW, DO NOT REVERSE TEST.
                    (3135)          RR      13 => RM , JUMP ON FCBIT TO CAS4-1
5C4: 0200 B404 0004 F1E3
                    (3136) * REVERSE TEST
                    (3137) CS3      ALU     INC 12 => RY JAMF JUMP ON GE TO CS4
5C5: 9204 A300 0004 65BE
                    (3138) * PATCH SPACE
                    (3139) DDV14    RR      11 => RY , GO TO DDV12
5C6: 0200 9304 0004 058B
                    (3140) DDV4     ALU     INC 13 + 0 => RY  SETCC  GO TO DDV30
5C7: 9200 B307 0004 0572
                    (3141) *
                    (3142) * DOUBLE PRECISION FLOATING EXCEPTION
                    (3143) *
                    (3144) DFLEX    RR      RCM = $200 => 11
5C8: F200 9A04 0000 0500
                    (3145)          ALU     CON -1 => NULL C= BD01 , GO TO FLEX5
5C9: 92CC 0634 0004 04D3
                    (3146) DFLEX1   RR      RCM = $201 => 11
5CA: F200 9A04 0000 0401
                    (3147)          ALU     CON -1 => NULL C= BD01 , GO TO FLEX5
5CB: 92CC 0634 0004 04D3
          002714    (3148)          END
```

```
A2A         01AD   A 2016
ACA         01B8   A 2029
ADD         01EF   A 2150
ADD3        00B9   A 1515  2150
ADJ8        0444   A 2298  2322
ADJ9        0443   A 2293  2321
ADJUST      043A   A 2293  2383  2390
ALL         01D6   A 2100
ALR         01D9   A 2107
ALS         01D7   A 2101
ANA         01E9   A 2142
AOA         0133   A 1797  2020
ARL         01C2   A 2055
ARR         01C4   A 2061
ARS         01C3   A 2058
AVECT       0077   A 0984  1362  1384  1416  1992
AVECT1      0077   A 1319  1345  1364  1393  1506
BOOT        0091   A 1012  1435
BOOT1       0094   A 1439  1451
CAI         01FD   A 2165
CAL         01B4   A 2024
CAR         01B3   A 2023
CAS         01E0   A 2131
CAS4        01E4   A 2135  2137  2152  3135
CAS5        01E5   A 1369  1710  1852  1877  1967  2006  2013  2126  2137
                     2138  2141  2186  2187  2188  2189  2190  2191  2430
                     2433
CAS6        01F1   A 2134  2152
CAZ         01AB   A 2013
CEA         018B   A 1969
CEA3        0128   A 1784  1969
CHS         01A0   A 1999
CMA         01AF   A 2019
CP1         0035   A 1008  1045  1114  1368  1845
CP10        0022   A 1010
CP11        0023   A 1011
CP12        0024   A 1012
CP13        0025   A 1018
```

CP14	0026	A	1025				
CP15	0027	A	1018	1030			
CP16	0028	A	1032				
CP17	002A	A	1036				
CP18	002B	A	1038				
CP19	002C	A	1011	1040			
CP2	0036	A	1115				
CP3	0037	A	1036	1038	1040	1116	
CP4	0038	A	1119				
CP5	0039	A	1121				
CP6	003A	A	1123				
CP7	003B	A	1126				
CP8	0020	A	0995	0997	1006	1126	
CP9	0021	A	1008	1010			
CPPAR	006A	A	1323				
CPPAR3	007E	A	1383	1711			
CRA	01CE	A	2003	2084	2590		
CRB	019F	A	1998	2009			
CREP	013C	A	1808				
CRL	019E	A	1997	2249	2252	2268	2581
CS1	05C3	A	3108	3133			
CS3	05C5	A	3116	3137			
CS4	05BE	A	3123	3137			
CSA	01AC	A	2014				
DBL	0188	A	1959				
DDV10	05AB	A	2963	3075			
DDV11	0582	A	2957	2967	3078	3083	
DDV12	058B	A	2982	2983	2989	3139	
DDV13	05B1	A	2976	2980	2984	3088	
DDV14	05C6	A	2978	3139			
DDV15	0584	A	2976	3003			
DDV20	059B	A	3023	3030			
DDV21	0596	A	3014	3020			
DDV23	0593	A	3014	3030			
DDV24	0598	A	3016	3017	3024		
DDV30	0572	A	2931	3140			
DDV4	05C7	A	2927	3140			
DDV5	05A5	A	2939	3064			

DDV6	0577	A	2945	3064						
DDV7	057D	A	2947	2957	3069					
DDV8	05A7	A	2954	3066						
DDV9	0591	A	3003	3093						
DFAD	01F8	A	2160							
DFAD1	050D	A	2160	2670						
DFAD4	050F	A	2670	2673						
DFAD6	0511	A	2673	2676						
DFCM	0434	A	2276	3059	3071					
DFCS	01FC	A	2164							
DFCS1	05B4	A	2164	3099						
DFDV	01FB	A	2163							
DFDV1	0570	A	2163	2927						
DFIX	0524	A	2675	2688	2728					
DFIX1	052F	A	2728	2759						
DFIX2	0532	A	2732	2771						
DFIX3	0527	A	2732	2761						
DFIX4	0537	A	2737	2774	2783	2785				
DFLD	01F6	A	2158							
DFLD1	0500	A	2158	2653						
DFLEX	05C8	A	2276	2822	2829	2850	2885	2945	3065	3082	3144
DFLEX1	05CA	A	2931	3146						
DFM1	055C	A	2849	2884						
DFM2	054F	A	2850	2884						
DFM3	055B	A	2868	2880						
DFM4	055E	A	2875	2889						
DFM5	0559	A	2874	2880						
DFM6	0565	A	2900	2921						
DFM7	055F	A	2892	2903						
DFM8	056F	A	2892	2921						
DFM9	056D	A	2909	2915						
DFMP	01FA	A	2162							
DFMP1	054B	A	2162	2842						
DFMP4	054D	A	2842	2847						
DFSB	01F9	A	2161							
DFSB1	0516	A	2161	2683						
DFSB4	0518	A	2683	2686						
DFSB6	051A	A	2686	2689						

DFST	01F7	A	2159						
DFST1	0507	A	2159	2663					
DGET	051F	A	2671	2684	2705	2843	2928		
DIN	004F	A	1215	1220	1230				
DIN1	0053	A	1230	1243	1404				
DIV	0144	A	1825						
DIV17	009E	A	1457	1478					
DIV27	009C	A	1453	1476					
DIV4	00A8	A	1478	1827					
DIVER	0146	A	1460	1482	1829				
DMA	004B	A	1132	1206					
DMA1	0047	A	1183	1221	1237				
DMC	0048	A	1189	1206					
DMC1	005C	A	1189	1267					
DMC2	0089	A	1400	1404					
DMC3	0086	A	1274	1398					
DMT	003E	A	1147	1190					
DMX1	0070	A	1339						
DMX2	003C	A	1130	1340					
DNRM1	0540	A	2804	2813					
DNRM2	0545	A	2806	2823					
DNRML	053A	A	2279	2282	2681	2694	2739	2802	2916	3058
DOUT	0054	A	1234	1246					
DOUT1	0058	A	1212	1227	1246	1402			
DRX	01AA	A	2011						
E16S	0189	A	1961						
E32R	0189	A	1963						
E32S	0189	A	1962						
E64R	0189	A	1964						
EAA	0128	A	1785						
EMCM	018D	A	1971						
ENB	0190	A	1975						
ENTR	0100	A	1721						
EPMJ	0120	A	1774						
EPMJ6	C116	A	1755						
EPMX	0120	A	1771						
ERA	01ED	A	2148						
ERM	0117	A	1756						

ERMJ	0110	A	1745	1776						
ERMX	0110	A	1743							
ESIM	0196	A	1988							
EVIM	0196	A	1989							
EVMJ	0110	A	1744							
EVMX	0110	A	1742							
EVMX1	01F2	A	1752	2153						
F1	0000	A	0862	1569	1717	1890	1895	1957	2138	2257	2352
			2422	2426	2431	2589	2612	2662	2812	3118	3124
F10	0005	A	0891							
F11	0006	A	0897							
F12	0007	A	0903							
F13	0008	A	0891	0906						
F14	0009	A	0912							
F16	000A	A	0918							
F16A	000B	A	0924							
F16B	000C	A	0930							
F17	000D	A	0906	0935						
F18	000E	A	0941							
F18A	000F	A	0942							
F19	0014	A	0935	0963						
F20	0015	A	0969							
F3	0001	A	0868	1006						
F4	0002	A	0874							
F5	0003	A	0880							
F6	0010	A	0880	0903	0930	0947	0969	1970		
F6A	0011	A	0950	0956	0957					
F7	0012	A	0956							
F7A	0013	A	0957							
F9	0004	A	0868	0886	1779					
FAD	0165	A	1907							
FAD1	040B	A	1907	2201						
FAD4	045A	A	2201	2382						
FAD6	045C	A	2382	2384						
FCM	0410	A	2210	2406	2572					
FCS	016A	A	1917							
FCS1	046B	A	1919	2411						
FCS2	0477	A	2411	2420	2421	2423	2428	2431	3100	3114	3123

			3126	3130	3131				
FCS3	04C8	A	2415	2611					
FCS4	0470	A	2421						
FCS5	0472	A	2423	2613					
FCS7	0479	A	2418	2433	3112				
FDV	0168	A	1913						
FDV1	042B	A	1913	2259					
FDV10	04A2	A	2539	2559					
FDV11	04AC	A	2563						
FDV13	046A	A	2406	2516					
FDV4	048D	A	2259	2497					
FDV5	0498	A	2512	2520					
FDV6	04AA	A	2521	2525	2527	2531	2551	2559	
FDV7	04A5	A	2539	2549					
FDV8	04A7	A	2543	2545	2553				
FGEN	010D	A	1737						
FGEN1	0406	A	1737	2192					
FGEN2	040C	A	2185	2194	2209	2241	2265		
FHALT	003C	A	0862	1069					
FHALT2	0074	A	1069	1349					
FHALT3	002D	A	1045	1349					
FLD	0163	A	1903						
FLD1	0456	A	1905	2376					
FLEX	04CE	A	2210	2369	2446	2511	2530	2634	2635
FLEX1	04CF	A	2401	2636					
FLEX2	04D1	A	2499	2638					
FLEX4	04D2	A	2633	2635	2637	2639			
FLEX5	04D3	A	2640	3145	3147				
FLEX8	04CD	A	2264	2634					
FLEX9	04CB	A	2253	2632					
FLOT	0420	A	2242						
FLOT1	0416	A	2219	2245					
FLX	0129	A	1786						
FLX1	0103	A	1724	1786					
FLX2	012A	A	1724	1788					
FMP	0167	A	1911						
FMP1	0417	A	1911	2221					
FMP4	047A	A	2221	2443					

FMP5	0487	A	2466	2473						
FMP6	0486	A	2472	2478						
FMP7	0489	A	2469	2478						
FOUT	00FC	A	1575	1580	1591	1602	1655	1681	1696	1703	1717
FOUT1	008E	A	1417	1718						
FPAGE	0060	A	1291							
FPAGE3	008A	A	1291	1408						
FRAC	0430	A	2266							
FRAC1	04B2	A	2272	2576						
FRAC2	04B9	A	2587	2593						
FRAC3	04BC	A	2577	2593						
FREAD	0066	A	1311							
FRN	0414	A	2216							
FSB	0166	A	1909							
FSB1	040D	A	1909	2204						
FSB4	045F	A	2204	2389						
FSB6	0461	A	2389	2391						
FSEQ	040C	A	2186							
FSGT	0405	A	2191							
FSLE	0404	A	2190							
FSMI	0402	A	2188							
FSNE	0401	A	2187							
FSPL	0403	A	2189							
FST	0169	A	1915							
FST1	0464	A	1915	2395						
GENB	0149	A	1843							
GENB1	0183	A	1844	1953						
HLT	014B	A	1845							
IAB	01A2	A	1453	1497	2001					
IAB2	018F	A	1974	2008						
ICA	01B9	A	2030							
ICL	01B5	A	2025							
ICR	01B6	A	2026							
ILL	0142	A	1821							
ILL3	00B5	A	1510	1822						
IMA	0178	A	1933							
INA	0151	A	1865	1883						
INA1	0156	A	1871	1879	1881					

INA2	018A	A	1874	1967	2147					
INH	019C	A	1976							
INK	0186	A	1957							
INT	0424	A	2246							
INT1	0428	A	2253							
INT2	04BF	A	2256	2600	2610					
INT3	0016	A	0974	0974	1060					
INT4	04C6	A	2604	2609						
INT5	04C2	A	2605	2609						
INT6	04C7	A	2600	2610						
INTEX	002E	A	1058							
IRS	01E6	A	2139							
IRX	01A5	A	2005	2011						
ISI	019A	A	1993							
JDX	0138	A	1803							
JEQ	012B	A	1789							
JGE	0136	A	1800							
JGT	0131	A	1795							
JIX	0139	A	1805							
JLE	012F	A	1793							
JLT	0134	A	1798							
JMP	01D0	A	1790	1792	1794	1796	1799	1801	1807	1812	2086
JNE	012D	A	1791	1803	1805					
JST	014E	A	1848							
JST1	0150	A	1852	2647						
JSX	013A	A	1806							
LDA	01DE	A	1454	1499	1928	1974	2128			
LDA3	0160	A	1898	2128						
LDX	0147	A	1830							
LEQ	01CB	A	2078							
LF	01CE	A	2083							
LGE	01CC	A	2080							
LGT	01CD	A	2082							
LLE	01C9	A	2074							
LLL	01D1	A	2088							
LLR	01C5	A	2063							
LLS	01D3	A	2092							
LLT	01C8	A	2072							

LMCM	018D	A	1972							
LNE	01CA	A	2076							
LOAD	0419	A	2202	2205	2222	2235	2260			
LOGIC	01BA	A	2035							
LPMJ	0120	A	1773							
LPMX	012C	A	1772							
LRL	01BB	A	2037							
LRR	01BF	A	2047							
LRS	01BD	A	2042							
LT	01CF	A	2072	2074	2076	2078	2080	2082	2085	
MC1	0031	A	1076							
MC2	0032	A	1079							
MC3	0033	A	1082							
MC4	0034	A	1085	1324	1709					
MEMP3	008C	A	1330	1414						
MEMPAR	006C	A	1329	1335						
MI1	00BF	A	0977	1522						
MI2	001C	A	0992	1522						
MMOD	006E	A	1334							
MOVE	0449	A	2321	2327	2333	2335				
MPY	017C	A	1941							
NRM	0118	A	1757							
NRM10	0452	A	2351	2362						
NRM17	0451	A	2358	2361						
NRM18	044E	A	2355	2361						
NRM7	042D	A	2262	2355						
NRML	044B	A	2213	2215	2218	2219	2312	2351	2387	2394	2482
			2486	2574						
NVECT	001A	A	0978	0982						
OSI	019C	A	1995							
OTA	0159	A	1883							
OTA1	015F	A	1891	1895						
OTK	0192	A	1770	1979						
PAGE	007C	A	1292	1376						
PAGE3	0063	A	1300	1378						
PAGE4	0080	A	1300	1386						
PAGE7	0083	A	1386	1391						
PFL	0075	A	1350							

```
PID       0174  A 1929
PIM       0172  A 1927
PIO       0159  A 1882
RCB       01A7  A 2007
READ      0064  A 1305
REST      0059  A 1183  1191  1251
RMC       0187  A 1843  1958
RTN       0104  A 1726
RXM       0068  A 1318  1745  1774  1972  1976  1989  1995  2165
S2A       01AE  A 2018
SCA       0185  A 1956
SCB       01B2  A 2022
SGL       0188  A 1960
SKP       01DA  A 2110
SKS       0157  A 1868  1880
SKS1      01EC  A 1880  2147
SOA       01A4  A 2004
SSM       01B1  A 2021
SSP       01A1  A 2000
STA       01EB  A 2146
STA3      00B7  A 1512  2146
STX       01B7  A 2028
SUB       014C  A 1846  1924
SUB3      016C  A 1846  1920
SVC       0198  A 1991
TCA       01BC  A 2020
TEXIT     004A  A 1159  1170  1191
TIN       0040  A 1156  1162
TIN1      0042  A 1148  1162
TOUT      0043  A 1167  1175
TOUT1     0046  A 1147  1175
UII       0140  A 1816
UII3      00B3  A 1503  1511  1735  1817
VIRY      0126  A 1782
VIRY1     00C4  A 1082  1323  1419  1562  1783
VIRY12    00CF  A 1599  1605  1606  1607  1618  1630
WRITE     0062  A 1297
WRITEP    0072  A 1344
  XCA     01A2  A 1930  2002
  XCB     01A8  A 1475  1493  2002  2008
  XEC     0122  A 1777
IDNT [MACRO]
CPU  [MACRO]
RR [MACRO]
ALU  [MACRO]
FRRD$  [MACRO]
EMIT$  [MACRO]
SET$ [MACRO]
GEN$ [MACRO]
LST1$  [MACRO]
CNT$ [MACRO]
RRSV$  [MACRO]
AVF$ [MACRO]
GBP$ [MACRO]
ORG  [MACRO]
SYM$ [MACRO]
SYM1$  [MACRO]
CLST$  [MACRO]
CLOCK  [MACRO]
RR$SV  [MACRO]
RRCLK  [MACRO]
VSALU  [MACRO]
ALUCLK [MACRO]

0000 ERRORS (PMA-1080.015)
```