```
 LOC        OBJECT CODE       ADDR1     ADDR2     STMT

                                                     2 ************************************************************************
                                                     3 *
                                                     4 *Testcase IEEE Test Data Classs and Load And Test
                                                     5 *  Exhaustively test results from the Test Data Class instruction.
                                                     6 *  Exhaustively test condition code setting from Load And Test.
                                                     7 *  The Condition Code, the only result from either instruction, is
                                                     8 *  saved for comparison with reference values.
                                                     9 *
                                                    10 *
                                                    11 *                        ********************
                                                    12 *                        **   IMPORTANT!   **
                                                    13 *                        ********************
                                                    14 *
                                                    15 *       This test uses the Hercules Diagnose X'008' interface
                                                    16 *       to display messages and thus your .tst runtest script
                                                    17 *       MUST contain a "DIAG8CMD ENABLE" statement within it!
                                                    18 *
                                                    19 *
                                                    20 ************************************************************************


                                                    22 ************************************************************************
                                                    23 *
                                                    24 *                          bfp-012-loadtest.asm
                                                    25 *
                                                    26 *       This assembly-language source file is part of the
                                                    27 *         Hercules Binary Floating Point Validation Package
                                                    28 *                          by Stephen R. Orso
                                                    29 *
                                                    30 * Copyright 2016 by Stephen R Orso.
                                                    31 * Runtest *Compare dependency removed by Fish on 2022-08-16
                                                    32 * PADCSECT macro/usage removed by Fish on 2022-08-16
                                                    33 *
                                                    34 * Redistribution and use in source and binary forms, with or without
                                                    35 * modification, are permitted provided that the following conditions
                                                    36 * are met:
                                                    37 *
                                                    38 * 1. Redistributions of source code must retain the above copyright
                                                    39 *    notice, this list of conditions and the following disclaimer.
                                                    40 *
                                                    41 * 2. Redistributions in binary form must reproduce the above copyright
                                                    42 *    notice, this list of conditions and the following disclaimer in
                                                    43 *    the documentation and/or other materials provided  with the
                                                    44 *    distribution.
                                                    45 *
                                                    46 * 3. The name of the author may not be used to endorse or promote
                                                    47 *    products derived from this software without specific prior written
                                                    48 *    permission.
                                                    49 *
                                                    50 * DISCLAMER: THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDER "AS IS"
                                                    51 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
                                                    52 * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
                                                    53 * PARTICULAR PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT
                                                    54 * HOLDER BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
                                                    55 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
                                                    56 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                     57 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
                                                     58 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
                                                     59 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
                                                     60 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
                                                     61 *
                                                     62 ************************************************************************
                                                     64 ************************************************************************
                                                     65 *
                                                     66 * Neither Load And Test nor Test Data Class can result in IEEE
                                                     67 * exceptions.  All tests are performed with the FPC set to not trap
                                                     68 * on any exception.
                                                     69 *
                                                     70 * The same test data are used for both Load And Test and Test Data
                                                     71 * Class.
                                                     72 *
                                                     73 * For Load And Test, the result value and condition code are stored.
                                                     74 * For all but SNaN inputs, the result should be the same as the input.
                                                     75 * For SNaN inputs, the result is the corresponding QNaN.
                                                     76 *
                                                     77 * For Test Data Class, 13 Condition codes are stored.  The  first
                                                     78 * 12 correspond to a one-bit in each of 12 positions of the Test Data
                                                     79 * class second operand mask, and the thirteenth is generated with a
                                                     80 * mask of zero.  Test Data Class mask bits:
                                                     81 *
                                                     82 *           1 0 0 0 | 0 0 0 0 | 0 0 0 0  + zero
                                                     83 *           0 1 0 0 | 0 0 0 0 | 0 0 0 0  - zero
                                                     84 *           0 0 1 0 | 0 0 0 0 | 0 0 0 0  + finite
                                                     85 *           0 0 0 1 | 0 0 0 0 | 0 0 0 0  - finite
                                                     86 *           0 0 0 0 | 1 0 0 0 | 0 0 0 0  + tiny
                                                     87 *           0 0 0 0 | 0 1 0 0 | 0 0 0 0  - tiny
                                                     88 *           0 0 0 0 | 0 0 1 0 | 0 0 0 0  + inf
                                                     89 *           0 0 0 0 | 0 0 0 1 | 0 0 0 0  - inf
                                                     90 *           0 0 0 0 | 0 0 0 0 | 1 0 0 0  + QNaN
                                                     91 *           0 0 0 0 | 0 0 0 0 | 0 1 0 0  - QNaN
                                                     92 *           0 0 0 0 | 0 0 0 0 | 0 0 1 0  + SNaN
                                                     93 *           0 0 0 0 | 0 0 0 0 | 0 0 0 1  - SNaN
                                                     94 *
                                                     95 * Tests 3 LOAD AND TEST and 3 TEST DATA CLASS instructions
                                                     96 *   LOAD AND TEST (BFP short, RRE) LTEBR
                                                     97 *   LOAD AND TEST (BFP long, RRE) LTDBR
                                                     98 *   LOAD AND TEST (BFP extended, RRE) LTXBR
                                                     99 *   TEST DATA CLASS (BFP short, RRE) LTEBR
                                                    100 *   TEST DATA CLASS (BFP long, RRE) LTDBR
                                                    101 *   TEST DATA CLASS (BFP extended, RRE) LTXBR
                                                    102 *
                                                    103 * Also tests the following floating point support instructions
                                                    104 *   EXTRACT FPC
                                                    105 *   LOAD  (Short)
                                                    106 *   LOAD  (Long)
                                                    107 *   LOAD ZERO (Long)
                                                    108 *   STORE (Short)
                                                    109 *   STORE (Long)
                                                    110 *   SET FPC
                                                    111 *
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2     STMT
                                                     112 ************************************************************************
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                          114 *
                                          115 *  Note: for compatibility with the z/CMS test rig, do not change
                                          116 *  or use R11, R14, or R15.  Everything else is fair game.
                                          117 *
                              00000000  00005703   118 BFPLTTDC START 0
                              00000000  00000001   119 STRTLABL EQU   *
                              00000000  00000001   120 R0       EQU   0                   Work register for cc extraction
                              00000001  00000001   121 R1       EQU   1                   Current Test Data Class mask
                              00000002  00000001   122 R2       EQU   2                   Holds count of test input values
                              00000003  00000001   123 R3       EQU   3                   Points to next test input value(s)
                              00000004  00000001   124 R4       EQU   4                   Available
                              00000005  00000001   125 R5       EQU   5                   Available
                              00000006  00000001   126 R6       EQU   6                   Test Data Class top of loop
                              00000007  00000001   127 R7       EQU   7                   Ptr to next result for Load And Test
                              00000008  00000001   128 R8       EQU   8                   Ptr to next CC for Load And Test
                              00000009  00000001   129 R9       EQU   9                   Ptr to next CC for Test Data Class
                              0000000A  00000001   130 R10      EQU   10                  Pointer to test address list
                              0000000B  00000001   131 R11      EQU   11                  **Reserved for z/CMS test rig
                              0000000C  00000001   132 R12      EQU   12                  Test value top of loop
                              0000000D  00000001   133 R13      EQU   13                  Mainline return address
                              0000000E  00000001   134 R14      EQU   14                  **Return address for z/CMS test rig
                              0000000F  00000001   135 R15      EQU   15                  **Base register on z/CMS or Hyperion
                                          136 *
                                          137 * Floating Point Register equates to keep the cross reference clean
                                          138 *
                              00000000  00000001   139 FPR0     EQU   0
                              00000001  00000001   140 FPR1     EQU   1
                              00000002  00000001   141 FPR2     EQU   2
                              00000003  00000001   142 FPR3     EQU   3
                              00000004  00000001   143 FPR4     EQU   4
                              00000005  00000001   144 FPR5     EQU   5
                              00000006  00000001   145 FPR6     EQU   6
                              00000007  00000001   146 FPR7     EQU   7
                              00000008  00000001   147 FPR8     EQU   8
                              00000009  00000001   148 FPR9     EQU   9
                              0000000A  00000001   149 FPR10    EQU   10
                              0000000B  00000001   150 FPR11    EQU   11
                              0000000C  00000001   151 FPR12    EQU   12
                              0000000D  00000001   152 FPR13    EQU   13
                              0000000E  00000001   153 FPR14    EQU   14
                              0000000F  00000001   154 FPR15    EQU   15
                                          155 *
00000000                      00000000    156          USING *,R15
00000000                      00005380    157          USING HELPERS,R12
                                          158 *
                                          159 * Above works on real iron (R15=0 after sysclear)
                                          160 * and in z/CMS (R15 points to start of load module)
                                          161 *


                                          163 ********************************************************************************
                                          164 *
                                          165 * Low core definitions, Restart PSW, and Program Check Routine.
                                          166 *
                                          167 ********************************************************************************
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
00000000                     00000000  0000008E   169          ORG   STRTLABL+X'8E'      Program check interrution code
0000008E  0000                                   170 PCINTCD  DS    H
                                                 171 *
                             00000150  00000001  172 PCOLDPSW EQU   STRTLABL+X'150'     z/Arch Program check old PSW
                                                 173 *
00000090                     00000090  000001A0  174          ORG   STRTLABL+X'1A0'     z/Arch Restart PSW
000001A0  00000001 80000000                      175          DC    X'0000000180000000',AD(START)
                                                 176 *
000001B0                     000001B0  000001D0  177          ORG   STRTLABL+X'1D0'     z/Arch Program check NEW PSW
000001D0  00000000 00000000                      178          DC    X'0000000000000000',AD(PROGCHK)
                                                 179 *
                                                 180 * Program check routine.  If Data Exception, continue execution at
                                                 181 * the instruction following the program check.  Otherwise, hard wait.
                                                 182 * No need to collect data.  All interesting DXC stuff is captured
                                                 183 * in the FPCR.
                                                 184 *
000001E0                     000001E0  00000200  185          ORG   STRTLABL+X'200'
00000200                                         186 PROGCHK  DS    0H              Program check occured...
00000200  9507 F08F                    0000008F  187          CLI   PCINTCD+1,X'07'  Data Exception?
00000204  A774 0004                    0000020C  188          JNE   PCNOTDTA       ..no, hardwait (not sure if R15 is ok)
00000208  B2B2 F150                    00000150  189          LPSWE PCOLDPSW       ..yes, resume program execution

0000020C  900F F23C                    0000023C  191 PCNOTDTA STM   R0,R15,SAVEREGS  Save registers
00000210  58C0 F27C                    0000027C  192          L     R12,AHELPERS     Get address of helper subroutines
00000214  4DD0 C000                    00005380  193          BAS   R13,PGMCK        Report this unexpected program check
00000218  980F F23C                    0000023C  194          LM    R0,R15,SAVEREGS  Restore registers

0000021C  12EE                                   196          LTR   R14,R14        Return address provided?
0000021E  077E                                   197          BNZR  R14            Yes, return to z/CMS test rig.
00000220  B2B2 F228                    00000228  198          LPSWE PROGPSW        Not data exception, enter disabled wait
00000228  00020000 00000000                      199 PROGPSW  DC    0D'0',X'0002000000000000',XL6'00',X'DEAD' Abnormal end
00000238  B2B2 F2C8                    000002C8  200 FAIL     LPSWE FAILPSW        Not data exception, enter disabled wait
0000023C  00000000 00000000                      201 SAVEREGS DC    16F'0'          Registers save area
0000027C  00005380                               202 AHELPERS DC    A(HELPERS)      Address of helper subroutines
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2    STMT

                                                     204 ************************************************************************
                                                     205 *
                                                     206 *  Main program.  Enable Advanced Floating Point, process test cases.
                                                     207 *
                                                     208 ************************************************************************


00000280                                             210 START    DS    0H
00000280  B600 F2D8                        000002D8  211          STCTL R0,R0,CTLR0   Store CR0 to enable AFP
00000284  9604 F2D9                        000002D9  212          OI    CTLR0+1,X'04' Turn on AFP bit
00000288  B700 F2D8                        000002D8  213          LCTL  R0,R0,CTLR0   Reload updated CR0
                                                     214 *
0000028C  41A0 F300                        00000300  215          LA    R10,SHORTS    Point to short BFP test inputs
00000290  4DD0 F330                        00000330  216          BAS   R13,TESTSBFP  Perform short BFP tests
                                                     217 *
00000294  41A0 F310                        00000310  218          LA    R10,LONGS     Point to long BFP test inputs
00000298  4DD0 F3DC                        000003DC  219          BAS   R13,TESTLBFP  Perform long BFP tests
                                                     220 *
0000029C  41A0 F320                        00000320  221          LA    R10,EXTDS     Point to extended BFP test inputs
000002A0  4DD0 F488                        00000488  222          BAS   R13,TESTXBFP  Perform short BFP tests
                                                     223 *
                                                     224 ************************************************************************
                                                     225 *                    Verify test results...
                                                     226 ************************************************************************
                                                     227 *
000002A4  58C0 F27C                        0000027C  228          L     R12,AHELPERS     Get address of helper subroutines
000002A8  4DD0 C0A0                        00005420  229          BAS   R13,VERISUB      Go verify results
000002AC  12EE                                       230          LTR   R14,R14          Was return address provided?
000002AE  077E                                       231          BNZR  R14              Yes, return to z/CMS test rig.
000002B0  B2B2 F2B8                        000002B8  232          LPSWE GOODPSW          Load SUCCESS PSW
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

000002B8                                          234          DS    0D            Ensure correct alignment for PSW
000002B8  00020000 00000000                       235 GOODPSW  DC    X'0002000000000000',AD(0)  Normal end - disabled wait
000002C8  00020000 00000000                       236 FAILPSW  DC    X'0002000000000000',XL6'00',X'0BAD' Abnormal end
                                                  237 *
000002D8  00000000                                238 CTLR0    DS    F
000002DC  00000000                                239 FPCREGNT DC    X'00000000'   FPCR, trap no IEEE exceptions, zero flags
000002E0  F8000000                                240 FPCREGTR DC    X'F8000000'   FPCR, trap all IEEE exceptions, zero flags
                                                  241 *
                                                  242 * Input values parameter list, four fullwords for each test data set
                                                  243 *      1) Count,
                                                  244 *      2) Address of inputs,
                                                  245 *      3) Address to place results, and
                                                  246 *      4) Address to place DXC/Flags/cc values.
                                                  247 *
000002E4                        000002E4  00000300 248          ORG   STRTLABL+X'300'  Enable run-time replacement
00000300                                          249 SHORTS   DS    0F           Input pairs for short BFP ests
00000300  0000000C                                250          DC    A(SBFPINCT)
00000304  0000055C                                251          DC    A(SBFPIN)
00000308  00001000                                252          DC    A(SBFPOUT)
0000030C  00001100                                253          DC    A(SBFPOCC)
                                                  254 *
00000310                                          255 LONGS    DS    0F           Input pairs for long BFP testing
00000310  0000000C                                256          DC    A(LBFPINCT)
00000314  00000590                                257          DC    A(LBFPIN)
00000318  00002000                                258          DC    A(LBFPOUT)
0000031C  00002100                                259          DC    A(LBFPOCC)
                                                  260 *
00000320                                          261 EXTDS    DS    0F           Input pairs for extendedd BFP testing
00000320  0000000C                                262          DC    A(XBFPINCT)
00000324  000005F8                                263          DC    A(XBFPIN)
00000328  00003000                                264          DC    A(XBFPOUT)
0000032C  00003200                                265          DC    A(XBFPOCC)
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                    267 ********************************************************************************
                                                    268 * Perform Short BFP Tests.  This includes one execution of Load And
                                                    269 * Test, followed by 13 executions of Test Data Class.  The result value
                                                    270 * and Condition code are saved for Load And Test, and the Condition
                                                    271 * Code is saved for each execution of Test Data Class.
                                                    272 *
                                                    273 ********************************************************************************


00000330                                            275 TESTSBFP DS    0H            Test short BFP input values
00000330  9823 A000                 00000000       276          LM    R2,R3,0(R10)  Get count and address of test input values
00000334  9878 A008                 00000008       277          LM    R7,R8,8(R10)  Get address of result and CC areas.
00000338  1222                                      278          LTR   R2,R2         Any test cases?
0000033A  078D                                      279          BZR   R13           ..No, return to caller
0000033C  0DC0                                      280          BASR  R12,0         Set top of loop
                                                    281 *
0000033E  7880 3000                 00000000       282          LE    FPR8,0(,R3)   Get short BFP test value
                                                    283 *                            Polute the CC result area.  Correct
                                                    284 *                            ..results will clean it up.
00000342  D20F 8000 F540  00000000  00000540       285          MVC   0(16,R8),=X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
                                                    286 *
00000348  B29D F2DC                 000002DC       287          LFPC  FPCREGNT      Set exceptions non-trappable
0000034C  7810 F58C                 0000058C       288          LE    FPR1,SBFPINVL Ensure an unchanged FPR1 is detectable
00000350  B222 0000                                 289          IPM   R0            Get current program mask and CC
00000354  5400 F550                 00000550       290          N     R0,=X'CFFFFFFF' Turn off condition code bits
00000358  5600 F554                 00000554       291          O     R0,=X'20000000' Force condition code two
0000035C  0400                                      292          SPM   R0            Set PSW CC to two
0000035E  B302 0018                                 293          LTEBR FPR1,FPR8     Load and Test into FPR1
00000362  7010 7000                 00000000       294          STE   FPR1,0(,R7)   Store short BFP result
00000366  B222 0000                                 295          IPM   R0            Retrieve condition code
0000036A  8800 001C                 0000001C       296          SRL   R0,28         Move CC to low-order r0, dump prog mask
0000036E  4200 8000                 00000000       297          STC   R0,0(,R8)     Store in CC result area
                                                    298 *
00000372  B29D F2E0                 000002E0       299          LFPC  FPCREGTR      Set exceptions non-trappable
00000376  7810 F58C                 0000058C       300          LE    FPR1,SBFPINVL Ensure an unchanged FPR1 is detectable
0000037A  B222 0000                                 301          IPM   R0            Get current program mask and CC
0000037E  5400 F550                 00000550       302          N     R0,=X'CFFFFFFF' Turn off condition code bits
00000382  5600 F554                 00000554       303          O     R0,=X'20000000' Force condition code two
00000386  0400                                      304          SPM   R0            Set PSW CC to two
00000388  B302 0018                                 305          LTEBR FPR1,FPR8     Load and Test into FPR1
0000038C  7010 7004                 00000004       306          STE   FPR1,4(,R7)   Store short BFP result
00000390  B222 0000                                 307          IPM   R0            Retrieve condition code
00000394  8800 001C                 0000001C       308          SRL   R0,28         Move CC to low-order r0, dump prog mask
00000398  4200 8001                 00000001       309          STC   R0,1(,R8)     Store in CC result area
0000039C  B38C 0000                                 310          EFPC  R0            Extract FPC contents to R0
000003A0  BE02 8002                 00000002       311          STCM  R0,B'0010',2(R8)  Store any DXC code
                                                    312 *
000003A4  A718 1000                                 313          LHI   R1,4096       Load Test Data Class mask starting point
000003A8  4190 8003                 00000003       314          LA    R9,3(,R8)     Point to first Test Data Class CC
000003AC  0D60                                      315          BASR  R6,0          Set start of Test Data Class loop
                                                    316 *
000003AE  8810 0001                 00000001       317          SRL   R1,1          Shift to get next class mask value
000003B2  ED80 1000 0010            00000000       318          TCEB  FPR8,0(,R1)   Test value against class mask
000003B8  B222 0000                                 319          IPM   R0            Retrieve condition code
000003BC  8800 001C                 0000001C       320          SRL   R0,28         Move CC to low-order r0, dump prog mask
000003C0  4200 9000                 00000000       321          STC   R0,0(,R9)     Store in CC result area
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

000003C4  4190 9001                    00000001   322          LA    R9,1(,R9)     Point to next CC slot
000003C8  1211                         00000000   323          LTR   R1,R1         Have we tested all masks including zero
000003CA  0776                         00000000   324          BNZR  R6            ..no, at least one more to test
                                       00000000   325
000003CC  4130 3004                    00000004   326          LA    R3,4(,R3)     Point to next short BFP test value
000003D0  4170 7008                    00000008   327          LA    R7,8(,R7)     Point to next Load And Test result pair
000003D4  4180 8010                    00000010   328          LA    R8,16(,R8)    Point to next CC result set
000003D8  062C                         00000000   329          BCTR  R2,R12        Loop through all test cases
                                       00000000   330 *
000003DA  07FD                         00000000   331          BR    R13           Tests done, return to mainline
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                                    333 ********************************************************************************
                                                    334 * Perform long BFP Tests.  This includes one execution of Load And
                                                    335 * Test, followed by 13 executions of Test Data Class.  The result value
                                                    336 * and Condition code are saved for Load And Test, and the Condition
                                                    337 * Code is saved for each execution of Test Data Class.
                                                    338 *
                                                    339 ********************************************************************************


000003DC                                            341 TESTLBFP DS    0H            Test long BFP input values
000003DC  9823 A000                 00000000    342          LM    R2,R3,0(R10)  Get count and address of test input values
000003E0  9878 A008                 00000008    343          LM    R7,R8,8(R10)  Get address of result and CC areas.
000003E4  1222                                  344          LTR   R2,R2         Any test cases?
000003E6  078D                                  345          BZR   R13           ..No, return to caller
000003E8  0DC0                                  346          BASR  R12,0         Set top of loop
                                                    347 *
000003EA  6880 3000                 00000000    348          LD    FPR8,0(,R3)   Get long BFP test value
                                                    349 *                         Polute the CC result area.  Correct
                                                    350 *                         ..results will clean it up.
000003EE  D20F 8000 F540  00000000  00000540    351          MVC   0(16,R8),=X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
                                                    352 *
000003F4  B29D F2DC                 000002DC    353          LFPC  FPCREGNT      Set exceptions non-trappable
000003F8  6810 F5F0                 000005F0    354          LD    FPR1,LBFPINVL Ensure an unchanged FPR1 is detectable
000003FC  B222 0000                             355          IPM   R0            Get current program mask and CC
00000400  5400 F550                 00000550    356          N     R0,=X'CFFFFFFF' Turn off condition code bits
00000404  5600 F558                 00000558    357          O     R0,=X'10000000' Force condition code one
00000408  0400                                  358          SPM   R0            Set PSW CC to one
0000040A  B312 0018                             359          LTDBR FPR1,FPR8     Load and Test into FPR1
0000040E  6010 7000                 00000000    360          STD   FPR1,0(,R7)   Store long BFP result
00000412  B222 0000                             361          IPM   R0            Retrieve condition code
00000416  8800 001C                 0000001C    362          SRL   R0,28         Move CC to low-order r0, dump prog mask
0000041A  4200 8000                 00000000    363          STC   R0,0(,R8)     Store in CC result area
                                                    364 *
0000041E  B29D F2E0                 000002E0    365          LFPC  FPCREGTR      Set exceptions trappable
00000422  6810 F5F0                 000005F0    366          LD    FPR1,LBFPINVL Ensure an unchanged FPR1 is detectable
00000426  B222 0000                             367          IPM   R0            Get current program mask and CC
0000042A  5400 F550                 00000550    368          N     R0,=X'CFFFFFFF' Turn off condition code bits
0000042E  5600 F558                 00000558    369          O     R0,=X'10000000' Force condition code one
00000432  0400                                  370          SPM   R0            Set PSW CC to one
00000434  B312 0018                             371          LTDBR FPR1,FPR8     Load and Test into FPR1
00000438  6010 7008                 00000008    372          STD   FPR1,8(,R7)   Store long BFP result
0000043C  B222 0000                             373          IPM   R0            Retrieve condition code
00000440  8800 001C                 0000001C    374          SRL   R0,28         Move CC to low-order r0, dump prog mask
00000444  4200 8001                 00000001    375          STC   R0,1(,R8)     Store in CC result area
00000448  B38C 0000                             376          EFPC  R0            Extract FPC contents to R0
0000044C  BE02 8002                 00000002    377          STCM  R0,B'0010',2(R8)  Store any DXC code
                                                    378 *
00000450  A718 1000                             379          LHI   R1,4096       Load Test Data Class mask starting point
00000454  4190 8003                 00000003    380          LA    R9,3(,R8)     Point to first Test Data Class CC
00000458  0D60                                  381          BASR  R6,0          Set start of Test Data Class loop
                                                    382
0000045A  8810 0001                 00000001    383          SRL   R1,1          Shift to get next class mask value
0000045E  ED80 1000 0011            00000000    384          TCDB  FPR8,0(,R1)   Test value against class mask
00000464  B222 0000                             385          IPM   R0            Retrieve condition code
00000468  8800 001C                 0000001C    386          SRL   R0,28         Move CC to low-order r0, dump prog mask
0000046C  4200 9000                 00000000    387          STC   R0,0(,R9)     Store in CC result area
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

00000470  4190 9001                      00000001   388          LA    R9,1(,R9)     Point to next CC slot
00000474  1211                                       389          LTR   R1,R1         Have we tested all masks including zero
00000476  0776                                       390          BNZR  R6            ..no, at least one more to test
                                                     391
00000478  4130 3008                      00000008    392          LA    R3,8(,R3)     Point to next long BFP test value
0000047C  4170 7010                      00000010    393          LA    R7,16(,R7)    Point to next Load And Test result pair
00000480  4180 8010                      00000010    394          LA    R8,16(,R8)    Point to next CC result set
00000484  062C                                       395          BCTR  R2,R12        Loop through all test cases
                                                     396 *
00000486  07FD                                       397          BR    R13           Tests done, return to mainline
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                    399 ************************************************************************
                                                    400 * Perform extended BFP Tests.  This includes one execution of Load And
                                                    401 * Test, followed by 13 executions of Test Data Class.  The result value
                                                    402 * and Condition code are saved for Load And Test, and the Condition
                                                    403 * Code is saved for each execution of Test Data Class.
                                                    404 *
                                                    405 ************************************************************************


00000488                                            407 TESTXBFP DS    0H            Test extended BFP input values
00000488  9823 A000                 00000000       408          LM    R2,R3,0(R10)  Get count and address of test input values
0000048C  9878 A008                 00000008       409          LM    R7,R8,8(R10)  Get address of result and CC areas.
00000490  1222                                      410          LTR   R2,R2         Any test cases?
00000492  078D                                      411          BZR   R13           ..No, return to caller
00000494  0DC0                                      412          BASR  R12,0         Set top of loop
                                                    413 *
00000496  6880 3000                 00000000       414          LD    FPR8,0(,R3)   Get extended BFP test value part 1
0000049A  68A0 3008                 00000008       415          LD    FPR10,8(,R3)  Get extended BFP test value part 2
                                                    416 *                           Polute the CC result area.  Correct
                                                    417 *                           ..results will clean it up.
0000049E  D20F 8000 F540  00000000  00000540       418          MVC   0(16,R8),=X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
                                                    419 *
000004A4  B29D F2DC                 000002DC       420          LFPC  FPCREGNT      Set exceptions non-trappable
000004A8  6810 F6B8                 000006B8       421          LD    FPR1,XBFPINVL   Ensure an unchanged FPR1-3 is detectable
000004AC  6830 F6C0                 000006C0       422          LD    FPR3,XBFPINVL+8 ..part 2 of load FPR pair
000004B0  B222 0000                                 423          IPM   R0            Get current program mask and CC
000004B4  5400 F550                 00000550       424          N     R0,=X'CFFFFFFF' Turn off condition code bits
000004B8  0400                                      425          SPM   R0            Set PSW CC to zero
000004BA  B342 0018                                 426          LTXBR FPR1,FPR8     Load and Test into FPR1
000004BE  6010 7000                 00000000       427          STD   FPR1,0(,R7)   Store extended BFP result part 1
000004C2  6030 7008                 00000008       428          STD   FPR3,8(,R7)   Store extended BFP result part 2
000004C6  B222 0000                                 429          IPM   R0            Retrieve condition code
000004CA  8800 001C                 0000001C       430          SRL   R0,28         Move CC to low-order r0, dump prog mask
000004CE  4200 8000                 00000000       431          STC   R0,0(,R8)     Store in CC result area
                                                    432 *
000004D2  B29D F2E0                 000002E0       433          LFPC  FPCREGTR      Set exceptions trappable
000004D6  6810 F6B8                 000006B8       434          LD    FPR1,XBFPINVL   Ensure an unchanged FPR1-3 is detectable
000004DA  6830 F6C0                 000006C0       435          LD    FPR3,XBFPINVL+8 ..part 2 of load FPR pair
000004DE  B222 0000                                 436          IPM   R0            Get current program mask and CC
000004E2  5400 F550                 00000550       437          N     R0,=X'CFFFFFFF' Turn off condition code bits
000004E6  0400                                      438          SPM   R0            Set PSW CC to zero
000004E8  B342 0018                                 439          LTXBR FPR1,FPR8     Load and Test into FPR1
000004EC  6010 7010                 00000010       440          STD   FPR1,16(,R7)  Store extended BFP result part 1
000004F0  6030 7018                 00000018       441          STD   FPR3,24(,R7)  Store extended BFP result part 2
000004F4  B222 0000                                 442          IPM   R0            Retrieve condition code
000004F8  8800 001C                 0000001C       443          SRL   R0,28         Move CC to low-order r0, dump prog mask
000004FC  4200 8001                 00000001       444          STC   R0,1(,R8)     Store in CC result area
00000500  B38C 0000                                 445          EFPC  R0            Extract FPC contents to R0
00000504  BE02 8002                 00000002       446          STCM  R0,B'0010',2(R8)  Store any DXC code
                                                    447 *
00000508  A718 1000                                 448          LHI   R1,4096       Load Test Data Class mask starting point
0000050C  4190 8003                 00000003       449          LA    R9,3(,R8)     Point to first Test Data Class CC
00000510  0D60                                      450          BASR  R6,0          Set start of Test Data Class loop
                                                    451
00000512  8810 0001                 00000001       452          SRL   R1,1          Shift to get next class mask value
00000516  ED80 1000 0012            00000000       453          TCXB  FPR8,0(,R1)   Test value against class mask
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

0000051C  B222 0000                            454          IPM   R0            Retrieve condition code
00000520  8800 001C                 0000001C   455          SRL   R0,28         Move CC to low-order r0, dump prog mask
00000524  4200 9000                 00000000   456          STC   R0,0(,R9)     Store in last byte of FPCR
00000528  4190 9001                 00000001   457          LA    R9,1(,R9)     Point to next CC slot
0000052C  1211                                 458          LTR   R1,R1         Have we tested all masks including zero
0000052E  0776                                 459          BNZR  R6            ..no, at least one more to test
                                               460
00000530  4130 3010                 00000010   461          LA    R3,16(,R3)    Point to next extended BFP test value
00000534  4170 7020                 00000020   462          LA    R7,32(,R7)    Point to next Load And Test result pair
00000538  4180 8010                 00000010   463          LA    R8,16(,R8)    Point to next CC result set
0000053C  062C                                 464          BCTR  R2,R12        Loop through all test cases
                                               465 *
0000053E  07FD                                 466          BR    R13           Tests done, return to mainline
                                               467 *
00000540                                       468          LTORG
00000540  FFFFFFFF FFFFFFFF                    469                =X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
00000550  CFFFFFFF                             470                =X'CFFFFFFF'
00000554  20000000                            471                =X'20000000'
00000558  10000000                            472                =X'10000000'
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2     STMT
                                                    474 ************************************************************************
                                                    475 *
                                                    476 * Short integer inputs for Load And Test and Test Data Class.  The same
                                                    477 * values are used for short, long, and extended formats.
                                                    478 *
                                                    479 ************************************************************************


0000055C                                            481 SBFPIN   DS    0F             Ensure fullword alignment for input table
0000055C  00000000                                  482          DC    X'00000000'      +0
00000560  80000000                                  483          DC    X'80000000'      -0
00000564  3F800000                                  484          DC    X'3F800000'      +1
00000568  BF800000                                  485          DC    X'BF800000'      -1
0000056C  007FFFFF                                  486          DC    X'007FFFFF'      +subnormal
00000570  807FFFFF                                  487          DC    X'807FFFFF'      -subnormal
00000574  7F800000                                  488          DC    X'7F800000'      +infinity
00000578  FF800000                                  489          DC    X'FF800000'      -infinity
0000057C  7FC00000                                  490          DC    X'7FC00000'      +QNaN
00000580  FFC00000                                  491          DC    X'FFC00000'      -QNaN
00000584  7F810000                                  492          DC    X'7F810000'      +SNaN
00000588  FF810000                                  493          DC    X'FF810000'      -SNaN
                            0000000C  00000001      494 SBFPINCT EQU   (*-SBFPIN)/4  count of short BFP test values
                                                    495 *
0000058C  0000DEAD                                  496 SBFPINVL DC    X'0000DEAD'   Invalid result, used to polute result FPR


                                                    498 ************************************************************************
                                                    499 *
                                                    500 * Long integer inputs for Load And Test and Test Data Class.  The same
                                                    501 * values are used for short, long, and extended formats.
                                                    502 *
                                                    503 ************************************************************************


00000590                                            505 LBFPIN   DS    0D
00000590  00000000 00000000                         506          DC    X'0000000000000000'      +0
00000598  80000000 00000000                         507          DC    X'8000000000000000'      -0
000005A0  3FF00000 00000000                         508          DC    X'3FF0000000000000'      +1
000005A8  BFF00000 00000000                         509          DC    X'BFF0000000000000'      -1
000005B0  000FFFFF FFFFFFFF                         510          DC    X'000FFFFFFFFFFFFF'      +subnormal
000005B8  800FFFFF FFFFFFFF                         511          DC    X'800FFFFFFFFFFFFF'      -subnormal
000005C0  7FF00000 00000000                         512          DC    X'7FF0000000000000'      +infinity
000005C8  FFF00000 00000000                         513          DC    X'FFF0000000000000'      -infinity
000005D0  7FF80000 00000000                         514          DC    X'7FF8000000000000'      +QNaN
000005D8  FFF80000 00000000                         515          DC    X'FFF8000000000000'      -QNaN
000005E0  7FF01000 00000000                         516          DC    X'7FF0100000000000'      +SNaN
000005E8  FFF01000 00000000                         517          DC    X'FFF0100000000000'      -SNaN
                            0000000C  00000001      518 LBFPINCT EQU   (*-LBFPIN)/8  count of long BFP test values
                                                    519 *
000005F0  0000DEAD 00000000                         520 LBFPINVL DC    X'0000DEAD00000000'  Invalid result, used to
                                                    521 *                                   ..polute result FPR


                                                    523 ************************************************************************
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2    STMT

                                                     524 *
                                                     525 * Extended integer inputs for Load And Test and Test Data Class.  The
                                                     526 * same values are used for short, long, and extended formats.
                                                     527 *
                                                     528 ********************************************************************************



000005F8                                             530 XBFPIN   DS    0D
000005F8  00000000 00000000                          531          DC    X'00000000000000000000000000000000' +0
00000608  80000000 00000000                          532          DC    X'80000000000000000000000000000000' -0
00000618  3FFF0000 00000000                          533          DC    X'3FFF0000000000000000000000000000' +1
00000628  BFFF0000 00000000                          534          DC    X'BFFF0000000000000000000000000000' -1
00000638  0000FFFF FFFFFFFF                          535          DC    X'0000FFFFFFFFFFFFFFFFFFFFFFFFFFFF' +subnormal
00000648  8000FFFF FFFFFFFF                          536          DC    X'8000FFFFFFFFFFFFFFFFFFFFFFFFFFFF' -subnormal
00000658  7FFF0000 00000000                          537          DC    X'7FFF0000000000000000000000000000' +infinity
00000668  FFFF0000 00000000                          538          DC    X'FFFF0000000000000000000000000000' -infinity
00000678  7FFF8000 00000000                          539          DC    X'7FFF8000000000000000000000000000' +QNaN
00000688  FFFF8000 00000000                          540          DC    X'FFFF8000000000000000000000000000' -QNaN
00000698  7FFF0100 00000000                          541          DC    X'7FFF0100000000000000000000000000' +SNaN
000006A8  FFFF0100 00000000                          542          DC    X'FFFF0100000000000000000000000000' -SNaN
                             0000000C  00000001      543 XBFPINCT EQU   (*-XBFPIN)/16  count of extended BFP test values
                                                     544 *
000006B8  0000DEAD 00000000                          545 XBFPINVL DC    X'0000DEAD000000000000000000000000'  Invalid result, used to
                                                     546 *                                  ..used to polute result FPR
```

```
 LOC        OBJECT CODE       ADDR1     ADDR2     STMT
                                          548 ********************************************************************************
                                          549 *                    ACTUAL results saved here
                                          550 ********************************************************************************
                                          551 *
                                          552 *                 Locations for ACTUAL results
                                          553 *
                                          554 *
                            00001000  00000001  555 SBFPOUT  EQU   STRTLABL+X'1000'    Integer short BFP Load and Test
                                          556 *                                      ..12 used, room for 60 tests
                            00001100  00000001  557 SBFPOCC  EQU   STRTLABL+X'1100'    Condition codes from Load and Test
                                          558 *                                      ..and Test Data Class.
                                          559 *                                      ..12 sets used, room for 60 sets
                                          560 *                                      ..next available is X'1500'
                                          561 *
                            00002000  00000001  562 LBFPOUT  EQU   STRTLABL+X'2000'    Integer long BFP Load And Test
                                          563 *                                      ..12 used, room for 32 tests,
                            00002100  00000001  564 LBFPOCC  EQU   STRTLABL+X'2100'     Condition codes from Load and Test
                                          565 *                                      ..and Test Data Class.
                                          566 *                                      ..12 sets used, room for 32 sets
                                          567 *                                      ..next available is X'2300'
                                          568 *
                            00003000  00000001  569 XBFPOUT  EQU   STRTLABL+X'3000'    Integer extended BFP Load And Test
                                          570 *                                      ..12 used, room for 32 tests,
                            00003200  00000001  571 XBFPOCC  EQU   STRTLABL+X'3200'    Condition codes from Load and Test
                                          572 *                                      ..and Test Data Class.
                                          573 *                                      ..12 sets used, room for 32 sets
                                          574 *                                      ..next available is X'3400'
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                                    576 ********************************************************************************
                                                    577 *                       EXPECTED results
                                                    578 ********************************************************************************
                                                    579 *
000006C8                      000006C8  00004000    580          ORG   STRTLABL+X'4000'   (past end of actual results)
                                                    581 *
                              00004000  00000001    582 SBFPOUT_GOOD EQU *
00004000  D3E3C5C2 D9404E61                         583  DC CL48'LTEBR +/-0'
00004030  00000000 00000000                         584  DC XL16'00000000000000008000000080000000'
00004040  D3E3C5C2 D9404E61                         585  DC CL48'LTEBR +/-1'
00004070  3F800000 3F800000                         586  DC XL16'3F8000003F800000BF800000BF800000'
00004080  D3E3C5C2 D9404E61                         587  DC CL48'LTEBR +/-tiny'
000040B0  007FFFFF 007FFFFF                         588  DC XL16'007FFFFF007FFFFF807FFFFF807FFFFF'
000040C0  D3E3C5C2 D9404E61                         589  DC CL48'LTEBR +/-inf'
000040F0  7F800000 7F800000                         590  DC XL16'7F8000007F800000FF800000FF800000'
00004100  D3E3C5C2 D9404E61                         591  DC CL48'LTEBR +/-QNaN'
00004130  7FC00000 7FC00000                         592  DC XL16'7FC000007FC00000FFC00000FFC00000'
00004140  D3E3C5C2 D9404E61                         593  DC CL48'LTEBR +/-SNaN'
00004170  7FC10000 0000DEAD                         594  DC XL16'7FC100000000DEADFFC100000000DEAD'
                              00000006  00000001    595 SBFPOUT_NUM EQU (*-SBFPOUT_GOOD)/64
                                                    596 *
                                                    597 *
                              00004180  00000001    598 SBFPOCC_GOOD EQU *
00004180  E3C3C5C2 40C3C340                         599  DC CL48'TCEB CC +0'
000041B0  00000001 00000000                         600  DC XL16'00000001000000000000000000000000'
000041C0  E3C3C5C2 40C3C340                         601  DC CL48'TCEB CC -0'
000041F0  00000000 01000000                         602  DC XL16'00000000100000000000000000000000'
00004200  E3C3C5C2 40C3C340                         603  DC CL48'TCEB CC +1'
00004230  02020000 00010000                         604  DC XL16'02020000000100000000000000000000'
00004240  E3C3C5C2 40C3C340                         605  DC CL48'TCEB CC -1'
00004270  01010000 00000100                         606  DC XL16'01010000000001000000000000000000'
00004280  E3C3C5C2 40C3C340                         607  DC CL48'TCEB CC +tiny'
000042B0  02020000 00000001                         608  DC XL16'02020000000000010000000000000000'
000042C0  E3C3C5C2 40C3C340                         609  DC CL48'TCEB CC -tiny'
000042F0  01010000 00000000                         610  DC XL16'01010000000000000100000000000000'
00004300  E3C3C5C2 40C3C340                         611  DC CL48'TCEB CC +inf'
00004330  02020000 00000000                         612  DC XL16'02020000000000000001000000000000'
00004340  E3C3C5C2 40C3C340                         613  DC CL48'TCEB CC -inf'
00004370  01010000 00000000                         614  DC XL16'01010000000000000000010000000000'
00004380  E3C3C5C2 40C3C340                         615  DC CL48'TCEB CC +QNaN'
000043B0  03030000 00000000                         616  DC XL16'03030000000000000000000100000000'
000043C0  E3C3C5C2 40C3C340                         617  DC CL48'TCEB CC -QNaN'
000043F0  03030000 00000000                         618  DC XL16'03030000000000000000000001000000'
00004400  E3C3C5C2 40C3C340                         619  DC CL48'TCEB CC +SNaN'
00004430  03028000 00000000                         620  DC XL16'03028000000000000000000000010000'
00004440  E3C3C5C2 40C3C340                         621  DC CL48'TCEB CC -SNaN'
00004470  03028000 00000000                         622  DC XL16'03028000000000000000000000000100'
                              0000000C  00000001    623 SBFPOCC_NUM EQU (*-SBFPOCC_GOOD)/64
                                                    624 *
                                                    625 *
                              00004480  00000001    626 LBFPOUT_GOOD EQU *
00004480  D3E3C4C2 D9404EF0                         627  DC CL48'LTDBR +0'
000044B0  00000000 00000000                         628  DC XL16'00000000000000000000000000000000'
000044C0  D3E3C4C2 D94060F0                         629  DC CL48'LTDBR -0'
000044F0  80000000 00000000                         630  DC XL16'80000000000000008000000000000000'
00004500  D3E3C4C2 D9404EF1                         631  DC CL48'LTDBR +1'
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

00004530  3FF00000 00000000                        632  DC XL16'3FF00000000000003FF0000000000000'
00004540  D3E3C4C2 D94060F1                        633  DC CL48'LTDBR -1'
00004570  BFF00000 00000000                        634  DC XL16'BFF0000000000000BFF0000000000000'
00004580  D3E3C4C2 D9404EA3                        635  DC CL48'LTDBR +tiny'
000045B0  000FFFFF FFFFFFFF                        636  DC XL16'000FFFFFFFFFFFF000FFFFFFFFFFFF'
000045C0  D3E3C4C2 D94060A3                        637  DC CL48'LTDBR -tiny'
000045F0  800FFFFF FFFFFFFF                        638  DC XL16'800FFFFFFFFFFFF800FFFFFFFFFFFF'
00004600  D3E3C4C2 D9404E89                        639  DC CL48'LTDBR +inf'
00004630  7FF00000 00000000                        640  DC XL16'7FF00000000000007FF0000000000000'
00004640  D3E3C4C2 D9406089                        641  DC CL48'LTDBR -inf'
00004670  FFF00000 00000000                        642  DC XL16'FFF0000000000000FFF0000000000000'
00004680  D3E3C4C2 D9404ED8                        643  DC CL48'LTDBR +QNaN'
000046B0  7FF80000 00000000                        644  DC XL16'7FF80000000000007FF8000000000000'
000046C0  D3E3C4C2 D94060D8                        645  DC CL48'LTDBR -QNaN'
000046F0  FFF80000 00000000                        646  DC XL16'FFF8000000000000FFF8000000000000'
00004700  D3E3C4C2 D9404EE2                        647  DC CL48'LTDBR +SNaN'
00004730  7FF81000 00000000                        648  DC XL16'7FF810000000000000DEAD00000000'
00004740  D3E3C4C2 D94060E2                        649  DC CL48'LTDBR -SNaN'
00004770  FFF81000 00000000                        650  DC XL16'FFF810000000000000DEAD00000000'
                             0000000C  00000001    651 LBFPOUT_NUM EQU (*-LBFPOUT_GOOD)/64
                                                   652 *
                                                   653 *
                             00004780  00000001    654 LBFPOCC_GOOD EQU *
00004780  E3C3C4C2 40C3C340                        655  DC CL48'TCDB CC +0'
000047B0  00000001 00000000                        656  DC XL16'00000001000000000000000000000000'
000047C0  E3C3C4C2 40C3C340                        657  DC CL48'TCDB CC -0'
000047F0  00000000 01000000                        658  DC XL16'00000000100000000000000000000000'
00004800  E3C3C4C2 40C3C340                        659  DC CL48'TCDB CC +1'
00004830  02020000 00010000                        660  DC XL16'02020000000100000000000000000000'
00004840  E3C3C4C2 40C3C340                        661  DC CL48'TCDB CC -1'
00004870  01010000 00000100                        662  DC XL16'01010000000001000000000000000000'
00004880  E3C3C4C2 40C3C340                        663  DC CL48'TCDB CC +tiny'
000048B0  02020000 00000001                        664  DC XL16'02020000000000010000000000000000'
000048C0  E3C3C4C2 40C3C340                        665  DC CL48'TCDB CC -tiny'
000048F0  01010000 00000000                        666  DC XL16'01010000000000000100000000000000'
00004900  E3C3C4C2 40C3C340                        667  DC CL48'TCDB CC +inf'
00004930  02020000 00000000                        668  DC XL16'02020000000000000001000000000000'
00004940  E3C3C4C2 40C3C340                        669  DC CL48'TCDB CC -inf'
00004970  01010000 00000000                        670  DC XL16'01010000000000000000010000000000'
00004980  E3C3C4C2 40C3C340                        671  DC CL48'TCDB CC +QNaN'
000049B0  03030000 00000000                        672  DC XL16'03030000000000000000000100000000'
000049C0  E3C3C4C2 40C3C340                        673  DC CL48'TCDB CC -QNaN'
000049F0  03030000 00000000                        674  DC XL16'03030000000000000000000001000000'
00004A00  E3C3C4C2 40C3C340                        675  DC CL48'TCDB CC +SNaN'
00004A30  03018000 00000000                        676  DC XL16'03018000000000000000000000010000'
00004A40  E3C3C4C2 40C3C340                        677  DC CL48'TCDB CC -SNaN'
00004A70  03018000 00000000                        678  DC XL16'03018000000000000000000000000100'
                             0000000C  00000001    679 LBFPOCC_NUM EQU (*-LBFPOCC_GOOD)/64
                                                   680 *
                                                   681 *
                             00004A80  00000001    682 XBFPOUT_GOOD EQU *
00004A80  D3E3E7C2 D9404E61                        683  DC CL48'LTXBR +/0 NT'
00004AB0  00000000 00000000                        684  DC XL16'00000000000000000000000000000000'
00004AC0  D3E3E7C2 D9404E61                        685  DC CL48'LTXBR +/0 TR'
00004AF0  00000000 00000000                        686  DC XL16'00000000000000000000000000000000'
00004B00  D3E3E7C2 D94060F0                        687  DC CL48'LTXBR -0 NT'
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

00004B30  80000000 00000000                    688  DC XL16'80000000000000000000000000000000'
00004B40  D3E3E7C2 D94060F0                    689  DC CL48'LTXBR -0 TR'
00004B70  80000000 00000000                    690  DC XL16'80000000000000000000000000000000'
00004B80  D3E3E7C2 D9404EF1                    691  DC CL48'LTXBR +1 NT'
00004BB0  3FFF0000 00000000                    692  DC XL16'3FFF0000000000000000000000000000'
00004BC0  D3E3E7C2 D9404EF1                    693  DC CL48'LTXBR +1 TR'
00004BF0  3FFF0000 00000000                    694  DC XL16'3FFF0000000000000000000000000000'
00004C00  D3E3E7C2 D94060F1                    695  DC CL48'LTXBR -1 NT'
00004C30  BFFF0000 00000000                    696  DC XL16'BFFF0000000000000000000000000000'
00004C40  D3E3E7C2 D94060F1                    697  DC CL48'LTXBR -1 TR'
00004C70  BFFF0000 00000000                    698  DC XL16'BFFF0000000000000000000000000000'
00004C80  D3E3E7C2 D9404EA3                    699  DC CL48'LTXBR +tiny NT'
00004CB0  0000FFFF FFFFFFFF                    700  DC XL16'0000FFFFFFFFFFFFFFFFFFFFFFFFFFFF'
00004CC0  D3E3E7C2 D9404EA3                    701  DC CL48'LTXBR +tiny'
00004CF0  0000FFFF FFFFFFFF                    702  DC XL16'0000FFFFFFFFFFFFFFFFFFFFFFFFFFFF'
00004D00  D3E3E7C2 D94060A3                    703  DC CL48'LTXBR -tiny NT'
00004D30  8000FFFF FFFFFFFF                    704  DC XL16'8000FFFFFFFFFFFFFFFFFFFFFFFFFFFF'
00004D40  D3E3E7C2 D94060A3                    705  DC CL48'LTXBR -tiny TR'
00004D70  8000FFFF FFFFFFFF                    706  DC XL16'8000FFFFFFFFFFFFFFFFFFFFFFFFFFFF'
00004D80  D3E3E7C2 D9404E89                    707  DC CL48'LTXBR +inf NT'
00004DB0  7FFF0000 00000000                    708  DC XL16'7FFF0000000000000000000000000000'
00004DC0  D3E3E7C2 D9404E89                    709  DC CL48'LTXBR +inf TR'
00004DF0  7FFF0000 00000000                    710  DC XL16'7FFF0000000000000000000000000000'
00004E00  D3E3E7C2 D9406089                    711  DC CL48'LTXBR -inf NT'
00004E30  FFFF0000 00000000                    712  DC XL16'FFFF0000000000000000000000000000'
00004E40  D3E3E7C2 D9406089                    713  DC CL48'LTXBR -inf TR'
00004E70  FFFF0000 00000000                    714  DC XL16'FFFF0000000000000000000000000000'
00004E80  D3E3E7C2 D9404ED8                    715  DC CL48'LTXBR +QNaN NT'
00004EB0  7FFF8000 00000000                    716  DC XL16'7FFF8000000000000000000000000000'
00004EC0  D3E3E7C2 D9404ED8                    717  DC CL48'LTXBR +QNaN'
00004EF0  7FFF8000 00000000                    718  DC XL16'7FFF8000000000000000000000000000'
00004F00  D3E3E7C2 D94060D8                    719  DC CL48'LTXBR -QNaN NT'
00004F30  FFFF8000 00000000                    720  DC XL16'FFFF8000000000000000000000000000'
00004F40  D3E3E7C2 D94060D8                    721  DC CL48'LTXBR -QNaN TR'
00004F70  FFFF8000 00000000                    722  DC XL16'FFFF8000000000000000000000000000'
00004F80  D3E3E7C2 D9404EE2                    723  DC CL48'LTXBR +SNaN NT'
00004FB0  7FFF8100 00000000                    724  DC XL16'7FFF8100000000000000000000000000'
00004FC0  D3E3E7C2 D9404EE2                    725  DC CL48'LTXBR +SNaN TR'
00004FF0  0000DEAD 00000000                    726  DC XL16'0000DEAD00000000000000000000000'
00005000  D3E3E7C2 D94060E2                    727  DC CL48'LTXBR -SNaN NT'
00005030  FFFF8100 00000000                    728  DC XL16'FFFF8100000000000000000000000000'
00005040  D3E3E7C2 D94060E2                    729  DC CL48'LTXBR -SNaN TR'
00005070  0000DEAD 00000000                    730  DC XL16'0000DEAD00000000000000000000000'
                             00000018  00000001   731 XBFPOUT_NUM EQU (*-XBFPOUT_GOOD)/64
                                                732 *
                                                733 *
                             00005080  00000001   734 XBFPOCC_GOOD EQU *
00005080  E3C3E7C2 40C3C340                    735  DC CL48'TCXB CC +0'
000050B0  00000001 00000000                    736  DC XL16'00000001000000000000000000000000'
000050C0  E3C3E7C2 40C3C340                    737  DC CL48'TCXB CC -0'
000050F0  00000000 01000000                    738  DC XL16'00000000100000000000000000000000'
00005100  E3C3E7C2 40C3C340                    739  DC CL48'TCXB CC +1'
00005130  02020000 00010000                    740  DC XL16'02020000000100000000000000000000'
00005140  E3C3E7C2 40C3C340                    741  DC CL48'TCXB CC -1'
00005170  01010000 00000100                    742  DC XL16'01010000000001000000000000000000'
00005180  E3C3E7C2 40C3C340                    743  DC CL48'TCXB CC +tiny'
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2    STMT

000051B0  02020000 00000001                          744  DC XL16'02020000000000010000000000000000'
000051C0  E3C3E7C2 40C3C340                          745  DC CL48'TCXB CC -tiny'
000051F0  01010000 00000000                          746  DC XL16'01010000000000000100000000000000'
00005200  E3C3E7C2 40C3C340                          747  DC CL48'TCXB CC +inf'
00005230  02020000 00000000                          748  DC XL16'02020000000000000001000000000000'
00005240  E3C3E7C2 40C3C340                          749  DC CL48'TCXB CC -inf'
00005270  01010000 00000000                          750  DC XL16'01010000000000000000010000000000'
00005280  E3C3E7C2 40C3C340                          751  DC CL48'TCXB CC +QNaN'
000052B0  03030000 00000000                          752  DC XL16'03030000000000000000000100000000'
000052C0  E3C3E7C2 40C3C340                          753  DC CL48'TCXB CC -QNaN'
000052F0  03030000 00000000                          754  DC XL16'03030000000000000000000001000000'
00005300  E3C3E7C2 40C3C340                          755  DC CL48'TCXB CC +SNaN'
00005330  03008000 00000000                          756  DC XL16'03008000000000000000000000010000'
00005340  E3C3E7C2 40C3C340                          757  DC CL48'TCXB CC -SNaN'
00005370  03008000 00000000                          758  DC XL16'03008000000000000000000000000100'
                            0000000C  00000001       759 XBFPOCC_NUM EQU (*-XBFPOCC_GOOD)/64
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

00005380                                          761 HELPERS  DS    0H       (R12 base of helper subroutines)

                                                  763 ********************************************************************
                                                  764 *                 REPORT UNEXPECTED PROGRAM CHECK
                                                  765 ********************************************************************

00005380                                          767 PGMCK    DS    0H
00005380  F342 C072 F08E      000053F2  0000008E  768          UNPK  PROGCODE(L'PROGCODE+1),PCINTCD(L'PCINTCD+1)
00005386  926B C076                     000053F6  769          MVI   PGMCOMMA,C','
0000538A  DC03 C072 C178      000053F2  000054F8  770          TR    PROGCODE,HEXTRTAB

00005390  F384 C07C F150      000053FC  00000150  772          UNPK  PGMPSW+(0*9)(9),PCOLDPSW+(0*4)(5)
00005396  9240 C084                     00005404  773          MVI   PGMPSW+(0*9)+8,C' '
0000539A  DC07 C07C C178      000053FC  000054F8  774          TR    PGMPSW+(0*9)(8),HEXTRTAB

000053A0  F384 C085 F154      00005405  00000154  776          UNPK  PGMPSW+(1*9)(9),PCOLDPSW+(1*4)(5)
000053A6  9240 C08D                     0000540D  777          MVI   PGMPSW+(1*9)+8,C' '
000053AA  DC07 C085 C178      00005405  000054F8  778          TR    PGMPSW+(1*9)(8),HEXTRTAB

000053B0  F384 C08E F158      0000540E  00000158  780          UNPK  PGMPSW+(2*9)(9),PCOLDPSW+(2*4)(5)
000053B6  9240 C096                     00005416  781          MVI   PGMPSW+(2*9)+8,C' '
000053BA  DC07 C08E C178      0000540E  000054F8  782          TR    PGMPSW+(2*9)(8),HEXTRTAB

000053C0  F384 C097 F15C      00005417  0000015C  784          UNPK  PGMPSW+(3*9)(9),PCOLDPSW+(3*4)(5)
000053C6  9240 C09F                     0000541F  785          MVI   PGMPSW+(3*9)+8,C' '
000053CA  DC07 C097 C178      00005417  000054F8  786          TR    PGMPSW+(3*9)(8),HEXTRTAB

000053D0  4100 0042                     00000042  788          LA    R0,L'PROGMSG     R0 <== length of message
000053D4  4110 C05E                     000053DE  789          LA    R1,PROGMSG       R1 --> the message text itself
000053D8  4520 C27A                     000055FA  790          BAL   R2,MSG           Go display this message
                                                  791
000053DC  07FD                                    792          BR    R13              Return to caller




000053DE                                          794 PROGMSG  DS   0CL66
000053DE  D7D9D6C7 D9C1D440                       795          DC    CL20'PROGRAM CHECK! CODE '
000053F2  88888888                                796 PROGCODE DC    CL4'hhhh'
000053F6  6B                                      797 PGMCOMMA DC    CL1','
000053F7  40D7E2E6 40                             798          DC    CL5' PSW '
000053FC  88888888 88888888                       799 PGMPSW   DC    CL36'hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh '
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

                                                      801 ********************************************************************************
                                                      802 *                        VERIFICATION ROUTINE
                                                      803 ********************************************************************************

00005420                                              805 VERISUB  DS    0H
                                                      806 *
                                                      807 **       Loop through the VERIFY TABLE...
                                                      808 *

00005420  4110 C32C                 000056AC          810          LA    R1,VERIFTAB      R1 --> Verify table
00005424  4120 0006                 00000006          811          LA    R2,VERIFLEN      R2 <== Number of entries
00005428  0D30                                        812          BASR  R3,0             Set top of loop

0000542A  9846 1000                 00000000          814          LM    R4,R6,0(R1)      Load verify table values
0000542E  4D70 C0C2                 00005442          815          BAS   R7,VERIFY         Verify results
00005432  4110 100C                 0000000C          816          LA    R1,12(,R1)       Next verify table entry
00005436  0623                                        817          BCTR  R2,R3            Loop through verify table

00005438  9500 C278                 000055F8          819          CLI   FAILFLAG,X'00'   Did all tests verify okay?
0000543C  078D                                        820          BER   R13              Yes, return to caller
0000543E  47F0 F238                 00000238          821          B     FAIL             No, load FAILURE disabled wait PSW


                                                      823 *
                                                      824 **       Loop through the ACTUAL / EXPECTED results...
                                                      825 *

00005442  0D80                                        827 VERIFY   BASR  R8,0             Set top of loop

00005444  D50F 4000 5030      00000000  00000030      829          CLC   0(16,R4),48(R5)  Actual results == Expected results?
0000544A  4770 C0DA                     0000545A      830          BNE   VERIFAIL         No, show failure
0000544E  4140 4010                     00000010      831 VERINEXT LA    R4,16(,R4)       Next actual result
00005452  4150 5040                     00000040      832          LA    R5,64(,R5)       Next expected result
00005456  0668                                        833          BCTR  R6,R8            Loop through results

00005458  07F7                                        835          BR    R7               Return to caller
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                                   837 ************************************************************************
                                                   838 *                      Report the failure...
                                                   839 ************************************************************************

0000545A  9005 C250                      000055D0   841 VERIFAIL STM   R0,R5,SAVER0R5   Save registers
0000545E  92FF C278                      000055F8   842          MVI   FAILFLAG,X'FF'   Remember verification failure
                                                   843 *
                                                   844 **       First, show them the description...
                                                   845 *
00005462  D22F C1E0 5000      00005560  00000000   846          MVC   FAILDESC,0(R5)   Save results/test description
00005468  4100 0044                      00000044   847          LA    R0,L'FAILMSG1    R0 <== length of message
0000546C  4110 C1CC                      0000554C   848          LA    R1,FAILMSG1      R1 --> the message text itself
00005470  4520 C27A                      000055FA   849          BAL   R2,MSG           Go display this message
                                                   850 *
                                                   851 **       Save address of actual and expected results
                                                   852 *
00005474  5040 C24C                      000055CC   853          ST    R4,AACTUAL       Save A(actual results)
00005478  4150 5030                      00000030   854          LA    R5,48(,R5)       R5 ==> expected results
0000547C  5050 C248                      000055C8   855          ST    R5,AEXPECT       Save A(expected results)
                                                   856 *
                                                   857 **       Format and show them the EXPECTED ("Want") results...
                                                   858 *
00005480  D205 C210 C378      00005590  000056F8   859          MVC   WANTGOT,=CL6'Want: '
00005486  F384 C216 C248      00005596  000055C8   860          UNPK  FAILADR(L'FAILADR+1),AEXPECT(L'AEXPECT+1)
0000548C  9240 C21E                      0000559E   861          MVI   BLANKEQ,C' '
00005490  DC07 C216 C178      00005596  000054F8   862          TR    FAILADR,HEXTRTAB

00005496  F384 C221 5000      000055A1  00000000   864          UNPK  FAILVALS+(0*9)(9),(0*4)(5,R5)
0000549C  9240 C229                      000055A9   865          MVI   FAILVALS+(0*9)+8,C' '
000054A0  DC07 C221 C178      000055A1  000054F8   866          TR    FAILVALS+(0*9)(8),HEXTRTAB

000054A6  F384 C22A 5004      000055AA  00000004   868          UNPK  FAILVALS+(1*9)(9),(1*4)(5,R5)
000054AC  9240 C232                      000055B2   869          MVI   FAILVALS+(1*9)+8,C' '
000054B0  DC07 C22A C178      000055AA  000054F8   870          TR    FAILVALS+(1*9)(8),HEXTRTAB

000054B6  F384 C233 5008      000055B3  00000008   872          UNPK  FAILVALS+(2*9)(9),(2*4)(5,R5)
000054BC  9240 C23B                      000055BB   873          MVI   FAILVALS+(2*9)+8,C' '
000054C0  DC07 C233 C178      000055B3  000054F8   874          TR    FAILVALS+(2*9)(8),HEXTRTAB

000054C6  F384 C23C 500C      000055BC  0000000C   876          UNPK  FAILVALS+(3*9)(9),(3*4)(5,R5)
000054CC  9240 C244                      000055C4   877          MVI   FAILVALS+(3*9)+8,C' '
000054D0  DC07 C23C C178      000055BC  000054F8   878          TR    FAILVALS+(3*9)(8),HEXTRTAB

000054D6  4100 0035                      00000035   880          LA    R0,L'FAILMSG2    R0 <== length of message
000054DA  4110 C210                      00005590   881          LA    R1,FAILMSG2      R1 --> the message text itself
000054DE  4520 C27A                      000055FA   882          BAL   R2,MSG           Go display this message
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                          884 *
                                          885 **       Format and show them the ACTUAL ("Got") results...
                                          886 *
000054E2  D205 C210 C37E      00005590  000056FE   887          MVC   WANTGOT,=CL6'Got:  '
000054E8  F384 C216 C24C      00005596  000055CC   888          UNPK  FAILADR(L'FAILADR+1),AACTUAL(L'AACTUAL+1)
000054EE  9240 C21E                     0000559E   889          MVI   BLANKEQ,C' '
000054F2  DC07 C216 C178      00005596  000054F8   890          TR    FAILADR,HEXTRTAB

000054F8  F384 C221 4000      000055A1  00000000   892          UNPK  FAILVALS+(0*9)(9),(0*4)(5,R4)
000054FE  9240 C229                     000055A9   893          MVI   FAILVALS+(0*9)+8,C' '
00005502  DC07 C221 C178      000055A1  000054F8   894          TR    FAILVALS+(0*9)(8),HEXTRTAB

00005508  F384 C22A 4004      000055AA  00000004   896          UNPK  FAILVALS+(1*9)(9),(1*4)(5,R4)
0000550E  9240 C232                     000055B2   897          MVI   FAILVALS+(1*9)+8,C' '
00005512  DC07 C22A C178      000055AA  000054F8   898          TR    FAILVALS+(1*9)(8),HEXTRTAB

00005518  F384 C233 4008      000055B3  00000008   900          UNPK  FAILVALS+(2*9)(9),(2*4)(5,R4)
0000551E  9240 C23B                     000055BB   901          MVI   FAILVALS+(2*9)+8,C' '
00005522  DC07 C233 C178      000055B3  000054F8   902          TR    FAILVALS+(2*9)(8),HEXTRTAB

00005528  F384 C23C 400C      000055BC  0000000C   904          UNPK  FAILVALS+(3*9)(9),(3*4)(5,R4)
0000552E  9240 C244                     000055C4   905          MVI   FAILVALS+(3*9)+8,C' '
00005532  DC07 C23C C178      000055BC  000054F8   906          TR    FAILVALS+(3*9)(8),HEXTRTAB

00005538  4100 0035                     00000035   908          LA    R0,L'FAILMSG2    R0 <== length of message
0000553C  4110 C210                     00005590   909          LA    R1,FAILMSG2      R1 --> the message text itself
00005540  4520 C27A                     000055FA   910          BAL   R2,MSG           Go display this message

00005544  9805 C250                     000055D0   912          LM    R0,R5,SAVER0R5   Restore registers
00005548  47F0 C0CE                     0000544E   913          B     VERINEXT         Continue with verification...



0000554C                                        915 FAILMSG1 DS   0CL68
0000554C  C3D6D4D7 C1D9C9E2                     916          DC    CL20'COMPARISON FAILURE! '
00005560  4D8485A2 83998997                     917 FAILDESC DC    CL48'(description)'


00005590                                        919 FAILMSG2 DS   0CL53
00005590  40404040 4040                         920 WANTGOT  DC    CL6' '           'Want: ' -or- 'Got:  '
00005596  C1C1C1C1 C1C1C1C1                     921 FAILADR  DC    CL8'AAAAAAAA'
0000559E  407E40                                922 BLANKEQ  DC    CL3' = '
000055A1  88888888 88888888                     923 FAILVALS DC    CL36'hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh '

000055C8  00000000                              925 AEXPECT  DC    F'0'             ==> Expected ("Want") results
000055CC  00000000                              926 AACTUAL  DC    F'0'             ==> Actual ("Got") results
000055D0  00000000 00000000                     927 SAVER0R5 DC    6F'0'            Registers R0 - R5 save area
000055E8  F0F1F2F3 F4F5F6F7                     928 CHARHEX  DC    CL16'0123456789ABCDEF'
                            000054F8  00000010  929 HEXTRTAB EQU   CHARHEX-X'F0'    Hexadecimal translation table
000055F8  00                                    930 FAILFLAG DC    X'00'            FF = Fail, 00 = Success
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                            932 ********************************************************************
                                            933 *        Issue HERCULES MESSAGE pointed to by R1, length in R0
                                            934 ********************************************************************

000055FA  4900 C374                 000056F4   936 MSG      CH    R0,=H'0'               Do we even HAVE a message?
000055FE  07D2                                 937          BNHR  R2                     No, ignore

00005600  9002 C2B0                 00005630   939          STM   R0,R2,MSGSAVE          Save registers

00005604  4900 C376                 000056F6   941          CH    R0,=AL2(L'MSGMSG)      Message length within limits?
00005608  47D0 C290                 00005610   942          BNH   MSGOK                  Yes, continue
0000560C  4100 005F                 0000005F   943          LA    R0,L'MSGMSG            No, set to maximum

00005610  1820                                 945 MSGOK    LR    R2,R0                  Copy length to work register
00005612  0620                                 946          BCTR  R2,0                   Minus-1 for execute
00005614  4420 C2BC                 0000563C   947          EX    R2,MSGMVC              Copy message to O/P buffer

00005618  4120 200A                 0000000A   949          LA    R2,1+L'MSGCMD(,R2)     Calculate true command length
0000561C  4110 C2C2                 00005642   950          LA    R1,MSGCMD              Point to true command

00005620  83120008                             952          DC    X'83',X'12',X'0008'    Issue Hercules Diagnose X'008'
00005624  4780 C2AA                 0000562A   953          BZ    MSGRET                 Return if successful
00005628  0000                                 954          DC    H'0'                   CRASH for debugging purposes

0000562A  9802 C2B0                 00005630   956 MSGRET   LM    R0,R2,MSGSAVE          Restore registers
0000562E  07F2                                 957          BR    R2                     Return to caller
```

```
00005630  00000000 00000000                    959 MSGSAVE  DC    3F'0'                  Registers save area
0000563C  D200 C2CB 1000      0000564B  00000000   960 MSGMVC   MVC   MSGMSG(0),0(R1)      Executed instruction

00005642  D4E2C7D5 D6C8405C                    962 MSGCMD   DC    C'MSGNOH * '           *** HERCULES MESSAGE COMMAND ***
0000564B  40404040 40404040                    963 MSGMSG   DC    CL95' '                The message text to be displayed
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                    965 ********************************************************************************
                                                    966 *                              VERIFY TABLE
                                                    967 ********************************************************************************
                                                    968 *
                                                    969 *        A(actual results), A(expected results), A(#of results)
                                                    970 *
                                                    971 ********************************************************************************

000056AC                                            973 VERIFTAB DC    0F'0'
000056AC  00001000                                  974          DC    A(SBFPOUT)
000056B0  00004000                                  975          DC    A(SBFPOUT_GOOD)
000056B4  00000006                                  976          DC    A(SBFPOUT_NUM)
                                                    977 *
000056B8  00001100                                  978          DC    A(SBFPOCC)
000056BC  00004180                                  979          DC    A(SBFPOCC_GOOD)
000056C0  0000000C                                  980          DC    A(SBFPOCC_NUM)
                                                    981 *
000056C4  00002000                                  982          DC    A(LBFPOUT)
000056C8  00004480                                  983          DC    A(LBFPOUT_GOOD)
000056CC  0000000C                                  984          DC    A(LBFPOUT_NUM)
                                                    985 *
000056D0  00002100                                  986          DC    A(LBFPOCC)
000056D4  00004780                                  987          DC    A(LBFPOCC_GOOD)
000056D8  0000000C                                  988          DC    A(LBFPOCC_NUM)
                                                    989 *
000056DC  00003000                                  990          DC    A(XBFPOUT)
000056E0  00004A80                                  991          DC    A(XBFPOUT_GOOD)
000056E4  00000018                                  992          DC    A(XBFPOUT_NUM)
                                                    993 *
000056E8  00003200                                  994          DC    A(XBFPOCC)
000056EC  00005080                                  995          DC    A(XBFPOCC_GOOD)
000056F0  0000000C                                  996          DC    A(XBFPOCC_NUM)
                                                    997 *
                            00000006  00000001      998 VERIFLEN EQU   (*-VERIFTAB)/12    #of entries in verify table
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

000056F4                                            1000          END
000056F4  0000                                      1001                =H'0'
000056F6  005F                                      1002                =AL2(L'MSGMSG)
000056F8  E68195A3 7A40                             1003                =CL6'Want: '
000056FE  C796A37A 4040                             1004                =CL6'Got:  '
```

```
     SYMBOL         TYPE  VALUE   LENGTH  DEFN  REFERENCES

AACTUAL              F    0055CC       4   926   853   888
AEXPECT              F    0055C8       4   925   855   860
AHELPERS             A    00027C       4   202   192   228
BFPLTTDC             J    000000   22276   118
BLANKEQ              C    00559E       3   922   861   889
CHARHEX              C    0055E8      16   928   929
CTLR0                F    0002D8       4   238   211   212   213
EXTDS                F    000320       4   261   221
FAIL                 I    000238       4   200   821
FAILADR              C    005596       8   921   860   862   888   890
FAILDESC             C    005560      48   917   846
FAILFLAG             X    0055F8       1   930   819   842
FAILMSG1             C    00554C      68   915   847   848
FAILMSG2             C    005590      53   919   880   881   908   909
FAILPSW              X    0002C8       8   236   200
FAILVALS             C    0055A1      36   923   864   865   866   868   869   870   872   873   874   876   877   878   892   893
                                               894   896   897   898   900   901   902   904   905   906
FPCREGNT             X    0002DC       4   239   287   353   420
FPCREGTR             X    0002E0       4   240   299   365   433
FPR0                 U    000000       1   139
FPR1                 U    000001       1   140   288   293   294   300   305   306   354   359   360   366   371   372   421   426
                                               427   434   439   440
FPR10                U    00000A       1   149   415
FPR11                U    00000B       1   150
FPR12                U    00000C       1   151
FPR13                U    00000D       1   152
FPR14                U    00000E       1   153
FPR15                U    00000F       1   154
FPR2                 U    000002       1   141
FPR3                 U    000003       1   142   422   428   435   441
FPR4                 U    000004       1   143
FPR5                 U    000005       1   144
FPR6                 U    000006       1   145
FPR7                 U    000007       1   146
FPR8                 U    000008       1   147   282   293   305   318   348   359   371   384   414   426   439   453
FPR9                 U    000009       1   148
GOODPSW              X    0002B8       8   235   232
HELPERS              H    005380       2   761   157   202
HEXTRTAB             U    0054F8      16   929   770   774   778   782   786   862   866   870   874   878   890   894   898   902
                                               906
IMAGE                1    000000   22276     0
LBFPIN               D    000590       8   505   518   257
LBFPINCT             U    00000C       1   518   256
LBFPINVL             X    0005F0       8   520   354   366
LBFPOCC              U    002100       1   564   259   986
LBFPOCC_GOOD         U    004780       1   654   679   987
LBFPOCC_NUM          U    00000C       1   679   988
LBFPOUT              U    002000       1   562   258   982
LBFPOUT_GOOD         U    004480       1   626   651   983
LBFPOUT_NUM          U    00000C       1   651   984
LONGS                F    000310       4   255   218
MSG                  I    0055FA       4   936   790   849   882   910
MSGCMD               C    005642       9   962   949   950
MSGMSG               C    00564B      95   963   943   960   941
MSGMVC               I    00563C       6   960   947
MSGOK                I    005610       2   945   942
```

SYMBOL	TYPE	VALUE	LENGTH	DEFN	REFERENCES
MSGRET	I	00562A	4	956	953
MSGSAVE	F	005630	4	959	939 956
PCINTCD	H	00008E	2	170	187 768
PCNOTDTA	I	00020C	4	191	188
PCOLDPSW	U	000150	1	172	189 772 776 780 784
PGMCK	H	005380	2	767	193
PGMCOMMA	C	0053F6	1	797	769
PGMPSW	C	0053FC	36	799	772 773 774 776 777 778 780 781 782 784 785 786
PROGCHK	H	000200	2	186	178
PROGCODE	C	0053F2	4	796	768 770
PROGMSG	C	0053DE	66	794	788 789
PROGPSW	D	000228	8	199	198
R0	U	000000	1	120	191 194 211 213 289 290 291 292 295 296 297 301 302 303 304 307 308 309 310 311 319 320 321 355 356 357 358 361 362 363 367 368 369 370 373 374 375 376 377 385 386 387 423 424 425 429 430 431 436 437 438 442 443 444 445 446 454 455 456 788 841 847 880 908 912 936 939 941 943 945 956
R1	U	000001	1	121	313 317 318 323 379 383 384 389 448 452 453 458 789 810 814 816 848 881 909 950 960
R10	U	00000A	1	130	215 218 221 276 277 342 343 408 409
R11	U	00000B	1	131	
R12	U	00000C	1	132	157 192 228 280 329 346 395 412 464
R13	U	00000D	1	133	193 216 219 222 229 279 331 345 397 411 466 792 820
R14	U	00000E	1	134	196 197 230 231
R15	U	00000F	1	135	156 191 194
R2	U	000002	1	122	276 278 329 342 344 395 408 410 464 790 811 817 849 882 910 937 939 945 946 947 949 956 957
R3	U	000003	1	123	276 282 326 342 348 392 408 414 415 461 812 817
R4	U	000004	1	124	814 829 831 853 892 896 900 904
R5	U	000005	1	125	829 832 841 846 854 855 864 868 872 876 912
R6	U	000006	1	126	315 324 381 390 450 459 814 833
R7	U	000007	1	127	277 294 306 327 343 360 372 393 409 427 428 440 441 462 815 835
R8	U	000008	1	128	277 285 297 309 311 314 328 343 351 363 375 377 380 394 409 418 431 444 446 449 463 827 833
R9	U	000009	1	129	314 321 322 380 387 388 449 456 457
SAVER0R5	F	0055D0	4	927	841 912
SAVEREGS	F	00023C	4	201	191 194
SBFPIN	F	00055C	4	481	494 251
SBFPINCT	U	00000C	1	494	250
SBFPINVL	X	00058C	4	496	288 300
SBFPOCC	U	001100	1	557	253 978
SBFPOCC_GOOD	U	004180	1	598	623 979
SBFPOCC_NUM	U	00000C	1	623	980
SBFPOUT	U	001000	1	555	252 974
SBFPOUT_GOOD	U	004000	1	582	595 975
SBFPOUT_NUM	U	000006	1	595	976
SHORTS	F	000300	4	249	215
START	H	000280	2	210	175
STRTLABL	U	000000	1	119	169 172 174 177 185 248 555 557 562 564 569 571 580
TESTLBFP	H	0003DC	2	341	219
TESTSBFP	H	000330	2	275	216
TESTXBFP	H	000488	2	407	222
VERIFAIL	I	00545A	4	841	830
VERIFLEN	U	000006	1	998	811

```
     SYMBOL         TYPE  VALUE   LENGTH  DEFN  REFERENCES

VERIFTAB             F    0056AC       4   973   998   810
VERIFY               I    005442       2   827   815
VERINEXT             I    00544E       4   831   913
VERISUB              H    005420       2   805   229
WANTGOT              C    005590       6   920   859   887
XBFPIN               D    0005F8       8   530   543   263
XBFPINCT             U    00000C       1   543   262
XBFPINVL             X    0006B8      16   545   421   422   434   435
XBFPOCC              U    003200       1   571   265   994
XBFPOCC_GOOD         U    005080       1   734   759   995
XBFPOCC_NUM          U    00000C       1   759   996
XBFPOUT              U    003000       1   569   264   990
XBFPOUT_GOOD         U    004A80       1   682   731   991
XBFPOUT_NUM          U    000018       1   731   992
=AL2(L'MSGMSG)       R    0056F6       2  1002   941
=CL6'Got:  '         C    0056FE       6  1004   887
=CL6'Want: '         C    0056F8       6  1003   859
=H'0'                H    0056F4       2  1001   936
=X'10000000'         X    000558       4   472   357   369
=X'20000000'         X    000554       4   471   291   303
=X'CFFFFFFF'         X    000550       4   470   290   302   356   368   424   437
=X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
                     X    000540      16   469   285   351   418
```

MACRO DEFN REFERENCES

No defined macros

```
   DESC      SYMBOL     SIZE      POS         ADDR

Entry: 0

Image       IMAGE      22276  0000-5703  0000-5703
  Region               22276  0000-5703  0000-5703
    CSECT   BFPLTTDC   22276  0000-5703  0000-5703
```

STMT FILE NAME

1 c:\Users\Fish\Documents\Visual Studio 2008\Projects\MyProjects\ASMA-0\bfp-012-loadtest\bfp-012-loadtest.asm

** NO ERRORS FOUND **