```
 LOC        OBJECT CODE       ADDR1     ADDR2     STMT
                                                     2 ************************************************************************
                                                     3 *
                                                     4 *Testcase IEEE CONVERT FROM LOGICAL 32
                                                     5 *  Test case capability includes ieee exceptions trappable and otherwise.
                                                     6 *  Test result, FPCR flags, and DXC saved for all tests.  (Convert From
                                                     7 *  Fixed does not set the condition code.)
                                                     8 *
                                                     9 *
                                                    10 *                        ********************
                                                    11 *                        **   IMPORTANT!   **
                                                    12 *                        ********************
                                                    13 *
                                                    14 *        This test uses the Hercules Diagnose X'008' interface
                                                    15 *        to display messages and thus your .tst runtest script
                                                    16 *        MUST contain a "DIAG8CMD ENABLE" statement within it!
                                                    17 *
                                                    18 *
                                                    19 ************************************************************************


                                                    21 ************************************************************************
                                                    22 *
                                                    23 *                        bfp-008-cvtfrlog.asm
                                                    24 *
                                                    25 *        This assembly-language source file is part of the
                                                    26 *        Hercules Binary Floating Point Validation Package
                                                    27 *                        by Stephen R. Orso
                                                    28 *
                                                    29 * Copyright 2016 by Stephen R Orso.
                                                    30 * Runtest *Compare dependency removed by Fish on 2022-08-16
                                                    31 * PADCSECT macro/usage removed by Fish on 2022-08-16
                                                    32 *
                                                    33 * Redistribution and use in source and binary forms, with or without
                                                    34 * modification, are permitted provided that the following conditions
                                                    35 * are met:
                                                    36 *
                                                    37 * 1. Redistributions of source code must retain the above copyright
                                                    38 *    notice, this list of conditions and the following disclaimer.
                                                    39 *
                                                    40 * 2. Redistributions in binary form must reproduce the above copyright
                                                    41 *    notice, this list of conditions and the following disclaimer in
                                                    42 *    the documentation and/or other materials provided  with the
                                                    43 *    distribution.
                                                    44 *
                                                    45 * 3. The name of the author may not be used to endorse or promote
                                                    46 *    products derived from this software without specific prior written
                                                    47 *    permission.
                                                    48 *
                                                    49 * DISCLAMER: THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDER "AS IS"
                                                    50 * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
                                                    51 * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
                                                    52 * PARTICULAR PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT
                                                    53 * HOLDER BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
                                                    54 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
                                                    55 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
                                                    56 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
```

```
 LOC         OBJECT CODE       ADDR1     ADDR2     STMT

                                                      57 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
                                                      58 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
                                                      59 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
                                                      60 *
                                                      61 ********************************************************************************



                                                      63 ********************************************************************************
                                                      64 *
                                                      65 * Tests the following three conversion instructions
                                                      66 *   CONVERT FROM LOGICAL (32 to short BFP, RRF-e)
                                                      67 *   CONVERT FROM LOGICAL (32 to long BFP, RRF-e)
                                                      68 *   CONVERT FROM LOGICAL (32 to extended BFP, RRF-e)
                                                      69 *
                                                      70 * Test data is compiled into this program.  The test script that runs
                                                      71 * this program can provide alternative test data through Hercules R
                                                      72 * commands.
                                                      73 *
                                                      74 * Test Case Order
                                                      75 * 1) Uint-32 to Short BFP
                                                      76 * 2) Uint-32 to Short BFP with all rounding modes
                                                      77 * 3) Uint-32 to Long BFP
                                                      78 * 4) Uint-32 to Extended BFP
                                                      79 *
                                                      80 * Conversion of uint-32 to long or extended is always exact because the
                                                      81 * number of bits in uint-32 is less than the number of bits in a long
                                                      82 * or extended significand.  For this reason, exhaustive rounding
                                                      83 * testing is not performed for long or extended.
                                                      84 *
                                                      85 * Provided test data is
                                                      86 *       1, 2, 4, 9,
                                                      87 *       4 294 967 294 (0xFFFFFFFE) (note 1)
                                                      88 *       4 294 967 040 (0xFFFFFF00) (note 2)
                                                      89 *       4 294 967 168 (0xFFFFFF40) (note 3)
                                                      90 *   The last three values will trigger inexact exceptions when
                                                      91 *   converted to to short BFP and are used for exhaustive rounding mode
                                                      92 *   testing for short BFP.  Specifics for each:
                                                      93 *   1) Fits in short BFP but always results in loss of precision.
                                                      94 *      Always reports incremented on trappable inexact.
                                                      95 *   2) Fits in short BFP with no loss of precision.
                                                      96 *   3) Fits in short BFP, always reports inexact, and sometimes
                                                      97 *      incremented depending on rounding mode.
                                                      98 *
                                                      99 * Also tests the following floating point support instructions
                                                     100 *   LOAD  (Short)
                                                     101 *   LOAD  (Long)
                                                     102 *   LOAD FPC
                                                     103 *   SET BFP ROUNDING MODE 2-BIT
                                                     104 *   SET BFP ROUNDING MODE 3-BIT
                                                     105 *   STORE (Short)
                                                     106 *   STORE (Long)
                                                     107 *   STORE FPC
                                                     108 *
                                                     109 ********************************************************************************
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                         111 *
                                         112 *  Note: for compatibility with the z/CMS test rig, do not change
                                         113 *  or use R11, R14, or R15.  Everything else is fair game.
                                         114 *


                          00000000  00004F9B   116 BFPCVTFL START 0
                          00000000  00000001   117 STRTLABL EQU   *
                          00000000  00000001   118 R0       EQU   0                 Work register for cc extraction
                          00000001  00000001   119 R1       EQU   1
                          00000002  00000001   120 R2       EQU   2                 Holds count of test input values
                          00000003  00000001   121 R3       EQU   3                 Points to next test input value(s)
                          00000004  00000001   122 R4       EQU   4                 Available
                          00000005  00000001   123 R5       EQU   5                 Available
                          00000006  00000001   124 R6       EQU   6                 Available
                          00000007  00000001   125 R7       EQU   7                 Pointer to next result value(s)
                          00000008  00000001   126 R8       EQU   8                 Pointer to next FPCR result
                          00000009  00000001   127 R9       EQU   9                 Rounding tests top of outer loop
                          0000000A  00000001   128 R10      EQU   10                Pointer to test address list
                          0000000B  00000001   129 R11      EQU   11                **Reserved for z/CMS test rig
                          0000000C  00000001   130 R12      EQU   12                Holds number of test cases in set
                          0000000D  00000001   131 R13      EQU   13                Mainline return address
                          0000000E  00000001   132 R14      EQU   14                **Return address for z/CMS test rig
                          0000000F  00000001   133 R15      EQU   15                **Base register on z/CMS or Hyperion
                                         134 *
                                         135 * Floating Point Register equates to keep the cross reference clean
                                         136 *
                          00000000  00000001   137 FPR0     EQU   0
                          00000001  00000001   138 FPR1     EQU   1
                          00000002  00000001   139 FPR2     EQU   2
                          00000003  00000001   140 FPR3     EQU   3
                          00000004  00000001   141 FPR4     EQU   4
                          00000005  00000001   142 FPR5     EQU   5
                          00000006  00000001   143 FPR6     EQU   6
                          00000007  00000001   144 FPR7     EQU   7
                          00000008  00000001   145 FPR8     EQU   8
                          00000009  00000001   146 FPR9     EQU   9
                          0000000A  00000001   147 FPR10    EQU   10
                          0000000B  00000001   148 FPR11    EQU   11
                          0000000C  00000001   149 FPR12    EQU   12
                          0000000D  00000001   150 FPR13    EQU   13
                          0000000E  00000001   151 FPR14    EQU   14
                          0000000F  00000001   152 FPR15    EQU   15
                                         153 *
00000000                  00000000             154          USING *,R15
00000000                  00004C00             155          USING HELPERS,R12
                                         156 *
                                         157 * Above works on real iron (R15=0 after sysclear)
                                         158 * and in z/CMS (R15 points to start of load module)
                                         159 *


                                         161 ************************************************************************
                                         162 *
                                         163 * Low core definitions, Restart PSW, and Program Check Routine.
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                                   164 *
                                                   165 ********************************************************************************


00000000                      00000000  0000008E   167          ORG   STRTLABL+X'8E'      Program check interrution code
0000008E  0000                                     168 PCINTCD  DS    H
                                                   169 *
                              00000150  00000001   170 PCOLDPSW EQU   STRTLABL+X'150'     z/Arch Program check old PSW
                                                   171 *
00000090                      00000090  000001A0   172          ORG   STRTLABL+X'1A0'     z/Arch Restart PSW
000001A0  00000001 80000000                        173          DC    X'0000000180000000',AD(START)
                                                   174 *
000001B0                      000001B0  000001D0   175          ORG   STRTLABL+X'1D0'     z/Arch Program check NEW PSW
000001D0  00000000 00000000                        176          DC    X'0000000000000000',AD(PROGCHK)
                                                   177 *
                                                   178 * Program check routine.  If Data Exception, continue execution at
                                                   179 * the instruction following the program check.  Otherwise, hard wait.
                                                   180 * No need to collect data.  All interesting DXC stuff is captured
                                                   181 * in the FPCR.
                                                   182 *
000001E0                      000001E0  00000200   183          ORG   STRTLABL+X'200'
00000200                                           184 PROGCHK  DS    0H              Program check occured...
00000200  9507 F08F                    0000008F   185          CLI   PCINTCD+1,X'07'  Data Exception?
00000204  A774 0004                    0000020C   186          JNE   PCNOTDTA       ..no, hardwait (not sure if R15 is ok)
00000208  B2B2 F150                    00000150   187          LPSWE PCOLDPSW       ..yes, resume program execution

0000020C  900F F23C                    0000023C   189 PCNOTDTA STM   R0,R15,SAVEREGS  Save registers
00000210  58C0 F27C                    0000027C   190          L     R12,AHELPERS     Get address of helper subroutines
00000214  4DD0 C000                    00004C00   191          BAS   R13,PGMCK        Report this unexpected program check
00000218  980F F23C                    0000023C   192          LM    R0,R15,SAVEREGS  Restore registers

0000021C  12EE                                     194          LTR   R14,R14        Return address provided?
0000021E  077E                                     195          BNZR  R14            Yes, return to z/CMS test rig.
00000220  B2B2 F228                    00000228   196          LPSWE PROGPSW        Not data exception, enter disabled wait
00000228  00020000 00000000                        197 PROGPSW  DC    0D'0',X'0002000000000000',XL6'00',X'DEAD' Abnormal end
00000238  B2B2 F2D0                    000002D0   198 FAIL     LPSWE FAILPSW        Not data exception, enter disabled wait
0000023C  00000000 00000000                        199 SAVEREGS DC    16F'0'          Registers save area
0000027C  00004C00                                 200 AHELPERS DC    A(HELPERS)      Address of helper subroutines
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                                    202 ********************************************************************************
                                                    203 *
                                                    204 *  Main program.  Enable Advanced Floating Point, process test cases.
                                                    205 *
                                                    206 ********************************************************************************


00000280  B600 F2E0                        000002E0   208 START    STCTL R0,R0,CTLR0    Store CR0 to enable AFP
00000284  9604 F2E1                        000002E1   209          OI    CTLR0+1,X'04'  Turn on AFP bit
00000288  B700 F2E0                        000002E0   210          LCTL  R0,R0,CTLR0    Reload updated CR0
                                                    211 *
0000028C  41A0 F2EC                        000002EC   212          LA    R10,SHORTS     Point to integer test inputs
00000290  4DD0 F32C                        0000032C   213          BAS   R13,CELFBR     Convert values from uint-32 to short BFP
00000294  41A0 F31C                        0000031C   214          LA    R10,RMSHORTS   Point to inputs for rounding mode tests
00000298  4DD0 F36E                        0000036E   215          BAS   R13,CELFBRA    Convert using all rounding mode options
                                                    216 *
0000029C  41A0 F2FC                        000002FC   217          LA    R10,LONGS      Point to integer test inputs
000002A0  4DD0 F440                        00000440   218          BAS   R13,CDLFBR     Convert values from uint-32 to long BFP
                                                    219 *
000002A4  41A0 F30C                        0000030C   220          LA    R10,EXTDS      Point to integer test inputs
000002A8  4DD0 F482                        00000482   221          BAS   R13,CXLFBR     Convert values from uint-32 to ext'd BFP
                                                    222 *
                                                    223 ********************************************************************************
                                                    224 *                    Verify test results...
                                                    225 ********************************************************************************
                                                    226 *
000002AC  58C0 F27C                        0000027C   227          L     R12,AHELPERS   Get address of helper subroutines
000002B0  4DD0 C0A0                        00004CA0   228          BAS   R13,VERISUB    Go verify results
000002B4  12EE                                        229          LTR   R14,R14        Was return address provided?
000002B6  077E                                        230          BNZR  R14            Yes, return to z/CMS test rig.
000002B8  B2B2 F2C0                        000002C0   231          LPSWE GOODPSW        Load SUCCESS PSW
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

000002C0                                              233          DS    0D             Ensure correct alignment for PSW
000002C0  00020000 00000000                           234 GOODPSW  DC    X'0002000000000000',AD(0)  Normal end - disabled wait
000002D0  00020000 00000000                           235 FAILPSW  DC    X'0002000000000000',XL6'00',X'0BAD' Abnormal end
                                                      236 *
000002E0  00000000                                    237 CTLR0    DS    F
000002E4  00000000                                    238 FPCREGNT DC    X'00000000'  FPCR, trap all IEEE exceptions, zero flags
000002E8  F8000000                                    239 FPCREGTR DC    X'F8000000'  FPCR, trap no IEEE exceptions, zero flags
                                                      240 *
                                                      241 * Input values parameter list, four fullwords:
                                                      242 *      1) Count,
                                                      243 *      2) Address of inputs,
                                                      244 *      3) Address to place results, and
                                                      245 *      4) Address to place DXC/Flags/cc values.
                                                      246 *
000002EC                                              247 SHORTS   DS    0F
000002EC  00000006                                    248          DC    A(INTCOUNT/4)
000002F0  000004CC                                    249          DC    A(INTIN)
000002F4  00001000                                    250          DC    A(SBFPOUT)
000002F8  00001100                                    251          DC    A(SBFPFLGS)
                                                      252 *
000002FC                                              253 LONGS    DS    0F           int-32 inputs for long BFP testing
000002FC  00000006                                    254          DC    A(INTCOUNT/4)
00000300  000004CC                                    255          DC    A(INTIN)
00000304  00002000                                    256          DC    A(LBFPOUT)
00000308  00002100                                    257          DC    A(LBFPFLGS)
                                                      258 *
0000030C                                              259 EXTDS    DS    0F           int-32 inputs for Extended BFP testing
0000030C  00000006                                    260          DC    A(INTCOUNT/4)
00000310  000004CC                                    261          DC    A(INTIN)
00000314  00003000                                    262          DC    A(XBFPOUT)
00000318  00003200                                    263          DC    A(XBFPFLGS)
                                                      264 *
0000031C  00000003                                    265 RMSHORTS DC    A(INTRMCT/4)
00000320  000004D8                                    266          DC    A(INTINRM)   Last two int-32 are only concerns
00000324  00001200                                    267          DC    A(SBFPRMO)   Space for rounding mode tests
00000328  00001500                                    268          DC    A(SBFPRMOF)  Space for rounding mode test flags
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                    270 ********************************************************************************
                                                    271 *
                                                    272 * Convert uint-32 to short BFP format.  A pair of results is generated
                                                    273 * for each input: one with all exceptions non-trappable, and the second
                                                    274 * with all exceptions trappable.   The FPCR is stored for each result.
                                                    275 *
                                                    276 ********************************************************************************


0000032C  9823 A000                 00000000   278 CELFBR   LM    R2,R3,0(R10)  Get count and address of test input values
00000330  9878 A008                 00000008   279          LM    R7,R8,8(R10)  Get address of result area and flag area.
00000334  1222                                 280          LTR   R2,R2         Any test cases?
00000336  078D                                 281          BZR   R13           ..No, return to caller
00000338  0DC0                                 282          BASR  R12,0         Set top of loop
                                               283 *
0000033A  5810 3000                 00000000   284          L     R1,0(,R3)     Get integer test value
0000033E  B29D F2E4                 000002E4   285          LFPC  FPCREGNT      Set exceptions non-trappable
00000342  B390 0081                            286          CELFBR FPR8,0,R1,0  Cvt uint-32 in GPR1 to float in FPR8
00000346  7080 7000                 00000000   287          STE   FPR8,0(,R7)   Store short BFP result
0000034A  B29C 8000                 00000000   288          STFPC 0(R8)         Store resulting FPCR flags and DXC
                                               289 *
0000034E  B29D F2E8                 000002E8   290          LFPC  FPCREGTR      Set exceptions trappable
00000352  B390 0081                            291          CELFBR FPR8,0,R1,0  Cvt uint-32 in GPR1 to float in FPR0
00000356  7080 7004                 00000004   292          STE   FPR8,4(,R7)   Store short BFP result
0000035A  B29C 8004                 00000004   293          STFPC 4(R8)         Store resulting FPCR flags and DXC
                                               294 *
0000035E  4130 3004                 00000004   295          LA    R3,4(,R3)     Foint to next input value
00000362  4170 7008                 00000008   296          LA    R7,8(,R7)     Point to next short BFP result pair
00000366  4180 8008                 00000008   297          LA    R8,8(,R8)     Point to next FPCR/CC result area
0000036A  062C                                 298          BCTR  R2,R12        Convert next input value.
0000036C  07FD                                 299          BR    R13           All converted; return.
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                    301 ********************************************************************************
                                                    302 *
                                                    303 * Convert uint-32 to short BFP format using every rounding mode.
                                                    304 * Ten test results are generated for each input.  A 48-byte test result
                                                    305 * section is used to keep results sets aligned on a quad-double word.
                                                    306 *
                                                    307 * The first four tests use rounding modes specified in the FPCR with
                                                    308 * the IEEE Inexact exception supressed.  SRNM (2-bit) is used  for the
                                                    309 * first two FPCR-controlled tests and SRNMB (3-bit) is used for the
                                                    310 * last two To get full coverage of that instruction pair.
                                                    311 *
                                                    312 * The next six results use instruction-specified rounding modes.
                                                    313 *
                                                    314 * The default rounding mode (0 for RNTE) is not tested in this section;
                                                    315 * prior tests used the default rounding mode.  RNTE is tested
                                                    316 * explicitly as a rounding mode in this section.
                                                    317 *
                                                    318 ********************************************************************************


0000036E  9823 A000                 00000000    320 CELFBRA  LM    R2,R3,0(R10)  Get count and address of test input values
00000372  9878 A008                 00000008    321          LM    R7,R8,8(R10)  Get address of result area and flag area.
00000376  1222                                  322          LTR   R2,R2         Any test cases?
00000378  078D                                  323          BZR   R13           ..No, return to caller
0000037A  0DC0                                  324          BASR  R12,0         Set top of loop
                                                    325 *
0000037C  5810 3000                 00000000    326          L     R1,0(,R3)    Get integer test value
                                                    327 *
                                                    328 * Test cases using rounding mode specified in the FPCR
                                                    329 *
00000380  B29D F2E4                 000002E4    330          LFPC  FPCREGNT      Set exceptions non-trappable, clear flags
00000384  B299 0001                 00000001    331          SRNM  1             SET FPCR to RZ, towards zero.
00000388  B390 0481                             332          CELFBR FPR8,0,R1,B'0100'  FPCR ctl'd rounding, inexact masked
0000038C  7080 7000                 00000000    333          STE   FPR8,0*4(,R7) Store short BFP result
00000390  B29C 8000                 00000000    334          STFPC 0(R8)         Store resulting FPCR flags and DXC
                                                    335 *
00000394  B29D F2E4                 000002E4    336          LFPC  FPCREGNT      Set exceptions non-trappable, clear flags
00000398  B299 0002                 00000002    337          SRNM  2             SET FPCR to RP, to +infinity
0000039C  B390 0481                             338          CELFBR FPR8,0,R1,B'0100'  FPCR ctl'd rounding, inexact masked
000003A0  7080 7004                 00000004    339          STE   FPR8,1*4(,R7) Store short BFP result
000003A4  B29C 8004                 00000004    340          STFPC 1*4(R8)       Store resulting FPCR flags and DXC
                                                    341 *
000003A8  B29D F2E4                 000002E4    342          LFPC  FPCREGNT      Set exceptions non-trappable, clear flags
000003AC  B2B8 0003                 00000003    343          SRNMB 3             SET FPCR to RM, to -infinity
000003B0  B390 0481                             344          CELFBR FPR8,0,R1,B'0100'  FPCR ctl'd rounding, inexact masked
000003B4  7080 7008                 00000008    345          STE   FPR8,2*4(,R7) Store short BFP result
000003B8  B29C 8008                 00000008    346          STFPC 2*4(R8)       Store resulting FPCR flags and DXC
                                                    347 *
000003BC  B29D F2E4                 000002E4    348          LFPC  FPCREGNT      Set exceptions non-trappable, clear flags
000003C0  B2B8 0007                 00000007    349          SRNMB 7             RFS, Prepare for Shorter Precision
000003C4  B390 0481                             350          CELFBR FPR8,0,R1,B'0100'  FPCR ctl'd rounding, inexact masked
000003C8  7080 700C                 0000000C    351          STE   FPR8,3*4(,R7) Store short BFP result
000003CC  B29C 800C                 0000000C    352          STFPC 3*4(R8)       Store resulting FPCR flags and DXC
                                                    353 *
                                                    354 * Test cases using rounding mode specified in the instruction M3 field
                                                    355 *
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

000003D0  B29D F2E4                    000002E4   356          LFPC  FPCREGNT      Set exceptions non-trappable, clear flags
000003D4  B390 1081                               357          CELFBR FPR8,1,R1,B'0000'  RNTA, to nearest, ties away
000003D8  7080 7010                    00000010   358          STE   FPR8,4*4(,R7) Store short BFP result
000003DC  B29C 8010                    00000010   359          STFPC 4*4(R8)       Store resulting FPCR flags and DXC
                                                  360 *
000003E0  B29D F2E4                    000002E4   361          LFPC  FPCREGNT      Set exceptions non-trappable, clear flags
000003E4  B390 3081                               362          CELFBR FPR8,3,R1,B'0000'  RFS, prepare for shorter precision
000003E8  7080 7014                    00000014   363          STE   FPR8,5*4(,R7) Store short BFP result
000003EC  B29C 8014                    00000014   364          STFPC 5*4(R8)       Store resulting FPCR flags and DXC
                                                  365 *
000003F0  B29D F2E4                    000002E4   366          LFPC  FPCREGNT      Set exceptions non-trappable, clear flags
000003F4  B390 4081                               367          CELFBR FPR8,4,R1,B'0000'  RNTE, to nearest, ties to even
000003F8  7080 7018                    00000018   368          STE   FPR8,6*4(,R7) Store short BFP result
000003FC  B29C 8018                    00000018   369          STFPC 6*4(R8)       Store resulting FPCR flags and DXC
                                                  370 *
00000400  B29D F2E4                    000002E4   371          LFPC  FPCREGNT      Set exceptions non-trappable, clear flags
00000404  B390 5081                               372          CELFBR FPR8,5,R1,B'0000'  RZ, toward zero
00000408  7080 701C                    0000001C   373          STE   FPR8,7*4(,R7) Store short BFP result
0000040C  B29C 801C                    0000001C   374          STFPC 7*4(R8)       Store resulting FPCR flags and DXC
                                                  375 *
00000410  B29D F2E4                    000002E4   376          LFPC  FPCREGNT      Set exceptions non-trappable, clear flags
00000414  B390 6081                               377          CELFBR FPR8,6,R1,B'0000'  RP, to +inf
00000418  7080 7020                    00000020   378          STE   FPR8,8*4(,R7) Store short BFP result
0000041C  B29C 8020                    00000020   379          STFPC 8*4(R8)       Store resulting FPCR flags and DXC
                                                  380 *
00000420  B29D F2E4                    000002E4   381          LFPC  FPCREGNT      Set exceptions non-trappable, clear flags
00000424  B390 7081                               382          CELFBR FPR8,7,R1,B'0000'  RM, to -inf
00000428  7080 7024                    00000024   383          STE   FPR8,9*4(,R7) Store short BFP result
0000042C  B29C 8024                    00000024   384          STFPC 9*4(R8)       Store resulting FPCR flags and DXC
                                                  385 *
00000430  4130 3004                    00000004   386          LA    R3,4(,R3)     Point to next input values
00000434  4170 7030                    00000030   387          LA    R7,12*4(,R7)  Point to next short BFP converted values
00000438  4180 8030                    00000030   388          LA    R8,12*4(,R8)  Point to next FPCR/CC result area
0000043C  062C                                    389          BCTR  R2,R12        Convert next input value.
0000043E  07FD                                    390          BR    R13           All converted; return.
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                    392 ************************************************************************
                                                    393 *
                                                    394 * Convert integers to long BFP format.  A pair of results is generated
                                                    395 * for each input: one with all exceptions non-trappable, and the second
                                                    396 * with all exceptions trappable.   The FPCR is stored for each result.
                                                    397 * Conversion of a 32-bit integer to long is always exact; no exceptions
                                                    398 * are expected
                                                    399 *
                                                    400 ************************************************************************


00000440  9823 A000                 00000000   402 CDLFBR   LM    R2,R3,0(R10)  Get count and addr of test input values
00000444  9878 A008                 00000008   403          LM    R7,R8,8(R10)  Get address of result area and flag area.
00000448  1222                                 404          LTR   R2,R2         Any test cases?
0000044A  078D                                 405          BZR   R13           ..No, return to caller
0000044C  0DC0                                 406          BASR  R12,0         Set top of loop
                                               407 *
0000044E  5810 3000                 00000000   408          L     R1,0(,R3)     Get integer test value
00000452  B29D F2E4                 000002E4   409          LFPC  FPCREGNT      Set exceptions non-trappable
00000456  B391 0081                            410          CDLFBR FPR8,0,R1,0  Cvt uint-32 in GPR1 to float in FPR8
0000045A  6080 7000                 00000000   411          STD   FPR8,0(,R7)   Store long BFP result
0000045E  B29C 8000                 00000000   412          STFPC 0(R8)         Store resulting FPCR flags and DXC
                                               413 *
00000462  B29D F2E8                 000002E8   414          LFPC  FPCREGTR      Set exceptions trappable
00000466  B391 0081                            415          CDLFBR FPR8,0,R1,0  Cvt uint-32 in GPR1 to float in FPR0
0000046A  6080 7008                 00000008   416          STD   FPR8,8(,R7)   Store long BFP result
0000046E  B29C 8004                 00000004   417          STFPC 4(R8)         Store resulting FPCR flags and DXC
                                               418 *
00000472  4130 3004                 00000004   419          LA    R3,4(,R3)     Point to next input value
00000476  4170 7010                 00000010   420          LA    R7,16(,R7)    Point to next long BFP result pair
0000047A  4180 8008                 00000008   421          LA    R8,8(,R8)     Point to next FPCR/CC result pair
0000047E  062C                                 422          BCTR  R2,R12        Convert next input value.
00000480  07FD                                 423          BR    R13           All converted; return.
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                    425 ************************************************************************
                                                    426 *
                                                    427 * Convert integers to extended BFP format.  A pair of results is
                                                    428 * generated for each input: one with all exceptions non-trappable,
                                                    429 * and the second with all exceptions trappable.   The FPCR is
                                                    430 * stored for each result.  Conversion of a 32-bit integer to
                                                    431 * extended is always exact; no exceptions are expected
                                                    432 *
                                                    433 ************************************************************************


00000482  9823 A000                 00000000   435 CXLFBR   LM    R2,R3,0(R10)  Get count and addr of test input values
00000486  9878 A008                 00000008   436          LM    R7,R8,8(R10)  Get address of result area and flag area.
0000048A  1222                                 437          LTR   R2,R2         Any test cases?
0000048C  078D                                 438          BZR   R13           ..No, return to caller
0000048E  0DC0                                 439          BASR  R12,0         Set top of loop
                                               440 *
00000490  5810 3000                 00000000   441          L     R1,0(,R3)     Get integer test value
00000494  B29D F2E4                 000002E4   442          LFPC  FPCREGNT      Set exceptions non-trappable
00000498  B392 0081                            443          CXLFBR FPR8,0,R1,0  Cvt uint-32 in GPR1 to float in FPR8-FPR10
0000049C  6080 7000                 00000000   444          STD   FPR8,0(,R7)   Store extended BFP result part 1
000004A0  60A0 7008                 00000008   445          STD   FPR10,8(,R7)    Store extended BFP result part 1
000004A4  B29C 8000                 00000000   446          STFPC 0(R8)         Store resulting FPCR flags and DXC
                                               447 *
000004A8  B29D F2E8                 000002E8   448          LFPC  FPCREGTR      Set exceptions trappable
000004AC  B392 0081                            449          CXLFBR FPR8,0,R1,0  Cvt uint-32 in GPR1 to float in FPR8-FPR10
000004B0  6080 7010                 00000010   450          STD   FPR8,16(,R7)  Store extended BFP result part 1
000004B4  60A0 7018                 00000018   451          STD   FPR10,24(,R7) Store extended BFP result part 2
000004B8  B29C 8004                 00000004   452          STFPC 4(R8)         Store resulting FPCR flags and DXC
                                               453 *
000004BC  4130 3004                 00000004   454          LA    R3,4(,R3)     Point to next input value
000004C0  4170 7020                 00000020   455          LA    R7,32(,R7)    Point to next extended BFP result pair
000004C4  4180 8008                 00000008   456          LA    R8,8(,R8)     Point to next FPCR/CC result pair
000004C8  062C                                 457          BCTR  R2,R12        Convert next input value.
000004CA  07FD                                 458          BR    R13           All converted; return.
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                                    460 ********************************************************************************
                                                    461 *
                                                    462 * Short integer inputs for Convert From Logical testing.  The same set
                                                    463 * of inputs are used for short, long, and extended formats.  The last
                                                    464 * two values are used for rounding mode and exception tests for short
                                                    465 * only; conversion of uint-32 to long or extended are always exact.
                                                    466 *
                                                    467 ********************************************************************************


000004CC                                            469 INTIN    DS    0F
000004CC  00000001                                  470          DC    F'U1'
000004D0  00000002                                  471          DC    F'U2'
000004D4  00000004                                  472          DC    F'U4'
000004D8  FFFFFFFE                                  473 INTINRM  DC    F'U4294967294'      X'FFFFFFFE' - fits in short BFP
                                                    474 *                                 ..with loss of precision. Short BFP
                                                    475 *                                 ..result is 4,294,967,296
000004DC  FFFFFF00                                  476          DC    F'U4294967040'      X'FFFFFF00' - fits in short BFP
                                                    477 *                                 ..with no loss of precision
000004E0  FFFFFF80                                  478          DC    F'U4294967168'      X'FFFFFF40' - short BFP precision
                                                    479 *                                 ..may be lost based on rounding mode
000004E4                                            480          DS    0F            required by asma for following EQU to work.
                          00000018  00000001        481 INTCOUNT EQU   *-INTIN       Count of uint-32 in list * 4
                          0000000C  00000001        482 INTRMCT  EQU   *-INTINRM    Count of uint-32 for rounding tests * 4
```

```
 LOC        OBJECT CODE       ADDR1     ADDR2     STMT
                                          484 ************************************************************************
                                          485 *                    ACTUAL results saved here
                                          486 ************************************************************************
                                          487 *
                                          488 *                 Locations for ACTUAL results
                                          489 *
                                          490 *
                            00001000  00000001    491 SBFPOUT  EQU   STRTLABL+X'1000'    Short BFP from uint-32 inputs
                                          492 *                                        ..7 pairs used, room for 32 pairs
                            00001100  00000001    493 SBFPFLGS EQU   STRTLABL+X'1100'    FPCR flags and DXC from short BFP
                                          494 *                                        ..7 pairs used, room for 32 pairs
                            00001200  00000001    495 SBFPRMO  EQU   STRTLABL+X'1200'    Short BFP rounding mode results
                                          496 *                                        ..3 sets used, room for 16
                            00001500  00000001    497 SBFPRMOF EQU   STRTLABL+X'1500'    Short BFP rndg mode FPCR contents
                                          498 *                                        ..3 sets used, 16+
                                          499 *
                            00002000  00000001    500 LBFPOUT  EQU   STRTLABL+X'2000'    Long BFP values from uint-32 inputs
                                          501 *                                        ..7 pairs used, room for 16 pairs
                            00002100  00000001    502 LBFPFLGS EQU   STRTLABL+X'2100'    FPCR flags and DXC from long BFP
                                          503 *                                        ..7 pairs used, room for 16 pairs
                                          504 *
                            00003000  00000001    505 XBFPOUT  EQU   STRTLABL+X'3000'    Extended BFP values from uint-32
                                          506 *                                        ..7 pairs used, room for 16 pairs
                            00003200  00000001    507 XBFPFLGS EQU   STRTLABL+X'3200'    FPCR flags and DXC from long BFP
                                          508 *                                        ..7 pairs used, room for 16 pairs
                                          509 *
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                                    511 ********************************************************************************
                                                    512 *                       EXPECTED results
                                                    513 ********************************************************************************
                                                    514 *
000004E4                      000004E4  00004000    515          ORG   STRTLABL+X'4000'   (past end of actual results)
                                                    516 *
                              00004000  00000001    517 SBFPOUT_GOOD EQU *
00004000  C3C5D3C6 C2D94099                         518  DC CL48'CELFBR result pairs 1-2'
00004030  3F800000 3F800000                         519  DC XL16'3F8000003F800004000000040000000'
00004040  C3C5D3C6 C2D94099                         520  DC CL48'CELFBR result pairs 3-4'
00004070  40800000 40800000                         521  DC XL16'40800000408000004F8000004F800000'
00004080  C3C5D3C6 C2D94099                         522  DC CL48'CELFBR result pairs 5-6'
000040B0  4F7FFFFF 4F7FFFFF                         523  DC XL16'4F7FFFFF4F7FFFFF4F8000004F800000'
                              00000003  00000001    524 SBFPOUT_NUM EQU (*-SBFPOUT_GOOD)/64
                                                    525 *
                                                    526 *
                              000040C0  00000001    527 SBFPFLGS_GOOD EQU *
000040C0  C3C5D3C6 C2D940C6                         528  DC CL48'CELFBR FPC pairs 1-2'
000040F0  00000000 F8000000                         529  DC XL16'00000000F8000000000000F8000000'
00004100  C3C5D3C6 C2D940C6                         530  DC CL48'CELFBR FPC pairs 3-4'
00004130  00000000 F8000000                         531  DC XL16'00000000F8000000000080000F8000C00'
00004140  C3C5D3C6 C2D940C6                         532  DC CL48'CELFBR FPC pairs 5-6'
00004170  00000000 F8000000                         533  DC XL16'00000000F8000000000080000F8000C00'
                              00000003  00000001    534 SBFPFLGS_NUM EQU (*-SBFPFLGS_GOOD)/64
                                                    535 *
                                                    536 *
                              00004180  00000001    537 SBFPRMO_GOOD EQU *
00004180  C3C5D3C6 C2D94094                         538  DC CL48'CELFBR maxint-32 result FPC modes 1-3, 7'
000041B0  4F7FFFFF 4F800000                         539  DC XL16'4F7FFFFF4F8000004F7FFFFF4F7FFFFF'
000041C0  C3C5D3C6 C2D94094                         540  DC CL48'CELFBR maxint-32 result M3 modes 1, 3-5'
000041F0  4F800000 4F7FFFFF                         541  DC XL16'4F8000004F7FFFFF4F8000004F7FFFFF'
00004200  C3C5D3C6 C2D94094                         542  DC CL48'CELFBR maxint-32 result M3 modes 6, 7'
00004230  4F800000 4F7FFFFF                         543  DC XL16'4F8000004F7FFFFF0000000000000000'
00004240  C3C5D3C6 C2D940F0                         544  DC CL48'CELFBR 0xFFFFFF00 result FPC modes 1-3, 7'
00004270  4F7FFFFF 4F7FFFFF                         545  DC XL16'4F7FFFFF4F7FFFFF4F7FFFFF4F7FFFFF'
00004280  C3C5D3C6 C2D940F0                         546  DC CL48'CELFBR 0xFFFFFF00 result M3 modes 1, 3-5'
000042B0  4F7FFFFF 4F7FFFFF                         547  DC XL16'4F7FFFFF4F7FFFFF4F7FFFFF4F7FFFFF'
000042C0  C3C5D3C6 C2D940F0                         548  DC CL48'CELFBR 0xFFFFFF00 result M3 modes 6, 7'
000042F0  4F7FFFFF 4F7FFFFF                         549  DC XL16'4F7FFFFF4F7FFFFF0000000000000000'
00004300  C3C5D3C6 C2D940F0                         550  DC CL48'CELFBR 0xFFFFFF40 result FPC modes 1-3, 7'
00004330  4F7FFFFF 4F800000                         551  DC XL16'4F7FFFFF4F8000004F7FFFFF4F7FFFFF'
00004340  C3C5D3C6 C2D940F0                         552  DC CL48'CELFBR 0xFFFFFF40 result M3 modes 1, 3-5'
00004370  4F800000 4F7FFFFF                         553  DC XL16'4F8000004F7FFFFF4F8000004F7FFFFF'
00004380  C3C5D3C6 C2D940F0                         554  DC CL48'CELFBR 0xFFFFFF40 result M3 modes 6, 7'
000043B0  4F800000 4F7FFFFF                         555  DC XL16'4F8000004F7FFFFF0000000000000000'
                              00000009  00000001    556 SBFPRMO_NUM EQU (*-SBFPRMO_GOOD)/64
                                                    557 *
                                                    558 *
                              000043C0  00000001    559 SBFPRMOF_GOOD EQU *
000043C0  C3C5D3C6 C2D94094                         560  DC CL48'CELFBR maxint-32 FPC modes 1-3, 7 FPCR'
000043F0  00000001 00000002                         561  DC XL16'00000001000000020000000300000007'
00004400  C3C5D3C6 C2D94094                         562  DC CL48'CELFBR maxint-32 M3 modes 1, 3-5 FPCR'
00004430  00080000 00080000                         563  DC XL16'00080000000800000008000000080000'
00004440  C3C5D3C6 C2D94094                         564  DC CL48'CELFBR maxint-32 M3 modes 5-7'
00004470  00080000 00080000                         565  DC XL16'00080000000800000000000000000000'
00004480  C3C5D3C6 C2D940F0                         566  DC CL48'CELFBR 0xFFFFFF00 FPC modes 1-3, 7 FPCR'
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

000044B0  00000001 00000002                          567  DC XL16'00000001000000020000000300000007'
000044C0  C3C5D3C6 C2D940F0                          568  DC CL48'CELFBR 0xFFFFFF00 M3 modes 1, 3-5 FPCR'
000044F0  00000000 00000000                          569  DC XL16'00000000000000000000000000000000'
00004500  C3C5D3C6 C2D940F0                          570  DC CL48'CELFBR 0xFFFFFF00 M3 modes 6-7'
00004530  00000000 00000000                          571  DC XL16'00000000000000000000000000000000'
00004540  C3C5D3C6 C2D940F0                          572  DC CL48'CELFBR 0xFFFFFF40 FPC modes 1-3, 7 FPCR'
00004570  00000001 00000002                          573  DC XL16'00000001000000020000000300000007'
00004580  C3C5D3C6 C2D940F0                          574  DC CL48'CELFBR 0xFFFFFF40 M3 modes 1, 3-5 FPCR'
000045B0  00080000 00080000                          575  DC XL16'00080000000800000008000000080000'
000045C0  C3C5D3C6 C2D940F0                          576  DC CL48'CELFBR 0xFFFFFF40 M3 modes 6-7'
000045F0  00080000 00080000                          577  DC XL16'00080000000800000000000000000000'
                            00000009  00000001      578 SBFPRMOF_NUM EQU (*-SBFPRMOF_GOOD)/64
                                                     579 *
                                                     580 *
                            00004600  00000001      581 LBFPOUT_GOOD EQU *
00004600  C3C4D3C6 C2D94099                          582  DC CL48'CDLFBR result pair 1'
00004630  3FF00000 00000000                          583  DC XL16'3FF00000000000003FF0000000000000'
00004640  C3C4D3C6 C2D94099                          584  DC CL48'CDLFBR result pair 2'
00004670  40000000 00000000                          585  DC XL16'40000000000000004000000000000000'
00004680  C3C4D3C6 C2D94099                          586  DC CL48'CDLFBR result pair 3'
000046B0  40100000 00000000                          587  DC XL16'40100000000000004010000000000000'
000046C0  C3C4D3C6 C2D94099                          588  DC CL48'CDLFBR result pair 4'
000046F0  41EFFFFF FFC00000                          589  DC XL16'41EFFFFFFFC0000041EFFFFFFFC00000'
00004700  C3C4D3C6 C2D94099                          590  DC CL48'CDLFBR result pair 5'
00004730  41EFFFFF E0000000                          591  DC XL16'41EFFFFFE000000041EFFFFFE0000000'
00004740  C3C4D3C6 C2D94099                          592  DC CL48'CDLFBR result pair 6'
00004770  41EFFFFF F0000000                          593  DC XL16'41EFFFFFF000000041EFFFFFF0000000'
                            00000006  00000001      594 LBFPOUT_NUM EQU (*-LBFPOUT_GOOD)/64
                                                     595 *
                                                     596 *
                            00004780  00000001      597 LBFPFLGS_GOOD EQU *
00004780  C3C4D3C6 C2D940C6                          598  DC CL48'CDLFBR FPC pairs 1-2'
000047B0  00000000 F8000000                          599  DC XL16'00000000F800000000000000F8000000'
000047C0  C3C4D3C6 C2D940C6                          600  DC CL48'CDLFBR FPC pairs 3-4'
000047F0  00000000 F8000000                          601  DC XL16'00000000F800000000000000F8000000'
00004800  C3C4D3C6 C2D940C6                          602  DC CL48'CDLFBR FPC pairs 5-6'
00004830  00000000 F8000000                          603  DC XL16'00000000F800000000000000F8000000'
                            00000003  00000001      604 LBFPFLGS_NUM EQU (*-LBFPFLGS_GOOD)/64
                                                     605 *
                                                     606 *
                            00004840  00000001      607 XBFPOUT_GOOD EQU *
00004840  C3E7D3C6 C2D94099                          608  DC CL48'CXLFBR result 1a'
00004870  3FFF0000 00000000                          609  DC XL16'3FFF0000000000000000000000000000'
00004880  C3E7D3C6 C2D94099                          610  DC CL48'CXLFBR result 1b'
000048B0  3FFF0000 00000000                          611  DC XL16'3FFF0000000000000000000000000000'
000048C0  C3E7D3C6 C2D94099                          612  DC CL48'CXLFBR result 2a'
000048F0  40000000 00000000                          613  DC XL16'40000000000000000000000000000000'
00004900  C3E7D3C6 C2D94099                          614  DC CL48'CXLFBR result 2b'
00004930  40000000 00000000                          615  DC XL16'40000000000000000000000000000000'
00004940  C3E7D3C6 C2D94099                          616  DC CL48'CXLFBR result 3a'
00004970  40010000 00000000                          617  DC XL16'40010000000000000000000000000000'
00004980  C3E7D3C6 C2D94099                          618  DC CL48'CXLFBR result 3b'
000049B0  40010000 00000000                          619  DC XL16'40010000000000000000000000000000'
000049C0  C3E7D3C6 C2D94099                          620  DC CL48'CXLFBR result 4a'
000049F0  401EFFFF FFFC0000                          621  DC XL16'401EFFFFFFFC0000000000000000000'
00004A00  C3E7D3C6 C2D94099                          622  DC CL48'CXLFBR result 4b'
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

00004A30  401EFFFF FFFC0000                           623  DC XL16'401EFFFFFFFC0000000000000000000000'
00004A40  C3E7D3C6 C2D94099                           624  DC CL48'CXLFBR result 5a'
00004A70  401EFFFF FE000000                           625  DC XL16'401EFFFFFE0000000000000000000000'
00004A80  C3E7D3C6 C2D94099                           626  DC CL48'CXLFBR result 5b'
00004AB0  401EFFFF FE000000                           627  DC XL16'401EFFFFFE0000000000000000000000'
00004AC0  C3E7D3C6 C2D94099                           628  DC CL48'CXLFBR result 6a'
00004AF0  401EFFFF FF000000                           629  DC XL16'401EFFFFFF0000000000000000000000'
00004B00  C3E7D3C6 C2D94099                           630  DC CL48'CXLFBR result 6b'
00004B30  401EFFFF FF000000                           631  DC XL16'401EFFFFFF0000000000000000000000'
                            0000000C  00000001        632 XBFPOUT_NUM EQU (*-XBFPOUT_GOOD)/64
                                                      633 *
                                                      634 *
                            00004B40  00000001        635 XBFPFLGS_GOOD EQU *
00004B40  C3E7D3C6 C2D940C6                           636  DC CL48'CXLFBR FPC pairs 1-2'
00004B70  00000000 F8000000                           637  DC XL16'00000000F80000000000000F8000000'
00004B80  C3E7D3C6 C2D940C6                           638  DC CL48'CXLFBR FPC pairs 3-4'
00004BB0  00000000 F8000000                           639  DC XL16'00000000F80000000000000F8000000'
00004BC0  C3E7D3C6 C2D940C6                           640  DC CL48'CXLFBR FPC pairs 5-6'
00004BF0  00000000 F8000000                           641  DC XL16'00000000F80000000000000F8000000'
                            00000003  00000001        642 XBFPFLGS_NUM EQU (*-XBFPFLGS_GOOD)/64
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

00004C00                                              644 HELPERS  DS    0H       (R12 base of helper subroutines)

                                                      646 ********************************************************************
                                                      647 *                 REPORT UNEXPECTED PROGRAM CHECK
                                                      648 ********************************************************************

00004C00                                              650 PGMCK    DS    0H
00004C00  F342 C072 F08E      00004C72  0000008E      651          UNPK  PROGCODE(L'PROGCODE+1),PCINTCD(L'PCINTCD+1)
00004C06  926B C076                     00004C76      652          MVI   PGMCOMMA,C','
00004C0A  DC03 C072 C178      00004C72  00004D78      653          TR    PROGCODE,HEXTRTAB

00004C10  F384 C07C F150      00004C7C  00000150      655          UNPK  PGMPSW+(0*9)(9),PCOLDPSW+(0*4)(5)
00004C16  9240 C084                     00004C84      656          MVI   PGMPSW+(0*9)+8,C' '
00004C1A  DC07 C07C C178      00004C7C  00004D78      657          TR    PGMPSW+(0*9)(8),HEXTRTAB

00004C20  F384 C085 F154      00004C85  00000154      659          UNPK  PGMPSW+(1*9)(9),PCOLDPSW+(1*4)(5)
00004C26  9240 C08D                     00004C8D      660          MVI   PGMPSW+(1*9)+8,C' '
00004C2A  DC07 C085 C178      00004C85  00004D78      661          TR    PGMPSW+(1*9)(8),HEXTRTAB

00004C30  F384 C08E F158      00004C8E  00000158      663          UNPK  PGMPSW+(2*9)(9),PCOLDPSW+(2*4)(5)
00004C36  9240 C096                     00004C96      664          MVI   PGMPSW+(2*9)+8,C' '
00004C3A  DC07 C08E C178      00004C8E  00004D78      665          TR    PGMPSW+(2*9)(8),HEXTRTAB

00004C40  F384 C097 F15C      00004C97  0000015C      667          UNPK  PGMPSW+(3*9)(9),PCOLDPSW+(3*4)(5)
00004C46  9240 C09F                     00004C9F      668          MVI   PGMPSW+(3*9)+8,C' '
00004C4A  DC07 C097 C178      00004C97  00004D78      669          TR    PGMPSW+(3*9)(8),HEXTRTAB

00004C50  4100 0042                     00000042      671          LA    R0,L'PROGMSG     R0 <== length of message
00004C54  4110 C05E                     00004C5E      672          LA    R1,PROGMSG       R1 --> the message text itself
00004C58  4520 C27A                     00004E7A      673          BAL   R2,MSG           Go display this message
                                                      674
00004C5C  07FD                                        675          BR    R13              Return to caller




00004C5E                                              677 PROGMSG  DS   0CL66
00004C5E  D7D9D6C7 D9C1D440                           678          DC    CL20'PROGRAM CHECK! CODE '
00004C72  88888888                                    679 PROGCODE DC    CL4'hhhh'
00004C76  6B                                          680 PGMCOMMA DC    CL1','
00004C77  40D7E2E6 40                                 681          DC    CL5' PSW '
00004C7C  88888888 88888888                           682 PGMPSW   DC    CL36'hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh '
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

                                                      684 ************************************************************************
                                                      685 *                      VERIFICATION ROUTINE
                                                      686 ************************************************************************

00004CA0                                              688 VERISUB  DS    0H
                                                      689 *
                                                      690 **       Loop through the VERIFY TABLE...
                                                      691 *

00004CA0  4110 C32C                 00004F2C          693          LA    R1,VERIFTAB      R1 --> Verify table
00004CA4  4120 0008                 00000008          694          LA    R2,VERIFLEN      R2 <== Number of entries
00004CA8  0D30                                        695          BASR  R3,0             Set top of loop

00004CAA  9846 1000                 00000000          697          LM    R4,R6,0(R1)      Load verify table values
00004CAE  4D70 C0C2                 00004CC2          698          BAS   R7,VERIFY         Verify results
00004CB2  4110 100C                 0000000C          699          LA    R1,12(,R1)       Next verify table entry
00004CB6  0623                                        700          BCTR  R2,R3            Loop through verify table

00004CB8  9500 C278                 00004E78          702          CLI   FAILFLAG,X'00'   Did all tests verify okay?
00004CBC  078D                                        703          BER   R13              Yes, return to caller
00004CBE  47F0 F238                 00000238          704          B     FAIL             No, load FAILURE disabled wait PSW


                                                      706 *
                                                      707 **       Loop through the ACTUAL / EXPECTED results...
                                                      708 *

00004CC2  0D80                                        710 VERIFY   BASR  R8,0            Set top of loop

00004CC4  D50F 4000 5030      00000000  00000030      712          CLC   0(16,R4),48(R5)  Actual results == Expected results?
00004CCA  4770 C0DA                     00004CDA      713          BNE   VERIFAIL         No, show failure
00004CCE  4140 4010                     00000010      714 VERINEXT LA    R4,16(,R4)       Next actual result
00004CD2  4150 5040                     00000040      715          LA    R5,64(,R5)       Next expected result
00004CD6  0668                                        716          BCTR  R6,R8            Loop through results

00004CD8  07F7                                        718          BR    R7               Return to caller
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                    720 ************************************************************************
                                                    721 *                     Report the failure...
                                                    722 ************************************************************************

00004CDA  9005 C250                     00004E50   724 VERIFAIL STM   R0,R5,SAVER0R5   Save registers
00004CDE  92FF C278                     00004E78   725          MVI   FAILFLAG,X'FF'   Remember verification failure
                                                    726 *
                                                    727 **       First, show them the description...
                                                    728 *
00004CE2  D22F C1E0 5000      00004DE0  00000000   729          MVC   FAILDESC,0(R5)   Save results/test description
00004CE8  4100 0044                     00000044   730          LA    R0,L'FAILMSG1    R0 <== length of message
00004CEC  4110 C1CC                     00004DCC   731          LA    R1,FAILMSG1      R1 --> the message text itself
00004CF0  4520 C27A                     00004E7A   732          BAL   R2,MSG           Go display this message
                                                    733 *
                                                    734 **       Save address of actual and expected results
                                                    735 *
00004CF4  5040 C24C                     00004E4C   736          ST    R4,AACTUAL       Save A(actual results)
00004CF8  4150 5030                     00000030   737          LA    R5,48(,R5)       R5 ==> expected results
00004CFC  5050 C248                     00004E48   738          ST    R5,AEXPECT       Save A(expected results)
                                                    739 *
                                                    740 **       Format and show them the EXPECTED ("Want") results...
                                                    741 *
00004D00  D205 C210 C390      00004E10  00004F90   742          MVC   WANTGOT,=CL6'Want: '
00004D06  F384 C216 C248      00004E16  00004E48   743          UNPK  FAILADR(L'FAILADR+1),AEXPECT(L'AEXPECT+1)
00004D0C  9240 C21E                     00004E1E   744          MVI   BLANKEQ,C' '
00004D10  DC07 C216 C178      00004E16  00004D78   745          TR    FAILADR,HEXTRTAB

00004D16  F384 C221 5000      00004E21  00000000   747          UNPK  FAILVALS+(0*9)(9),(0*4)(5,R5)
00004D1C  9240 C229                     00004E29   748          MVI   FAILVALS+(0*9)+8,C' '
00004D20  DC07 C221 C178      00004E21  00004D78   749          TR    FAILVALS+(0*9)(8),HEXTRTAB

00004D26  F384 C22A 5004      00004E2A  00000004   751          UNPK  FAILVALS+(1*9)(9),(1*4)(5,R5)
00004D2C  9240 C232                     00004E32   752          MVI   FAILVALS+(1*9)+8,C' '
00004D30  DC07 C22A C178      00004E2A  00004D78   753          TR    FAILVALS+(1*9)(8),HEXTRTAB

00004D36  F384 C233 5008      00004E33  00000008   755          UNPK  FAILVALS+(2*9)(9),(2*4)(5,R5)
00004D3C  9240 C23B                     00004E3B   756          MVI   FAILVALS+(2*9)+8,C' '
00004D40  DC07 C233 C178      00004E33  00004D78   757          TR    FAILVALS+(2*9)(8),HEXTRTAB

00004D46  F384 C23C 500C      00004E3C  0000000C   759          UNPK  FAILVALS+(3*9)(9),(3*4)(5,R5)
00004D4C  9240 C244                     00004E44   760          MVI   FAILVALS+(3*9)+8,C' '
00004D50  DC07 C23C C178      00004E3C  00004D78   761          TR    FAILVALS+(3*9)(8),HEXTRTAB

00004D56  4100 0035                     00000035   763          LA    R0,L'FAILMSG2    R0 <== length of message
00004D5A  4110 C210                     00004E10   764          LA    R1,FAILMSG2      R1 --> the message text itself
00004D5E  4520 C27A                     00004E7A   765          BAL   R2,MSG           Go display this message
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                          767 *
                                          768 **       Format and show them the ACTUAL ("Got") results...
                                          769 *
00004D62  D205 C210 C396      00004E10  00004F96   770          MVC   WANTGOT,=CL6'Got:  '
00004D68  F384 C216 C24C      00004E16  00004E4C   771          UNPK  FAILADR(L'FAILADR+1),AACTUAL(L'AACTUAL+1)
00004D6E  9240 C21E                     00004E1E   772          MVI   BLANKEQ,C' '
00004D72  DC07 C216 C178      00004E16  00004D78   773          TR    FAILADR,HEXTRTAB

00004D78  F384 C221 4000      00004E21  00000000   775          UNPK  FAILVALS+(0*9)(9),(0*4)(5,R4)
00004D7E  9240 C229                     00004E29   776          MVI   FAILVALS+(0*9)+8,C' '
00004D82  DC07 C221 C178      00004E21  00004D78   777          TR    FAILVALS+(0*9)(8),HEXTRTAB

00004D88  F384 C22A 4004      00004E2A  00000004   779          UNPK  FAILVALS+(1*9)(9),(1*4)(5,R4)
00004D8E  9240 C232                     00004E32   780          MVI   FAILVALS+(1*9)+8,C' '
00004D92  DC07 C22A C178      00004E2A  00004D78   781          TR    FAILVALS+(1*9)(8),HEXTRTAB

00004D98  F384 C233 4008      00004E33  00000008   783          UNPK  FAILVALS+(2*9)(9),(2*4)(5,R4)
00004D9E  9240 C23B                     00004E3B   784          MVI   FAILVALS+(2*9)+8,C' '
00004DA2  DC07 C233 C178      00004E33  00004D78   785          TR    FAILVALS+(2*9)(8),HEXTRTAB

00004DA8  F384 C23C 400C      00004E3C  0000000C   787          UNPK  FAILVALS+(3*9)(9),(3*4)(5,R4)
00004DAE  9240 C244                     00004E44   788          MVI   FAILVALS+(3*9)+8,C' '
00004DB2  DC07 C23C C178      00004E3C  00004D78   789          TR    FAILVALS+(3*9)(8),HEXTRTAB

00004DB8  4100 0035                     00000035   791          LA    R0,L'FAILMSG2    R0 <== length of message
00004DBC  4110 C210                     00004E10   792          LA    R1,FAILMSG2      R1 --> the message text itself
00004DC0  4520 C27A                     00004E7A   793          BAL   R2,MSG           Go display this message

00004DC4  9805 C250                     00004E50   795          LM    R0,R5,SAVER0R5   Restore registers
00004DC8  47F0 C0CE                     00004CCE   796          B     VERINEXT         Continue with verification...



00004DCC                                          798 FAILMSG1 DS    0CL68
00004DCC  C3D6D4D7 C1D9C9E2                       799          DC    CL20'COMPARISON FAILURE! '
00004DE0  4D8485A2 83998997                       800 FAILDESC DC    CL48'(description)'


00004E10                                          802 FAILMSG2 DS    0CL53
00004E10  40404040 4040                           803 WANTGOT  DC    CL6' '           'Want: ' -or- 'Got:  '
00004E16  C1C1C1C1 C1C1C1C1                       804 FAILADR  DC    CL8'AAAAAAAA'
00004E1E  407E40                                  805 BLANKEQ  DC    CL3' = '
00004E21  88888888 88888888                       806 FAILVALS DC    CL36'hhhhhhhh hhhhhhhh hhhhhhhh hhhhhhhh '

00004E48  00000000                                808 AEXPECT  DC    F'0'             ==> Expected ("Want") results
00004E4C  00000000                                809 AACTUAL  DC    F'0'             ==> Actual ("Got") results
00004E50  00000000 00000000                       810 SAVER0R5 DC    6F'0'            Registers R0 - R5 save area
00004E68  F0F1F2F3 F4F5F6F7                       811 CHARHEX  DC    CL16'0123456789ABCDEF'
                            00004D78  00000010   812 HEXTRTAB EQU   CHARHEX-X'F0'    Hexadecimal translation table
00004E78  00                                      813 FAILFLAG DC    X'00'            FF = Fail, 00 = Success
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2     STMT

                                           815 ********************************************************************
                                           816 *        Issue HERCULES MESSAGE pointed to by R1, length in R0
                                           817 ********************************************************************

00004E7A  4900 C38C                 00004F8C   819 MSG      CH    R0,=H'0'              Do we even HAVE a message?
00004E7E  07D2                              820          BNHR  R2                    No, ignore

00004E80  9002 C2B0                 00004EB0   822          STM   R0,R2,MSGSAVE         Save registers

00004E84  4900 C38E                 00004F8E   824          CH    R0,=AL2(L'MSGMSG)     Message length within limits?
00004E88  47D0 C290                 00004E90   825          BNH   MSGOK                 Yes, continue
00004E8C  4100 005F                 0000005F   826          LA    R0,L'MSGMSG           No, set to maximum

00004E90  1820                              828 MSGOK    LR    R2,R0                 Copy length to work register
00004E92  0620                              829          BCTR  R2,0                  Minus-1 for execute
00004E94  4420 C2BC                 00004EBC   830          EX    R2,MSGMVC             Copy message to O/P buffer

00004E98  4120 200A                 0000000A   832          LA    R2,1+L'MSGCMD(,R2)    Calculate true command length
00004E9C  4110 C2C2                 00004EC2   833          LA    R1,MSGCMD             Point to true command

00004EA0  83120008                          835          DC    X'83',X'12',X'0008'   Issue Hercules Diagnose X'008'
00004EA4  4780 C2AA                 00004EAA   836          BZ    MSGRET                Return if successful
00004EA8  0000                              837          DC    H'0'                  CRASH for debugging purposes

00004EAA  9802 C2B0                 00004EB0   839 MSGRET   LM    R0,R2,MSGSAVE         Restore registers
00004EAE  07F2                              840          BR    R2                    Return to caller
```

```
00004EB0  00000000 00000000                 842 MSGSAVE  DC    3F'0'                 Registers save area
00004EBC  D200 C2CB 1000      00004ECB  00000000   843 MSGMVC   MVC   MSGMSG(0),0(R1)     Executed instruction

00004EC2  D4E2C7D5 D6C8405C                 845 MSGCMD   DC    C'MSGNOH * '          *** HERCULES MESSAGE COMMAND ***
00004ECB  40404040 40404040                 846 MSGMSG   DC    CL95' '               The message text to be displayed
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2     STMT
                                                     848 ********************************************************************************
                                                     849 *                            VERIFY TABLE
                                                     850 ********************************************************************************
                                                     851 *
                                                     852 *        A(actual results), A(expected results), A(#of results)
                                                     853 *
                                                     854 ********************************************************************************
00004F2C                                             856 VERIFTAB DC    0F'0'
00004F2C  00001000                                   857          DC    A(SBFPOUT)
00004F30  00004000                                   858          DC    A(SBFPOUT_GOOD)
00004F34  00000003                                   859          DC    A(SBFPOUT_NUM)
                                                     860 *
00004F38  00001100                                   861          DC    A(SBFPFLGS)
00004F3C  000040C0                                   862          DC    A(SBFPFLGS_GOOD)
00004F40  00000003                                   863          DC    A(SBFPFLGS_NUM)
                                                     864 *
00004F44  00001200                                   865          DC    A(SBFPRMO)
00004F48  00004180                                   866          DC    A(SBFPRMO_GOOD)
00004F4C  00000009                                   867          DC    A(SBFPRMO_NUM)
                                                     868 *
00004F50  00001500                                   869          DC    A(SBFPRMOF)
00004F54  000043C0                                   870          DC    A(SBFPRMOF_GOOD)
00004F58  00000009                                   871          DC    A(SBFPRMOF_NUM)
                                                     872 *
00004F5C  00002000                                   873          DC    A(LBFPOUT)
00004F60  00004600                                   874          DC    A(LBFPOUT_GOOD)
00004F64  00000006                                   875          DC    A(LBFPOUT_NUM)
                                                     876 *
00004F68  00002100                                   877          DC    A(LBFPFLGS)
00004F6C  00004780                                   878          DC    A(LBFPFLGS_GOOD)
00004F70  00000003                                   879          DC    A(LBFPFLGS_NUM)
                                                     880 *
00004F74  00003000                                   881          DC    A(XBFPOUT)
00004F78  00004840                                   882          DC    A(XBFPOUT_GOOD)
00004F7C  0000000C                                   883          DC    A(XBFPOUT_NUM)
                                                     884 *
00004F80  00003200                                   885          DC    A(XBFPFLGS)
00004F84  00004B40                                   886          DC    A(XBFPFLGS_GOOD)
00004F88  00000003                                   887          DC    A(XBFPFLGS_NUM)
                                                     888 *
                          00000008  00000001         889 VERIFLEN EQU   (*-VERIFTAB)/12    #of entries in verify table
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

00004F8C                                              891          END
00004F8C  0000                                        892                =H'0'
00004F8E  005F                                        893                =AL2(L'MSGMSG)
00004F90  E68195A3 7A40                               894                =CL6'Want: '
00004F96  C796A37A 4040                               895                =CL6'Got:  '
```

SYMBOL	TYPE	VALUE	LENGTH	DEFN	REFERENCES
AACTUAL	F	004E4C	4	809	736 771
AEXPECT	F	004E48	4	808	738 743
AHELPERS	A	00027C	4	200	190 227
BFPCVTFL	J	000000	20380	116	
BLANKEQ	C	004E1E	3	805	744 772
CDLFBR	I	000440	4	402	218
CELFBR	I	00032C	4	278	213
CELFBRA	I	00036E	4	320	215
CHARHEX	C	004E68	16	811	812
CTLR0	F	0002E0	4	237	208 209 210
CXLFBR	I	000482	4	435	221
EXTDS	F	00030C	4	259	220
FAIL	I	000238	4	198	704
FAILADR	C	004E16	8	804	743 745 771 773
FAILDESC	C	004DE0	48	800	729
FAILFLAG	X	004E78	1	813	702 725
FAILMSG1	C	004DCC	68	798	730 731
FAILMSG2	C	004E10	53	802	763 764 791 792
FAILPSW	X	0002D0	8	235	198
FAILVALS	C	004E21	36	806	747 748 749 751 752 753 755 756 757 759 760 761 775 776 777 779 780 781 783 784 785 787 788 789
FPCREGNT	X	0002E4	4	238	285 330 336 342 348 356 361 366 371 376 381 409 442
FPCREGTR	X	0002E8	4	239	290 414 448
FPR0	U	000000	1	137	
FPR1	U	000001	1	138	
FPR10	U	00000A	1	147	445 451
FPR11	U	00000B	1	148	
FPR12	U	00000C	1	149	
FPR13	U	00000D	1	150	
FPR14	U	00000E	1	151	
FPR15	U	00000F	1	152	
FPR2	U	000002	1	139	
FPR3	U	000003	1	140	
FPR4	U	000004	1	141	
FPR5	U	000005	1	142	
FPR6	U	000006	1	143	
FPR7	U	000007	1	144	
FPR8	U	000008	1	145	286 287 291 292 332 333 338 339 344 345 350 351 357 358 362 363 367 368 372 373 377 378 382 383 410 411 415 416 443 444 449 450
FPR9	U	000009	1	146	
GOODPSW	X	0002C0	8	234	231
HELPERS	H	004C00	2	644	155 200
HEXTRTAB	U	004D78	16	812	653 657 661 665 669 745 749 753 757 761 773 777 781 785 789
IMAGE	1	000000	20380	0	
INTCOUNT	U	000018	1	481	248 254 260
INTIN	F	0004CC	4	469	481 249 255 261
INTINRM	F	0004D8	4	473	482 266
INTRMCT	U	00000C	1	482	265
LBFPFLGS	U	002100	1	502	257 877
LBFPFLGS_GOOD	U	004780	1	597	604 878
LBFPFLGS_NUM	U	000003	1	604	879
LBFPOUT	U	002000	1	500	256 873
LBFPOUT_GOOD	U	004600	1	581	594 874
LBFPOUT_NUM	U	000006	1	594	875
LONGS	F	0002FC	4	253	217
MSG	I	004E7A	4	819	673 732 765 793

SYMBOL	TYPE	VALUE	LENGTH	DEFN	REFERENCES
MSGCMD	C	004EC2	9	845	832 833
MSGMSG	C	004ECB	95	846	826 843 824
MSGMVC	I	004EBC	6	843	830
MSGOK	I	004E90	2	828	825
MSGRET	I	004EAA	4	839	836
MSGSAVE	F	004EB0	4	842	822 839
PCINTCD	H	00008E	2	168	185 651
PCNOTDTA	I	00020C	4	189	186
PCOLDPSW	U	000150	1	170	187 655 659 663 667
PGMCK	H	004C00	2	650	191
PGMCOMMA	C	004C76	1	680	652
PGMPSW	C	004C7C	36	682	655 656 657 659 660 661 663 664 665 667 668 669
PROGCHK	H	000200	2	184	176
PROGCODE	C	004C72	4	679	651 653
PROGMSG	C	004C5E	66	677	671 672
PROGPSW	D	000228	8	197	196
R0	U	000000	1	118	189 192 208 210 671 724 730 763 791 795 819 822 824 826 828 839
R1	U	000001	1	119	284 286 291 326 332 338 344 350 357 362 367 372 377 382 408 410 415 441 443 449 672 693 697 699 731 764 792 833 843
R10	U	00000A	1	128	212 214 217 220 278 279 320 321 402 403 435 436
R11	U	00000B	1	129	
R12	U	00000C	1	130	155 190 227 282 298 324 389 406 422 439 457
R13	U	00000D	1	131	191 213 215 218 221 228 281 299 323 390 405 423 438 458 675 703
R14	U	00000E	1	132	194 195 229 230
R15	U	00000F	1	133	154 189 192
R2	U	000002	1	120	278 280 298 320 322 389 402 404 422 435 437 457 673 694 700 732 765 793 820 822 828 829 830 832 839 840
R3	U	000003	1	121	278 284 295 320 326 386 402 408 419 435 441 454 695 700
R4	U	000004	1	122	697 712 714 736 775 779 783 787
R5	U	000005	1	123	712 715 724 729 737 738 747 751 755 759 795
R6	U	000006	1	124	697 716
R7	U	000007	1	125	279 287 292 296 321 333 339 345 351 358 363 368 373 378 383 387 403 411 416 420 436 444 445 450 451 455 698 718
R8	U	000008	1	126	279 288 293 297 321 334 340 346 352 359 364 369 374 379 384 388 403 412 417 421 436 446 452 456 710 716
R9	U	000009	1	127	
RMSHORTS	A	00031C	4	265	214
SAVER0R5	F	004E50	4	810	724 795
SAVEREGS	F	00023C	4	199	189 192
SBFPFLGS	U	001100	1	493	251 861
SBFPFLGS_GOOD	U	0040C0	1	527	534 862
SBFPFLGS_NUM	U	000003	1	534	863
SBFPOUT	U	001000	1	491	250 857
SBFPOUT_GOOD	U	004000	1	517	524 858
SBFPOUT_NUM	U	000003	1	524	859
SBFPRMO	U	001200	1	495	267 865
SBFPRMOF	U	001500	1	497	268 869
SBFPRMOF_GOOD	U	0043C0	1	559	578 870
SBFPRMOF_NUM	U	000009	1	578	871
SBFPRMO_GOOD	U	004180	1	537	556 866
SBFPRMO_NUM	U	000009	1	556	867
SHORTS	F	0002EC	4	247	212
START	I	000280	4	208	173
STRTLABL	U	000000	1	117	167 170 172 175 183 491 493 495 497 500 502 505 507 515
VERIFAIL	I	004CDA	4	724	713
VERIFLEN	U	000008	1	889	694

SYMBOL	TYPE	VALUE	LENGTH	DEFN	REFERENCES
VERIFTAB	F	004F2C	4	856	889 693
VERIFY	I	004CC2	2	710	698
VERINEXT	I	004CCE	4	714	796
VERISUB	H	004CA0	2	688	228
WANTGOT	C	004E10	6	803	742 770
XBFPFLGS	U	003200	1	507	263 885
XBFPFLGS_GOOD	U	004B40	1	635	642 886
XBFPFLGS_NUM	U	000003	1	642	887
XBFPOUT	U	003000	1	505	262 881
XBFPOUT_GOOD	U	004840	1	607	632 882
XBFPOUT_NUM	U	00000C	1	632	883
=AL2(L'MSGMSG)	R	004F8E	2	893	824
=CL6'Got: '	C	004F96	6	895	770
=CL6'Want: '	C	004F90	6	894	742
=H'0'	H	004F8C	2	892	819

MACRO DEFN REFERENCES

No defined macros

```
   DESC      SYMBOL     SIZE      POS         ADDR

Entry: 0

Image       IMAGE      20380  0000-4F9B  0000-4F9B
  Region               20380  0000-4F9B  0000-4F9B
    CSECT   BFPCVTFL   20380  0000-4F9B  0000-4F9B
```

```
   STMT                                                    FILE NAME
1     c:\Users\Fish\Documents\Visual Studio 2008\Projects\MyProjects\ASMA-0\bfp-008-cvtfrlog\bfp-008-cvtfrlog.asm

** NO ERRORS FOUND **
```