```
 LOC         OBJECT CODE       ADDR1     ADDR2     STMT

                                                      2 *
                                                      3 ********************************************************************************
                                                      4 *
                                                      5 *Testcase str-001-mvst
                                                      6 *  Test cases for variations on the MVST (Move String) instruction.
                                                      7 *
                                                      8 ********************************************************************************
                                                      9 *
                                                     10 *                            str-001-mvst.asm
                                                     11 *
                                                     12 * Created and placed into public domain 2018-12-27 by Bob Polmanter.
                                                     13 * Remove runtest *Compare dependency on 2022-03-08 by Fish.
                                                     14 *
                                                     15 * The MVST instruction is tested against the definition in the
                                                     16 * z/Architecture Principles of Operation, SA22-7832.
                                                     17 *
                                                     18 * Test data is assembled into this program, and some test data is
                                                     19 * generated by this program. The program itself verifies the resulting
                                                     20 * status of registers and condition codes via simple CLC comparison.
                                                     21 *
                                                     22 *
                                                     23 * Tests performed with MVST (Move String):
                                                     24 *
                                                     25 * 1. Ensure that a non-zero bit in R0 bits 32-55 gives PIC06
                                                     26 * 2. Simple move; no operands cross page boundary
                                                     27 * 3. First byte moved is the termination character
                                                     28 * 4. Operand 1 crosses page boundary
                                                     29 * 5. Operand 2 crosses page boundary
                                                     30 * 6. Both operands cross page boundary, operand 1 is closer to boundary
                                                     31 * 7. Both operands cross page boundary, operand 2 is closer to boundary
                                                     32 * 8  Both operands cross boundary, both operands are the same distance
                                                     33 *       to the page boundary; large multipage move.
                                                     34 *
                                                     35 *
                                                     36 * NOTE - the nature of the string instructions is such that this test
                                                     37 *        case will only validate properly for the string instruction
                                                     38 *        improvement modifications committed in December 2018.  The
                                                     39 *        computation of the CPU determined number of bytes is an
                                                     40 *        unpredictable number on real hardware (at least above the
                                                     41 *        minimum value) and the method used in Hercules prior to
                                                     42 *        instruction improvements calculated it differently than the
                                                     43 *        improved method.  As a result, the operand registers will
                                                     44 *        likely contain different values when compared by the test
                                                     45 *        script due to the different CPU number of bytes
                                                     46 *        determined.  None of the methods are wrong, and failing
                                                     47 *        results in the test script are not necessarily wrong.
                                                     48 *        But this program and the resulting test script comparisons
                                                     49 *        were written for the method used by the improved string
                                                     50 *        instructions (CLST, MVST, SRST).
                                                     51 *
                                                     52 *
                                                     53 ********************************************************************************
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT
                                                      54 *
                                                      55 *
                                                      56 *
                              00000000  0000089F    57 MVST001  START 0
                              00000000  00000001    58 STRTLABL EQU   *
                              00000000  00000001    59 R0       EQU   0
                              00000001  00000001    60 R1       EQU   1
                              00000002  00000001    61 R2       EQU   2
                              00000003  00000001    62 R3       EQU   3
                              00000004  00000001    63 R4       EQU   4
                              00000005  00000001    64 R5       EQU   5
                              00000006  00000001    65 R6       EQU   6
                              00000007  00000001    66 R7       EQU   7
                              00000008  00000001    67 R8       EQU   8
                              00000009  00000001    68 R9       EQU   9
                              0000000A  00000001    69 R10      EQU   10
                              0000000B  00000001    70 R11      EQU   11                  **Reserved for z/CMS test rig
                              0000000C  00000001    71 R12      EQU   12
                              0000000D  00000001    72 R13      EQU   13
                              0000000E  00000001    73 R14      EQU   14                  **Return address for z/CMS test rig
                              0000000F  00000001    74 R15      EQU   15                  **Base register on z/CMS or Hyperion
                                                      75 *
                                                      76 *
00000000                      00000000              77          USING *,R15
                                                      78 *
                                                      79 * Selected z/Arch low core layout
                                                      80 *
00000000                      00000000  0000008C    81          ORG   STRTLABL+X'8C'      Program check interrution code
0000008C  00000000                                  82 PGMINTC  DS    F
                                                      83 *
                              00000150  00000001    84 PGMOPSW  EQU   STRTLABL+X'150'     z/Arch Program check old PSW
                                                      85 *
00000090                      00000090  000001A0    86          ORG   STRTLABL+X'1A0'     z/Arch Restart PSW
000001A0  00000001 80000000                         87          DC    X'0000000180000000',A(0,START)
                                                      88 *
000001B0                      000001B0  000001D0    89          ORG   STRTLABL+X'1D0'     z/Arch Program check new PSW
000001D0  00000001 80000000                         90 PGMNPSW  DC    X'0000000180000000',A(0,PROGCHK)
                                                      91 *
                                                      92 * Program check routine.  We are looking for a single specification
                                                      93 * exeception.  Any other program check is not expected to occur and
                                                      94 * results in a hard wait.
                                                      95 *
000001E0                      000001E0  00000200    96          ORG   STRTLABL+X'200'
00000200                                            97 PROGCHK  DS    0H                  Program check occured...
00000200  9500 F21C                     0000021C    98          CLI   DIDTHIS,X'00'      First/only time here?
00000204  4770 F218                     00000218    99          BNE   FAIL                No?! Then something is wrong!
00000208  9506 F08F                     0000008F   100          CLI   PGMINTC+3,X'06'     Specification Exception?
0000020C  4770 F218                     00000218   101          BNE   FAIL                No?! Then something is wrong!
00000210  92FF F21C                     0000021C   102          MVI   DIDTHIS,X'FF'      Remember we did this once already
00000214  47F0 F22E                     0000022E   103          B     CONTINUE            Continue, as this is expected (once!)
00000218  B2B2 F448                     00000448   104 FAIL     LPSWE FAILPSW             Unexpected PIC, disabled wait
0000021C  00                                       105 DIDTHIS  DC    X'00'               X'FF' == we already did this
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT
                                                      107 ********************************************************************************
                                                      108 *
                                                      109 *  Main program.
                                                      110 *
0000021E                                              111 START    DS    0H
                                                      112 *
                                                      113 **********
                                                      114 * TEST 1 * Ensure any non-zero bits in R0 bits 32-55 gives PIC 06
                                                      115 **********
                                                      116 *
0000021E  4100 0400                   00000400        117          LA    R0,X'400'              Set invalid termination char
00000222  4160 F800                   00000800        118          LA    R6,DEST1               -> destination field
00000226  4160 F700                   00000700        119          LA    R6,SHORT               -> source field
0000022A  B255 0067                                   120          MVST  R6,R7                  Attempt a move, should get PIC 6
                                                      121 *
                      0000022E        00000001        122 CONTINUE EQU   *
0000022E  95FF F21C                   0000021C        123          CLI   DIDTHIS,X'FF'          Did PIC 06 happen?
00000232  4770 F218                   00000218        124          BNE   FAIL                   No?! Then something is wrong!
00000236  D207 F1D0 F448  000001D0    00000448        125          MVC   PGMNPSW,FAILPSW        All other p checks should halt
                                                      126 *
                                                      127 **********
                                                      128 * TEST 2 * Move a short string; no page boundary crossings
                                                      129 **********
                                                      130 *
0000023C  4160 F810                   00000810        131          LA    R6,DEST2               -> destination field
00000240  4170 F700                   00000700        132          LA    R7,SHORT               -> source field
00000244  4D50 F418                   00000418        133          BAS   R5,MOVE                Move the string
00000248  9068 F820                   00000820        134          STM   R6,R8,RESULT2          Save test 2 result regs
                                                      135 *
                                                      136 **********
                                                      137 * TEST 3 * Move a single byte, which is the termination character
                                                      138 **********
                                                      139 *
0000024C  4160 F830                   00000830        140          LA    R6,DEST3               -> destination field
00000250  4170 F710                   00000710        141          LA    R7,TERM                -> String with only the term chr
00000254  4D50 F418                   00000418        142          BAS   R5,MOVE                Move the string
00000258  9068 F840                   00000840        143          STM   R6,R8,RESULT3          Save test 3 result regs
                                                      144 *
                                                      145 **********
                                                      146 * TEST 4 * Move a string; operand 1 (only) crosses a page boundary
                                                      147 **********
                                                      148 *
                                                      149 *-- First, generate a source string.  319 bytes, all FFs, + 1 $ char
                                                      150 *
0000025C  5820 F724                   00000724        151          L     R2,ASOURCE4            -> source string area
00000260  5830 F72C                   0000072C        152          L     R3,ALEN4               -> get length we will build
00000264  5850 F720                   00000720        153          L     R5,PAD                 Get the pad char
00000268  0E24                                        154          MVCL  R2,R4                  Fill the area with FFs
0000026A  0620                                        155          BCTR  R2,0                   -> last byte filled
0000026C  925B 2000                   00000000        156          MVI   0(R2),C'$'             Plug termination character
                                                      157 *
                                                      158 *-- Move the string to the destination area
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                          159 *
00000270  5860 F728                 00000728   160          L     R6,ADEST4               -> destination field
00000274  5870 F724                 00000724   161          L     R7,ASOURCE4             -> String to be moved
00000278  4D50 F418                 00000418   162          BAS   R5,MOVE                  Move the string
0000027C  9068 F850                 00000850   163          STM   R6,R8,RESULT4           Save test 4 result regs
                                          164 *
                                          165 *-- Finally, verify source and destination match completely
                                          166 *
00000280  D203 F85C F711      0000085C  00000711   167          MVC   RESULT4+12(4),FFS       Initialize later result field
00000286  5820 F724                 00000724   168          L     R2,ASOURCE4             -> source string area
0000028A  5830 F72C                 0000072C   169          L     R3,ALEN4                get length to validate
0000028E  5840 F728                 00000728   170          L     R4,ADEST4               -> destination area
00000292  1853                              0   171          LR    R5,R3                   Copy validation length
00000294  0F24                              0   172          CLCL  R2,R4                   Check if the strings match
00000296  B222 0000                         0   173          IPM   R0                      Get the condition code
0000029A  8800 001C                 0000001C   174          SRL   R0,28                   Adjust CC in register
0000029E  5000 F85C                 0000085C   175          ST    R0,RESULT4+12           Put in 4th word of result
                                          176 *
                                          177 **********
                                          178 * TEST 5 * Move a string; operand 2 (only) crosses a page boundary
                                          179 **********
                                          180 *
                                          181 *-- First, generate a source string.  599 bytes, all FFs, + 1 $ char
                                          182 *
000002A2  5820 F730                 00000730   183          L     R2,ASOURCE5             -> source string area
000002A6  5830 F738                 00000738   184          L     R3,ALEN5                -> get length we will build
000002AA  5850 F720                 00000720   185          L     R5,PAD                  Get the pad char
000002AE  0E24                              0   186          MVCL  R2,R4                   Fill the area with FFs
000002B0  0620                              0   187          BCTR  R2,0                    -> last byte filled
000002B2  925B 2000                 00000000   188          MVI   0(R2),C'$'              Plug termination character
                                          189 *
                                          190 *-- Move the string to the destination area
                                          191 *
000002B6  5860 F734                 00000734   192          L     R6,ADEST5               -> destination field
000002BA  5870 F730                 00000730   193          L     R7,ASOURCE5             -> String to be moved
000002BE  4D50 F418                 00000418   194          BAS   R5,MOVE                 Move the string
000002C2  9068 F860                 00000860   195          STM   R6,R8,RESULT5           Save test 4 result regs
                                          196 *
                                          197 *-- Finally, verify source and destination match completely
                                          198 *
000002C6  D203 F86C F711      0000086C  00000711   199          MVC   RESULT5+12(4),FFS       Initialize later result field
000002CC  5820 F730                 00000730   200          L     R2,ASOURCE5             -> source string area
000002D0  5830 F738                 00000738   201          L     R3,ALEN5                get length to validate
000002D4  5840 F734                 00000734   202          L     R4,ADEST5               -> destination area
000002D8  1853                              0   203          LR    R5,R3                   Copy validation length
000002DA  0F24                              0   204          CLCL  R2,R4                   Check if the strings match
000002DC  B222 0000                         0   205          IPM   R0                      Get the condition code
000002E0  8800 001C                 0000001C   206          SRL   R0,28                   Adjust CC in register
000002E4  5000 F86C                 0000086C   207          ST    R0,RESULT5+12           Put in 4th word of result
                                          208 *
                                          209 **********
                                          210 * TEST 6 * Move a string; both operands cross page boundary, but
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                                   211 ********** operand 1 is closer to the boundary than operand 2.
                                                   212 *
                                                   213 *-- First, generate a source string.  319 bytes, all FFs, + 1 $ char
                                                   214 *
000002E8  5820 F73C                    0000073C   215          L     R2,ASOURCE6              -> source string area
000002EC  5830 F744                    00000744   216          L     R3,ALEN6                 -> get length we will build
000002F0  5850 F720                    00000720   217          L     R5,PAD                   Get the pad char
000002F4  0E24                                    218          MVCL  R2,R4                    Fill the area with FFs
000002F6  0620                                    219          BCTR  R2,0                     -> last byte filled
000002F8  925B 2000                    00000000   220          MVI   0(R2),C'$'               Plug termination character
                                                   221 *
                                                   222 *-- Move the string to the destination area
                                                   223 *
000002FC  5860 F740                    00000740   224          L     R6,ADEST6                -> destination field
00000300  5870 F73C                    0000073C   225          L     R7,ASOURCE6              -> String to be moved
00000304  4D50 F418                    00000418   226          BAS   R5,MOVE                  Move the string
00000308  9068 F870                    00000870   227          STM   R6,R8,RESULT6            Save test 4 result regs
                                                   228 *
                                                   229 *-- Finally, verify source and destination match completely
                                                   230 *
0000030C  D203 F87C F711  0000087C     00000711   231          MVC   RESULT6+12(4),FFS        Initialize later result field
00000312  5820 F73C                    0000073C   232          L     R2,ASOURCE6              -> source string area
00000316  5830 F744                    00000744   233          L     R3,ALEN6                 get length to validate
0000031A  5840 F740                    00000740   234          L     R4,ADEST6                -> destination area
0000031E  1853                                    235          LR    R5,R3                    Copy validation length
00000320  0F24                                    236          CLCL  R2,R4                    Check if the strings match
00000322  B222 0000                               237          IPM   R0                       Get the condition code
00000326  8800 001C                    0000001C   238          SRL   R0,28                    Adjust CC in register
0000032A  5000 F87C                    0000087C   239          ST    R0,RESULT6+12            Put in 4th word of result
                                                   240 *
                                                   241 **********
                                                   242 * TEST 7 * Move a string; both operands cross page boundary, but
                                                   243 ********** operand 2 is closer to the boundary than operand 1.
                                                   244 *
                                                   245 *-- First, generate a source string.  319 bytes, all FFs, + 1 $ char
                                                   246 *
0000032E  5820 F748                    00000748   247          L     R2,ASOURCE7              -> source string area
00000332  5830 F750                    00000750   248          L     R3,ALEN7                 -> get length we will build
00000336  5850 F720                    00000720   249          L     R5,PAD                   Get the pad char
0000033A  0E24                                    250          MVCL  R2,R4                    Fill the area with FFs
0000033C  0620                                    251          BCTR  R2,0                     -> last byte filled
0000033E  925B 2000                    00000000   252          MVI   0(R2),C'$'               Plug termination character
                                                   253 *
                                                   254 *-- Move the string to the destination area
                                                   255 *
00000342  5860 F74C                    0000074C   256          L     R6,ADEST7                -> destination field
00000346  5870 F748                    00000748   257          L     R7,ASOURCE7              -> String to be moved
0000034A  4D50 F418                    00000418   258          BAS   R5,MOVE                  Move the string
0000034E  9068 F880                    00000880   259          STM   R6,R8,RESULT7            Save test 4 result regs
                                                   260 *
                                                   261 *-- Finally, verify source and destination match completely
                                                   262 *
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

00000352  D203 F88C F711      0000088C  00000711   263          MVC   RESULT7+12(4),FFS       Initialize later result field
00000358  5820 F748                     00000748   264          L     R2,ASOURCE7             -> source string area
0000035C  5830 F750                     00000750   265          L     R3,ALEN7                get length to validate
00000360  5840 F74C                     0000074C   266          L     R4,ADEST7               -> destination area
00000364  1853                                     267          LR    R5,R3                   Copy validation length
00000366  0F24                                     268          CLCL  R2,R4                   Check if the strings match
00000368  B222 0000                                269          IPM   R0                      Get the condition code
0000036C  8800 001C                     0000001C   270          SRL   R0,28                   Adjust CC in register
00000370  5000 F88C                     0000088C   271          ST    R0,RESULT7+12           Put in 4th word of result
                                                   272 *
                                                   273 ********** Move a string; both operands cross page boundary; both
                                                   274 * TEST 8 * operands are the same distance from a page boundary;
                                                   275 ********** larger multipage move.
                                                   276 *
                                                   277 *-- First, generate a source string.  12599 bytes, all FFs, + 1 $ char
                                                   278 *
00000374  5820 F754                     00000754   279          L     R2,ASOURCE8             -> source string area
00000378  5830 F75C                     0000075C   280          L     R3,ALEN8                -> get length we will build
0000037C  5850 F720                     00000720   281          L     R5,PAD                  Get the pad char
00000380  0E24                                     282          MVCL  R2,R4                   Fill the area with FFs
00000382  0620                                     283          BCTR  R2,0                    -> last byte filled
00000384  925B 2000                     00000000   284          MVI   0(R2),C'$'              Plug termination character
                                                   285 *
                                                   286 *-- Move the string to the destination area
                                                   287 *
00000388  5860 F758                     00000758   288          L     R6,ADEST8               -> destination field
0000038C  5870 F754                     00000754   289          L     R7,ASOURCE8             -> String to be moved
00000390  4D50 F418                     00000418   290          BAS   R5,MOVE                 Move the string
00000394  9068 F890                     00000890   291          STM   R6,R8,RESULT8           Save test 4 result regs
                                                   292 *
                                                   293 *-- Finally, verify source and destination match completely
                                                   294 *
00000398  D203 F89C F711      0000089C  00000711   295          MVC   RESULT8+12(4),FFS       Initialize later result field
0000039E  5820 F754                     00000754   296          L     R2,ASOURCE8             -> source string area
000003A2  5830 F75C                     0000075C   297          L     R3,ALEN8                get length to validate
000003A6  5840 F758                     00000758   298          L     R4,ADEST8               -> destination area
000003AA  1853                                     299          LR    R5,R3                   Copy validation length
000003AC  0F24                                     300          CLCL  R2,R4                   Check if the strings match
000003AE  B222 0000                                301          IPM   R0                      Get the condition code
000003B2  8800 001C                     0000001C   302          SRL   R0,28                   Adjust CC in register
000003B6  5000 F89C                     0000089C   303          ST    R0,RESULT8+12           Put in 4th word of result
                                                   304 *
                                                   305 **       Verify results...
                                                   306 *
000003BA  D50F F468 F810      00000468  00000810   307          CLC   GDEST2,DEST2            Expected results?
000003C0  4770 F218                     00000218   308          BNE   FAIL                    No?! Then something is wrong!
000003C4  D50B F478 F820      00000478  00000820   309          CLC   GRESLT2,RESULT2         Expected results?
000003CA  4770 F218                     00000218   310          BNE   FAIL                    No?! Then something is wrong!
000003CE  D503 F484 F830      00000484  00000830   311          CLC   GDEST3,DEST3            Expected results?
000003D4  4770 F218                     00000218   312          BNE   FAIL                    No?! Then something is wrong!
000003D8  D50B F488 F840      00000488  00000840   313          CLC   GRESLT3,RESULT3         Expected results?
000003DE  4770 F218                     00000218   314          BNE   FAIL                    No?! Then something is wrong!
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2    STMT

000003E2  D50F F494 F850      00000494  00000850   315          CLC   GRESLT4,RESULT4        Expected results?
000003E8  4770 F218                     00000218   316          BNE   FAIL                   No?! Then something is wrong!
000003EC  D50F F4A4 F860      000004A4  00000860   317          CLC   GRESLT5,RESULT5        Expected results?
000003F2  4770 F218                     00000218   318          BNE   FAIL                   No?! Then something is wrong!
000003F6  D50F F4B4 F870      000004B4  00000870   319          CLC   GRESLT6,RESULT6        Expected results?
000003FC  4770 F218                     00000218   320          BNE   FAIL                   No?! Then something is wrong!
00000400  D50F F4C4 F880      000004C4  00000880   321          CLC   GRESLT7,RESULT7        Expected results?
00000406  4770 F218                     00000218   322          BNE   FAIL                   No?! Then something is wrong!
0000040A  D50F F4D4 F890      000004D4  00000890   323          CLC   GRESLT8,RESULT8        Expected results?
00000410  4770 F218                     00000218   324          BNE   FAIL                   No?! Then something is wrong!
                                                   325 *
00000414  B2B2 F438                     00000438   326          LPSWE GOODPSW                load SUCCESS disabled wait PSW
                                                   327 *
                                                   328 *-- MVST routine used by tests
                                                   329 *
                                00000418  00000001 330 MOVE     EQU   *
00000418  4100 005B                     0000005B   331          LA    R0,C'$'                 Load termination character
0000041C  1B88                                     332          SR    R8,R8                   Init MVST counter
                                                   333 *
                                0000041E  00000001 334 INVOKE   EQU   *
0000041E  B255 0067                                335          MVST  R6,R7                   Move the string
00000422  4180 8001                     00000001   336          LA    R8,1(,R8)               Count executions of MVST
00000426  4710 F41E                     0000041E   337          BC    1,INVOKE                Restart the move
0000042A  0745                                     338          BCR   4,R5                    Complete if CC=1
                                                   339 *
0000042C  B222 0000                                340          IPM   R0                      Load failing CC
00000430  B2B2 F458                     00000458   341          LPSWE BADCC                   Here if invalid CC encountered
                                                   342 *
00000438                                           343          DS    0D              Ensure correct alignment for psw
00000438  00020000 00000000                        344 GOODPSW  DC    X'0002000000000000',A(0,0) Normal end - disabled wait
00000448  00020000 00000000                        345 FAILPSW  DC    X'0002000000000000',XL4'00',X'0000DEAD' Abnormal end
00000458  00020000 00000000                        346 BADCC    DC    X'0002000000000000',XL4'00',X'000BADCC' Abnormal end
                                                   347 *
00000468  E2C8D6D9 E340E2E3                        348 GDEST2   DC    XL16'E2C8D6D9E340E2E3D9C9D5C75B000000'
00000478  0000081C 00000700                        349 GRESLT2  DC    XL12'0000081C0000070000000001'
00000484  5B000000                                 350 GDEST3   DC    XL4'5B000000'
00000488  00000830 00000710                        351 GRESLT3  DC    XL12'000008300000071000000001'
00000494  0001204F 000012F0                        352 GRESLT4  DC    XL16'0001204F000012F00000000200000000'
000004A4  00013257 00004000                        353 GRESLT5  DC    XL16'00013257000040000000000200000000'
000004B4  0001611F 00006000                        354 GRESLT6  DC    XL16'0001611F00006000000000300000000'
000004C4  000180CF 00008030                        355 GRESLT7  DC    XL16'000180CF00008030000000300000000'
000004D4  0001C6F3 0000C000                        356 GRESLT8  DC    XL16'0001C6F30000C000000000400000000'
                                                   357 *
000004E4                        000004E4  00000700 358          ORG   STRTLABL+X'700'
00000700  E2C8D6D9 E340E2E3                        359 SHORT    DC    CL16'SHORT STRING$   '  Used by test 1 and 2
00000710  5B                                       360 TERM     DC    C'$'                    Used by test 3
00000711  FFFFFFFF FFFFFFFF                        361 FFS      DC    15X'FF'                 Program use
00000720  FF000000                                 362 PAD      DC    X'FF000000'             MVCL/CLCL pad char
00000724  00001200                                 363 ASOURCE4 DC    X'00001200'   op2       -> source string area (test 4)
00000728  00011F10                                 364 ADEST4   DC    X'00011F10'   op1       -> destination area (test 4)
0000072C  00000140                                 365 ALEN4    DC    F'320'                  Build len source 4 (incl term)
00000730  00003E02                                 366 ASOURCE5 DC    X'00003E02'   op2       -> source string area (test 5)
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2     STMT

00000734  00013000                                 367 ADEST5   DC    X'00013000'   op1       -> destination area (test 5)
00000738  00000258                                 368 ALEN5    DC    F'600'                  Build len source 5 (incl term)
0000073C  00005F80                                 369 ASOURCE6 DC    X'00005F80'   op2       -> source string area (test 6)
00000740  00015FE0                                 370 ADEST6   DC    X'00015FE0'   op1       -> destination area (test 6)
00000744  00000140                                 371 ALEN6    DC    F'320'                  Build len source 6 (incl term)
00000748  00007FC0                                 372 ASOURCE7 DC    X'00007FC0'   op2       -> source string area (test 7)
0000074C  00017F90                                 373 ADEST7   DC    X'00017F90'   op1       -> destination area (test 7)
00000750  00000140                                 374 ALEN7    DC    F'320'                  Build len source 7 (incl term)
00000754  00009620                                 375 ASOURCE8 DC    X'00009620'   op2       -> source string area (test 8)
00000758  00019620                                 376 ADEST8   DC    X'00019620'   op1       -> destination area (test 8)
0000075C  000030D4                                 377 ALEN8    DC    F'12500'                Build len source 8 (incl term)
                                                   378 *
                                                   379 *  Locations for results
                                                   380 *
                                                   381 * Result fields are kept on 16-byte boundaries to more easily
                                                   382 * track their assembled offsets for use in the .tst script.
                                                   383 *
                                                   384 *                                    offset
00000760                       00000760  00000800  385          ORG   STRTLABL+X'800'   8xx
00000800  00000000 00000000                        386 DEST1    DS    CL16                 00   Destination area test 1
00000810  00000000 00000000                        387 DEST2    DS    CL16                 10   Destination area test 2
00000820  00000000 00000000                        388 RESULT2  DS    4F                   20   Register results test 2
00000830  00000000 00000000                        389 DEST3    DS    CL16                 30   Destination area test 3
00000840  00000000 00000000                        390 RESULT3  DS    4F                   40   Register results test 3
00000850  00000000 00000000                        391 RESULT4  DS    4F                   50   Register results test 4
00000860  00000000 00000000                        392 RESULT5  DS    4F                   60   Register results test 5
00000870  00000000 00000000                        393 RESULT6  DS    4F                   70   Register results test 6
00000880  00000000 00000000                        394 RESULT7  DS    4F                   80   Register results test 7
00000890  00000000 00000000                        395 RESULT8  DS    4F                   90   Register results test 8
                                                   396 *
                                                   397          END
```

```
     SYMBOL         TYPE  VALUE   LENGTH  DEFN  REFERENCES

ADEST4               X    000728       4   364  160  170
ADEST5               X    000734       4   367  192  202
ADEST6               X    000740       4   370  224  234
ADEST7               X    00074C       4   373  256  266
ADEST8               X    000758       4   376  288  298
ALEN4                F    00072C       4   365  152  169
ALEN5                F    000738       4   368  184  201
ALEN6                F    000744       4   371  216  233
ALEN7                F    000750       4   374  248  265
ALEN8                F    00075C       4   377  280  297
ASOURCE4             X    000724       4   363  151  161  168
ASOURCE5             X    000730       4   366  183  193  200
ASOURCE6             X    00073C       4   369  215  225  232
ASOURCE7             X    000748       4   372  247  257  264
ASOURCE8             X    000754       4   375  279  289  296
BADCC                X    000458       8   346  341
CONTINUE             U    00022E       1   122  103
DEST1                C    000800      16   386  118
DEST2                C    000810      16   387  131  307
DEST3                C    000830      16   389  140  311
DIDTHIS              X    00021C       1   105   98  102  123
FAIL                 I    000218       4   104   99  101  124  308  310  312  314  316  318  320  322  324
FAILPSW              X    000448       8   345  104  125
FFS                  X    000711       1   361  167  199  231  263  295
GDEST2               X    000468      16   348  307
GDEST3               X    000484       4   350  311
GOODPSW              X    000438       8   344  326
GRESLT2              X    000478      12   349  309
GRESLT3              X    000488      12   351  313
GRESLT4              X    000494      16   352  315
GRESLT5              X    0004A4      16   353  317
GRESLT6              X    0004B4      16   354  319
GRESLT7              X    0004C4      16   355  321
GRESLT8              X    0004D4      16   356  323
IMAGE                1    000000    2208     0
INVOKE               U    00041E       1   334  337
MOVE                 U    000418       1   330  133  142  162  194  226  258  290
MVST001              J    000000    2208    57
PAD                  X    000720       4   362  153  185  217  249  281
PGMINTC              F    00008C       4    82  100
PGMNPSW              X    0001D0       8    90  125
PGMOPSW              U    000150       1    84
PROGCHK              H    000200       2    97   90
R0                   U    000000       1    59  117  173  174  175  205  206  207  237  238  239  269  270  271  301  302  303  331
                                                340
R1                   U    000001       1    60
R10                  U    00000A       1    69
R11                  U    00000B       1    70
R12                  U    00000C       1    71
R13                  U    00000D       1    72
R14                  U    00000E       1    73
R15                  U    00000F       1    74   77
```

SYMBOL	TYPE	VALUE	LENGTH	DEFN	REFERENCES
R2	U	000002	1	61	151 154 155 156 168 172 183 186 187 188 200 204 215 218 219 220 232 236 247 250 251 252 264 268 279 282 283 284 296 300
R3	U	000003	1	62	152 169 171 184 201 203 216 233 235 248 265 267 280 297 299
R4	U	000004	1	63	154 170 172 186 202 204 218 234 236 250 266 268 282 298 300
R5	U	000005	1	64	133 142 153 162 171 185 194 203 217 226 235 249 258 267 281 290 299 338
R6	U	000006	1	65	118 119 120 131 134 140 143 160 163 192 195 224 227 256 259 288 291 335
R7	U	000007	1	66	120 132 141 161 193 225 257 289 335
R8	U	000008	1	67	134 143 163 195 227 259 291 332 336
R9	U	000009	1	68	
RESULT2	F	000820	4	388	134 309
RESULT3	F	000840	4	390	143 313
RESULT4	F	000850	4	391	163 167 175 315
RESULT5	F	000860	4	392	195 199 207 317
RESULT6	F	000870	4	393	227 231 239 319
RESULT7	F	000880	4	394	259 263 271 321
RESULT8	F	000890	4	395	291 295 303 323
SHORT	C	000700	16	359	119 132
START	H	00021E	2	111	87
STRTLABL	U	000000	1	58	81 84 86 89 96 358 385
TERM	C	000710	1	360	141

MACRO DEFN REFERENCES

No defined macros

```
   DESC      SYMBOL   SIZE    POS       ADDR

Entry: 0

Image       IMAGE    2208  000-89F  000-89F
  Region             2208  000-89F  000-89F
    CSECT   MVST001  2208  000-89F  000-89F
```

```
   STMT                                                FILE NAME
1     c:\Users\Fish\Documents\Visual Studio 2008\Projects\MyProjects\ASMA-0\str-001-mvst\str-001-mvst.asm

** NO ERRORS FOUND **
```