```
  LOC        OBJECT CODE       ADDR1     ADDR2     STMT

                                                      2 ************************************************************************
                                                      3 *
                                                      4 *          Concurrent Block Update Consistency (CBUC) test
                                                      5 *
                                                      6 ************************************************************************
                                                      7 *
                                                      8 *  According to the POP, when storing a doubleword into a doubleword
                                                      9 *  using a memory copy operations, the destination storage area as
                                                     10 *  seen by other CPUs should ALWAYS present the complete operation,
                                                     11 *  and not any intermediate value.
                                                     12 *
                                                     13 *  What this means is, if the destination doubleword is 111... and
                                                     14 *  another CPU moves 222... to that area, any CPU that accesses the
                                                     15 *  destination doubleword should ALWAYS see either all 111... or all
                                                     16 *  222... but NEVER any intermediate value such as 1122111122221122.
                                                     17 *
                                                     18 *  Even though the 'MVC' and other instructions behave as if they
                                                     19 *  were moving one byte at a time, the hardware ensures that all
                                                     20 *  "Block Updates" (doubleword updates) are always CONSISTENT (i.e.
                                                     21 *  atomic), such that all bytes of a block are always updated at the
                                                     22 *  same time and never piecemeal.
                                                     23 *
                                                     24 *  This test attempts to detect any discrepancy in this area.
                                                     25 *
                                                     26 ************************************************************************
                                                     27 *
                                                     28 *                          Example test scripts
                                                     29 *
                                                     30 *                               (CBUC.tst)
                                                     31 *
                                                     32 * *Testcase CBUC (Concurrent Block Update Consistency)
                                                     33 * defsym      testdur  30   # (maximum test duration in seconds)
                                                     34 * mainsize    1
                                                     35 * numcpu      2
                                                     36 * sysclear
                                                     37 * archlvl     z/Arch
                                                     38 * loadcore    "$(testpath)/CBUC.core"
                                                     39 * script      "$(testpath)/CBUC.subtst"  &     # ('&' = async thread!)
                                                     40 * runtest     300                              # (subtst will stop it)
                                                     41 * *Done
                                                     42 * numcpu 1
                                                     43 *
                                                     44 *                             (CBUC.subtst)
                                                     45 *
                                                     46 * # CBUC test 'stop' thread...
                                                     47 * # This script is designed to run in a separate thread!
                                                     48 * pause $(testdur)    # Sleep for desired number of seconds
                                                     49 * r 500=FF            # And then force our test to stop
                                                     50 *
                                                     51 *
                                                     52 ************************************************************************
```

```
 LOC        OBJECT CODE       ADDR1     ADDR2     STMT

                                                    54 ********************************************************************************
                                                    55 *
                                                    56 *                          PROGRAMMING NOTE
                                                    57 *
                                                    58 *  The below loop values do NOT determine our test duration.  Rather,
                                                    59 *  it is our asynchronous 'cbuc.subtst' script that controls how long
                                                    60 *  our test runs by sleeping for the desired test duration number of
                                                    61 *  seconds and then sets the 'STOPFLAG' to a non-zero value to force
                                                    62 *  our test to end.  Using a value of zero for our loop value ensures
                                                    63 *  we can always support the maximum possible test duration.
                                                    64 *
                                                    65 ********************************************************************************

                          00000000  00000001      67 WRLOOPS  EQU   0                    Number of writer thread loops
                          00000000  00000001      68 RDLOOPS  EQU   0                    Number of reader thread loops


                                                    70 ********************************************************************************
                                                    71 *
                                                    72 *  CPU 1, in a tight loop, moves to the test area, using, in turn,
                                                    73 *  MVC, MVCL, and MVCLE, two alternate values: X'1111111111111111'
                                                    74 *  and X'2222222222222222'.
                                                    75 *
                                                    76 *  At the same time, CPU 0, also in a tight loop, using MVC, copies
                                                    77 *  the test area to a separate work area and verifies that the value
                                                    78 *  is either X'1111111111111111' or X'2222222222222222'. If any other
                                                    79 *  value is seen, then the test fails.
                                                    80 *
                                                    81 *  For the test to be relevant, it is best to perform this test on a
                                                    82 *  host system with more than one processor core. The more processors
                                                    83 *  (cores) that host system has, the better.
                                                    84 *
                                                    85 *  CPU 0:
                                                    86 *
                                                    87 *     MVC    WORK(8),DEST
                                                    88 *     CLC    WORK(4),WORK+4
                                                    89 *     BNE    FAIL
                                                    90 *     CLC    WORK(4),SRC1
                                                    91 *     BE     OK
                                                    92 *     CLC    WORK(4),SRC2
                                                    93 *     BNE    FAIL
                                                    94 *
                                                    95 *  CPU 1:
                                                    96 *
                                                    97 *     MVC    DEST(8),SRC1
                                                    98 *     MVCL   DEST(8),SRC2
                                                    99 *     MVCLE  DEST(8),SRC1
                                                   100 *     MVC    DEST(8),SRC2
                                                   101 *     MVCL   DEST(8),SRC1
                                                   102 *     MVCLE  DEST(8),SRC2
                                                   103 *
                                                   104 ********************************************************************************
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

                                                      106          PRINT OFF
                                                     3487          PRINT ON

                                                     3489 ********************************************************************************
                                                     3490 *        SATK prolog stuff...
                                                     3491 ********************************************************************************

                                                     3493          ARCHLVL  MNOTE=NO
                                                     3495+$AL      OPSYN AL
                                                     3496+$ALR     OPSYN ALR
                                                     3497+$B       OPSYN B
                                                     3498+$BAS     OPSYN BAS
                                                     3499+$BASR    OPSYN BASR
                                                     3500+$BC      OPSYN BC
                                                     3501+$BCTR    OPSYN BCTR
                                                     3502+$BE      OPSYN BE
                                                     3503+$BH      OPSYN BH
                                                     3504+$BL      OPSYN BL
                                                     3505+$BM      OPSYN BM
                                                     3506+$BNE     OPSYN BNE
                                                     3507+$BNH     OPSYN BNH
                                                     3508+$BNL     OPSYN BNL
                                                     3509+$BNM     OPSYN BNM
                                                     3510+$BNO     OPSYN BNO
                                                     3511+$BNP     OPSYN BNP
                                                     3512+$BNZ     OPSYN BNZ
                                                     3513+$BO      OPSYN BO
                                                     3514+$BP      OPSYN BP
                                                     3515+$BXLE    OPSYN BXLE
                                                     3516+$BZ      OPSYN BZ
                                                     3517+$CH      OPSYN CH
                                                     3518+$L       OPSYN L
                                                     3519+$LH      OPSYN LH
                                                     3520+$LM      OPSYN LM
                                                     3521+$LPSW    OPSYN LPSW
                                                     3522+$LR      OPSYN LR
                                                     3523+$LTR     OPSYN LTR
                                                     3524+$NR      OPSYN NR
                                                     3525+$SL      OPSYN SL
                                                     3526+$SLR     OPSYN SLR
                                                     3527+$SR      OPSYN SR
                                                     3528+$ST      OPSYN ST
                                                     3529+$STM     OPSYN STM
                                                     3530+$X       OPSYN X
                                                     3531+$AHI     OPSYN AHI
                                                     3532+$B       OPSYN J
                                                     3533+$BC      OPSYN BRC
                                                     3534+$BE      OPSYN JE
                                                     3535+$BH      OPSYN JH
                                                     3536+$BL      OPSYN JL
                                                     3537+$BM      OPSYN JM
                                                     3538+$BNE     OPSYN JNE
```

```
 LOC         OBJECT CODE       ADDR1     ADDR2     STMT

                                                   3539+$BNH     OPSYN JNH
                                                   3540+$BNL     OPSYN JNL
                                                   3541+$BNM     OPSYN JNM
                                                   3542+$BNO     OPSYN JNO
                                                   3543+$BNP     OPSYN JNP
                                                   3544+$BNZ     OPSYN JNZ
                                                   3545+$BO      OPSYN JO
                                                   3546+$BP      OPSYN JP
                                                   3547+$BXLE    OPSYN JXLE
                                                   3548+$BZ      OPSYN JZ
                                                   3549+$CHI     OPSYN CHI
                                                   3550+$AHI     OPSYN AGHI
                                                   3551+$AL      OPSYN ALG
                                                   3552+$ALR     OPSYN ALGR
                                                   3553+$BCTR    OPSYN BCTGR
                                                   3554+$BXLE    OPSYN JXLEG
                                                   3555+$CH      OPSYN CGH
                                                   3556+$CHI     OPSYN CGHI
                                                   3557+$L       OPSYN LG
                                                   3558+$LH      OPSYN LGH
                                                   3559+$LM      OPSYN LMG
                                                   3560+$LPSW    OPSYN LPSWE
                                                   3561+$LR      OPSYN LGR
                                                   3562+$LTR     OPSYN LTGR
                                                   3563+$NR      OPSYN NGR
                                                   3564+$SL      OPSYN SLG
                                                   3565+$SLR     OPSYN SLGR
                                                   3566+$SR      OPSYN SGR
                                                   3567+$ST      OPSYN STG
                                                   3568+$STM     OPSYN STMG
                                                   3569+$X       OPSYN XG
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                                  3571 ********************************************************************************
                                                  3572 *        Initiate the CBUC CSECT in the CODE region
                                                  3573 *        with the location counter at 0
                                                  3574 ********************************************************************************

                                                  3576 CBUC     ASALOAD  REGION=CODE
                                00000000  00000807 3577+CBUC     START 0,CODE
00000000  00020000 00000000                       3579+          PSW   0,0,2,0,X'008'      64-bit Restart ISR Trap New PSW
00000010                        00000010  00000058 3580+          ORG   CBUC+X'058'
00000058  00020000 00000000                       3582+          PSW   0,0,2,0,X'018'      64-bit External ISR Trap New PSW
00000068  00020000 00000000                       3583+          PSW   0,0,2,0,X'020'      64-bit Supervisor Call ISR Trap New PSW
00000078  00020000 00000000                       3584+          PSW   0,0,2,0,X'028'      64-bit Program ISR Trap New PSW
00000088  00020000 00000000                       3585+          PSW   0,0,2,0,X'030'      64-bit Machine Check Trap New PSW
00000098  00020000 00000000                       3586+          PSW   0,0,2,0,X'038'      64-bit Input/Output Trap New PSW
000000A8                        000000A8  000001A0 3587+          ORG   CBUC+X'1A0'
000001A0  00020000 00000000                       3589+          PSWZ  0,0,2,0,X'120'     Restart ISR Trap New PSW
000001B0  00020000 00000000                       3590+          PSWZ  0,0,2,0,X'130'     External ISR Trap New PSW
000001C0  00020000 00000000                       3591+          PSWZ  0,0,2,0,X'140'     Supervisor Call ISR Trap New PSW
000001D0  00020000 00000000                       3592+          PSWZ  0,0,2,0,X'150'     Program ISR Trap New PSW
000001E0  00020000 00000000                       3593+          PSWZ  0,0,2,0,X'160'     Machine Check Trap New PSW
000001F0  00020000 00000000                       3594+          PSWZ  0,0,2,0,X'170'     Input/Output Trap New PSW


                                                  3596 ********************************************************************************
                                                  3597 *        Define the z/Arch RESTART PSW
                                                  3598 ********************************************************************************

                                00000200  00000001 3600 PREVORG  EQU   *
00000200                        00000200  000001A0 3601          ORG   CBUC+X'1A0'
                                                  3602 *        PSWZ  <sys>,<key>,<mwp>,<prog>,<addr>[,amode]
000001A0  00000001 80000000                       3603          PSWZ  0,0,0,0,X'200',64
000001B0                        000001B0  00000200 3604          ORG   PREVORG


                                                  3606 ********************************************************************************
                                                  3607 *        Create IPL (restart) PSW
                                                  3608 ********************************************************************************

                                                  3610          ASAIPL   IA=BEGIN
                                00000000  00000807 3611+CBUC     CSECT
00000200                        00000200  00000000 3612+          ORG   CBUC
00000000  00080000 00000200                       3613+          PSWE390 0,0,0,0,BEGIN,24
00000008                        00000008  00000200 3614+          ORG   CBUC+512         Reset CSECT to end of assigned storage area
                                00000000  00000807 3615+CBUC     CSECT
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
                                                  3617 ********************************************************************************
                                                  3618 *                The actual CBUC program itself...
                                                  3619 ********************************************************************************

00000200                       00000000           3621          USING CBUC,R0                    No base registers needed

00000200  1F00                                    3623 BEGIN    SLR   R0,R0                     Start clean
00000202  4110 0001                    00000001   3624          LA    R1,1                       Request z/Arch mode
00000206  1F22                                    3625          SLR   R2,R2                     Start clean
00000208  1F33                                    3626          SLR   R3,R3                     Start clean
0000020A  AE02 0012                    00000012   3627          SIGP  R0,R2,X'12'               Request z/Arch mode

0000020E  1F11                                    3629          SLR   R1,R1                     Start clean
00000210  4120 0000                    00000000   3630          LA    R2,0                      Get our CPU number
00000214  4140 0224                    00000224   3631          LA    R4,BEGIN2                 Our restart entry point
00000218  4040 01AE                    000001AE   3632          STH   R4,X'1AE'                 Update restart PSW
0000021C  AE02 0006                    00000006   3633          SIGP  R0,R2,X'06'               Restart our CPU
00000220  47F0 0358                    00000358   3634          B     SIG1FAIL                  WTF?! How did we get here?!

00000224  4120 0001                    00000001   3636 BEGIN2   LA    R2,1                      Second CPU number
00000228  4140 028C                    0000028C   3637          LA    R4,WRITER                 Point to its entry point
0000022C  4040 01AE                    000001AE   3638          STH   R4,X'1AE'                 Update restart PSW
00000230  AE02 0006                    00000006   3639          SIGP  R0,R2,X'06'               Restart second CPU
00000234  4770 0368                    00000368   3640          BNZ   SIG2FAIL                  WTF?! (SIGP failed!)

00000238  B27C 0410                    00000410   3642          STCKF BEGCLOCK                  Get entry TOD
0000023C  D407 0410 0380       00000410  00000380  3643          NC    BEGCLOCK,=X'FFFFFFFFC0000000'  (0.25 seconds)

00000242  B27C 0418                    00000418   3645 WAITLOOP STCKF NOWCLOCK                  Get current TOD
00000246  D407 0418 0380       00000418  00000380  3646          NC    NOWCLOCK,=X'FFFFFFFFC0000000'  (0.25 seconds)
0000024C  D507 0418 0410       00000418  00000410  3647          CLC   NOWCLOCK,BEGCLOCK        Has 0.25 seconds passed yet?
00000252  4780 0242                    00000242   3648          BE    WAITLOOP                  Not yet. Keep waiting.




00000256  5800 0378                    00000378   3650 READER   L     R0,RDCOUNT                R0 <== loop count
0000025A  9500 0500                    00000500   3651 READLOOP CLI   STOPFLAG,X'00'           Are we being asked to stop?
0000025E  4770 0322                    00000322   3652          BNE   STOPTEST                  Yes, then do so.

00000262  D207 0800 0400       00000800  00000400  3654          MVC   WORK,READDEST             Grab copy of test value

00000268  D507 0800 0501       00000800  00000501  3656          CLC   WORK,PATTERN1             Is it all the first pattern?
0000026E  4770 027A                    0000027A   3657          BNE   READ2                     No, check if second pattern
00000272  4600 025A                    0000025A   3658          BCT   R0,READLOOP               Otherwise keep looping...
00000276  47F0 0322                    00000322   3659          B     STOPTEST                  Done!

0000027A  D507 0800 0513       00000800  00000513  3661 READ2    CLC   WORK,PATTERN2             Is it all the second pattern?
00000280  4770 0340                    00000340   3662          BNE   FAILTEST                  No?! Then *FAIL* immediately!
00000284  4600 025A                    0000025A   3663          BCT   R0,READLOOP               Otherwise keep looping...
00000288  47F0 0322                    00000322   3664          B     STOPTEST                  Done!
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

0000028C  5800 037C                    0000037C  3666 WRITER   L     R0,WRCOUNT              R0 <== loop count
00000290  9500 0500                    00000500  3667 WRITLOOP CLI   STOPFLAG,X'00'         Are we being asked to stop?
00000294  4770 0322                    00000322  3668          BNE   STOPTEST                Yes, then do so.
                                                  3669
00000298  9180 0600                    00000600  3670          TM    OPTFLAG,OPTMVC
0000029C  4780 02A6                    000002A6  3671          BZ              NOMVC1
000002A0  D20F 03FD 0501      000003FD  00000501  3672          MVC   WRITDEST,PATTERN1       Move 1st pattern to target
                              000002A6  00000001  3673 NOMVC1   EQU   *

000002A6  9140 0600                    00000600  3675          TM    OPTFLAG,OPTMVCL
000002AA  4780 02BE                    000002BE  3676          BZ              NOMVCL1
000002AE  4160 03FD                    000003FD  3677          LA    R6,WRITDEST             R6 --> destination
000002B2  4170 0010                    00000010  3678          LA    R7,L'WRITDEST           R7 <== destination length
000002B6  4180 0513                    00000513  3679          LA    R8,PATTERN2             R8 --> source
000002BA  1897                                   3680          LR    R9,R7                   R9 <== source length
000002BC  0E68                                   3681          MVCL  R6,R8                   move source to destination
                              000002BE  00000001  3682 NOMVCL1  EQU   *

000002BE  9120 0600                    00000600  3684          TM    OPTFLAG,OPTMVCLE
000002C2  4780 02D8                    000002D8  3685          BZ              NOMVCLE1
000002C6  4160 03FD                    000003FD  3686          LA    R6,WRITDEST             R6 --> destination
000002CA  4170 0010                    00000010  3687          LA    R7,L'WRITDEST           R7 <== destination length
000002CE  4180 0501                    00000501  3688          LA    R8,PATTERN1             R8 --> source
000002D2  1897                                   3689          LR    R9,R7                   R9 <== source length
000002D4  A868 0000                    00000000  3690          MVCLE R6,R8,0                 move source to destination
                              000002D8  00000001  3691 NOMVCLE1 EQU   *

000002D8  9180 0600                    00000600  3693          TM    OPTFLAG,OPTMVC
000002DC  4780 02E6                    000002E6  3694          BZ              NOMVC2
000002E0  D20F 03FD 0513      000003FD  00000513  3695          MVC   WRITDEST,PATTERN2       Move 1st pattern to target
                              000002E6  00000001  3696 NOMVC2   EQU   *

000002E6  9140 0600                    00000600  3698          TM    OPTFLAG,OPTMVCL
000002EA  4780 02FE                    000002FE  3699          BZ              NOMVCL2
000002EE  4160 03FD                    000003FD  3700          LA    R6,WRITDEST             R6 --> destination
000002F2  4170 0010                    00000010  3701          LA    R7,L'WRITDEST           R7 <== destination length
000002F6  4180 0501                    00000501  3702          LA    R8,PATTERN1             R8 --> source
000002FA  1897                                   3703          LR    R9,R7                   R9 <== source length
000002FC  0E68                                   3704          MVCL  R6,R8                   move source to destination
                              000002FE  00000001  3705 NOMVCL2  EQU   *

000002FE  9120 0600                    00000600  3707          TM    OPTFLAG,OPTMVCLE
00000302  4780 0318                    00000318  3708          BZ              NOMVCLE2
00000306  4160 03FD                    000003FD  3709          LA    R6,WRITDEST             R6 --> destination
0000030A  4170 0010                    00000010  3710          LA    R7,L'WRITDEST           R7 <== destination length
0000030E  4180 0513                    00000513  3711          LA    R8,PATTERN2             R8 --> source
00000312  1897                                   3712          LR    R9,R7                   R9 <== source length
00000314  A868 0000                    00000000  3713          MVCLE R6,R8,0                 move source to destination
                              00000318  00000001  3714 NOMVCLE2 EQU   *

00000318  4600 0290                    00000290  3716          BCT   R0,WRITLOOP             Otherwise keep looping...
0000031C  47F0 0322                    00000322  3717          B     STOPTEST                Done.
```

```
 LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                 3719 ********************************************************************************
                                                 3720 *                                PSWs
                                                 3721 ********************************************************************************


00000320  00                                     3723 FAILFLAG DC       X'00'                 X'FF' == test has failed

00000322  9500 0320                  00000320   3725 STOPTEST CLI      FAILFLAG,X'00'        Should test end normally?
00000326  4770 0340                  00000340   3726          BNE      FAILTEST              No! Test has failed!

0000032A  92FF 0500                  00000500   3728          MVI      STOPFLAG,X'FF'        Tell the other CPU to stop
                                                 3729          DWAITEND LOAD=YES              Normal completion
0000032E  8200 0338                  00000338   3731+         LPSW  DWAT0009
00000338  000A0000 00000000                     3732+DWAT0009 PSWE390 0,0,2,0,X'000000'


00000340  92FF 0320                  00000320   3734 FAILTEST MVI      FAILFLAG,X'FF'        Indicate test has failed!
00000344  92FF 0500                  00000500   3735          MVI      STOPFLAG,X'FF'        Tell the other CPU to stop
                                                 3736          DWAIT    LOAD=YES,CODE=BAD     Abnormal termination
00000348  8200 0350                  00000350   3737+         LPSW  DWAT0010
00000350  000A0000 00010BAD                     3738+DWAT0010 PSWE390 0,0,2,0,X'010BAD'


00000358  92FF 0500                  00000500   3740 SIG1FAIL MVI      STOPFLAG,X'FF'        Tell the other CPU to stop
                                                 3741          DWAIT    LOAD=YES,CODE=111     First SIGP failed
0000035C  8200 0360                  00000360   3742+         LPSW  DWAT0011
00000360  000A0000 00010111                     3743+DWAT0011 PSWE390 0,0,2,0,X'010111'


00000368  92FF 0500                  00000500   3745 SIG2FAIL MVI      STOPFLAG,X'FF'        Tell the other CPU to stop
                                                 3746          DWAIT    LOAD=YES,CODE=222     Second SIGP failed
0000036C  8200 0370                  00000370   3747+         LPSW  DWAT0012
00000370  000A0000 00010222                     3748+DWAT0012 PSWE390 0,0,2,0,X'010222'
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2     STMT
                                                    3750 ********************************************************************************
                                                    3751 *                          Working Storage
                                                    3752 ********************************************************************************

00000378  00000000                                  3754 RDCOUNT  DC    A(RDLOOPS)                Number of reader thread loops
0000037C  00000000                                  3755 WRCOUNT  DC    A(WRLOOPS)                Number of writer thread loops

00000380                                            3757          LTORG ,                         Literals pool
00000380  FFFFFFFF C0000000                         3758                =X'FFFFFFFFC0000000'



00000388                      00000388  000003F8    3760          ORG   CBUC+X'400'-8

000003F8  00000000 00                               3762          DC    XL5'0000000000'           Unaligned writer destination
000003FD                                            3763 WRITDEST DS    0CL16                     Writer thread destination
000003FD  C1C1C1                                    3764          DC    CL3'AAA'
00000400  C2C2C2C2 C2C2C2C2                         3765 READDEST DC    CL8'BBBBBBBB'             MUST be doubleword ALIGNED!
00000408  C1C1C1C1 C1                               3766          DC    CL5'AAAAA'

00000410  00000000 00000000                         3768 BEGCLOCK DC    D'0'                      CPU 0 entry TOD
00000418  00000000 00000000                         3769 NOWCLOCK DC    D'0'                      CPU 0 start TOD



00000420                      00000420  00000500    3771          ORG   CBUC+X'500'               Fixed address of 'stop' flag

00000500  00                                        3773 STOPFLAG DC    X'00'                     Set to non-zero to stop test
00000501  C1C1C1C1 C1C1C1C1                         3774 PATTERN1 DC    CL16'AAAAAAAAAAAAAAAA'    Should be unaligned
00000511  0000                                      3775          DC    XL2'0000'
00000513  C2C2C2C2 C2C2C2C2                         3776 PATTERN2 DC    CL16'BBBBBBBBBBBBBBBB'    Should also be unaligned



00000523                      00000523  00000600    3778          ORG   CBUC+X'600'               Fixed address of 'option' flag

                              00000080  00000001    3780 OPTMVC   EQU   X'80'                     Use 'MVC'   in write loop
                              00000040  00000001    3781 OPTMVCL  EQU   X'40'                     Use 'MVCL'  in write loop
                              00000020  00000001    3782 OPTMVCLE EQU   X'20'                     Use 'MVCLE' in write loop

00000600  E0                                        3784 OPTFLAG  DC    AL1(OPTMVC+OPTMVCL+OPTMVCLE)   Test options flag



00000601                      00000601  00000800    3786          ORG   CBUC+X'800'

00000800  40404040 40404040                         3788 WORK     DC    CL8' '                    MUST be doubleword ALIGNED!
```

```
  LOC         OBJECT CODE        ADDR1     ADDR2     STMT

                                 00000000  00000001  3791 R0        EQU   0
                                 00000001  00000001  3792 R1        EQU   1
                                 00000002  00000001  3793 R2        EQU   2
                                 00000003  00000001  3794 R3        EQU   3
                                 00000004  00000001  3795 R4        EQU   4
                                 00000005  00000001  3796 R5        EQU   5
                                 00000006  00000001  3797 R6        EQU   6
                                 00000007  00000001  3798 R7        EQU   7
                                 00000008  00000001  3799 R8        EQU   8
                                 00000009  00000001  3800 R9        EQU   9
                                 0000000A  00000001  3801 R10       EQU   10
                                 0000000B  00000001  3802 R11       EQU   11
                                 0000000C  00000001  3803 R12       EQU   12
                                 0000000D  00000001  3804 R13       EQU   13
                                 0000000E  00000001  3805 R14       EQU   14
                                 0000000F  00000001  3806 R15       EQU   15



                                           3808           END
```

SYMBOL	TYPE	VALUE	LENGTH	DEFN	REFERENCES
BEGCLOCK	D	000410	8	3768	3642 3643 3647
BEGIN	I	000200	2	3623	3613
BEGIN2	I	000224	4	3636	3631
CBUC	J	000000	2056	3577	3580 3587 3601 3612 3614 3760 3771 3778 3786 3621
CODE	2	000000	2056	3577	
DWAT0009	3	000338	8	3732	3731
DWAT0010	3	000350	8	3738	3737
DWAT0011	3	000360	8	3743	3742
DWAT0012	3	000370	8	3748	3747
FAILFLAG	X	000320	1	3723	3725 3734
FAILTEST	I	000340	4	3734	3662 3726
IMAGE	1	000000	2056	0	
NOMVC1	U	0002A6	1	3673	3671
NOMVC2	U	0002E6	1	3696	3694
NOMVCL1	U	0002BE	1	3682	3676
NOMVCL2	U	0002FE	1	3705	3699
NOMVCLE1	U	0002D8	1	3691	3685
NOMVCLE2	U	000318	1	3714	3708
NOWCLOCK	D	000418	8	3769	3645 3646 3647
OPTFLAG	R	000600	1	3784	3670 3675 3684 3693 3698 3707
OPTMVC	U	000080	1	3780	3670 3693 3784
OPTMVCL	U	000040	1	3781	3675 3698 3784
OPTMVCLE	U	000020	1	3782	3684 3707 3784
PATTERN1	C	000501	16	3774	3656 3672 3688 3702
PATTERN2	C	000513	16	3776	3661 3679 3695 3711
PREVORG	U	000200	1	3600	3604
R0	U	000000	1	3791	3621 3623 3627 3633 3639 3650 3658 3663 3666 3716
R1	U	000001	1	3792	3624 3629
R10	U	00000A	1	3801	
R11	U	00000B	1	3802	
R12	U	00000C	1	3803	
R13	U	00000D	1	3804	
R14	U	00000E	1	3805	
R15	U	00000F	1	3806	
R2	U	000002	1	3793	3625 3627 3630 3633 3636 3639
R3	U	000003	1	3794	3626
R4	U	000004	1	3795	3631 3632 3637 3638
R5	U	000005	1	3796	
R6	U	000006	1	3797	3677 3681 3686 3690 3700 3704 3709 3713
R7	U	000007	1	3798	3678 3680 3687 3689 3701 3703 3710 3712
R8	U	000008	1	3799	3679 3681 3688 3690 3702 3704 3711 3713
R9	U	000009	1	3800	3680 3689 3703 3712
RDCOUNT	A	000378	4	3754	3650
RDLOOPS	U	000000	1	68	3754
READ2	I	00027A	6	3661	3657
READDEST	C	000400	8	3765	3654
READER	I	000256	4	3650	
READLOOP	I	00025A	4	3651	3658 3663
SIG1FAIL	I	000358	4	3740	3634
SIG2FAIL	I	000368	4	3745	3640
STOPFLAG	X	000500	1	3773	3651 3667 3728 3735 3740 3745
STOPTEST	I	000322	4	3725	3652 3659 3664 3668 3717

```
     SYMBOL         TYPE   VALUE    LENGTH  DEFN  REFERENCES

WAITLOOP             I    000242        4  3645  3648
WORK                 C    000800        8  3788  3654  3656  3661
WRCOUNT              A    00037C        4  3755  3666
WRITDEST             C    0003FD       16  3763  3672  3677  3678  3686  3687  3695  3700  3701  3709  3710
WRITER               I    00028C        4  3666  3637
WRITLOOP             I    000290        4  3667  3716
WRLOOPS              U    000000        1    67  3755
=X'FFFFFFFFC0000000'
                     X    000380        8  3758  3643  3646
```

```
 MACRO     DEFN  REFERENCES

ANTR        172
APROB       304
ARCHIND     464   3494
ARCHLVL     605   3493
ASAIPL      731   3610
ASALOAD     811   3576
ASAREA      866
ASAZAREA   1051
CPUWAIT    1134
DSECTS     1460
DWAIT      1663   3730   3736   3741   3746
DWAITEND   1720   3729
ENADEV     1728
ESA390     1828
IOCB       1839
IOCBDS     2015
IOFMT      2049
IOINIT     2387
IOTRFR     2428
ORB        2476
POINTER    2665
PSWFMT     2693
RAWAIT     2827
RAWIO      2923
SIGCPU     3081
SMMGR      3139
SMMGRB     3239
TRAP128    3288   3588
TRAP64     3265   3578   3581
TRAPS      3301
ZARCH      3375
ZEROH      3387
ZEROL      3415
ZEROLH     3443
ZEROLL     3466
```

```
   DESC    SYMBOL  SIZE    POS      ADDR

Entry: 0

Image      IMAGE   2056  000-807  000-807
  Region   CODE    2056  000-807  000-807
    CSECT  CBUC    2056  000-807  000-807
```

```
   STMT                                          FILE NAME

1     c:\Users\Fish\Documents\Visual Studio 2008\Projects\MyProjects\ASMA-0\CBUC\CBUC.asm
2     C:\Users\Fish\Documents\Visual Studio 2008\Projects\Hercules\_Git\_Harold\SATK-0\srcasm\satk.mac


** NO ERRORS FOUND **
```