```
  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 **********************************************************************

                                                    28 **********************************************************************
                                                    29 *
                                                    30 *                        Example test scripts
                                                    31 *
                                                    32 *                             (CBUC.tst)
                                                    33 *
                                                    34 * *Testcase CBUC (Concurrent Block Update Consistency)
                                                    35 * defsym      testdur  30   # (maximum test duration in seconds)
                                                    36 * mainsize    1
                                                    37 * numcpu      2
                                                    38 * sysclear
                                                    39 * archlvl     z/Arch
                                                    40 * loadcore    "$(testpath)/CBUC.core"
                                                    41 * script      "$(testpath)/CBUC.subtst"  &     # ('&' = async thread!)
                                                    42 * runtest     300                              # (subtst will stop it)
                                                    43 * *Done
                                                    44 *
                                                    45 *                           (CBUC.subtst)
                                                    46 *
                                                    47 * # CBUC test 'stop' thread...
                                                    48 * # This script is designed to run in a separate thread!
                                                    49 * pause $(testdur)    # Sleep for desired number of seconds
                                                    50 * r 500=FF            # And then force our test to stop
                                                    51 *
                                                    52 *
                                                    53 **********************************************************************
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                    55 ***********************************************************************
                                                    56 *
                                                    57 *                        PROGRAMMING NOTE
                                                    58 *
                                                    59 *  The below loop values do NOT determine our test duration.  Rather,
                                                    60 *  it is our asynchronous 'cbuc.subtst' script that controls how long
                                                    61 *  our test runs by sleeping for the desired test duration number of
                                                    62 *  seconds and then sets the 'STOPFLAG' to a non-zero value to force
                                                    63 *  our test to end.  Using a value of zero for our loop value ensures
                                                    64 *  we can always support the maximum possible test duration.
                                                    65 *
                                                    66 **********************************************************************

                                    00000000  00000001    68 WRLOOPS  EQU   0                    Number of writer thread loops
                                    00000000  00000001    69 RDLOOPS  EQU   0                    Number of reader thread loops


                                                    71 **********************************************************************
                                                    72 *
                                                    73 *  CPU 1, in a tight loop, moves to the test area, using, in turn,
                                                    74 *  MVC, MVCL, and MVCLE, two alternate values: X'1111111111111111'
                                                    75 *  and X'2222222222222222'.
                                                    76 *
                                                    77 *  At the same time, CPU 0, also in a tight loop, using MVC, copies
                                                    78 *  the test area to a separate work area and verifies that the value
                                                    79 *  is either X'1111111111111111' or X'2222222222222222'. If any other
                                                    80 *  value is seen, then the test fails.
                                                    81 *
                                                    82 *  For the test to be relevant, it is best to perform this test on a
                                                    83 *  host system with more than one processor core. The more processors
                                                    84 *  (cores) that host system has, the better.
                                                    85 *
                                                    86 *  CPU 0:
                                                    87 *
                                                    88 *     MVC    WORK(8),DEST
                                                    89 *     CLC    WORK(4),WORK+4
                                                    90 *     BNE    FAIL
                                                    91 *     CLC    WORK(4),SRC1
                                                    92 *     BE     OK
                                                    93 *     CLC    WORK(4),SRC2
                                                    94 *     BNE    FAIL
                                                    95 *
                                                    96 *  CPU 1:
                                                    97 *
                                                    98 *     MVC    DEST(8),SRC1
                                                    99 *     MVCL   DEST(8),SRC2
                                                   100 *     MVCLE  DEST(8),SRC1
                                                   101 *     MVC    DEST(8),SRC2
                                                   102 *     MVCL   DEST(8),SRC1
                                                   103 *     MVCLE  DEST(8),SRC2
                                                   104 *
                                                   105 ***********************************************************************
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                   107          PRINT OFF
                                                  3488          PRINT ON

                                                  3490 ***********************************************************************
                                                  3491 *        SATK prolog stuff...
                                                  3492 ***********************************************************************

                                                  3494          ARCHLVL  MNOTE=NO
                                                  3496+$AL      OPSYN AL
                                                  3497+$ALR     OPSYN ALR
                                                  3498+$B       OPSYN B
                                                  3499+$BAS     OPSYN BAS
                                                  3500+$BASR    OPSYN BASR
                                                  3501+$BC      OPSYN BC
                                                  3502+$BCTR    OPSYN BCTR
                                                  3503+$BE      OPSYN BE
                                                  3504+$BH      OPSYN BH
                                                  3505+$BL      OPSYN BL
                                                  3506+$BM      OPSYN BM
                                                  3507+$BNE     OPSYN BNE
                                                  3508+$BNH     OPSYN BNH
                                                  3509+$BNL     OPSYN BNL
                                                  3510+$BNM     OPSYN BNM
                                                  3511+$BNO     OPSYN BNO
                                                  3512+$BNP     OPSYN BNP
                                                  3513+$BNZ     OPSYN BNZ
                                                  3514+$BO      OPSYN BO
                                                  3515+$BP      OPSYN BP
                                                  3516+$BXLE    OPSYN BXLE
                                                  3517+$BZ      OPSYN BZ
                                                  3518+$CH      OPSYN CH
                                                  3519+$L       OPSYN L
                                                  3520+$LH      OPSYN LH
                                                  3521+$LM      OPSYN LM
                                                  3522+$LPSW    OPSYN LPSW
                                                  3523+$LR      OPSYN LR
                                                  3524+$LTR     OPSYN LTR
                                                  3525+$NR      OPSYN NR
                                                  3526+$SL      OPSYN SL
                                                  3527+$SLR     OPSYN SLR
                                                  3528+$SR      OPSYN SR
                                                  3529+$ST      OPSYN ST
                                                  3530+$STM     OPSYN STM
                                                  3531+$X       OPSYN X
                                                  3532+$AHI     OPSYN AHI
                                                  3533+$B       OPSYN J
                                                  3534+$BC      OPSYN BRC
                                                  3535+$BE      OPSYN JE
                                                  3536+$BH      OPSYN JH
                                                  3537+$BL      OPSYN JL
                                                  3538+$BM      OPSYN JM
                                                  3539+$BNE     OPSYN JNE
  LOC        OBJECT CODE        ADDR1     ADDR2     STMT

                                                  3540+$BNH     OPSYN JNH
                                                  3541+$BNL     OPSYN JNL
                                                  3542+$BNM     OPSYN JNM
                                                  3543+$BNO     OPSYN JNO
                                                  3544+$BNP     OPSYN JNP
                                                  3545+$BNZ     OPSYN JNZ
                                                  3546+$BO      OPSYN JO
                                                  3547+$BP      OPSYN JP
                                                  3548+$BXLE    OPSYN JXLE
                                                  3549+$BZ      OPSYN JZ
                                                  3550+$CHI     OPSYN CHI
                                                  3551+$AHI     OPSYN AGHI
                                                  3552+$AL      OPSYN ALG
                                                  3553+$ALR     OPSYN ALGR
                                                  3554+$BCTR    OPSYN BCTGR
                                                  3555+$BXLE    OPSYN JXLEG
                                                  3556+$CH      OPSYN CGH
                                                  3557+$CHI     OPSYN CGHI
                                                  3558+$L       OPSYN LG
                                                  3559+$LH      OPSYN LGH
                                                  3560+$LM      OPSYN LMG
                                                  3561+$LPSW    OPSYN LPSWE
                                                  3562+$LR      OPSYN LGR
                                                  3563+$LTR     OPSYN LTGR
                                                  3564+$NR      OPSYN NGR
                                                  3565+$SL      OPSYN SLG
                                                  3566+$SLR     OPSYN SLGR
                                                  3567+$SR      OPSYN SGR
                                                  3568+$ST      OPSYN STG
                                                  3569+$STM     OPSYN STMG
                                                  3570+$X       OPSYN XG
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                  3572 ********************************************************************
                                                  3573 *        Initiate the CBUC CSECT in the CODE region
                                                  3574 *        with the location counter at 0
                                                  3575 ********************************************************************

                                                  3577 CBUC     ASALOAD  REGION=CODE
                              00000000  00000807  3578+CBUC     START 0,CODE
00000000  00020000 00000000                       3580+         PSW   0,0,2,0,X'008'       64-bit Restart ISR Trap New PSW
00000010                      00000010  00000058  3581+         ORG   CBUC+X'058'
00000058  00020000 00000000                       3583+         PSW   0,0,2,0,X'018'       64-bit External ISR Trap New PSW
00000068  00020000 00000000                       3584+         PSW   0,0,2,0,X'020'       64-bit Supervisor Call ISR Trap New PSW
00000078  00020000 00000000                       3585+         PSW   0,0,2,0,X'028'       64-bit Program ISR Trap New PSW
00000088  00020000 00000000                       3586+         PSW   0,0,2,0,X'030'       64-bit Machine Check Trap New PSW
00000098  00020000 00000000                       3587+         PSW   0,0,2,0,X'038'       64-bit Input/Output Trap New PSW
000000A8                      000000A8  000001A0  3588+         ORG   CBUC+X'1A0'
000001A0  00020000 00000000                       3590+         PSWZ  0,0,2,0,X'120'      Restart ISR Trap New PSW
000001B0  00020000 00000000                       3591+         PSWZ  0,0,2,0,X'130'      External ISR Trap New PSW
000001C0  00020000 00000000                       3592+         PSWZ  0,0,2,0,X'140'      Supervisor Call ISR Trap New PSW
000001D0  00020000 00000000                       3593+         PSWZ  0,0,2,0,X'150'      Program ISR Trap New PSW
000001E0  00020000 00000000                       3594+         PSWZ  0,0,2,0,X'160'      Machine Check Trap New PSW
000001F0  00020000 00000000                       3595+         PSWZ  0,0,2,0,X'170'      Input/Output Trap New PSW


                                                  3597 ********************************************************************
                                                  3598 *        Define the z/Arch RESTART PSW
                                                  3599 ********************************************************************

                              00000200  00000001  3601 PREVORG  EQU   *
00000200                      00000200  000001A0  3602          ORG   CBUC+X'1A0'
                                                  3603 *        PSWZ  <sys>,<key>,<mwp>,<prog>,<addr>[,amode]
000001A0  00000001 80000000                       3604          PSWZ  0,0,0,0,X'200',64
000001B0                      000001B0  00000200  3605          ORG   PREVORG


                                                  3607 ********************************************************************
                                                  3608 *        Create IPL (restart) PSW
                                                  3609 ********************************************************************

                                                  3611          ASAIPL   IA=BEGIN
                              00000000  00000807  3612+CBUC     CSECT
00000200                      00000200  00000000  3613+         ORG   CBUC
00000000  00080000 00000200                       3614+         PSWE390 0,0,0,0,BEGIN,24
00000008                      00000008  00000200  3615+         ORG   CBUC+512         Reset CSECT to end of assigned storage area
                              00000000  00000807  3616+CBUC     CSECT
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                  3618 **********************************************************************
                                                  3619 *              The actual CBUC program itself...
                                                  3620 **********************************************************************

00000200                      00000000            3622          USING CBUC,R0                   No base registers needed

00000200  1F00                                    3624 BEGIN    SLR   R0,R0                     Start clean
00000202  4110 0001                     00000001  3625          LA    R1,1                      Request z/Arch mode
00000206  1F22                                    3626          SLR   R2,R2                     Start clean
00000208  1F33                                    3627          SLR   R3,R3                     Start clean
0000020A  AE02 0012                     00000012  3628          SIGP  R0,R2,X'12'               Request z/Arch mode

0000020E  1F11                                    3630          SLR   R1,R1                     Start clean
00000210  4120 0000                     00000000  3631          LA    R2,0                      Get our CPU number
00000214  4140 0224                     00000224  3632          LA    R4,BEGIN2                 Our restart entry point
00000218  4040 01AE                     000001AE  3633          STH   R4,X'1AE'                 Update restart PSW
0000021C  AE02 0006                     00000006  3634          SIGP  R0,R2,X'06'               Restart our CPU
00000220  47F0 0328                     00000328  3635          B     SIG1FAIL                  WTF?! How did we get here?!

00000224  4120 0001                     00000001  3637 BEGIN2   LA    R2,1                      Second CPU number
00000228  4140 026E                     0000026E  3638          LA    R4,WRITER                 Point to its entry point
0000022C  4040 01AE                     000001AE  3639          STH   R4,X'1AE'                 Update restart PSW
00000230  AE02 0006                     00000006  3640          SIGP  R0,R2,X'06'               Restart second CPU
00000234  4770 0338                     00000338  3641          BNZ   SIG2FAIL                  WTF?! (SIGP failed!)
                                                  3642 *        B     READER                    Enter our own work loop




00000238  5800 0348                     00000348  3644 READER   L     R0,RDCOUNT                R0 <== loop count
0000023C  9500 0500                     00000500  3645 READLOOP CLI   STOPFLAG,X'00'            Are we being asked to stop?
00000240  4770 0302                     00000302  3646          BNE   GOODEOJ                   Yes, then do so.

00000244  D207 0800 0400      00000800  00000400  3648          MVC   WORK,READDEST             Grab copy of test value

0000024A  D507 0800 0501      00000800  00000501  3650          CLC   WORK,PATTERN1             Is it all the first pattern?
00000250  4770 025C                     0000025C  3651          BNE   READ2                     No, check if second pattern
00000254  4600 023C                     0000023C  3652          BCT   R0,READLOOP               Otherwise keep looping...
00000258  47F0 0302                     00000302  3653          B     GOODEOJ                   Done!

0000025C  D507 0800 0513      00000800  00000513  3655 READ2    CLC   WORK,PATTERN2             Is it all the second pattern?
00000262  4770 0318                     00000318  3656          BNE   FAILEOJ                   No?! Then *FAIL* immediately!
00000266  4600 023C                     0000023C  3657          BCT   R0,READLOOP               Otherwise keep looping...
0000026A  47F0 0302                     00000302  3658          B     GOODEOJ                   Done!
```

```
  LOC        OBJECT CODE        ADDR1     ADDR2     STMT

0000026E  5800 034C                     0000034C  3660 WRITER   L     R0,WRCOUNT               R0 <== loop count
00000272  9500 0500                     00000500  3661 WRITLOOP CLI   STOPFLAG,X'00'           Are we being asked to stop?
00000276  4770 0302                     00000302  3662          BNE   GOODEOJ                  Yes, then do so.
                                                  3663
0000027A  9180 0600                     00000600  3664          TM    OPTFLAG,OPTMVC
0000027E  4780 0288                     00000288  3665          BZ             NOMVC1
00000282  D20F 03FD 0501      000003FD  00000501  3666          MVC   WRITDEST,PATTERN1        Move 1st pattern to target
                              00000288  00000001  3667 NOMVC1   EQU   *

00000288  9140 0600                     00000600  3669          TM    OPTFLAG,OPTMVCL
0000028C  4780 02A0                     000002A0  3670          BZ             NOMVCL1
00000290  4160 03FD                     000003FD  3671          LA    R6,WRITDEST              R6 --> destination
00000294  4170 0010                     00000010  3672          LA    R7,L'WRITDEST            R7 <== destination length
00000298  4180 0513                     00000513  3673          LA    R8,PATTERN2              R8 --> source
0000029C  1897                                    3674          LR    R9,R7                    R9 <== source length
0000029E  0E68                                    3675          MVCL  R6,R8                    move source to destination
                              000002A0  00000001  3676 NOMVCL1  EQU   *

000002A0  9120 0600                     00000600  3678          TM    OPTFLAG,OPTMVCLE
000002A4  4780 02BA                     000002BA  3679          BZ             NOMVCLE1
000002A8  4160 03FD                     000003FD  3680          LA    R6,WRITDEST              R6 --> destination
000002AC  4170 0010                     00000010  3681          LA    R7,L'WRITDEST            R7 <== destination length
000002B0  4180 0501                     00000501  3682          LA    R8,PATTERN1              R8 --> source
000002B4  1897                                    3683          LR    R9,R7                    R9 <== source length
000002B6  A868 0000                     00000000  3684          MVCLE R6,R8,0                  move source to destination
                              000002BA  00000001  3685 NOMVCLE1 EQU   *

000002BA  9180 0600                     00000600  3687          TM    OPTFLAG,OPTMVC
000002BE  4780 02C8                     000002C8  3688          BZ             NOMVC2
000002C2  D20F 03FD 0513      000003FD  00000513  3689          MVC   WRITDEST,PATTERN2        Move 1st pattern to target
                              000002C8  00000001  3690 NOMVC2   EQU   *

000002C8  9140 0600                     00000600  3692          TM    OPTFLAG,OPTMVCL
000002CC  4780 02E0                     000002E0  3693          BZ             NOMVCL2
000002D0  4160 03FD                     000003FD  3694          LA    R6,WRITDEST              R6 --> destination
000002D4  4170 0010                     00000010  3695          LA    R7,L'WRITDEST            R7 <== destination length
000002D8  4180 0501                     00000501  3696          LA    R8,PATTERN1              R8 --> source
000002DC  1897                                    3697          LR    R9,R7                    R9 <== source length
000002DE  0E68                                    3698          MVCL  R6,R8                    move source to destination
                              000002E0  00000001  3699 NOMVCL2  EQU   *

000002E0  9120 0600                     00000600  3701          TM    OPTFLAG,OPTMVCLE
000002E4  4780 02FA                     000002FA  3702          BZ             NOMVCLE2
000002E8  4160 03FD                     000003FD  3703          LA    R6,WRITDEST              R6 --> destination
000002EC  4170 0010                     00000010  3704          LA    R7,L'WRITDEST            R7 <== destination length
000002F0  4180 0513                     00000513  3705          LA    R8,PATTERN2              R8 --> source
000002F4  1897                                    3706          LR    R9,R7                    R9 <== source length
000002F6  A868 0000                     00000000  3707          MVCLE R6,R8,0                  move source to destination
                              000002FA  00000001  3708 NOMVCLE2 EQU   *

000002FA  4600 0272                     00000272  3710          BCT   R0,WRITLOOP              Otherwise keep looping...
000002FE  47F0 0302                     00000302  3711          B     GOODEOJ                  Done.
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                  3713 ********************************************************************
                                                  3714 *                            PSWs
                                                  3715 ********************************************************************

00000302  92FF 0500                      00000500  3717 GOODEOJ  MVI      STOPFLAG,X'FF'        Tell the other CPU to stop
                                                  3718          DWAITEND LOAD=YES              Normal completion
00000306  8200 0310                      00000310  3720+         LPSW  DWAT0009
00000310  000A0000 00000000                       3721+DWAT0009 PSWE390 0,0,2,0,X'000000'


00000318  92FF 0500                      00000500  3723 FAILEOJ  MVI      STOPFLAG,X'FF'        Tell the other CPU to stop
                                                  3724          DWAIT    LOAD=YES,CODE=BAD     Abnormal termination
0000031C  8200 0320                      00000320  3725+         LPSW  DWAT0010
00000320  000A0000 00010BAD                       3726+DWAT0010 PSWE390 0,0,2,0,X'010BAD'


00000328  92FF 0500                      00000500  3728 SIG1FAIL MVI      STOPFLAG,X'FF'        Tell the other CPU to stop
                                                  3729          DWAIT    LOAD=YES,CODE=111     First SIGP failed
0000032C  8200 0330                      00000330  3730+         LPSW  DWAT0011
00000330  000A0000 00010111                       3731+DWAT0011 PSWE390 0,0,2,0,X'010111'


00000338  92FF 0500                      00000500  3733 SIG2FAIL MVI      STOPFLAG,X'FF'        Tell the other CPU to stop
                                                  3734          DWAIT    LOAD=YES,CODE=222     Second SIGP failed
0000033C  8200 0340                      00000340  3735+         LPSW  DWAT0012
00000340  000A0000 00010222                       3736+DWAT0012 PSWE390 0,0,2,0,X'010222'
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                  3738 **********************************************************************
                                                  3739 *                       Working Storage
                                                  3740 **********************************************************************

00000348  00000000                                3742 RDCOUNT  DC    A(RDLOOPS)                Number of reader thread loops
0000034C  00000000                                3743 WRCOUNT  DC    A(WRLOOPS)                Number of writer thread loops

00000350                                          3745          LTORG ,                         Literals pool




00000350                    00000350  000003F8    3747          ORG   CBUC+X'400'-8

000003F8  00000000 00                             3749          DC    XL5'0000000000'           Unaligned writer destination
000003FD                                          3750 WRITDEST DS    0CL16                     Writer thread destination
000003FD  C1C1C1                                  3751          DC    CL3'AAA'
00000400  C2C2C2C2 C2C2C2C2                       3752 READDEST DC    CL8'BBBBBBBB'             MUST be doubleword ALIGNED!
00000408  C1C1C1C1 C1                             3753          DC    CL5'AAAAA'




0000040D                    0000040D  00000500    3755          ORG   CBUC+X'500'               Fixed address of 'stop' flag

00000500  00                                      3757 STOPFLAG DC    X'00'                     Set to non-zero to stop test
00000501  C1C1C1C1 C1C1C1C1                       3758 PATTERN1 DC    CL16'AAAAAAAAAAAAAAAA'    Should be unaligned
00000511  0000                                    3759          DC    XL2'0000'
00000513  C2C2C2C2 C2C2C2C2                       3760 PATTERN2 DC    CL16'BBBBBBBBBBBBBBBB'    Should also be unaligned




00000523                    00000523  00000600    3762          ORG   CBUC+X'600'               Fixed address of 'stop' flag

                            00000080  00000001    3764 OPTMVC   EQU   X'80'                     Use 'MVC'   in write loop
                            00000040  00000001    3765 OPTMVCL  EQU   X'40'                     Use 'MVCL'  in write loop
                            00000020  00000001    3766 OPTMVCLE EQU   X'20'                     Use 'MVCLE' in write loop

00000600  E0                                      3768 OPTFLAG  DC    AL1(OPTMVC+OPTMVCL+OPTMVCLE)   Test options flag




00000601                    00000601  00000800    3770          ORG   CBUC+X'800'

00000800  40404040 40404040                       3772 WORK     DC    CL8' '                    MUST be doubleword ALIGNED!
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT



                              00000000  00000001  3775 R0        EQU   0
                              00000001  00000001  3776 R1        EQU   1
                              00000002  00000001  3777 R2        EQU   2
                              00000003  00000001  3778 R3        EQU   3
                              00000004  00000001  3779 R4        EQU   4
                              00000005  00000001  3780 R5        EQU   5
                              00000006  00000001  3781 R6        EQU   6
                              00000007  00000001  3782 R7        EQU   7
                              00000008  00000001  3783 R8        EQU   8
                              00000009  00000001  3784 R9        EQU   9
                              0000000A  00000001  3785 R10       EQU   10
                              0000000B  00000001  3786 R11       EQU   11
                              0000000C  00000001  3787 R12       EQU   12
                              0000000D  00000001  3788 R13       EQU   13
                              0000000E  00000001  3789 R14       EQU   14
                              0000000F  00000001  3790 R15       EQU   15



                                                  3792           END
```

```
     SYMBOL        TYPE  VALUE   LENGTH  DEFN  REFERENCES

BEGIN               I    000200       2  3624  3614
BEGIN2              I    000224       4  3637  3632
CBUC                J    000000    2056  3578  3581  3588  3602  3613  3615  3747  3755  3762  3770  3622
CODE                2    000000    2056  3578
DWAT0009            3    000310       8  3721  3720
DWAT0010            3    000320       8  3726  3725
DWAT0011            3    000330       8  3731  3730
DWAT0012            3    000340       8  3736  3735
FAILEOJ             I    000318       4  3723  3656
GOODEOJ             I    000302       4  3717  3646  3653  3658  3662  3711
IMAGE               1    000000    2056     0
NOMVC1              U    000288       1  3667  3665
NOMVC2              U    0002C8       1  3690  3688
NOMVCL1             U    0002A0       1  3676  3670
NOMVCL2             U    0002E0       1  3699  3693
NOMVCLE1            U    0002BA       1  3685  3679
NOMVCLE2            U    0002FA       1  3708  3702
OPTFLAG             R    000600       1  3768  3664  3669  3678  3687  3692  3701
OPTMVC              U    000080       1  3764  3664  3687  3768
OPTMVCL             U    000040       1  3765  3669  3692  3768
OPTMVCLE            U    000020       1  3766  3678  3701  3768
PATTERN1            C    000501      16  3758  3650  3666  3682  3696
PATTERN2            C    000513      16  3760  3655  3673  3689  3705
PREVORG             U    000200       1  3601  3605
R0                  U    000000       1  3775  3622  3624  3628  3634  3640  3644  3652  3657  3660  3710
R1                  U    000001       1  3776  3625  3630
R10                 U    00000A       1  3785
R11                 U    00000B       1  3786
R12                 U    00000C       1  3787
R13                 U    00000D       1  3788
R14                 U    00000E       1  3789
R15                 U    00000F       1  3790
R2                  U    000002       1  3777  3626  3628  3631  3634  3637  3640
R3                  U    000003       1  3778  3627
R4                  U    000004       1  3779  3632  3633  3638  3639
R5                  U    000005       1  3780
R6                  U    000006       1  3781  3671  3675  3680  3684  3694  3698  3703  3707
R7                  U    000007       1  3782  3672  3674  3681  3683  3695  3697  3704  3706
R8                  U    000008       1  3783  3673  3675  3682  3684  3696  3698  3705  3707
R9                  U    000009       1  3784  3674  3683  3697  3706
RDCOUNT             A    000348       4  3742  3644
RDLOOPS             U    000000       1    69  3742
READ2               I    00025C       6  3655  3651
READDEST            C    000400       8  3752  3648
READER              I    000238       4  3644
READLOOP            I    00023C       4  3645  3652  3657
SIG1FAIL            I    000328       4  3728  3635
SIG2FAIL            I    000338       4  3733  3641
STOPFLAG            X    000500       1  3757  3645  3661  3717  3723  3728  3733
WORK                C    000800       8  3772  3648  3650  3655
WRCOUNT             A    00034C       4  3743  3660
WRITDEST            C    0003FD      16  3750  3666  3671  3672  3680  3681  3689  3694  3695  3703  3704
```

```
     SYMBOL        TYPE  VALUE   LENGTH  DEFN  REFERENCES

WRITER               I    00026E        4  3660  3638
WRITLOOP             I    000272        4  3661  3710
WRLOOPS              U    000000        1    68  3743
```

```
 MACRO     DEFN  REFERENCES

ANTR        173
APROB       305
ARCHIND     465   3495
ARCHLVL     606   3494
ASAIPL      732   3611
ASALOAD     812   3577
ASAREA      867
ASAZAREA   1052
CPUWAIT    1135
DSECTS     1461
DWAIT      1664   3719   3724   3729   3734
DWAITEND   1721   3718
ENADEV     1729
ESA390     1829
IOCB       1840
IOCBDS     2016
IOFMT      2050
IOINIT     2388
IOTRFR     2429
ORB        2477
POINTER    2666
PSWFMT     2694
RAWAIT     2828
RAWIO      2924
SIGCPU     3082
SMMGR      3140
SMMGRB     3240
TRAP128    3289   3589
TRAP64     3266   3579   3582
TRAPS      3302
ZARCH      3376
ZEROH      3388
ZEROL      3416
ZEROLH     3444
ZEROLL     3467
```

```
   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 **
```