```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                     2 *********************************************************************
                                                     3 *
                                                     4 *                     CLCLE Unaligned Buffers Test
                                                     5 *
                                                     6 *        NOTE: This is a copy of the CLCL Unaligned Buffers Test
                                                     7 *              modified to test the CLCLE instruction.
                                                     8 *              James Wekel August 2022
                                                     9 *********************************************************************
                                                    10 *
                                                    11 *  This program tests proper functioning of the CLCLE instruction's
                                                    12 *  optimization logic (specifically, the "mem_cmp" function that the
                                                    13 *  CLCLE instruction makes use of) to ensure the location of the in-
                                                    14 *  equality is properly reported.
                                                    15 *
                                                    16 *  Depending on the alignment of the two operands being compared, if
                                                    17 *  the length of the compare is large enough that it would cause the
                                                    18 *  comparison to cross a page boundary for both operands and the in-
                                                    19 *  equality occurs at an offset past the distance each operand is
                                                    20 *  from its respective page boundary added together, then the address
                                                    21 *  of the inequality that CLCLEreturns would be off by the shorter
                                                    22 *  of the two distances.
                                                    23 *
                                                    24 *  For example, if the operand addresses were X'123456' and X'456789'
                                                    25 *  (and the page size was X'800') and the inequality was at (or past)
                                                    26 *  X'123877', then CLCLE would incorrectly report the address of the
                                                    27 *  inequality as being at address X'123877' - X'77' = X'123800':
                                                    28 *
                                                    29 *  X'123456' is X'3AA' bytes from the end of its page boundary.
                                                    30 *  X'456789' is X'77'  bytes from the end of its page boundary.
                                                    31 *  The true inequality is at X'123877' (X'123456' + X'77' + X'3AA').
                                                    32 *
                                                    33 *  The optimization logic would perform three separate compares: the
                                                    34 *  first starting at X'123456' for a length of X'77'. The second one
                                                    35 *  at address X'1234CD' (X'123456' + X'77') for a length of X'3AA',
                                                    36 *  and the third and final compare at address X'123877' (X'123456' +
                                                    37 *  X'77' + X'3AA') for a length of at least one byte.
                                                    38 *
                                                    39 *  Due to a bug in the original optimization logic however the length
                                                    40 *  of the first compare would not be added to the calculated offset of
                                                    41 *  where the inequality was located at.  That is to say, the offset of
                                                    42 *  the inequality would be calculated as operand-1 + X'3AA' instead of
                                                    43 *  operand-1 + X'77' + X'3AA'.  The X'77' offset would get erroneously
                                                    44 *  lost, thereby causing the location of the inequality to be reported
                                                    45 *  X'77' bytes BEFORE where the actual inequality was actually located
                                                    46 *  at! (Oops!)
                                                    47 *
                                                    48 *  The bug has since been fixed of course but to ensure such does not
                                                    49 *  occur again, this standalone runtest test performs a series of CLCL
                                                    50 *  comparisons whose parameters are such that they end up tripping the
                                                    51 *  bug as described. Thank you to Dave Kreiss for reporting the bug.
                                                    52 *
                                                    53 *********************************************************************
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                    55 ***********************************************************************
                                                    56 *
                                                    57 *                  EXAMPLE RUNTEST TEST CASE
                                                    58 *
                                                    59 *        *Testcase CLCLE-01-unaligned-buffers Test
                                                    60 *
                                                    61 *        archlvl     390
                                                    62 *        mainsize    3
                                                    63 *        numcpu      1
                                                    64 *        sysclear
                                                    65 *
                                                    66 *        loadcore    "$(testpath)/CLCLE-01-unaligned-buffers.core"
                                                    67 *
                                                    68 *        runtest     0.1
                                                    69 *
                                                    70 *        *Done
                                                    71 *
                                                    72 ***********************************************************************
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                    74          PRINT OFF
                                                  3455          PRINT ON

                                                  3457 ***********************************************************************
                                                  3458 *        SATK prolog stuff...
                                                  3459 ***********************************************************************

                                                  3461          ARCHLVL  ZARCH=NO,MNOTE=NO
                                                  3463+$AL      OPSYN AL
                                                  3464+$ALR     OPSYN ALR
                                                  3465+$B       OPSYN B
                                                  3466+$BAS     OPSYN BAS
                                                  3467+$BASR    OPSYN BASR
                                                  3468+$BC      OPSYN BC
                                                  3469+$BCTR    OPSYN BCTR
                                                  3470+$BE      OPSYN BE
                                                  3471+$BH      OPSYN BH
                                                  3472+$BL      OPSYN BL
                                                  3473+$BM      OPSYN BM
                                                  3474+$BNE     OPSYN BNE
                                                  3475+$BNH     OPSYN BNH
                                                  3476+$BNL     OPSYN BNL
                                                  3477+$BNM     OPSYN BNM
                                                  3478+$BNO     OPSYN BNO
                                                  3479+$BNP     OPSYN BNP
                                                  3480+$BNZ     OPSYN BNZ
                                                  3481+$BO      OPSYN BO
                                                  3482+$BP      OPSYN BP
                                                  3483+$BXLE    OPSYN BXLE
                                                  3484+$BZ      OPSYN BZ
                                                  3485+$CH      OPSYN CH
                                                  3486+$L       OPSYN L
                                                  3487+$LH      OPSYN LH
                                                  3488+$LM      OPSYN LM
                                                  3489+$LPSW    OPSYN LPSW
                                                  3490+$LR      OPSYN LR
                                                  3491+$LTR     OPSYN LTR
                                                  3492+$NR      OPSYN NR
                                                  3493+$SL      OPSYN SL
                                                  3494+$SLR     OPSYN SLR
                                                  3495+$SR      OPSYN SR
                                                  3496+$ST      OPSYN ST
                                                  3497+$STM     OPSYN STM
                                                  3498+$X       OPSYN X
                                                  3499+$AHI     OPSYN AHI
                                                  3500+$B       OPSYN J
                                                  3501+$BC      OPSYN BRC
                                                  3502+$BE      OPSYN JE
                                                  3503+$BH      OPSYN JH
                                                  3504+$BL      OPSYN JL
                                                  3505+$BM      OPSYN JM
                                                  3506+$BNE     OPSYN JNE
                                                  3507+$BNH     OPSYN JNH
                                                  3508+$BNL     OPSYN JNL
                                                  3509+$BNM     OPSYN JNM
                                                  3510+$BNO     OPSYN JNO
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                  3511+$BNP     OPSYN JNP
                                                  3512+$BNZ     OPSYN JNZ
                                                  3513+$BO      OPSYN JO
                                                  3514+$BP      OPSYN JP
                                                  3515+$BXLE    OPSYN JXLE
                                                  3516+$BZ      OPSYN JZ
                                                  3517+$CHI     OPSYN CHI
  LOC          OBJECT CODE        ADDR1     ADDR2     STMT

                                                      3519 *******************************************************************
                                                      3520 *        Initiate the CLCLE CSECT in the CODE region
                                                      3521 *        with the location counter at 0
                                                      3522 *******************************************************************

                                                      3524 CLCLE    ASALOAD  REGION=CODE
                                 00000000  00081031  3525+CLCLE    START 0,CODE
00000000  000A0000 00000008                           3527+         PSW   0,0,2,0,X'008'      64-bit Restart ISR Trap New PSW
00000008                         00000008  00000058  3528+         ORG   CLCLE+X'058'
00000058  000A0000 00000018                           3530+         PSW   0,0,2,0,X'018'      64-bit External ISR Trap New PSW
00000060  000A0000 00000020                           3531+         PSW   0,0,2,0,X'020'      64-bit Supervisor Call ISR Trap New PSW
00000068  000A0000 00000028                           3532+         PSW   0,0,2,0,X'028'      64-bit Program ISR Trap New PSW
00000070  000A0000 00000030                           3533+         PSW   0,0,2,0,X'030'      64-bit Machine Check Trap New PSW
00000078  000A0000 00000038                           3534+         PSW   0,0,2,0,X'038'      64-bit Input/Output Trap New PSW
00000080                         00000080  00000200  3535+         ORG   CLCLE+512

                                                      3537 *******************************************************************
                                                      3538 *        Create IPL (restart) PSW
                                                      3539 *******************************************************************

                                                      3541          ASAIPL   IA=BEGIN
                                 00000000  00081031  3542+CLCLE    CSECT
00000200                         00000200  00000000  3543+         ORG   CLCLE
00000000  00080000 00000200                           3544+         PSW   0,0,0,0,BEGIN,24
00000008                         00000008  00000200  3545+         ORG   CLCLE+512        Reset CSECT to end of assigned storage area
                                 00000000  00081031  3546+CLCLE    CSECT


                                                      3548 *******************************************************************
                                                      3549 *              The actual "CLCLE" program itself...
                                                      3550 *******************************************************************
                                                      3551 *
                                                      3552 *  Architecture Mode:   ESA/390
                                                      3553 *
                                                      3554 *  Addressing Mode:     31-bit
                                                      3555 *
                                                      3556 *  Register Usage:      R12 - R13     Base registers
                                                      3557 *                       R0  - R1      CLCLE Operand-1
                                                      3558 *                       R14 - R15     CLCLE Operand-2
                                                      3559 *                       R2  - R11     Work registers
                                                      3560 *
                                                      3561 *******************************************************************

00000200                         00000200             3563          USING  BEGIN,R12       FIRST Base Register
00000200                         00001200             3564          USING  BEGIN+4096,R13  SECOND Base Register

00000200  05C0                                        3566 BEGIN    BALR  R12,0            Initalize FIRST base register
00000202  06C0                                        3567          BCTR  R12,0            Initalize FIRST base register
00000204  06C0                                        3568          BCTR  R12,0            Initalize FIRST base register

00000206  41D0 C800                        00000800  3570          LA    R13,2048(,R12)   Initalize SECOND base register
0000020A  41D0 D800                        00000800  3571          LA    R13,2048(,R13)   Initalize SECOND base register
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2     STMT

                                                   3573 *****************************************************************
                                                   3574 *        Compare DATA1 and DATA2 one BUFFSIZE at a time...
                                                   3575 *****************************************************************

                                                   3577 *                         R4      R5    R6      R7      R8       R9
0000020E  9849 C090                      00000290  3578          LM    R4,R9,=A(BUFFER1,DATA1,BUFFER2,DATA2,BUFFSIZE,DATASIZE)

00000212  1598                                     3580          CLR   R9,R8             DATASIZE greater than BUFFSIZE?
00000214  47B0 C01A                      0000021A  3581          BNL   CHNKLOOP          Yes, get started...
00000218  1889                                     3582          LR    R8,R9             No, only compare however much we have!

                                                   3584 *             Fill buffers with next chunk of data...

0000021A  1804                                     3586 CHNKLOOP LR    R0,R4             R0 --> BUFFER1
0000021C  1825                                     3587          LR    R2,R5             R2 --> DATA1
0000021E  1818                                     3588          LR    R1,R8             R1 <== BUFFSIZE
00000220  1838                                     3589          LR    R3,R8             R3 <== BUFFSIZE
00000222  0E02                                     3590          MVCL  R0,R2             Copy into BUFFER1 <== next DATA1 chunk

00000224  1806                                     3592          LR    R0,R6             R0 --> BUFFER2
00000226  1827                                     3593          LR    R2,R7             R2 --> DATA2
00000228  1818                                     3594          LR    R1,R8             R1 <== BUFFSIZE
0000022A  1838                                     3595          LR    R3,R8             R3 <== BUFFSIZE
0000022C  0E02                                     3596          MVCL  R0,R2             Copy into BUFFER2 <== next DATA2 chunk

                                                   3598 *                   Prepare for CLCLE...

0000022E  1804                                     3600          LR    R0,R4             R0  --> BUFFER1
00000230  18E6                                     3601          LR    R14,R6            R14 --> BUFFER2
00000232  1818                                     3602          LR    R1,R8             R1  <== BUFFSIZE
00000234  18F8                                     3603          LR    R15,R8            R15 <== BUFFSIZE

                                                   3605 *                 Compare the two buffers...

                                                   3607 *                          Compare BUFFER1 with BUFFER2..
00000236  A90E 0000                      00000000  3608 CONTINUE CLCLE R0,R14,0    with padding x'00'
0000023A  4710 C036                      00000236  3609          BC    B'0001',CONTINUE      CC=3, not finished
0000023E  4780 C05C                      0000025C  3610          BE    NXTCHUNK          Equal: Buffer compare complete

                                                   3612 *           Inequality found: VERIFY ITS ACCURACY!

00000242  18A0                                     3614          LR    R10,R0            R10 --> Supposed unequal byte
00000244  D500 A000 E000    00000000  00000000  3615          CLC   0(1,R10),0(R14)  Valid inequality?
0000024A  4780 C080                      00000280  3616          BE    FAILURE           Bogus inequality!  CLCLE BUG!  FAIL!

                                                   3618 *           CLCLE was correct.  Get past inequality
                                                   3619 *           and finish comparing the buffer data if
                                                   3620 *           there is any data remaining in the buffer
                                                   3621 *           that we haven't compared yet...

0000024E  4A00 C0A8                      000002A8  3623          AH    R0,=H'1'          Get past unequal byte
00000252  4AE0 C0A8                      000002A8  3624          AH    R14,=H'1'         Get past unequal byte
00000256  0610                                     3625          BCTR  R1,0              Get past unequal byte
00000258  46F0 C036                      00000236  3626          BCT   R15,CONTINUE      Go finish buffer if any bytes remain...

                                                   3628 *          Go on to next chunk of data  -- if there is one.
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT


0000025C  1E58                                    3630 NXTCHUNK ALR   R5,R8            R5 --> Next DATA1 chunk
0000025E  1E78                                    3631          ALR   R7,R8            R7 --> Next DATA2 chunk

00000260  1B98                                    3633          SR    R9,R8            Decrement DATA bytes remaining
00000262  4780 C070                     00000270  3634          BZ    SUCCESS          None: We're done...
00000266  4720 C01A                     0000021A  3635          BP    CHNKLOOP         Lots: Go compare next chunk...
0000026A  1089                                    3636          LPR   R8,R9            Some: Make R8 <== positive remaining
0000026C  47F0 C01A                     0000021A  3637          B     CHNKLOOP         Go compare final chunk...
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                  3639 ***********************************************************************
                                                  3640 *        Normal completion or Abnormal termination PSWs
                                                  3641 ***********************************************************************


                                                  3643 SUCCESS  DWAITEND LOAD=YES          Normal completion
00000270                                          3645+SUCCESS  DS    0H
00000270  8200 C078                     00000278  3646+         LPSW  DWAT0008
00000278  000A0000 00000000                       3647+DWAT0008 PSW   0,0,2,0,X'000000'


                                                  3649 FAILURE  DWAIT LOAD=YES,CODE=BAD    Abnormal termination
00000280                                          3650+FAILURE  DS    0H
00000280  8200 C088                     00000288  3651+         LPSW  DWAT0009
00000288  000A0000 00010BAD                       3652+DWAT0009 PSW   0,0,2,0,X'010BAD'
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT

                                                  3654 ***********************************************************************
                                                  3655 *        Working Storage
                                                  3656 ***********************************************************************
                                                  3657 *
                                                  3658 *        The specific bug that was reported:
                                                  3659 *
                                                  3660 *        4DE   787FE B54   87F46 B54
                                                  3661 *        4DF   787FF B53   87F47 B53
                                                  3662 *
                                                  3663 *        F32   79252 100   8899A 100    (BOGUS!)
                                                  3664 *
                                                  3665 *        FEA   7930A 048   88A52 048
                                                  3666 *        FEB   7930B 047   88A53 047
                                                  3667 *
                                                  3668 ***********************************************************************

00000290                                          3670          LTORG ,                 Literals pool
00000290  00020320 00060000                       3671                =A(BUFFER1,DATA1,BUFFER2,DATA2,BUFFSIZE,DATASIZE)
000002A8  0001                                    3672                =H'1'

                               00002000  00000001 3674 BUFFSIZE EQU   (8*1024)
                               00001032  00000001 3675 DATASIZE EQU   X'1032'

                               00000320  00000001 3677 BUFF1OFF EQU   X'320'
                               00000A68  00000001 3678 BUFF2OFF EQU   X'A68'

000002AA                       000002AA  00020320 3680          ORG   CLCLE+(1*(128*1024))+BUFF1OFF
00020320  00000000 00000000                       3681 BUFFER1  DC    (BUFFSIZE/8)XL8'00'

00022320                       00022320  00040A68 3683          ORG   CLCLE+(2*(128*1024))+BUFF2OFF
00040A68  00000000 00000000                       3684 BUFFER2  DC    (BUFFSIZE/8)XL8'00'

00042A68                       00042A68  00060000 3686          ORG   CLCLE+(3*(128*1024))    X'60000'
00060000  00000000 00000000                       3687 DATA1    DC    (DATASIZE)X'00'        X'60000'

00061032                       00061032  00080000 3689          ORG   CLCLE+(4*(128*1024))    X'80000'
00080000  00000000 00000000                       3690 DATA2    DC    (DATASIZE)X'00'        X'80000'

00081032                       00081032  000804DE 3692          ORG   DATA2+X'04DE'
000804DE  FF                                      3693          DC    X'FF'

000804DF                       000804DF  000804DF 3695          ORG   DATA2+X'04DF'
000804DF  FF                                      3696          DC    X'FF'

000804E0                       000804E0  00080FEA 3698          ORG   DATA2+X'0FEA'
00080FEA  FF                                      3699          DC    X'FF'

00080FEB                       00080FEB  00080FEB 3701          ORG   DATA2+X'0FEB'
00080FEB  FF                                      3702          DC    X'FF'

00080FEC                       00080FEC  00081032 3704          ORG   DATA2+DATASIZE
```

```
  LOC        OBJECT CODE        ADDR1     ADDR2    STMT

                                                   3706 ***********************************************************************
                                                   3707 *        Register equates
                                                   3708 ***********************************************************************

                              00000000  00000001  3710 R0       EQU   0
                              00000001  00000001  3711 R1       EQU   1
                              00000002  00000001  3712 R2       EQU   2
                              00000003  00000001  3713 R3       EQU   3
                              00000004  00000001  3714 R4       EQU   4
                              00000005  00000001  3715 R5       EQU   5
                              00000006  00000001  3716 R6       EQU   6
                              00000007  00000001  3717 R7       EQU   7
                              00000008  00000001  3718 R8       EQU   8
                              00000009  00000001  3719 R9       EQU   9
                              0000000A  00000001  3720 R10      EQU   10
                              0000000B  00000001  3721 R11      EQU   11
                              0000000C  00000001  3722 R12      EQU   12
                              0000000D  00000001  3723 R13      EQU   13
                              0000000E  00000001  3724 R14      EQU   14
                              0000000F  00000001  3725 R15      EQU   15


                                                   3727          END
```

```
     SYMBOL        TYPE  VALUE   LENGTH  DEFN  REFERENCES

BEGIN               I    000200       2  3566  3544  3563  3564
BUFF1OFF            U    000320       1  3677  3680
BUFF2OFF            U    000A68       1  3678  3683
BUFFER1             X    020320       8  3681  3578
BUFFER2             X    040A68       8  3684  3578
BUFFSIZE            U    002000       1  3674  3681  3684  3578
CHNKLOOP            I    00021A       2  3586  3581  3635  3637
CLCLE               J    000000  528434  3525  3528  3535  3543  3545  3680  3683  3686  3689
CODE                2    000000  528434  3525
CONTINUE            I    000236       4  3608  3609  3626
DATA1               X    060000       1  3687  3578
DATA2               X    080000       1  3690  3692  3695  3698  3701  3704  3578
DATASIZE            U    001032       1  3675  3687  3690  3704  3578
DWAT0008            3    000278       8  3647  3646
DWAT0009            3    000288       8  3652  3651
FAILURE             H    000280       2  3650  3616
IMAGE               1    000000  528434     0
NXTCHUNK            I    00025C       2  3630  3610
R0                  U    000000       1  3710  3586  3590  3592  3596  3600  3608  3614  3623
R1                  U    000001       1  3711  3588  3594  3602  3625
R10                 U    00000A       1  3720  3614  3615
R11                 U    00000B       1  3721
R12                 U    00000C       1  3722  3563  3566  3567  3568  3570
R13                 U    00000D       1  3723  3564  3570  3571
R14                 U    00000E       1  3724  3601  3608  3615  3624
R15                 U    00000F       1  3725  3603  3626
R2                  U    000002       1  3712  3587  3590  3593  3596
R3                  U    000003       1  3713  3589  3595
R4                  U    000004       1  3714  3578  3586  3600
R5                  U    000005       1  3715  3587  3630
R6                  U    000006       1  3716  3592  3601
R7                  U    000007       1  3717  3593  3631
R8                  U    000008       1  3718  3580  3582  3588  3589  3594  3595  3602  3603  3630  3631  3633  3636
R9                  U    000009       1  3719  3578  3580  3582  3633  3636
SUCCESS             H    000270       2  3645  3634
=A(BUFFER1,DATA1,BUFFER2,DATA2,BUFFSIZE,DATASIZE)
                    A    000290       4  3671  3578
=H'1'               H    0002A8       2  3672  3623  3624
```

```
 MACRO    DEFN  REFERENCES

ANTR       140
APROB      272
ARCHIND    432   3462
ARCHLVL    573   3461
ASAIPL     699   3541
ASALOAD    779   3524
ASAREA     834
ASAZAREA  1019
CPUWAIT   1102
DSECTS    1428
DWAIT     1631   3644   3649
DWAITEND  1688   3643
ENADEV    1696
ESA390    1796
IOCB      1807
IOCBDS    1983
IOFMT     2017
IOINIT    2355
IOTRFR    2396
ORB       2444
POINTER   2633
PSWFMT    2661
RAWAIT    2795
RAWIO     2891
SIGCPU    3049
SMMGR     3107
SMMGRB    3207
TRAP128   3256
TRAP64    3233   3526   3529
TRAPS     3269
ZARCH     3343
ZEROH     3355
ZEROL     3383
ZEROLH    3411
ZEROLL    3434
```

```
   DESC    SYMBOL   SIZE       POS          ADDR

Entry: 0

Image      IMAGE   528434  00000-81031  00000-81031
  Region   CODE    528434  00000-81031  00000-81031
    CSECT  CLCLE   528434  00000-81031  00000-81031
   STMT                                                                 FILE NAME

1     c:\Users\Fish\Documents\Visual Studio 2008\Projects\MyProjects\ASMA-0\CLCLE-01-unaligned-buffers\CLCLE-01-unaligned-buffers.asm
2     C:\Users\Fish\Documents\Visual Studio 2008\Projects\Hercules\_Git\_Harold\SATK-0\srcasm\satk.mac


** NO ERRORS FOUND **
```