```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT
                                                        2 *
                                                        3 ********************************************************************************
                                                        4 *
                                                        5 *Testcase str-001-cksm
                                                        6 *  Test cases for variations on the CKSM (Checksum) instruction.
                                                        7 *
                                                        8 ********************************************************************************
                                                        9 *
                                                       10 *                            str-001-cksm.asm
                                                       11 *
                                                       12 * Created and placed into the public domain 2018-12-30 by Bob Polmanter
                                                       13 * Remove runtest *Compare dependency on 2022-03-08 by Fish
                                                       14 *
                                                       15 * The CKSM instruction is tested against the definition in the
                                                       16 * z/Architecture Principles of Operation, SA22-7832.
                                                       17 *
                                                       18 * Test data is assembled into this program, and some test data is
                                                       19 * generated by this program. The program itself verifies the resulting
                                                       20 * status of registers and condition codes via simple CLC comparison.
                                                       21 *
                                                       22 *
                                                       23 *
                                                       24 * Tests performed with CKSM (Checksum):
                                                       25 *
                                                       26 * 1.  Checksum; 2nd operand does not cross page boundary,
                                                       27 *                length is a multiple of 4.
                                                       28 * 2.  Checksum; 2nd operand does not cross page boundary,
                                                       29 *                length is NOT a multiple of 4.
                                                       30 * 3.  Checksum; 2nd operand fully crosses page boundary,
                                                       31 *                length is a multiple of 4.
                                                       32 * 4.  Checksum; 2nd operand fully crosses page boundary,
                                                       33 *                length is NOT a multiple of 4.
                                                       34 * 5.  Checksum; 2nd operand ends on page boundary,
                                                       35 *                length is a multiple of 4.
                                                       36 * 6.  Checksum; 2nd operand ends on page boundary,
                                                       37 *                length is NOT a multiple of 4.
                                                       38 * 7.  Checksum; 2nd operand ends on page boundary+2,
                                                       39 *                length is a multiple of 4.
                                                       40 * 8.  Checksum; 2nd operand ends on page boundary+2,
                                                       41 *                length is NOT a multiple of 4.
                                                       42 * 9.  Checksum; 2nd operand crosses multiple pages
                                                       43 *
                                                       44 * NOTE: the variation between lengths with a multiple of 4 and
                                                       45 *       not a multiple of 4 is to test the conceptual adding of
                                                       46 *       zero values to complete the checksum with 4-byte elements
                                                       47 *       as described in the Principles of Operation.
                                                       48 *
                                                       49 ********************************************************************************
                                                       50 *
                                                       51 *
00000000  0000088F                                     52 CKSM001  START 0
00000000  00000001                                     53 STRTLABL EQU   *
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

                                00000000  00000001    54 R0       EQU   0
                                00000001  00000001    55 R1       EQU   1
                                00000002  00000001    56 R2       EQU   2
                                00000003  00000001    57 R3       EQU   3
                                00000004  00000001    58 R4       EQU   4
                                00000005  00000001    59 R5       EQU   5
                                00000006  00000001    60 R6       EQU   6
                                00000007  00000001    61 R7       EQU   7
                                00000008  00000001    62 R8       EQU   8
                                00000009  00000001    63 R9       EQU   9
                                0000000A  00000001    64 R10      EQU   10
                                0000000B  00000001    65 R11      EQU   11
                                0000000C  00000001    66 R12      EQU   12
                                0000000D  00000001    67 R13      EQU   13
                                0000000E  00000001    68 R14      EQU   14
                                0000000F  00000001    69 R15      EQU   15
                                                      70 *
                                                      71 *
00000000                        00000000              72          USING *,R15
                                                      73 *
                                                      74 * Selected z/Arch low core layout
                                                      75 *
00000000                        00000000  000001A0    76          ORG   STRTLABL+X'1A0'     z/Arch Restart PSW
000001A0  00000001 80000000                           77          DC    X'0000000180000000',A(0,START)
                                                      78 *
000001B0                        000001B0  000001D0    79          ORG   STRTLABL+X'1D0'     z/Arch Program check new PSW
000001D0  00020000 00000000                           80          DC    X'0002000000000000',XL4'00',X'0000DEAD' Abnormal end
                                                      81 *
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT
                                                      83 ********************************************************************************
                                                      84 *
                                                      85 *  Main program.
                                                      86 *
000001E0                        000001E0  00000200    87          ORG   STRTLABL+X'200'
00000200                                              88 START     DS    0H
                                                      89 *
                                                      90 *
                                                      91 **********
                                                      92 * TEST 1 * No page boundary crossed; len=multiple of 4
                                                      93 **********
                                                      94 *
00000200  4120 F700                     00000700     95          LA    R2,TDATA1               -> buffer to checksum
00000204  4130 0010                     00000010     96          LA    R3,16                   Length
00000208  4D90 F318                     00000318     97          BAS   R9,CHECKSUM             compute
0000020C  9013 F800                     00000800     98          STM   R1,R3,RESULT1           Save test result regs
                                                      99 *
                                                     100 **********
                                                     101 * TEST 2 * No page boundary crossed; len=NOT multiple of 4
                                                     102 **********
                                                     103 *
00000210  4120 F700                     00000700     104         LA    R2,TDATA1               -> buffer to checksum
00000214  4130 000D                     0000000D     105         LA    R3,13                   Length
00000218  4D90 F318                     00000318     106         BAS   R9,CHECKSUM             compute
0000021C  9013 F810                     00000810     107         STM   R1,R3,RESULT2           Save test result regs
                                                     108 *
                                                     109 **********
                                                     110 * TEST 3 * Page boundary crossed; len=multiple of 4
                                                     111 **********
                                                     112 *
00000220  5820 F710                     00000710     113         L     R2,BOUND1               -> where to place the buffer
00000224  D20F 2000 F700      00000000  00000700     114         MVC   0(16,R2),TDATA1         Move data across boundary
0000022A  4130 0010                     00000010     115         LA    R3,16                   Length
0000022E  4D90 F318                     00000318     116         BAS   R9,CHECKSUM             compute
00000232  9013 F820                     00000820     117         STM   R1,R3,RESULT3           Save test result regs
                                                     118 *
                                                     119 **********
                                                     120 * TEST 4 * Page boundary crossed; len=NOT multiple of 4
                                                     121 **********
                                                     122 *
00000236  5820 F710                     00000710     123         L     R2,BOUND1               -> where to place the buffer
0000023A  D20F 2000 F700      00000000  00000700     124         MVC   0(16,R2),TDATA1         Move data across boundary
00000240  4130 000D                     0000000D     125         LA    R3,13                   Length
00000244  4D90 F318                     00000318     126         BAS   R9,CHECKSUM             compute
00000248  9013 F830                     00000830     127         STM   R1,R3,RESULT4           Save test result regs
                                                     128 *
                                                     129 **********
                                                     130 * TEST 5 * Operand ends on a page boundary; len=multiple of 4
                                                     131 **********
                                                     132 *
0000024C  5820 F714                     00000714     133         L     R2,BOUND2               -> where to place the buffer
00000250  D20F 2000 F700      00000000  00000700     134         MVC   0(16,R2),TDATA1         Place the data
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2     STMT

00000256  4130 0010                       00000010   135          LA    R3,16                  Length
0000025A  4D90 F318                       00000318   136          BAS   R9,CHECKSUM            compute
0000025E  9013 F840                       00000840   137          STM   R1,R3,RESULT5          Save test result regs
                                                     138 *
                                                     139 *
                                                     140 **********
                                                     141 * TEST 6 * Operand ends on a page boundary; len=NOT multiple of 4
                                                     142 **********
                                                     143 *
00000262  5820 F718                       00000718   144          L     R2,BOUND3              -> where to place the buffer
00000266  D20F 2000 F700      00000000    00000700   145          MVC   0(16,R2),TDATA1        Place the data
0000026C  4130 000D                       0000000D   146          LA    R3,13                  Length
00000270  4D90 F318                       00000318   147          BAS   R9,CHECKSUM            compute
00000274  9013 F850                       00000850   148          STM   R1,R3,RESULT6          Save test result regs
                                                     149 *
                                                     150 **********
                                                     151 * TEST 7 * Operand ends on a page boundary+2; len=multiple of 4
                                                     152 **********
                                                     153 *
00000278  5820 F71C                       0000071C   154          L     R2,BOUND4              -> where to place the buffer
0000027C  D20F 2000 F700      00000000    00000700   155          MVC   0(16,R2),TDATA1        Place the data
00000282  4130 0010                       00000010   156          LA    R3,16                  Length
00000286  4D90 F318                       00000318   157          BAS   R9,CHECKSUM            compute
0000028A  9013 F860                       00000860   158          STM   R1,R3,RESULT7          Save test result regs
                                                     159 *
                                                     160 *
                                                     161 **********
                                                     162 * TEST 8 * Operand ends on a page boundary+2; len=NOT multiple of 4
                                                     163 **********
                                                     164 *
0000028E  5820 F720                       00000720   165          L     R2,BOUND5              -> where to place the buffer
00000292  D20F 2000 F700      00000000    00000700   166          MVC   0(16,R2),TDATA1        Place the data
00000298  4130 000D                       0000000D   167          LA    R3,13                  Length
0000029C  4D90 F318                       00000318   168          BAS   R9,CHECKSUM            compute
000002A0  9013 F870                       00000870   169          STM   R1,R3,RESULT8          Save test result regs
                                                     170 *
                                                     171 **********
                                                     172 * TEST 9 * Operand crosses multiple pages
                                                     173 **********
                                                     174 *
000002A4  9825 F724                       00000724   175          LM    R2,R5,AREA             Load multi-page area ptrs
000002A8  0E24                                       176          MVCL  R2,R4                  Pad the buffer area
                                                     177 *
000002AA  5820 F724                       00000724   178          L     R2,AREA                -> multipage buffer
000002AE  5830 F734                       00000734   179          L     R3,TEST9LEN            Length to checksum
000002B2  4D90 F318                       00000318   180          BAS   R9,CHECKSUM            compute
000002B6  9013 F880                       00000880   181          STM   R1,R3,RESULT9          Save test result regs
                                                     182 *
                                                     183 **       Verify correct results...
                                                     184 *
000002BA  D50B F368 F800      00000368    00000800   185          CLC   GRESULT1,RESULT1
000002C0  4770 F330                       00000330   186          BNE   BAD99
```

```
  LOC         OBJECT CODE       ADDR1     ADDR2    STMT

000002C4  D50B F374 F810      00000374  00000810   187          CLC   GRESULT2,RESULT2
000002CA  4770 F330                     00000330   188          BNE   BAD99
000002CE  D50B F380 F820      00000380  00000820   189          CLC   GRESULT3,RESULT3
000002D4  4770 F330                     00000330   190          BNE   BAD99
000002D8  D50B F38C F830      0000038C  00000830   191          CLC   GRESULT4,RESULT4
000002DE  4770 F330                     00000330   192          BNE   BAD99
000002E2  D50B F398 F840      00000398  00000840   193          CLC   GRESULT5,RESULT5
000002E8  4770 F330                     00000330   194          BNE   BAD99
000002EC  D50B F3A4 F850      000003A4  00000850   195          CLC   GRESULT6,RESULT6
000002F2  4770 F330                     00000330   196          BNE   BAD99
000002F6  D50B F3B0 F860      000003B0  00000860   197          CLC   GRESULT7,RESULT7
000002FC  4770 F330                     00000330   198          BNE   BAD99
00000300  D50B F3BC F870      000003BC  00000870   199          CLC   GRESULT8,RESULT8
00000306  4770 F330                     00000330   200          BNE   BAD99
0000030A  D50B F3C8 F880      000003C8  00000880   201          CLC   GRESULT9,RESULT9
00000310  4770 F330                     00000330   202          BNE   BAD99
                                                   203 *
00000314  B2B2 F338                     00000338   204          LPSWE GOODPSW                 load SUCCESS disabled wait PSW
                                                   205 *
                                                   206 *-- CKSM routine used by tests
                                                   207 *
                          00000318  00000001   208 CHECKSUM EQU   *
00000318  1B11                                     209          SR    R1,R1                   Init checksum accum
                                                   210 *
                          0000031A  00000001   211 INVOKE   EQU   *
0000031A  B241 0012                                212          CKSM  R1,R2                   Compute checksum
0000031E  4740 F32C                0000032C   213          BC    4,BADCC                 CC=1 SHOULD NEVER HAPPEN
00000322  4720 F32C                0000032C   214          BC    2,BADCC                 CC=2 SHOULD NEVER HAPPEN
00000326  4710 F31A                0000031A   215          BC    1,INVOKE                Restart the checksum
0000032A  07F9                                 216          BR    R9                      Return if CC=0
                                               217 *
0000032C  B2B2 F348                00000348   218 BADCC    LPSWE BADCCPSW                Stop on invalid CKSUM CC
00000330  B2B2 F358                00000358   219 BAD99    LPSWE BAD99PSW                Stop on invalid CKSUM result
                                               220 *
00000338                                       221          DS    0D              Ensure correct alignment for psw
00000338  00020000 00000000                    222 GOODPSW  DC    X'0002000000000000',A(0,0) Normal end - disabled wait
00000348  00020000 00000000                    223 BADCCPSW DC    X'0002000000000000',XL4'00',X'000BADCC' Abnormal end
00000358  00020000 00000000                    224 BAD99PSW DC    X'0002000000000000',XL4'00',X'00099BAD' Abnormal end
                                               225 *
                                               226 *
00000368  99DE2265 00000710                    227 GRESULT1 DC    XL12'99DE22650000071000000000'
00000374  99003366 0000070D                    228 GRESULT2 DC    XL12'990033660000070D00000000'
00000380  99DE2265 0000300B                    229 GRESULT3 DC    XL12'99DE22650000300B00000000'
0000038C  99003366 00003008                    230 GRESULT4 DC    XL12'9900336600003008000000000'
00000398  99DE2265 00003000                    231 GRESULT5 DC    XL12'99DE22650000300000000000'
000003A4  99003366 00003000                    232 GRESULT6 DC    XL12'9900336600003000000000000'
000003B0  99DE2265 00003002                    233 GRESULT7 DC    XL12'99DE22650000300200000000'
000003BC  99003366 00003002                    234 GRESULT8 DC    XL12'9900336600003002000000000'
000003C8  E1E1E1E1 0000BFF8                    235 GRESULT9 DC    XL12'E1E1E1E10000BFF800000000'
                                               236 *
                                               237 *
                                               238 *
```

```
  LOC        OBJECT CODE       ADDR1     ADDR2    STMT
000003D4                     000003D4  00000700   239          ORG   STRTLABL+X'700'
                                                  240 *
00000700  00112233                                241 TDATA1   DC    X'00112233'            Buffer data to be checksummed
00000704  44556677                                242          DC    X'44556677'
00000708  8899AABB                                243          DC    X'8899AABB'
0000070C  CCDDEEFF                                244          DC    X'CCDDEEFF'
                                                  245 *
00000710  00002FFB                                246 BOUND1   DC    X'00002FFB'            -> data crosses boundary
00000714  00002FF0                                247 BOUND2   DC    X'00002FF0'            -> data ends at boundary
00000718  00002FF3                                248 BOUND3   DC    X'00002FF3'            -> data ends at boundary
0000071C  00002FF2                                249 BOUND4   DC    X'00002FF2'            -> data ends at boundary+2
00000720  00002FF5                                250 BOUND5   DC    X'00002FF5'            -> data ends at boundary+2
                                                  251 *
00000724  00004000                                252 AREA     DC    X'00004000'            multi=page area
00000728  00010000                                253 AREALEN  DC    A(4096*16)             Size of multi=page area
0000072C  00000000                                254 ZERO     DC    A(0)
00000730  87000000                                255 PAD      DC    X'87000000'            MVCL pad char
00000734  00007FF8                                256 TEST9LEN DC    F'32760'               Length to checksum test 9
                                                  257 *
                                                  258 *
                                                  259 *
                                                  260 *  Locations for results
                                                  261 *
                                                  262 * Result fields are kept on 16-byte boundaries to more easily
                                                  263 * track their assembled offsets for use in the .tst script.
                                                  264 *
                                                  265 *                                 offset
00000738                     00000738  00000800   266          ORG   STRTLABL+X'800'   8xx
00000800  00000000 00000000                       267 RESULT1  DS    4F                  00   Register results test 1
00000810  00000000 00000000                       268 RESULT2  DS    4F                  10   Register results test 2
00000820  00000000 00000000                       269 RESULT3  DS    4F                  20   Register results test 3
00000830  00000000 00000000                       270 RESULT4  DS    4F                  30   Register results test 4
00000840  00000000 00000000                       271 RESULT5  DS    4F                  40   Register results test 5
00000850  00000000 00000000                       272 RESULT6  DS    4F                  50   Register results test 6
00000860  00000000 00000000                       273 RESULT7  DS    4F                  60   Register results test 7
00000870  00000000 00000000                       274 RESULT8  DS    4F                  70   Register results test 8
00000880  00000000 00000000                       275 RESULT9  DS    4F                  80   Register results test 9
                                                  276 *
                                                  277          END
```

SYMBOL	TYPE	VALUE	LENGTH	DEFN	REFERENCES
AREA	X	000724	4	252	175 178
AREALEN	A	000728	4	253	
BAD99	I	000330	4	219	186 188 190 192 194 196 198 200 202
BAD99PSW	X	000358	8	224	219
BADCC	I	00032C	4	218	213 214
BADCCPSW	X	000348	8	223	218
BOUND1	X	000710	4	246	113 123
BOUND2	X	000714	4	247	133
BOUND3	X	000718	4	248	144
BOUND4	X	00071C	4	249	154
BOUND5	X	000720	4	250	165
CHECKSUM	U	000318	1	208	97 106 116 126 136 147 157 168 180
CKSM001	J	000000	2192	52	
GOODPSW	X	000338	8	222	204
GRESULT1	X	000368	12	227	185
GRESULT2	X	000374	12	228	187
GRESULT3	X	000380	12	229	189
GRESULT4	X	00038C	12	230	191
GRESULT5	X	000398	12	231	193
GRESULT6	X	0003A4	12	232	195
GRESULT7	X	0003B0	12	233	197
GRESULT8	X	0003BC	12	234	199
GRESULT9	X	0003C8	12	235	201
IMAGE	1	000000	2192	0	
INVOKE	U	00031A	1	211	215
PAD	X	000730	4	255	
R0	U	000000	1	54	
R1	U	000001	1	55	98 107 117 127 137 148 158 169 181 209 212
R10	U	00000A	1	64	
R11	U	00000B	1	65	
R12	U	00000C	1	66	
R13	U	00000D	1	67	
R14	U	00000E	1	68	
R15	U	00000F	1	69	72
R2	U	000002	1	56	95 104 113 114 123 124 133 134 144 145 154 155 165 166 175 176 178 212
R3	U	000003	1	57	96 98 105 107 115 117 125 127 135 137 146 148 156 158 167 169 179 181
R4	U	000004	1	58	176
R5	U	000005	1	59	175
R6	U	000006	1	60	
R7	U	000007	1	61	
R8	U	000008	1	62	
R9	U	000009	1	63	97 106 116 126 136 147 157 168 180 216
RESULT1	F	000800	4	267	98 185
RESULT2	F	000810	4	268	107 187
RESULT3	F	000820	4	269	117 189
RESULT4	F	000830	4	270	127 191
RESULT5	F	000840	4	271	137 193
RESULT6	F	000850	4	272	148 195
RESULT7	F	000860	4	273	158 197
RESULT8	F	000870	4	274	169 199

```
     SYMBOL         TYPE  VALUE    LENGTH  DEFN  REFERENCES

RESULT9              F    000880       4   275  181  201
START                H    000200       2    88   77
STRTLABL             U    000000       1    53   76   79   87  239  266
TDATA1               X    000700       4   241   95  104  114  124  134  145  155  166
TEST9LEN             F    000734       4   256  179
ZERO                 A    00072C       4   254
```

MACRO DEFN REFERENCES

No defined macros

```
   DESC      SYMBOL   SIZE    POS      ADDR

Entry: 0

Image       IMAGE    2192  000-88F  000-88F
  Region             2192  000-88F  000-88F
    CSECT   CKSM001  2192  000-88F  000-88F
```

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

** NO ERRORS FOUND **
```