```
 0001 MUR11 DOMUS MACRO ASSEMBLER REV 02.00
01                ;                                       RCSL: 43-GL 10266
02                ;                                       AUTHOR: DHA/KDH/JHA
03                ;                                       EDITED: 80.05.13
04
05
06
07
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23                ;                   MUR11
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42                ; KEYWORD: MUS,RECORD I/O, MOVE, PROCEDURES,
43                ;           LISTING.
44                ; ABSTRACT: MUS RECORD I/O PROCEDURES.
45
46                ; ASCII PAPER TAPE RCSL 43-GL10265
47
48                ; REL. BINARY PAPER TAPE RCSL 43-GL10267
49
50
51
```

```
10002 MUR11
01                  ;    ****** RECORD I/O PROCEDURES ******
02
                    .TITL MUR11
04                  .XREL
05          000001  .TXTM 1
06          000012  .RDX 10
07
08
09
```

```
10003 MUR11
01              ; PROCEDURE GETREC(ZONE,BYTES,ADDR);
02              ; THE CALL PARAMETER BYTES IS ONLY RELEVANT FOR UNFORMATTED BLOCKED
03              ; RECORDS. GETREC GETS THE NEXT RECORD FROM THE ZONE ACCORDING
04              ; TO THE FOLLOWING ALGORITHM:
05              ;
06              ; LOCAL PROCEDURE GETBLOCK;
07              ;    REPEAT INBLOCK(ZONE) UNTIL REM.ZONE<>0;
08              ;
09              ; LOCAL PROCEDURE GETHEAD;
10              ;    BEGIN
11              ;    LENGTH.ZONE:=4; IF REM<4 THEN GOTO BLOCKERROR;
12              ;    BYTES:= BYTE(TOP,ZONE) CON BYTE(TOP.ZONE+1)-4;
13              ;    LENGTH.ZONE:= BYTES;
14              ;    UPDATE(4);
15              ;    IF BYTES>REM THEN GOTO BLOCKERROR;
16              ;    END;
17              ;
18              ; LOCAL PROCEDURE UPDATE;
19              ;    BEGIN
20              ;    FIRST.ZONE:=ADDR:=TOP.ZONE;
21              ;    TOP.ZONE:=TOP.ZONE+COUNT;
22              ;    REM.ZONE:=REM.ZONE-COUNT;
23              ;    END;
24              ;
25              ; GETREC:
26              ;   CASE FORMAT.ZONE OF
27              ;    BEGIN
28              ; U:   GETBLOCK; BYTES:=REM.ZONE;
29              ;
30              ; UB:  WHILE BYTES>REM.ZONE DO GETBLOCK;
31              ;
32              ; F:   GETBLOCK; BYTES:=LENGTH.ZONE;
33              ;      IF BYTES<>REM.ZONE THEN GOTO BLOCKERROR;
34              ;
35              ; FB:  BYTES:=LENGTH.ZONE;
36              ;      IF BYTES>REM.ZONE THEN
37              ;        IF REM.ZONE<>0 THEN GOTO BLOCKERROR
38              ;      ELSE BEGIN GETBLOCK; GOTO FB END;
39              ;
40              ; V:   GETBLOCK; GETHEAD;
41              ;      IF BYTES>REM.ZONE THEN GOTO BLOCKERROR;
42              ;      IF BYTES=REM.ZONE THEN GOTO VB;
43              ;      IF ZONE.KIND(11)<>1 THEN GOTO BLOCKERROR
44              ;                          ELSE REM.ZONE:=BYTES;
45              ;      GOTO VB
46              ;
47              ; VB:  IF REM.ZONE=0 THEN GOTO V;
48              ;      GETHEAD;
49              ;    END !CASE!;
50              ;
51              ;  LENGTH.ZONE:=BYTES;
52              ;  UPDATE(BYTES);
53              ; RETURN;
54              ;         CALL:              RETURN:
55              ; AC0     BYTES              BYTES
56              ; AC1                        ADDR
57              ; AC2     ZONE               ZONE
58              ; AC3     LINK
59
```

```
10004 MUR11
01 00000*055031 A50:  STA    3   Z5,2        ; GETREC:
02 00001*004450       JSR        A503        ;  CASE FORMAT.ZONE OF
03 00002*000010*      A5009                  ;  U-ADDR;
04 00003*000014*      A5010+1                ;  UB-ADDR;
05 00004*000020*      A5011                  ;  F-ADDR;
06 00005*000023*      A5011+3                ;  FB-ADDR;
07 00006*000027*      A5012                  ;  V-ADDR;
08 00007*000043*      A5014                  ;  VB-ADDR;
09 00010*004456 A5009:JSR        A505        ; U:  GETBLOCK(ZONE,REM,BYTES);
10 00011*045016       STA    1   ZLENGTH,2   ;     LENGTH.ZONE:=REM;
11 00012*000434       JMP        A502        ;
12 00013*004453 A5010:JSR        A505        ; R1: GETBLOCK(ZONE,REM,BYTES);
13 00014*041016       STA    0   ZLENGTH,2   ; UB: LENGTH.ZONE:=BYTES;
14 00015*122032       ADCZ#  1,0 SZC         ;     IF BYTES>REM THEN
15 00016*000775       JMP         A5010      ;      GOTO R1;
16 00017*000427       JMP        A502        ;
17 00020*125004 A5011:MOV    1,1 SZR         ; F:  IF REM.ZONE<>0 THEN
18 00021*000501       JMP        A508        ;      GOTO ERROR;
19 00022*004444       JSR        A505        ;     GETBLOCK(ZONE,REM,BYTES);
20 00023*021016       LDA    0   ZLENGTH,2   ; FB: BYTES:=LENGTH.ZONE;
21 00024*122032       ADCZ#  1,0 SZC         ;     IF BYTES>REM THEN
22 00025*000773       JMP        A5011       ;      GOTO F;
23 00026*000420       JMP        A502        ;
24 00027*004437 A5012:JSR        A505        ; V:  GETBLOCK(ZONE,REM,BYTES);
25 00030*004445       JSR        A506        ;     GETHEAD(ZONE,REM,BYTES);
26 00031*106433       SUBZ#  0,1 SNC         ;     IF BYTES>REM THEN
27 00032*000470       JMP        A508        ;      GOTO ERROR;
28 00033*106415       SUB#   0,1 SNR         ;     IF BYTES=REM THEN
29 00034*000407       JMP        A5014       ;       GOTO VB;
30 00035*024114       LDA    1   .1B11       ;
31 00036*035005       LDA    3   ZKIND,2     ;
32 00037*137405       AND    1,3 SNR         ;     IF ZONE.KIND(11)<>1 THEN
33 00040*000462       JMP        A508        ;       GOTO ERROR         ELSE
34 00041*105000       MOV    0,1             ;       BEGIN
35 00042*045023       STA    1   ZREM,2      ;         REM:=BYTES; REM.ZONE:=REM;
36                    A5014:                 ;       END;
37 00043*125005       MOV    1,1 SNR         ; VB: IF REM=0 THEN
38 00044*000763       JMP        A5012       ;      GOTO V;
39 00045*004430       JSR        A506        ;     GETHEAD(ZONE,REM,BYTES);
40                    A502:                  ;  ESAC;
41 00046*021016       LDA    0   ZLENGTH,2   ;  BYTES:=LENGTH.ZONE;
42                    A5021:                 ; UPDATE:
43 00047*004406       JSR        A504        ;  UPDATE(BYTES);
44 00050*007031       JSR@       Z5,2        ; RETURN;
45
46 00051*025015 A503: LDA    1   ZFORMAT,2   ;  GET RECORDFORMAT;
47 00052*137000       ADD    1,3             ;
48 00053*025023       LDA    1   ZREM,2      ;  REM:=REM.ZONE;
49 00054*003400       JMP@       +0,3        ;
50
51
52
```

```
!0005 MUR11
01
02
03                ; PROCEDURE UPDATE(BYTES);
04                ;       CALL:           RETURN:
05                ; AC0   BYTES           BYTES
06                ; AC1                   FIRST.ZONE
07                ; AC2   ZONE            ZONE
08                ; AC3   LINK
09                ;
10  00055*025023  A504: LDA     1   ZREM,2      ; UPDATE:
11  00056*106400        SUB     0,1            ;
12  00057*045023        STA     1   ZREM,2      ;  REM.ZONE:=REM.ZONE-BYTES;
13  00060*025020        LDA     1   ZTOP,2      ;
14  00061*107000        ADD     0,1            ;
15  00062*045020        STA     1   ZTOP,2      ;  TOP.ZONE:=TOP.ZONE+BYTES;
16  00063*106400        SUB     0,1            ;
17  00064*045017        STA     1   ZFIRST,2    ;   FIRST.ZONE:=TOP.ZONE-BYTES;
18  00065*001400        JMP         0,3         ; RETURN;
19
20
```

```
10006 MUR11
01                ; PROCEDURE GETBLOCK(ZONE,REM,BYTES);
02                ;
03                ;       CALL:           RETURN:
04                ; AC0                   BYTES
05                ; AC1                   REM
06                ; AC2  ZONE             ZONE
07                ; AC3  LINK
08                ;
09 00066*055030 A505: STA     3   Z4,2        ; GETBLOCK:
10 00067*006205       INBLOCK                 ; REPEAT
11 00070*025023       LDA     1   ZREM,2      ;   INBLOCK(ZONE);
12 00071*125005       MOV     1,1 SNR         ;   REM:=REM.ZONE
13 00072*000775       JMP         .-3         ;  UNTIL REM<>0;
14 00073*021016       LDA     0   ZLENGTH,2   ;  BYTES:=LENGTH.ZONE;
15 00074*003030       JMP@        Z4,2        ; RETURN;
16
17                ; PROCEDURE GETHEAD(ZONE,REM,BYTES);
18                ;       CALL:           RETURN:
19                ; AC0                   BYTES
20                ; AC1  REM              REM
21                ; AC2  ZONE             ZONE
22                ; AC3  LINK
23                ;
24 00075*055027 A506: STA     3   Z3,2        ; GETHEAD:
25 00076*020116       LDA     0  .4           ;
26 00077*041016       STA     0   ZLENGTH,2   ;
27 00100*122032       ADCZ#   1,0 SZC         ;  IF REM<4 THEN
28 00101*000421       JMP         A508        ;   GOTO ERROR;
29 00102*004753       JSR         A504        ;  UPDATE(4);
30 00103*135220       MOVZR   1,3             ;  ADDRESS:=FIRST/2;
31 00104*021400       LDA     0   0,3         ;  BYTES:=WORD(ADDRESS);
32 00105*025401       LDA     1   1,3         ;  TAIL:=WORD(ADDRESS+1);
33 00106*125303       MOVS    1,1 SNC         ;  IF ODD(FIRST) THEN
34 00107*000405       JMP         A5064       ;   BEGIN
35 00110*034143       LDA     3  .255         ;
36 00111*167400       AND     3,1             ;   B2:=TAIL(0:7);
37 00112*163700       ANDS    3,0             ;   B1:=BYTES(8:15) SHIFT 8;
38 00113*123000       ADD     1,0             ;   BYTES:= B1+B2;
39 00114*024116 A5064:LDA     1  .4           ;   END;
40 00115*122400       SUB     1,0             ;  BYTES:=BYTES-4;
41 00116*025023       LDA     1   ZREM,2      ;  REM:=REM.ZONE;
42 00117*041016       STA     0   ZLENGTH,2   ;  LENGTH.ZONE:=BYTES;
43 00120*122033       ADCZ#   1,0 SNC         ;   IF BYTES<=REM THEN
44 00121*003027       JMP@        Z3,2        ;   RETURN;
45
46 00122*024111 A508: LDA     1   SBLOCK      ; BLOCK ERROR:
47 00123*125400       INC     1,1             ;  STATUS:=BLOCKERROR+SOFT;
48 00124*045024       STA     1   Z0,2        ;  Z0.ZONE:=STATUS;
49                ; ENTRY TO GIVEUP WITH STATUS=1B8+1B15, LENGTH.ZONE=ATTEMPTED
50                ; RECSIZE, AND REM.ZONE=AVAILABLE BYTES. IN CASE OF NORMAL RETURN
51                ; RECSIZE IS SET TO REM.ZONE.
52 00125*035007       LDA     3   ZGIVEUP,2   ;
53 00126*175004       MOV     3,3 SZR         ;  IF GIVEUP.ZONE<>0 THEN
54 00127*005400       JSR         0,3         ;   GIVEUP.ZONE;
55 00130*021023       LDA     0   ZREM,2      ;  BYTES:=REM.ZONE;
56 00131*000716       JMP         A5021       ;  GOTO UPDATE;
57
58
59
```

```
!0007 MUR11
01              ; PROCEDURE PUTREC(ZONE,BYTES,ADDR);
02              ; MAKES ROOM FOR A RECORD IN THE ZONE ACCORDING
03              ; TO THE FOLLOWING DESCRIPTION:
04              ;
05              ; LOCAL PROCEDURE UPDATE(COUNT);
06              ;   SEE DESCRIPTION IN GETREC;
07              ;
08              ; LOCAL PROCEDURE CHANGEBLOCK;
09              ;   BEGIN
10              ;   OUTBLOCK(ZONE);
11              ;   IF REM.ZONE<BYTES THEN
12              ;     GOTO BLOCKERROR;
13              ;   END;
14              ;
15              ;   CASE FORMAT.ZONE OF
16              ;
17              ; U:  CHANGEBLOCK;
18              ;
19              ; UB: IF REM.ZONE<BYTES THEN CHANGEBLOCK;
20              ;
21              ; F:  BYTES:=LENGTH.ZONE;
22              ;     CHANGEBLOCK;
23              ;
24              ; FB: BYTES:=LENGTH.ZONE;
25              ;     IF REM.ZONE<BYTES THEN CHANGEBLOCK;
26              ;
27              ; V:  BYTES:=BYTES+8;
28              ;     CHANGEBLOCK;
29              ;     UPDATE(4);
30              ;     BYTES:=BYTES-8;
31              ;     GOTO VB;
32              ;
33              ; VB: IF TOP.ZONE=FIRST.SHARE THEN GOTO V;
34              ;     IF REM.ZONE<BYTES+4 THEN GOTO V;
35              ;     UPDATE(4);
36              ;     FIRST.ZONE(0:1):=BYTES+4;
37              ;     FIRST.ZONE(2:3):=0;
38              ;     LENGTH:=TOP.ZONE-FIRST.SHARE+BYTES;
39              ;     HEAD:=FIRST.USED.ZONE;
40              ;     HEAD(0:1):=LENGTH;
41              ;     HEAD(2:3):=0;
42              ;
43              ;   END !CASE! ;
44
45              ;   LENGTH.ZONE:=BYTES;
46              ;   UPDATE(BYTES);
47              ;
48              ;         CALL:              RETURN:
49              ; AC0     BYTES              BYTES
50              ; AC1                        ADDR
51              ; AC2     ZONE               ZONE
52              ; AC3     LINK               DESTROYED
53
54
55
```

```
10008 MUR11
01                A51:                                ; PUTREC:
02  00132*055031          STA     3    Z5,2          ;
03  00133*025015          LDA     1    ZFORMAT,2     ;   FORMAT:=FORMAT.ZONE;
04  00134*034117          LDA     3   .2              ;
05  00135*137414          AND#    1,3  SZR           ;   IF FORMAT IS F THEN
06  00136*021016          LDA     0    ZLENGTH,2     ;     BYTES:=LENGTH.ZONE;
07  00137*175120          MOVZL   3,3                 ;
08  00140*137404          AND     1,3  SZR           ;    IF FORMAT>=4 THEN
09  00141*163000          ADD     3,0                ;     BYTES:=BYTES+4
10  00142*055030          STA     3    Z4,2          ;   COUNT:=FORMAT AND 4;
11  00143*041016          STA     0    ZLENGTH,2     ;   LENGTH.ZONE:=BYTES;
12  00144*035023          LDA     3    ZREM,2        ;   REM:=REM.ZONE;
13  00145*162033          ADCZ#   3,0  SNC           ;   IF REM<BYTES OR
14  00146*125233          MOVZR#  1,1  SNC           ;      EVEN(FORMAT) THEN
15  00147*000456          JMP           A514         ;    CHANGEBLOCK;
16                A512:
17  00150*035021          LDA     3    ZUSED,2       ;
18  00151*035406          LDA     3    SFIRST,3      ;
19  00152*025020          LDA     1    ZTOP,2        ;
20  00153*021030          LDA     0    Z4,2          ;
21  00154*166405          SUB     3,1  SNR           ;   IF ZTOP=SFIRST THEN
22  00155*004700          JSR           A504         ;   UPDATE(COUNT);
23  00156*021016          LDA     0    ZLENGTH,2     ;
24  00157*004676          JSR           A504         ;    UPDATE(BYTES,FIRST);
25  00160*035030          LDA     3    Z4,2          ;
26  00161*175005          MOV     3,3  SNR           ;   IF COUNT=0 THEN
27  00162*003031          JMP@          Z5,2         ;    RETURN;
28  00163*135220          MOVZR   1,3                ;   ADDR:=FIRST//2;
29  00164*126463          SUBC    1,1  SNC           ;   IF FIRST(15) THEN
30  00165*000414          JMP           A5135        ;    BEGIN
31  00166*045402          STA     1    2,3           ;    WORD(ADDR+2):=0;
32  00167*024147          LDA     1   .M256          ;
33  00170*107400          AND     0,1                ;    B1:=BYTES(0:7) SHIFT 8;
34  00171*122700          SUBS    1,0                ;    B2:=BYTES(8:15) SHIFT 8;
35  00172*041401          STA     0    1,3           ;    WORD(ADDR+1):=B2;
36  00173*021400          LDA     0    0,3           ;
37  00174*034147          LDA     3   .M256          ;
38  00175*163700          ANDS    3,0                ;    B0:=WORD(ADDR)(0:7);
39  00176*035017          LDA     3    ZFIRST,2      ;
40  00177*175220          MOVZR   3,3                ;
41  00200*123301          ADDS    1,0  SKP           ;    BYTES:=B0 SHIFT 8+(B1 SHIFT(-8));
42
```

```
!0009 MUR11
01 00201*045401 A5135:STA     1    1,3         ;    END ELSE WORD(ADDR+1):=0;
02 00202*041400       STA     0    0,3         ;   WORD(ADDR):=BYTES;
03 00203*021016       LDA     0    ZLENGTH,2   ;
04 00204*024116       LDA     1   .4           ;
05 00205*122400       SUB     1,0              ;
06 00206*041016       STA     0    ZLENGTH,2   ;   LENGTH.ZONE:=LENGTH.ZONE-4;
07 00207*021017       LDA     0    ZFIRST,2    ;
08 00210*123000       ADD     1,0              ;   FIRST.ZONE:=FIRST.ZONE+4;
09 00211*041017       STA     0    ZFIRST,2    ;
10 00212*035021       LDA     3    ZUSED,2     ;   SH:=USED.ZONE;
11 00213*025406       LDA     1    SFIRST,3    ;   FIRST:=FIRST.SH;
12 00214*135220       MOVZR   1,3              ;   ADDR:=FIRST//2;
13 00215*021020       LDA     0    ZTOP,2      ;
14 00216*122400       SUB     1,0              ;   LENGTH:=TOP.ZONE-FIRST.SHARE;
15 00217*041400       STA     0    0,3         ;   WORD(ADDR):=LENGTH;
16 00220*102400       SUB     0,0              ;
17 00221*041401       STA     0    1,3         ;   WORD(ADDR+1):=0;
18 00222*025017       LDA     1    ZFIRST,2    ;
19 00223*021016       LDA     0    ZLENGTH,2   ;
20 00224*003031       JMP@          Z5,2       ;
21
22
23 00225*006206 A514: OUTBLOCK                 ; CHANGEBLOCK:
24 00226*035023       LDA     3    ZREM,2      ;
25 00227*021016       LDA     0    ZLENGTH,2   ;
26 00230*162033       ADCZ#   3,0  SNC         ;   IF REM.ZONE<LENGTH.ZONE THEN
27 00231*000717       JMP           A512       ;
28 00232*000670       JMP           A508       ;   GOTO BLOCK ERROR;
29
30
31
```

```
10010 MUR11
01                ; PROCEDURE MOVE(PARAMADDR);
02                ; MOVES A NUMBER OF BYTES FROM ONE LOCATION TO ANOTHER.
03                ; THE LOCATION-ADDR.S MUST BE BYTE ADDR.S.THE PARAMADDR IS A WORDAD
04                ;     CALL:                   RETURN:
05                ; AC0                         DESTROYED
06                ; AC1                         DESTROYED
07                ; AC2 PARAMADDR               PARAMADDR
08                ; AC3 LINK                    DESTROYED
09                ;
10                ; PARAMADDR: COUNT
11                ;         +1: TO BYTEADDRESS
12                ;         +2: FROM BYTEADDRESS
13                ;         +3: SAVE RETURN (WORDADDR);
14                                              ; MOVE:
15 00233*055003 A83:  STA     3  +3,2           ;
16 00234*034040       LDA     3   CUR           ; SAVE RETURN;
17 00235*051424       STA     2   SAVE,3        ; TEMP SAVE(PARAMADDR);
18 00236*035000 A831: LDA     3   +0,2          ; START:
19 00237*174513       NEGL#   3,3 SNC           ;   IF COUNT<=0 THEN
20 00240*003003       JMP@        +3,2          ;    RETURN;
21 00241*025001       LDA     1   +1,2          ;
22 00242*125223       MOVZR   1,1 SNC           ;
23 00243*175225       MOVZR   3,3 SNR           ;   IF  TO (15) OR
24 00244*000402       JMP         .+2           ;      COUNT = 1 THEN
25 00245*000413       JMP         A832          ;    BEGIN
26 00246*025002       LDA     1   +2,2          ;
27 00247*006174       GETBYTE                   ;    GETBYTE(FROMADDR,BYTE);
28 00250*031024       LDA     2   SAVE,2        ;
29 00251*025001       LDA     1   +1,2          ; LASTBYTE:
30 00252*006175 A8315:PUTBYTE                   ;    PUTBYTE(TOADDR,BYTE);
31 00253*031024       LDA     2   SAVE,2        ;
32 00254*015000       DSZ         +0,2          ;    DECR(COUNT);
33 00255*011001       ISZ         +1,2          ;    INCR(TO);
34 00256*011002       ISZ         +2,2          ;    INCR(FROM);
35 00257*000757       JMP         A831          ;    GOTO START;
36                                              ;    END;
37                A832:                         ;
38 00260*160400       NEG     3,0               ;   ODD:=CARRY := COUNT EXTRACT 1;
39 00261*041000       STA     0  +0,2           ;   SAVE COUNT := -COUNT//2;
40 00262*045001       STA     1  +1,2           ;   SAVE TO := TO//2;
41 00263*165100       MOVL    3,1               ;   AC1(15):= ODD;
42 00264*035002       LDA     3   +2,2          ;   FROM:= FROM ADDR//2;
43 00265*175223       MOVZR   3,3 SNC           ;   IF FROM ADDR(15)=0 THEN MATCH
44 00266*000432       JMP         A834          ;     ELSE PREPARE NO-MATCH;
45 00267*055002       STA     3   +2,2          ; PREPARE NO-MATCH:
46 00270*034040       LDA     3   CUR           ;   SAVE FROM;
47 00271*045424       STA     1   SAVE,3        ;   TEMP SAVE ODD;
48 00272*034143       LDA     3  .255           ;   (NOTICE: PARAMADDR IN AC2);
49 00273*027002       LDA@    1   +2,2          ;   WORD:= 0,FROM;
50 00274*167400       AND     3,1               ;   LEFT:= WORD(8:15);
51
52
53
```

```
!0011 MUR11
01                A833:                          ; NO MATCH:
02  00275*011002          ISZ         +2,2       ;   INCR(FROM);
03  00276*023002          LDA*   0    +2,2       ;   WORD:= 0.FROM;
04  00277*107000          ADD    0,1             ;   BOTH:= WORD+LEFT;
05  00300*163400          AND    3,0             ;   BYTE:= WORD(8:15);
06  00301*106700          SUBS   0,1             ;   WORD:= BOTH-BYTE;
07  00302*047001          STA*   1    +1,2       ;   0.TO:= WORD;
08  00303*011001          ISZ         +1,2       ;   INCR(TO);
09  00304*105000          MOV    0,1             ;   LEFT:= BYTE;
10  00305*011000          ISZ         +0,2       ;   IF INCR(COUNT)<>0 THEN
11  00306*000767          JMP         A833       ;     GOTO NO MATCH;
12  00307*034040          LDA    3    CUR        ;
13  00310*025424          LDA    1    SAVE,3     ;
14  00311*051424          STA    2    SAVE,3     ;   TEMP SAVE(PARAMADDR);
15  00312*135000          MOV    1,3             ;   AC3(15):= ODD;
16  00313*025001          LDA    1    +1,2       ;   TO := TO*2;
17  00314*125120          MOVZL  1,1             ; CHECKLAST:
18  00315*175223 A8335:   MOVZR  3,3 SNC        ;   IF -,CARRY THEN
19  00316*003003          JMP*       +3,2       ;    RETURN
20  00317*000733          JMP          A8315     ;   ELSE GOTO LASTBYTE;
21  00320*125220 A834:    MOVZR  1,1             ; PREPARE MATCH:
22  00321*031001          LDA    2  +1,2         ;   ***** NOTICE!!!!!!: CARRY := ODD;
23  00322*025400 A835:    LDA    1  +0,3         ; MATCH:
24  00323*045000          STA    1  +0,2         ;   WORD.TO:= WORD.FROM;
25  00324*175400          INC    3,3             ;   INCR(FROM);
26  00325*151400          INC    2,2             ;   INCR(TO);
27  00326*101404          INC    0,0 SZR         ;   IF INCR(COUNT)<>0 THEN
28  00327*000773          JMP          A835      ;     GOTO MATCH;
29  00330*021400          LDA    0    0,3        ;
30  00331*024147          LDA    1   .M256       ;   BYTE:=FROM(0:7);
31  00332*123700          ANDS   1,0             ;
32  00333*175160          MOVCL  3,3             ;   AC3(15):= ODD;
33  00334*145120          MOVZL  2,1             ;   TO := TO*2;
34  00335*030040          LDA    2    CUR        ;
35  00336*031024          LDA    2    SAVE,2     ;   RESTORE PARAMADDR;
36  00337*000756          JMP          A8335     ;   GOTO  CHECKLAST;
37
38                ;   ******   END OF RECORD I/O PROCEDURES   ******
39
40          000200 .LOC GETREC-GOS
41  00200   000000*       A50
42          000201 .LOC PUTREC-GOS
43  00201   000132*       A51
44          000224 .LOC MOVE-GOS
45  00224   000233*       A83
46
47
48
49
50                .END
0000 SOURCE LINES IN ERROR
```

0012 MUR11

A50	000000*	4/01	11/41				
A5009	000010*	4/03	4/09				
A5010	000013*	4/04	4/12	4/15			
A5011	000020*	4/05	4/06	4/17	4/22		
A5012	000027*	4/07	4/24	4/38			
A5014	000043*	4/08	4/29	4/36			
A502	000046*	4/11	4/16	4/23	4/40		
A5021	000047*	4/42	6/56				
A503	000051*	4/02	4/46				
A504	000055*	4/43	5/10	6/29	8/22	8/24	
A505	000066*	4/09	4/12	4/19	4/24	6/09	
A506	000075*	4/25	4/39	6/24			
A5064	000114*	6/34	6/39				
A508	000122*	4/18	4/27	4/33	6/28	6/46	9/28
A51	000132*	8/01	11/43				
A512	000150*	8/16	9/27				
A5135	000201*	8/30	9/01				
A514	000225*	8/15	9/23				
A83	000233*	10/15	11/45				
A831	000236*	10/18	10/35				
A8315	000252*	10/30	11/20				
A832	000260*	10/25	10/37				
A833	000275*	11/01	11/11				
A8335	000315*	11/18	11/36				
A834	000320*	10/44	11/21				
A835	000322*	11/23	11/28				

9/28

A51	000132*	8/01	11/43
A512	000150*	8/16	9/27
A5135	000201*	8/30	9/01
A514	000225*	8/15	9/23
A83	000233*	10/15	11/45
A831	000236*	10/18	10/35
A8315	000252*	10/30	11/20
A832	000260*	10/25	10/37
A833	000275*		