```
9500 IF n$=t$ THEN  LET n$=r$
9505 LET Altpointer=Altpointer-1
9510 FOR p=1 TO LEN (n$)
9520 POKE Altpointer,CODE (n$(p))
9530 LET Altpointer=Altpointer+1
9540 NEXT p
9550 IF n$<>r$ THEN  GO TO 9160
9560 LET LengthLow=PEEK (Lengthaddr)+LEN
 (r$)-LEN (t$)
9570 IF LengthLow>255 THEN  LET Lengthlo
w=LengthLow-256: POKE Lengthaddr+1,1+PEE
K (Lengthaddr+1)
9580 POKE Lengthaddr,LengthLow
9590 GO TO 9160
9599 REM Prepare to move altered program
 back to main program area
9600 LET Oldlength=Textpointer-(PEEK (PR
OG)+256*PEEK (PROG+1))
9610 LET Newlength=Altpointer-Altprog
9620 POKE 45060,Newlength-256*INT (Newle
ngth/256)
9630 POKE 45061,INT (Newlength/256)
9660 POKE 45056,Textpointer-256*INT (Tex
tpointer/256)
9670 POKE 45057,INT (Textpointer/256)
9680 IF Oldlength=Newlength THEN  RANDOM
IZE USR (45084)
9690 IF Oldlength<Newlength THEN  LET X=
Newlength-Oldlength
9700 IF Oldlength>Newlength THEN  LET X=
Textpointer-(Oldlength-Newlength)
9710 POKE 45058,X-256*INT (X/256)
9720 POKE 45059,INT (X/256)
9730 IF Oldlength<Newlength THEN  RANDOM
IZE USR 45062
9740 IF Oldlength>Newlength THEN  RANDOM
IZE USR 45074
9800 FOR p=Textpointer TO Textpointer+q-
1
9810 POKE Altpointer,PEEK (p)
9820 LET Altpointer=Altpointer+1
9830 NEXT p
9840 LET Textpointer=p
9850 RETURN
9900 SAVE "REPLACE" LINE 9910: SAVE "REP
MC"CODE 45064,37: STOP
9910 CLEAR 45055: LOAD "REPMC"CODE
```

Machine Code Loader

```
 10 CLEAR 45055
 20 LET a=45062
 30 FOR l=1000 TO 1040 STEP 10
 40 LET s=0
 50 FOR a=a TO a+7
 60 READ b
 70 POKE a,b
 80 LET s=s+b
 90 NEXT a
100 READ c
110 IF s<>c THEN  PRINT "DATA ERROR IN
LINE ";l: STOP
120 NEXT l
1000 DATA 42,0,176,237,75,2,176,205,913
1010 DATA 85,22,24,10,42,0,176,237,596
1020 DATA 91,2,176,205,229,25,33,176,937
1030 DATA 179,237,91,83,92,237,75,4,998
1040 DATA 176,237,176,207,255,0,0,0,1051
```

Replace Assembly Program

```
0000              MKROOM  EQU     $1655
0000              RCLAM1  EQU     $19E5
0000              T0      EQU     $B000
0000              T1      EQU     $B002
0000              T2      EQU     $B004
0000              ALTPRG  EQU     $B3B0
0000              PROG    EQU     $5C53
B006                      ORG     $B006
B006  2A00B0      UP      LD      HL, (T0)
B009  ED4B02B0            LD      BC, (T1)
B00D  CD5516              CALL    MKROOM
B010  180A                JR      COPY
B012  2A00B0      DOWN    LD      HL, (T0)
B015  ED5B02B0            LD      DE, (T1)
B019  CDE519              CALL    RCLAM1
B01C  21B0B3      COPY    LD      HL, ALTPRG
B01F  ED5B535C            LD      DE, (PROG)
B023  ED4B04B0            LD      BC, (T2)
B027  EDB0                LDIR
B029  CF                  RST     8
B02A  FF                  DB      $FF
```

the number of bytes specified by q, and updates the pointers to the old and new programs.

Variable names are copied by the code starting at line 9500, and, if the variable name has been changed, the two bytes at the beginning of the line that hold the length of the line are altered to reflect the change in length.

After the whole program has been altered and copied to the new area, the BASIC program calculates the values needed by the machine code for copying back the altered program, and then POKEs these values into the memory locations where the machine code expects to find them.

If the new and old programs are the same length, the new program can be copied into the same space that is occupied by the old program. In this case, the only information needed by the machine code program is the program's length.

If the new program is longer than the old program, we have to make extra space in the program area by moving up the variable replace routine, which we want to keep. The extra space is made by calling the ROM subroutine, MAKE-ROOM, at address 1655 hex. When MAKE-ROOM is called, the HL register pair must contain the address after the place where space is to be made, and the BC register pair must hold the length of the space needed. The value required for HL is just the final value of the variable Textpointer, and the value for BC is the difference between the old and the new lengths.

If the new program is shorter than the old, we have to move the variable replace program down. We can do this by using the ROM subroutine RECLAIM-1 at address 19E5 hex. When RECLAIM-1 is called, the HL register pair must hold the address of the first byte to be left alone, and the DE register pair must contain the address of the first byte to be reclaimed. The value required for HL is again the final value of the variable Textpointer, and the value required for DE is calculated by subtracting the difference between the old and new lengths from Textpointer.

The altered program is copied back to the main program area by the block move instruction LDIR (LoaD with Increment and Repeat). The start address of the altered program area is loaded into HL, the start address of the main program area into DE, the length of the altered program into BC, and then the LDIR instruction moves the whole of the altered program, byte to byte.

The last two lines in the Assembly language program use another ROM routine, at address 8. This is the 'report' routine that prints an error message and other comments. The routine is called by the RST 8 instruction, and the report produced is specified by the byte following the RST 8 instruction. The value of the byte is one less than the report number, so FF hex, or -1, gives the OK or Program finished report; 0 gives NEXT without FOR, and so on. The machine code program ends with RST8, instead of the usual RET instruction, to avoid returning to the BASIC program that has been moved.