CompUser

INTERNATIONAL COMPUTING

ISSN: 0922-4289

Vol. 4, Number 2, April 1991.

COMPUSER is a magazine issued by:
COMPUSER International Computing;
Exchanging computer knowledge.
ISSN: 0922-4203.
Founded at Krimpen a.d. IJssel,
The Netherlands, by:
Willem L. van Pelt
Ir. Coen J. Boltjes.
COMPUSER International Computing
Jacob Jordaensstraat 15
NL 2923 CK Krimpen a.d. IJssel
The Netherlands.
Postgiro: 841433 W.L. van Pelt
Phone:

 +31 1807 19881.

Regular participants:
Fred A. Behringer (W-Germany)
Hans Ebert (W-Germany)
Andrew Gregory (England)
Marc Lachaert (Belgium)
Iddy Oort (Holland)
Leif Rasmussen (Denmark)
Ronald van Vugt (Holland)
Drawings:
Leo de Kok (Holland)
Translations:
Cor Bergshoeff (Denmark)
Piet K. de Vries (Holland)
Ron van Herk and others (Holland)

The articles to be published have to be written by the sender. All in english language, unless one needs help by one of COMPUSER's translators.
To participate in COMPUSER International Computing the yearly contribution is :
Holland :
f 50,00 (eurocheque or postgiro)
Inside Europe :
DM 50,00 (enclosed in letter)
DM or *f* 50,00 (eurocheque)
DM or *f* 70,00 (bankcheque)
DM or *f* 70,00 postgiro 841433 W.L. van Pelt, Krimpen a.d.IJssel.
Outside Europe: ask for prices.

CONTENTS VOLUME 4, NUMBER 2, APR 1991.

MAIN ARTICLES

MISCELLANEOUS

RELOCATION PROGRAM FOR 6502 uP BASED COMPUTERS

By : Cor Bergshoeff, Denmark.
System: MC-65 computer

Have you ever been in a situation where you would like to relocate a program to another part of memory, especially a program of which you only had a HEX-dump?
It is not so bad if it is only a short program, but what about programs of, for instance, 4 Kbyte or more? It is a tedious job to check every byte.
This program makes it easier to relocate a program from one part of memory to another part of memory. It does not do the whole job, but most of it and I will explain what it does and not does for you.

When the program is started up, it asks for the START- and END address of where you want to have the program relocated to; the END address is not asked for.
You are then asked to ENTER the START and END address of each table which might be situated in the program and whether it should be recalculated (VECTOR addresses) or not (i.e. TEXT). Each table should be entered separately and in the right sequence in which they occur in the program. If the last table does not coincidence with the END of the program, you must ENTER the END address as the START of a table and END+1 as the END address of this table and answer the question which is answered by pressing the <ENTER> or <RETURN> key. After a few seconds, your program is relocated and has also recalculated the HI-bytes of all 3-bytes instructions. That is, all HI-bytes of the address which are situated within the program. All the HI-bytes of the addresses which are situated within the program. All the HI-bytes of addresses outside the program to be relocated are, of course, to be left unchanged. The VECTOR addresses in a VECTOR table are also changed, but VECTORS pointing to addresses outside the program are left unchanged.

As I said, something is not done by the program and have to be done manual after the program is relocated at the new address. The problem is that, in some programs, the ACCU is loaded with the HI-byte and the Y- or X-register with the LO-byte of an address (or the other way around) after which a JMP or JSR is performed where these HI- and LO-bytes are used. The program can not detect whether a LDA- or LDX- or LDY-byte is an address or a value, so these instructions have to be checked manually. You can of course try and start the program at it's new location and see if it works properly before checking it for these instructions.

I have used only a few (AIM-65) MC-65-MONITOR routines and these can easily be replaced by your own routines.
(Red: The author suggested in his letter of 4.2.'91 that someone having another microprocessor in his computer like 8086/88 or 68000 or 6809 might be challenged to develop this kind of routine for his own computer and publish it in COMPUSER.)

```
PASS 1

PASS 2


0000          ;Relocate program for MC65-Computer
0000          ;
0000          ;AIM-65 Monitor addresses
0000          ;
0000 FROM             =$E7A3          ;Print 'FROM'
0000 TO               =$E7A7          ;PRINT 'TO'
0000 BLANK2           =$E83B          ;Outputs two spaces
0000 REDOUT           =$E973          ;Reads char.input from KB.
0000 OUTPUT           =$E97A          ;Outputs char.to display
0000 COMIN            =$E1A1          ;Back to monitor
0000 CRLOW            =$EA13          ;Outputs one CR and one LF
```

```
0000          ;
0000          ;Monitor RAM-Addresses
0000          ;
0000 ADDR             =$A41C          ;Address input from KB is stored her
0000 CKSUM            =$A41E          ;Check for correct input
0000          ;
0000          ;Zero page addresses
0000          ;
0000                  *=$0
0000 FRADDR           *=*+2           ;Start- and  End-address of program
0002 TOADDR           *=*+2           ;to be relocated.
0004 TIADDR           *=*+2           ;Start address where prgrm. should be
0006 TABFRA           *=*+2           ;Start- and End address of table.
0008 TABTO            *=*+2
000A SAVADD           *=*+2           ;HI-bytes of START- and END of progr
000C HIBYTE           *=*+1           ;HI-byte of NEW address.
000D SAVEY            *=*+1
000E SAVEX            *=*+1
000F COUNT            *=*+2           ;Keeps trace of how far we have come
0011 TEMP             *=*+1           ;For temporary storage of HI-byte
0012 SAVY             *=*+1
0013 TABTYP           *=*+1           ;Used for saving type of table
0014          ;
0014                  *=$20
0020          ;
0020 VECADD           *=*+64          ;Pointers to table addressses
0060 START            =$3000
0060          ;
0060          ;Main program
0060          ;
0060                  *=START
3000          A000    LDY £0
3002          840E    STY SAVEX
3004          840D    STY SAVEY
3006          8412    STY SAVY
3008          A9AA    LDA £$AA
300A LP3      992000  STA VECADD,Y    ;Fill this area with 'AA'
300D          C8      INY
300E          C040    CPY £$40
3010          D0F8    BNE LP3
3012          202730  JSR M1          ;Print start mess.
3015          2013EA  JSR CRLOW
3018          203730  JSR M2          ;Print new start addr.
301B          2013EA  JSR CRLOW
301E          206130  JSR M3          ;Print table from/to.
3021          2013EA  JSR CRLOW
3024 NXTAB    4C7830  JMP M4          ;Get tables
3027          ;
3027 M1       A000    LDY £0          ;Print messages
```

```
3029 MS     B91831 LDA MESS,Y
302C        C903   CMP £3          ;Last char. in message ?
302E        F006   BEQ CR
3030        207AE9 JSR OUTPUT      ;Display char.
3033        C8     INY
3034        D0F3   BNE MS
3036 CR     60     RTS
3037        ;
3037 M2     20A3E7 JSR FROM
303A        B0FB   BCS M2          ;Wrong input
303C        AD1CA4 LDA ADDR        ;Transfer address LO byte
303F        8500   STA FRADDR      ;from ADDR to FRADDR
3041        850F   STA COUNT       ;and COUNT
3043        AD1DA4 LDA ADDR+1      ;Do the same with HI byte
3046        8501   STA FRADDR+1
3048        8510   STA COUNT+1
304A        850A   STA SAVADD
304C        203BE8 JSR BLANK2
304F T1     20A7E7 JSR TO
3052        B0FB   BCS T1
3054        AD1CA4 LDA ADDR
3057        8502   STA TOADDR
3059        AD1DA4 LDA ADDR+1
305C        8503   STA TOADDR+1
305E        850B   STA SAVADD+1
3060        60     RTS
3061        ;
3061 M3     A03B   LDY £$3B
3063        202930 JSR MS
3066 T2     20A7E7 JSR TO
3069        B0FB   BCS T2
306B        AD1CA4 LDA ADDR
306E        8504   STA TIADDR
3070        AD1DA4 LDA ADDR+1
3073        8505   STA TIADDR+1
3075        850C   STA HIBYTE
3077        60     RTS
3078        ;
3078 M4     2013EA JSR CRLOW
307B        A05A   LDY £$5A
307D        202930 JSR MS
3080        2013EA JSR CRLOW
3083 F1     20A3E7 JSR FROM
3086        B0FB   BCS F1
3088        AD1EA4 LDA CKSUM       ;Is input an address or 'ENTER' ?
308B        D003   BNE BLK         ;Not address, go and output 2 Spaces
308D        200731 JSR STOADD      ;Save address.
3090 BLK    203BE8 JSR BLANK2
3093 T3     20A7E7 JSR TO
3096        B0FB   BCS T3
3098        AD1EA4 LDA CKSUM       ;Is input an address or 'ENTER' ?
```

```
309B          D033    BNE NOTAB       ;Input is 'ENTER'.
309D          200731  JSR STOADD
30A0          2013EA  JSR CRLOW
30A3          A0B2    LDY £$B2
30A5          202930  JSR MS
30A8          2013EA  JSR CRLOW
30AB          2073E9  JSR REDOUT      ;Get char. Y/N or y/n.
30AE          C94E    CMP £$4E
30B0          F011    BEQ NO
30B2          C96E    CMP £$6E
30B4          F00D    BEQ NO
30B6          A40E    LDY SAVEX
30B8          A9FF    LDA £$FF        ;Mark table to be
30BA          992000  STA VECADD,Y    ;recalculated.
30BD          C8      INY
30BE          840E    STY SAVEX
30C0          4C7830  JMP M4
30C3 NO       A900    LDA £0          ;Mark table not to be
30C5          A40E    LDY SAVEX       ;recalculated.
30C7          992000  STA VECADD,Y
30CA          C8      INY
30CB          840E    STY SAVEX
30CD          4C7830  JMP M4
30D0          ;
30D0 NOTAB    A50E    LDA SAVEX       ;Last KB input was 'ENTER'..
30D2          C900    CMP £0          ;See if there are any tables.
30D4          F01E    BEQ NTAB        ;No tables
30D6          A412    LDY SAVY        ;Start of first table
30D8          A200    LDX £0
30DA LP       B92000  LDA VECADD,Y    ;Get start- and End address
30DD          9506    STA TABFRA,X    ;and store it in TABFRA
30DF          C8      INY
30E0          E8      INX
30E1          C60E    DEC SAVEX       ;Decrm. pointer to VECADD
30E3          E004    CPX £4          ;Have we both addresses ?
30E5          D0F3    BNE LP          ;2 LO + 2 HI bytes
30E7          B92000  LDA VECADD,Y    ;Get mark '00' or 'FF' of
30EA          8513    STA TABTYP      ;table and save it
30EC          C60E    DEC SAVEX
30EE          C8      INY
30EF          8412    STY SAVY        ;Here we came to on VECADD
30F1          4C6132  JMP M5
30F4          ;
30F4 NTAB     A500    LDA FRADDR      ;No tables or recalculations
30F6          850F    STA COUNT       ;of program, only relocation
30F8          A501    LDA FRADDR+1    ;of program.
30FA          8510    STA COUNT+1
30FC          A502    LDA TOADDR
30FE          8508    STA TABTO
3100          A503    LDA TOADDR+1
3102          8509    STA TABTO+1
3104          4CE232  JMP STTAB       ;Go and relocate program
```

```
3107          ;
3107 STOADD A60E    LDX SAVEX       ;Save addresses of tables
3109        AD1CA4  LDA ADDR
310C        9520    STA VECADD,X
310E        E8      INX
310F        AD1DA4  LDA ADDR+1
3112        9520    STA VECADD,X
3114        E8      INX
3115        860E    STX SAVEX
3117        60      RTS             ;Back for more tables
3118        ;
3118 MESS
3118        2045    .BYT ' Enter start- and end addresses',$0D,$0A
3137        0D
3138        0A
3139        2020    .BYT '           of programm.',$0D,$0A,$03
3150        0D
3151        0A
3152        03
3153        2045    .BYT ' Enter the new start address',$0D,$0A,$03
316F        0D
3170        0A
3171        03
3172        2045    .BYT ' Enter start- and end adresses',$0D,$0A
3190        0D
3191        0A
3192        2020    .BYT '      of first/next table',$0D,$0A
31AB        0D
31AC        0A
31AD        2050    .BYT ' Press "ENTER" if no tables' ;$0D,$0A,$03
31C8        0A
31C9        03
31CA        2052    .BYT ' Recalculate table ? (Y/N)',$0D,$0A,$03
31E4        0D
31E5        0A
31E6        03
31E7        ;
31E7        ;All 3-bytes instr. of 6502 UP.
31E7        ;
31E7 MNEM3  6D      .BYT $6D,$7D,$79,$2D,$3D,$39,$0E,$1E,$2C,$CD
31E8        7D
31E9        79
31EA        2D
31EB        3D
31EC        39
31ED        0E
31EE        1E
31EF        2C
31F0        CD
31F1        DD      .BYT $DD,$D9,$EC,$CC,$CE,$DE,$4D,$5D,$59,$EE
31F2        D9
31F3        EC
```

```
31F4        CC
31F5        CE
31F6        DE
31F7        4D
31F8        5D
31F9        59
31FA        EE
31FB        FE      .BYT $FE,$4C,$6C,$20,$AD,$BD,$B9,$AE,$BE,$AC
31FC        4C
31FD        6C
31FE        20
31FF        AD
3200        BD
3201        B9
3202        AE
3203        BE
3204        AC
3205        BC      .BYT $BC,$4E,$5E,$0D,$1D,$19,$2E,$3E,$6E,$7E
3206        4E
3207        5E
3208        0D
3209        1D
320A        19
320B        2E
320C        3E
320D        6E
320E        7E
320F        ED      .BYT $ED,$FD,$F9,$8D,$9D,$99,$8E,$8C
3210        FD
3211        F9
3212        8D
3213        9D
3214        99
3215        8E
3216        8C
3217        ;
3217        ;All 2-bytes instr. of 6502 UP.
3217        ;
3217 MNEM2  69      .BYT $69,$65,$61,$71,$75,$29,$25,$21,$31,$35
3218        65
3219        61
321A        71
321B        75
321C        29
321D        25
321E        21
321F        31
3220        35
3221        06      .BYT $06,$16,$90,$B0,$F0,$24,$30,$D0,$10,$50
3222        16
3223        90
3224        B0
3225        F0
3226        24
```

```
3227    30
3228    D0
3229    10
322A    50
322B    70      .BYT $70,$C9,$C5,$C1,$D1,$D5,$E0,$E4,$C0,$C4
322C    C9
322D    C5
322E    C1
322F    D1
3230    D5
3231    E0
3232    E4
3233    C0
3234    C4
3235    C6      .BYT $C6,$D6,$49,$45,$41,$51,$55,$E6,$F6,$A9
3236    D6
3237    49
3238    45
3239    41
323A    51
323B    55
323C    E6
323D    F6
323E    A9
323F    A5      .BYT $A5,$A1,$B1,$B5,$A2,$A6,$B6,$A0,$A4,$B4
3240    A1
3241    B1
3242    B5
3243    A2
3244    A6
3245    B6
3246    A0
3247    A4
3248    B4
3249    46      .BYT $46,$56,$09,$05,$01,$11,$15,$26,$36,$66
324A    56
324B    09
324C    05
324D    01
324E    11
324F    15
3250    26
3251    36
3252    66
3253    76      .BYT $76,$E9,$E5,$E1,$F1,$F5,$85,$81,$91,$95
3254    E9
3255    E5
3256    E1
3257    F1
3258    F5
3259    85
325A    81
325B    91
```

```
325C         95
325D         86      .BYT $86,$96,$84,$94
325E         96
325F         84
3260         94
3261         ;
3261 M5      A506    LDA TABFRA      ;If table start addr. and current
3263         C500    CMP FRADDR      ;addr. are different then we must
3265         D009    BNE RECPRG      ;recalculte that part of program
3267         A507    LDA TABFRA+1    ;which contains 3, 2 and 1 bytes
3269         C501    CMP FRADDR+1    ;instr. until addr. are equal
326B         D003    BNE RECPRG      ;If equal then jump and
326D         4CE232  JMP STTAB       ;relocate table.
3270         ;
3270 RECPRG  A000    LDY £0
3272         B100    LDA (FRADDR),Y  ;Get the first instruction
3274         A030    LDY £$30        ;Number of 3-byte instr.+1
3276 LP1     88      DEY
3277         C0FF    CPY £$FF        ;Had all instr.? If so, it
3279         F007    BEQ MNM2        ;must be a 2-byte instr.
327B         D9E731  CMP MNEM3,Y     ;Find instr. in table
327E         D0F6    BNE LP1
3280         F01B    BEQ RET3        ;and go and process it.
3282         ;
3282 MNM2    A04A    LDY £$4A        ;Number of 2-bytes instr.+1
3284 LP4     88      DEY
3285         C0FF    CPY £$FF        ;Had all instr. ? If so, it
3287         F007    BEQ MNM1        ;must be a 1-byte instr.
3289         D91732  CMP MNEM2,Y
328C         D0F6    BNE LP4
328E         F03E    BEQ RET2        ;Go and process it.
3290         ;
3290 MNM1    A000    LDY £0
3292         9104    STA (TIADDR),Y  ;Store 1-byte instr.
3294         200133  JSR CTRL        ;Update COUNT, FRADDR and TIADDR
3297         202A33  JSR CTRL1       ;Check if TABFRA and COUNT are equal
329A         4C7032  JMP RECPRG      ;Not yet, get next instr.
329D RET3    A000    LDY £0
329F         9104    STA (TIADDR),Y  ;Save instr.
32A1         200133  JSR CTRL
32A4         B100    LDA (FRADDR),Y  ;Get next (LO) byte of ABS. address
32A6         9104    STA (TIADDR),Y  ;Save it.(Need not to be altered).
32A8         200133  JSR CTRL
32AB         B100    LDA (FRADDR),Y  ;Get next (HI) byte of ABS.adress
32AD         AA      TAX             ;Save in in X-reg.
32AE         29F0    AND £$F0        ;Mask-off LO-byte
32B0         C9A0    CMP £$A0        ;$Axxx is I/O in MC65 and AIM65 and
32B2         F00E    BEQ OK          ;must therefore not be altered.!!
32B4         203B33  JSR TEST        ;Not Axxx so go and process it.
32B7         9104    STA (TIADDR),Y  ;Save HI-byte after processing
32B9         200133  JSR CTRL
32BC         202A33  JSR CTRL1
32BF         4C7032  JMP RECPRG      ;Get next instr.
```

```
32C2 OK     8A      TXA              ;Get byte from X-reg.
32C3        9104    STA (TIADDR),Y   ;and save it.
32C5        200133  JSR CTRL
32C8        202A33  JSR CTRL1
32CB        4C7032  JMP RECPRG
32CE        ;
32CE RET2   A000    LDY £0           ;Two-bytes instr. need not be
32D0        9104    STA (TIADDR),Y   ;tested and are relocated as
32D2        200133  JSR CTRL         ;they are but COUNT etc. must
32D5        B100    LDA (FRADDR),Y   ;be updated for each byte and a
32D7        9104    STA (TIADDR),Y   ;check carried out to see if we
32D9        200133  JSR CTRL         ;have come to the start of table.
32DC        202A33  JSR CTRL1        ;or crossed a page.
32DF        4C7032  JMP RECPRG
32E2        ;
32E2 STTAB  A513    LDA TABTYP       ;Is table type '00' or 'FF'
32E4        C9FF    CMP ££FF         ;If 'FF'then CALCULATE new
32E6        F073    BEQ CALC         ;HI-byte
32E8        A000    LDY £0           ;Table type is '00'
32EA NXT    B100    LDA (FRADDR),Y   ;Start with first byte and
32EC        9104    STA (TIADDR),Y   ;relocate it to new addr.
32EE        200133  JSR CTRL
32F1        A508    LDA TABTO        ;Have we come to end of
32F3        C50F    CMP COUNT        ;table ?
32F5        D0F3    BNE NXT          ;Not yet, continue.
32F7        B100    LDA (FRADDR),Y   ;Yes, get last byte and
32F9        9104    STA (TIADDR),Y   ;relocate it.
32FB        200133  JSR CTRL
32FE        4CD030  JMP NOTAB        ;Go see if more tables.
3301        ;
3301 CTRL   E600    INC FRADDR
3303        E604    INC TIADDR
3305        E60F    INC COUNT
3307        A900    LDA £0           ;Have we passed a page ?
3309        C50F    CMP COUNT
330B        D017    BNE RT4          ;If not RETURN
330D        A510    LDA COUNT+1      ;Have we finished relocating
330F        C503    CMP TOADDR+1     ;program ?
3311        F012    BEQ DONE         ;Yes, return to monitor.
3313        18      CLC              ;No, update HI-bytes
3314        6901    ADC £1           ;One page only
3316        8510    STA COUNT+1
3318        A501    LDA FRADDR+1
331A        6901    ADC £1
331C        8501    STA FRADDR+1
331E        A505    LDA TIADDR+1
3320        6901    ADC £1
3322        8505    STA TIADDR+1
3324 RT4    60      RTS
3325        ;
3325 DONE   68      PLA              ;Have finished, clear stack
3326        68      PLA              ;for subroutine addr.
```

```
3327        4CA1E1 JMP COMIN       ;Go back to monitor.
332A        ;
332A CTRL1  A506   LDA TABFRA      ;Get start of table LO-byte
332C        C50F   CMP COUNT       ;and compare it with COUNT LO
332E        D0F4   BNE RT4         ;Not equal, continue
3330        A507   LDA TABFRA+1    ;Get HI-byte of table and
3332        C510   CMP COUNT+1     ;compare it with COUNT HI.
3334        D0EE   BNE RT4         ;Not equal, continue
3336        68     PLA             ;have finished, clear stack
3337        68     PLA
3338        4CE232 JMP STTAB       ;go and relocate table
333B        ;
333B TEST   8611   STX TEMP        ;Save X in Temp. for later use
333D        A50A   LDA SAVADD      ;get HI-byte of START
333F        38     SEC
3340        E511   SBC TEMP        ;Subtract HI-byte of addr.of instr.
3342        F002   BEQ TST         ;Branch if Z-flag is 1
3344        B013   BCS NOADD       ;Branch if C-flag is 1
3346 TST    A50B   LDA SAVADD+1    ;Get HI-byte of END
3348        38     SEC
3349        E511   SBC TEMP        ;Subtract HI-byte of addr. of instr.

334B        B004   BCS ADD         ;Branch if C-flag is 1
334D        F002   BEQ ADD         ;Branch if Z-flag is 1
334F        9008   BCC NOADD       ;Branch if C-flag is 0
3351 ADD    8A     TXA             ;Get org. addr. from X-reg.
3352        38     SEC
3353        E50A   SBC SAVADD      ;Subtract HI-byte of START
3355        18     CLC
3356        650C   ADC HIBYTE      ;Add HI-byte of the NEW addr.
3358        60     RTS             ;Return to prgr. save NEW HI-byte
3359 NOADD  8A     TXA             ;No addition, transfer byte to ACCU
335A RT5    60     RTS             ;Return and save byte.
335B        ;
335B CALC   A000   LDY £0
335D NXT1   B100   LDA (FRADDR),Y  ;Get first (LO) byte of address
335F        9104   STA (TIADDR),Y  ;and relocate it.
3361        200133 JSR CTRL
3364        B100   LDA (FRADDR),Y  ;Get second (HI)byte of address
3366        AA     TAX             ;and test it like we did before
3367        29F0   AND £$F0
3369        C9A0   CMP £$A0
336B        F00E   BEQ OK1
336D        203B33 JSR TEST
3370        9104   STA (TIADDR),Y
3372        208733 JSR CTRL2
3375        200133 JSR CTRL
3378        4C5D33 JMP NXT1
337B OK1    8A     TXA
337C        9104   STA (TIADDR),Y
337E        208733 JSR CTRL2
3381        200133 JSR CTRL
```

```
3384          4C5D33 JMP NXT1
3387          ;
3387 CTRL2    A508   LDA TABTO       ;Get LO-byte of table
3389          C50F   CMP COUNT       ;and compare with COUNT
338B          D0CD   BNE RT5         ;not equal, return
338D          A509   LDA TABTO+1     ;Was equal, now check HI-byte
338F          C510   CMP COUNT+1     ;with HI-byte of COUNT
3391          D0C7   BNE RT5         ;not equal, return
3393          200133 JSR CTRL        ;Was equal, update COUNT etc.
3396          68     PLA             ;Clear stack
3397          68     PLA
3398          4CD030 JMP NOTAB       ;See if more tables.
339B                 .END
 ERRORS= 0000
```

APPLE NEWS

Apple Petitions FCC for Use of Radio Waves For Data Transmission by All Computer Makers

Apple Computer, Inc. filed a petition with the Federal Communications Commission (FCC) that, if approved, would let computers transmit and receive information over radio waves instead of through a wired network. The petition asks the FCC to allocate a part of the radio spectrum so that all computer manufacturers be permitted use of radio waves for wireless computing. Apple believes that approval of the petition is an important step in the establishment of the next generation of personal computing.

Apple's petition paves the way for the establishment of a new class of data communications, called Data Personal Communications Services (Data-PCS). If Apple's petition is approved, personal computer users in the future will be able to communicate with other users and with computer peripherals within a building or a campus over radio waves. This innovation would eliminate the need, in many cases, for local communications to travel on wired networks.

"With the rapid advances in portable computing and wireless communications, we believe it is essential that computer users have access to this vital communications resource in the future", said John Sculley, Apple's chairman and chief executive officer. "Wireless networks will change the nature of information tools, making them as mobile and spontaneous as the individuals using them. Apple's action. which will benefit all personal computer users, is motivated by a desire to ensure that the United States will have made the most forward-looking public decisions, allowing wireless networking to become a reality," Sculley added.

Specifically, Apple petitioned the FCC to allow computer communications exclusively on 40 MHz of the radio frequency bandwidth between 1850-1990 MHz to transmit data at high speeds (for example, 10 megabits per second) over short distances (up to about 150 feet).

"The convergence of wireless communications and computers will dramatically change the nature of computing," said David Nagel, vice president of Apple's Advanced Technology Group. "For example, students and teachers would no longer be confined to a rigid classroom set-up. Instead, computing and communications - and therefore learning - could happen any place. Users in the workplace would enjoy similar advantages. Employees would be liberated from the constraints of physical networks, which would enhance creativity and personal productivity," Nagel said.

This type of "spontaneous" or "ad hoc" local are networking would supplement today's wired network configurations, which typically consist of telephone lines, coaxial cables, and fiber optics. The cost, particularly the capital cost, of hardwiring a building is high and then users are restricted as to when, how and where they can use their computers to move data.

Apple recognizes that radio spectrum is scarce and in high demand. Considering this, along with the intense activity being focused on proposals for new voice communications services, Apple is requesting that the FCC move quickly in giving equitable consideration to data communications when determining future bandwidth allocations.

"We're urging the public to support Apple's appeal that the allocation of radio spectrum go beyond voice communications to include an appropriate emphasis on data communications," Sculley said. "Our hope is that computer users will view the allocation of the radio spectrum for wireless computing as Apple does - as an important step in advancing the future of personal computing technology.

RECTIFICATION

For each redaction always a dragon: jumping into a text that is at last overwhelming your power to keep awake late in the evening.
When debugging the intr. ~ of Cor Bergshoeff's Relocation Program For 6502 uP Based Computers, the moon was already gone, so that now debugging of the debugged text is necessary. We decided to do better by replacing the full text.

RELOCATION PROGRAM FOR 6502 uP BASED COMPUTERS

By: Cor Bergshoeff, Denmark.
System: MC-65 computer

Have you ever been in a situation where you would like to relocate a program to another part of memory, especially a program of which you only had a HEX-dump?
It is not so bad if it is only a short program, but what about programs of, for instance, 4 Kbyte or more? It is a tedious job to check every byte.
This program makes it easier to relocate a program from one part of memory to another part of memory. It does not do the whole job, but most of it and I will explain what it does and not does for you.

When the program is started up, it asks for the START- and END address of the program to be relocated. It then asks for the START address of where you want to have the program relocated to, the END address is not asked for.
You are then asked to ENTER the START and END address of each table which might be situated in the program and whether it should be recalculated (VECTOR addresses) or not (i.e. TEXT). Each table should be entered separately and in the right sequence in which they occur in the program. If the last table does not coincidence with the END of the program, you must ENTER the END address as the START of a table and END+1 as the END address of this table and answer the question for 'relocate' with 'N or n'. The program will again ask for the next table which is answered by pressing the <ENTER> or <RETURN> key. After a few seconds, your program is relocated and has also recalculated the HI-bytes of all 3-bytes instructions. That is, all HI-bytes of the address which are situated within the program. All the HI-bytes of addresses outside the program to be relocated are, of course, to be left unchanged. The VECTOR addresses in a VECTOR table are also changed, but VECTORS pointing to addresses outside the program are left unchanged.

As I said, something is not done by the program and have to be done manual after the program is relocated at the new address. The problem is that, in some programs, the ACCU is loaded with the HI-byte and the Y- or X-register with the LO-byte of an address (or the other way around) after which a JMP or JSR is performed where these HI- and LO-bytes are used. The program can not detect whether a LDA- or LDX- or LDY-byte is an address or a value, so these instructions have to be checked manually. You can of course try and start the program at it's new location and se it works properly before checking it for these instru tions.

I have used only a few (AIM-65) MC-65-MONITO routines and these can easily be replaced by your o routines.

COHERENT; UNIX-like OS on AT-Clor

By: Wolfram Schimke, D(O) Berlin.

In October last year I bought a UNIX-like OS for my AT-Clone. It is COHERENT by Mark Williams Company COHERENT is created for 286- and 386-based PCs. You can't use it on machines with micro- channel-architecture requires at least 640KB RAM, one high-density floppy di drive and a minimum of 7MB space on your hard disk. If you want to use the on-line-manual and the spell-checker you should give at least 10MB. With the included bootstrapper you can boot from different partitions of you hard disk(s), selecting only by pressing one key. COHER ENT supports various types of harddisks but nevertheless there are some it won't work with. (The list of known incompatibilities included in the manual you can obtain fro me.) COHERENT will work with some SCSI- adaptors b it's not sure. The Perstore RLL harddisk controllers are n usable neither. My 80MB AT-Bus harddisk was fully acc ed by COHERENT. Modems are also supported. The only supported laser printer is the HP-LaserJet.
The installation is very easy. You are fully guided, but sometimes the printed manual is not more informative tha the message on the screen.
The manual (over 1000 pages) is easy to read. it is divide into two parts. The first is like every manual: some chapt on several topics. The second is kind of 'lexicon' (Mark Williams called it so). Thus you can quickly find answers your questions.
COHERENT is comparable with the UNIX-System III sta dard and covers the most UNIX commands. Of course the famous "man" (the on-line-manual) is present. Tools like yacc, awk, troff, nroff, uucp and others are also included. The source code is not available (except the vi-like editor, which is PD). With the editor also comes a spell-checker with English dictionaries.
Unfortunately you only have the bourne-shell. On the othe side you get a powerful development environment. Not as fine as TURBO-C++ - but with 'make' and extensive libr ies (e.g. termcap and curses). The editor allows multiple windows and has direct interface to the compiler. The com piler supports the ANSI-C standard only.
One reason that many people use UNIX is that DOS isn't able to easily run bigger applications. If you therefore wa to use COHERENT you will be soon resignated. COHER ENT is not capable of running task which exceed the 64 segment limit. You have to use multiple parallel tasks. There is DOS much easier! (Take the 900K EXE of TURBO-C++ for instance.) COHERENT has only this on memory model (it's like the TINY in DOS).
Nevertheless COHERENT is very useful for education an training.
I have got COHERENT directly from Mark Williams Com pany, 60 Revere Dr., Northbrook, IL 60062-9620, USA fo an introducers price of $99,95 (plus $30 shipping). You al can obtain it from Siener Soft, Wiesbaden or eMedia Gmb Hannover.
Remark: Another report on COHERENT is to be found i c't 11/90, page 150.