```
A/65     09-11     DIR      16-16     PATCH    23-23     TERMC    56-61
BEXEC*   12-12     INT      36-43     SETUP    32-37     TEST1    24-24
BRDGA    62-67     LOADER   17-17     SYSEXP   26-31     TEST2    25-25
DBG      68-73     LOADR2   18-18     TEMP     19-22
CREATE   13-14     MSC      74-76     TERMA    44-49
DELETE   15-15     OS65D3   00-08     TERMB    50-55

21 Files Defined

OK
```

```
OK
RUN DIR

A/65    09-11   DELETE 15-15   LOADR2 18-18   TEMP   19-22
  EXEC* 12-12   DIR    16-16   OSSUB  00-08   TEST   24-24
CREATE  13-14   LOADER 17-17   PATCH  23-23   TEST2  25-25

12 Files Declined

OK
!

?SN
OK
LIST BEXEC*

 10 REM BASIC EXECUTIVE
 20 REM
 30 REM SETUP INFLAG & OUFLAG FROM DEFAUL
 40 X=PEEK(10950): POKE 8993,X: POKE 8994,X
 50 POKE 9680,161 : REM CHANGE CURSOR CHARACTER (VIDEO)
 60 :
 100 :
 110 PRINT:PRINT
 120 PRINT"A/65 Assembler  Version 1.11"
 150 PRINT
 160 PRINT"Copyrisht 1980,81 by Pesasus Software"
 170 PRINT
 180 REM
 190 REM The file named PATCH contains the modifications
 200 REM which were made to DOS to allow the callins
 210 REM of A/65 from the DOS command loop.
 220 REM PATCH is an assembler file.
 230 REM
 240 REM If you have any questions or sussestions plesse
 250 REM call or write:
 260 REM
 270 REM       Pesasus Software
 280 REM         P.O Box 10014
 290 REM    Honolulu, Hawaii 96816
 300 REM
 310 REM       (808) 735-5013
 320 REM
 330 REM
 340 REM         Thank You.

OK

!

?SN
```

PATCH

```
.P

   10; 65D3 patch to vector A/65
   20;
   30          *=$2B23     ;formerly the D9 command function
   40;
   50          LDA #0
   60          JMP PATCH
   70;
   80          *=$2B68     ;entry for INIT
   90;
  100          JSR $2D73    ;print ascii strins subroutine
  110          .BYT 'OK??',0  ;new messase for INIT command
  120          JSR $2340
  130          CMP #'Y
  140          BNE $2BA6
  150          JMP $2768   ;initialize entire disk
  160;
  170 PATCH    STA $2CE5   ;index into kernel buffer
  180          JSR $2DA6   ;set track number from directory
  190          JMP $2AE8   ;brins in system
  200;
  210          *=$2E3C     ;function name table
  220;
  230          .BYT 'A//'  ;put name into table
  240;
  250;
  260          .END
```

TEMP
A/65

```
.P

   10          *=$6000
   20;
   30          JSR LABEL   ;this is a forward reference
   40          NOP
   50          NOP
   60;
   70 LABEL    RTS
   80;
   90;
  100; this is a sort of double forward reference
  110; on the first pass when the assembler encounters
  120; the LDA instruction, it doesn't know the values
  130; of L2 or AA.
  140;
  150          AA=L2/256*256
  160;
  170          LDA #L2-AA
  180;
  190;
  200;
  210; when an 8 bit operand is required, both A/65 and the
  220; OSI assembler will simple ignore the high byte if a
  230; 16 bit value is provided. This was not true of earlier
  240; versions of the OSI assembler.
  250; the above example should have been coded as follows:
  260;
  270          LDA #L2
  280;
  290; or for the older OSI assembler:
  300;
  310          LDA #L2*256/256 ;discard the high byte
  320;
  330; the newer version will accept either of these
  340;
  350;
  360 L2       .BYTE 0
  370;
  380          .END


.A
_^
CMD ERR

.
```

TEST1

A/65

```
!LO

.F

   10   .PAGE 'test file for the A/65 assembler'
   20;
   30;
   40          *=$8000
   50;
   60          OUTCH   =$2343
   70          CRLF    =$206A
   80          HEX     =$2092
   90          DISK    =$2A84
  100          TEMP    =$D0
  110;
  120 START   LDA #'A+1   ;this now works (should = 'B')
  130         JSR OUTCH   ;print it
  140         JSR CRLF
  150;
  160;
  170         LDY #0
  180 LOOP    TYA
  190         JSR HEX
  200         LDA #'
  210         JSR OUTCH
  220         INY
  230         CPY #11
  240         BNE LOOP
  250;
  260         JSR CRLF
  270;
  280         .FILE TEST2    ;chain next file
```

TEST2
A/65
1/2

```
.P
   10   .PAGE 'Second file of test program'
   20;
   30;
   40 DIR
   50          LDA #CMD
   60          STA $E1
   70          LDA #CMD/256
   80          STA $E2
   90          LDA #$11
  100          STA $2CED
  110          JSR DISK
  120;
  130          JSR DIRS
  140;
  150          LDA #CMD2
  160          STA $E1
  170          LDA #CMD2/256
  180          STA $E2
  190          LDA #$11
  200          STA $2CED
  210          JSR DISK
  220;
  230          JMP DIRS
  240;
  250;
  260 DIRS     LDY #0
  270;
  280 DIR3     LDX #6
  290          STX TEMP
  300          LDA $2E79,Y
  310          CMP #'#
  320          BNE DIR6
  330          CLC
  340          TYA
  350          ADC #6
  360          TAY
  370          BEQ PAU
  380          BNE DIR3
  390;
  400 DIR6     JSR OUTCH
  410;
  420 DIR4     INY
  430          LDA $2E79,Y
  440          DEC TEMP
  450          BNE DIR6
  460;
  470          PHA
  480          LDA #'
  490          JSR OUTCH
  500          PLA
  510;
  520 DIR5     JSR HEX
  530          LDA #'-
  540          JSR OUTCH
  550          INY
  560          LDA $2E79,Y
  570          INY
  580          JSR HEX
  590          JSR CRLF
  600          TYA
  610          BEQ PAU
```

```
620             BNE  DIR3
630;
640;
650 PAU         RTS
660;
670;
680 CMD         .BYT 'CA 2E79=08,1'
690 CMD2        .BYT 'CA 2E79=08,2'
700;
710             .END TEST1
```

TEST2
A165

(2/2)

TEST1


```
LINE # LOC      CODE          LINE

0002  0000                    ;
0003  0000                    ;
0004  0000                          *=$8000
0005  8000                    ;
0006  8000                    OUTCH   =$2343
0007  8000                    CRLF    =$206A
0008  8000                    HEX     =$2D92
0009  8000                    DISK    =$2A84
0010  8000                    TEMP    =$D0
0011  8000                    ;
0012  8000  A9 42             START   LDA #'A+1       ;this now works (should = 'B')
0013  8002  20 43 23                  JSR OUTCH       ;print it
0014  8005  20 6A 20                  JSR CRLF
0015  8008                    ;
0016  8008                    ;
0017  8008  A0 00                    LDY #0
0018  800A  98              LOOP    TYA
0019  800B  20 92 2D                JSR HEX
0020  800E  A9 20                    LDA #'
0021  8010  20 43 23                JSR OUTCH
0022  8013  C8                      INY
0023  8014  C0 0B                    CPY #11
0024  8016  D0 F2                    BNE LOOP
0025  8018                    ;
0026  8018  20 6A 20                JSR CRLF
0027  801B                    ;
0028  801B                          .FILE TEST2     ;chain next file
```

TEST?


```
LINE #  LOC      CODE          LINE

0030   801B                   ;
0031   801B                   ;
0032   801B                   DIR
0033   801B   A9 82                 LDA  #CMD
0034   801D   85 E1                 STA  $E1
0035   801F   A9 80                 LDA  #CMD/256
0036   8021   85 E2                 STA  $E2
0037   8023   A9 11                 LDA  #$11
0038   8025   8D ED 2C              STA  $2CED
0039   8028   20 84 2A              JSR  DISK
0040   802B                   ;
0041   802B   20 41 80              JSR  DIRS
0042   802E                   ;
0043   802E   A9 8E                 LDA  #CMD2
0044   8030   85 E1                 STA  $E1
0045   8032   A9 80                 LDA  #CMD2/256
0046   8034   85 E2                 STA  $E2
0047   8036   A9 11                 LDA  #$11
0048   8038   8D ED 2C              STA  $2CED
0049   803B   20 84 2A              JSR  DISK
0050   803E                   ;
0051   803E   4C 41 80              JMP  DIRS
   52   8041                   ;
   53   8041                   ;
0054   8041   A0 00           DIRS  LDY  #0
0055   8043                   ;
0056   8043   A2 06           DIR3  LDX  #6
0057   8045   86 D0                 STX  TEMP
0058   8047   B9 79 2E              LDA  $2E79,Y
0059   804A   C9 23                 CMP  #'#
0060   804C   D0 09                 BNE  DIR6
0061   804E   18                    CLC
0062   804F   98                    TYA
0063   8050   69 08                 ADC  #8
0064   8052   A8                    TAY
0065   8053   F0 2C                 BEQ  PAU
0066   8055   D0 EC                 BNE  DIR3
0067   8057                   ;
0068   8057   20 43 23        DIR6  JSR  OUTCH
0069   805A                   ;
0070   805A   C8              DIR4  INY
0071   805B   B9 79 2E              LDA  $2E79,Y
0072   805E   C6 D0                 DEC  TEMP
0073   8060   D0 F5                 BNE  DIR6
0074   8062                   ;
0075   8062   48                    PHA
0076   8063   A9 20                 LDA  #'
   77   8065   20 43 23              JSR  OUTCH
   78   8068   68                    PLA
0079   8069                   ;
0080   8069   20 92 2D        DIR5  JSR  HEX
0081   806C   A9 2D                 LDA  #'-
0082   806E   20 43 23              JSR  OUTCH
0083   8071   C8                    INY
0084   8072   B9 79 2E              LDA  $2E79,Y
```

```
0085   8075   C8                       INY
```

TEST2


```
LINE # LOC      CODE          LINE

0086   8076   20 92 2D                 JSR HEX
0087   8079   20 6A 2D                 JSR CRLF
0088   807C   98                       TYA
0089   807D   F0 02                    BEQ PAU
0090   807F   D0 C2                    BNE DIR3
0091   8081                     ;
0092   8081                     ;
0093   8081   60                PAU    RTS
0094   8082                     ;
0095   8082                     ;
0096   8082   43 41             CMD    .BYT 'CA 2E79=08,1'
0097   808E   43 41             CMD2   .BYT 'CA 2E79=08,2'
0098   809A                     ;
0099   809A                            .END TEST1
```

ERRORS = 0000 <0000>

SYMBOL TABLE
TEST 1 & 2

SYMBOL TABLE

SYMBOL VALUE

CMD	8052	CMD2	805E	CRLF	206A	DIR	8018
DIR3	8043	DIR4	805A	DIR5	8069	DIR6	8057
DIR8	8041	DISK	2A84	HEX	2D92	LOOP	800A
OUTCH	2343	PAU	8081	START	8000	TEMP	00D0

END OF ASSEMBLY

TEMP

```
0001   0000                        *=$8000
0002   8000                    ;
0003   8000   20 05 80            JSR LABLL       ;this is a forward reference
0004   8003   EA                  NOP
0005   8004   EA                  NOP
0006   8005                    ;
0007   8005   60              LABEL   RTS
0008   8006                    ;
0009   8006                    ;
0010   8006                    ; this is a sort of double forward reference
0011   8006                    ; on the first pass when the assembler encounters
0012   8006                    ; the LDA instruction, it doesn't know the values
0013   8006                    ; of L2 or AA.
0014   8006                    ;
0015   8006                    AA=L2/256*256
0016   8006                    ;
0017   8006   A9 0C                 LDA #L2-AA
0018   8008                    ;
0019   8008                    ;
0020   8008                    ;
0021   8008                    ; when an 8 bit operand is required, both A/65 and the
0022   8008                    ; OSI assembler will simple ignore the high byte if a
0023   8008                    ; 16 bit value is provided. This was not true of earlier
0024   8008                    ; versions of the OSI assembler.
0025   8008                    ; the above example should have been coded as follows:
0026   8008                    ;
0027   8008   A9 0C                 LDA #L2
0028   800A                    ;
0029   800A                    ; or for the older OSI assembler:
0030   800A                    ;
0031   800A   A9 0C                 LDA #L2*256/256 ;discard the high byte
0032   800C                    ;
0033   800C                    ; the newer version will accept either of these
0034   800C                    ;
0035   800C                    ;
0036   800C   00              L2      .BYTE 0
0037   800D                    ;
0038   800D                            .END

ERRORS = 0000 <0000>
```

SYMBOLE TABLE
TEMP

```
SYMBOL TABLE

SYMBOL    VALUE

..        8000    L2          800C    LABEL     8005
END OF ASSEMBLY
```