1 \*

2 \* PLOAD.INSTALL Source Code

3 \* BY STEVE ELLIS

4 \* COPYRIGHT (C) 1989

5 \* MICROSPARC, INC.

6 \* CONCORD, MA 01742

7 \*

8 \* MERLIN 16 ASSEMBLER

9 XC turn on 65816 opcodes

10 XC

11 ORG $4000 ;run at $4000

12

13 \* ProDOS equates

14 HIMEM EQU $73 ;himem pointer

15 EXTCMD EQU $BE06 ;vector to external commands

16 ERROUT EQU $BE09 ;ProDOS error handler

17 XTRNADR EQU $BE50 ;external command address for BI

18 XLEN EQU $BE52 ;length of command string minus 1

19 XCNUM EQU $BE53 ;BASIC command number (0 if external)

20 PBITS EQU $BE54 ;BI perms to be parsed

21 MLI EQU $BE70 ;MLI interface

22 FIFILID EQU $BEBB ;file ID type

23 FIAUXID EQU $BEB9 ;auxiliary file type

24 SREFNUM EQU $BEC7 ;GET\_FILE\_INFO reference number

25 MARK EQU $BEC8 ;in-file position mark

26 OSYSBUF EQU $BECE ;buffer for OPEN

27 OREFNUM EQU $BED0 ;OPEN file reference number

28 RWRFNUM EQU $BED6 ;READ/WRITE file reference number

29 RWDATA EQU $BED7 ;pointer to data to be used

30 RWCOUNT EQU $BED9 ;number of bytes to read/write

31 RWTRANS EQU $BEDB ;returned N of bytes read

32 CREFNUM EQU $BEDE ;CLOSE file reference number

33 GETBUFR EQU $BEF5 ;ProDOS buffer allocation routine

34 BITMAP EQU $BF58 ;ProDOS system bit map

35 IVERSION EQU $BFFD ;BI version number

36 COUT EQU $FDED ;character out routine

37

38 \* Storage for program variables

39 PTR EQU $00

40 UPKAR EQU $02

41 UPKSZ EQU $06

42

43 \* Constants

44 BLKSIZE EQU $1000 ;size of each data block

45 ;BLKSIZE MUST be an even divisor of $8000!

46 SOURCE EQU $00005000 ;source address of data block

47 DEST EQU $00E12000 ;super hi-res page

48

49 \* General purpose macros

50 \* Put the 65816 in emulation mode. 8 bit acc and registers

51 EMULATE MAC

52 SEC

53 XCE

54 <<<

55 \* Switch to native mode. 16 bit acc and registers

56 NATIVE MAC

57 CLC

58 XCE

59 REP #$30

60 <<<

61 \* Macro to simulate a branch to subroutine instruction

62 BSR MAC

63 PER \*+5

64 BRL ]1

65 <<<

66 \* Following are macros to perform MLI calls and

67 \* file error handling:

68 FILERR MAC

69 BCC \*+5

70 BRL MLIERR

71 <<<

72 GET\_FILE INFO MAC

73 LDA #$C4

74 JSR MLI

7S FILERR

76 <<<

77 OPEN MAC

78 LDA #$C8

79 JSR MLI

80 FILERR

81 <<<

82 READ MAC

83 LDA #$CA

84 JSR MLI

85 FILERR

86 <<<

87 SET\_MARK MAC

88 LDA #$CE

89 JSR MLI

90 FILERR

91 <<<

92

93 EMULATE

94 LDA EXTCMD+2 ;get page of other commands

95 CMP #$BE ;there are none

96 BEQ GETROOM ; so don't bother looking

97 STA SRCHNG+2 ;save the address in zero page

98 SEARCH0 LDY #0 ;start at byte 0

99 SEARCH INY ;bump that to byte 1

100 BEQ ALREADY ; means we have a match

101 SRCHNG LDA $0000,Y ;get a byte

102 CMP COMMAND+256,Y ;compare to our code

103 BEQ SEARCH ;if equal, look some more

104 INC SRCHNG+2 ;otherwise look at next higher page

105 LDA SRCHNG+2

106 CMP #$9A ;up to start of DOS yet?

107 BCC SEARCH0 ; no, search some more

108 BRA GETROOM ; yes, skip installed message

109

110 \* Print an error message stating that PLOAD has already

111 \* been installed and return to BASIC.

112 ALREADY LDY #0

113 :1 LDA AINSTL,Y ;get a character

114 BEQ :2 ;stop on 0

115 JSR COUT ;print the char.

116 INY ;finish message

117 BNE :1 ;always

118 :2 RTS ;back to BASIC

119

120 \* Ask ProDOS for room for our command

121 GETROOM LDA HIMEM+1 ;get top of free memory

122 CLC

123 ADC #4 ; add to that the ProDOS general buffer

124 STA OHIMEM ; save the result

125 LDA #>CMDEND-COMMAND ;get number of pages for our command

126 INC ;add one for total pages needed

127 JSR GETBUFR

121 BCC GOTBUF ;got them

129 JMP ERROUT ;otherwise exit with an error

130

131 \* Now that we've got the space, we relocate a few

132 \* addresses, and move our code up to its new home.

133 GOTBUF STA REL1+2

134 STA REL2+2

135 \* Update the system bitmap

136 MRKPAGE TAX ;get page number into acc

137 PHA ;save it

138 LSR ;shift it right a few times

139 LSR

140 LSR

141 TAY ; to address byte in bitmap

142 TXA

143 AND #7 ;isolate bit position

144 TAX

145 LDA #0

146 SEC ;mark the page with a 1 bit

147 :1 ROR

148 DEX

149 BPL :1

150 ORA BITMAP,Y ;mask with previous value

151 STA BITMAP,Y ;and store it

152 PLA ;get page number

153 INC ;bump it

154 CMP OHIMEM ;done all the pages?

155 BCC MRKPAGE ; no, finish it up

156

157 \* Check for BASIC version 1.1

158 CATMOD LDA IVERSION ;get BI version number

159 CMP #1 ;must be version 1.1 for catalog mods

160 BNE DCHAIN ;don't change anything

161 LDA #$C0 ;replace IVR and INT file desciptors

162 STA $B98E ; with PNT and PIC file descriptors

163 INC

164 STA $B98D

165 LDA #"P

166 STA $B9AF

167 STA $B9B2

168 LDA #"I

169 STA $B9B3

170 LDA #"C

171 STA $B9B4

172 LDA #"N

173 STA $B9B0

174 LDA #"T

175 STA $B9B1

176

177 \* Daisy-chain our command

178 DCHAIN NATIVE

179 LDA EXTCMD+1 ;get previous address of ext. commands

180 STA CMDLINK+4 ;and save so we can jump to it

181 RELI LDA #COMMAND ;put address of our command

182 STA EXTCMD+1 ; into external jump

183 LDA #CMDEND-COMMAND.$00FF ;last byte of program

184 LDX #COMMAND ;get source address

185 REL2 LDY #$0000 ;and destination address

186 MVN $00,$00 ;and move the program up

187 EMULATE

188 RTS

189 OHIMEM DS 1 ;room for old himem value

190 DS \ ;skip to next page

191

192 \* Scan the input line for our command

193 COMMAND CLD ;valid command handler identifier

194 PER PLOAD ;push run-time address of string PLOAD

195 LDY #0 ;scan for command

196 TYX ; on input line

197 :1 LDA $200,X ;get a char

198 INX

199 CMP #" " ;skip blanks

200 BEQ :1

201 AND #$DF ;convert lower case to upper

202 CMP (01,S),Y ;compare char. to command string

203 BNE CMDLINK

204 INY

205 CPY #5 ;got the whole word?

206 BCC :1 ; no, keep looking

207 DEY

208 STY XLEN ;put the len-1 in BI global page

209 STZ XCNUM ;command code = 0 means external handler

210 NATIVE

211 PER DO\_CMD ;push address of command handler

212 PLA ;find out what it is

213 STA XTRNADR ;and let the BI know where it is

214 LDA #$0401 ;require pathname 1, allow slot & drive

215 STA PBITS ; for BI parser

216 PLA ;pull address of 'PLOAD' off the stack

217 EMULATE

218 CLC ;let BI parse it

219 RTS

220 CMDLINK PLA ;clean up the stack

221 PLA

222 SEC ;not our command

223 JMP $0000 ; so jump to any other handlers

224

225 DO\_CMD NATIVE

226 PER SAVBUF ;push run-time address of save area

227 LDY #6

228 :1 LDA PTR,Y ;get a zero page byte

229 STA (01,S),Y ;save it

230 DEY

231 DEY

232 BPL :1 ;finish all the 8 bytes

233 PLA

234 LDA #SOURCE

235 STA PTR ;point to source data area

236 STA RWDATA ;tell MLI where to load data

237 LDA #DEST

238 STA UPKAR ;pointer to super hi-res screen

239 LDA #BLKSIZE

240 STA RWCOUNT ;read one data block at a time

241 STZ MARK ;zero file mark (start at byte 0)

242 LDA #$00E1 ;hi word of super hi-res screen location

243 STA UPKAR+2

244 LDA HIMEM ;set HIMEM address

245 STA OSYSBUF ;as buffer for OPEN

246 EMULATE

247 LDA #$40 ;initialize super hi-res

248 TSB $C029 ;without changing its current status

249 OPEN

250 LDA OREFNUM ;copy our reference number

251 STA RWRFNUM ; to read/write,

252 STA CREFNUM ; close, and

253 STA SREFNUM ; get\_info refnums

254 GET\_FILE\_INFO

255 LDA FIFILID ;check file ID type

256 CMP #$C1 ;full 32K image. no need to unpack

257 BEQ BIGPIC

258 CMP #$06 ;assume BIN files are 32K images

259 BEQ BIGPIC

260 CMP #$C0 ;packed picture image

261 BEQ PACPIC

262 BADTYPE LDA #$0D ;FILE TYPE MISMATCH

263 BRA MLIERR ;exit with error back to BASIC

264

265 CLOSE EMULATE

266 LDA #$CC ;CLOSE the file

267 JSR MLI

268 PER SAVBUF

269 LDY #7

270 :1 LDA (01,S),Y ;restore the ZP we trampled

271 STA PTR,Y

272 DEY

273 BPL :1

274 PLA

275 PLA

276 RTS ;return to BASIC

277

278 MLIERR PHA ;save acc.

279 BSR CLOSE ;close the file

280 PLA ;get acc.

281 JMP ERROUT ;abort

282

283 \* Load 32K images

284 BIGPIC EMULATE

285 READ

286 NATIVE

287 PHB ;save data bank

288 LDA #BLKSIZE-1 ;move one data block

289 LDX PTR ;from source address

290 LDY UPKAR ;to super hi-res page

291 MVN SOURCE,DEST

292 PLB ;restore data bank

293 LDA UPKAR ;find location on super hi-res page

294 CLC

295 ADC #BLKSIZE ;increment screen pointer by size of block

296 STA UPKAR

297 CMP #$A000 ;done with the picture (up to $A000)?

298 BNE BIGPIC ; no, do some more

299 BRA CLOSE ;close the file and exit

300

301 PACPIC NATIVE

302 LDA FIAUXID ;get file AUX type

303 BEQ TYPE00

304 CMP #0001

305 BEQ TYPE01

306 CMP #0002

307 BEQ T02JMP

308 EMULATE

309 BRL BADTYPE ;not a recognized packed file type

310 TO2JMP BRL TYPE02 ;can't reach it with a normal branch

311

312 \* Load and unpack aux type $00 files.

313 TYPE00 NATIVE

314 LDA #$7D00 ;only interested in data for unpacking

315 STA UPKSZ

316 LDA #$0020 ;read the palette

317 STA RWCOUNT

318 EMULATE

319 READ

320 LDX #0

321 TXA

322 SCBLP0 STAL $E19D00,X ;zero out the scan line area, since all

323 INX ;Paintworks pictures are 320 mode, palette 0

324 BNE SCBLP0

325 LDX #$1F

326 PALTLP0 LDA SOURCE,X

327 STAL $E19E00,X ;move palette to palette area

328 DES

329 BPL PALTLP0

330 NATIVE

331 LDA #$222 ;position past palette in file

332 STA MARK

333 LDA #BLKSIZE

334 STA RWCOUNT

335 T00LOOP EMULATE

336 SET\_MARK

337 READ

338 NATIVE

339 BSR UNPACK

340 LDA UPKAR

341 CMP #$9D00

342 BLT T00LOOP

343 BRL CLOSE

344

345 \* Load and unpack aux type $01 files.

346 TYPE01 NATIVE

347 LDA #$8000 ;SHR pic is $8000 bytes long

348 STA UPKSZ ;tell toolbox

349 T01LOOP EMULATE

350 SET\_MARK

351 READ

352 NATIVE

353 BSR UNPACK

354 LDA UPKSZ ;unpacked the entire picture?

355 BNE TOILOOP ; no. do some more

356 BRL CLOSE ; else exit through CLOSE

357

358 \* Load and unpack aux type $02 files.

359 TYPE02 NATIVE

360 LDA #$7D00 ;unpack only screen data (not SCB's, etc.)

361 STA UPKSZ

362 EMULATE

363 READ

364 NATIVE

365 LDA SOURCE+11 ;get number of horizontal pixels

366 CMP #320

367 BEQ PIXOK

368 CMP #640

369 BEQ PIXOK

370

371 \* If the picture doesn't have either 320 or 640 pixels, exit

372 \* to BASIC with a RANGE ERROR.

373 EMULATE

374 LDA #2

375 BRL MLIERR

376

377 \* Continue unpacking after determining a standard screen width.

378 \* First, copy the screen control byte for each scan line.

379 PIXOK

380 MX 00

381 LDA SOURCE+9 ;get hi-byte of SCB byte

382 XBA ;move it to high-byte of acc

383 ORA SOURCE+9 ; and get it in low-byte of acc

384 AND #$F0F0 ;only interested in high nibbles

385 LDX #0

386 SCBLP2 STAL $E19D00,X ;put it in SCB storage area

387 INX

388 INX

389 CPX #$C8 ;only up to $E19DC7

390 BNE SCBLP2

391 LDA #0 ;zero out from $E19DC8 -> $E19DFF

392 :1 STAL $E19D00,X

393 INX

394 INX

395 CPX #$100 ;done the whole page?

396 BNE :1 ; no, finish it up

397

398 \* Count the number of palettes and move them to where

399 \* they belong (from $E19E00 up).

400 LDX SOURCE+13 ;index with number of palettes

401 LDA #$00 ;use acc. to hold address

402 PALTLP2 CLC

403 ADC #$20 ;point to next palette

404 DEX

405 BNE PALTLP2 ;more palettes

406 TAX ;copy address of the end of the palettes

407 PHA ; and save it

408 :2 LDA SOURCE+15.X

409 STAL $E19E00,X ;move the data into palette area

410 DEX

411 DEX

412 BPL :2 ;more palette data

413

414 \* We have the start of the ScanLineDirectory now. Skip

415 \* over each entry (4 bytes) to find the beginning of the

416 \* packed picture data.

417 PLY

418 TYA ;get start addr. of entries in A

419 LDX SOURCE+15,Y ;get number of scan lines as index

420 PICLP2 CLC

421 ADC #4 ;skip an entry

422 DEX

423 BNE PICLP2 ;more to do

424 ADC #17 ;adjust pointer to correct address

425 STA MARK ;load from that point in file

426 LDA #BLKSIZE

427 STA RWCOUNT

428 T02LOOP EMULATE

429 SET\_MARK

430 READ

431 NATIVE

432 BSR UNPACK

433 LDA UPKAR

434 CMP #$9D00

435 BLT T02LOOP

436 BRL CLOSE

437

438 \* Call the toolbox to unpack the picture.

439 UNPACK NATIVE

440 LDX RWTRANS ;number of bytes actually read

441 LDA #0

442 PHA ;space for result

443 PHA ;pointer to buffer holding packed data

444 LDY PTR ;low word of buffer

445 PHY

446 PHX ;number of bytes read

447 PHA ;pointer to pointer to

448 PEA #UPKAR ; area to unpack into

449 PHA ;pointer to word holding length

450 PEA #UPKSZ ; of size of area to unpack into

451 LDX #$2703 ;tool number for UnPackBytes

452 JSL $E10000 ;call the toolbox

453 PLA ;get number of bytes unpacked

454 CLC

455 ADC MARK ;update the file mark by adding the number

456 STA MARK ; of bytes unpacked to previous mark

457 RTS

458

459 PLOAD ASC "PLOAD"

460 AINSTL HEX 8D

461 ASC "PLOAD ALREADY INSTALLED"

462 HEX 8D8D00

463 SAVBUF DS 8

464 CMDEND EQU \*