http://www.strotmann.de/twiki/bin/view/APG/AsmMiniAssembler
31 October 2004

# Apple II Mini-Assembler

The attached listing is a relocated version of the mini-assembler for the Apple II+ with instructions for the Apple II version. This version can be BRUN from the disk or BLOADed and start with CALL 2048 from Applesoft. To restart use CAll 2051. From machine language the start is 800G and the restart is 803G. Users who have Integer Basic available need only enter that language and use the instructions in part one of this note.

Please note that the mini-assembler performs a NEW command, so it will wipe out any resident Applesoft program. Also note that the mini-assembler loads from $800 to $947. Don't try to assemble anything into those locations.

This note covers the operation of the mini-assembler only. It is not a course in assembly language programming. For a reference on programming the 6502 microprocessor, refer to the Synertek Programming manual or any of the tutorials available. This note assumes the user has a working knowledge of 6502 programming and mnemonics.

The mini-assembler is a programming aid aimed at reducing the amount of time required to convert a handwritten program to object code. The mini-assembler is basically a look-up table for opcodes. With it, you can type mnemonics with their absolute addresses, and the assembler will convert it to the correct object code and store it in memory.

Typing "F666G" puts the user in mini-assembler mode. While in this mode, any line typed in will be interpreted as an assembly language instruction, assembled, and stored in binary form unless the first character on the command line is a "$".

If the first character of a command line is a "$", the remainder of the line will be interpreted as a normal monitor command, executed, and control returned to the mini-assembler. To get out of the mini-assembler, press RESET.

If the first character on the line is blank, the assembled instruction will be stored starting at the address immediately following the previously assembled instruction. If the first character is not a blank nor a "$", the line is assumed to contain an assembly language instruction preceded by the instruction address (a hex number followed by a ":"). In either case, the instruction will be retyped over the line just entered in dis-assembler format to provide a visual check of what has been assembled.

The counter that keeps track of where the next instruction will be stored is the pseudo PC (Program Counter) and it can be changed by many monitor commands (eg. 'L', 'T', . . .). Therefore, it is advisable to use the explicit instruction address mode after every monitor command and, of course, when the mini-assembler is first entered.

Errors (unrecognized mnemonic, illegal format, etc.) are signalled by a "beep" and a caret ("^") will be printed beneath the last character read from the input line by the mini-assembler.

The mnemonics and formats accepted by the mini assembler are the same as those listed by the 6502 Programmers Manual, with the following exceptions and differences:

1. All imbedded blanks are ignored, except inside addresses.
2. All addresses typed in are assumed to be in hex (rather than decimal or symbolic). A preceding "$" (indicating hex rather than decimal or symbolic) is therefore optional, except that it should not precede the instruction address).
3. Instructions that operate on the accumulator have a blank operand field instead of "A".
4. When entering a branch instruction, the argument of the branch mnemonic should be the address of the target of the branch. If the destination address is not known at the time the instruction is entered, simply enter an address that is in the neighborhood, and later re-enter the branch instruction with the correct target address. NOTE: If a branch target is specified that is out of range, the mini-assembler will flag the address as being in error.
5. The operand field of an instruction can only be followed by a comment field, which starts with a semicolon (";"). Obviously, the mini-assembler ignores the field and in fact will type over it when the line is typed over in disassembler format.
6. Any page zero references will generate page zero instruction formats if such a mode exists. There is no way to force a page zero address to be two bytes, even if the address has leading zeroes.

In general, to specify an addressing type, simply enter it as it would be listed in the disassembly. For information on the disassembler, see page 49 of the Apple II Reference Manual.

```
1    **********************
2    *                     *
3    *      APPLE-II       *
4    *   MINI-ASSEMBLER    *
5    *                     *
6    *  COPYRIGHT 1977 BY  *
7    * APPLE COMPUTER INC. *
8    *                     *
9    * ALL RIGHTS RESERVED *
10   *                     *
11   *     S. WOZNIAK      *
12   *      A. BAUM        *
13   **********************
14                              ; TITLE "APPLE-II MINI-ASSEMBLER"
15   FORMAT   EQU   $2E
16   LENGTH   EQU   $2F
17   MODE     EQU   $31
18   PROMPT   EQU   $33
19   YSAV     EQU   $34
20   L        EQU   $35
21   PCL      EQU   $3A
22   PCH      EQU   $3B
23   A1H      EQU   $3D
24   A2L      EQU   $3E
25   A2H      EQU   $3F
26   A4L      EQU   $42
27   A4H      EQU   $43
28   FMT      EQU   $44
29   IN       EQU   $200
30   INSDS2   EQU   $F88E
```

```
                31   INSTDSP  EQU   $F8D0
                32   PRBL2    EQU   $F94A
                33   PCADJ    EQU   $F953
                34   CHAR1    EQU   $F9B4
                35   CHAR2    EQU   $F9BA
                36   MNEML    EQU   $F9C0
                37   MNEMR    EQU   $FA00
                38   CURSUP   EQU   $FC1A
                39   GETLNZ   EQU   $FD67
                40   COUT     EQU   $FDED
                41   BL1      EQU   $FE00
                42   A1PCLP   EQU   $FE78
                43   BELL     EQU   $FF3A
                44   GETNUM   EQU   $FFA7
                45   TOSUB    EQU   $FFBE
                46   ZMODE    EQU   $FFC7
                47   CHRTBL   EQU   $FFCC
                48            ORG   $F500
F500: E9 81     49   REL      SBC   #$81       ;IS FMT COMPATIBLE
F502: 4A        50            LSR               ;WITH RELATIVE MODE?
F503: D0 14     51            BNE   ERR3       ;  NO.
F505: A4 3F     52            LDY   A2H
F507: A6 3E     53            LDX   A2L        ;DOUBLE DECREMENT
F509: D0 01     54            BNE   REL2
F50B: 88        55            DEY
F50C: CA        56   REL2     DEX
F50D: 8A        57            TXA
F50E: 18        58            CLC
F50F: E5 3A     59            SBC   PCL        ;FORM ADDR-PC-2
F511: 85 3E     60            STA   A2L
F513: 10 01     61            BPL   REL3
F515: C8        62            INY
F516: 98        63   REL3     TYA
F517: E5 3B     64            SBC   PCH
F519: D0 6B     65   ERR3     BNE   ERR        ;ERROR IF >1-BYTE BRANCH
F51B: A4 2F     66   FINDOP   LDY   LENGTH
F51D: B9 3D 00  67   FNDOP2   LDA   A1H,Y      ;MOVE INST TO (PC)
F520: 91 3A     68            STA   (PCL),Y
F522: 88        69            DEY
F523: 10 F8     70            BPL   FNDOP2
F525: 20 1A FC  71            JSR   CURSUP
F528: 20 1A FC  72            JSR   CURSUP     ;RESTORE CURSOR
F52B: 20 D0 F8  73            JSR   INSTDSP    ;TYPE FORMATTED LINE
F52E: 20 53 F9  74            JSR   PCADJ      ;UPDATE PC
F531: 84 3B     75            STY   PCH
F533: 85 3A     76            STA   PCL
F535: 4C 95 F5  77            JMP   NXTLINE    ;GET NEXT LINE
F538: 20 BE FF  78   FAKEMON3 JSR   TOSUB      ;GO TO DELIM HANDLER
F53B: A4 34     79            LDY   YSAV       ;RESTORE Y-INDEX
F53D: 20 A7 FF  80   FAKEMON  JSR   GETNUM     ;READ PARAM
F540: 84 34     81            STY   YSAV       ;SAVE Y-INDEX
F542: A0 17     82            LDY   #$17       ;INIT DELIMITER INDEX
F544: 88        83   FAKEMON2 DEY              ;CHECK NEXT DELIM
F545: 30 4B     84            BMI   RESETZ     ;ERR IF UNRECOGNIZED DELIM
F547: D9 CC FF  85            CMP   CHRTBL,Y   ;COMPARE WITH DELIM TABLE
F54A: D0 F8     86            BNE   FAKEMON2   ;NO MATCH
F54C: C0 15     87            CPY   #$15       ;MATCH, IS IT CR?
F54E: D0 E8     88            BNE   FAKEMON3   ;NO, HANDLE IT IN MONITOR
F550: A5 31     89            LDA   MODE
F552: A0 00     90            LDY   #$0
F554: C6 34     91            DEC   YSAV
F556: 20 00 FE  92            JSR   BL1        ;HANDLE CR OUTSIDE MONITOR
F559: 4C 95 F5  93            JMP   NXTLINE
F55C: A5 3D     94   TRYNEXT  LDA   A1H        ;GET TRIAL OPCODE
```

```
F55E: 20 8E F8  95            JSR   INSDS2     ;GET FMT+LENGTH FOR OPCODE
F561: AA        96            TAX
F562: BD 00 FA  97            LDA   MNEMR,X    ;GET LOWER MNEMONIC BYTE
F565: C5 42     98            CMP   A4L        ;MATCH?
F567: D0 13     99            BNE   NEXTOP     ;NO, TRY NEXT OPCODE.
F569: BD C0 F9  100           LDA   MNEML,X    ;GET UPPER MNEMONIC BYTE
F56C: C5 43     101           CMP   A4H        ;MATCH?
F56E: D0 0C     102           BNE   NEXTOP     ;NO, TRY NEXT OPCODE
F570: A5 44     103           LDA   FMT
F572: A4 2E     104           LDY   FORMAT     ;GET TRIAL FORMAT
F574: C0 9D     105           CPY   #$9D       ;TRIAL FORMAT RELATIVE?
F576: F0 88     106           BEQ   REL        ;YES.
F578: C5 2E     107  NREL     CMP   FORMAT     ;SAME FORMAT?
F57A: F0 9F     108           BEQ   FINDOP     ;YES.
F57C: C6 3D     109  NEXTOP   DEC   A1H        ;NO, TRY NEXT OPCODE
F57E: D0 DC     110           BNE   TRYNEXT
F580: E6 44     111           INC   FMT        ;NO MORE, TRY WITH LEN=2
F582: C6 35     112           DEC   L          ;WAS L=2 ALREADY?
F584: F0 D6     113           BEQ   TRYNEXT    ;NO.
F586: A4 34     114  ERR      LDY   YSAV       ;YES, UNRECOGNIZED INST.
F588: 98        115  ERR2     TYA
F589: AA        116           TAX
F58A: 20 4A F9  117           JSR   PRBL2      ;PRINT ^ UNDER LAST READ
F58D: A9 DE     118           LDA   #$DE       ;CHAR TO INDICATE ERROR
F58F: 20 ED FD  119           JSR   COUT       ;POSITION.
F592: 20 3A FF  120  RESETZ   JSR   BELL
F595: A9 A1     121  NXTLINE  LDA   #$A1       ;'!'
F597: 85 33     122           STA   PROMPT     ;INITIALIZE PROMPT
F599: 20 67 FD  123           JSR   GETLNZ     ;GET LINE.
F59C: 20 C7 FF  124           JSR   ZMODE      ;INIT SCREEN STUFF
F59F: AD 00 02  125           LDA   IN         ;GET CHAR
F5A2: C9 A0     126           CMP   #$A0       ;ASCII BLANK?
F5A4: F0 13     127           BEQ   SPACE      ;YES
F5A6: C8        128           INY
F5A7: C9 A4     129           CMP   #$A4       ;ASCII '$' IN COL 1?
F5A9: F0 92     130           BEQ   FAKEMON    ;YES, SIMULATE MONITOR
F5AB: 88        131           DEY              ;NO, BACKUP A CHAR
F5AC: 20 A7 FF  132           JSR   GETNUM     ;GET A NUMBER
F5AF: C9 93     133           CMP   #$93       ;':' TERMINATOR?
F5B1: D0 D5     134  ERR4     BNE   ERR2       ;NO, ERR.
F5B3: 8A        135           TXA
F5B4: F0 D2     136           BEQ   ERR2       ;NO ADR PRECEDING COLON.
F5B6: 20 78 FE  137           JSR   A1PCLP     ;MOVE ADR TO PCL, PCH.
F5B9: A9 03     138  SPACE    LDA   #$3        ;COUNT OF CHARS IN MNEMONIC
F5BB: 85 3D     139           STA   A1H
F5BD: 20 34 F6  140  NXTMN    JSR   GETNSP     ;GET FIRST MNEM CHAR.
F5C0: 0A        141  NXTM     ASL
F5C1: E9 BE     142           SBC   #$BE       ;SUBTRACT OFFSET
F5C3: C9 C2     143           CMP   #$C2       ;LEGAL CHAR?
F5C5: 90 C1     144           BCC   ERR2       ;NO.
F5C7: 0A        145           ASL              ;COMPRESS-LEFT JUSTIFY
F5C8: 0A        146           ASL
F5C9: A2 04     147           LDX   #$4
F5CB: 0A        148  NXTM2    ASL              ;DO 5 TRIPLE WORD SHIFTS
F5CC: 26 42     149           ROL   A4L
F5CE: 26 43     150           ROL   A4H
F5D0: CA        151           DEX
F5D1: 10 F8     152           BPL   NXTM2
F5D3: C6 3D     153           DEC   A1H        ;DONE WITH 3 CHARS?
F5D5: F0 F4     154           BEQ   NXTM2      ;YES, BUT DO 1 MORE SHIFT
F5D7: 10 E4     155           BPL   NXTMN      ;NO
F5D9: A2 05     156  FORM1    LDX   #$5        ;5 CHARS IN ADDR MODE
F5DB: 20 34 F6  157  FORM2    JSR   GETNSP     ;GET FIRST CHAR OF ADDR
F5DE: 84 34     158           STY   YSAV
```

```
F5E0: DD B4 F9  159           CMP   CHAR1,X    ;FIRST CHAR MATCH PATTERN?
F5E3: D0 13     160           BNE   FORM3      ;NO
F5E5: 20 34 F6  161           JSR   GETNSP     ;YES, GET SECOND CHAR
F5E8: DD BA F9  162           CMP   CHAR2,X    ;MATCHES SECOND HALF?
F5EB: F0 0D     163           BEQ   FORM5      ;YES.
F5ED: BD BA F9  164           LDA   CHAR2,X    ;NO, IS SECOND HALF ZERO?
F5F0: F0 07     165           BEQ   FORM4      ;YES.
F5F2: C9 A4     166           CMP   #$A4       ;NO,SECOND HALF OPTIONAL?
F5F4: F0 03     167           BEQ   FORM4      ;YES.
F5F6: A4 34     168           LDY   YSAV
F5F8: 18        169  FORM3    CLC              ;CLEAR BIT-NO MATCH
F5F9: 88        170  FORM4    DEY              ;BACK UP 1 CHAR
F5FA: 26 44     171  FORM5    ROL   FMT        ;FORM FORMAT BYTE
F5FC: E0 03     172           CPX   #$3        ;TIME TO CHECK FOR ADDR.
F5FE: D0 0D     173           BNE   FORM7      ;NO
F600: 20 A7 FF  174           JSR   GETNUM     ;YES
F603: A5 3F     175           LDA   A2H
F605: F0 01     176           BEQ   FORM6      ;HIGH-ORDER BYTE ZERO
F607: E8        177           INX              ;NO, INCR FOR 2-BYTE
F608: 86 35     178  FORM6    STX   L          ;STORE LENGTH
F60A: A2 03     179           LDX   #$3        ;RELOAD FORMAT INDEX
F60C: 88        180           DEY              ;BACKUP A CHAR
F60D: 86 3D     181  FORM7    STX   A1H        ;SAVE INDEX
F60F: CA        182           DEX              ;DONE WITH FORMAT CHECK?
F610: 10 C9     183           BPL   FORM2      ;NO.
F612: A5 44     184           LDA   FMT        ;YES, PUT LENGTH
F614: 0A        185           ASL              ;IN LOW BITS
F615: 0A        186           ASL
F616: 05 35     187           ORA   L
F618: C9 20     188           CMP   #$20
F61A: B0 06     189           BCS   FORM8      ;ADD "$" IF NONZERO LENGTH
F61C: A6 35     190           LDX   L          ;AND DON'T ALREADY HAVE IT
F61E: F0 02     191           BEQ   FORM8
F620: 09 80     192           ORA   #$80
F622: 85 44     193  FORM8    STA   FMT
F624: 84 34     194           STY   YSAV
F626: B9 00 02  195           LDA   IN,Y       ;GET NEXT NONBLANK
F629: C9 BB     196           CMP   #$BB       ;';' START OF COMMENT?
F62B: F0 04     197           BEQ   FORM9      ;YES
F62D: C9 8D     198           CMP   #$8D       ;CARRIAGE RETURN?
F62F: D0 80     199           BNE   ERR4       ;NO, ERR.
F631: 4C 5C F5  200  FORM9    JMP   TRYNEXT
F634: B9 00 02  201  GETNSP   LDA   IN,Y
F637: C8        202           INY
F638: C9 A0     203           CMP   #$A0       ;GET NEXT NON BLANK CHAR
F63A: F0 F8     204           BEQ   GETNSP
F63C: 60        205           RTS
                206           ORG   $F666
F666: 4C 92 F5  207  MINIASM  JMP   RESETZ
```

-- CarstenStrotmann - 26 Mar 2003