```
0001                        *  Move Block .01
0002                        *  By Adam Trionfo, based on a program by Eric Beckett.
0003                        *  July 14, 2010
0004                        *
0005                        *  This is a re-written version of the Move Square program by Eric
0006                        *  Beckett from the Aug, Sept, and Nov, 1984 issues of the APF
0007                        *  Imagination  Machine Newletter, "I-M 1 in a Million."  That
0008                        *  version required BASIC.  This version runs as a cartridge.
0009                        *
0010                        *  This program is written to be run on the circa 1978 APF
0011                        *  MP-1000 computer game console which uses the
0012                        *  Motorola 6800 8-Bit CPU.
0013                        *
0014                        *  Program Description
0015                        *  -------------------
0016                        *
0017                        *  The APF Menu gives one choice to the user: 1) RUN PROGRAM
0018                        *
0019                        *  This program places a white block in the middle of the screen.
0020                        *  The user can control it using the Right Joystick.  The four
0021                        *  cardinal directions are supported-- diagnals are NOT supported.
0022                        *
0023                        *  Notes:
0024                        *  ------
0025                        *  There is NO boundry checking of the White Block.  Moving the
0026                        *  cursor left or right will cause the Block to "wrap."  If the
0027                        *  White Block is moved up or down off the screen, then it is
0028                        *  possible to crash the program because the Block is moving
0029                        *  into RAM.
0030                        *
0031                        *  Version
0032                        *  -------
0033                        *  .01 - First Release
0034
0035                        *  Equates
0036
0037 4296                  FILLSCRN  EQU  $4296        ; Fill Screen Routine
0038 02ee                  MIDSCR    EQU  $02EE        ; Location of Middle of Screen
0039 0180                  CURLOC    EQU  $0180        ; Current Location of the White Block
0040 01f2                  JOYDATA   EQU  $01F2        ; Joystick Data
0041 41d9                  READJOY2  EQU  $41D9        ; Read Right Joystick
0042 004e                  NORTH     EQU  $4E          ; ASCII Capital N
0043 0053                  SOUTH     EQU  $53          ; ASCII Capital S
0044 0045                  EAST      EQU  $45          ; ASCII Capital E
0045 0057                  WEST      EQU  $57          ; ASCII Capital W
0046 0080                  BACKGRND  EQU  $80          ; ASCII of Background
0047 00cf                  WHITEBLK  EQU  $CF          ; ASCII of White Block
0048
0049 8000                            ORG   $8000       ; Start of APF Cartridge Area
0050
0051                        *  The BIOS uses the first five bytes of the cartridge
0052
0053 8000 bb                         FCB   $BB         ; Tell BIOS a cart is present
0054 8001 80 87                      FDB   MENUSTR     ; Points to Menu string on cartridge
0055 8003 31                         FCB   $31         ; LSB 4 bits is # of Menu Choices
0056                        *                            Menu has one choice
0057 8004 00                         FCB   $00         ; Checksum-type byte.  Must be $00 else
0058                        *                            BIOS restarts right after Init
0059
0060                        *  After a menu choice is made from the APF Menu via the
0061                        *  BIOS, then control passes to the program here ($8005).
0062
0063 8005 bd 42 96                   JSR   FILLSCRN    ; Fill Screen Background with Black
```

```
0064
0065                        **********************
0066                        *                    *
0067                        *  Main Program Loop *
0068                        *                    *
0069                        **********************
0070
0071 8008 bd 80 11                   JSR   INITPROG    ; Initialize Program
0072 800b bd 80 1c          MOVBLOCK  JSR   READSTIK    ; Read Joystick and Move Block
0073 800e 7e 80 0b                   JMP   MOVBLOCK    ; Continue Running Program Forever
0074
0075                        *****************
0076                        *               *
0077                        *  Subroutines  *
0078                        *               *
0079                        *****************
0080
0081                        * Initialize Program
0082                        * ------------------
0083                        * Move the White Block to the middle of the screen.
0084
0085 8011 ce 02 ee          INITPROG  LDX   #MIDSCR     ; Middle of Screen
0086 8014 ff 01 80                    STX   CURLOC      ; Current location of the White Block
0087 8017 86 cf                       LDAA  #WHITEBLK   ; Place ASCII White Block into Accum A
0088 8019 a7 00                       STAA  $00,X       ; Store White Block on Screen
0089 801b 39                          RTS
0090
0091                        *  Has the Joystick been moved?
0092                        *  ----------------------------
0093
0094 801c bd 41 d9          READSTIK  JSR   READJOY2    ; Right Joystick Subroutine
0095 801f 25 01                       BCS   JOY2USED    ; Branch if Joystick Used
0096 8021 39                          RTS               ; No Movement- Return and keep checking
0097
0098                        *  Check which direction the joystick has been moved
0099                        *  -------------------------------------------------
0100
0101                        * Check North
0102 8022 b6 01 f2          JOY2USED  LDAA  JOYDATA     ; Put Joystick Direction Data into Accum A
0103
0104 8025 81 4e                       CMPA  #NORTH      ; Did Joystick Move North?
0105 8027 26 03                       BNE   CHKSOUTH    ; If not, then Check South Movement
0106 8029 7e 80 58                    JMP   MOVNORTH    ; Direction was North, Branch
0107
0108                        * Check South
0109 802c 81 53             CHKSOUTH  CMPA  #SOUTH      ; Did Joystick Move South?
0110 802e 26 03                       BNE   CHKEAST     ; If not, then Check East Movement
0111 8030 7e 80 68                    JMP   MOVSOUTH    ; Direction was South, Branch
0112
0113                        * Check East
0114 8033 81 45             CHKEAST   CMPA  #EAST       ; Did Joystick Move East?
0115 8035 26 03                       BNE   CHKWEST     ; If not, then Check West Movement
0116 8037 7e 80 42                    JMP   MOVEAST     ; Direction was East, Branch
0117
0118                        * Check West
0119 803a 81 57             CHKWEST   CMPA  #WEST       ; Did Joystick Move West?
0120 803c 26 03                       BNE   NODIR       ; If not, then No Direction was Picked
0121 803e 7e 80 4d                    JMP   MOVWEST     ; Direction was West, Branch
0122
0123 8041 39                NODIR     RTS               ; No Movement- Return and keep checking
0124
0125                        * Update White Block's Movement on the Screen
0126                        * ----------------------------------------
```

```
0127
0128                        * White Block's East Movement
0129 8042 fe 01 80          MOVEAST   LDX   CURLOC      ; Load Current Location of White Block
0130 8045 86 80                       LDAA  #BACKGRND   ; Load Background Color into Accum A
0131 8047 a7 00                       STAA  $00,X       ; Store Background Color onto screen
0132 8049 08                          INX               ; Move Block Location East
0133 804a 7e 80 75                    JMP   DRAWBLOK    ; Jump to Draw White Block Subroutine
0134
0135                        * White Block's West Movement
0136 804d fe 01 80          MOVWEST   LDX   CURLOC      ; Load Current Location of White Block
0137 8050 86 80                       LDAA  #BACKGRND   ; Load Background Color into Accum A
0138 8052 a7 00                       STAA  $00,X       ; Store Background Color onto screen
0139 8054 09                          DEX               ; Move Block Location West
0140 8055 7e 80 75                    JMP   DRAWBLOK    ; Jump to Draw White Block Subroutine
0141
0142                        * White Block's North Movement
0143 8058 fe 01 80          MOVNORTH  LDX   CURLOC      ; Load Current Location of White Block
0144 805b 86 80                       LDAA  #BACKGRND   ; Load Background Color into Accum A
0145 805d a7 00                       STAA  $00,X       ; Store Background Color onto screen
0146 805f 86 20                       LDAA  #32D        ; Initialize Loop for 32 Times
0147 8061 09                BLOCKUP   DEX               ; Countdown Loop for "Up" Movement
0148 8062 4a                          DECA              ; Decrement "Up" Movement Loop
0149 8063 26 fc                       BNE   BLOCKUP     ; Has screen RAM moved "Up"
0150 8065 7e 80 75                    JMP   DRAWBLOK    ; Jump to Draw White Block Subroutine
0151
0152                        * White Block's South Movement
0153 8068 fe 01 80          MOVSOUTH  LDX   CURLOC      ; Load Current Location of White Block
0154 806b 86 80                       LDAA  #BACKGRND   ; Load Background Color into Accum A
0155 806d a7 00                       STAA  $00,X       ; Store Background Color onto screen
0156 806f 86 20                       LDAA  #32D        ; Initialize Loop for 32 Times
0157 8071 08                BLOKDOWN  INX               ; Countdown Loop for "Down" Movement
0158 8072 4a                          DECA              ; Decrement "Down" Movement Loop
0159 8073 26 fc                       BNE   BLOKDOWN    ; Has screen RAM moved "Down"
0160
0161                        * Draw White Block - Draw Block at New Locaction.
0162                        * ----------------
0163                        DRAWBLOK
0164 8075 86 cf                       LDAA  #WHITEBLK   ; Place ASCII White Block into Accum A
0165 8077 a7 00                       STAA  $00,X       ; Draw White Block to Screen
0166 8079 ff 01 80                    STX   CURLOC      ; Update Current Location of White Block
0167
0168                        *  Delay Loop
0169                        *  ----------
0170                        *
0171                        *  This loop determines the speed of the block.
0172
0173 807c c6 32             DELYLOOP  LDAB  #50D         ; Number of Outer Loops
0174 807e 86 ff             OUTLOOP   LDAA  #255D        ; Number of Inner Loops
0175 8080 4a                INLOOP    DECA               ; Decrement Inner Loop
0176 8081 26 fd                       BNE   INLOOP       ; All Inner Loops Complete?
0177 8083 5a                          DECB               ; Decrement Outer Loop
0178 8084 26 f8                       BNE   OUTLOOP      ; All Outer Loops Complete?
0179
0180 8086 39                          RTS                ; Check for More Joystick Movement
0181
0182                        *  Cartridge Menu String
0183                        *  ---------------------
0184                        *
0185                        *  The data required by the BIOS for the main menu is here.
0186
0187 8087 e9                MENUSTR   FCB   $E9          ; 9 spaces before Program Name string
0188 8088 4d 4f 56 45 20 42           FCC   "MOVE BLOCK  .01"  ; Program Name String
     4c 4f 43 4b 20 20
```

```
     2e 30 31
0189 8097 e8                           FCB   $E8         ; 8 spaces after text (EOL)
0190 8098 e1                           FCB   $E1         ; 1 spaces (Before Author)
0191 8099 42 59 20 41 44 41            FCC   "BY ADAM TRIONFO, JULY 14, 2010"  ; Author
     4d 20 54 52 49 4f
     4e 46 4f 2c 20 4a
     55 4c 59 20 31 34
     2c 20 32 30 31 30
0192 80b7 e3                           FCB   $E3         ; 3 spaces (After Author)
0193 80b8 e3                           FCB   $E3         ; 3 spaces (Before Author)
0194 80b9 42 41 53 45 44 20            FCC    "BASED ON ERIC BECKETT'S"
     4f 4e 20 45 52 49
     43 20 42 45 43 4b
     45 54 54 27 53
0195 80d0 e7                           FCB   $E7         ; 7 spaces (After Author)
0196 80d1 42 41 53 49 43 2f            FCC    "BASIC/ML EXAMPLE FROM 1984"
     4d 4c 20 45 58 41
     4d 50 4c 45 20 46
     52 4f 4d 20 31 39
     38 34
0197 80eb fe                           FCB   $FE         ; Another 30 spaces
0198 80ec e5                           FCB   $E5         ; Another 5 spaces
0199 80ed 31 2e 20 52 55 4e            FCC   "1. RUN PROGRAM                  "
     20 50 52 4f 47 52
     41 4d 20 20 20 20
     20 20 20 20 20 20
     20 20 20 20 20 20
     20 20
0200 810d ff                           FCB   $FF         ; Control Byte - End of String
0201
0202 810e ff ff ff ff ff ff  CARTEND   FILL  $FF,$9000-CARTEND ; Pad with $FF for a 4K Cart
     ff ff ff ff ff ff
     ff ff ff ff ff ff
     ff ff ff ff ff ff
     ff ff ff ff ff ff
     ff ff ff ff ff ff
     ff ff ff ff ff ff
     ff ff ff ff ff ff
     ff ff ff ff ff ff
     ff ff ff ff
0203
0204                         * End of Program
```