MLS - FLIP FLOP BOARD

```
09A4  22  DEC   R2 ;Stack pointer free
  A5  8A  GLO   RA ;D=M(R(A))
  A6  52  STR   R2 ;Push RA.0 for later done test
  A7  FC  ADI      ;Add 58 hex to RA.0 to address last board square
  A8  58
  A9  AA  PLO   RA ;RA.0=D (place new address in RA.0)
  AA  0A  LDN   RA ;D=M(R(A)) -- get a byte from board
  AB  32  BZ       ;If = 00, empty space, branch past next part
  AC  B5           ;      which compliments pieces
  AD  FB  XRI      ;Test if = FF (border byte)/pieces
  AE  FF
  AF  32  BZ       ;If = FF, branch to skip next

09B0  B5
  B1  0A  LDN   RA ;D=M(R(A)) -- get byte from board (piece)
  B2  FB  XRI      ;Exclusive "OR" with 81 to compliment
  B3  81
  B4  5A  STR   RA ;Replace complimented byte in board
  B5  2A  DEC   RA ;RA points back to next byte in board
  B6  8A  GLO   RA ;D=RA.0 to test for done
  B7  F3  XOR      ;Compare RA.0:byte on stack (original RA.0)
  B8  3A  BNZ      ;If ≠, branch to continue flip flop
  B9  AA
  BA  12  INC   R2 ;Else reset stack pointer
  BB  D4  SEP   R4 ;Return control to Chip-8 Interpreter
```

MLS - GET BEST MOVE

```
09BC  0A  LDN   RA ;Get first byte from move table
  BD  FB  XRI      ;Test if = FF (no moves in list)
  BE  FF
  BF  3A  BNZ      ;If ≠ FF, skip early return

09C0  C2
  C1  D4  SEP   R4 ;Early return--game (or look ahead)over now
  C2  22  DEC   R2 ;Stack pointer free
  C3  1A  INC   RA ;RA+2 points to first weight in list
  C4  1A  INC   RA ;   "          "          "
  C5  9A  GHI   RA ;Set RE=RA
  C6  BE  PHI   RE ;  "    "
  C7  8A  GLO   RA ;  "    "
  C8  AE  PLO   RE ;  "    "
  C9  9E  GHI   RE ;Set RA=RE (needed on subsequent loops)
```