Arduino 6502 Emulation and Games.

Came across this interesting post while I was looking for a chess program for the Arduino.

I was sure the Arduino UNO could play the game, even if it would be very slow. What I found, was a 6502 Emulator, that could run MicroChess - you play the game in the serial monitor. It’s a little tricky, doesn’t have any help, or instructions. But it is very playable, and pretty good.

I found a couple of errors in the logic, like It clearly won a couple of times, but never took advantage of the lead it had. And in the same game, gave up it’s Queen, one Rook, and one Knight, never using the other Rook, or Knight. In the one I did win, but that was 5 or 6 moves after it should have declared that it had won.

So not perfect, but pretty good. I also noticed that a couple of times, it let me make an illegal move, by letting move a pawn backward, and too many spaces.

Still, pretty good effort for an 8bit, no memory micro-controller.

The Following is taken from Obsolescenceguaranteed Blog: (Link Below)

#### **Playing Microchess - Command Summary**

**---------------------------------------------------------------**

|**CH2 (Serial)** | **CH2 (Serial)** |

|**-------------------------------------------------------------**|

|**\*7** (C) Clear Board | **1-7** (1-7) Keys to enter move|

|**\*8** (E) Exchange sides | **#** (Return) Register move |

|**\*9** (P) Play | |

|**-------------------------------------------------------------**|

|**Joystick up**: see hex digit 1 | **Joystick down**: see hex digit 3|

|**-------------------------------------------------------------**|

|**\*4** (L) Load Board | **\*5** (S) Save Board |

|**\*6** (W) Blitz play (fast & dumb) |

---------------------------------------------------------------

Gameplay is intuitive if you think of it this way:

* Start with C-E-P, or \*7 for reset, then \*8 to swap sides and \*9 to ask Microchess for a move.
* On the CH2 these are all on the bottom line of numbers in their logical order...
* If you are lazy, you keep doing \*8, \*9 to let Microchess play itself.
* Or, you enter your own moves by entering (for example - see further below) 1333#,
* followed by \*9 to let Microchess make its next move in response.
* The board is printed on the serial port. On the stand-alone CH2, use the joystick to see:
* (up) the piece moved/to move,
* (down) its destination TO square or
* (middle position) its FROM square.

Some extra functions have been added, based on suggestions in the Microchess manual. These are implemented as interventions on top of the 6502 emulator. Rest assured, no vintage bytes from within Microchess have been hurt in the process. You're running 100% original 6502 Microchess :)

**Blitz mode**: W, or \*6, toggles between normal speed (approx. 100 sec per move) and Blitz mode (approx. 10 sec. per move). Note that the CH2 LED flashes for every step taken during Microchess' next move calculation. So the frequency of flashes reminds you which mode you're in.

**Save Board:** You can save your game using the EEPROM in the Arduino. This simply saves the current chess board so you can continue playing at a later time. Press S (serial) or \*5 (CH2) and then enter the slot number you want to save to - a number from 0 to 9. Confirm by hitting return (serial) or # (CH2) and the board is saved.

**Load board:** done by L (serial) or \*4 (CH2). This retrieves a saved game and allows you to continue to play it.

**1. Initialise**

=============

Enter C for Clear. Actually, at any time, hit C to reset the game.

--> The board is reprinted & bottom shows CC CC CC to confirm.

**2. Let the computer play against itself**

=======================================

Enter E to make the computer take the opposite side of the board.

--> The bottom line shows EE EE EE to confirm you swapped chairs.

Enter P to make the computer Play its move.

--> Deep thought happens. Every few seconds, a dot appears to indicate

one more possible move has been thought through.

--> Typically, you'll see 30 or so dots before the computer is done.

It takes about 100 seconds on a real KIM-I, more or less the same

on the Arduino.

When the best move has been decided, MicroChess prints out the board again.

--> The bottom line shows its move in 3 hex numbers.

Hex number 1, left digit: 0 if a Black piece was moved, 1 for White

Hex number 1, right digit: Tells you what piece this was:

----------------------------------------------------------------

| 0 - KING | 4 - King Bishop | B - K R Pawn | C - K B Pawn |

| 1 - Queen | 5 - Queen Bishop | 9 - Q R Pawn | D - Q B Pawn |

| 2 - King Rook | 6 - King Knight | A - K N Pawn | E - Q Pawn |

| 3 - Queen Rook| 7 - Queen Knight | B - Q N Pawn | F - K Pawn |

----------------------------------------------------------------

Hex number 2 is the FROM square (row, column).

Hex number 3 is the TO square (row, column).

So you can let the computer play both sides by hitting E, then P, E again, P...

**3. Entering your own move**

=========================

Instead of hitting P, you can enter the 2-digit FROM square and then the 2-digit TO square. After every digit, you'll see the board reprinted. But focus on the bottom line: the digit you press "rolls in" to the bottom line's Numbers 2 and 3 from right to left. This is a remnant of how the KIM-I uses its onboard LEDs.

After four digits, the move is defined. MicroChess shows the piece involved in Hex Number 1: first digit is 0 for white, 1 for black. Second digit is as per the table above.

Once your move is complete, hit Return to register your move.

--> FF appears in Hex Number 1: you have now moved the piece,

so FF is there to show that the From square is now empty.

After you've registered the move you can still undo it. Just correct the wrong move by entering four more digits in a correcting move, so that the board looks like you intended it to.

Indeed - this is the fundamental point to grasp: MicroChess does not check what you're doing when you move pieces around. You can move its pieces as well as your own. All you are doing here is rearranging the board for the next round of Microchess Deep Thought. You can make a normal move, or shift stuff around on the board as you wish to create a new situation that MicroChess should play against next. It's a feature, not a limitation!

--> Hit P to make MicroChess Play its next move.

**4. Special moves:**

=================

Castling: just move the two pieces and hit Return after each one to register them both. En passant: break the move up in two moves. The mid-point being on the piece you'll strike out. Queening pawns: yes. Well. Remember which of your pawns has been Queened and give it the according moves afterwards. MicroChess will not Queen on its side. (It can be done on a Kim-I by leaving the program and manipulating its memory through the KIM Monitor. But no such option in the Arduino version. Slight imperfection).

Here is the information:

<http://obsolescenceguaranteed.blogspot.com/2014/06/6502-microchess-on-arduino.html?showComment=1432598515374>

Here is the manual to MicroChess:

<http://users.telenet.be/kim1-6502/microchess/microchess.html>

The original MicroChess in assembly code:

<http://benlo.com/files/Microchess6502.txt>

The Chess program, is/was based off the Arduino 6502 Emulator. The 6502 Emulator lets you run BASIC code on your Arduino. And can be found here:

<http://forum.arduino.cc/index.php?topic=193216.0>

There are still limitations, 750ish bytes of program space, not much. No way to save or load programs, but it’s very cool to see BASIC running on a UNO

Thou there is a lot of talk about adding SPI memory, and maybe emulation of a APPLE IIe

Still more Basic for the Arduino: (not based on the 6502)

<https://coronax.wordpress.com/tag/basic/>

<https://github.com/BleuLlama/TinyBasicPlus>

Arduino Retro Computer: Basic Interpreter

<http://clintkprojects.blogspot.com/2015/06/arduino-retro-computer-basic.html>

Arduino Basic:

<http://hamsterworks.co.nz/mediawiki/index.php/Arduino_Basic>

<https://github.com/robinhedwards/ArduinoBASIC>