Card Shuffling Program for KIM - 1

Your 6502 might play poker like Amarillo KIM, but does it always have to pass the deal? Not if you teach it to shuffle cards!

Hark Chan
P.O. Box 714
Cambridge, MA 02139

Entertaining friends with computer games certainly makes all the effort of assembling a personal computer worthwhile. However, if you happen to have a small microcomputer with limited memory and very few software tools, there are not many games available. As an example, most card games need a random number generator to shuffle cards.

The standard method to generate random numbers (as used in most BASIC interpreters) is not suitable for this purpose. Since some of the bare-bone computers do not even have the software to perform multiplication, it is asking too much for them to generate floating-point random numbers. To make these small computers more entertaining, a simple method to shuffle cards is described here. This method is implemented in a KIM. The machine instructions use about 80 bytes. There is lots of memory left for playing card games. The only drawback is that it requires the operator to press the interrupt key in order to stop the program.

The card shuffling program consists of two portions. The second portion is the main program that shuffles cards. It just keeps on shuffling until the interrupt key is pressed. The first portion is an interrupt service routine used to ensure an orderly ending of the program. The program is relocatable, and the two portions can be in separate locations.

This feature makes it easy to incorporate the shuffling program into a complete card-playing program. However, it is important that the user initialize the interrupt vectors to jump to the interrupt service routine.

To keep the computer code relocatable, the initialization of the 2 byte address is left to the user. The storage area for the cards, together with 4 bytes of working space, are in page 0. In this program, the storage area starts at address 0001. However, the program can be changed easily to move the storage area to other locations in page 0.

The deck of cards is stored in an array at locations (hex) 0001 to 0034. The value of

```
0120: 0200                ORG   $0200
0130: 0200 A2 36          LDXIM $36
0140: 0202 8A       L1    TXA
0150: 0203 95 00          STAZX $00
0160: 0205 CA             DEX
0170: 0206 D0 FA          BNE   L1
0180: 0208 86 38          STXZ  $38
0190: 020A A5 35    LOOP  LDAZ  $35
0200: 020C 38       L2    SEC
0210: 020D E9 34          SBCIM $34
0220: 020F B0 FB          BCS   L2
0230: 0211 18             CLC
0240: 0212 69 35          ADCIM $35
0250: 0214 AA             TAX
0260: 0215 85 35          STAZ  $35
0270: 0217 B5 00          LDAZX $00
0280: 0219 85 37          STAZ  $37
0290: 021B A5 36          LDAZ  $36
0300: 021D 0A             ASLA
0310: 021E 0A             ASLA
0320: 021F 18             CLC
0330: 0220 65 36          ADCZ  $36
0340: 0222 18             CLC
0350: 0223 69 01          ADCIM $01
0360: 0225 85 36          STAZ  $36
0370: 0227 18             CLC
0380: 0228 65 35          ADCZ  $35
0390: 022A 38       L3    SEC
0400: 022B E9 33          SBCIM $33
0410: 022D B0 FB          BCS   L3
0420: 022F 18             CLC
0430: 0230 69 34          ADCIM $34
0440: 0232 AA             TAX
0450: 0233 B4 00          LDYZX $00
0460: 0235 A5 37          LDAZ  $37
0470: 0237 95 00          STAZX $00
0480: 0239 A6 35          LDXZ  $35
0490: 023B 94 00          STYZX $00
0500: 023D A5 38          LDAZ  $38
0510: 023F C9 00          CMPIM $00
0520: 0241 F0 C7          BEQ   LOOP
```

```
0010:
0020:                    * INTERRUPT SERVICE ROUTINE
0030:
0040: 0243 A5 F3          LDAZ  $F3
0050: 0245 A4 F4          LDYZ  $F4
0060: 0247 A6 F5          LDXZ  $F5
0070: 0249 E6 38          INCZ  $38
0080: 024B 40             RTI
```

each address is distinct and is between hex 1 to 34 (decimal 1 to 52). After the interrupt key is pressed, the content of these addresses represents a deck of random cards.

The program uses a simple random number generator to generate random pointers with values between 1 and 52. The first card in the deck is interchanged with the card selected by the random pointer. The position of all the cards is next shifted one place so that the last card becomes the first, the first card becomes the second, and so on. This is to make sure that the first card is always changing, and a different card is interchanged with each randomly selected card. A random pointer is again generated and the whole operation is repeated.

After a sufficient number of operations, the deck is suitable for card games. One or two hundred shufflings are sufficient.

When the interrupt key is pressed, the interrupt service routine sets a memory location, hex 0038, that serves as a flag to signal the end of the shuffling. This routine also restores the accumulator and the X and Y registers. It is important that the user initialize the interrupt vector to address the service routine instead of the operating system.

The sequence of cards being shuffled is actually predetermined because it is calculated from a prescribed series of operations. However, if the stop command is activated by a human operator the cards can be very random. It takes about 10^{-4} second to do one shuffle. The time to activate the stop command can easily vary by more than 0.1 second. Thus, the number of shufflings can be uncertain by about 1000, which is sufficient to generate a deck of random cards.