(Approx. 959 words)

KIM-1

By Dick Maybach, Brookdale Computer User Group

www.bcug.com

n2nd (at) att.net

The KIM-1 (Keyboard Input Monitor), <https://en.wikipedia.org/wiki/KIM-1>, was a single-board "computer" launched in 1976 based on the MOS Technology 6502 processor. The company introduced the processor as the 6501, a pin-compatible clone of the Motorola 6800. When Motorola objected, the 6501 was replaced by the 6502, which was identical except that the pins were scrambled. Developers could experiment with the 6501 by plugging into boards designed for the Motorola processor, but the 6502 didn't allow this. So the KIM-1 was developed to provide a 6502-test environment for engineers and enthusiasts.

Calling the KIM-1 a computer was generous and misleading by today's standards. Figure 1 shows an early advertisement.

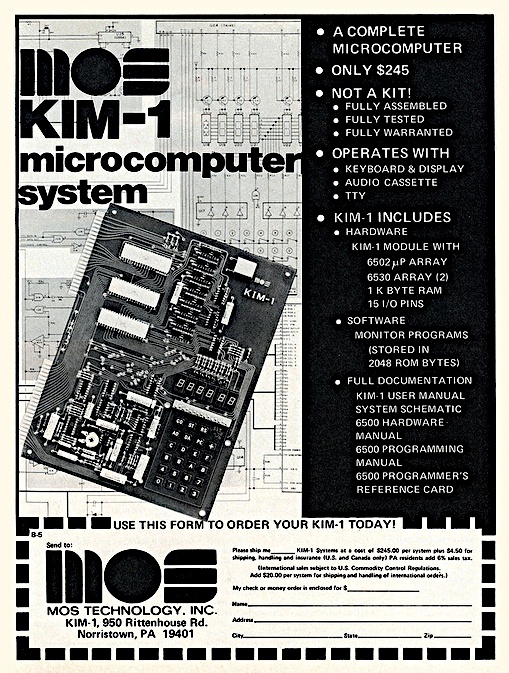


Figure 1. Introductory KIM-1 Advertisement.

The $245 price (equivalent to about $1200 today) included only the single board. You needed two power supplies, plus a terminal and permanent storage, to have something close to a (quite primitive) computer. More capable and expensive personal computers, such as the Altair 8800, were available, and the Radio Shack TRS-80 would appear a year later. However, the KIM-1 provided a relatively inexpensive computer technology introduction for those willing to get their hands dirty.

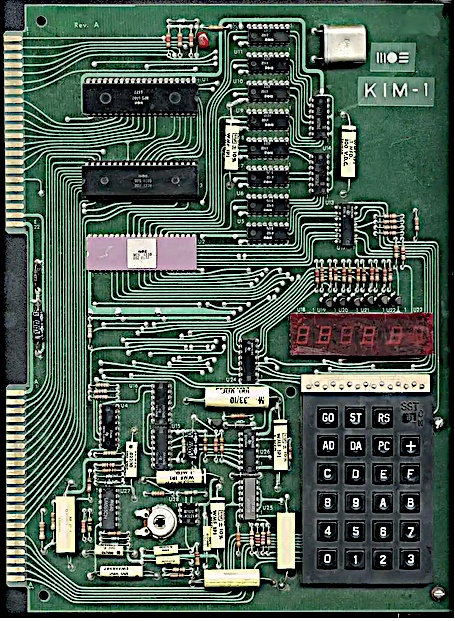


Figure 2. KIM-1 Circuit Board.

Figure 2 shows 24 keys for input, a six-digit display for output, and two edge connectors for expansion. Unfortunately, the board was shipped with a connector for the lower set of pins of Figure 2, to which you connected a 5-Volt and a 12-Volt regulated power supply, which were not included. The resulting computer was limited indeed, with only the keypad for input and the numerical display for output.

|  |  |
| --- | --- |
| Processor | MOS Technology 6502 |
| Clock Rate | 1 MHz |
| Word size | 8 bits |
| RAM | 1024 bytes |
| Peripherals | Keyboard port, display port, Teletype port, audio tape port, 15 I/O pins |

The unit was shipped with a users manual (<https://archive.org/details/KIM-1_Users_Manual/mode/2up>), hardware, and programming manual.

Nevertheless, you could write programs using hex-digit machine language, as there was no assembler, let alone a compiler. Since there was no permanent storage, your work was destroyed when you turned off the unit. Most users quickly tired of coping with the limitations of the KIM-1 and expanded it with one or more of the following:

* a cassette tape drive for data storage,
* a terminal, usually built from a kit,
* an additional 4 Kbytes of RAM, which allowed running Tiny Basic.

Each bit was encoded as a group of three audio tones for cassette tape storage, each lasting about 2.5 msec. The data rate was about 134 bits/second, but the high error rate made data transfer tedious. Nevertheless, it was popular despite its shortcomings because it was far cheaper than a paper-tape punch and reader. Moreover, later developments significantly improved this technique's speed and accuracy.

Commercial terminals were quite expensive, but users could assemble kits, such as the TV Typewriter, <https://en.wikipedia.org/wiki/TV_Typewriter>, shown in Figure 3. This had a 55-key keyboard and could display 16 text lines of 32 characters each on a small TV. The user-supplied the case. Those few hobbyists who could afford a Teletype could enjoy its built-in paper-tape reader, punch, keyboard, and printer.



Figure 3. TV Typewriter.

Tiny Basic (<https://en.wikipedia.org/wiki/Tiny_BASIC>) required perseverance, as loading it using an audio recorder was a 15-minute ordeal, but you could now save your work on cassettes. Tiny BASIC was a popular, small (4K of ROM), open source interpreted language. (At the time, the cost of a commercial version of BASIC was comparable to that of a KIM-1, with the result that some hobbyists used pirated copies.) Tiny BASIC made many compromises to achieve its small footprint; most versions lacked string variables and floating-point math and allowed only single-letter variable names. It was a family of related languages with various features and various processors.

Many hobbyists chose to program their KIM-1s in machine code, where each instruction was a two-digit hex number, which was tedious. You first wrote the program in assembly language, which used three-letter mnemonics for the operations, and then you converted these to two-digit hex machine instructions. (Professional programmers at the time used an assembler program to make the conversions.) See The First Book of KIM, <https://archive.org/details/The_First_Book_of_KIM/mode/2up>, for some examples. A more thorough programming introduction is in the 6502 Programming Manual, <http://retro.hansotten.nl/uploads/files/MCS6500%20Programming%20Manual.pdf>, but it assumes you have access to an assembler. Figure 4 shows a simple program from The First Book of KIM that reads a key on the KIM-1 keyboard and displays its value on the display.

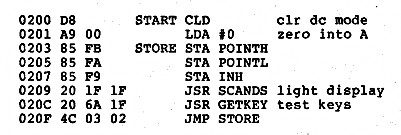


Figure 4. Sample Program.

The left column shows the four-digit hex address addresses, and the second column is the two-digit hex instruction. Next are (zero, one, or two) two-digit hex addresses. Depending on the instruction length, the address increases by one, two, or three digits. The next columns assume you have an assembler. A home programmer would write the information on paper, then convert it manually to that shown on the left. You can probably see why Tiny BASIC was popular.

KIM-1 disappeared quickly as far more usable computers, such as the Commodore PET, the Radio Shack TRS-80, and the Apple II, became available for hobbyists. Today, the devices closest to the KIM-1 are the Arduino, <https://www.arduino.cc/>, and Raspberry Pi, <https://www.raspberrypi.org/>, families. Both are far more powerful and easier to use, but if you have a masochistic urge to return to the computing dark ages, you can recreate the KIM-1 with a kit based on an Arduino, <https://create.arduino.cc/projecthub/obsolescence/kim-uno-a-6502-kim-1-computer-on-arduino-e5c82c>. As shown in Figure 5, the result is physically different, although it performs the same as the original.



Figure 5. KIM-1 Simulated Using an Arduino.

Before spending much time with this, you would undoubtedly want to devise a way of labeling the keys. You can experience programming the 6502 without building anything, as several online emulators are available, <http://www.6502.org/tools/emu/>.