Kaypro Emergency Monitor ROM
May 15, 2023

Table of Contents

Introduction

The purpose of the monitor is to help debugging a broken Kaypro. Replace the normal boot/BIOS ROM with a ROM created from this monitor image. The Kaypro will now interact using the serial port (“Serial I/O” on early models, “Serial Data” on later models). No other hardware will be initialized, which means the display and keyboard will not work, nor will the disk. The display will usually be filled with random characters, or at least will not be cleared.

The ROM Image

The monitor binary image may be downloaded here: http://sebhc.durgadas.com/kaypro/monitor.bin. Note that this is a 2K image, the same size as the 2716 EPROMs used for the Kaypro II and Kaypro IV (2/83, 4/83) models. It may also be put in larger EPROMs (2732 or 2764) for use with the Kaypro 10 and */84 models. When using larger EPROMs, it should not be necessary to replicate the image throughout the extra space (this needs to be confirmed). Note that the Kaypro IV/83 model (may also apply to some later Kaypro II) has a jumper-selectable ROM socket that takes either 2716 or 2732. Be certain to set the jumpers appropriate for whatever EPROM is being installed. The Kaypro 10/83 has a socket wired exclusively for the 2732. The */84 models should work with either 2732 or 2764 without changing any configuration.

Note that the AT28C16 EEPROM (FLASH) should work in place of the 2716 in Kaypro circuits. The AT28C64 should work in place of the 2764.

Using The Monitor

The monitor prompt is the colon character (‘:’). When the system is RESET the monitor will initialize the serial port and print a signon string and then prompt for input. Input lines may be edited using the Backspace key. Ctrl-C will abort the input and go back to the prompt. Upon pressing RETURN, the input line is then parsed by the monitor.

Note that while the ROM is enabled by the sysport, Low memory address contain the ROM contents and are not writeable. The exact boundary between ROM and RAM depends on the model. Also, the Kaypro II and IV models map the video RAM into addresses 3000-3BFF. In order to access the RAM at low addresses, a special program must be used to transfer control into high memory and update the sysport to turn off ROM (and video RAM).

The command ‘?’ will print a brief list of commands with some help text.

The following commands are implemented:

D *start end*

Dump memory in hex. Both *start* and *end* are required, and are interpreted as hexadecimal. Up to 16 bytes are displayed per line, with ASCII representation following the hexadecimal byte values.

S *start*

Substitute memory content interactively. The current address (initially *start*) and byte are displayed in hex and a single character of input is accepted. Pressing RETURN will go to the next address (+1) without changing the current byte. Pressing dash ('-') will go to the previous (-1) address without changing the current byte. Pressing period ('.') will end the command and return to the monitor prompt. Entering hex digits and RETURN will replace the current byte with the new value and advance to the next address (+1).

G *start*

Go (jump) to *start* and begin executing the code there. If the code executes any RST instruction, control will be returned to the monitor, where the message "*** RST" and the calling address (after the RST instruction) will be printed. No registers are saved and the code may not be resumed.

F *start end data*

Fill memory with the byte *data* repeated throughout the range.

M *start end dest*

Move a block of memory. If *start end* overlaps with *dest* then the results are undefined.

I *port* **[** *num* **]**

Input from *port* xxx and display the value in hex. If *num* is given, then the same port is read that number of times, and each value displayed. Both *port* and *num* are hexadecimal. There is no delay between inputs, other than the time it takes to print the value in hex.

O *port data* **[** ... **]**

Output *data* to *port*. If more than one data bytes are given, then those values are successively output to *port*. There is no delay between outputs, other than the time it takes to parse the next value.

V

Display the ROM version. This just prints the ROM signon message again, which contains the version.