MITS Programming System II with Altair Floppy Support

The MITS Programming System II allows development of 8080 assembly language applications on the Altair 8800 using paper tape or cassette for mass storage. The package consists of an editor, assembler, debugger and a monitor that allows execution of these programs as well as the programs you may develop.

In this folder are the files and a disk image to add support for the Altair floppy drive to the Programming System. This dramatically improves performance as detailed in a few videos I have posted on YouTube. Perform a search in YouTube for "Programming System II Floppy" to find the video for the Altair drive. The description below the video includes links to other videos that provide additional detail about the process of adding floppy support to the Programming System.

To begin, familiarize yourself with Programming System II by watching the recommended videos and by looking at the related documentation found at the links below:

https://deramp.com/downloads/altair/software/manuals/Programming%20System%20II.pdf

https://deramp.com/downloads/altair/software/papertape_cassette/MITS%20Programming%20System%20II/ReadMe.pdf

Floppy support was added to the Programming System by providing the ability to load and save a 24K "workspace." The workspace includes the Programming System monitor, the editor, the assembler, and typically, your source code. A workspace can also include the debugger instead of the editor (they are at the same address). The workspace also captures the first 24K of BASIC in memory, so as demonstrated, early paper tape/cassette based versions of BASIC like 4K, 8K, and Extended BASIC (along with a program) can also be saved and loaded with the disk code.

Typically, the disk code is placed in EPROM so that it is available at cold start. I have located the code at F000h so that monitors like ALTMON and AMON can remain at F800h. This also allows running the EPROM code from RAM by using a hex file loader to load the code into RAM at F000h instead of EPROM.

Creating a Disk

Use PC2FLOP from this same directory to write the disk image "PS2DEMO.DSK" to a floppy disk. You can load the file PC2FLOP.HEX using an Intel hex file loader or monitor like ALTMON or AMON. Once loaded, execute PC2FLOP by jumping to 0100h. Alternatively, Altair Clone users can use the Configuration Monitor to rapidly upload the disk image.

Booting a Disk

Burn the code PS2PROM.HEX or PS2PROM.BIN into an EPROM at F000h. Or, use a hex file loader or monitor to load the file PS2PROM.HEX into RAM at F000h. Jump to F000h using the front panel or your monitor's jump command to boot the disk. The boot code will prompt for a workspace. Type the digit of the workspace you want to load (e.g., zero).

The following workspaces are present on the demo disk image:

Workspace	Content
0	Floppy disk routines
1	EPROM code mover
2	8K BASIC and "Chase"
3	DBG and AM2
4 – C	EDT and AM2, empty source file

Memory Allocation

As detailed in the document links above and as mentioned in the Programming System II videos, it's up to you to provide sufficient room for the assembler symbol table, to define the location of the edit buffer, to determine a location for your executable code, and make sure none of these clobber each other or the monitor, editor, or assembler. For the code I have written, I have used the memory allocation shown below. This is the allocation in all workspaces on the demo disk.

Item	Start (hex)	Start (octal)	
Monitor	40	100	
EDT or DBG	0A40	5100	
(not used)	11B2	10662	414 bytes not used
AM2	1350	11520	
SymTab Start	1E06	17006	Leaves 1530 bytes (5FAh) for symbol table
Edit Buf Start	2400	22000	
Edit Buf End	5000		Provides a 11264 (2C00h) byte edit buffer
Reserved for Disk Code	7C00	76000	Top 1K of RAM (e.g., 32K, 7FFFh)

As shown in the table above, the disk routines use the last 1K of RAM. The demo code is configured for a system with 32K of RAM. This location can easily changed in the source files to allow for more or less RAM.

Programming System Changes/Patches

The Programming System workspaces have a few patches applied to customize the environment to my preferences. See the file PATCHES.ASM for more information.