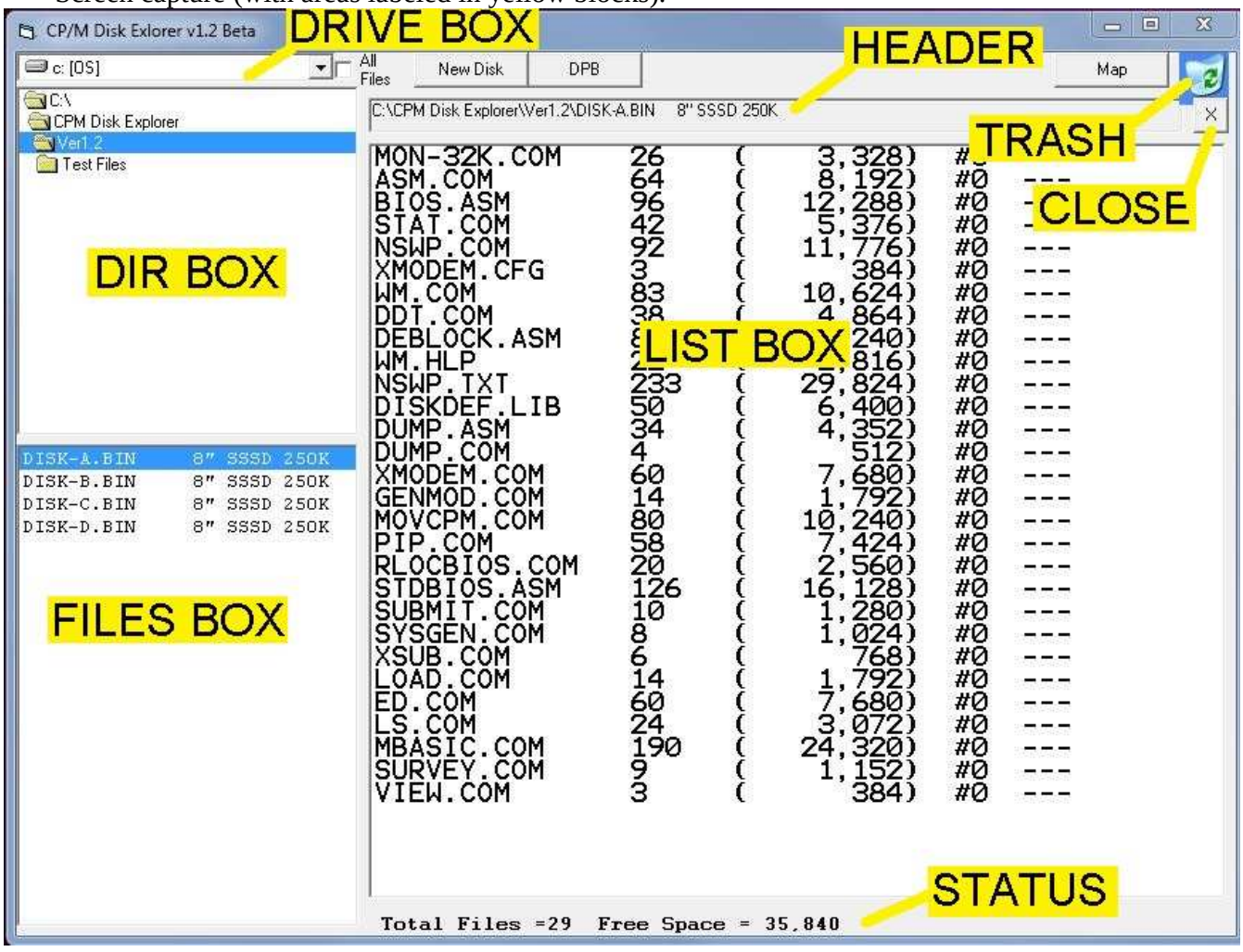
CP/M Disk Explorer V1.2
April 16, 2017

To allow easy distribution, the file extension was changed.
Please rename the .XEX file to .EXE

Here is a utility that will allow you to copy files into and out of a virtual CP/M disk file.
Warning: Back up your files before using this utility.

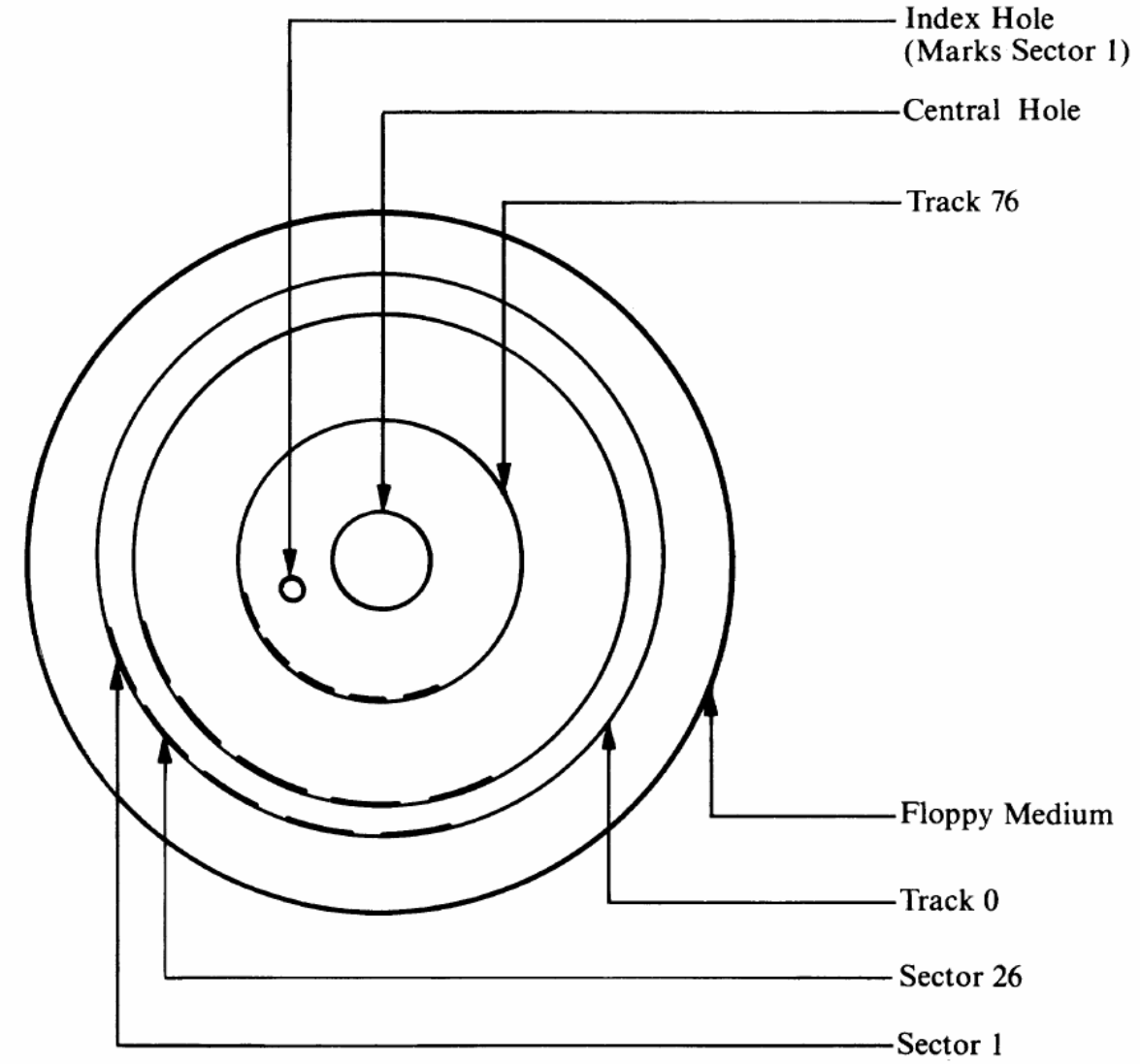
Screen capture (with areas labeled in yellow blocks).



What is a virtual CP/M DISK file

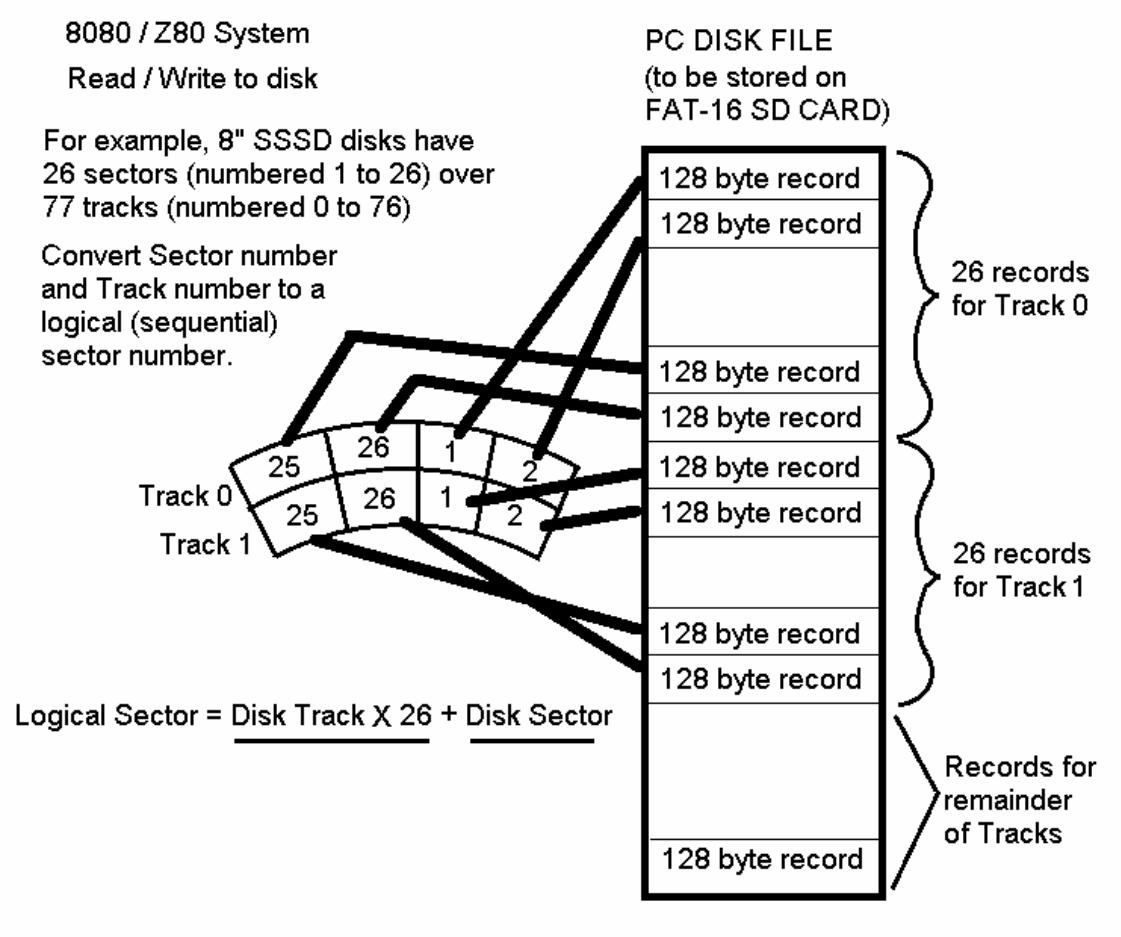
The CP/M disk file is a large file that will act as an entire floppy disk for a CP/M system. The Z80Pack uses such files as does the JAIR 8080 (my ALTAIR / IMSAI 8080 CPU replacement board) and Lee Hart's Z80MC. JAIR and Z80MC are real hardware systems which accesses data in these files and pass the data to CP/M as if they were real disks.

The example disk, an 8" SSSD, uses 26 sectors and 77 tracks. This multiplies to 2,002 sectors of 128 bytes each or a total of 256,256 bytes of raw formatted data. This is the IBM 3740 format used on early 8" floppy disks.



Picture from **The Programmers CP/M Handbook** by Andy Johnson-Laird.
(Used without permission).

The sectors are reorganized as sequential records of 128 bytes in a random access file stored on a modern SD Card formatted as FAT-16. Of course this file can be copied to any files system that Microsoft Windows can access. The CPM Disk Explorer program will access the records through Windows.



Of course CP/M has its own encoding method for storing the operating system and data files on the disk. See Fig 3-8, also from **The Programmers CP/M Handbook** (which I will say is a very well written and detailed book). The CPM-Disk-Explorer program will use those methods to access the CP/M files within these Virtual Disk images. This utility can create new blank disk images, copy CP/M Files into and out of those CP/M virtual disks and delete CP/M Files.

The first reserved tracks are for the operating system and are not accessed nor are they of interest by this program, with the exception of changing the skew factor.

Disk Types

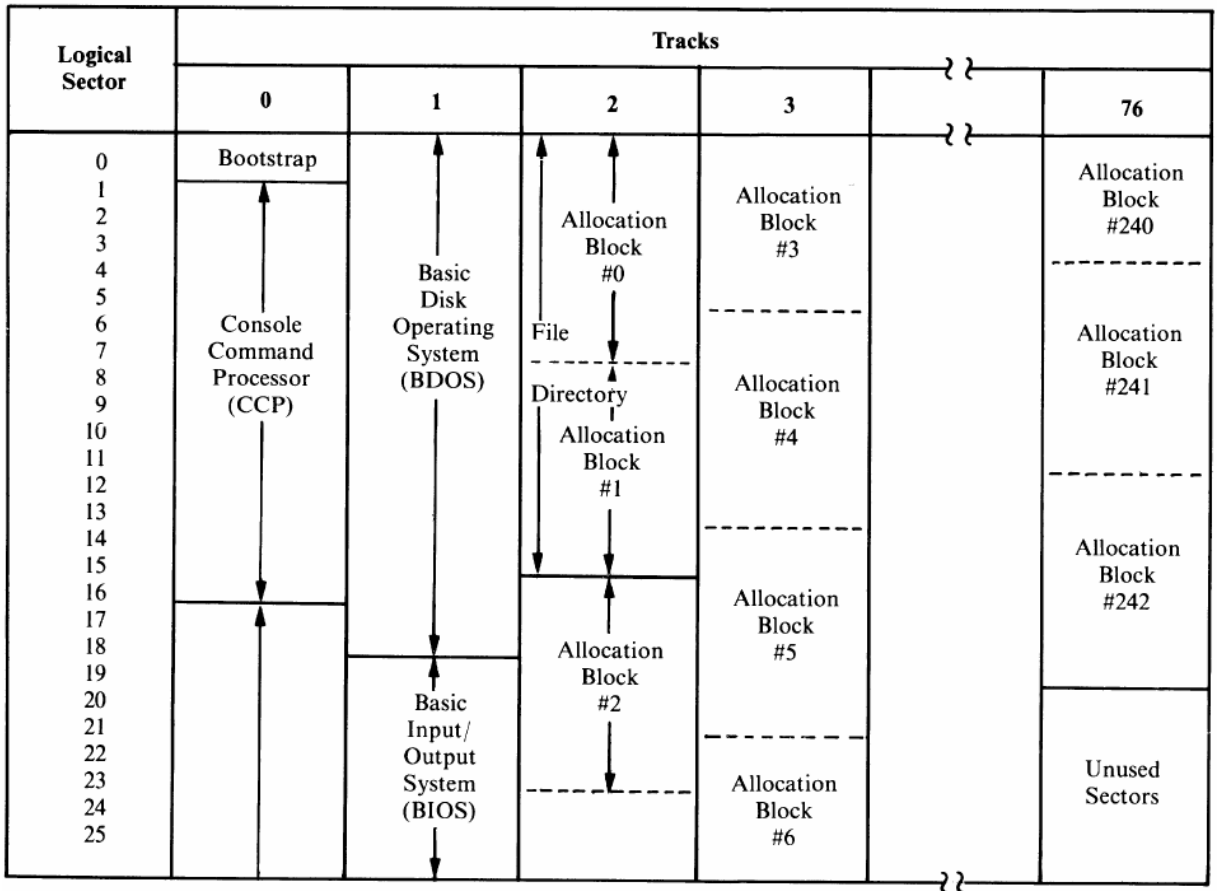


Figure 3-8. Layout of standard diskette

On these 8" disks, the first 2 tracks hold the CP/M Operating system (CCP + BDOS + BIOS). The remainder of the disk is used for file storage of CP/M Programs and data. The sectors are grouped into allocation blocks. Each block uses 8 sectors. There are other (larger) CP/M disk formats which use 16, 32, 64 and up to 128 sectors per block. The type (and size) of disk is defined by the **Drive Parameter Block (DPB)**. The DPB is buried in the BIOS section. **CP/M Disk Explorer** will open any type & size of disk. These types are all defined in a configuration file named "**CPMDISKS.TXT**". This is a text file that is used by the JAIR 8080 and Z80 Membership Card to automatically adjust the BIOS to the correct file type. There is more information about this file in the "**New Disk**" section.

How to use the program

Use **DRIVE BOX** and **DIR BOX** to select a working drive and folder containing the desired files. These files will be listed in the **FILES BOX.** You may either open an existing *"Disk File"* or create a **New Disk**. New disks created will not automatically open. The **LIST BOX** will populate with all the CP/M files stored within the CP/M virtual disk. Similar to a ZIP file concept, only the files are not compressed, but instead stored in the CP/M disk format. **HEADER** displays the CP/M Disk File opened. **STATUS** displays total # of files and Free Space available.

The **FILES BOX** will only display CP/M Disk Files of known types (as listed in the "CPMDISKS.TXT" configuration file). To display ***ALL*** files, click the check box next to the **DRIVE BOX** labeled **"All Files"**. This feature allows you to easily find a CP/M Disk files. Also, CP/M Disk files will have their description type listed, while all other files will have the size of the file listed. Cycle checking (check then uncheck) "**All Files**" is an easy way to refresh (fetch & relist) the **FILES BOX**.

It should be noted that the system identifies a file as a CP/M Disk File by only its size. A file of size 256,256 bytes will be labeled and assumed to be a CP/M Disk File (this is the previously described 8" SSSD disk). If a CP/M file of this exact size (or any listed size) is used, then it can still be copied into and out of (larger) CP/M disk images. It will always simply be listed with the interpreted label, this is a display thing and does not affect the file or its behavior during copying in any way. The program will allow it to be opened like a disk image file, but the data within will not make sense and cause gibberish. Just close the file and no harm done.

Double click a file in the **FILES BOX** or Drag and Drop any *"Disk File"* to the **LIST BOX** to open it. Drag and dropping a file from the **FILES BOX** to the **LIST BOX** will do one of the following two things:

1. If a *Disk File* is NOT open then this action will open the *Disk File* dragged.
2. If a *Disk File* is open then this action will copy the file into the *CP/M Disk File*.

Copy Files To *"Disk Image"*
Drag and Drop any file from the **FILES BOX** to the **LIST BOX**

Copy Files From *"Disk Image"*
Drag and Drop any file from the **LIST BOX** to the **FILES BOX**

Delete Files
Either Windows files or CP/M files with a *"Disk File"* can be deleted. Drag and Drop any file from either the **FILES BOX** (Windows files) or **FILES BOX** (CP/M files) to the **TRASH** icon.

Use the **CLOSE** to finish viewing the *"Disk File"*. Note: Changes to the CP/M Disk file happen immediately upon any files copied to the disk or files deleted from the disk. There is no "Close without saving changes" option.

LIST BOX

All the files within the CP/M Disk are listed here. The size of each file in sectors (1 sector = 128 bytes) and bytes are displayed. Note: CP/M does record the exact size of a file to the byte, but only the number of sectors it occupies. A file that needs to be 129 bytes will have its size reported as 2 sectors and 256 bytes.

Number in parenthesis are the bytes (# of sectors times 128). The number after the hash symbol is the User Number, which can be from 0 to 15. The next 3 characters are the file attributes (a dash indicates not set, a letter indicates set)

- **R** = Read Only.
- **A** = Archive (ie file has been changed since last copy).
- **S** = System.

The User number is an interesting way that CP/M was able to partition the files to various users. A user #5 would only see files that are #5's. In CP/M, the command to change your user number is:

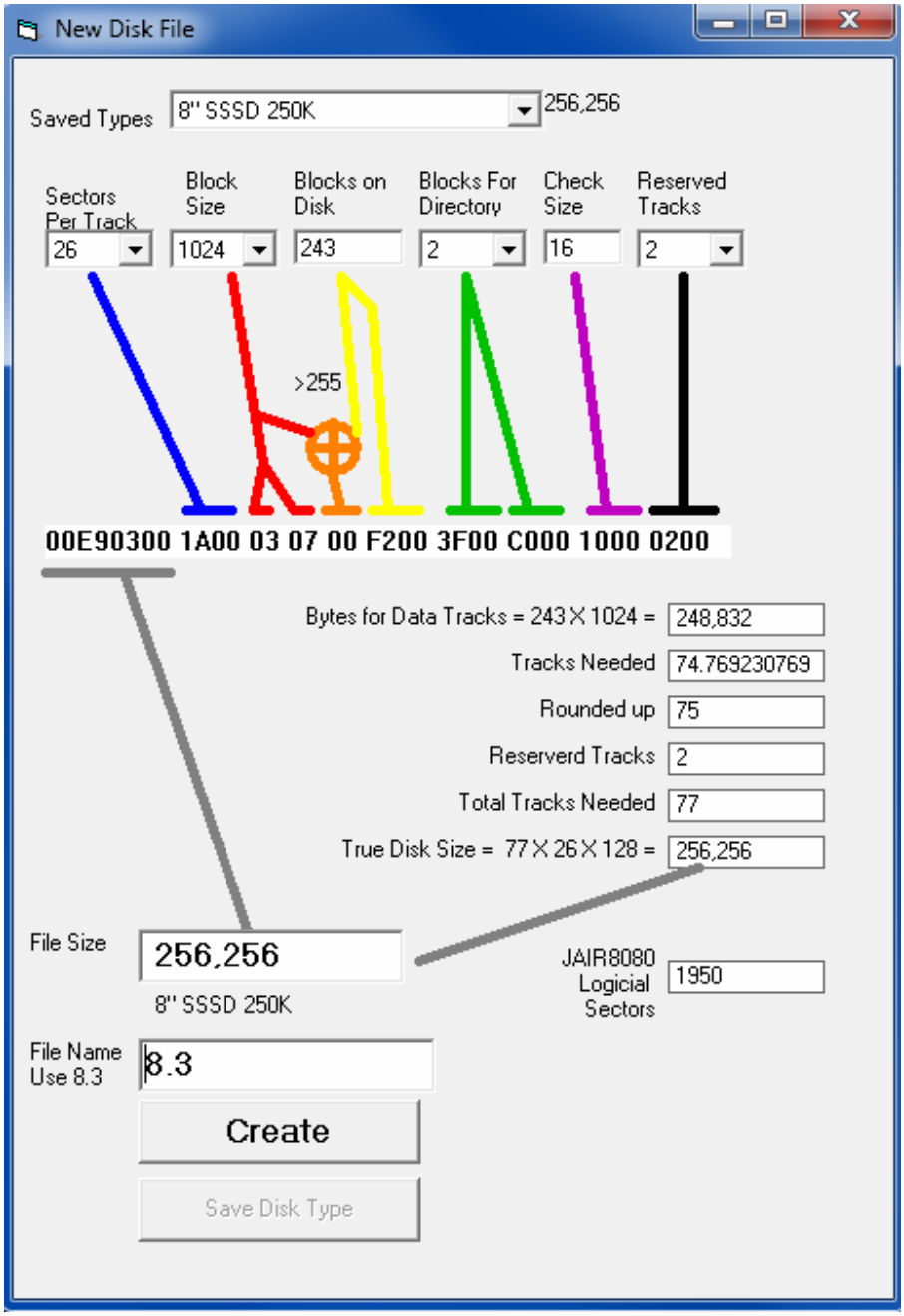
a> **USER 5**

The use and behavior of the file attributes goes beyond the scope of this manual.

CP/M Disk Explorer is not programed with any feature to modify these values directly. However, when writing files to a disk image, these values can be set in the **DPB** screen.

New Disk

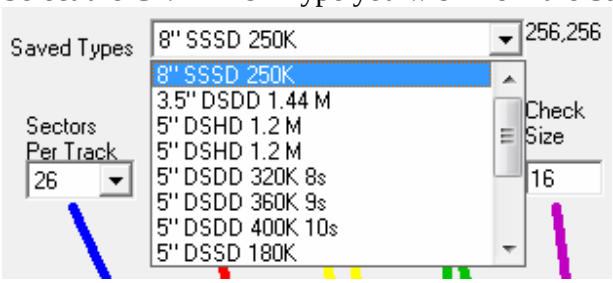
In this screen, new CP/M Disk image files of any type can be created. Also, you can define new types of CP/M Disks.



Creating CP/M Disk Files

To simply create a new CP/M Disk file follow these steps:

1. Select the CP/M Disk Type you wish from the **Saved Types** drop list.



2. Enter the Windows **File Name** for this CP/M Disk file. Use 8.3 format

3. Click **Create**. If file already exists, you will be prompted to overwrite or cancel.

4. Close the New Disk window.

A new file will just be a blank file of the correct size filled with the blank formated character 0xE5. The file should appear in the FILES BOX and can now be drag and dropped into the LIST BOX. If the file does not appear, you can cycle check “ALL Files” to refresh the display.

The 8.3 file name format is the old DOS file nameing format. This is required for use with the SD Card and FAT-16 by the JAIR and Z80MC system. The format uses 1 to 8 valid characters for the file name optionally followed by a period (pronounced dot) and 0 to 3 valid characters for the extension. Valid characters are any numbers, letters and these special characters:

! # $ % & ' ( ) - @ ^ _ ` { } ~

More information is available at
https://en.wikipedia.org/wiki/8.3_filename

CP/M Disk Types

CP/M 2.0 and higher allows the use of various types and sizes of disks. The 8" SSSD (Single Sided & Single Density) disk was improved by better hardware like:

- Double sided disks.
- Double or Quad density media.
- More Sectors.
- More Tracks.

The Number of sides was never tracked by CP/M, so the BIOS would handle it by doubling the number of sectors per track, then selecting the correct side and adjusting the sector before accessing the physical drive.

These paramenters are pretty much useless for a virtual CP/M Disk file since all the sectors are just stored in a random access file. All that really matters is the final disk size which is the key to identifying the correct Disk Type.

CP/M uses 9 values to define how it will interpret and use the disk information. There is a whole chapter that neatly describes all these in **The Programmers CP/M Handbook** by Andy Johnson-Laird. We can simplify these 9 to just 6 key values. The colourful lines on this screen show how the nine are derived. "The nine" sounds like something from Lord of the Rings! There is a 10th value on this line, the first value, which is the size of the file. Note: Some of these values are store in hex pairs as little endian and others as big endian. Do not concern yourself over that, I have already lost enough hair to get it right. The nine (or 10?) can be seen in the middle of this screen and will be saved to the CPMDISKS.TXT file if you play your cards right.

Selecting different Saved Types will show how the paramenters change for various possible disk types. There are no standard lists of CP/M disk types, this changed depending on the Computer or Disk Controller card manufacturers. Some of the predefined values are guestimates and will require further research and corrections. Let's backup 1 step to the CPMDISKS.TXT file. This is where those saved types are all stored.

CPMDISKS.TXT

The JAIR 8080 and Z80 Membership card systems will use this file to set the correct Drive Parameter Blocks in BIOS for CP/M to successfully read the Disk Images. The first 4 lines hold the file names to be opened as the default drives for CP/M Disks A to D. These file names are 8.3 but must be padded with spaces and exclude the dot. The first character will be ignored but must not be a valid hexadecimal character. The line must be at least 12 characters wide, characters after the 12th will just be ignored.

```
>DISK-A  BIN
>DISK-B  BIN
>DISK-C  BIN
>DISK-D  BIN
```

The lines that follow will list all the CP/M Disk Types, starting with a file size and ending with a description. Any line that does not create a valid hexadecimal size will be ignored and can be used to comment the file.

00E90300 1A00030700F2003F00C00010000200 = 8" SSSD 250K

The first line is of course the original 8” SSSD disk type. The size is 00E90300, which is little endian talk for 0x0003E900. Plug that into your hexadecimal calculator and it reveals the magic number 256,256. As seen earlier, this is the number of bytes required to store all the sectors of an 8” Disk (26 sectors X 77 Tracks).

If the System Boot Loader encounters a file of this size, it displays the type on the menu screen and loads the DPB table with the correct nine values. CP/M loads correctly and everybody is happy.

The Six

1. **Sectors Per Track**. From 8 to 255, this value has a small effect on the file size of the image disk file, but does not have any effect on the amount of data that can be stored. It’s like changing one side of a rectangle who’s area remains constant by adjusting the adjacent side to grow or shrink.
2. **Block Size**. This selects the number of sectors per block and is expressed in the size of the block as 1K (8 sectors per block) to 16K (128 sectors per block).
3. **Blocks on Disk**. This value is the most significant figure that determines the disk size. It will be multiplied by the Block Size to determine the number of bytes that can be stored.
4. **Blocks for Directory**. From 1 to 16, this is the number of blocks used to hold the directory entries. There are 4 directory entries per sector, 64 in a 2K sector. But files may require multiple directory entries, in fact, they will need 1 entry for every 8 (or 16) blocks the file occupies. Which is it 8 or 16? Well, if the total

number of blocks on a disk is less than 256, then any block number can fit into a single byte and there will be 16 blocks defined within the 16 bytes of 1 directory entry's allocation list. If the total number of blocks is greater than 255, then a block number will require 2 bytes and that makes for only storing 8 in the allocation list.

5. **Check Size**. This is not well enough documented to understand it's function within CP/M.
6. **Reserved Tracks**. This is the number of tracks that will be skipped over to access the data on the disk. These tracks were set aside to hold the CP/M operating system. This number can be reduced to 0 on data only disks.

Creating the largest disk possible.

The JAIR and Z80 internal system of converting sectors & tracks to sequential sectors has a limitation of 16 bits for the sequential sector. This limits a disk to be no more than 65,536 sectors. This amounts to a disk of 8Meg in size. It is suspected that CP/M also holds this limitation as it's known to have the 8Meg limit. In creating the largest disk, the Sectors per Track needs to be set to a power of 2. I suggest 64, just because it reminds me of the Beatles song. Block Size does not play a big role, it only needs to be 2K to allow larger than 256 blocks to be defined. Defining 4096 blocks will result in an 8Meg disk. The Reserved tracks need to be reduced to 0 to keep the whole disk to the 65,536 sector limit. In defining the Blocks for Directory, it is best to max this number. The exact math is based on the number of blocks that can be recorded as follows:

Total blocks divided blocks per Directory Entry divided by Directory Entries per block.

4096 / 8 / 64 = 8 Blocks of Directory entries required for a full disk.

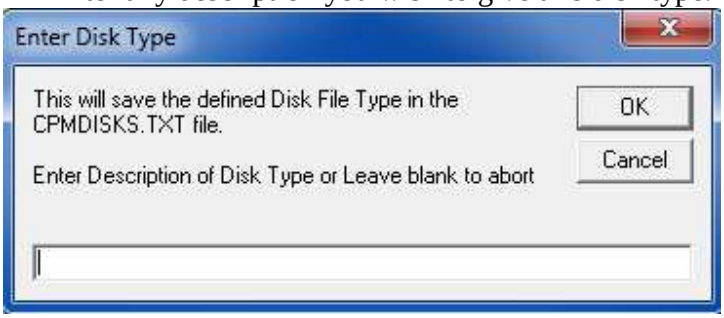
But this assumes that each file will use an exact multiple of 8 blocks. As this is not the case in the real world, the full 16 blocks are selected.

Save Disk Type

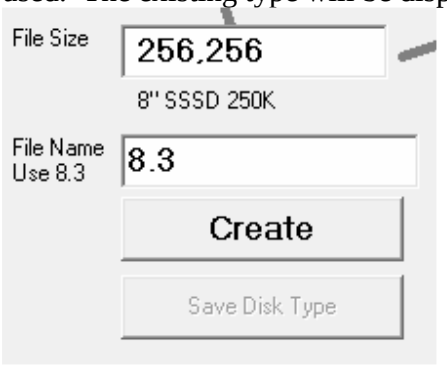
After creating the new disk type by altering the 6 parameters, the disk type can be saved to the **CPMDISKS.TXT** file. Be sure to copy this updated configuration file to the SD Card in your JAIR or Z80MC, so they will recognize the new disk type(s).

- Click "Save Disk Type"

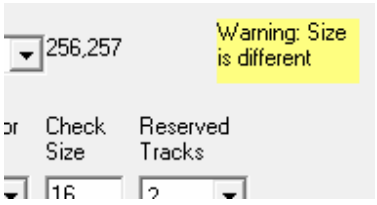
- Enter any description you wish to give this disk type.



If the "Save Disk Type" button is greyed out, then it's because there is another disk type that produces the exact same file size. Recall that the file size defines which type will be used. The existing type will be displayed right under the **File Size** box.

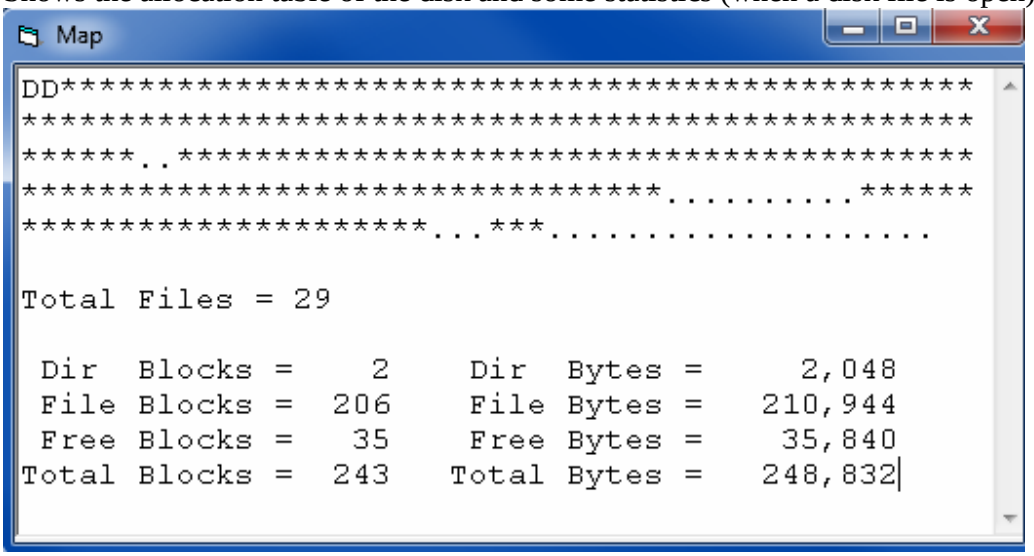


It is not necessary to use the exact size given by the calculations. A new size can be manually entered. You must of course make the new size larger than the calculated size (or else the systems will surely crash). Just make the size a unique number and then this type can be saved. Files created under this type will have those extra bytes at the end and that will suffice to identify them correctly. Please ignore the "Warning: Size is different" message.



MAP

Shows the allocation table of the disk and some statistics (when a disk file is open).



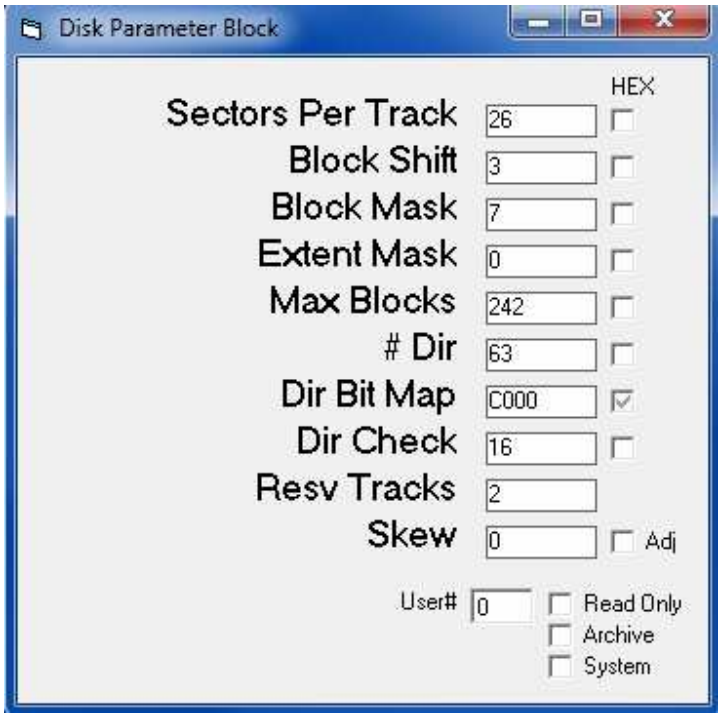
The D's represent allocation blocks used by the Directory. Asterix (*) are used blocks. Periods are free blocks. Counts are given below in blocks (left) and bytes (right).

DPB

Allows you to view current Drive Parameters Block in CP/M (with Skew factor). HEX changes view of field.

Skew and Adj will be discussed in Disk Skewing section.

User, Read Only, Archive & System are attributes given to files as they are copied into the CP/M Disk Image. Set these values before copying the file.

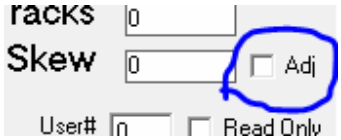


Disk Skewing

This is a technique to improve performance. Recall the image of the round disk. Imagine reading sector 1 as the disk is spinning, then by the time that data is processed, the disk is way past sector 2, now the program waits 16mSec until sector 2 comes around again. What disk skewing does, is it staggers the sectors by a count factor so when processing is done, the next sector is much closer. Of course, processing time may vary so finding a good Skew Factor can be a process of hit and miss or a lot of calculations. A value that sets the next sector to within a few sectors reach most of the time is ideal. Too big a number and it adds to the wasted time over all, too low and there may be too many occasions where the processing time exceeds that skew threshold and you literally "loose a turn". A common number of 6 was used for CP/M disks, but this is not a rule to live by. The Z80pack emulator disk images' uses a skew factor of 6. A skew factor of 0 is the same as 1 and will basically have no change on the order of the sectors (1, 2, 3, 4…). A skew factor of 2 will skip by 2's ie 1+2=3, 3+2=5, etc. This results in (1, 3, 5, 7…). A skew factor of 3 results in (1, 4, 7, 10 …). The JAIR and Z80MC system don't use any skew factor (ie 0 or 1). This is because the sectors must be read in sequence to prevent back peddling in the FAT table (which not only causes you to lose a turn, but cause a return to the start of the FAT table list).

If reading a file from Z80pack, or other sources that may be skewed, the program can let you adjust this on the fly to help decode the correct skew factor.

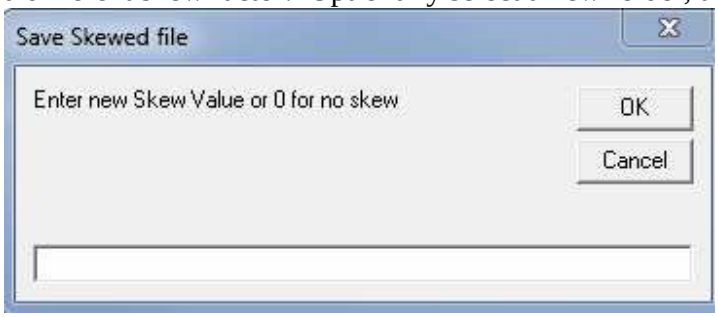
- First, enable the Skew Controls by clicking the "**Adj**" check box in the **DPB** screen.



These new controls will appear in the main screen:

- Open the file, observe the gibberish in the list of files.
- Click the right arrow to increase the skew factor and watch the file list and make the gibberish disappear. The more files in a disk image, the more certain you can be that the right skew factor was found. Skew factors like 6 can allow the gibberish to disappear on numbers like 2 and 3 (both are factors of 6). Further experiments in copying files out will be needed to know if the correct skew factor has been found.

- Once the correct skew factor is found, you can save the disk file to a new file with a different skew factor. Optionally select a new folder, then click "**Save Skewed**"



- The new file will have the same name but the extension will be appended with a dot followed by the new skew factor. Ie Disk-A.BIN.6
- Files converted for use with JAIR/Z80MC will need to be renamed to the 8.3 format. This can be done in Windows by removing the "Hide extensions for known types" or done in a DOS / CMD box.

You can also convert JAIR/Z80MC files to Z80Pack files by opening, leaving the skew at 0, then saving with a skew factor of 6.

If a file is repeatedly skewed with different numbers, it will become a tangled mess and you may say such a file is "skewed up".

Caution

Please back up your files before using this utility as there are no "undo" or "do not save" features. Files being added into or deleted from a disk image are done immediately.