Welcome to Vcc

The Color Computer 3 was the last of a line of micro-computers designed by, and distributed through Radio Shack stores. Released in 1986, it developed a rather large and loyal following that continues to this day. Some of the system's specs are as follows.
128K via a bank switch technique. A 512K upgrade was also available.
A maximum resolution of 640x225 at 4 color or 320 x 225 at 16 with a clock speed of 1.79 MHz. Twice its predecessors. Unfortunately in 1991 Tandy decided to discontinue the line.

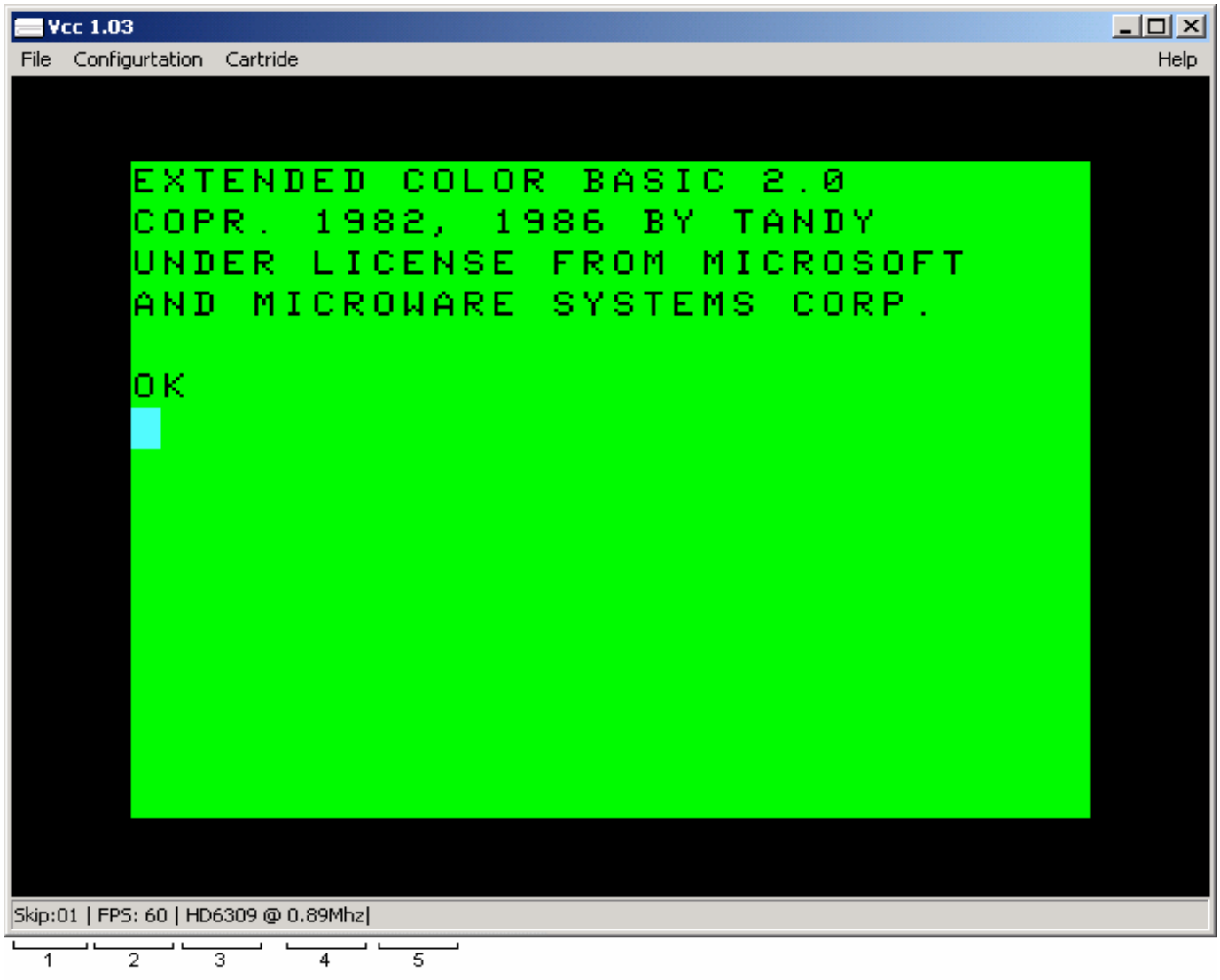
Strangely, lack of support didn't seem to deter the fans. Third party vendors stepped in to fill the void. New products were and are still being developed. Today you can buy 2 and 8 Meg Memory cards, IDE and SCSI hard disk interfaces and faster, more powerful CPUs. There is even a free user supported Multi-tasking Multi-User OS available for this poor little 2 decade old 8 bit machine.

Vcc is a Tandy © Color Computer 3 emulator that runs on the Windows© operating system. It attempts to simulate the hardware that comprised this system. As such it allows software written for this 20+ year old computer to run on modern hardware. Please take a moment to read this guide to discover some of the features and yes shortcomings of this emulator.

If you have ever used an emulator before you should have no trouble understanding most of Vcc's functions, however there are some differences that you should be aware of. Vcc not only attempts to emulate the hardware of the coco3 but also its organization. As such Vcc only emulates the coco3 with no peripherals. Expansion is possible via a system of run-time loadable plug-ins that emulate the various add-in cards that were/are available. Sort of an Emulator within an Emulator. See the section on Loadable Modules for an explanation of this.

Copyrights
All code that comprises this emulator was written by me with the following exceptions:
The RGB to composite color conversion algorithm, Floppy disk CRC algorithm and the 32x16 font set were stolen…. umm borrowed! from M.E.S.S. The MESS source was also a valuable source of technical information, without which, this emulator would not be as good as it is today.
The Basic and Disk Basic ROM images were and probably still are copyrighted by Tandy.
The RGB-DOS image was taken from Robert Gault's website. Please consult his website for support and documentation. It can be found here:
http://home.att.net/~robert.gault/Coco/CoCo_main.htm

The Status Line



The status line contains useful information about the status of the emulation.

1. Current frame skip setting. It means draw every *n*th frame. So 1 is every frame , 2 is every other frame etc.

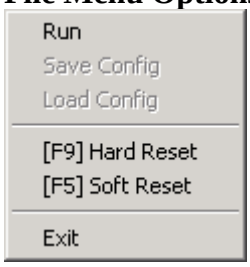
2. Average frames per second. Should always read 60. if it consistently reads lower try selecting a higher **frame skip** or turning on **scan lines**. See the configuration dialogs section.

3. CPU Type currently being emulated. MC6809 or HD6309.

4. Clock speed. .89 and 1.79 Mhz are stock. Overclocking up to 90Mhz is supported.

5. Cartridge data field. Some plugable carts will also display information on the status line. More on this later.

File Menu Options.



Run: Used in conjunction with the “AutoStart Emulation” configuration option. If AutoStart in not selected the emulator will start up in the “off” state. Use this option to start the emulation.

Save Config:
Load Config: Not currently used. Future version will allow the saving and loading of multiple custom configurations.

Hard Reset: Simulated a power off – power on cycle.

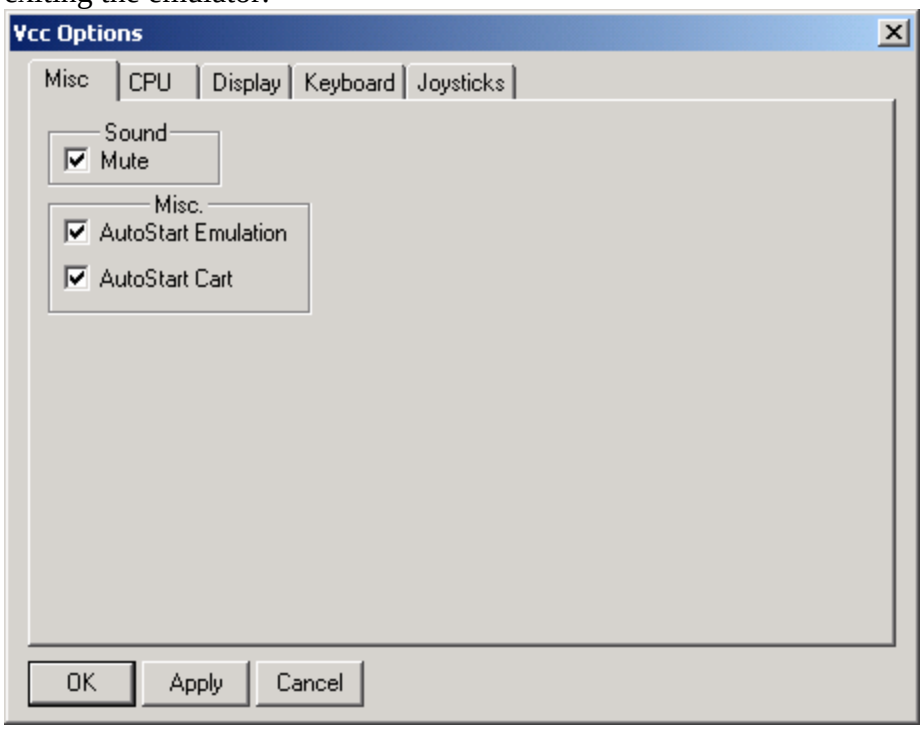
Soft Reset: Same as pressing the reset button.

Exit: closes the emulator.

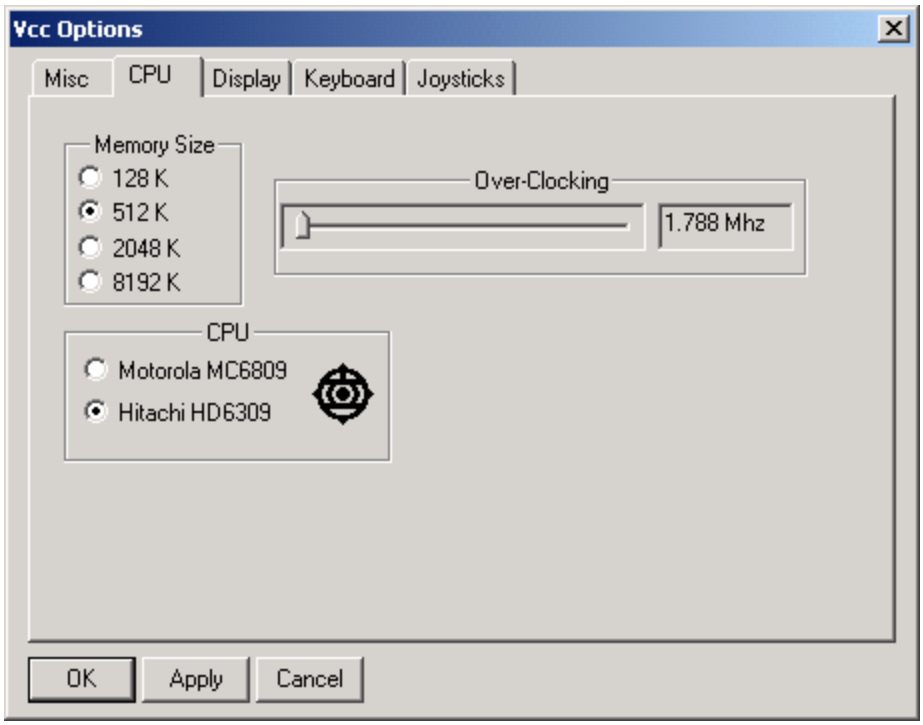
Cartridge Menu.

There are only two options here "Load Cart" and "Eject Cart". A cart in this context is any program pack (.rom file) or run time loadable DLL file. See the section entitled "loadable modules" for more information on this.

Configuration Dialogs.
The configuration dialog is broken up into five sections.
Most are fairly self explanatory however some my require explanation.
All these options are stored in the Vcc.ini file located in the emulator's home directory and are saved upon exiting the emulator.



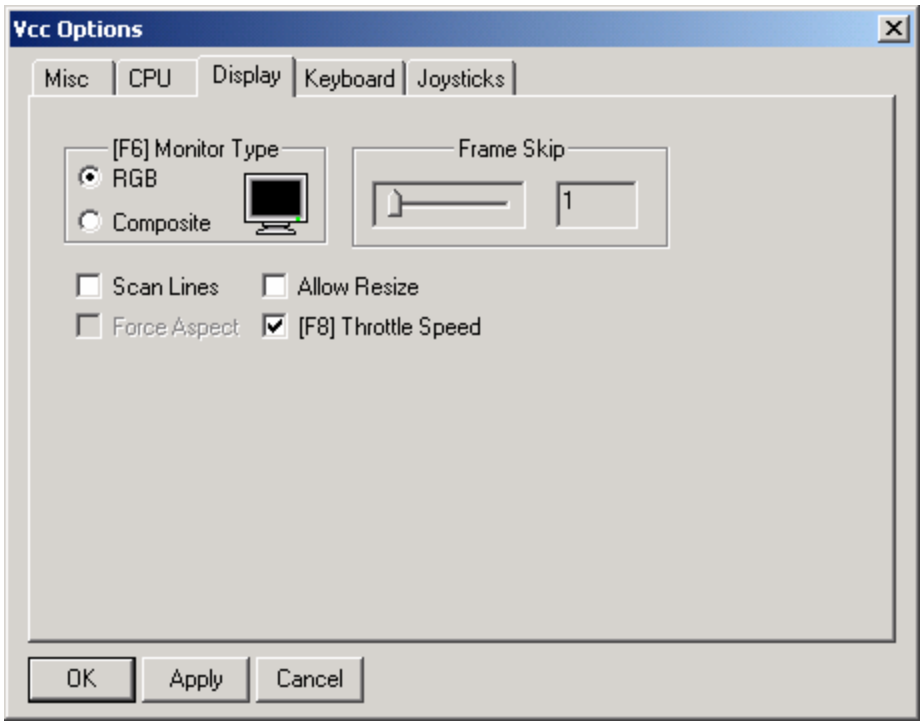
Mute does just that. When checked the emulator will not produce sound.
AutoStart Emulation. When starting the emulator program the emulation will start immediately if this is checked. If unchecked the static screen will be displayed until you select File->Run.
AutoStart Cart. If checked when loading a rom cartridge the cart will begin running immediately as the real hardware would. Un-checking this allows basic to boot without transferring control to the cart. For the tech minded it suppresses the FIRQ signal.



These are all straight forward.
Memory Size selects the amount of RAM the emulator sees. 128K & 512K are standard configurations. 2048K & 8192K are emulations of third party add in boards. Changing this will cause the emulator to hard reset.

Over-Clocking Normal .89MHz operation is **unaffected** by this. This slider controls the "Double speed" poke. Speeds up to 90MHz can be selected but usability will be determined by the host computers speed. This option can be changed freely during emulation.

CPU The original Color computer 3 used the Motorola MC68B09E cpu. It was discovered that Hitachi made a pin compatible but superior CPU called the HD63B09E. This selects which chip to emulate. Changing this will cause the emulator to hard reset.



Monitor Type The Color computer 3 could be attached to 2 types of displays; composite such as a TV or component RGB as in the CM-8 Monitor. Strangely enough each display shows different colors with RGB being the preferred. This allows you to select what "color set" the display will use. This can also be changed via a keyboard shortcut [F6]. See the section on function keys for more information.
Note: If you are running a PMODE 4 program that uses **ARTIFACTS** you should select the Composite option as the RGB monitor was incapable of this.

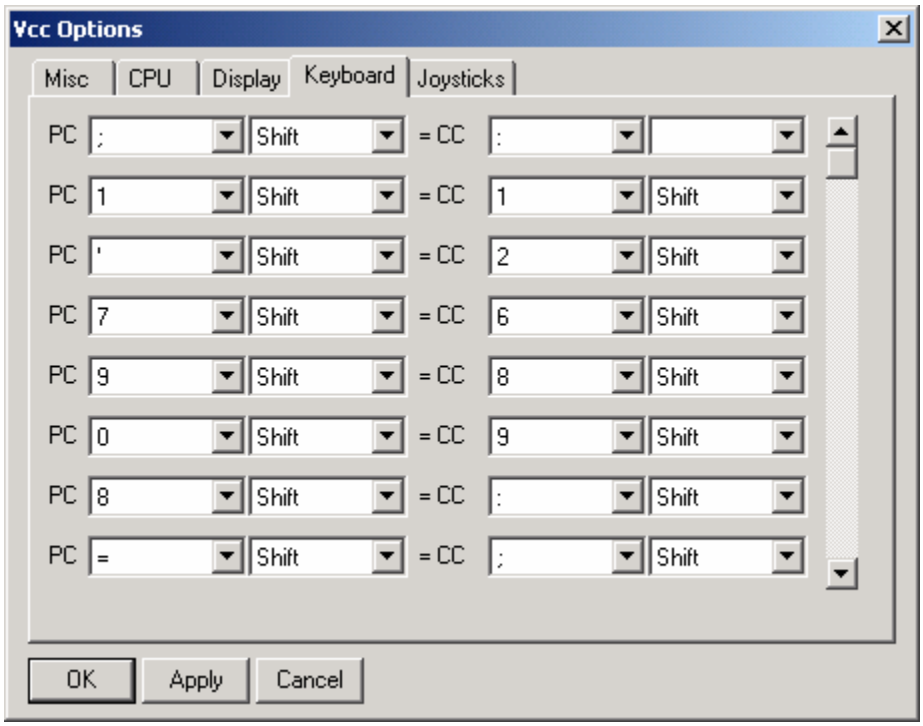
Frame Skip Selects how many of the 60 Frames per second will actually be drawn. A setting of 1 will draw every frame and is the preferred setting. "skipping" frames is useful only if the host computer is not fast enough to draw every frame (see the description of the status line on page 2).

Scan Lines has two functions. It looks more like a classic TV/RGB monitor and can be used as an alternative to frame skipping as only half the lines need to be rendered per frame. Useful for slower Host machines.

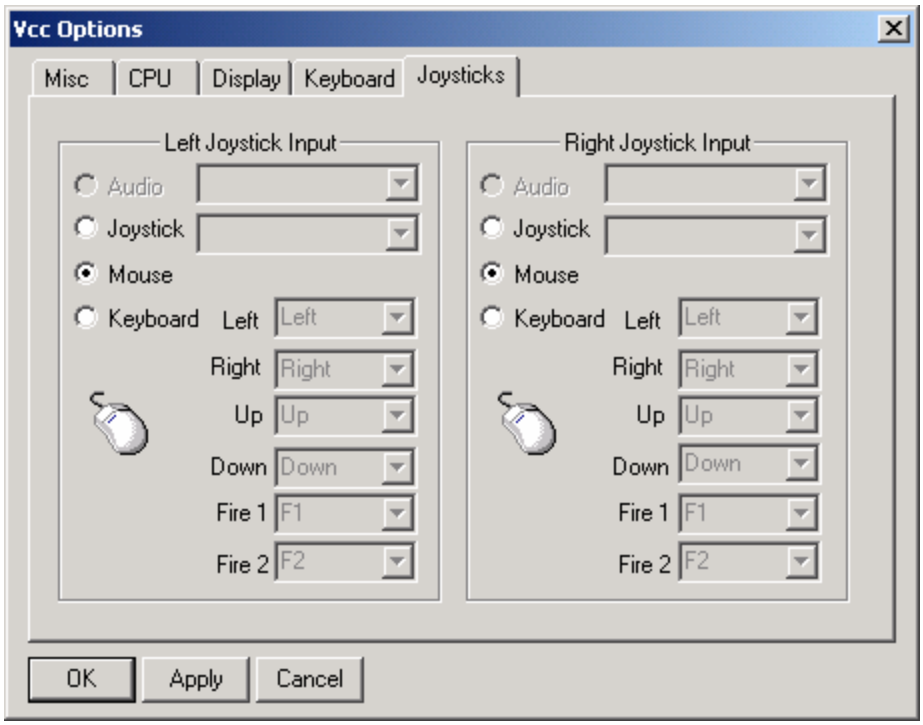
Allow resize If this in unchecked the program window size will always be 640 x 480. checking this will allow the window to be freely resized. Newer video cards can do this resizing in hardware so there is little cpu overhead, However older cards must use the cpu to resize. This can lead to a major performance hit.

Force Aspect *Not coded for yet*! Will be used in conjunction with the **Allow resize** option for force the normal 4:3 aspect ratio of a TV/Monitor when resizing.

Throttle Speed. Normally the emulator will try to run at the original 60 frames per second that the real hardware runs at. Un-checking this will allow the emulation to run as fast as possible. Note that some video cards don't allow more screen updates than what their refresh rate is set for. This can also be changed via a keyboard shortcut [F8]. See the section on function keys for more information.



Due to the fact that the coco keyboard is vastly different from the standard PC keyboard there needs to be a way to map the PC keys to the corresponding coco keys. This dialog allows you to map PC keys to their coco equivalents. On the left are the PC keys that will need to be pressewd in order for the coco to see the key(s) on the right. For example on the first line pressing [Shift] and ; on the PC will send the code for the : (colon) character to the coco. Due to the way this information is processed internally it is necessary to resort to this list. Consequently key-maps added to the end of the list may not stay there.
Future versions will allow multiple Keymap files be selected and hopefully will have a more Intuitive and user friendly interface.



This dialog allow you to select how the joysticks will be emulated.
Currently there are 3 options available

1) Joystick. This is a real Direct-X compatible Joystick connected to your system. The current version allows only the first Joystick found on the system to be used although all joysticks connected will be displayed in the drop-down. **This will be fixed in future versions.**

2) Mouse. The Virtual Joystick will follow the position of the mouse pointer.

3) Keyboard. This allow "bang-bang" steering, IE: up/down/left/right.
By default the arrow keys are used with F1 and F2 used as fire1 and fire2 respectively. This can be changed via the 12 drop-downs provided.
Please note that when this option is selected the joystick will take precedence over the keyboard and you will be **unable to type these characters.**

4) The Audio option will be available in a future version. The reason this is here is that some coco programs can actually use the joystick port's ADC (analog to digital converter) as a rudimentary audio recording device. This will allow redirection of the host computer's soundcard audio input to the joystick port.

Function keys.

F1 & **F2** By default these are mapped to the F1 and F2 keys on the standard coco keyboard. Alternatively they are the default Fire 1 and Fire 2 when using the keyboard to emulate joystick input.

F5 Soft Reset. Same as pressing the reset button on a real machine.

F6 RGB/Composite toggle. Has the same effect as the Display Dialog setting except that unlike the configuration dialog option, changing this will not save its state to the .ini file.

F8 Throttle Toggle. Normally the emulator will try to run at the original 60 frames per second that the real hardware runs at regardless of the speed of the host cpu. This is known as “throttling”. Alternatively the emulation can be allowed to run as fast as possible. This key is used to toggle between these two modes. It's useful during long loading or processing tasks to shorten the wait time. Note that unlike the configuration dialog option, changing this will not save its state to the .ini file.

F9 Hard reset. Same as pressing the Power button on a real machine.

F10 Only used in Full screen mode. In windowed mode there is a status bar at the bottom of program window. In Full screen mode this information is displayed in a band at the top of the screen. This is used to toggle that band on and off.

F11 Switches between Full screen and Windowed mode.

Loadable Modules.

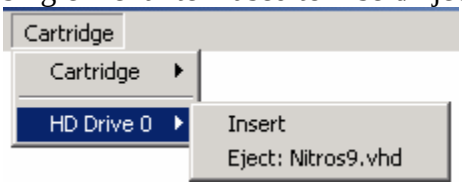
As stated before the Vcc emulator does not know anything about the various peripherals that were available. It depends entirely on run time loadable DLL files or Modules.
Modules (DLL) or program packs (ROM) images are loaded via the Cartridge menu option. After a module is loaded it will, if needed, add its own options to the Cartridge Menu and status line. There are currently 4 modules included, They are:
Mpi.dll, orch90.dll, harddisk.dll, and fd502.dll. These are installed in Vcc's home directory "C:\Program Files\Vcc" by default.

orch90.dll

This is the simplest module. It emulates the orchestra-90 program pack. It has no menu options and returns no status. It contains a dump of the program rom that came with the original hardware.

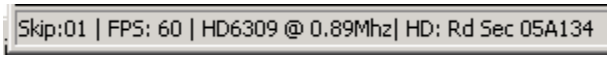
harddisk.dll

This is an implementation of the "emulator hard disk" that is supported in many Coco emulators. It adds a single menu item used to Insert/Eject a virtual hard disk image (VHD).



On the status line an additional value simply labeled HD:IDLE will appear when there is no disk activity.

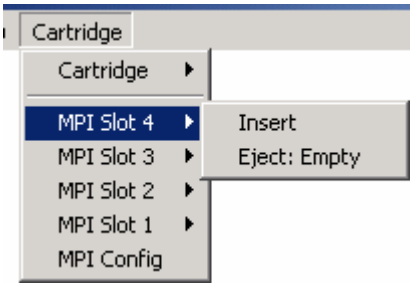
During disk access IDLE will be replaced with the 24Bit HEX Address of the sector being read from or written to.



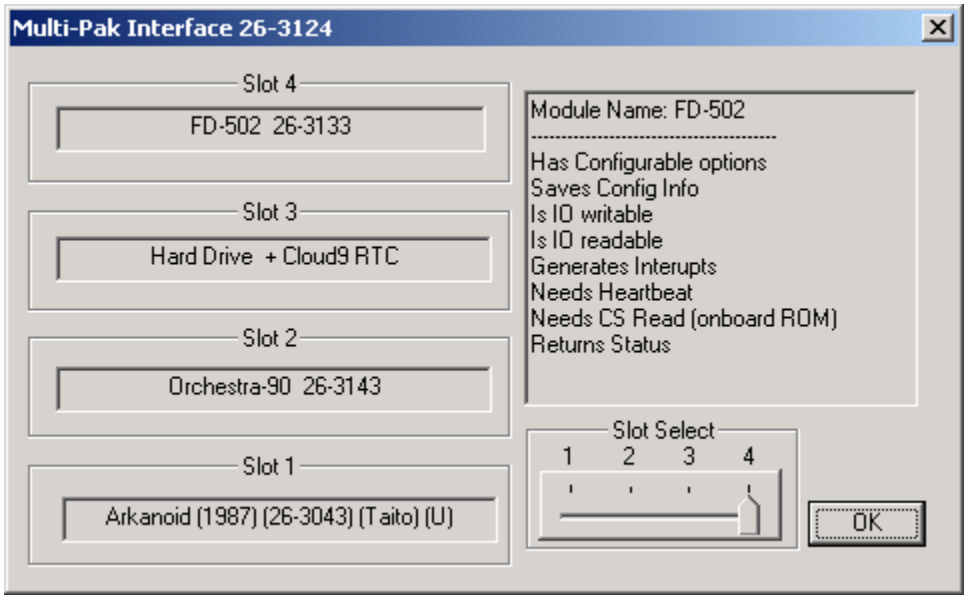
It contains an image of the RGB-DOS rom. This is a version of DOS that has been modified by Robert Gault for use with this type of emulation. It also contains an implementation of the Dallas DS1315 real time clock as used by Cloud-9. This can be used under OS9/Nitros9 with the appropriate driver.

mpi.dll

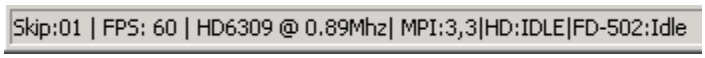
The mpi module emulates the standard Tandy Multi-Pack interface. It adds 5 menu options. The first four allow Inserting/Ejecting carts from the Multi-pack. Any loaded modules will also add their own options to the menu. As with the main emulator, each slot will accept either a Module DLL or a Program ROM pack with the following exceptions: don't try to insert the mpi.dll module into an mpi slot. Windows does not support recursive library loading. Also don't try to insert the same module into more than one slot. This is a limitation of the way the dynamic menu system currently works and will be fixed in a future version.



The MPI Config option is mostly informational. The Slot Select Slider picks the slot the MPI will use on startup. Displayed on the left are the names of all loaded modules. On the right is a list of API interfaces the currently selected module needs.



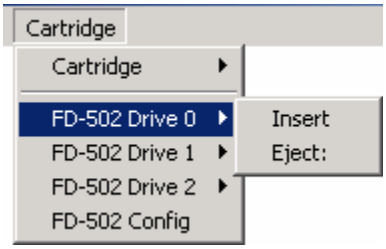
This module will add an MPI item to the status line. Following this are two numbers. The first is the slot the Chip select signal is routed to, the second is the destination of the Spare Chip Select line. Following this is any status info returned from loaded modules. In the example below the MPI is set for slot 4 (internally slots are numbered 0 to 3) and the harddisk and fd502 modules are also loaded.

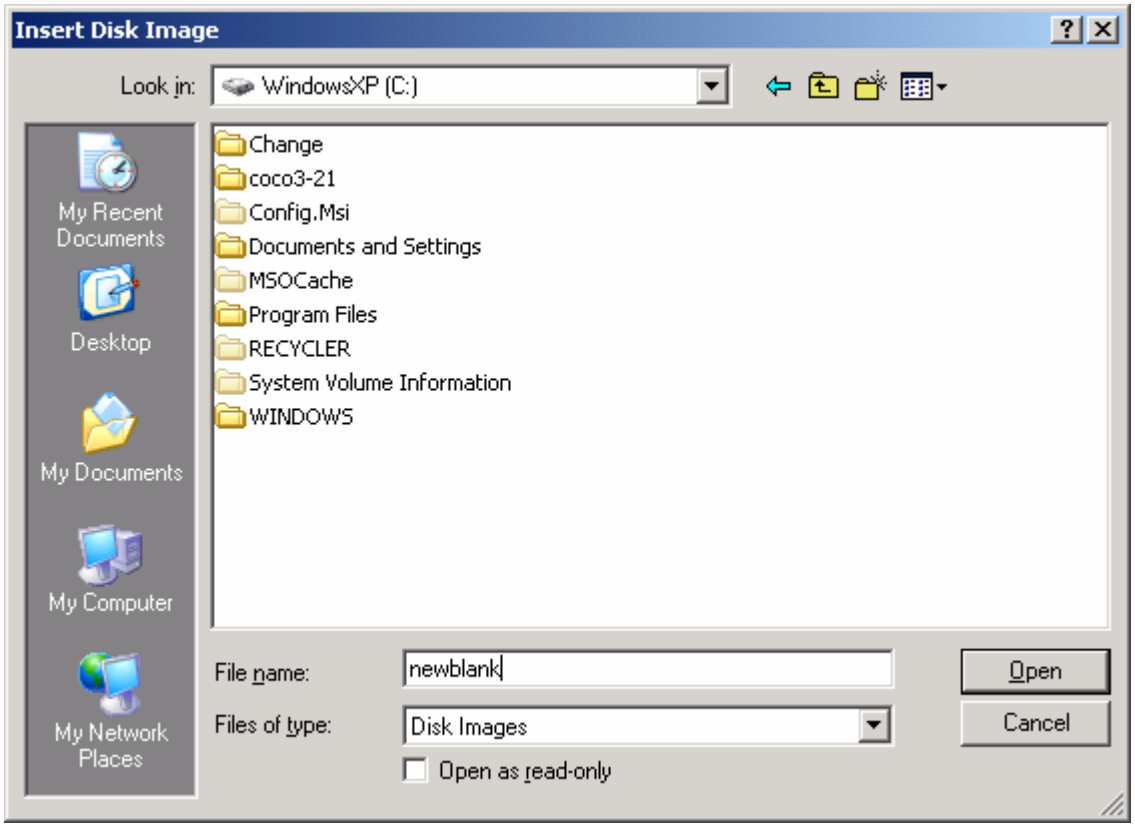


fd502.dll
This module emulates the Tandy FD-502 Floppy disk controller with 4 Double Sided/ Double Density disk drives attached. It adds 5 options to the Menu. The first 4 are simply to Insert / Eject Virtual disk images. Most image types are supported including DMK,OS9,JVC and VDK. To insert a virtual Disk simply select the drive you wish to put it in and select insert. Note: Only the first side of Drive 3 can be accessed.

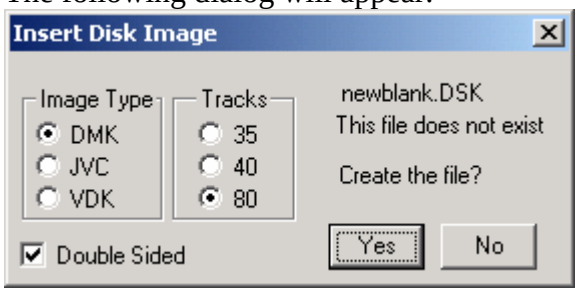
If you wish to have the disk "**write protected**" simply set the "read only" attribute from windows. This can be done by right-clicking the file, selecting Properties and checking the "read-only" box under Attributes. This will work for all image types. The DMK format uses a byte in the header to indicate "write protect". This will be respected if present but there is currently no way to change it from within this emulator.

Creating a new blank disk will be similar to inserting an existing image.
Simple click Insert, and instead of picking an existing image type the name of the new disk image you wish to create.



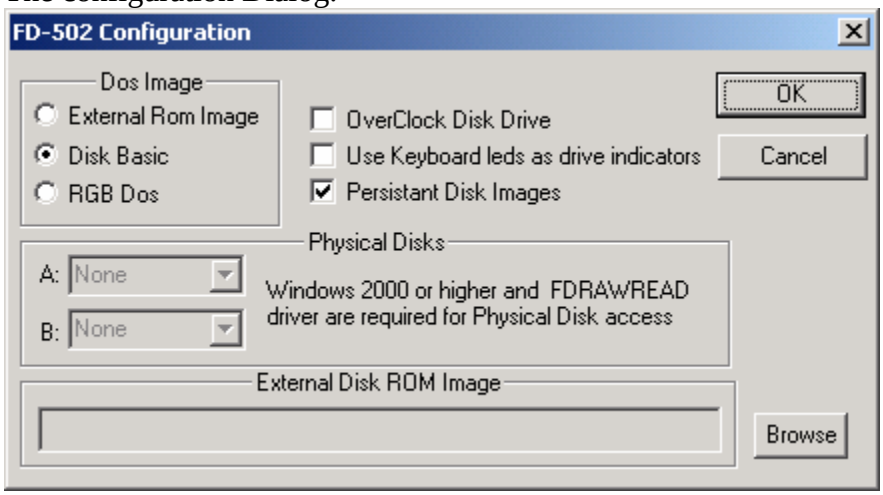


The following dialog will appear.



Just select the parameters for the new image and click yes. The image will be created, Zero filled and mounted in the drive you've selected. Note that unlike most other emulators the act of creating a new disk will **NOT** format it. Just as you would with a real disk you must either DSKINI (DECB) or format (OS9 et al) the image before it can be used.

The configuration Dialog.



Dos Images:
There are two built in DOS roms to choose from. Disk Basic is the standard DOS 1.1 that shipped with the FD-502 controller. RGB Dos is a version of DOS modified by Robert Gault to take advantage of the virtual hard disk emulation described elsewhere in the document. External Rom Image is just that. If you have a dump of a custom version of DOS (ADOS3 for example) simply click "Browse" and select it. then check the "External Rom Image" radio button. It is recommended that you keep this image in the same directory that VCC is installed to, but it's not required.

OverClock Disk Drive Sounds more sophisticated than it is. By default I attempt to emulate the time it would take a real disk to move the read/write head. Clicking this will reduce that time to almost nothing. If you are using the CPU overclocking option and experience I/O errors try turning this on.

Use Keyboard leds as drive indicators !EXPERIMENTAL!
Lights are mapped as follows:
Drive 0 = Num Lock
Drive 1 = Caps Lock
Drive 2 = Scroll Lock
Drive 3 = None. Ran out of leds ☺

Note that this will really change the state of these keys. IE if you have Caps Lock on and Drive 1 is accessed. When the Emulator shuts off drive 1 and the light, Caps lock will now be off. Still Working on this.

Persistent Disk Images:
When checked any Disk images mounted when the emulator is shut down will be remounted when it's started back up. Un-checking this will cause the emulator to start with empty Floppy disk drives.

Physical Disks:
This is also experimental code. Please only use backup disks as I can't guarantee everything works 100% yet. Currently only "standard format" 18 sector per track disks are supported. I'm hoping get "protected" disk formats working in a later version.

To use this option 3 requirements must be met. The drop downs will be grayed otherwise.

1 The host computer must be running Windows 2000,XP or Vista. Note I have only tested this on XP so far.

2 The host computer must have a supported Floppy disk controller chip. (µPD765a or equiv.). a USB floppy drive will NOT do.

3 The FDRAWCMD driver must be loaded. These drivers were written by Simon Owen and can be downloaded from his website here:
http://simonowen.com/fdrawcmd/
Download and run the fdinstall.exe file. As of this writing the current version is 1.0.1.9. To avoid problems this is the only version the emulator will use. This will change in a later version of Vcc.

There are two drop downs. One for each Physical disk drive you may have. Simply select the Virtual disk drive from the dropdown next to the Physical disk Letter. For Example: To Map Virtual disk 1 to real Drive A:, Select "Drive 1" from the dropdown next to A:.
To un-map a disk either select None from the dropdown or select Eject *Floppy ?? from the menu, where *Floppy?? Is either A or B.

Technical Information.
These pages contain various technical information. Nothing on the following pages is vital to the emulator's operation. It is here only for the technically minded and the curious.

Technical specs on the Virtual Hard Disk interface (borrowed from the MESS Source).

Address	Description
FF80	Logical record number (high byte)
FF81	Logical record number (middle byte)
FF82	Logical record number (low byte)
FF83	Command/status register
FF84	Buffer address (high byte)
FF85	Buffer address (low byte)

Set the other registers, and then issue a command to FF83 as follows:
0 = read 256-byte sector at LRN
1 = write 256-byte sector at LRN
2 = flush write cache (Closes and then opens the image file)
Note: Vcc just issues a "FlushFileBuffers" command.
Error values:
0 = no error
-1 = power-on state (before the first command is received)
-2 = invalid command
2 = VHD image does not exist
4 = Unable to open VHD image file
5 = access denied (may not be able to write to VHD image)
IMPORTANT: The I/O buffer must NOT cross an 8K MMU bank boundary. Note:
This is not an issue for Vcc but should be respected to maintain compatibility with other emulators.