Vcc
A Tandy Color Computer 3 Emulator for Windows
'Cause we're still preoccupied with 1985

Change Log

- VCC v2.1.0b Nov/2020   Added "Copy/Paste" Edit menu items for copying text from VCC screen and pasting text into VCC. Improved Composite Palettes (not fixed, WIP). Fixed "file paths" so that each type of file; vhd, dsk, cas, rom, dll, etc. has it's own pathlist in the "vcc.ini". Added "Flip Artifact Colors" to the config menu. - James Rye. Added "Save Config" and "Load Config" to the "File" menu (previously disabled). Also added the ability to load custom "*.ini" files when running VCC in a Command Box. Both of these features allow "Custom ini" files for save different configurations – EJ Jaquay. Updated manual – Bill Pierce
- VCC v2.1.0a Oct/2020   Fixed VCC "ini" file to save in "user/home/appdata" to avoid permissions problem in Win7-10. Fixed "Force Aspect" to actually work in all modes but full sceen mode. Updated manual to reflect changes.- Peter Westburg, James Rye, & Bill Pierce
- VCC v2.0.1f Aug/2020   Improved PMODE artifact color scheme, - Peter Westburg
- VCC v2.0.1e Dec/2019   Reverted "exit bug" fix as it caused VCC to hang in full screen mode – Bill Pierce.
- VCC v2.0.1d Dec/2019   Added "*.ccc" to the program pak extensions. "Exit bug" fixed. Several "cosmetic" changes – Bill Pierce.
- VCC v.2.0.1c Dec/2019   Completed 6309 opcodes that were missing or incorrect – Walter Zambotti. Updataed manual – Bill Pierce
- VCC v2.0.1b June/2016   Several "Edited" releases under same versions. Some "cosmetic" changes to dialogs.
- VCC v2.0.1b June/2016   Minor syntx changes to achieve working build for VS2015 – Wes Gayle
- VCC v2.0.1a June/2016   Moved code to "MS Visual Studio 2015 Community" (No Release) - Gary Coulborne
- VCC v1.42c   ??/2016   VCC source code released to "Open Source" by Joseph Forgeone. GitHub repository created by Bill Pierce (No Release)
- VCC v1.43c   ??/2015   Move the "Boisy/Becker" port to a chartridge format to eliminate crashes (never released) – Aaron Wolfe, David Ladd
- VCC v1.43b   ??/2012   Added support for the "Boisy/Becker" interface for DriveWire4 (Binary Release Only) – Aaron Wolfe, Davidd Ladd
- VCC v1.42   ??/2012   Source code aquired for VCC by Aaron Wolfe, David Ladd, & Bill Pierce (No Release)
- VCC v1.42   ??/????   Last known release from original author Joseph Forgeone

Welcome to VCC

(Virtual Color Computer)

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.

- 64k system memory, 128K via bank switching.
- 512K upgrade available from Radio Shack as well as 1, 2. & 8 meg memory upgrades from 3rdparty vendors.
- 32, 40, and 80 column hardware text modes.
- Hi-res graphics resolutions up to 320x225 at 16 colors and 640 x 225 at 4 colors.
- Clock speeds of 0.89 and 1.78 MHz. Software selectable.
- Super Extended Color Basic ROM.
- Color Computer 2 compatibility mode.

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 512k memory cards, IDE and SCSI hard disk interfaces and faster, more powerful CPUs in the HD63B09E. There is even a free user supported Multi-tasking Multi-User OS (NitrOS9) 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 (Windows XP or greater). 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 Coco 3 with no peripherals. Expansion is possible via a system of run-time loadable plug-ins that emulate the various add-in card that were/are available. Sort of an Emulator within an Emulator. See the section on Loadable Modules for an explanation of this.

Joseph Forgeone (Original VCC Author)

Copyrights

All code that comprises this emulator was written by Joseph Forgeone and is under the "GPL3" license with the following exceptions:

The RGB to composite color conversion algorithm, Floppy disk CRC algorithm and the 32x16 font set were stole…. 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 "Becker Port" implementation was patched by Aaron Wolfe and David Ladd, giving "VCC 1.4.3b" the ability to communicate to the host PC via the DriveWire4 protocol and since then, in VCC 2.0.1, the Becker Port was moved to a cartridge emulation "becker.dll" and must be loaded into the MPI.

The "MS Visual Studio 2015 Community" port of the old "MS Visual C++ 6" sources was done by Gary Coulborne (Thanks Gary!) and is allowing VCC to now move forward on the more "modern" versions of Windows.

Various bugs cropped up in the move from VS6 to VS2015, and Wes Gale stepp in to lend a hand at getting the code usable.

The HD6309 code was unfinished and incomplete and was finally put in place by Walter Zambotti

There have been various enhancements and bug fixes by Bill Pierce, Gary Coulbourne, Wes Gale, Walter Zambotti, James Ross, Peter Westberg, James Rye and EJ Jaquay.

VCC is maintained by the VCCE Open Source Project Team:

Administrators:
Joe Forgeone (absent)
Bill Pierce
Gary Coulborne (absent)
Wes Gale (absent)

Programming Contributors:
Walter Zambotti
James Ross
Peter Westburg
James Rye
EJ Jaquay

VCC Manual, Repository Maintenence, Packaging & Distribution:
Bill Pierce

Inspiration and Continued Support:
The Tandy Color Computer Community

We welcome anyone interested in programming & enhancement of the VCCE project. Contact me on the "Color Computer" FaceBook group with your ideas and we will determine if it fits the direction we are headed with the VCC project.
Bill Pierce

The home of "VCC" is:
https://github.com/VCCE/VCC/releases

Where you will find the latest binary downloads as well as the sources for VCC.

Forward

VCC version 1.4.2 was originally written by Joseph Forgeone. In or about 2012, several of the Coco Community had contacted Joseph about VCC since he hadn't updated the software in several years. Joseph informed us that he would like to release the VCC source files to the public as an "Open Source Project" so that it could be enhanced by others. His desire was to remove the Tandy ROMs (for copyright reasons) and make small changes in his copyrights before releasing the sources. He told us he should have it ready for release in a couple of months. As a "token of good faith", he offered to send us the sources to "play with" until he made his release. After that, Joseph seemed to disappear from the map. He had not replied to many emails nor could any posts be found on any of the forums as he was a regular on some Coco forums. He seemed to have disappeared into thin air.

As the events above describe, several of us ended up with sources to VCC. No, we could not release these sources without Joseph's permission. But in frustration of the situation, holders of the sources started "playing" with them as Joseph had suggested. The main goal being to add support for the Becker Port system developed by Gary Becker as well as to fix a few minor known bugs. Since Joseph could no longer be contacted, the decision was made to release some changes to VCC only in "binary" form in executable packages therefore keeping the sources from public view. Thus VCC 1.4.3b Beta w/Becker Port was born.

As the "Becker Port" patch eventually started causing a few *internal* problems in VCC, the Becker Port developers decided to move the port to an external "cartridge port" as a "Rom Cart", therefore removing all *internal* influences of the Becker Port to VCC. So now, to use the Becker Port, you must load the "becker.dll" into either VCC's cartridge slot or a slot in VCC's MPI. I find this new implementation of the Becker Port to be much more stable than the original version.

ENTER JOESEPH FORGEONE...

Several of the Coco community decided that after a couple of years of not hearing from Joseph (VCC author), that they would follow his intentions and release the VCC sources as an "Open Source Project". In doing so, we decided to make a public announcement in advance to allow anyone having claims to the copyrights of VCC to step up. Amazingly, the first person to reply was Joseph Forgeone!!
Not only did we get the "VCC Open Source Project" started, but we did so with Joseph on board and contributing to the project. We have now moved the VCC sources from the old "MS Visual Studio 6" programming environment into the "MS Visual Studio 2015 Community" programming suite which is a free download from the Microsoft website. The VCC sources can now be compiled in Windows 7, 8, & 10.

Needless to say, we now have new development of "VCC" which should be much more compatible with modern versions of Windows. There are new enhancements, bug fixes and there are plans to port the code to Linux and Mac operating systems.... The future is bright for VCC!!

This document is meant to be an enhanced version of Joseph's original VCC User's Manual with expanded content that was never added to the original manual as well as documentation of the new enhancements and bug fixes. Hopefully, it will cover a larger amount of information than Joseph's original document with more information on the operation of VCC. It is my goal to make VCC as easy to use as possible so that everyone can enjoy this package as much as I have, as I am an avid user of VCC as well as the "real" Color Computer 1, 2, & 3 hardware. As new developements are made on VCC, I hope to expand this manual as needed.

As one last note: VCC *will always be* in a BETA state and by no means is a finished product. As of this release, quite a few things have been changed from the original. There are still many bugs to be worked out and enhancements to be added. It is this very reason the VCC Color Computer 3 emulator has been released as an open source project in hopes that outside contributors and/or programmers will get involved and help this become the best Color Computer 3 emulator available!

Enjoy ☺
Bill Pierce

Introduction

VCC is a move forward in the Color Computer 3 emulation. The emulator covers many aspects of the Color Computer 3 hardware. The author has tried to keep the "look and feel" of the real machine while maintaining a sense of a newer and better machine. There may be a few bugs lurking in the remenants of the old C code, but for the most part, the emulation is still one of the best Coco 3 emulations available. Currently, work is underway to convert VCC to *multi-platform* software, enabling VCC to be run on Windows, Mac, Linux and possibly even Mobile Devices. If this is accomplished, we will have a Coco 3 emulator available for most *modern* systems.

System Requirements

Currently, VCC runs only under Microsoft Windows but has been successfully installed on Linux systems using the Wine Windows emulation package for Linux as well as "Winebottle" on Mac. I do not know too much about these installations, but I am told they work well.

The Windows requirements are:

1. Microsoft Windows XP SP2 or greater (Windows Vista, 7, 8, & 10 are supported).
2. A Hard Drive large enough for the Windows OS and the VCC installation (2.5 meg) as well as space for any Coco ".dsk" (virtual floppy disk images) and ".vhd" (virtual hard drive images) files you plan to be using. Also you will need space for DriveWire4 if you are planning to use this system as well. Today's *modern* computers have no problem with storage space with their large capacity storage systems.
3. On older Windows systems (XP), I suggest at least 512 meg of memory. On newer systems (Vista and greater), I suggest at least 1 gigabyte or memory, and 4 gigs for heavy users, especially if using DriveWire4.
4. DirectX 9 or greater. Most Windows PC graphics systems will work well with VCC. (some newer versions of Windows may need the "DirectX 9 runtime" dll).
5. If you are using VCC with DriveWire4, an internet connection is also desirable (but not mandatory). We'll get into this later.

VCC Installation

VCC is distributed in a self-installing executable package. Just "double click" the "VCC setup.exe" and follow the prompts. Windows will install all components and provide a Desktop icon for starting VCC.

If you want to install VCC in a different location than the default, choose the "Custom" option when installing, otherwise, VCC will normally install to "C:\Program Files (x86)\VCC" on your hard drive.

Setting up and Configuring VCC

VCC allows several setup configurations with RGB or Composite monitor emulation, 6809 or 6309 CPU, from 128k to 8 megabytes of memory, and many others. In these next sections I will try to step through each option and explain its function.

NOTE: In previous versions of VCC running under Windows 7, 8, & 10, the user was forced to run VCC in "Administrator Mode" to allow VCC to maintain it's configuration file due to permissions changes in the newer versions of Windows. Hopefully we now have that problem solved and VCC will no longer need to be given administrator rights to work properly. Please let us know on the "Issues" page of our GitHub distribution site is there are any problems with this.

In fixing the permissions problem, VCC now stores it's "vcc.ini" file under "User\%Home%\AppData\VCC", which allows VCC to update and keep it's settings. We now have the ability to Load/Save "Custom" *.ini files, so those with multiple VCC installations can load different configurations.

The VCC Installer

The VCC team uses the "Inno Setup Compiler" & the "Inno Script Studio" to compile it's installation system. The Inno system is an open source project and is used by literrally thousands of software distributors including many commercial software houses. Care has been taken to make the VCC installation as easy as possible.

When first run, the VCC installer, like any other installer, requests the usual license agreement. After the license agreement, a text file is displayed, explaining a few of the distribution items contained in the package and how to use them. It is best, if you have not read this article, to do so now. You can click "Cancel" at any time to cancel the VCC installation.

Next you are given the chance to select your installation directory path. The default path is "C:\Program Files (x86)/VCC 2.x.xx". To install to a different directory, click the "Browse" button to the right and select your directory. You will need at least 2.5 meg available for installation. Click "Next" and VCC is set to install in the directory displayed.

The next dialog let's you choose the components to be installed. Checking/Unchecking the boxes determines which components the installer will use. These choices are:

X VCC 2.x.xx Coco 3 Emulator
     X Support File for the VCC 2.x.xx Coco 3 6809-6309 Emulator
X Vcc 2.x.xx Color Computer 3 Manuals
     X Welcome to VCC 2.x.xx

Basically, you want to install all files, as all the first category are need, and the manual includes any updates to the current version being installed. This manual will be installed to "Documents\Coco Manuals". Clicking "Next" continues the installation.

The next dialog allows you to select the name of your "Start Menu" group in which the VCC shortcut will reside. This also determines the name given to your VCC shortcut. Unless you have more than one instalation of VCC, I suggest this be left "as is". Click "Next" to continue.

This dialog lets you select if you want a "Desktop Shortcut" or not. The shortcut will be named using the above selected name. Click "Next" to continue.

This final dialog is your last chance to cancel, or change your VCC setup. A review of your previous choices are shown to make sure this is what you want.
Clicking "Back" will of course take you back to previous dialogs, allowing you to make changes to your setup. Once you click "Install", the installer will install the VCC files in the manner you have chosen.

In the installation, if another VCC folder of the same name is present, the installer will overwrite any previously installed files with the new installation and your old version of VCC will be updated. Any old "Configurations" from the prevous version will be maintained.

Before the installation proceeds, you may or may not be greeted by a standard Windows permission dialog asking if you to allow this installation to proceed. Just click "Allow" or "Ok" (based on your Windows version) and installation will continue.

The VCC Color Computer 3 installation is complete.

VCC 2.x.xx Installation Zip File

Alternately, you can download the VCC 2.x.xx zip file and install VCC as you please. This file will unzip to your desired folder and you will have to manually copy files to any destination you choose. The VCC Development Team is not responsible for installatiions installed in this manner and you're on your own.

VCC Installation Package Contents

The VCC installation package & VCC Zip Package contain these files:

VCC 2.x.xx Coco 3 Emulator (Installed to the VCC program folder)

Vcc.exe (The VCC emulator itself)	
wimgtool-os9.exe	(OS9 Disk Utility)
wimgtool-rsdos.exe	(RSDOS Disk Utilty)
becker.dll	(Becker Port Cartridge for DriveWire4)
fd502.dll	(Tandy FD-502 Disk Controller)
harddisk.dll	(Generic Hard Drive Controller)
mpi.dll	(Tandy Multi-Pak Interface)
orch90.dll	(Orchhestra90cc Program Pack)
Ramdisk.dll	(256k External Ram Disk Cartridge)
SuperIDE.dll	(Cloud9 IDE Hard Drive Controller)
coco3.rom	(the Coco 3 BASIC ROM, auto loaded)
disk11.rom	(the Disk BASIC Rom, used by the FD502.dll, auto loaded)
hdbdw3bc3 w-offset 5A000.rom	(Variant of the HDBDOS ROM, see ROM descriptions)
hdbdw3bc3.rom	(Variant of the HDBDOS ROM, see ROM descriptions)
hdbdw3bck w-offset 5A000.rom	(Variant of the HDBDOS ROM, see ROM descriptions)
hdbdw3bck.rom	(Variant of the HDBDOS ROM, see ROM descriptions)
hdbla.rom	(Variant of the HDBDOS ROM for SuperIDE, see ROM descriptions)
orch90.rom	(Orchestra90 ROM used by Orc90.dll, auto loaded)
rgbdos.rom	(RGBDOS ROM for VCC's harddisk.dll)
license.txt	(License Agreement)
ReadMe-2.x.xx.txt	(ReadMe file displayed in installation)

Vcc 2.x.xx Color Computer 3 Manuals (Installed to the user's "Documents\Coco Manuals" folder)

Welcome to VCC 2.x.xx.pdf	(VCC User's Manual)

Starting VCC:

To start the VCC emulator, just double click the VCC icon on your Windows Desktop or clcik the VCC menu item in your Windows Start Menu.
Alternately, VCC can be started from a Windows Command box by Cding to the VCC installation folder and typing:

VCC.exe
VCC now has a new feature to allow you to load a custom configuration from the cmd line by typing:
VCC.exe -i xxxxx.ini (xxxxx being the custom configuration name)

NOTE:
The ROM images are provided for your convenience and to assure the proper ROMs for VCC are present.
The VCC Development Team is not responsible for the ROM contents as they are each developed by 3rd party developers, though verified to be accurate to the original ROMs of the same name. These ROM imags are the same as used by most Color Computer 3 emulations.

NEW Features for this Edition of VCC:

- **Copy/Paste**
  Probably the most requested feature added to VCC. You can now Copy from the VCC screen to the PC clipboard or Paste to the VCC screen from the PC clipboard just as if you typed the data in by hand! (see the "Edit" menu section for details)

- **Persistent Filepaths**
  VCC will now remember all your ROM, DLL, Disk, VHD, Cassette, & Cart/Pak directories individually. No more browsing from one directory to another each time you load a different item. This is a change I personally have wanted for quite a few years! All current filepaths are save to the configurantion file and remembered each time you run VCC.

- **Custom Config Save/Load**
  In solving the problem users were having on Windows 8 & above with having to use "Administrator's Permission" to make VCC properly save the Config file, we broke the ability to have more than one VCC installation with different configurations. The new Save/Load Config menu has been activated allowing you to save custom configurations and load any configuration you want after starting VCC. (See "Menus/File/Save Config" & "Menus/File/Load Config" for details)

- **Load Custom Config from Cmd Line**
  After several requests, we have added the ability to load a custom configuration from the command line in a Windows Command box. (see "Starting VCC" at the end of the previous "Installation" section)

- **Flip Artifact Colors**
  VCC now allows you to "Flip" the artifact colors on the PMODE graphics screens to the alternate color set without interupting the emulation. Just click "Configuration/Flip Artifact Colors" on the menu bar and the alternate color set will be activated without interrupting your program.

- .**Slightly Better Composite Palettes**
  The composite palettes have been altered to be a little better, but still not correct.

- **Improved VCC Manual**
  I have reoganized this manual a little better to make navigation easier. I also addedd the much requested "Quik Start Guide".

I would like to thank James Rye & EJ Jaquay personally for stepping up to help with the VCC project and their work on these new features. I would also like to thank Peter Westburg who, at the moment, is fighting his own battles. He was a very large contributer in our last release but could not make it though this release due to his health. We hope you are back with us soon. Our prayers are with you Peter.

Bill P.

VCC Quick Start Guide

Installation Defaults

In response to many user requests, I am including this Quick Start Guide to get you up and running as fast as possible. Any description to components used in this guide can be found elesewhere in this manual. This guide is to get you started and *not* an explination of it's components.

Unless you have a previous installation of VCC of the same name and location as this one, the VCC emulation will start as a Tandy Color Computer 3 just as you bought it from Radio Shack, which consists of:

Color Computer 3 (Installation Defaults)
     128k memory
     PC Keyboard remapped to match the Coco 3 layout as close as we possibly could
     Cassette Tape interface for load tape images ("*.cas" & "*.wav")
     Bitbanger (RS232) Port for printing listings to a PC file of your choice.

Through it's additional modules ("Cartridge" menu & DLL files) and configuration settings ("Configuration" menu), VCC can be set up to mimic a variety of custom Color Computer 3 setups. I'm sure you will find the settings to match your own desired configuration.

I will briefly step through the menus and show you your choices for configuring VCC to your own custom setup. Any changes you make are remembered and recalled each time you run VCC and any subsiquent changes as well.

VCC's File Menu

There is reall nothing in the *File* menu that is involved in a VCC setup unless you already have a *Custon Config File* saved. I will run throught the *File* menu functions briefly

- **File**
  - ○ **[F9] Run** – Turns on the VCC Color Computer 3's emulation. If you have "Auto Start Emulation" check in the "Config/Misc" tab below, the VCC will stat emulation on startup. Hitting F9 twice has the effect of turning the computer off, then on.
  - ○ **Save Config (New)** – Saves you current configuration to the "VCC.ini" file or you can save a custom ini file of your own to recall at any time.
  - ○ **Load Config (New)** – You can load any custon configuration file you may have saved. As the default, "VCC.ini" is loaded automatically on startup.
  - ○ **[F9] Hard Reset** - Like turning you Coco 3 on or off. This will clear all memoory when used.
  - ○ **[F5] Soft Reset** – The same as hitting the reset button on your Coco 3.

VCC's Edit Menu (new)

The *Edit* menu contains probably the most requested feature for VCC... **COPY & PASTE,** and it's finally HERE!
NOTE: Copy and Paste **ONLY** work on the Coco 3's hardware text screens (both RSDOS and OS-9/NitrOS9). You **CANNOT** copy or paste from/to a Coco 3 graphics screen.

- **Edit**
  - ○ **Copy Text** – Clicking this item imediately copies *ALL* text on the VCC screen into your PC's clipboard for pasting into your PC text documents. (RSDOS & OS-9/NitrOS9)
  - ○ **Paste Text** – Click this item will paste any *raw* data in your PC's clipboard onto your VCC's screen as if yo typed it in. Good for command line entries and works well in most text editor. This function does not work well pasting BASIC listings, see below. (RSDOS & OS-9/NitrOS9)

- ○ **Paste BASIC Code (Merge)** – Clicking this will paste a BASIC listing from your PC's clipboard onto VCC's screen jstas if you typed it, merging your file with any file that may already be in memory. It works very similar to DECB's "MERGE" command. (RSDOS only)
- ○ **Paste BASIC Code (with NEW)** - Clicking this item will first issue a **NEW** command, wiping any program in memory, the pasting the contents of your PC's clipboard. (RSDOS only)

VCC's Configuration Menu

After installation when you start VCC, you will be gretted with the familiar "green screen" and Tandy/Microsoft/ Microware Extended BASIC logo. You have basically started a Tandy Color Computer 3 (Coco3) just as it came out the box with 128k and *nothing* attatched. From here you have *many* choices and they all start with the "Configuration" & "Cartridge" menus at the top of the VCC Window.

First, you want to configure you Coco3 to the basic machine you desire, whether you run RSDOS or OS-9 or both, it really doesn't matter. Maybe you want to emulate your actual Coco3 setup, or maybe you want to setup that *Dream* System you never had. This starts with the "Configuration" menu:

- **Configuration:**
  - ○ **Flip Artifact Colors (New)**
  - ○ **Config: (Tabs)**
    - ▪ **Audio**
    - ▪ **CPU**
    - ▪ **Display**
    - ▪ **Keyboard**
    - ▪ **Joysticks**
    - ▪ **Misc**
    - ▪ **Tape**
    - ▪ **Bitbanger**

Each tab having it's own menu options and settings, which I will highlight below. Through these tabs, you'll configure your Coco3 just like *you* want it. These settings reflect the *internal* settings/expansions of the Coco3. The *external hardware* will be done through the "Cartridge" menu in the next section.

NOTE:
Clicking "Apply" immediately sets the current settings for any altered setting, but remains in the config menu. Clicking "OK" will also set any altered settings but will leave the menu and return to the emulation window. Any feature marked with an "*" will (after one of the above actions) reset the Coco3 emulation to a newly started Coco3 (w/new settings) and all memory contents will be erased. All other functions will not disturb your current machine state.

Flip Artifact Colors (New):

This will "flip" the PMODE artifact colors to the alternate set. Handy for games and such.

CONFIG:

Audio:

This tab can usually be left "as is" as it automatically recognizes your Windows computer defaults and should not have to be changed. However… there are those users who may have more than one audio output or card and may want to change these setting to suit their needs. Personally, as a musician & recording engineer, I have several sound options on my computer which makes this tab useful to me.

- **Sound**
  - **Output Device** – A drop down menu which shows your PC system's choices in sound output devices. Most will use the "Primary Sound Drive" used by most Windows programs.
  - **Sound Quality** – Selects the available sound quality in khz available for you selected device (above). Most will use "44100" khz.

CPU:

This tab allows you to select various Coco 3 system defaults such as memory, CPU type, and Overclocking.

- **Over-Clocking** – The slider will set the speed in which the CPU operates. Minimum is .87mhz/1.778mhz", maximum is 89.4mhz.
- **Memory Size*** - Set the maximum amount of memory your Coco3 will be allowed to use.
  - **128k*** - Emulates a stock Coco3 straight out of the box
  - **512k*** - Emulates the 512k upgrade available through Tandy and many 3rd party vendors
  - **2048k*** - Emulates the "Disto" 2 meg upgrade
  - **8192k* -** Emulates Paul Barden's 8 meg board
- **CPU***
  - **Motorola MC6809 CPU*** - Emulates the stock CPU installed in the Coco3.
  - **Hitachi HD6309 CPU*** - Emulates the aftermarket Hitachi 6309 instruction/speed enhanced CPU.

Display:

Allows you to set your VCC windowed PC display to match your choice of standard Coco 3 display types

- **[F6] Monitor Type:** - Allows you to choose your monitor type.
  - **RGB** – Selects an RGB color monitor suchas the Tandy CM-8.
  - **Composite** – Selects a composite video monitor such as a color TV.
- **Fram Skip** – Number of frames to skip in rendering the Coco3 screen (I've seen no use for this and assume it was from the days of slower computers, we may remove it in the future).
- **Scan Lines** – Allows VCC to "blank" the odd scans to the screen (as a real Coco would).
- **Allow Resize – Allows you to resize the VCC window, otherwise it will be lock in default size.**
- **Force Aspect (New)** – Forces VCC to keep the screen in the proper porportions when resizes "Allow Resize" must be checked for this to work. The "Full Screen" mode will NOT retain it's aspect and will stretch. We are working on this.
- **[F8] Throttle Speed** – Turns off all speed restraints and allows VCC to run at the speed of the host PC, including the keyboard rollover which will cause pressed keys to multiply at unusable rates. Not of much use in my opinion, but each time I mention removing it, people say to keep it.

Keyboard

Allows you to remap your PC keyboard to different keymaps to emulate the layout of a real Coco 3 keyboard, or use it similar to an AT or PS/2 keyboard interface. Custom key mapping coming soon!

- **Keyboard Mapping** – Drop down menu of 4 keymap selections
  - **Coco (DECB)** – The PC keyboard is remapped to emulate the latout of an actual Coco 3 (as much as possible)

- ○ **Natural (OS-9) –** The PC keyboard is mapped "as is" with the exception of a few special keys
- ○ **Compact (OS-9)** – Same as above but mapped with consideration of laptop keyboards with compact key sets
- ○ **Custom** – Not yet available. This is actually being worked on and should be available in the next revision of VCC
- **Keyboard Map Descriptions** – Just a desciption box describing the 4 key map types.
- **Edit Custom Keyboard** – Not yet available

Joysticks

- **Left Joystick Input** – Pushbutton selection, customizes what will define the Left Joystick
- ○ **Audio** – Not available, originally was to be used to emulate 6 bit audio sampling through the Joystick port similar to some Coco software sampling packages.
- ○ **Joystick** – Allows selection of any joystick connected to the PC
- ○ **Mouse** – Selects the PC mouse as your joystick
- ○ **Keyboard** – "JoyKey", Defines keys on the PC keyboard to be used as a 2 button joystick WARNING this will disable the use of these keys while "JoyKey" is in use.
  - **Left** – Select key to define Left joystick movement (default <Left Arrow>)
  - **Right** – Select key to define Right joystick movement (default <Right Arrow>)
  - **Up** – Select key to define Up joystick movement (default <Up Arrow>)
  - **Down** - Select key to define Down joystick move ment (default <Down Arrow>)
  - **Fire 1** – Select key to define Fire button 1 (default <F1 key>)
  - **Fire 2** – Select key to define Fire button 2 (default <F2 key>)
- **Right Joystick Input** – Pushbutton selection, customizes what will define the Right Joystick
- ○ **Audio** – Not available, originally was to be used to emulate 6 bit audio sampling through the Joystick port similar to some Coco software sampling packages.
- ○ **Joystick** – Allows selection of any joystick connected to the PC
- ○ **Mouse** – Selects the PC mouse as your joystick
- ○ **Keyboard** – "JoyKey", Defines keys on the PC keyboard to be used as a 2 button joystick WARNING this will disable the use of these keys while "JoyKey" is in use.
  - **Left** – Select key to define Left joystick movement (default <Left Arrow>)
  - **Right** – Select key to define Right joystick movement (default <Right Arrow>)
  - **Up** – Select key to define Up joystick movement (default <Up Arrow>)
  - **Down** - Select key to define Down joystick move ment (default <Down Arrow>)
  - **Fire 1** – Select key to define Fire button 1 (default <F1 key>)
  - Fire 2 – Select key to define Fire button 2 (default <F2 key>)

MISC

Not much here but Cartridge emulation settings

- **Misc** – Catridge default settings

- ○ **AutoStart Emulation** – Unchecked starts VCC in an "Off" state. You must press "F9" to turn the emulation on. Checked, VCC will start with the power "On".
- ○ **AutoStart Cart** – Unchecked, program paks will not "autostart". This simulates "taping" the cartridge detect pin on a cart. This is useful for using the Orchestra90 cart emulation for stereo sound. I did this same thing on my real Coco to use Orch90 for sound output back in the 80s. Checked, all carts will autostart as normal. When using the MPI (MultiPak Interface) module, the carts will only autostart when the MPI slot switch is set to the slot in which the cart is inserted.

Tape

This tab is for the cassette tape emulation in VCC. As a note, the BASIC
commands "MOTOR ON/OFF" and "AUDIO ON/OFF" do not affect the VCC tape
recorder as on a real Coco.

- **Browse** – this button allows you to *browse* your PC for ".cas" or ".wav" files of recorded cassette tape format programs. The 2 types differ in their formats as follows:
  - ○ **wav** - An 8 bit, 44.199 khz audio recording of the actual cassette file. The file MUST be 8 bit & 44.199 khz (see blow).
  - ○ **cas** - A specially encoded, digital representation of the program data. This format is much more compact than wav and loads much faster.
  - ○ **WARNING: If you use ".wav" files, be warned!! If the ".wav" file is NOT recorded in "44,100 khz, 8-bit", VCC will mangle the file just by loading it, you don't even have to "play" it. Most "standard" wave files are in 16-bit, 44,199 khz. These will not work in VCC and even if you just browse to them and select them into the tape interface, VCC will try to convert them to 8-bit and ruin the file. This issue is being looked into. You were warned.**
- **Record** – Sets the tape emulation to record. The tape will stay on "pause" until you issue a "CSAVE" or "CSAVEM" command from basic.
- **Play** – Sets the tape player to "play" and as above, stays on pause until a "CLOAD" or "CLOADM" command is issued fro BASIC.
- **Stop** – This will stop the recorder just as on a real tape deck. It will not respond to any BASIC commands while stopped. The current "cas" or "wav" file remains in the buffer.
- **Eject** – This button closes then "unloads" the tape image from VCC. You will no longer have access to this image until it's loaded again.
- **Rewind** – This will rewind the "tape" image to the beginning of the tape.
- **Mode –** Displays the current mode of the tape recorder (Play, Stop, Record Etc)
- **Counter** – Displays the position of the tape file just as a real tape counter would. Handy for tape images containing multiple programs

BitBanger

The BitBanger port is included as a means for VCC to print Coco text files to PC text file. There is no form of conversion and all transfers are in raw ASCII text

- **Open** – This button "opens" a selected capture file. Any data sent to the serial port will be sent to this file in raw ASCII format.
- **Close** – This closes the serial capture file.
- **Add LF to CR** – Checking this item will cause VCC to add a Line Feed character to every Carriage Return VCC encounters while sending data to the capture file. PC text files use CR/LF as a standard where the Coco uses CR alone.
- **Print Monitor Window** – Opens an extra window panel on your Windows desktop that will display any data sent to the bitbanger port

VCC's Cartridge Menu

The "Cartridge" selection is an emulation of the Coco's cartridge slot in that here, you will
insert ROM images or special (included) cartridge dll files emulating various Coco controller carts, such
as, the Multipak Interface, Orchestra 90, the FD-502 disk controller, several HD controllers, and the

Becker Port cart. Each cart emulation may or may not add items to the "Cartridge" menu. I will try to list these additions below. The full description of each cart emulation can be found in the "Loadable Modules" section:

Inserting the mpi.dll module into the emulator cartridge slot results in the following menu:
- **mpi.dll** - Tandy Multipak Interface. Allows insertion of up to 4 cart dlls
  - **MPI Slot 4** - Slot 4 of the MPI (duh!)
  - **MPI Slot 3** - Slot 3 of the MPI
  - **MPI Slot 2** - Slot 2 of the MPI
  - **MPI Slot 1** - Slot 1 of the MPI
  - **MPI Config** - Configures the MPI settings
    - **Slot 4** – Browse Cart images to loaded into Slot 4
    - **Slot 3** – Browse Cart images to loaded into Slot 3
    - **Slot 2** – Browse Cart images to loaded into Slot 2
    - **Slot 1** – Browse Cart images to loaded into Slot 1
    - **Slot Select** – Selcts the Slot # to become the *active* Slot (this action may or may not reset the Coco 3 based on what cart is in the slot and your cartridge settings in the Misc tab)
    - **Persistent Pak Images (New)** – Allows VCC to *remember* what carts you have selected and reload them each time you run VCC. This funct is still a little wonky, but works unless you remove the MPI and reinsert it.

Inserting these cart images into the MPI slots or the Cartridge slot results in the following menu additions:
- **fd502.dll** - Tandy FD-502 Disk controller
  - **FD-502 Drive 0** - Insert virtual disk images (.dsk) into the drive slots
  - **FD-502 Drive 1 - "**
  - **FD-502 Drive 2 - "**
  - **FD-502 Drive 3 - "**
  - **FD-502 Config** - Configuration utilities for the FD-502 controller
    - **DOS Image –** Selects what DOS rom is to be used
      - **External ROM Image** – Loads an external DOS ROM (HDBDOS, RGBDOS, etc)
      - **Disk BASIC** – Use the DECB ROM
      - **RGD DOS** – Uses the included RGBDOS ROM for use with "harddisk.dll"
  - **OverClock Disk Drive** – Checked, allows VCC to use it's virtual drives at PC speeds. Unchecked, simulates normal Coco drive speeds. (default is ON)
  - **Persistant Disk Images** – Forces VCC to load the last disk used when booting VCC (default is ON)
  - **Clock at 0xFF50-51** – Moves the RTC port to $FF50-$FF51 for use with some Hard Drive controllers (default is On)
  - **Physical Disks** – Originally meant to use the old FDRAWREAD driver for Windows that allowed VCC to read/write to real Coco floppy disks. I doubt any mnodern PC's can access these disks any longer due to a change in format with the introcuction of the High Density 1.44meg floppy disk. This function most likely be removed in the future.
    - **A** – Selects Physical Drive A
    - **B** – Selects Physical Drive B
  - **External Disk ROM Image** – Selects the External Disk ROM image to use as an alternative Disk Operating System.
    - **Browse** – Browse to select the desired External Disk ROM Image to be used
- **harddisk.dll** - Hard drive emulation
  - **Hard Drive 0** - Insert hard drive images (.vhd) here
- **SuperIDE.dll** - Glenside IDE / SuperIDE emulation for using CF card images
  - **IDE Master** - Insert CF (master) card image here (.img)

- ○ **IDE Slave -** Insert 2nd (slave) CF card image here
- ○ **IDE Config -** IDE interface configuration settings
  - ▪ **Base Address –** Base port address used by the controller. Must NOT conflict with Clock address in FD502 Configuration
  - ▪ **Clock at 0xFF70** – RTC port address. Must NOT conflict with clock address in FD502 Configuration.
  - ▪ **Clock is Read-Only** – Unchecked allow the clock to be set by the OS.
- • **orch90.dll** - Orchestra 90 Music cart
- • **becker.dll** - Becker Port emulation for DriveWire4 and TCP communication
  - ○ **DriveWire Server** - Becker Port configuration for DW4
    - ▪ **Server Address** – DriveWire4 server address#
    - ▪ **Server Port** – DriveWire4 server port#

Help

This menu selection has only one choice:

- • **About -** Displays the VCC "About Box" which contains the "Function Key" list and Copyright information.

Quick Start Configurations

Here are a few "Quick Start Configurations" of some standard Coco 3 setups you can use to get you started

Gaming VCC

A good setup for playing Coco 3 games.

- **Configuration:**
  - **Audio:** Use defualts
  - **CPU:**
    - **Overclocking:** Set to minimum
    - **Memory Size:** 512k (unless game rquires more)
    - **CPU:** Some games will require the HD6309 (Nick Marantez's "GunStar" for example), but most run fine with MC6809
  - **Display:**
    - **Monitor Type:** - Choice based on game requirements, most Coco 3 games will use RGB but some do require CMP
    - **Frame Skip:** Default
    - **Scan Lines:** – Unchecked unless you like the look of blanked odd scanlines
    - **Allow Resize:** - Checked
    - **Force Apsect:** Checked
    - **Throttle Speed:** - Checked
  - **Keyboard:**
    - **Keyboard Mapping:** - Coco (DECB), unless you are more comfortable with a PC keyboard setup, then use "Natural (OS-9)
  - **Joysticks:** - Select the Joysticks of your choice
  - **Misc:**
    - **AutoStart Emulation:** – Checked
    - **AutoStart Cart:** – Checked
  - **Tape:** - Default unless using a tape based game
  - **BitBanger** – Defaults
- **Cartridge:** - Insert "mpi.dll"
  - **MPI:**
    - **Slot 4:** - Insert "fd502.dll"
    - **Slot 1:** - If using a cartridge based game, insert cart here
    - **MPI Config:** – If using a cart based game, move Slot seletor slider to the slot in which the game is inserted, otherwise, set it to Slot 4 for the disk controller
  - **FD502:**
    - **FD-502 Drive 0:** - If using disk based game, insert game disk here unless otherwise instructed by the game. Some games can use Drive 1 as it's "game data" disk.

Press [F9] Twice to start emullation

Orchestra90cc VCC

This setup is for using the Orchestra90cc Program Pak.

- **Configuration:**
  - **Audio:** Use defualts
  - **CPU:**
    - **Overclocking:** Set to minimum
    - **Memory Size:** 512k
    - **CPU:** - MC6809
  - **Display:**
    - **Monitor Type:** - RGB (doesn't really matter)
    - **Frame Skip:** Default

- Scan Lines: – Unchecked unless you like the look of blanked odd scanlines
- Allow Resize: - Checked
- Force Apsect: Checked
- Throttle Speed: - Checked
  - **Keyboard:**
    - **Keyboard Mapping:** - Coco (DECB), unless you are more comfortable with a PC keyboard setup, then use "Natural (OS-9)
  - **Joysticks:** - None
  - **Misc:**
    - **AutoStart Emulation:** – Checked
    - **AutoStart Cart:** – Checked
  - **Tape:** - Default
  - **BitBanger** – Defaults
- **Cartridge:** - Insert "mpi.dll"
  - **MPI:**
    - **Slot 4:** - Insert "fd502.dll"
    - **Slot 1:** - Insert "orch90.dll"
    - **MPI Config:** – set it to Slot 1 for the Orchestra90cc Program Pak
  - **FD502:**
    - **FD-502 Drive 0:** - Insert music disks here

Press [F9] Twice to start emullation

OS-9/NitrOS9 VCC

A basic VCC setup for using OS-9/NitrOS9 with a Hard Drive. I will not go into explaining making a bootdisk for this system, but they can be found on the internet or you can make your own.

- **Configuration:**
  - **Audio:** Use defualts
  - **CPU:**
    - **Overclocking:** Set to minimum unless PC speed is desired (not good for games)
    - **Memory Size:** 2048 (NitrOS9 can transparently use the 2 meg expansion)
    - **CPU:** If using the HD6309, you will want to Use the NitrOS9 6309 boot system, but if using the MC6809, you MUST use a 6809 NitrOS9/OS-9 boot system
  - **Display:**
    - **Monitor Type:** - RGB
    - **Frame Skip:** Default
    - **Scan Lines:** – Unchecked unless you like the look of blanked odd scanlines
    - **Allow Resize:** - Checked
    - **Force Apsect:** Checked
    - **Throttle Speed:** - Checked
  - **Keyboard:**
    - **Keyboard Mapping:** - use "Natural (OS-9) unless you like the old Coco key sequences used in OS-9.
  - **Joysticks:** - Select the Joysticks of your choice
  - **Misc:**
    - **AutoStart Emulation:** – Checked
    - **AutoStart Cart:** – Checked
  - **Tape:** - Default
  - **BitBanger** – Defaults
- **Cartridge:** - Insert "mpi.dll"
  - **MPI:**

- **Slot 4:** - Insert "fd502.dll"
- **Slot 3:** - Insert "harddisk.dll"
- **MPI Config:** – set it to Slot 4 for the disk controller
  - **FD502:**
    - **FD-502 Drive 0:** - Insert OS-9/NitrOS9 boot disk, unless you have a harddrive boot system or are using HDBDOS and DriveWire4. I will not be explaining the OS-9 booting methods as that is available on the internet in many places. (See the "Links" section of this manual).
    - **FD-502 Config**
      - **DOS Image**
        - **External ROM Image** – Use if you want a Custom ROM or if you want DriveWire4 booting, use HDBDOS (explained in the ROM section). Images are loaded in the "Browse" box below.
        - **Disk Basic**- If you are not using DriveWire4, Disk Basic is probably your best choice
        - **RGBDOS –** An alternate DOS which can be used to boot OS-9/NitrOS9
      - **OverClock Disk Drive** – Checked (makes your floppy drives faster, uncheck if you want the old Coco speed drives)
      - **Persistent Drive Images** – (Checked)
      - **Clock at 0xFF50-51** – Checked
      - **External Disk ROM Image** – Use the "Browse" button for loadin any "Custom" ROM is "DOS Image" is set to "External ROM Image"
    - **Hard Drive 0** – If using a hard drive, mount your OS-9/NitrOS9 hard drive image here.
    - **DriveWire Server**
      - **Server Address –** If using DriveWire4, set your DW4 server address here (See DriveWire4 section in this manual)
      - **Server Port** – If using DriveWire4, set your DW4 port setting here (See the DriveWire4 section in this manual.

Press [F9] Twice to start emullation Then type "DOS" to start OS-9/NitrOS9

This about covers the basic VCC setups. These are only basic setups. You will eventually tweak these settings to your preferred system. There are many more option to explore and many more combinations of setups that may suit your needsll of these components and more are explained in more detail in the sections dedicated to those components.

Next we will cover each component in VCC with deatiled explainations of each function as best we can. These components and function allow you to custom configure VCC into the Color Computer 3 system that you remember or maybe the Color Computer system you dreamed of having.
Enjoy!

The MenuBar

The MenuBar resides across the top of the VCC window. Here you'll find the various functions of the VCC emulator. All of VCC's options start here. No *command line* parameters are needed (or wanted).
The Menu choices and their options are:

File, Edit, Configuration, Cartridge, & Help

I will now break down these menus into their various components and attempt to explain their functions.

- File – These options are the basic operating functions of VCC

  - **Run** – If the "Configuration/Config/Misc/AutoStart Emulation" box is unchecked, this selection will start the Coco 3 emulation. If the box is checked, emulation will start when VCC starts and this selection does nothing. Consider it a "Power Button" of sorts. This is also invoked by the "F9" function key on the PC keyboard.
  - **Save Config (***NEW***)** – Disabled (VCC actually auto saves the config file). Originally, Joseph was going to allow you to save and load different Coco 3 configurations. But alas, the feature was never finished. We may implement this feature at a later time.
  - **Load Config (***NEW***)** – Disabled (VCC actually auto loads the config file). We may implement this feature at a later time.
  - **[F9] Hard Reset** – Hitting [F9] twice simulates switching the power to the Coco "Off" then "On" again. You must hit [F9] two times to complete the cycle. Anything in memory will be lost.
  - **[F5] Soft Reset** – Hitting [F5] simulates pressing the "Reset" button on a real Coco 3. The results are the same as a real Coco. Anything in memory will be preserved.
  - **Exit** – This ends the VCC emulation and returns you to Windows. Any unsaved data and or programs will be lost as the program ends with no prompts and control returns to the Windows OS.

- **Edit (***NEW***)** – Editing options for VCC
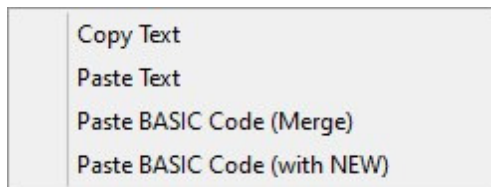
  - **Copy Text:** Copies all text on the VCC/Coco screen (40 & 80 column modes are still being worked on)
  - **Paste Text:** Pastes any text in the Windows clipboard into the VCC keyscan just as you would type it in.
  - **Paste BASIC Code (merge):** Pastes BASIC text to the input buffer acting as a "MERGE" with any existing prorgram in memory.
  - **Past BASIC Code (with NEW):** Pastes BASIC text to the input buffer, but first executing a "NEW" command, clearing the Coco 3 memory of any loaded program.

- **Configuration** – This menu tab contains only one item, but most "configuration" items reside here.
  - o **Flip Artifact Colors (***New***)** – Changes the PMODE artifact color set to the alternate color set without interupting the emulation. Handy for Coco 2 games that come up in the wrong color set.
  - o **Config** – This sub-menu contains a tabbed panel with most of the config items. On each of these panels, you must click "Apply" for your changes to take place. Clicking "OK" will also finalize your choices, but also exits the "Config" menu.
    - **Audio** – This controls VCC's audio emulation.
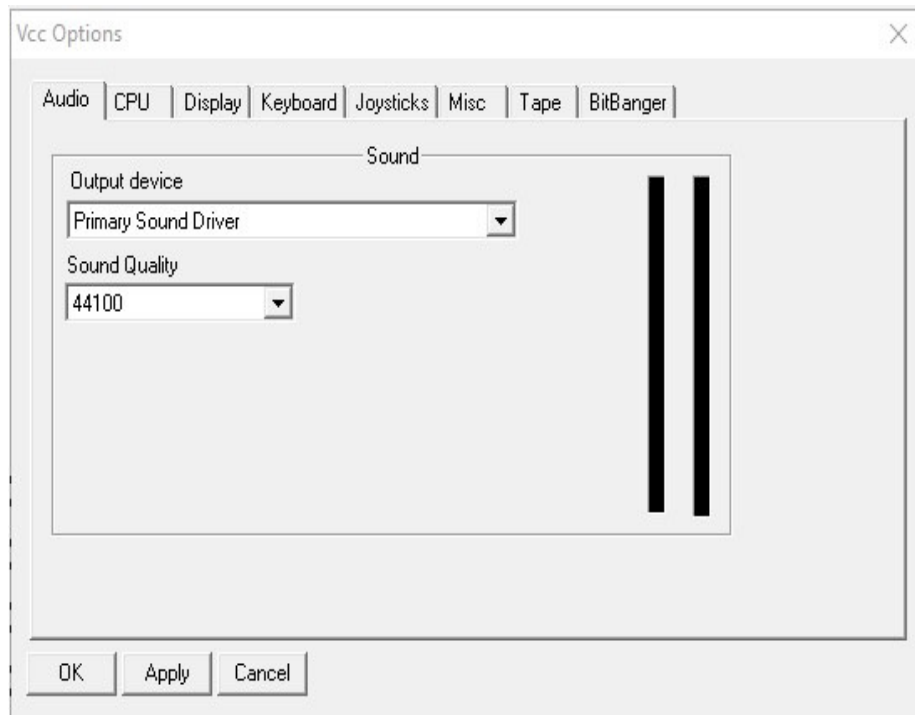
  - o **Sound** – VCC's sound controls
    - **Output Device** – A pull down menu to select the Windows sound device for VCC to use for the Coco's 6-bit DAC and Orchestra 90 sound.
    - **Sound Quality** – Select the frequency resolution for your PC's soundcard (normally 44100).
    - **VU Meters** (vertical bars to the right) – These meters are a visual representation of the sound volume in VCC

- **CPU** – This tab is where you select the CPU type used in VCC and the memory size of your Coco 3 emulation.
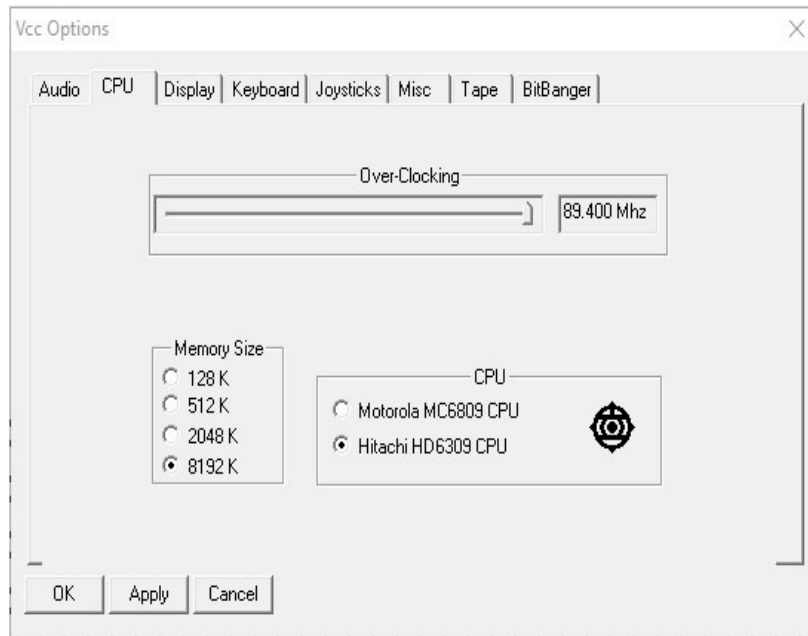
- o **Over-Clocking** – Speeds up the CPU cycle speed but does not affect the internal clock speed. This can speed up text scrolling and graphics functions as well as some aspects of disk access. The slider can be adjusted from 1.78 MHZ (normal Coco 3 speed) to 89.400 MHZ which is ridiculously fast. Over-clocking the CPU to extremely high speeds may not work well on older PCs. Use with caution.
  - o **Memory Size** – Select the memory size for your Coco 3 emulation
    - **128 K** – The stock Coco 3 memory size.
    - **512 K** – Emulates a Coco with the 512k upgrade
    - **2048 K** – Emulates Disto's 2 meg memory board
    - **8192 K** – Emulates Paul T. Barton's 8 meg memory board
  - o **CPU** – This option allows CPU selection for your Coco 3
    - **Motorola MC6809** – The standard Color Computer 6809 CPU
    - **Hitachi HD6309** – Emulates use of Hitachi's HD68B09 CPU. Slightly faster than the 6809 with an enhanced instruction set. This CPU was a popular mod for Coco users in the early 1990s.

- **Display** – Here you set the display type of your Coco 3 emulation.
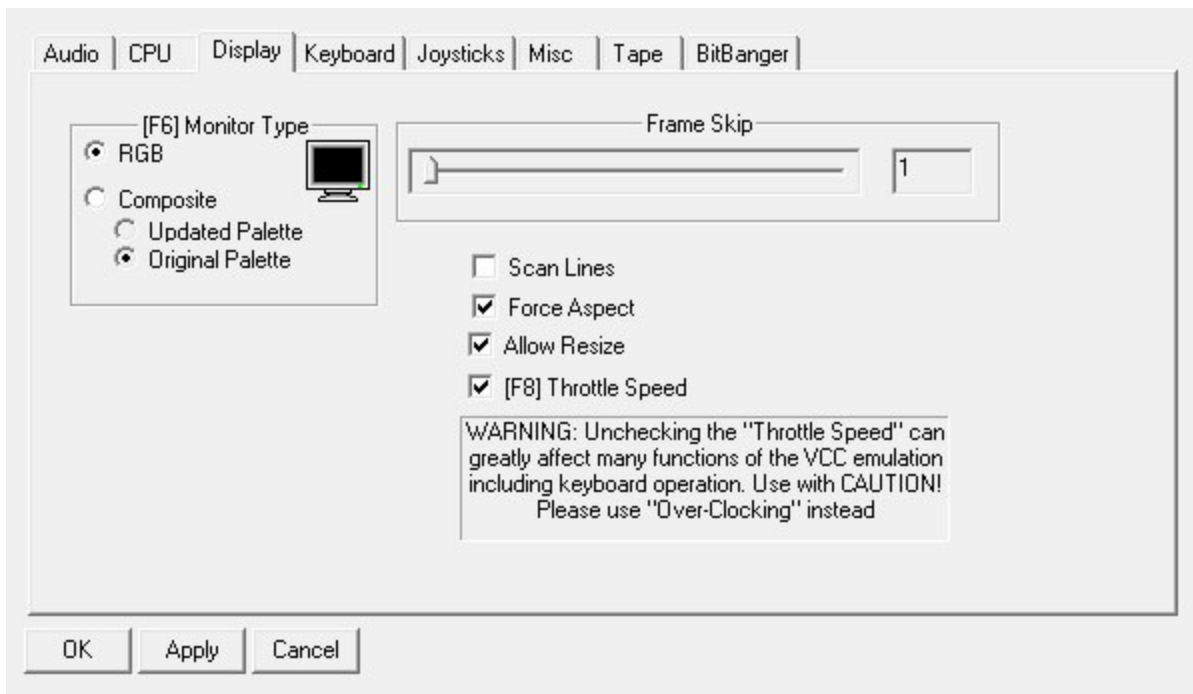
- o
  - **[F6] Monitor Type** – Select the monitor type you want to emulate.
    - **RGB** – Emulates the use of the Tandy CM-8 RGB monitor
    - **Composite** – Emulates the use of a composite monitor or TV. Mostly used when "artifact colors" are desired.
  - o **Frame Skip** – Selects how many frames to skip in the video scans. Mostly used for older, slower PCs with cheap graphics cards.
  - o **Scan Lines** – used to simulate the scan lines seen in a TV display.
  - o **Force Aspect** – We've finally fixed this feature which will maintain the Color Computer's screen aspect when stretched or enlarged. As of this writing, the "Full Screen" mode will NOT retain it's aspect and will stretch the full width of your computer screen. We hope to have this resolved soon.
  - o **Allow Resize** – Checking this option allows you to resize the VCC window, not checked will result in the default window size only. This needs to be checked to allow VCC to be run in a "maximized" window.
  - o **[F8] Throttle Speed** – Un-checking this box allows VCC to run at the full speed of the host PC. This is *not* desirable in most situations, but does have its uses. In most cases, un-checking this will cause *double & triple* keystrokes.

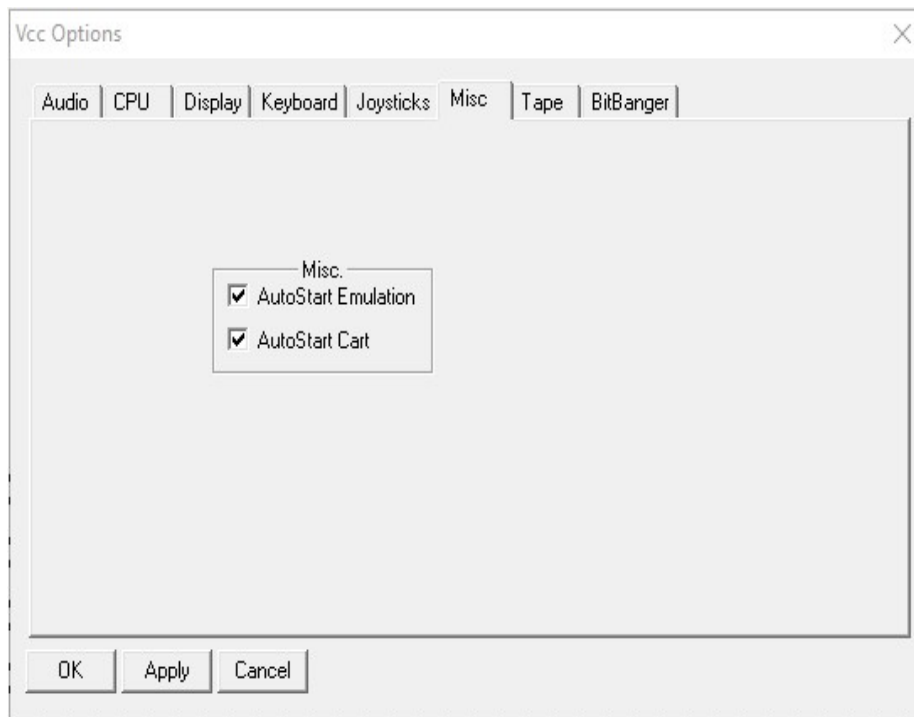- **Keyboard** – Only one selection resides in this tab
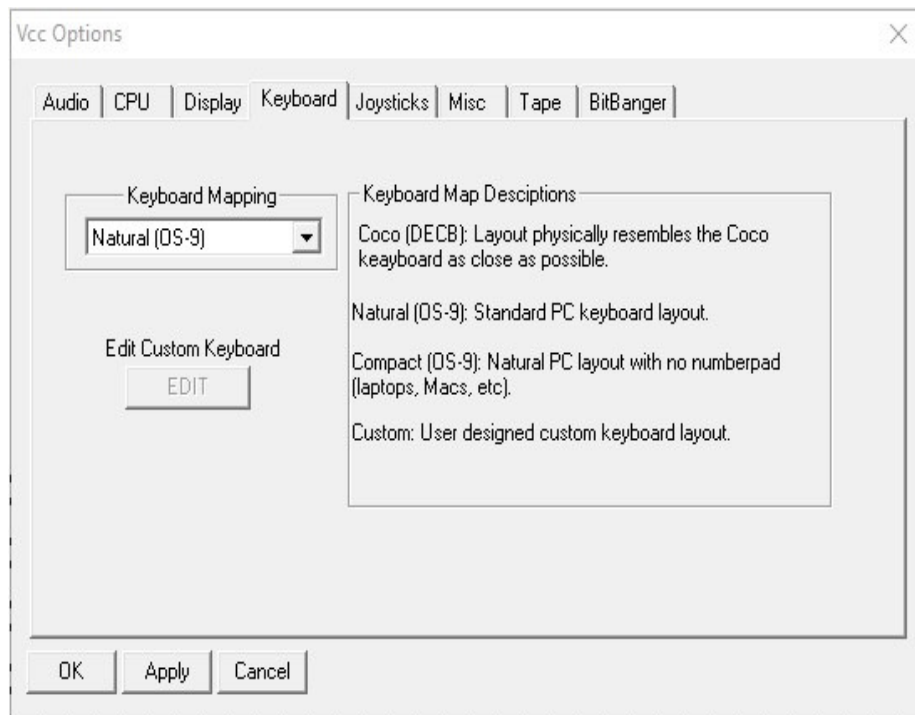
- **Keyboard Mapping** – Here you select "Basic" or "OS-9" keyboard emulation. (*see the "VCC Keyboard Map" section for actual keymaps*)
  - **Coco (DECB)** your PC's keyboard is mapped in a layout similar to an actual Coco 3 keyboard.
  - **Natural (OS-9)** your PC's keyboard is in its "normal" state with the exception of a few special keys (later).
  - **Compact (OS-9)** same as above (PC), but with a few keys altered to compensate for missing keys on laptops and hopefully in the future... IPads and Android devices.
  - **Custom** – Disabled at the moment, but soon to be activated. This feature will load a user created custom keyboard layout.
- **Edit Custom Keyboard –** Also disabled but soon it will allow you to edit the keyboard layout to your liking. You will beable to create multiple custome layouts.

- **Joysticks** – This tab contains controls for using the mouse or keyboard as Coco joysticks.
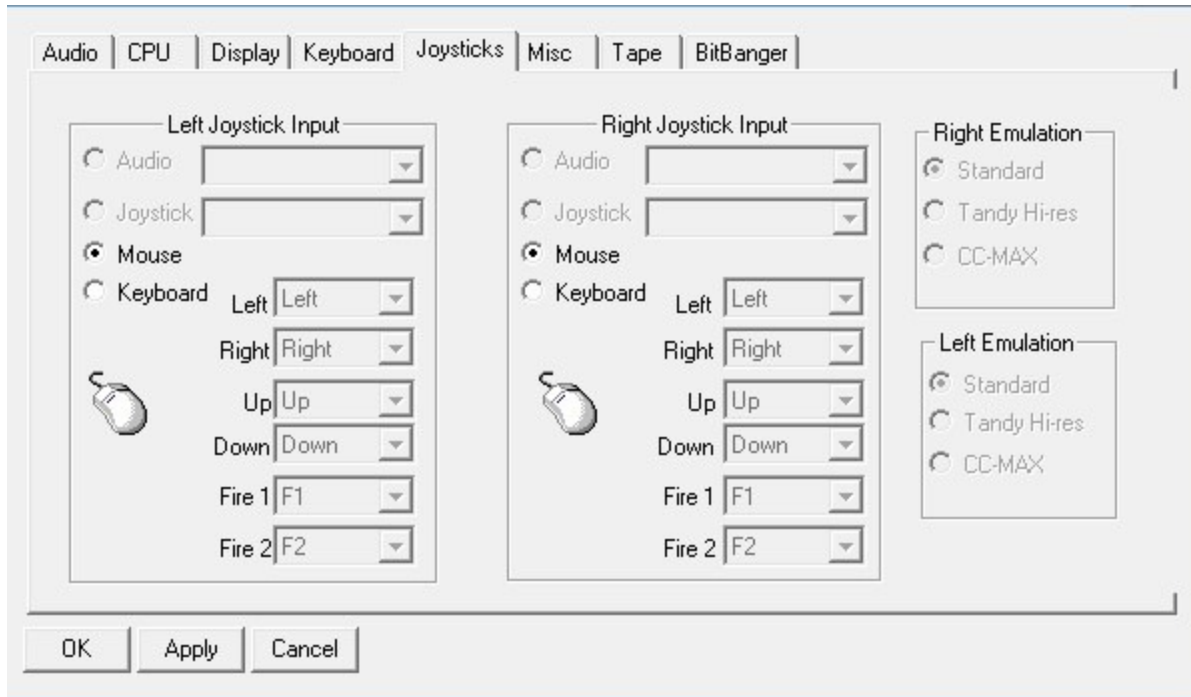
- o
  - Left Joystick Input – Set's the left joystick emulation.
    - Audio – Disabled. In the future, this is hoped to be used as on the real Color Computer 3 as an *audio input* for recording digital 6 bit audio just as on the real machine.
    - **Joystick** – This option will only be available if you have a real PC type joystick plugged into your PC. Your Joystick should available through the pull down menu.
    - **Mouse** – Allows use of the PC's mouse as the left Coco 3 joystick.
    - **Keyboard** – Allows the user to select custom keyboard keys as the various joystick directional controls.
  - Right Joystick Input – Same as above but for the right Coco 3 joystick port.
  - **Right Emulation** – Coming soon (disabled)
  - **Left Emulation** – Coming Soon (disabled)

- **Misc** – Set the cartridge emulation type.

- Misc
  - **AutoStart Emulation** – Unchecked starts VCC in an "Off" state. You must press "F9" to turn the emulation on. Checked, VCC will start with the power "On".
  - **AutoStart Cart** – Unchecked, program paks will not "autostart". This simulates "taping" the cartridge detect pin on a cart. This is useful for using the Orchestra90 cart emulation for stereo sound. I did this same thing on my real Coco to use Orch90 for sound output back in the 80s. Checked, all carts will autostart as normal. When using the MPI (MultiPak Interface) module, the carts will only autostart when the MPI slot switch is set to the slot in which the cart is inserted.

- **Tape** – This tab is for the cassette tape emulation in VCC. As a note, the BASIC commands "MOTOR ON/OFF" and "AUDIO ON/OFF" do not affect the VCC tape recorder as on a real Coco.

- Browse – this button allows you to *browse* your PC for ".cas" or ".wav" files of recorded cassette tape format programs. The 2 types differ in their formats as follows:
  wav - An 8 bit, 44.199 khz audio recording of the actual cassette file. The file MUST be 8 bit & 44.199 khz (see blow).
  cas - A specially encoded, digital representation of the program data. This format is much more compact than wav and loads much faster.
  WARNING: If you use ".wav" files, be warned, if the ".wav" file is *not* recorded in "8-bit, 44,100 khz", VCC will mangle the file just by loading it, you don't even have to "play" it. Most "standard" wave files are in 16-bit, 44,199 khz. These will not work in VCC and even if you just browse to them and select them into the tape interface, VCC will try to convert them to 8-bit and ruin the file. This issue is being looked into. You were warned.
- Record – Sets the tape emulation to record. The tape will stay on "pause" until you issue a "CSAVE" or "CSAVEM" command from basic.
- Play – Sets the tape player to "play" and as above, stays on pause until a "CLOAD" or "CLOADM" command is issued fro BASIC.
- Stop – This will stop the recorder just as on a real tape deck. It will not respond to any BASIC commands while stopped. The current "cas" or "wav" file remains in the buffer.
- Eject – This button closes then "unloads" the tape file from VCC. You will no longer have access to this file until it's loaded again.
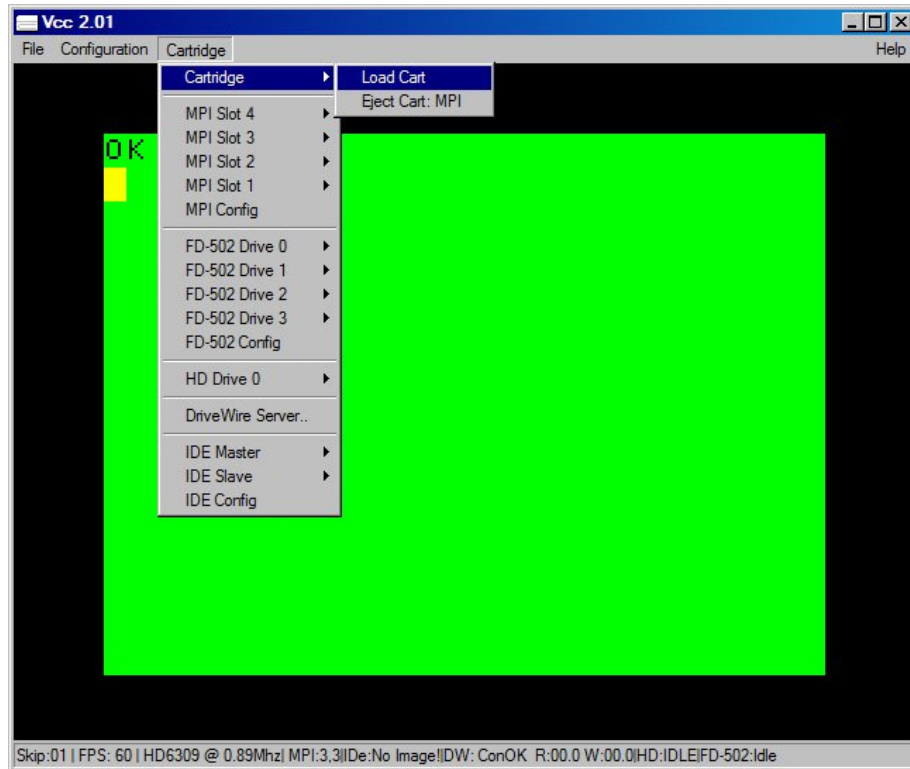- Rewind – This will rewind the "tape" file to the beginning of the tape.

- **BitBanger** – This tab is the emulated "serial" port on the VCC Coco 3.
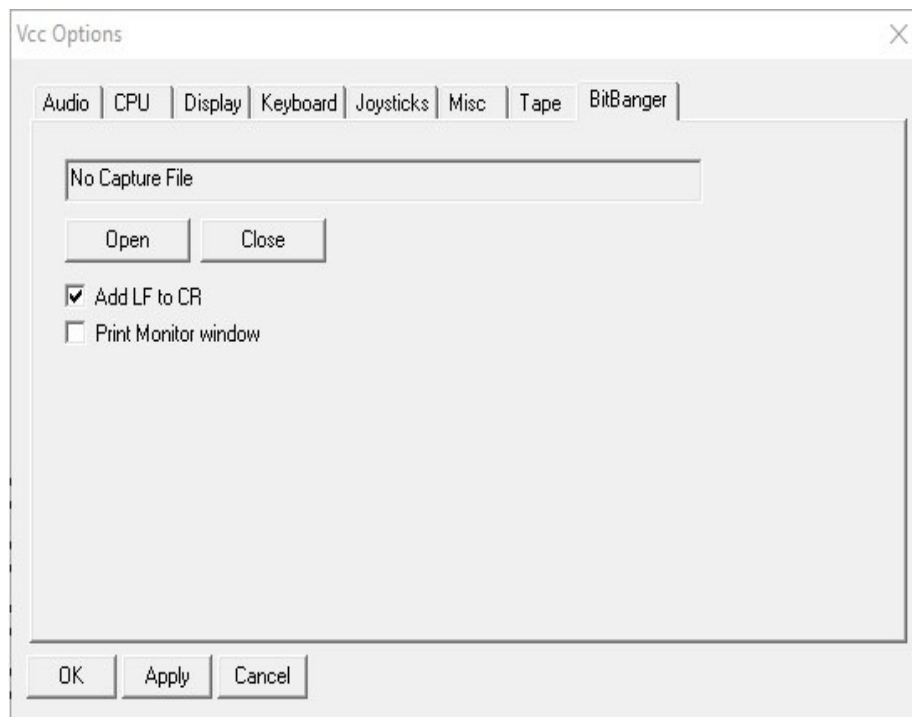
- o **Open** – This button "opens" a selected capture file. Any data sent to the serial port will be sent to this file in raw ASCII format.
- o **Close** – This closes the serial capture file.
- o **Add LF to CR** – Checking this item will cause VCC to add a Line Feed character to every Carriage Return VCC encounters while sending data to the capture file. PC text files use CR/LF as a standard where the Coco uses CR alone.
- o **Print Monitor Window** – Opens an extra window panel on your Windows desktop that will display any data sent to the bitbanger port.

NOTE: The "DriveWire4" server features that were available here in "VCC 1.43b" have now been moved to the "Becker.dll" rom cart emulation. To access the DW4/Becker Port features , you must insert the Becker cart into the cartridge slot or a slot in the MPI (Multi Pak Interface).

- **Cartridge** - The "Cartridge" selection is an emulation of the Coco's cartridge slot in that here, you will insert ROM images or special (included) cartridge dll files emulating various Coco controller carts, such as, the Multipak Interface, Orchestra 90, the FD-502 disk controller, several HD controllers, and the Becker Port cart. Each cart emulation may or may not add items to the "Cartridge" menu. I will try to list these additions below. The full description of each cart emulation can be found in the "Loadable Modules" section:

Inserting the mpi.dll module into the emulator cartridge slot results in the following menu:

- **mpi.dll** - Tandy Multipak Interface. Allows insertion of up to 4 cart dlls
    - **MPI Slot 4** - Slot 4 of the MPI (duh!)
    - **MPI Slot 3** - Slot 3 of the MPI
    - **MPI Slot 2** - Slot 2 of the MPI
    - **MPI Slot 1** - Slot 1 of the MPI
    - **MPI Config** - Configures the MPI settings
        - **Slot 4** – Browse Cart images to loaded into Slot 4
        - **Slot 3** – Browse Cart images to loaded into Slot 3
        - **Slot 2** – Browse Cart images to loaded into Slot 2
        - **Slot 1** – Browse Cart images to loaded into Slot 1
        - **Slot Select** – Selcts the Slot # to become the *active* Slot (this action may or may not reset the Coco 3 based on what cart is in the slot and your cartridge settings in the Misc tab)
        - **Persistent Pak Images (New)** – Allows VCC to *remember* what carts you have selected and reload them each time you run VCC. This funct is still a little wonky, but works unless you remove the MPI and reinsert it.

- *Inserting these cart images into the MPI slots or the Cartridge slot results in the following menu additions:*

- **fd502.dll** - Tandy FD-502 Disk controller
  - **FD-502 Drive 0** - Insert virtual disk images (.dsk) into the drive slots
  - **FD-502 Drive 1 - "**
  - **FD-502 Drive 2 - "**
  - **FD-502 Drive 3 - "**
  - **FD-502 Config** - Configuration utilities for the FD-502 controller
    - **DOS Image –** Selects what DOS rom is to be used
      - **External ROM Image** – Loads an external DOS ROM (HDBDOS, RGBDOS, etc)
      - **Disk BASIC** – Use the DECB ROM
      - **RGD DOS** – Uses the included RGBDOS ROM for use with "harddisk.dll"
      - **OverClock Disk Drive** – Checked, allows VCC to use it's virtual drives at PC speeds. Unchecked, simulates normal Coco drive speeds. (default is ON)
      - **Persistant Disk Images** – Forces VCC to load the last disk used when booting VCC (default is ON)
      - **Clock at 0xFF50-51** – Moves the RTC port to $FF50-$FF51 for use with some Hard Drive controllers (default is On)
      - **Physical Disks** – Originally meant to use the old FDRAWREAD driver for Windows that allowed VCC to read/write to real Coco floppy disks. I doubt any mnodern PC's can access these disks any longer due to a change in format with the introcuction of the High Density 1.44meg floppy disk. This function most likely be removed in the future.
        - **A** – Selects Physical Drive A
        - **B** – Selects Physical Drive B
      - **External Disk ROM Image** – Selects the External Disk ROM image to use as an alternative Disk Operating System.
      - **Browse** – Browse to select the desired External Disk ROM Image to be used
- **harddisk.dll** - Hard drive emulation

  - **Hard Drive 0** - Insert hard drive images (.vhd) here

- **SuperIDE.dll** - Glenside IDE / SuperIDE emulation for using CF card images

  - **IDE Master** - Insert CF (master) card image here (.img)

  - **IDE Slave -** Insert 2nd (slave) CF card image here

  - **IDE Config -** IDE interface configuration settings

- **orch90.dll** - Orchestra 90 Music cart

  - **NONE**

- **becker.dll** - Becker Port emulation for DriveWire4 and TCP communication

  - **DriveWire Server** - Becker Port configuration for DW4

- **Help** - This menu selection has only one choice:

  - **About -** Displays the VCC "About Box" which contains the "Function Key" list and Copyright information.

This concludes the "MenuBar" selections and their uses. I hope my explanations are helpful to the new VCC user.

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 in OS-9's "MouseKey" mode.
- **F3 & F4** – Unused at the moment but soon to be the home of "**CPU-10**" & "**CPU+10**" for setting the CPU speed without having to use the menus.
- **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. Press once for "Off" and again for "On"
- **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.
- **F12** – Unused.

We hope to be introducing a lot of new "Function keys in the future using the <SHIFT> <Fx> format, this open 12 more function keys. If more are needed, we can always resort to <CNTRL><Fx> & <ALT><Fx>.

NOTE: Holding "F1" and pressing "F9" twice will reverse the color artifacting in VCC. If you are getting the wrong colors in Composite mode, then this should help.

VCC Keyboard Layout

When the keyboard layout is set to "Coco (DECB)", the PC keyboard is reconfigured to that of the Coco 3 as close as possible. The "Natural (OS-9)" mode is meant to be the full PC keyboard with a few slight modifications. The "Compact (OS-9)" mode is similar to the Natural mode, but configured for laptop computers with limited keys. The layout looks something like this:

```
PC Keyboard:
[Esc]   [F1][F2][F3][F4][F5][F6][F7][F8][F9][F10][F11][F12]    [Prnt][Scr][Paus]

[`][1!][2@][3#][4$][5%][6^][7&][8*][9(][0)][-_][=+][BkSpc]    [Inst][Hom][PgUp]
[ Tab][Qq][Ww][Ee][Rr][Tt][Yy][Uu][Ii][Oo][Pp][[{][]}][\|]    [Dlet][End][PgDn]
[ Caps][Aa][Ss][Dd][Ff][Gg][Hh][Jj][Kk][Ll][;:]['"][Enter]
[ Shift ][Zz][Xx][Cc][Vv][Bb][Nn][Mm][,<][.>][/?][ Shift ]         [UpA]
[Cntl][Win][Alt][      Space      ][Alt][Win][Prp][Cntl]    [LftA][DnA][RgtA]


VCC Coco (DECB) Keyboard:
[   ][F1][F2][   ][   ][Rst][RGB][   ][Thr][Pwr][StB][FSc][   ]   [   ][   ][   ]

[ ][1!][2"][3#][4$][5%][6&][7'][8(][9)][0 ][:*][-=][BkSpc]    [   ][Clr][   ]
[     ][Qq][Ww][Ee][Rr][Tt][Yy][Uu][Ii][Oo][Pp][[{][]}][\|]    [   ][Brk][   ]
[ Caps][Aa][Ss][Dd][Ff][Gg][Hh][Jj][Kk][Ll][;:]['"][Enter]
[ Shift ][Zz][Xx][Cc][Vv][Bb][Nn][Mm][,<][.>][/?][ Shift ]         [UpA]
[Cntl][   ][Alt][      Space      ][Alt][   ][   ][Cntl]    [LftA][DnA][RgtA]


VCC Natural (OS-9) Keyboard **
[BRK][F1][F2][   ][   ][Rst][RGB][   ][Thr][Pwr][StB][FSc][   ]   [   ][   ][   ]

[`][1!][2@][3#][4$][5%][6^][7&][8*][9(][0)][-_][=+][BkSpc]    [INST][Clr][PgUp]
[ Tab][Qq][Ww][Ee][Rr][Tt][Yy][Uu][Ii][Oo][Pp][[{][]}][\|]    [DEL ][EOL][PgDn]
[ Caps][Aa][Ss][Dd][Ff][Gg][Hh][Jj][Kk][Ll][;:]['"][Enter]
[ Shift ][Zz][Xx][Cc][Vv][Bb][Nn][Mm][,<][.>][/?][ Shift ]         [UpA]
[Cntl][   ][Alt][      Space      ][Alt][   ][   ][Cntl]    [LftA][DnA][RgtA]

VCC Compact (OS-9) Keyboard
[   ][F1][F2][   ][   ][Rst][RGB][   ][Thr][Pwr][StB][FSc][   ]   [   ][   ][   ]

[BRK~][1!][2@][3#][4$][5%][6^][7&][8*][9(][0)][-_][=+][BkSpc][   ][   ][   ]
[ CLR][Qq][Ww][Ee][Rr][Tt][Yy][Uu][Ii][Oo][Pp][[{][]}][\|]    [   ][   ][   ]
[ Caps][Aa][Ss][Dd][Ff][Gg][Hh][Jj][Kk][Ll][;:]['"][Enter]
[ Shift ][Zz][Xx][Cc][Vv][Bb][Nn][Mm][,<][.>][/?][ Shift ]         [UpA]
[Cntl][   ][Alt][      Space      ][   ][   ][   ][Cntl]    [LftA][DnA][RgtA]
```

** Some changes have been made to the "Natural (OS-9)" keyboard layout to better faciliate it's use in NitrOS9 and some text editors. These changes are:

- <END> has been changed from <BRK> to <SHIFT><RIGHT ARROW> to faciliate the "End Of Line" sequence in NitrOS9 and ShellPlus. Also, in the "Ed 3.1" text editor, this key will now produce a true <TAB>.
- <ESC> is now mapped as <BRK>
- <INSERT> has been mapped to <CTRL><RIGHT ARROW>
- <DELETE> has been mapped to <CTRL><LEFT ARROW>
- <PAGE UP> has been mapped to <SHIFT><UP ARROW>
- <PAGE DOWN> has been mapped to <SHIFT><DOWN ARROW>

The Status Line

This is a view of the original VCC 1.4.2 screen showing the "status line".
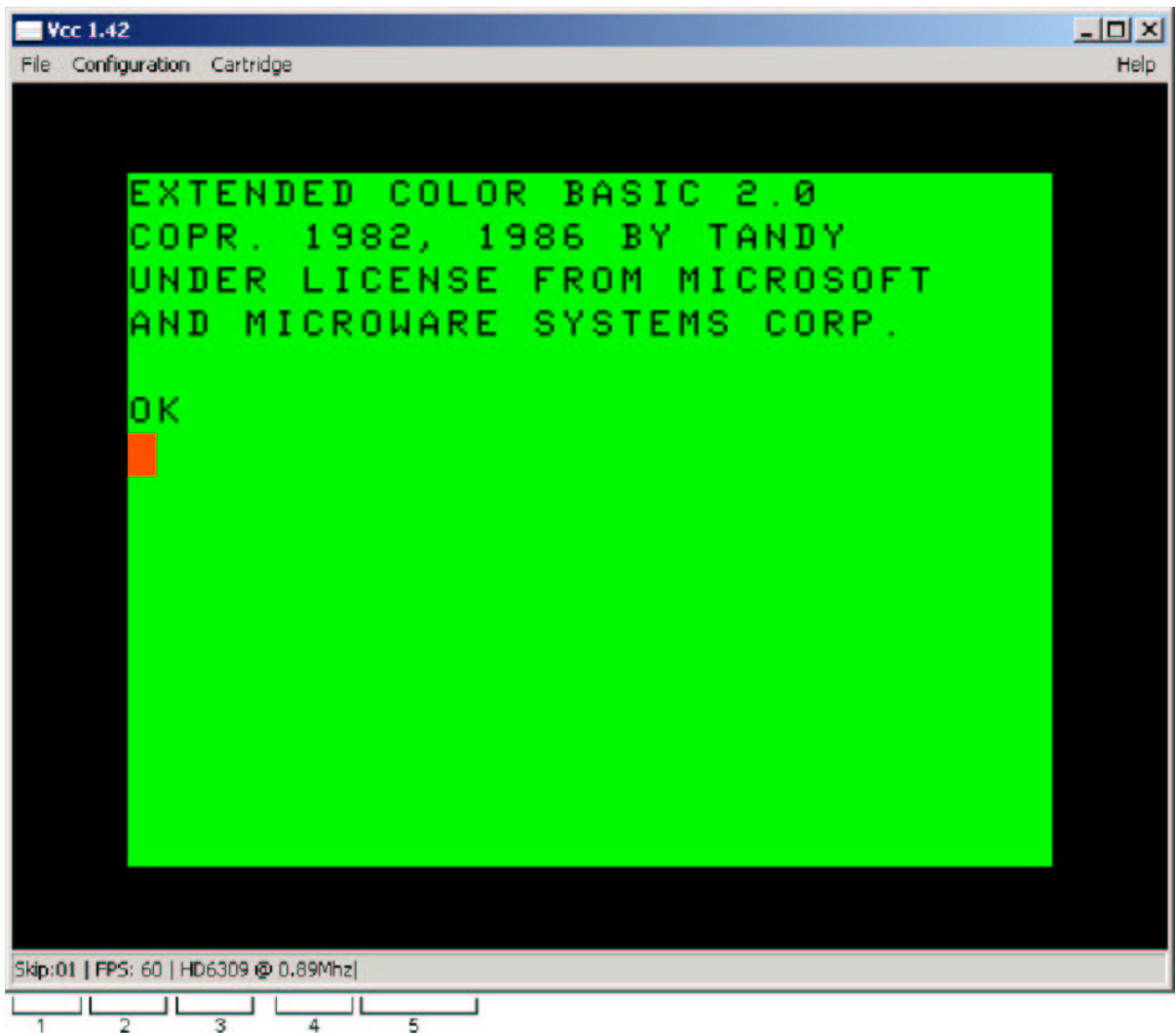
The status line contains useful information about the status of the emulation.

1. Current frame skip setting. It means draw every nth 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. Over-clocking up to 90 MHz is supported.
5. Cartridge data field. Some pluggable carts will also display information on the status line. More on this later.

This is a view of the *new* VCC 2.x.x screen showing the "status line".
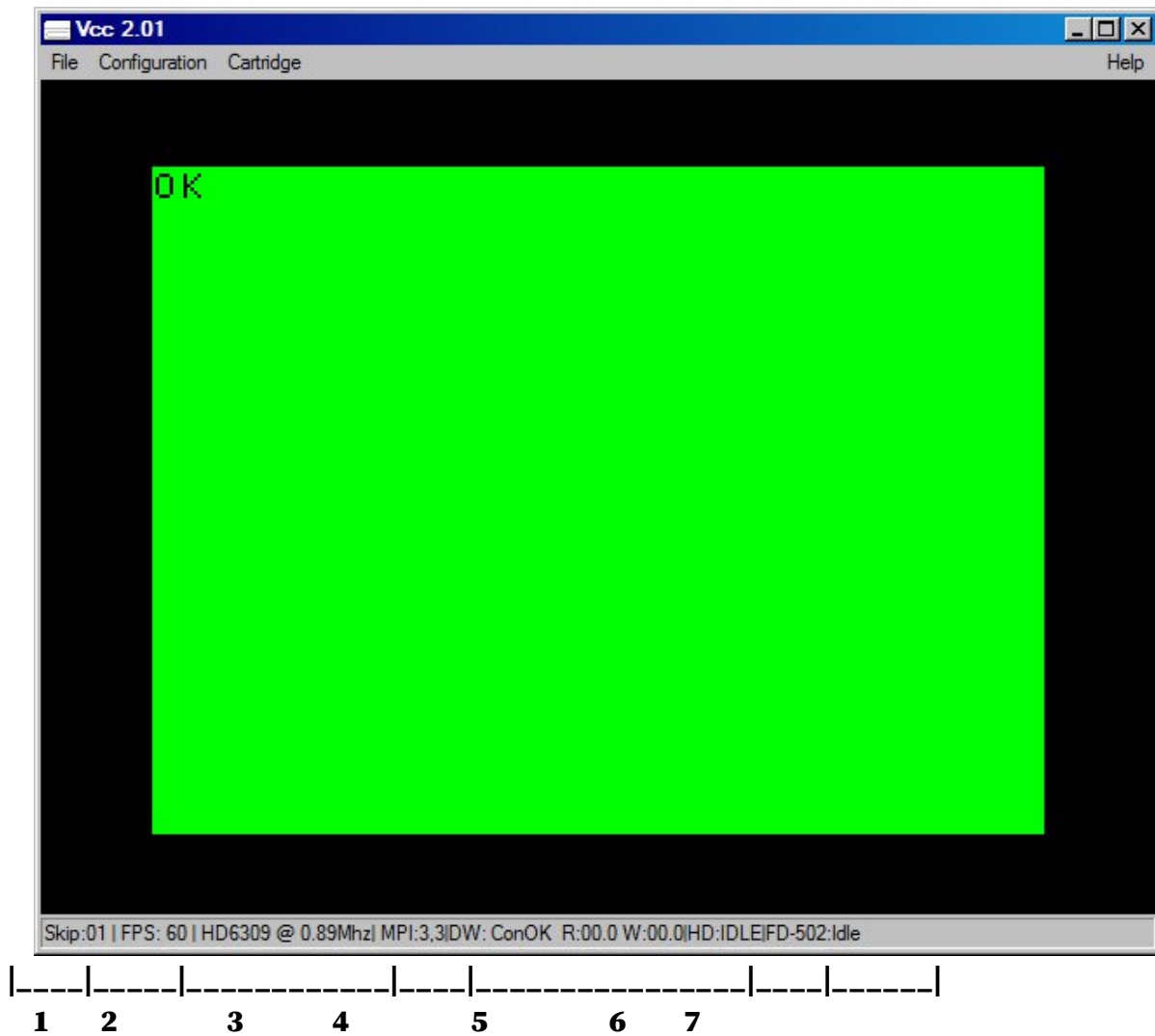
The status line contains useful information about the status of the emulation.

1. Current frame skip setting. It means draw every nth 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 HD6309and clock speed. .89 and 1.79 MHz are stock. Over-clocking up to 89 MHz is supported.
4. Cartridge data field. Some pluggable carts will also display information on the status line. More on this later.
5. DriveWire4 info field. Displays "ConOK" if there is a DW4 connection. The "R" & "W" fields display data as DW4 is accessed (only displays when the "becker.dll" cart is inserted)
6. Hard Drive info is only displayed if the "harddisk.dll" cart is loaded
7. FD-502 info is only displayed if the "fd502.dll" is loaded

Configuring VCC

This section will try to explain the options to configure VCC for normal use.

Setting up VCC 2.0.1 should be as easy as double clicking the VCC setup file icon and following the onscreen prompts. I have tried to make the *default* as close to a "stock" Tandy Color Computer 3, just as you would have bought it in 1986.

Here are the default settings for VCC 2.0.1.

VCC 2.0.1 Defaults:
Configuration/Config

- Audio – Primary Sound Driver. This should default to what ever your PC is using.
- CPU/CPU – Motorola MC6809
- CPU/Memory – 128k (up to 8 meg).
- CPU/Overclocking – 1.788 MHz (stock Coco 3 speed). VCC will actually start at .89MHz with 1.788MHz available through software POKEs.
- Display/Monitor Type – RGB
- Display/Frame Skip – 1
- Display/Scan Lines – Unchecked
- Display/Allow Resize – Checked
- Display/Throttle Speed – Checked
- Keyboard/Keyboard Mapping – Coco (DECB), emulates the Coco 3 keyboard
- Joysticks/Left Joystick - Mouse
- Joysticks/Right Joystick - Mouse
- Misc/AutoStart Emulation – Checked
- Misc/AutoStart Cart – Checked
- BitBanger/Add LF to CR – Checked
- BitBanger/Print Monitor Window – Unchecked

Cartridge

- Cartridge – Empty

This about covers the VCC default settings. You can change most of these settings to suite your needs. The CPU overclocking is particularly handy when running an assembler or compiling "C" code in OS-9. Overclocking also speeds up screen scrolling in text editors as well, but I would not advise using overclocking while running games. It may speed up the emulator enough to make the game unplayable.

Setting Up A "Custom" VCC to Suit Your Needs

Here I will try to explain some of the setup possiblities with VCC. The emulator can be configured many ways and emulate some of the more unique Color Computer 3 settups.

VCC will remember any settings you make and these settings become the "default" fo running VCC in the future.

Using the Multipack Interface

Included with VCC is an emulation of the Tandy Multipack Interface (MPI). To use the MPI, Click the "Cartridge" menu selection at the top of the emulator window. Select "Cartridge" (actually, the only choice initially). Select "Load Cart". If the navigation panel does not start in your VCC installation folder, you must navigate to this folder, usually "C:\Program Files (x86)\VCC 2.01c" (or whereever you installed it). Select the "mpi.dll" and click "Open". On some more *modern* versions of Windows (Vista, 7, 8 & 10), the ".dll" extension may be hidden from view. If this is the case, just select "mpi".

Once you have loaded the MPI, you will notice now that the "Cartridge" menu has expanded. You will see 4 "Cartridge Slots" and an "MPI Config" selection. These "slots" are where you insert the custom "ROM carts" included with the VCC installation or most any of the "ROM carts" you can find and download from any of the Color Computer archive websites. To use a ROM cart, you must *select* the proper slot in the "MPI Config". Just move the "Slot Select" slider to the slot in which you have inserted the cart.

VCC comes with several *custom* ROM carts:
- FD-502 Disk Controller (fd502.dll) - Standard Tandy FD-502 Disk controller with an DECB ROM installed.
- Hard Drive Controller (harddisk.dll) - An emulation of the a typrical MFM HD controller.
- SuperIDE Hard Drive Controller (SuperIDE.dll)* - An emulation of Cloud9's "SuperIDE" HD w/CF cards.
- Becker Port Cart - A custom designed cart of the "Becker Port" for use with DriveWire4.
- Orchestra90cc (orch90.dll) - An emulation of the Tandy Orchestra90cc cart.
- RamDisk (ramdisk,dll)* – An emulation of a 512k RAM card similar to those made by Disto.

"*"=Not currently working, but being worked on.

Several of these carts require an external ROM image to run (included). The ROMs should *autoload* when the modules are inserted into the MPI slot. These carts & ROMS are:
- fd502.dll - disk11.rom, rgbdos.rom, or one of the hdbdos ROMs (explained in the "Loadable Modules" section of this manual).
- orch90.dll - orch90.rom

The "becker.dll", "harddisk.dll", and "SuperIDE.dll" are dependant on whatever ROM image is loaded by the "fd502.dll". This is handled in the "FD502 Config" menu under the "Cartridge" menu.

The "hdbdos" and "rgbdos" ROM uses are best explained in the "HDBDOS User's Manual" available on the Cloud9 website and the "RGB-DOS User's Guide" found in "The Color Computer Archives".

The ROMs included in this installation are specifically written for use with emulators.

Loadable Modules

As stated before the VCC emulator does not know anything about the various peripherals that were available. It depends entirely on runtime 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 7 modules included. They are:

"mpi.dll", "orch90.dll", "harddisk.dll", "Superide.dll", "fd502.dll", "becker.dll" and "ramdisk.dll". These are installed in VCC's home directory "C:\Program Files\VCC 2.01" by default.

Any "ROMs" needed by these cart emulations are loaded externally and are included in the installation. The roms are stored in the same folder as the VCC executable and DLL files, usually "C:/Program Files/VCC 2.01"

orch90.dll

This is the simplest module. It emulates the Orchestra-90 program pack. It has no menu options and returns no status. The sound from the Orchestra-90 pak will play in full stereo through your PC's speakers. To use the Orchestra-90 as the original, you must have the Orchestra-90 rom in the VCC installation folder along with the other roms. The "Autostart Cart" checkbox must be checked in the "Config/Misc" menu and if using the MPI, the MPI selector switch must be set to the same slot as the Orch90 pak. Hit "F5" to reset the emulator and start the cart.

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). It uses 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.
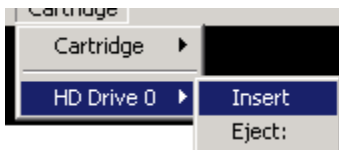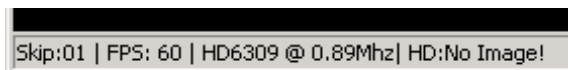

HD: [STATUS] will be added to the status bar and will change during the following events:

No Image!, Indicates that no VHD image has been selected.

Idle, VHD image is loaded but not being accessed.

RD: ###### or WR: ###### a sector is being read or written.

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 there 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.
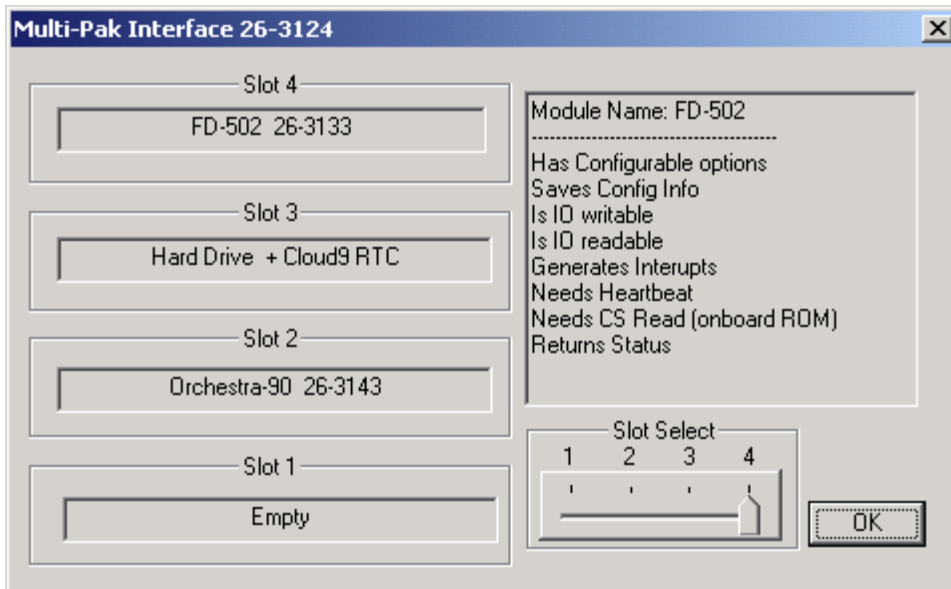
This module will add an MPI item to the status line. Following this are two numbers. The first is the slot the Chip select signal is routed to, The second is the destination of the Spare Select line. Following this is any status info returned from loaded modules. In the example above the MPI is set for 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:
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 (with the extension ".dsk").
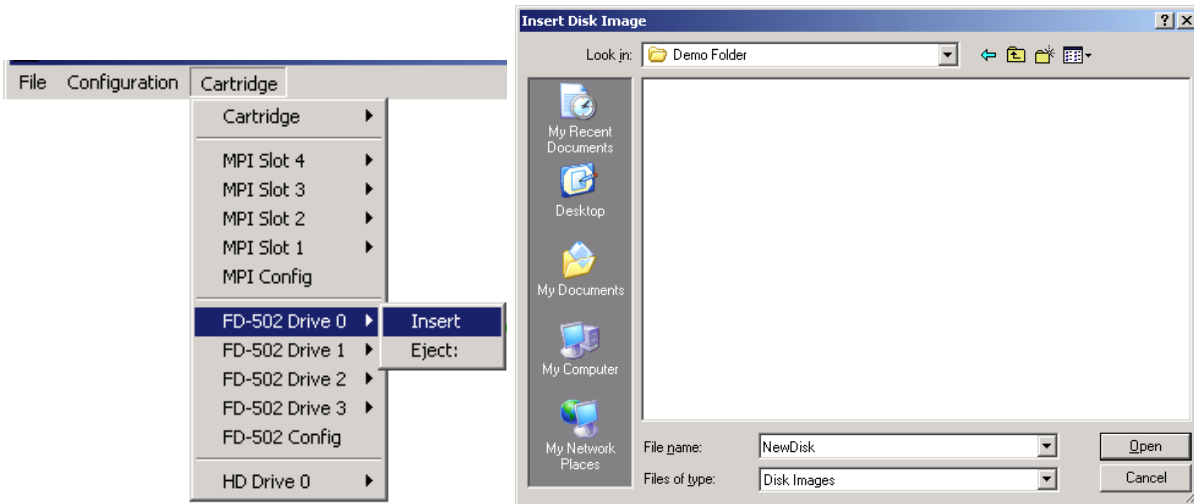
The following dialog will appear.
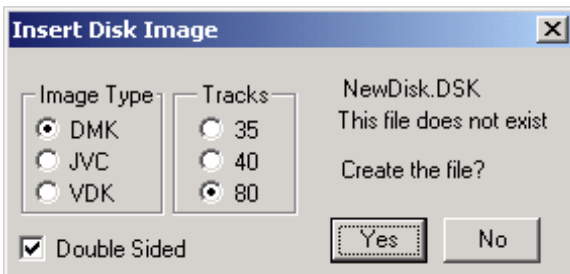
Just select the parameters for the new image and click yes. The image will be created, Zero filled and mounted in the drive you've selected. Note that unlike most other emulators the act of creating a new **disk will NOT format it**. Just as you would with a real disk you must either DSKINI (DOS) or format (OS9 et al) the image before it can be used.
For most disk use, you will want to use the JVC as it is the format most emulators use. The DMK and VDK formats are not fully implemented and in some cases, cause problems.

The configuration dialog:
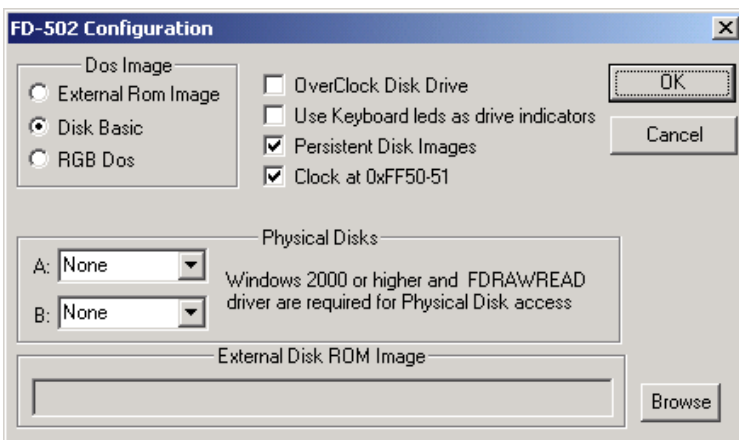
Dos Image:
There are two built in DOS ROMS to choose from. Disk Basic is the standard DOS 1.1 that shipped with the FD-502 controller. RGB Dos is a version of DOS modified by Robert Gault to take advantage of the virtual hard disk emulation described elsewhere in the document. External Rom Image is just that. If you have a dump of a custom version of DOS (ADOS3 or HDB-DOS 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. To use the Becker Port cart and DW4, you will need to use this feature to mount the "hdbdw3bck.rom" image, and check "External Rom Image".

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 over-clocking option and experience I/O errors try turning this on.

Use Keyboard LEDs as drive indicators: 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. I.E. 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. This is also true with any hard drive controller modules and VHD files.

Clock at 0xFF50-51: Selected whether the Disto RTC will be enabled or not.

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/ (I don't think the website exists anymore)

Download and run the fdinstall.exe file.
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.

DISCLAIMER
* We have no association with either the Glenside computer club or Cloud-9.
They do not endorse or support this program in any way. If you experience any issues with this software please DO NOT bother either of these entities. J.F.

SuperIDE.dll:

NOTE: At the moment, this cart does not seem to function properly. The issue is being investigated. The SuperIDE worked before conversion to VS2015, so it's most likely a syntax problem and should be fixed soon.

This is simulation of the Glenside IDE and Cloud9's SuperIDE controller with the Dallas DS1315 RTC. It provides no more functionality than RGB-DOS and the hard disk module that are included with VCC. Its primary purpose is to allow owners of the Glenside IDE or Cloud-9 Super-IDE Controller to mount and run a backup of their CF card under VCC without any driver modifications. As such it is compatible the both HDB-DOS and the Super-Driver currently in the Toolshed and NitrOS9 repositories. It adds three menu items used to Insert/Eject backup images (.IMG files) created by my sideutil program and to set the base address of the virtual controller. Also note that currently the sideutil program only allows dumping of removable CF cards, Hard disks are not currently supported for safety reasons.

Base Address:

This must be set to match the address of the real hardware the CF card image was taken from.

FF40 Do not use this address with the FD-502 module loaded as it uses part of this range as does the becker.dll.

FF50 Default base address. If selected verify the RTC in the FD-502 module is disabled as it uses part of this range.

FF60

FF70 Also used by the Modules RTC, Selecting this base will disable the internal RTC.

Clock at 0xFF70:

Enables the DS1315 RTC at this address. It will be unavailable if the IDE base address is set to 0xFF70.

Clock is Read-Only:

If this option is not selected changing the time in the emulator will affect the time of the host computer.

becker.dll

The "Becker Port" cart emulation has been added to allow the user to utilize the features of the DriveWire4 file server. DriveWire4 can be used for multiple dsk/vhd access, TelNet, modem emulation (through telnet), internet FTP access, DriveWire MIDI through the PC host's soundcard, virtual printing, and much, much more!

DriveWire4 (DW4) has opened the doors to the outside world to the Coco and now it's available for VCC 2.0.1. To use DW4 with VCC 2.0.1c, you must insert the "becker.dll" into one of the MPI slots, but DO NOT SELECT THE SLOT. You will need to use the "External ROM" feature of the FD-502 controller to load the becker version of HDBDOS.

The becker.dll cart will add another menu option to the "Cartridge" menu:
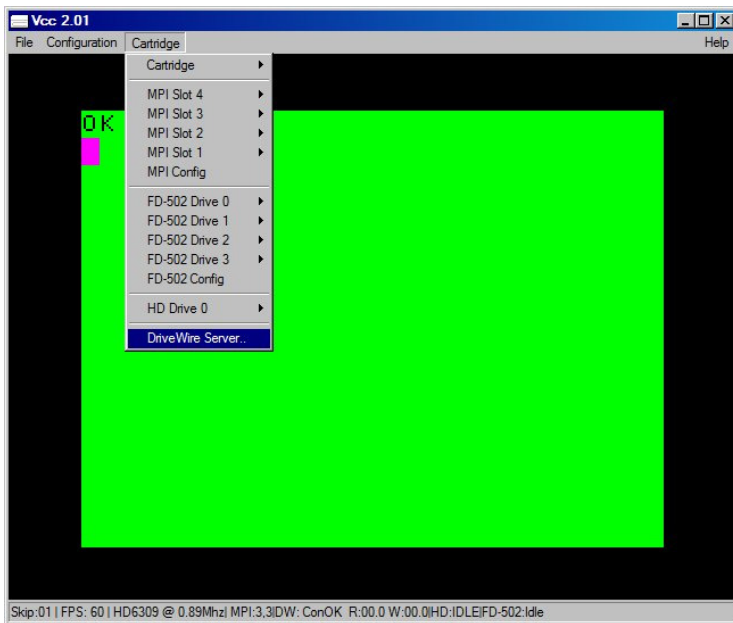
This menu option allows the setting for the Becker port to be changed to fit the user's needs. In most cases, the default settings are what is needed and it should not have to be changed.

The default settings are:
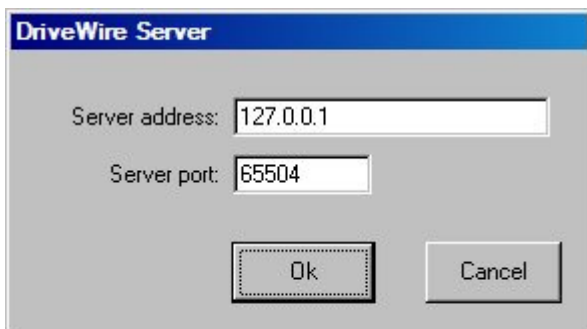
These settings should be sufficient for most DriveWire4 uses. In the DriveWire4 server "Simple Config Wizard" utility, you will use the "Emulator or other TCP/IP" selection and use the default settings from there.

To use DW4 from BASIC, you must have the "Cartridge/FD502 Config/External Rom" selected and the browse to your HDBDOS rom in the "Browse" box. This rom is also used when "auto booting" NitrOS9 from an HDBDOS virtual floppy drive using Robert Gault's RGBDOS menu system. If you are using "multi-partitioned" VHD images with NitrOS9 and HDBDOS images on one VHD, then you will have to use an "offset" in HDBDOS to reflect the size of the NitrOS9 partition to get to the HDBDOS partition. An explanation to this offset can be found on Robert Gault's website @ http://aaronwolfe.com/robert.gault/Coco/Downloads/Downloads.htm.
Scroll to the bottom of the page for RGBDOS. Almost all that applies to RGBDOS will also apply to HDBDOS (in most cases).

ramdisk.dll

This cart image emulates a 512k. RAM cart. It's very simple to use and can give you 512k of extra data storage. To use the RamDisk cart, you use a 24 bit addressing mode (3 bytes) and read/write single bytes of data to/from the cart.

To set the address in the cart of the data byte you want to read or write, just poke your 24 bit address variable like this:

The address range of the cart is $000000 - $80000 or 0 – 524,288

To read/write to/from address $6F4972 you would poke:
POKE &HFF40,&H72   = lswlsb of address
POKE &HFF41,&H49   = lswmsb of addresses
POKE &HFF42,&H6F   = mswlsb of address

To read the above address once set:
AA=PEEK(&HFF43)

To write &H55 to the above address: (overwrites any existing data at that address)
POKE &HFF43,&H55

A NitrOS9 driver for this cart would be a very simple project.

System ROMs:

This installation of "VCC 2.0.1" comes with the following ROM images installed in the VCC installation folder along with the emulator and it's supporting software:

"coco3.rom" - A combined copy of the original Tandy "Color BASIC", "Extended Color BASIC", and "Super Extended Color BASIC" ROMs installed in the Color Computer 3. This ROM was compiled from the "ToolShed" repository sources and not "ripped" from a real Color Computer 3.

"disk11.rom" - A copy of the "Disk Extended Color BASIC" ROM (v1.1) supplied with the Tandy "FD-502" disk controller. This ROM was compiled from the "ToolShed" repository sources and not "ripped" from a real FD-502 disk controller.

"rgbdos.rom" - A copy of the RGBDOS ROM by Robert Gault and downloaded from his website. This ROM has "offsets" for dual partitioned VHD images with NitrOS9 and RSDOS partitions. The offsets are set at "$5A000" for *standard* VHD images that use a 90 megabyte OS-9 partition and and RSDOS partition with 256 virtual drives. To change this ROM for use with *single partition* VHD images, (in RSDOS) use:

> POKE&HD938,0:POKE&HD939,0:POKE&HD93A,0

 Or in a Windows hex editor, you can change the values permanently by changing the ROM image values directly at locations $1938, $1939, and $193A, all to $00.

"hdbdw3bc3.rom" - Standard "Becker Port" HDBDOS ROM. Compiled from the HDBDOS sources at the ToolShed repository. Not to be used with *dual partitioned* VHD images.

"hdbdw3bc3 w-offset 5A000.rom" - Standard "Becker Port" HDBDOS ROM. Compiled from the HDBDOS sources at the ToolShed repository, but modified for using dual partitioned VHD images for OS-9 and RSDOS. The offsets are set at "$5A000" for *standard* VHD images that use a 90 megabyte OS-9 partition and and RSDOS partition with 256 virtual drives.

"hdbdw3bck.rom" - "Becker Port" HDBDOS ROM without the *speedup* (2 mhz) poke installed. This ROM allows time critical Games and Apps that do not run well (or run too fast) with the speedup poke. Compiled from the HDBDOS sources at the ToolShed repository. Not to be used with *dual partitioned* VHD images.

"hdbdw3bck w-offset 5A000.rom" - "Becker Port" HDBDOS ROM without the *speedup* (2 mhz) poke installed. This ROM allows time critical Games and Apps that do not run well (or run too fast) with the speedup poke. Compiled from the HDBDOS sources at the ToolShed repository, but modified for using dual partitioned VHD images for OS-9 and RSDOS. The offsets are set at "$5A000" for *standard* VHD images that use a 90 megabyte OS-9 partition and and RSDOS partition with 256 virtual drives.

The RGBDOS and HDBDOS ROMs offsets can be changed to match any size VHD images with any size partitions using the methods used above for the RGBDOS ROM. The same ROM addresses apply to HDBDOS. To calculate the proper offset for your VHD partition, I suggest using Robert Gault's "SPECS.BAS" program found on "RGBDOS Tools Disk image" found on his website (listed above). This program will calculate the size of your OS-9 partition (which must come first) and give you the proper offset to the RSDOS partition. The SPECS.BAS program can be found on the "Tools.dsk" in the RGBDOS zip available for download on his site.

The VCC Command Line Options:

VCC *does* allw a couple of command line options. One has been available since at least VCC 1.42, and the other is a new addition in this release.

NOTE: Command line options must be used in a Windows "Command Prompt". You must CD to the location of the VCC installation directory or use the complete file path to "VCC.exe" before using any command line options.

File Load – The "File Loading" function has been here all along but was never mentioned in previous manuals. In fact, I didn't even know this function worked until someone pointed it out to me recently. First, the filenames must be in UTF(8) format and they must also reside in the VCC program folder (hopefully, this will change in the future). Only ROM, CCC, Pak, and BIN files may be loaded (also to be changed). To use the file load function you must use the command line as follows:

vcc.exe filname.bin
vcc.exe filename.rom
vcc .exe filename.ccc
vcc.exe filename.pak

Alternatively, if you double click on any of these file types in Windows File Explorer (if already assosiated with another program, you may need to right click, select "Open With", then assosiate this extension to VCC.exe), then assosiate these extensions with VCC, they will open VCC and run, from any directory.

Custom VCC.ini Load – You can now load a "custom" vcc.ini file from the command line in a Windows Command Prompt. You must first have created such a file from manually editing a vcc.ini file and saving it under a different name (w/the ini extesion), or havin "Saved" the custom ini file from the "File/Save Config" feature (below). O load a custom ini from the cmd line:

VCC.ini -i IniFileName.ini

Loading and Saving Custom VCC.ini Files
Alternately, you can Load/Save custom ini files now from the "File" menu while VCC is running. To save a custom ini file, just click "File/Save Config", type in a new custom.ini name, and click Save.
To load a custom ini file (you must have created one first), Click "File/Load Config" , click the custom.ini file you want to load, then click Load. For your new configuration to boot, you'll need to hit F9 twice.
NOTE: Before creating a "custom" version of VCC, I suggest you "Save" the current configuration under the new custom name first. This is because VCC writes many of the changes to an ini file as you go, so any changes you make will be saved under the current name. VCC starts under the "vcc.ini" file, so I suggest leaving that file "plain", saving under a custom name, then make your changes. Most changes will be save to the "new" file, but don't forget to "Save Config" when you're done just to make sure. When you exit VCC, the current name is used to save the final state of your configuration.

The VCC.ini File Format

The "vcc.ini" and any custom "ini" file you may save, will be stored in you "user/appdata/roaming/vcc" folder. The ini files are kept in a strict format and users must use cation if editing these files manually or you may ruin you VCC configuration and have to redo it from inside VCC.
The format is as follows:

[Version]
Release=VCC x.x (x=version #)

[CPU]
DoubleSpeedClock=x (x=speed value 0-255)
FrameSkip=x (x=1-6)
Throttle=x (x=0-1, 0=off, 1=on)
CpuType=x (x=0-1, 0=MC6809, 1=HD6309)
MaxOverClock=x (1-256)

[Audio]
SndCard=SndCardName
Rate=x (x=0-3, 0=none, 1=11,025khz, 2=22,050khz, 3=44,100khz)

[Video]
MonitorType=x (x=0-1, 0=RGB, 1=CMP)
Scanlines=x (x=0-1, 0=off, 1=on)
AllowResize=x (x=0-1, 0=off, 1=on)
ForceAspect=x (x=0-1, 0=off, 1=on)

[Memory]
RamSize=x (0=128k, 1=512k, 2=2048k, 3=8192k)
ExternalBasicImage=FilePath\FileName (not normally used)

[Misc]
AutoStart=x (x=0-1, 0=off, 1=on)
CartAutoStart=x (x=0-1, 0=off, 1=on)
KeyMapIndex=x (0-3, 0=DECB, 1=OS9, 2=Compact, 3=Custom)

[Module]
OnBoot=FilePath\ROM\CCC\DLL\PAKFilename

[LeftJoyStick]
UseMouse=x (x=0-1, 0=off, 1=on)
Left=x (x=keycode)
Right=x (x=keycode)
Up=x (x=keycode)
Down=x (x=keycode)
Fire1=x (x=keycode)
Fire2=x (x=keycode)
DiDevice=x (x=DeviceName)
HiRezDevise=x (0-2, 0=none, 1=RS HiRez, 2=CocoMax)

[RightJoyStick]
UseMouse=x (x=0-1, 0=off, 1=on)

Left=x (x=keycode)
Right=x (x=keycode)
Up=x (x=keycode)
Down=x (x=keycode)
Fire1=x (x=keycode)
Fire2=x (x=keycode)
DiDevice=x (x=DeviceName)
HiRezDevise=x (0-2, 0=none, 1=RS HiRez, 2=CocoMax)

[MPI]
SWPosition=x (x=0-3 slot number)
SLOT1=PakFilePath\PakFileName
SLOT2=PakFilePath\PakFileName
SLOT3=PakFilePath\PakFileName
SLOT4=PakFilePath\PakFileName

[Hard Drive]
VHDImage=VHDFilePath\VHDFileName

[FD-502]
DiskRom=x (x=0-2, 0=ExternalRom, 1=RSDOSRom, 2=RGBDOSRom)
RomPath=DiskRomPath
Persist=x (x=0-1, 0=off, 1=on)
Disk#0=DiskFileName1
Disk#1=DiskFileName2
Disk#2=DiskFileName3
Disk#3=DiskFileName4
ClkEnable=x (x=0-1, 0=off, 1=on)
TruboDisk=x (x=0-1, 0=off, 1=on)

[Hard Disk]
VHDImage=VHDFilePath\VHDFileName

[DefaultPaths]
FloppyPath=FloppyPathName
HardDiskPath=HardDiskPathName
MPIPath=MPIPathName
CassPath=CassPathName
PalPath=PakPathName

[HDBDOS/DW/Becker]
DWServerAddr=x.x.x.x (x=Server IP addresss i.e. 127.0.0.1)
DWServerPort=x (x=server port # i.e 65504)

[Glenside-IDE /w Clock]
Master=IDEMasterImageName
Slave=IDESlaveImageName
BaseAddr=x (x=0-3, 0=40, 1=50, 2=60, 3=70)
ClkEnable=x (x=0-1, 0=off, 1=on)
ClkRdOnly= (x=0-1, 0=off, 1=on)

Credits:

First and foremost, I would like to thank Joseph Forgeone, the original author of the VCC Color Computer 3 emulator. Without all his hard work, this wonderful emulator would have never come about.

I would like to thank Robert Gault for his contribution of the "RGBDOS" implementation and all the testing and suggestions during Joseph's development of VCC. Robert has contributed more to the Coco Community than anyone could know.

I would like to thank Aaron Wolfe and David Ladd for the addition of the "Becker Port" for using VCC with DriveWire4. This has allowed VCC to communicate to the outside world and has opened many doors that were closed before.

I would like to thank Gary Colbourn for his work in converting Joseph's Visual Studio 6 C++ code to compile in Visual Studio 2015 Community Edition.

I would like to thank Wes Gale for his bug hunting and squashing during the initial release of VCC to open source.

I would like to thank Walter Zambotti for his contribution in fixing the HD6309 CPU to funtion properly and adding in the missing instructions.

I would especially like to thank James Rye, Peter Westburg, and EJ, Jaquay for their endless work on the enhancements and bug fixes of the lat couple of releases. Without them, I would not be making this release.

Last but not least, I would like to thank everyone in the Coco Community as the help and support of the Coco Community is what has kept the interest in the Color Computer going strong and made VCC possible in the first place!

Let's make VCC the BEST Color Computer 3 emulator ever!
Bill Pierce 1/19/2020