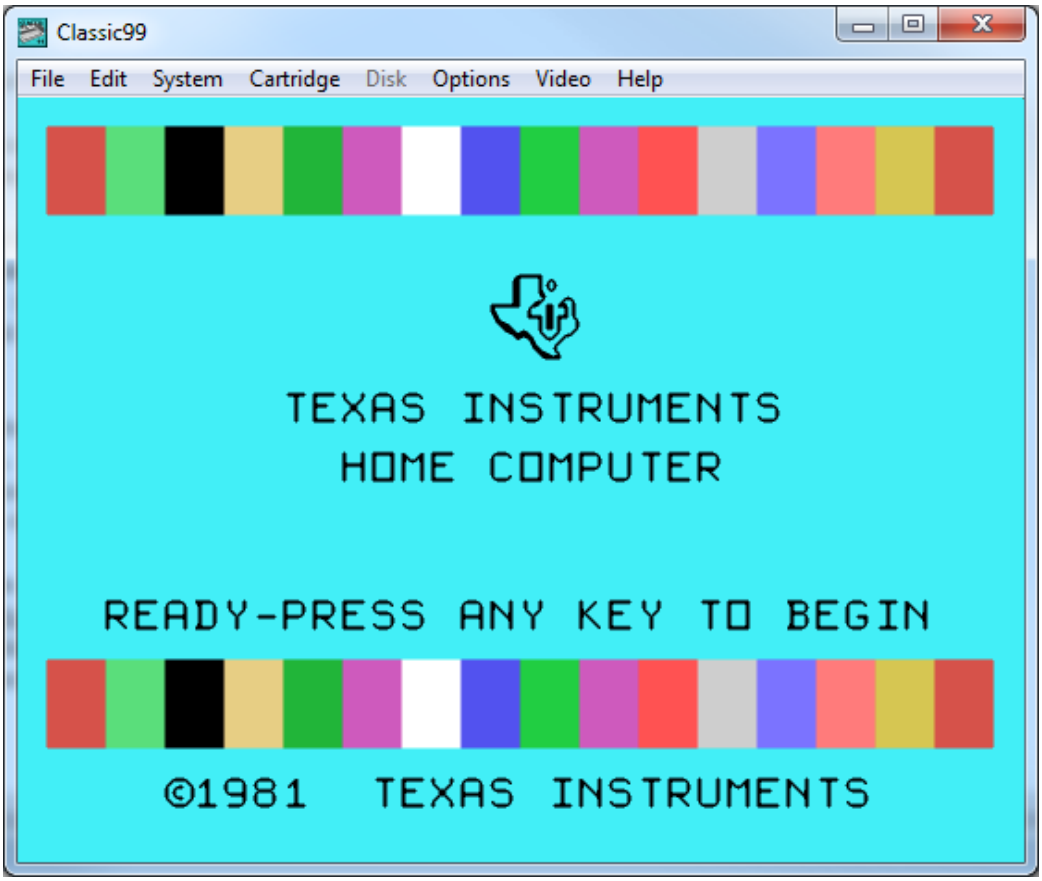
Classic99



A TI-99/4A and TI-99/4 Home Computer Emulator for Windows

by M.Brent/Tursi
http://www.harmlesslion.com/software/classic99

last update, emulator version 346, 16 February 2011

Table of Contents

1. Copyrights....3
2. Background....4
3. Support....6
4. Setup....7
5. Using the TI....8
   5.1 Keyboard....8
   5.2 Joystick....9
   5.3 Disk....9
6. Using the Emulator....12
   6.1 File....12
   6.2 Edit....12
   6.3 System....13
   6.4 Cartridge....13
   6.5 Disk....13
   6.6 Options....13
   6.7 Video....15
   6.8 Help....16
7. Debugger....17
   7.1 Information Pane....18
   7.2 Register Dump....19
   7.3 Breakpoints....20
   7.4 Uninitialized Memory Detection....22
   7.5 Emulation Control....23
   7.6 Radio Buttons....24
   7.7 Changing Memory....26
   7.8 Keyboard Shortcuts....27
8. Loading Files....28
   8.1 TI Console ROMs....28
   8.2 Cartridge ROMs....28
   8.3 Disk Images (*.DSK)....28
   8.4 Files on a Disk....28
   8.4 Archiver 3.03 Files....29
   8.5 TI BASIC Files....29
   8.6 Extended BASIC (XB) Files....30
   8.7 PROGRAM IMAGE files (E/A#5)....30
   8.8 Object Files (E/A#3)....30
9. Using the TI Keyboard....31
10. Classic99.ini....32
   10.1 Description of Settings....32
   10.2 Example of adding a User Cartridge....37
11. Files Included on DSK1 with Classic99....38

1. Copyrights

The following programs are embedded within Classic99, and included along with the source code, and are distributed under license from Texas Instruments (TI).

TI makes no warranty with respect to the programs and is under no obligation to provide any support or assistance with respect to the programs. TI is under no obligation to provide upgrades to the programs.

No liability is accepted on the part of Texas Instruments or the author with respect to use, copying or distribution of the programs.

System ROMs and GROMs: TI-99/4
System ROMs and GROMs: TI-99/4A
System ROMs and GROMs: TI-99/4A v2.2
P-Code ROMs and GROMs
Demonstration
Diagnostics
Editor/Assembler (With Editor and Assembler disk files)
Extended BASIC
Home Finance
Mini Memory
Terminal Emulator 2
TI Logo][
Alpiner
A-Maze-Ing
BlackJack & Poker
Car Wars
Chisholm Trail
Football
Hustle
Hunt the Wumpus
Mind Challengers
Munch Man
Parsec
TI Invaders
Tombstone City
Tunnels of Doom (with Pennies, Penny and Quest disk files)
Video Chess

The following programs are embedded within Classic99 or distributed along with it, and are owned by Mike Brent. No warranty or liability is offered with these programs.

Super Space Acer
CarDemo
EPSGMod Sample
Julius Demo
Space Fighter
Stranger
XB Demo

The following program is embedded within Classic99 or distributed along with it, and is a derived work by Mike Brent of art and music created by Capcom. No warranty or liability is offered with these programs.

MegaMan 2 Music

The following program is included with Classic99, and is owned by Barry Boone. It is believed to be freely distributable, and no warranty or liability is offered with this program.

Archiver 3.03

The following program is included with Classic99, and are owned by Asgard and/or the SW99ers group. It is believed to be freely distributable and no warranty or liability is offered with this program.

SAMS System Boot Version 2.0

The following program is included with Classic99, and is owned by Shawn Baron, B. Carmany and B. Harrison. It is believed to be freely distributable and no warranty or liability is offered with this program.

SAMS Memory Tester v4.0

The following program is included with Classic99, and is owned by Jon Dyer and Joe Delekto. It is included with permission from Joe Delekto and no warranty or liability is offered with this program.

TI-NOPOLY Version 1.0

The following program is included with Classic99, and is owned by DataBioTics Ltd. It is included with permission of Edgar Dohmann and no warranty or liability is offered with this program.

TI Workshop

The following program is included with Classic99, and is owned by Marc Hull. It is believed to be freely distributable, and no warranty or liability is offered with this program.

MICH_SID (SID test program)

The following software is included in Classic99, and is owned by Derek Liauw Kie Fa. It is believed to be freely usable so long as credit is given, which this notice is doing.

2xSAIMMX.ASM

The following software is included in Classic99, and is owned by the Snes9x team. It is believed to be freely usable so long as credit is given, which this notice is doing.

2xsaiwin.cpp

The following software is included in Classic99, and is owned by John Butler. It is believed to be freely usable so long as credit is given, which this notice is doing.

9900dism.cpp

The following software is included with Classic99, and is a derived work based on code owned by Ralph Nebet and the MESS Team. It is believed freely distributable.

SpeechDLL

The following software is included with Classic99, and is a derived work based on code owned by Dag Lem and the reSID team. It is believed freely distributable with its license.

SIDdll

The following software is included with Classic99, and is a derived work based on code owned by Maxim Stepin. It is believed freely distributable with its license.

Hq4x.dll

The following software is included with Classic99, and is a derived work based on code owned by Shay Green. It is believed freely distributable with its license.

Filter.dll

The following image is included in Classic99 with permission from Ron Reuter

TI-99/4A Keyboard Map

Thanks, too, to the following:

Texas Instruments
Roland Meier
Jeff Brown
Frank Palazolo
Ralph Nebet
Thierry Nouspikel
Joe Delekto
Shay Green
Mark Wills
Matthew Hagerty
Ben Yates
Ron Reuter

2. Background

Welcome to Classic99! This program attempts to reproduce, more or less accurately, the operation of the TI-99/4A Home Computer. I created it because the TI was my favourite machine, and I wanted to be able to develop for it on my PC.

The TI-99/4A was not a calculator, as many people think when they hear the numbers. Indeed, it was a full-fledged home computer. At the time, the term “home computer” was slightly differentiated from “personal computer”. Usually the “home” computer was a less expensive machine, with less of a business aspect. Home computers usually connected to a television set rather than a dedicated monitor, as well.

Illustration 1: Not a calculator...

TI aggressively targeted education, and in part due to this, the TI-99/4A made inroads into elementary schools across the United States, which contributed to a high estimation of 35% of the US home computer market, with roughly 2.8 million consoles sold. TI competed with Atari, Commodore, Sinclair, and many others for this market.

TI first entered the home computer market in 1979 with the TI-99/4. It was a powerhouse of a home computer for the time, with 16k of RAM, a 16-bit CPU running at 3MHz, 15-color video with 32-sprites onscreen via TI's own TMS9918 video chip, 4-channel sound, BASIC in ROM; and the ability to expand the system with plug-in cartridge-based software, as well as peripherals for the side expansion port. One of the promised (and released) peripherals was the impressive Speech Synthesis unit, which brought computer speech home at a low price. The keyboard was a somewhat less impressive collection of plastic buttons, often called a 'chicklet' keyboard (as the buttons look like the small candy-coated gum). The machine did not sell well, or few were manufactured, and it was extremely expensive, selling for over $1000. Today it is somewhat rare, and most are slightly different, demonstrating early changes in the machine's target market and manufacture.

In 1981 TI released their upgraded machine, the TI-99/4A. It was a minor upgrade in version number, but a pleasant upgrade to the machine. Most notably the build was made more consistent, the keyboard was replaced with a real 48-key full-stroke keyboard, and the video chip was upgraded to the TMS9918A. The price was reduced to just over $500 for the console. The 'A' added only a single new internal feature, a bitmapped graphics mode, with no additional RAM or other upgrades needed. This machine was better priced and more comfortable to use than the 99/4, and began to catch on.

Peripherals became more available as well. Cartridges like Extended BASIC and Editor/Assembler allowed developers to create their own software, although TI was criticized for presenting a 'closed' business model. Games like Parsec and MunchMan were popular in their own right. The Speech Synthesizer landed on nearly every desk, while the power users invested in the Peripheral Expansion Box, which provided slots for expansion cards. Prior to this device, users were expected to “chain” devices from the expansion port on the right side of the machine, a concept many quickly found unreasonable. With the expansion box most users also invested in the 32k RAM expansion, floppy disk controller, and RS232/PIO (printer) card.

TI soon found themselves in a battle with the other home computer manufacturers, notably Commodore and Atari, whose machines were much cheaper. With Commodore's release of the vastly popular C64, it became very hard for the expensive 99/4A to compete. TI offered large rebates, rewards like free speech synthesizers, and eventually lowered the console's price to $150, reportedly selling for a loss, to keep them moving off shelves.

To slow the loss, TI developed the “QI” version of the 99/4A, quoted as standing for “Quality Improved”. This was a redesigned motherboard and cheaper, beige plastic case, designed to save costs. The QI machines quickly

became notorious for another change, however, TI changed the operating system in order to prevent third party cartridges, such as those from AtariSoft, from booting. The QIs were not on the market for long, and because stock was matched into various cases, not all beige machines are QI. The best way to tell, if you can power it on, is to look for the copyright “1983” and “V2.2” on the title screen.

TI also released a 'baby brother' of sorts, the TI-99/2 was a small portable machine that ran TI BASIC.

In order to take back the market from the C64, seen as superior and vastly more popular than anything else on the market, TI engineers designed the TI-99/8. This powerhouse came with 64k RAM, upgradable to 15MB, both BASIC and Pascal languages built-in, built-in speech synthesizer, a 10MHz upgrade to the 99/4's processor, a 'hexbus' peripheral interface which was similar in concept to today's USB, and backwards compatibility with the original machine. In 1983 these specifications were almost unheard of. Unfortunately, it was not to be. In October 1983, citing losses of $330 million in a single quarter, TI management pulled the plug on the entire home computer division, shutting down all projects current and future. Approximately 150 TI-99/8 prototypes were built, and today they are valuable collectors items.

User groups continued for years after the machine was discontinued, in some cases 27 years later they are still continuing. Over the years many third party accessories have been created, including memory expansions, RAMdisks, hard drive controllers, keyboard replacements, and much more.

Classic99 focuses on the TI-99/4 machines, and is capable of emulating all three of them.

Illustration 2: TI-99/4

Illustration 3: TI-99/4A

Illustration 4: TI-99/4A 2.2

Classic99 itself was started in 1994 as Ami99, a TI emulator for the Commodore Amiga 2000. Although it functioned, it never reached a releasable stage, and some years later, was ported to the PC running under DOS. Later I ported to Windows, and later still I renamed it to Classic99, as most of the websites were still calling it an Amiga emulator.

Classic99 is written in C and C++ under Microsoft Visual C++, currently with Visual Studio 2005. It uses DirectX 8 interfaces and is capable of running on Windows 98, ME, 2000, XP, Vista and 7. There are also reports that it works under Wine and Parallels, but these are not supported configurations.

3. Support

Classic99 emulates the following:

TI-99/4 – including lack of bitmap mode and different keyboard layout. No special hardware or tricks are emulated on this machine mode.

 BUG: The 99/4 does not always start correctly. If it hangs, close Classic99 and restart it, and it should boot.

TI-99/4A and TI-99/4A V 2.2 – as the hardware for these machines is functionally identical, Classic99 treats them the same for hardware. The ROMs differ and both ROM sets are available.

All console hardware, including keyboard, 9900 CPU with timing, 9901 interface, 9918A video chip (and known tricks/undocumented features), 9919 sound chip, and system 16 bit/8 bit bus multiplexer.

32k memory expansion is emulated, as well as the 8k 'supercart'.

AMS up to 1MB – this is a third party memory expansion card.

Disk Controller – through a custom disk DSR, the file system interacts with the file system on your PC, rather than with floppy disk images.

Cartridges – both ROM and GROM cartridges are supported. In addition, 'Super Space' bank switching cartridges up to 64k, and '379/Jon' bank switching cartridges up to 128k are supported.

P-Code card – a UCSD Pascal operating system.

 BUG: The P-Code card does not even start at the moment;

Speech Synthesizer – implemented with the 5220 core. The speech is usually recognizable.

 BUG: the Speech Synthesizer core should be a 5200, and the speech timing does not work correctly in Classic99.

PS/2 keyboard interface – provides a natural keyboard layout to the 99/4A machines. This is based on my own (Mike Brent)'s PS/2 adapter code. It is not available for the 99/4 due to differences in the keyboard layout.

SID Blaster card – expansion card developed in 2010 by Marc Hull that puts a C64 SID audio chip in the console. This is disabled by default but may be turned on in the options menu.

 BUG: Enabling the SID Blaster may cause significant slowdown on some systems.

There are also several emulator-only features in Classic99:

Paste Text – by selecting “Edit->Paste”, Classic99 will “type” any text data that you have copied to the Windows clipboard. During paste the emulation automatically goes into *CPU Overdrive* mode.

High Quality Display – Classic99 includes numerous filters to improve the quality of the image on modern high resolution displays.

'TV' mode – Classic99 also includes a filter that simulates the artifacts and blurring of an original television.

Debugger – Classic99 has an integrated debugger, as well as integrated support for “Bug99”, an external debugger by Thierry Nouspikel.

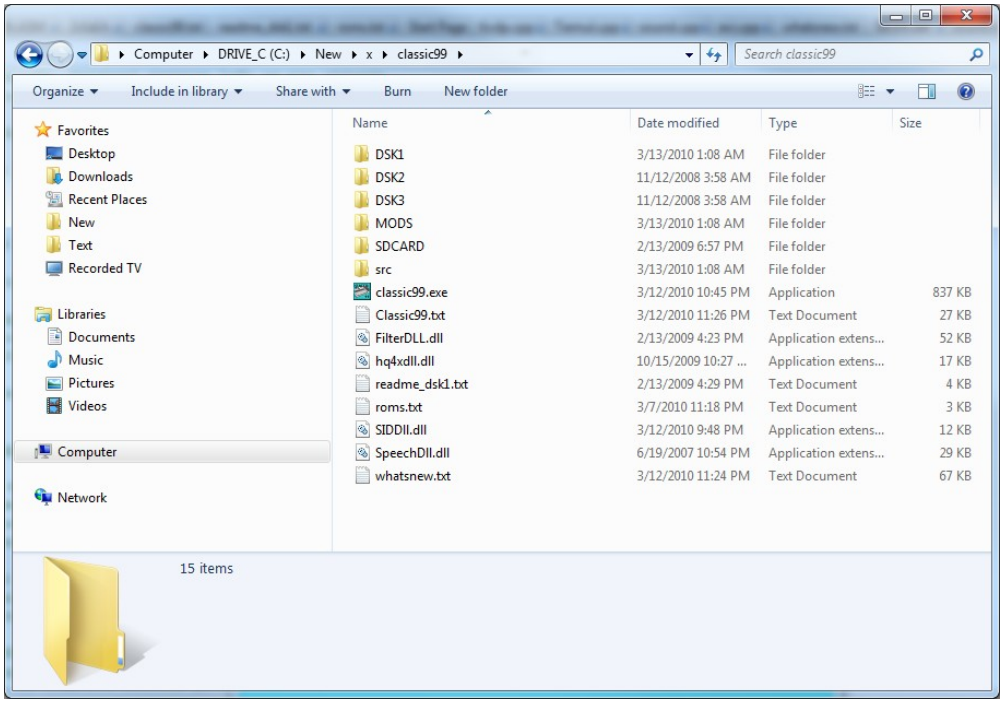
Direct Text File Access – Although TI files normally are stored in a native format, Classic99 can read and write Windows text files as though they were native files, allowing easier interface to the host system.

Clipboard file access – Classic99 can also use the Clipboard as a text-only file device, both reading and writing data.

Mouse Menu select – double click the display to “type” the character under the mouse for menus.

4. Setup

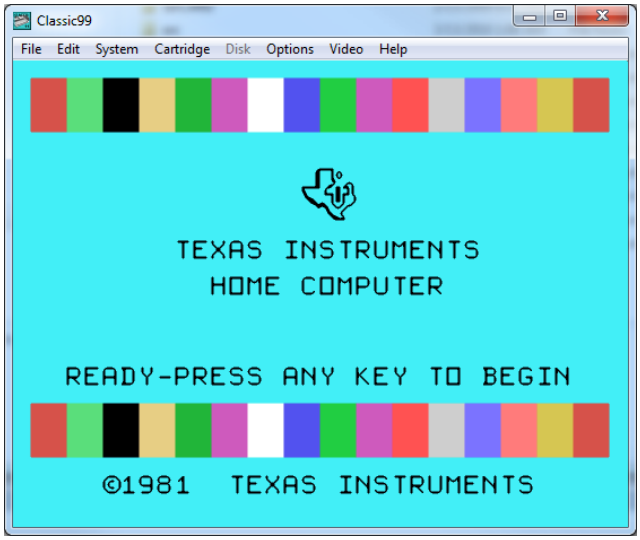
Unzip the archive with folders - in theory, you've already done that. The folder layout should look something like this:



To start the emulator, double-click the Classic99.exe.

No installer is available in the current version of the program. You may place the program anywhere that you have access to read and write folder data. By default a new installation will be configured with default settings, and have access to the three disk folders shown.

On startup, you will be presented with the TI master title page:



5. Using the TI

It is not the intent of this section to teach you how to use a TI Home Computer, but rather to summarize what you have available to you in Classic99. If you want help loading software, refer to the 'Loading Files' or 'Files included on DSK1' sections of this document.

By default, the machine starts up as an NTSC model TI-99/4A running at 3MHz, using the 2xSAI filter to double the screen resolution. Three disk devices are available; DSK1, DSK2, and DSK3. These correspond to the like-named folders from the ZIP archive. No cartridge is selected, speech is enabled, SID Blaster is disabled, AMS is enabled at 1MB, and video and audio are set to defaults. Finally, the keyboard slow-down hack is enabled, PS/2 keyboard mode is enabled, joystick 1 is set to the keyboard, and joystick 2 is set to the PC's joystick 1, if it has one.

5.1 Keyboard

By default, the keyboard runs in PS/2 mode (except for the 99/4), which is a straight port of the physical PS/2 adapter created by Mike Brent for the machine. There are a few special tweaks for Classic99. Most keys should work pretty much like you would expect, including Num Lock and the numeric keypad.

Caps Lock works like you would expect, but it is *deliberately* reversed. This is so you don't have to switch back and forth between PC and TI. Most TI applications expect uppercase typing, while the PC normally defaults to lowercase.

Scroll Lock is special. When off (normal), all keys map to the TI. The function (F) keys map to FCTN+number.

When scroll lock is on, the function keys will control debug functions of the emulator (F1-F12). All F keys stop being passed to the emulator. Also, Home is used to launch the debugger window. See the Debugger section for more information.

The arrow keys normally return FCTN+E/S/D/X, except if joystick emulation is set to keyboard, and in use. In that case, the emulator multiplexes their functionality. See the joystick notes below.

Should the PS2 emulation cause any problems, exit Classic99, and edit Classic99.ini. There is a setting PS2Keyboard - set it to equal 0 and the original TI layout will be used. See the Classic99.INI section for details.

Many of the control keys map to FCTN keys as well, reflecting the key uses from the Editor/Assembler template.

PC Key	**TI Keys**
Up Arrow	FCTN-E
Down Arrow	FCTN-X
Left Arrow	FCTN-S
Right Arrow	FCTN-D
Tab	FCTN-7
F1	FCTN-1
F2	FCTN-2
F3	FCTN-3
F4	FCTN-4
F5	FCTN-5
F6	FCTN-6
F7	FCTN-7
F8	FCTN-8
F9	FCTN-9
F10	FCTN-0
Insert	FCTN-2
Delete	FCTN-1
Page Down	FCTN-4
Page Up	FCTN-6
ESC	FCTN-9

5.2 Joystick

The TI supports two joysticks, or “wired remote controllers”. In Classic99, they may be configured as any combination of keyboard, or the first and second PC joysticks (as per the Gaming Controller Control Panel).

If using real joysticks, Classic99 will respond to the first four buttons – any will respond as the TI's sole button. The analog X and Y axes are automatically converted to digital responses for the console.

In keyboard mode, the joystick uses the arrow keys for movement, and the Tab key for fire. These keys are normally mapped to keyboard functions, however, if Classic99 detects that the program is scanning the joystick, and the joystick is configured to the keyboard, then these keys (and Tab) respond to the joystick request only. After about 3 seconds if the joystick is not scanned again, they return to being keyboard keys.

5.3 Disk

Classic99 supports up to 10 disk directories. These are configured in the Classic99.ini file. By default as it ships, DSK1, DSK2 and DSK3 are configured as like-named folders in the Classic99 folder. Supported disk indexes are from 0-9. See the section on Classic99.ini for details on extending this.

Classic99 supports both FIAD (Files In A Directory) and DSK images (DOAD or Sector dump).

5.3.1 FIAD

Using the FIAD method of file storage, each TI file is a separate file on your PC harddrive (or any other file device). Classic99 can be configured to access any file that your system normally can access. By default it can read both TIFILES and V9T9 files, as well as Windows text files if they have the appropriate extension (as DF80 or DV80). It can also import non-TI files without adding a header, automatically using DF128 format, which is the same format that would have been used if you had downloaded the file to the TI via Xmodem. Configuration options let you disable any of these options if they are problematic.

FIAD-TIFILES
TIFILES format is also known as XMODEM format. This is the native format for a file transferred from a TI to another machine via XMODEM, and includes a 128 byte header which details what kind of file it was, its size, and so forth. Some extensions include the filename, but Classic99 deliberately does not support this, in order to avoid dependencies between the internal filename and the filename on the host disk. This is the native format of Classic99 to maximize interoperability with a real TI-99/4A. Files may have any legal Windows filename, but note that many applications limit you to 10 characters. All legal file types are supported and all standard file access modes are supported.

FIAD-V9T9
V9T9 format was created for the V9T9 emulator by Ed Swartz, and is similar to TIFILES. It includes the original filename, to try and support all legal TI files under the old DOS file system. Under modern Windows, most (but not all) of the original limitations no longer apply, however, some applications still support this format. Because this format includes the original filename in the header, it must match the file on the disk. The original V9T9 format used extended ASCII characters to match against characters not legal in DOS, but Classic99 does not recognize these. Thus, filenames with non-alphanumeric characters may have trouble loading. To support these files with a minimum of issue, Classic99 recognizes the tilde ('~') as a replacement character for the high ASCII characters of the original format. If you have trouble loading such a file, try renaming it as such. Characters that should be replaced are '?', '/', '>', '<', ':', '"', '*' or '|'. Classic99 will automatically handle the replacement if it creates the file. Note that the backslash (\) is legal in V9T9 files but Classic99 will not replace it. This is because Classic99 uses the backslash to support subdirectories. Programs that require the backslash in filenames may need to be modified to work, but these have shown to be very rare. Classic99 considers the V9T9 file format to be deprecated.

FIAD-Windows Text
Windows text files may be accessed as well, so long as the open mode is Display, and the record length is 80 characters. (Additional options allow flexibility on the record length, as well as whether this option is enabled at

all – see the Classic99.ini section for details.) Windows text files are recognized by extension, in order to work the extension must be one of: .TXT, .OBJ, or .COB. By default, Classic99 will only **read** Windows text files, but it may also be configured to **write** them.

FIAD-Files Without Headers
By convention, files transferred to the TI without a TIFILES header were stored as DF128. Classic99 will automatically read such files as DF128 if they have no extension. Note that if text files are configured to be read without extensions (this was a default in old versions), this option won't work. You can correct this in Classic99.ini (see the section on Classic99.ini).

FIAD-Subdirectories
In all FIAD modes, Classic99 supports accessing subdirectories inside the DSK directories by using a standard Windows backslash ('\'). Note that some TI applications may limit the length of a path to as little as 10 characters, but this is still useful in many cases. Subdirectories are not included when listing a disk from the TI.

FIAD-TI Artist+ Files
Classic99 also supports a naming extension for TI Artist+ files. There is a plugin for Photoshop that recognizes TI Artist+ pictures so long as they have the extensions .TIAP and .TIAC. There is also an image converter that uses these extensions. However, TI Artist+ on the real machine uses _P and _C respectively. Therefore, Classic99 can automatically map these – if you enter _C or _P, it will automatically try .TIAC or .TIAP respectively.

FIAD-Disk Sector Access
FIAD does not support true disk access, however, it does emulate some sector level services. The first 128 sectors on the disk are emulated for read only:

0 - This contains the disk header. This will contain the first 10 characters of the path, and show the allocation bitmap as full. The rest of the data will reflect an unprotected DSSD disk, regardless of the data in the folder.

1 – The File Descriptor Records index will have one entry for each file in the folder, up to a maximum of 127 files, and will sequentially reference incrementing sectors for each. If there are more than 127 files in the folder, the additional files are not indexed (but may still be accessed by software).

2-128 – Each sector contains an FDR for one of the files in the folder. This will reflect the file type information, but the cluster map is not filled in. If there are fewer than 127 files, then not all sectors are available. If there are more, the additional files are not listed here (but may still be accessed by software).

FIAD-File Sector Access
Classic99 fully supports reading and writing files via sector access, using the SBRLNK functions.

FIAD-Other SBRLNK Low Level Access
Classic99 does not support Write Sector, Format Disk, Protect File, Unprotect File, or Rename File on files in the file folders.

5.3.2 Disk Images

Classic99 has experimental support for DSK images. Only sector dump images (V9T9/MESS style) are supported, including support for the reverse sector numbering on side two of a DSSD diskette (as V9T9 sometimes wrote). At the moment, this feature is not configurable – if you would like to test it, see the *Classic99.ini* section for details on activating it.

This feature is currently experimental and not intended for general use. This section will be expanded once it is properly implemented.


BUG: Image access is currently read only. In addition, the SBRLNK command to read files via sector access is not implemented. It is *not possible* to change the disk image without quitting and re-starting Classic99.

5.3.2 CLIP Device

In addition to the DSK devices, Classic99 also supports DISPLAY mode access to the Windows clipboard. The devicename is "CLIP". Full access is available, including read, write, update and append. Note, however, for update and append, the clipboard is read and cached on the open call, then replaced when the file is flushed or closed. Intermediate external changes won't be reflected.

The CLIP device does not support non-text mode access. Also, the opcodes LOAD, SAVE, DELETE, SCRATCH, and STATUS are not supported. No SBRLNK opcodes are supported to the CLIP device.

Note, the CLIP device does not support filenames, so unlike the disk system, there is no need for there to be a period in the device name. Some programs, like Editor/Assembler, will report error 0 if you attempt to load from any device without the period. Appending a period is enough to work around this error, so E/A can load from or print to "CLIP." If a filename is specified, the CLIP device will ignore it.

The CLIP device supports fixed and variable length records up to 254 bytes.

6. Using the Emulator

The Classic99 emulator usually keeps itself in the background, allowing you to focus on the experience of using the machine it emulates instead. Most functionality can be accessed via the standard Windows user experience. For instance, the Window may be re-sized by dragging the borders, or moved by dragging the title bar. The application can be exited by clicking the close button. Most other functions are available through the menu bar.

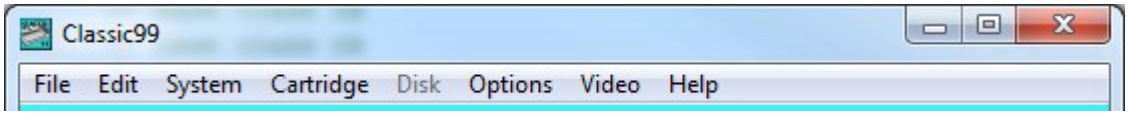


Illustration 5: Classic99 Menu Bar

6.1 File

File → Reset

Resets the emulator as if it had been powered off, and then on again. Brings all memory and systems to a clean state and restarts. Does not change configuration state or loaded cartridges.

File → Reset – Scramble RAM

Resets the emulator but fills all RAM with random values. This can be used by programmers to try and find bugs with uninitialized memory. Note that due to bugs in the TI's own ROMs that sometimes the system can not start with this option, and you will have to select it again.

File → Quit

Exits the application. Classic99 will ask for confirmation before exitting.

6.2 Edit

Edit → Paste

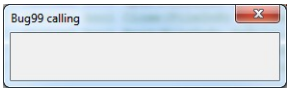
If there is text on the Windows clipboard, Classic99 will enter the text as if a user was typing it. Classic99 will do this by intercepting the calls to KSCAN and inserting the appropriate return values, so will work in any application that uses KSCAN.

Edit → Debugger

Launches the Classic99 Debugger window. See the Debugger section for more details.

Edit → Bug99 Window

Launches the Bug99 Calling Window for Bug99 to communicate with. This window is not user interactive but must be present for the Bug99 application to communicate with Classic99. Note that you must also load the Bug99 software inside Classic99 to use this tool. See http://www.nouspikel.com/ti99/bug99.htm



Edit → Heatmap

Launches the heatmap window. Part debug tool, part visualization, the heatmap shows you memory accesses inside Classic99. Red shows CPU (RAM and ROM) access, Green shows GROM access, and Blue shows VDP access. Address >0000 is at the top left, and address >FFFF is at the bottom right. Each pixel lights up when the address is hit, and fades out slowly. The update of the display may impact performance on some PCs. Click the Close button to exit it.

6.3 System

System → TI-99/4

Selects the TI-99/4 emulation system. This will reset the emulator into the 1979 model of the TI-99/4. Note that in the current build, this is not working reliably. You will be warned that you may need to exit and restart Classic99 if it does not start correctly. If you do not see the title page after dismissing the warning, close Classic99 and launch it again, and this system should come up.

System → TI-99/4A

Selects the TI-99/4A emulation system. This will reset the emulator into the 1981 model of the TI-99/4A.

System → TI-99/4A V2.2

Selects the V2.2 QI TI-99/4A emulation system. This will reset the emulator into the 1983 model of the TI-99/4A.

6.4 Cartridge

Cartridge → Apps → ????

Allows you to select one of many application cartridges included with Classic99. This will cause the emulator to reset and the cartridge to appear on the master title page. Note that TI Workshop (379) will not show up on the V2.2 TI-99/4A due to the Third Party ROM lockout.

Cartridge → Games → ????

Allows you to select one of many game cartridges included with Classic99. This will cause the emulator to reset and the cartridge to appear on the master title page. Note that Super Space Acer will not show up on the V2.2 TI-99/4A due to the Third Party ROM lockout.

Cartridge → User → ????

Allows you to select a cartridge that you have added to the Classic99.ini file. If you have not added any files this list will be empty.

Cartridge → User → Open...

Brings up a file dialog that allows you to select a cartridge image to load. Currently only the V9T9 .BIN format is supported – these cartridges are usually found as multiple files with the same filename, except for the last letter, which varies as C, D, or G, and a .BIN extension. Simply selecting one file in the group is enough, Classic99 will automatically check for the other files in the group. Note that 379/Jon format, and SuperSpace format bank switched cartridges can not be loaded with this shortcut.

6.5 Disk

This menu item is not implemented and is grayed out on all installations.

6.6 Options

Options → Pause When Window Inactive

When this item is checked, selecting a different PC window for focus will cause Classic99 to automatically pause. That is, it will run only when it has focus.

Options → CPU Throttling → Normal

Makes the emulator run at the 'normal' speed, which is approximately the speed a real TI would run.

Options → CPU Throttling → CPU Overdrive

Runs the CPU at a much faster rate – as fast as a simulated 150MHz, but does not speed up the rest of the system. This can overcome issues such as slow performance in a program without interfering with video updates or sprite movement.

Options → CPU Throttling → System Maximum

This speeds up the entire emulator, including video and sound updates. However, it currently causes performance problems and on some systems may actually run slower than normal.

Options → CPU Throttling → CPU Slow

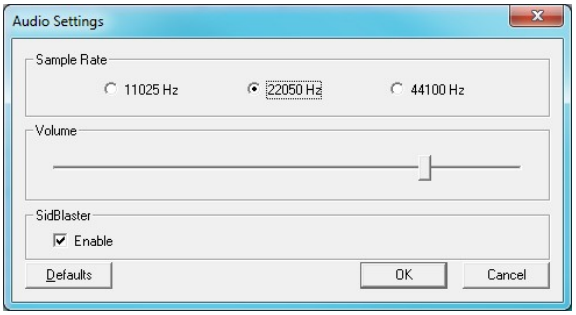
This slows down the CPU to a single clock per frame, meaning it runs at 60Hz when in NTSC mode. All other systems are affected. This can be useful when debugging but is generally too slow to watch updates on the screen.

Options → Speech Enabled

When checked, the Speech Synthesizer is emulated. When cleared, the Speech Synthesizer will not be available to the emulator. Speech is provided by an external DLL (speechdll.dll) that may not be available.

Options → Audio...

Loads the Audio configuration dialog, which has the following options:



Sample Rate: Select 11025Hz, 22050Hz, or 44100Hz. Higher rates sound better but use more CPU time.

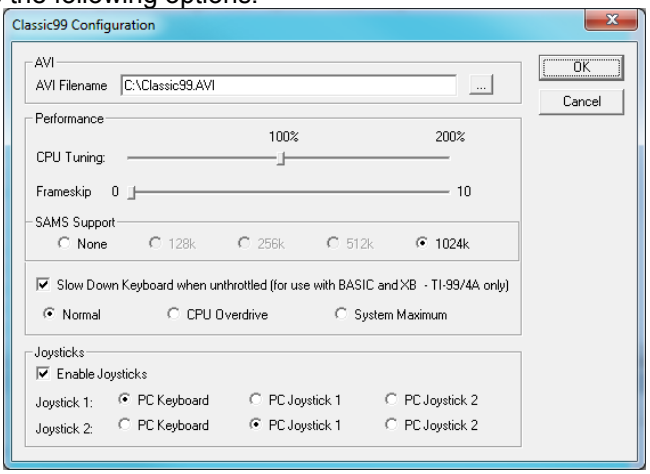
Volume: Push the slider right for louder sounds, or left for softer sounds.

SidBlaster: Check this to enable the SID Blaster card emulation. Clear it to disable (which may improve performance.) SID is provided by an external DLL (siddll.dll) which may not be available.

Defaults: Click this to reset settings to the default.

Options → Options...

Loads the General configuration dialog, which has the following options:



AVI Filename: Enter a path and filename to save a recorded AVI at, or click the '…' button to browse for a path. The AVI will not be recorded until you select Video->Start Recording Video.

CPU Tuning: Drag the slider left for a slower 'normal' CPU, or right for faster. 100% represents a standard 3MHz CPU.

Frameskip: Set the slider right to skip drawing video frames for performance. This option may have little effect in the current build and may cause audio issues.

SAMS Support: Set the size of the emulated SAMS card. The only valid options today are None (no SAMS card installed), or 1024 (1MB).

Slow Down Keyboard when unthrottled: This option manipulates the console operating system to slow down key repeat when the system is configured for CPU Overdrive or System Maximum. Select *Cpu Overdrive* to take effect only in overdrive, or *System Maximum* to take effect only in System Maximum.

Enable Joysticks: When checked, joysticks will be emulated using the settings below. *PC Keyboard* emulates the joysticks using the arrow keys and the Tab key. *PC Joystick 1* and *2* are numbered as per the Windows Gaming Devices Control Panel applet.

6.7 Video

Video → Maintain Aspect Ratio on Window
When checked, this option forces the window aspect to remain constant as the window is sized. It does not affect full screen mode, or if the window is maximized. If unchecked, the window may be distorted into incorrect aspects.

Video → Flicker
When checked, Classic99 will emulate the sprite limitation of 4 sprites displayed per scanline. Additional sprites are not drawn. If cleared, Classic99 ignores this limitation and draws all sprites regardless of position.

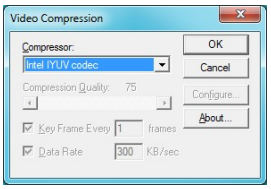
Video → Layers → Disable Blanking
When checked, Classic99 will ignore the blanking bit in the VDP, and always draw the display.

Video → Layers → Disable Sprites
When checked, Classic99 will not draw sprites.

Video → Layers → Disable Background
When checked, Classic99 will not draw the background graphics.

Video → 50Hz (60Hz if off)
When checked, Classic99 will run with a PAL refresh rate of 50Hz, instead of the default NTSC 60Hz. This may make software appear to run more slowly.

Video → Start Recording Video
When this option is selected, a compression selection dialog is displayed. Select a video compressor to start recording video from Classic99 to the file configured in the Options dialog. No audio is recorded with this option, and the video uses the original unfiltered image.



Video → Start Recording Video+Audio
This option also starts recording video, but includes audio at the current sampling rate. The audio may not be compressed. This option is currently not working correctly.

Video → Stop Recording Video
If video is being recorded to an AVI file, this option will stop it.

Video → Screenshot (Basic)
Allows you to save a screenshot using the raw, unfiltered image.

Video → Screenshot (Filtered)
Allows you to save a screenshot using the currently displayed image.

Video → Change Size → 1x / 2x / 3x/ 4x
Changes the size of the window to be exactly 1 times, 2 times, 3 times, or 4 times the actual resolution of the current filtered mode image. 1x will produce 1:1 pixels, 2x will produce 4:1 pixels, etc.

Video → Stretch Mode → None
Sets the video stretch mode to none. Classic99 will use GDI to draw the image on the screen, and will not scale it to fit the window. This is the simplest mode if problems are occurring.

Video → Stretch Mode → DIB
Sets the video stretch mode to DIB. Classic99 will use GDI Device Independent Bitmaps to stretch the image to fit the Window. This is usually slower but on some older cards it can be faster or more stable than DirectX.

Video → Stretch Mode → DX

Sets the video stretch mode to DirectX. Classic99 will use the card hardware to stretch the image to fit the window, if available. This is usually the fastest windowed mode.

Video → Stretch Mode → DX Full → ????

Switches the display to full screen mode using DirectX at the resolution specified. Not all resolutions are available on all cards. Note that bit-depths less than 16 may cause color shift. To exit full screen mode, press Alt+Enter, or right-click the mouse.

Video → Filter Mode → None

Displays the image without filtering. It is sized to match a true TI display, then stretched to fill the window (if the stretch mode is set). The effective resolution without borders is 256x192.

Video → Filter Mode → hq4x

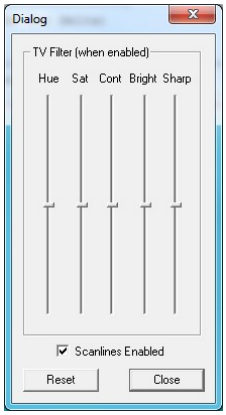
This implements the hq4x filter by Maxim Stepin to scale the image up four times each way and smooth pixels and lines. The effective resolution without borders is 1024x768. This filter is provided by a DLL and may not be available.

Video → Filter Mode → 2xSAI / Super2xSAI / SuperEagle

These options implement the 2xSAI filters by Derek Liauw Kie Fa. Each uses a slightly different technique to scale the image up twice each way, and smooth pixels and lines. The effective resolution without borders is 512x384.

Video → Filter Mode → TV Mode

This option implements the NTSC TV Mode filter by Shay Green. It uses several techniques to reproduce artifacts, ghosting, blur, and other effects of NTSC and RF modulation. It also responds to the TV Controls configuration dialog.

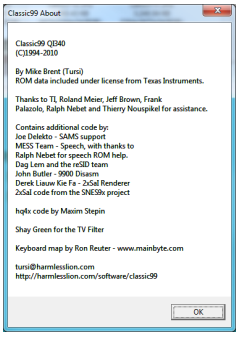


Video → TV Filter Controls

These provide controls analogous to what was available on older NTSC television sets. *Hue* changes the colors displayed. *Sat* adjusts the color saturation. *Cont* adjusts the contrast of the image. *Bright* adjusts the brightness. *Sharp* adjusts the sharpness.

Scanlines Enabled turns the horizontal scanline effect on and off.

Reset restores all the sliders to their default, centered positions.



6.8 Help

Help → About

Reports the version of the emulator, and all the people to thank!

Help → Keyboard Map

Displays an image of the TI-99/4A keyboard map, showing the FCTN keys.

Help → Known issues with current cartridges

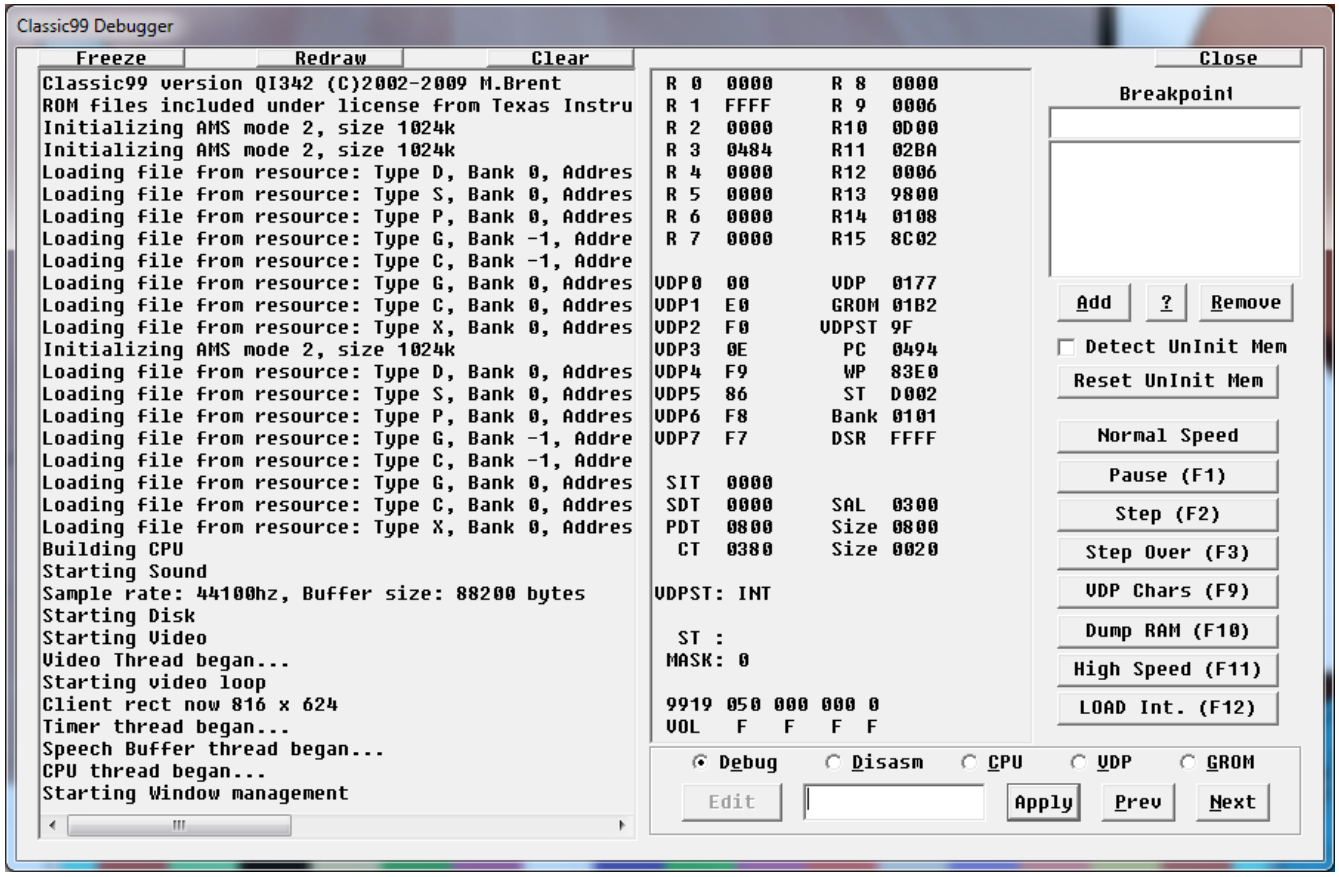
If the currently selected cartridge has any known issues, this will display them.

7. Debugger

Classic99 includes an integrated visual debugger to aid in troubleshooting and debugging programs.

The most important note to the debugger is that keyboard shortcuts, and all breakpoints (including 'step over' breakpoints) require that *Scroll Lock* be **ON** on your keyboard. When Scroll Lock is on, the 'F' keys (*F1-F12*) stop working like TI FCTN+<number> keys and instead map to the debug console.

To display the debugger Window, make sure Scroll Lock is enabled, and press *Home*. Alternately, you may select Edit->Debugger from the window menu.



The initial debugger window looks as above.

7.1 Information Pane

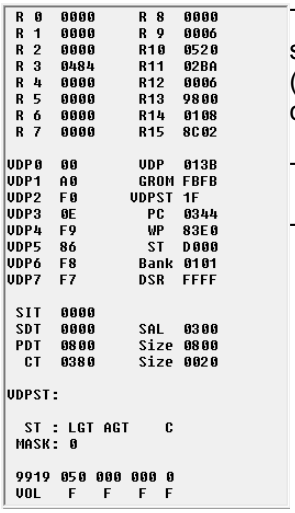
The *Leftmost* Information pane contains the current text display, as selected by the radio buttons in the bottom right. (Read on for details on the content). Above the display are three buttons:

> *Freeze:* The display tends to be continuously updated when the emulator is running. You can press this button to stop all updates, to enable copy and paste from the display. The button text will change to *Unfreeze*, which is used to resume normal updates.
>
> *Redraw:* If for any reason parts of the display does not seem to be properly drawing everything, press Redraw to force a full update.
>
> *Clear:* This button is only meaningful on the Debug view, and clears all old messages from the log.

The main user interface to the information dialog is located at the bottom right of the dialog, and contains a five-selection set of Radio Buttons, which control the information displayed in the leftmost information pane, as well as a text entry box and several buttons.

 BUG*: the 'Edit' button is permanently disabled as that functionality has not yet been implemented.*

7.2 Register Dump



The Register Dump information Pane contains information about the current state of the system registers. The first section summarizes the 16 CPU Registers from R0-R15. (While you could obtain this information from the CPU memory dump, this is a convenience).

The 9918A VDP's Write-only registers are displayed as VDP0 through VDP7.

The next column contains some generic information:

VDP has the current value of the 9918A VDP's address counter
GROM has the current value of the GROM address counter
VDPST is the current value of the 9918A VDP's status register
PC is the 9900 CPU's Program Counter
WP is the 9900 CPU's Workspace Pointer
ST is the 9900 CPU's Status register
Bank is the current cartridge bank-switch value. The first byte indicates the banking mask, which is determined by the size of the ROM, and the second is the current bank.
DSR indicates the currently paged DSR. FFFF means no DSR.

The next group calculates the addresses of the VDP tables as a shortcut.

SIT gives the address of the Screen Image Table, calculated from VDP Register 2.
SDT gives the address of the Sprite Descriptor Table, calculated from VDP Register 6.
SAL gives the address of the Sprite Attribute List, calculated from VDP Register 5.
PDT gives the address of the Pattern Descriptor Table, calculated from VDP Register 4. Its size is also shown to assist with masking in Bitmap mode.
CT gives the address of the Color Table, calculated from VDP Register 3. Its size is also shown to assist with masking in Bitmap mode.

VDPST gives a visual representation of which VDP Status bits are currently set. They are:

INT	VDP Interrupt Request is set
5SP	5 sprites on a line bit is set (the 5th Sprite number is shown at the end of the line)
COL	Sprite collision bit is set

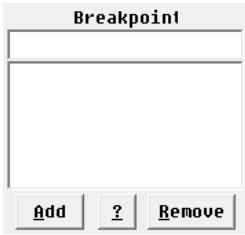
ST then breaks down the CPU register. The following bits are shown if they are set:

LGT	Logical Greater Than
AGT	Arithmetic Greater Than
EQ	Equal
C	Carry
OV	Overflow
OP	Odd Parity
X	Extended Operation

MASK shows the current value of the CPU interrupt mask (which is part of the Status Register).

Finally, there is a breakdown of the sound chip, labelled as *9919* and *Vol*. The first row is the current value of the countdown reset register for each channel (ie: for the tone channels, the frequency count, and for the noise channel, the noise type). The second row shows the current attenuation value for each channel.

7.3 Breakpoints



The Breakpoints area allows you to control automatic breakpoint functionality in the emulator. You type your Breakpoint description in the top line, and then click *Add* to add it to the list below. You can also select an existing Breakpoint and click *Remove* to remove it from the list. The *?* button can be used to get a popup summary of the various breakpoint syntaxes.

When Classic99 detects that a Breakpoint condition is met, the emulation is stopped, and the title bar of the emulator will reflect that a Breakpoint has occurred.

Important Note: Breakpoints will only halt the emulation (or execute timing) when Scroll Lock is on!

A Breakpoint Description consists of a single character to indicate the type of the test, an address or address range, and an optional match value.

7.3.1 Match Values

A match value is specified after an address or address range by adding an equals sign (=) and either 2 hex characters (for a byte) or 4 hex characters (for a word). The type of match value required is dependent upon the type of the breakpoint. As a special note, it is the value that is written, not what ends up in memory, that is compared. Therefore you can use match values to detect specific writes to hardware devices (such as cartridge bank switching), even if the value is never stored.

7.3.3 Addresses and Address Ranges

An address is simply specified as 4 hexadecimal digits, and represented an exact address to be matched. Addresses may also be substituted by an address range, which is specified as an open parenthesis '(', the first address in the range, a dash '-', the last address in the range, and a close parenthesis ')'. For example: (8300-83FF) is an address range that specifies all of scratchpad memory. Addresses, whether a single value or a range, may specify a specific cartridge ROM bank by appending a colon ':' and the bank number. The first bank is 0, and the highest possible bank in Classic99 is 'F', for a 128k cartridge. When not specified, all banks are tested.

7.3.3 Types of Breakpoints

There are 9 types of Breakpoint, plus one non-breaking Watch action.

(none) Break when the CPU PC equals the address specified
Classic99 will break only when the address or address range specified is matched by the CPU program counter.

Example1: A000	*- break when PC is equal to A000*
Example2: (7000-7100)	*- break when PC is any value from 7000 to 7100 inclusive*
Example3: 7000:1	*- break when PC is equal to 7000 in ROM bank 1 only*

***** Break when the address specified is accessed (by any read or write)
Classic99 will break when the CPU address or address range is accessed, whether read or write, whether program or data access.

*Example1: *2000*	*- break when address 2000 is accessed*
*Example2: *(8300-83FF)*	*- break when any address from 8300 to 83FF is accessed*
*Example3: *(6300-63FF):0*	*- break when an address from 6300 to 63FF is accessed in ROM bank 0*

> Break when the address specified is written to
Classic99 will break when the CPU address or address range is written to only.

Example1: >2000	*- break when address 2000 is written to*
Example2: >(8300-83FF)	*- break when any address from 8300 to 83FF is written to*

< Break when the address specified is read from
Classic99 will break when the CPU address or address range is read from only.

Example1: <2000 *- break when address 2000 is read from*
Example2: <(8300-83FF) *- break when any address from 8300 to 83FF is read from*
Example3: <(6300-63FF):2 *- break when an address from 6300 to 63FF is read from ROM bank 2*

M Break when the BYTE address specified is written a matching value
Classic99 will break when the CPU address or address range is written a byte matching the one specified. A byte match argument must be included.

Example1: M2005=FF *- break when address 2005 is set to FF*
Example2: M(8300-83FF)=55 *- break when any address from 8300 to 83FF is set to 55*

W Break when the WORD address specified is written a matching value
Classic99 will break when the CPU address or address range is written a word matching the one specified. A word match argument must be included. Note that the address specified must be even.

Example1: W2004=FFFF *- break when address 2004 is set to FFFF*
Example2: W(8300-83FE)=55AA- break when any address from 8300 to 83FE is set to 55AA

V Break when the VDP memory byte specified is written a matching value
Classic99 will break when the VDP is written a byte to the VDP address or address range specified matching the byte match specified. A byte match argument must be included.

Example1: V2005=FF *- break when VDP address 2005 is set to FF*
Example2: V(0000-03FF)=40 *- break when a VDP address from 0000 to 03FF is set to 40*

U Break when the VDP Register specified (0-7) is written a matching value
Classic99 will break when the specified VDP register is written the matching byte value. The VDP register must be from 0-7 (ranges are not permitted), and a byte match argument must be included.

Example1: U7=02 *- break when VDP register 7 is set to 02*

R Break when the CPU Register specified (0-15) is written a matching word value
Classic99 will break when the specified CPU Register is written the matching byte value. The CPU register must be from 0-15 (ranges are not permitted), and a word match argument must be included.

Example1: R14=83E0 *- break when CPU register 14 is set to 83E0*

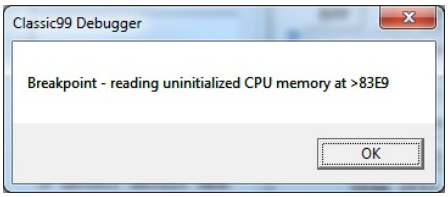
T Perform cycle counting between a range of addresses (address range required)
Classic99 will count cycles from the first address match until the second, and emit the result each time the second address is reached into the debug log. Addresses must match exactly, but need not be in any particular order. (For example, the first address does not need to be lower than the second). The debug log output will show the word 'Timer', followed by the number of cycles counted. Minimum and Maximum counts are reported, as well as the overall average. The number in parenthesis after 'Average' is the number of times the timer has been hit.

Example1: T(0060-0074) *- Count cycles between PC 0060 and 0074*
Debug log shows:
Timer: 102 CPU cycles - Min: 102 Max: 722 Average(5326): 106
Timer: 722 CPU cycles - Min: 102 Max: 722 Average(5327): 106
Timer: 102 CPU cycles - Min: 102 Max: 722 Average(5328): 106

7.4 Uninitialized Memory Detection

Classic99 has the ability to trigger a Breakpoint on read access to RAM that has never been written to. This can help to find uninitialized memory bugs.

To use this functionality, simply enable the 'Detect UnInit Mem' checkbox before launching your program. When uninitialized memory is read, a popup box will display while the emulator breakpoints in the background.



Classic99 tracks writes to memory from the moment that emulation is started. At any time you can press the *Reset UnInit Mem* button to clear this tracking. From the moment you press it, ALL RAM is considered uninitialized. If a program is running at the time, including the TI operating system, it is very likely that you will immediately breakpoint for reading uninitialized memory (when in fact, it was initialized before you pressed the button.)

The best way to use this feature is to leave it off while you load your program, and to set a breakpoint on the first instruction of your code. When your breakpoint is hit, then press the Reset button, and then continue the program.

NOTE: if you use any console ROM or GROM functions, including the console interrupt routine, those functions expect that the operating system has set up certain addresses in scratchpad. This will almost certainly cause your program to throw an uninitialized memory breakpoint when those functions are called. Thus you must use some judgement as to whether or not you should reset the memory tracking before starting your program.

7.5 Emulation Control

There are several buttons that control the emulation. These buttons have shortcut function keys that are active only when Scroll Lock is ON, but the dialog buttons will work regardless.

Normal Speed will set the emulator running normally. Use this when breakpointed and you want to resume operation.

Pause (F1) will cause an immediate breakpoint of the emulator, stopping all operation.

Step (F2) is used when the emulator is breakpointed already, and will allow the CPU to execute exactly one instruction. This may or may not allow other hardware to run depending on the exact CPU timing.

Step Over (F3) is a specialized version of Step that will skip displaying all code executed by a BL or BLWP instruction, and stop upon return to the same area of code. Note that this function can detect the return only if the code returns to the next instruction address, or 2 bytes after that (to allow for BLWP calls with DATA arguments).

BUG: *Step Over sets an internal breakpoint, and so will only stop executing when Scroll Lock is ON!*

VDP Chars (F9) disables the normal screen draw function, and instead displays the entire character set. The effect is as if the Screen Image Table were filled with the values 0-255 repeatedly.

BUG: *VDP Chars does not function in Bitmap mode.*

Dump RAM (F10) saves the contents of CPU RAM and VDP RAM to files in the current working directory. The CPU RAM is saved as the file MEMDUMP.BIN and is 64k in size, and the VDP RAM is saved as the file VDPDUMP.BIN and is 16k + 8 bytes in size. The last 8 bytes are the 8 VDP Read-Only Registers.

BUG: *The CPU memory saved is a snapshot of the currently visible 64k of CPU RAM. Paged cartridges, DSRs, and AMS memory is not taken into account.*

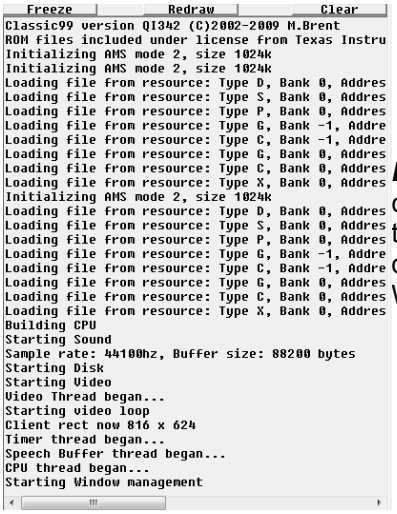
High Speed (F11) activates *System Maximum* speed. Use the *Normal Speed* button to return to normal operation.

BUG: *System Maximum may consume too many resources and actually run slower on some systems.*

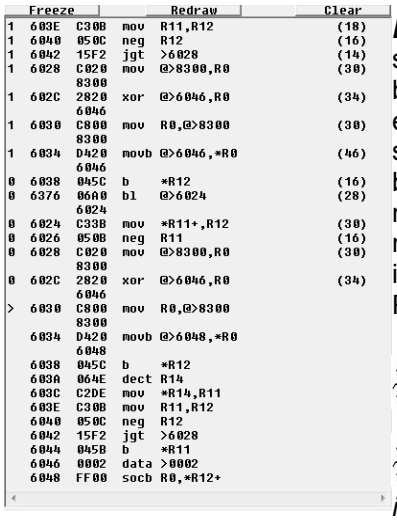
Load Int. (F12) triggers a non-maskable LOAD interrupt to the 9900 CPU. Before doing this, the emulator checks whether the LOAD vector at >FFFC has been initialized with non-zero values, and displays an error if not. NOTE: the real 9900 does not perform any such check.

7.6 Radio Buttons

The purpose of the buttons and the text-entry box is modified by the radio buttons:



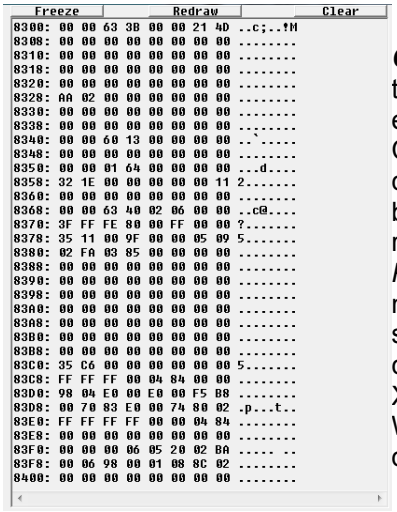
Debug contains a log of debugging information from the emulator itself. This can be helpful when something does not appear to be working correctly, or troubleshooting issues like a file that fails to load. This is just a raw dump of debug text without a specific format. In this mode you can change the Workspace Pointer (WP) or Program Counter (PC) in the entry box, see below.



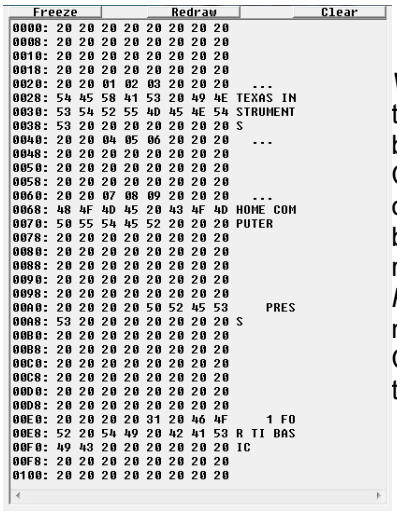
Disasm contains a running disassembly of the emulator. The disassembly shows each instruction as it is executed, with the current instruction indicated by a greater-than symbol (>). Instructions above this symbol were previously executed, and provide a short history. If these instructions were in bank-switched cartridge space, they will be prefixed by a number 0-F, indicating the bank the code came from. Executed instructions include, in parentheses, the number of CPU cycles calculated for this instruction's execution, including memory access time. Instructions below this symbol show the next instructions in memory. In this mode you can change the Workspace Pointer (WP) or Program Counter (PC) in the entry box, see below.

BUG: *the CPU cycle count is not correctly calculated for the DIV instruction, instead, a fixed average count is always returned.*

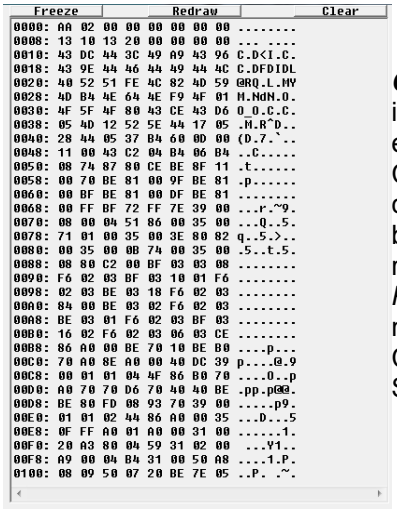
BUG: *the disassembly is taken from memory at the time that the display is drawn, therefore if AMS or DSR memory pages, the history may show incorrect instructions at addresses executed before the page changed.*



CPU shows a dump of 1024 bytes of CPU memory starting at the address in the text box. For each line, first the memory address is displayed, followed by eight bytes in hexadecimal, followed by the same eight bytes in ASCII text. Characters not valid for printing as 7-bit ASCII are represented as a period. A 4-character hex address may be entered here, or a CPU register (R0-R15) may be entered, and the memory display will track the address pointed to by that register. Press *Apply* when changing the value in the text box. In addition, the *Prev* and *Next* buttons may be used to advance 1k forwards or backwards in memory. This display normally shows the memory currently visible to the CPU, so applications with pages of memory will only show the current one. For cartridge ROM, you can specify a different bank by entering the address as XXXX:Y, where Y is the bank number. In this mode you can change the Workspace Pointer (WP) or Program Counter (PC) in the entry box, as well as directly modifying CPU memory and the CPU registers. See below.



VDP shows a dump of 1024 bytes of VDP memory starting at the address in the text box. For each line, first the memory address is displayed, followed by eight bytes in hexadecimal, followed by the same eight bytes in ASCII text. Characters not valid for printing as 7-bit ASCII are represented as a period. A 4-character hex address may be entered here, or a CPU register (R0-R15) may be entered, and the memory display will track the address pointed to by that register. Press *Apply* when changing the value in the text box. In addition, the *Prev* and *Next* buttons may be used to advance 1k forwards or backwards in memory. In this mode you can change the Workspace Pointer (WP) or Program Counter (PC) in the entry box, as well as directly modifying VDP memory and the VDP registers. See below.



GROM shows a dump of 1024 bytes of GROM memory starting at the address in the text box. For each line, first the memory address is displayed, followed by eight bytes in hexadecimal, followed by the same eight bytes in ASCII text. Characters not valid for printing as 7-bit ASCII are represented as a period. A 4-character hex address may be entered here, or a CPU register (R0-R15) may be entered, and the memory display will track the address pointed to by that register. Press *Apply* when changing the value in the text box. In addition, the *Prev* and *Next* buttons may be used to advance 1k forwards or backwards in memory. In this mode you can change the Workspace Pointer (WP) or Program Counter (PC) in the entry box, as well as directly modifying GROM memory. See below.

7.7 Changing Memory

In addition to controlling the memory address to view in the control input box, you can also change memory and registers by entering an address or register followed by an equals sign '=', and the value that you wish to set. It is recommended, but not required, that the emulator be breakpointed while changing values, to avoid unexpected side effects.

The following values may be entered. You must click *Apply* for the change to take effect.

PC=xxxx Where xxxx is a 4-digit hex value, this changes the Program Counter to the specified value. Valid in all views.

WP=xxxx Where xxxx is a 4-digit hex value, this changes the Workspace Pointer to the specified value. Valid in all views.

xxxx=yy Sets the byte at address xxxx to be yy, where both values are hexadecimal. Valid in the CPU view, where CPU memory is modified, the VDP view, where VDP memory is modified, and the GROM view, where GROM is modified. Note that if you attempt to change any type of ROM, the emulator will verify the change before it does it. If you agree to the ROM change, the ROM in memory is modified with no side-effects, and the change will be lost when the emulator is reset. If you decline the ROM change, then the write is performed as if the CPU attempted it. This can be used to trigger hardware and may invoke side effects.

xxxx=yyyy Sets the word at address xxxx to be yyyy, where both values are hexadecimal. Valid only in the CPU view, and modifies CPU memory. See the previous entry for byte writes for notes on ROM access and modification.

Rx=yy Sets the register numbered 'x' to the hexadecimal byte value yy. Valid in the CPU view, where the register is a DECIMAL number from 0 to 15 and modifies CPU registers, or the VDP view, where the register is a number from 0-7, and modifies VDP registers.

Rx=yyyy Sets the register numbered 'x' to the hexadecimal word value yyyy. Valid only in the CPU view, where the register is a DECIMAL number from 0 to 15 and modifies CPU registers.

7.8 Keyboard Shortcuts

All keyboard shortcuts require that Scroll Lock be ON. Some keys may work only with the TI-99/4A configuration with Extended Keyboard active (this is the default). Most of these options also have dialog buttons on the debug dialog. More information on those options can be found in section 7.5.

F1	Run – continues execution if breakpointed.
F2	Step into one instruction (if breakpointed)
F3	Step over one instruction (if breakpointed) (if BL or BLWP, will run till it returns)
F5	Take incrementing screenshot, unfiltered. The first one will ask for a filename if needed. This is the same as Video->Screenshot (Basic), except that it will not ask for a filename for subsequent captures, instead appending a sequence number.
F6	Take incrementing screenshot, filtered. The first one will ask for a filename if needed. This is the same as Video->Screenshot (Filtered), except that it will not ask for a filename for subsequent captures, instead appending a sequence number.
F9	VDP character dump - shows the current character set instead of the picture.
F10	Dump CPU and VDP to files.
F11	System Maximum mode. This is the same as Options->CPU Throttling->System Maximum.
F12	Trigger LOAD interrupt.
CTRL+F5	Set CPU normal. This is the same as Options->CPU Throttling->Normal.
CTRL+F6	Set CPU Overdrive. This is the same as Options->CPU Throttling->CPU Overdrive.
CTRL+F7	Set System Maximum. This is the same as Options->CPU Throttling->System Maximum.
CTRL+F8	Set CPU slow. This is the same as Options->CPU Throttling->CPU Slow.
CTRL+F12	Reset Emulator (reloads all ROMs). This is the same as File->Reset.
Home	Bring up debugger window. This is the same as Edit->Debugger.

8. Loading Files

While it is not the intention of this document to provide extensive training on how to use a TI, the loading of files takes a combination of emulator understanding and original TI usage understanding. As such, this section provides a brief summary as to how to get files loaded into Classic99 for use.

8.1 TI Console ROMs

It is not necessary to provide console ROMs - they are built in! The system will automatically start with Classic99, and you may choose another under the System menu. However, note that only the TI-99/4A is fully supported.

If you wish to include a separate ROM set, you can override the built-in ROMs by loading them in Classic99.ini, as described under the cartridge section. Classic99 does not provide native support for a custom console ROM set, so you will need to create a cartridge that includes both the custom ROMs and any cartridge ROM images that you wish to use with it. See the INI section for how to create a cartridge configuration.

8.2 Cartridge ROMs

A number of cartridges are now built into Classic99, however, you may want to add more yourself.

Currently, you must modify the Classic99.ini (it will be created after the first time you run and close Classic99). See the description of the INI file below for details on the file and how a cartridge ROM is specified therein.

You may also select Cartridge->User->Open. This will pop up a dialog that asks for the cartridge to load. It only recognizes V9T9 style cartridges, with the naming convention of <name><T>.bin, where 'T' is a letter indicating the type of ROM (either C, D, or G). You need select only one file of the group, and Classic99 will automatically find and load the rest of the files in the same folder. (If you have trouble, you can look at the debug log to see what it attempted to do). Cartridge ROMs may not be zipped or packed into one file, and '379' style cartridges may not be loaded with this method.

8.3 Disk Images (.DSK)*

These are images of diskettes made popular by PC99 and V9T9. These images come in various sizes, with the most common sizes being 90k, 180k, and 360k. The TI file system allows 127 files on a single disk, with no subdirectories.

Classic99 does not currently handle disk images. Classic99 stores files directly on the disk, which appears to the emulated TI as just another kind of disk controller.

The most popular way to extract file images from disk images is to use TI99Dir by Fred Kaal, which you can find here: http://members.ziggo.nl/fgkaal/Software/sw_ti99dir.html

8.4 Files on a Disk

Once you have the files on the disk, they are often in V9T9 format, especially if you have used TI99Dir to extract them from a disk image. V9T9 was one of the original TI Emulators for DOS - and the first free one. To work around differences in the TI filesystem to DOS, it does a couple of unusual things.

First, certain characters are replaced with high ASCII replacements in order to work in the file system. For instance, a forward slash becomes an overscore line. Classic99 doesn't support this renaming scheme. Instead, the following TI characters should be replaced with '~' on the disk: ?, /, >, <, :, ", *, |

(Note: in a pinch, Classic99 will attempt to work with the extended filenames, but this is not guaranteed).

Classic99 will deal with the internal name. Also, the length of these files was limited to the old DOS 8.3 naming format, meaning any filename longer than 8 characters (TI disk controllers supported at least 10) has an extension. For instance: MYFILENA.ME -- rename it to remove the '.' (MYFILENAME) for Classic99.

Except for the changing of certain characters to '~' and removing the period, do not rename V9T9 files on the disk. Doing so will break them, because the real filename is embedded in the header and must match.

Note that these changes hurt interaction back to V9T9 and other tools that recognize this format. If you need to interwork with these tools, I recommend keeping filenames to 8 characters or less, and avoid all punctuation except for ';' and '_'.

Classic99's preferred file format is called 'TIFILES' or sometimes 'XMODEM'. It is the format that was globally used by file transfer programs to upload and download files to BBSs, which were likely running on PCs, while preserving the TI standard information. The filename is not included in the header, meaning that the file on disk is the only filename. The same limitation of punctuation above must be followed, however, you are free to rename or move these files on your disks at will. They can also be sent directly back and forth with real TIs using a terminal program and a serial connection, while V9T9 files must be converted to do so.

It may be most convenient to convert V9T9 files to TIFILES format in order to use them with Classic99. To do this, you can run a copy program inside Classic99, or use TI99Dir.

Classic99's implementation of the file system has none of the restrictions of the TI disk system. Folders may contain an unlimited number of files, filenames may be of any length, and subdirectories may be used. However, the original TI disk system imposed a limit of 127 files in a folder, and 10 character filenames. Therefore, exceeding these limits may cause some software, such as disk management software, to malfunction or not see all the files. It is recommended that file management be performed in Windows.

8.4 Archiver 3.03 Files

Archiver 3.03, often abbreviated as Arc303, is the TI's equivalent of ZIP files - they are compressed files that usually contain more than one other files inside them.

Usually these end with an '@' character to indicate they are archived, although some PC distributions may use a .ARC or .ARK extension instead. To extract the files, you will need a copy of Archiver 3.03, which is included in the Classic99 distribution on DSK1 as *ARC303G.*

Copy the archived file (@) into the DSK1\ folders (or any other if you set up more) using Windows. The example assumes you are putting everything into DSK1. If not, just substitute the correct drive number.

Start CLASSIC99.EXE. Select *Editor/Assembler* under Carts->Apps. Press a key then select *EDITOR/ASSEMBLER* from the main menu. Next select *5 RUN PROGRAM FILE* from the Editor/Assembler menu. *(NOTE: Is is from this sequence that 'program' image files on the TI are often called "EA#5" files.)* Enter **DSK1.ARC303G**

Press a key to pass the Shareware notice, then select *2) Extract Files*. Enter your source drive *(1)*, and the name of the '@' file for Source Filename. Select your output drive (may still be 1), and for *Extract all files?* you will usually say *Y.* If you select the same source and output drive, it will also ask *Swap Disks?*, of course with Classic99 you may say *N*.

You'll see *BACK / REDO / Any Key to Begin*, just hit the space bar and the files will be extracted. You can then quit the archive program and move on to running the files themselves.

8.5 TI BASIC Files

No cartridges are required to be enabled.

Select *TI BASIC* from the main menu. The system will report *TI BASIC READY* and provide a cursor.

You will almost certainly need CPU THROTTLING *on* - it's under Options. If you get rapid key repeat, make sure it is set to Normal.

Enter **OLD DSK1.FILENAME**, where DSK1 is the disk folder you have the file in, and FILENAME is the name of the file. Note that the TI likes everything in uppercase, but it often doesn't matter with Classic99 because the Windows file system is not usually case-sensitive. Classic99 will also support DSK being uppercase or lowercase (although the real TI will not.)

Once it loads successfully, type **RUN** will start it up. (For programs that you don't have to type during, turning CPU throttling to *CPU Overdrive* at this point makes slow programs more bearable.)

8.6 Extended BASIC (XB) Files

You must have Extended BASIC selected as the cartridge under Carts->Apps.

Select Extended BASIC from the main menu. The system will report ** READY ** and provide a cursor.

You will almost certainly need CPU THROTTLING *on* - it's under Options. If you get rapid key repeat, make sure it is set to Normal.

 BUG: The 'slow down keyboard when unthrottled' option under 'Options' does not work in Extended BASIC.

Enter **OLD DSK1.FILENAME**, where DSK1 is the disk folder you have the file in, and FILENAME is the name of the file. Note that the TI likes everything in uppercase, but it often doesn't matter with Classic99 because the Windows file system is not usually case-sensitive. Classic99 will also support DSK being uppercase or lowercase (although the real TI will not.)

Once it loads successfully, type **RUN** will start it up. ((For programs that you don't have to type during, turning CPU throttling to *CPU Overdrive* at this point makes slow programs more bearable. In XB it usually will not affect the speed of sprites, but it will improve the responsiveness!)

8.7 PROGRAM IMAGE files (E/A#5)

You must have Editor/Assembler selected as the cartridge under Carts->Apps.

Select *EDITOR/ASSEMBLER* from the main menu. Select *5 RUN PROGRAM FILE* from the menu. For *File Name?*, enter **DSK1.FILENAME**, where DSK1 is the disk you have the file(s) on, and FILENAME is the first program file to load. Program images were limited to 8k, so for programs larger than 8k, multiple files were created by adding 1 to the last character of each name. Some programs numbered the files (ie: POPEYE1, POPEYE2, POPEYE3), but many just allowed the last letter to increment (ie: LASSO, LASSP, LASSQ).

Program files will autostart after being loaded.

8.8 Object Files (E/A#3)

It's very rare to come across these as Program Images were faster, smaller, and easier to load, however, just in case you do need to load one, here are the steps. NOTE: Some object files are intended for Extended BASIC, they are not discussed here as these are usually programmer utilities and not standalone programs.

You must have Editor/Assembler selected as the cartridge under Carts->Apps.

Select *EDITOR/ASSEMBLER* from the main menu. Select *3 LOAD AND RUN* from the menu. For *File Name?*, enter **DSK1.FILENAME**, where DSK1 is the disk you have the file(s) on, and FILENAME is the file you are currently loading. It's possible, though even more rare, to have to load more than one file. In that case, keep entering filenames until all files have been loaded. You may need additional details from the program's author.

If the program does not autostart after loading the file(s), hit Enter on the *File Name?* prompt to enter a blank line. You will be prompted with *PROGRAM NAME?,* which unfortunately you need to know. However, most E/A#3 files use **START** or **MAIN**.

9. Using the TI Keyboard

Many programs refer to 'REDO', 'BACK', etc, keys that are not immediately obvious. The TI used FCTN and the number keys for these special terms, so on Classic99 it's Alt and the number key. If Scroll Lock is OFF, and the extended keyboard is active (this is the default), you can also use your PC's 'F' keys F1 through F10.

BUG: This entire discussion covers the TI-99/4A and 2.2 TI99/4A. The TI-99/4 had a smaller keyboard which is not extended with PS/2 support in Classic99. Most 99/4 functions are handled with Shift plus a letter key, as it did not have a FCTN key. These keys are not currently documented!

The TI key names are laid out as follows:

FCTN+ 1	DELETE	Deletes the character under the cursor.
2	INSERT	Toggles Insert mode.
3	ERASE	Erases the current line
4	CLEAR	Stops a BASIC or XB program.
5	BEGIN	Depends on the program being used.
6	PROCEED	Depends on the program being used.
7	AID	Depends on the program being used.
8	REDO	In XB, recalls the last entered line.
9	BACK	Depends on the program being used.
=	QUIT	Restarts the TI (soft reset).

The following keys are not necessary to know in Classic99 as the PS/2 keyboard maps them to the keys you would expect to press, but they are useful to know for standard mode or just reference in documentation.

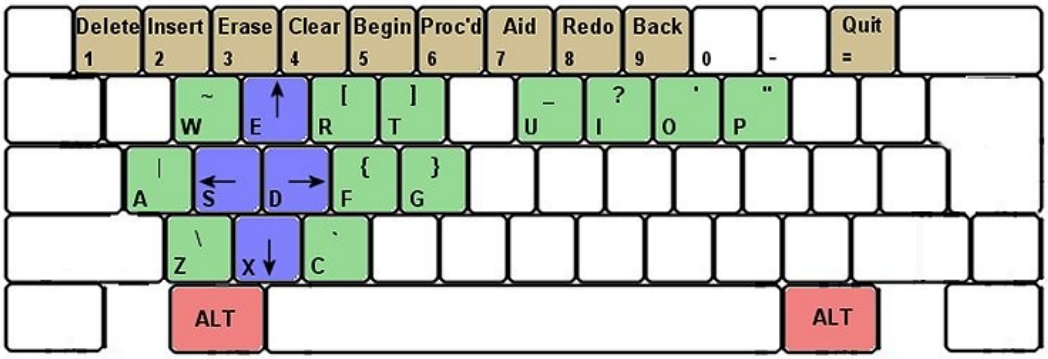
Arrow Keys:

FCTN+ E - Up
S - Left
D - Right
X - Down

Extra characters:

FCTN+	W - ~	I - ?	F - {
	R - [	O - '	G - }
	T -]	P - "	Z - \
	U - _	A - |	C - `

You can reference this map in Classic99 as Help->Keyboard Map



10. Classic99.ini

10.1 Description of Settings

This is the main configuration file for Classic99. It contains all settings needed to operate and configure the emulator, including many settings that are not available through the GUI. Since these functions may be useful to some people, the settings are documented here.

Use care with changing the values – invalid settings may cause the emulator to crash. If you have trouble, you can delete this file and Classic99 will create a new one with default settings after being started and closed normally.

[audio]

max_volume

scaler for maximum volume from (percentage) 0-100 (all audio is scaled to this)

sid_blaster

0 SID Blaster is disabled
1 SID Blaster is enabled

[Disk0]

Note: this can be Disk0 through Disk9, representing DSK0-DSK9 in the emulator

Type

0 this disk is not configured
1 this disk uses FIAD (files in a directory) access
2 this value can not be set – the disk will not be created.
3 this disk uses sector-based DSK image files.

Path

string Specifies the path to the folder for FIAD drives. May be a full path or a relative path. For sector-based DSK images, specify the path and filename to the DSK file.

Entries prefixed with FIAD are applicable only to FIAD disks.

FIAD_WriteV9T9

0 write files using TIFILES headers (default)
1 write files using V9T9 headers (not recommended, V9T9 filenames are limited to 10 characters and have mapping problems with non-alphanumeric characters)

FIAD_ReadTIFILES

0 Reading TIFILES files is disabled, files are never detected as TIFILES
1 Reading TIFILES files is enabled (default)

FIAD_ReadV9T9

0 Reading V9T9 files is disabled, files are never detected as V9T9
1 Reading V9T9 files is enabled (default)

FIAD_ReadTextAsDV

0 Do not read Windows text files as Display/Variable
1 Read Windows text files as Display/Variable (extensions must be TXT, COB or OBJ)

FIAD_ReadTextAsDF

0 Do not read Windows text files as Display/Fixed
1 Read Windows text files as Display/Fixed (extensions must be TXT, COB or OBJ)

FIAD_ReadTextWithoutExt

0 Do not treat files without extension as Windows text (must be this value to read as DF128)
1 Treat files without extension as Windows text (if they are not detected as V9T9 or TIFILES)

FIAD_ReadImgAsTIAP

0 Do not read images as TI Artist files
1 Read images as TI Artist files (NOT IMPLEMENTED YET)

FIAD_WriteDV80AsText

0 Write DV80 files in TI format (default)
1 Write DV80 files as Windows text files

FIAD_WriteAllDVAsText

0 Write all DV files in TI format (unless WriteDV80AsText is set)
1 Write all DV files as Windows text files (regardless of the setting of WriteDV80AsText)

FIAD_WriteDF80AsText

0 Write DF80 files in TI format (default)
1 Write DF80 files as Windows text files

FIAD_WriteAllDFAsText

0 Write all DF files in TI format (unless WriteDF80AsText is set)
1 Write all DF files as Windows text files (regardless of the setting of WriteDF80AsText)

FIAD_AllowNoHeaderAsDF128

0 Do not recognize Windows files without a header
1 Read Windows files without a header and no extension as DF128 (default)

Entries prefixed with Image are applicable only to Image (sector) disks.

IMAGE_UseV9T9DSSD

0 Disk image files on this drive are numbered from 0-1439 normally (default)
1 Disk images on this drive reverse the order of sectors 360-719 (V9T9)

[emulation]

AVIFilename

string Filename that will be used with 'Start Recording' to write an AVI file.
Defaults to C:\Classic99.AVI

cputhrottle

0 run CPU overdrive mode (fast CPU)
1 run system with normal timing
2 run CPU at system maximum speed

systemthrottle

0 VDP interrupts sync with the CPU speed (system maximum)
1 VDP runs in realtime, not tied to CPU speed

maxcpf

50000 Maximum cycles per frame (either 50hz or 60hz). Assumes a 3MHz clock.

pauseinactive

0 Do not pause emulation when window is not focused
1 pause emulation when window is not focused

slowdown_keyboard

0	Do not slow down keyboard auto-repeat
1	Slow down keyboard auto-repeat (99/4A only!) (during GPL reads only)

system

0	TI-99/4
1	TI-99/4A
2	TI-99/4A v2.2

ps2keyboard

0	Disable PS/2 keyboard (always set for 99/4)
1	Enable PS/2 keyboard (99/4A only)

sams_enabled

0	Disable SAMS memory expansion card
1	Enable SAMS memory expansion card

sams_size

0	128k SAMS card
1	256k SAMS card
2	512k SAMS card
3	1MB SAMS card (default)

[joysticks]

active

0	Do not map joysticks
1	Map joysticks using joy1mode and joy2mode

joy1mode (and *joy2mode* for joystick 2)

0	Keyboard (arrow keys and tab)
1	PC Joystick 1
2	PC Joystick 2

[debug]

scrambleRAM

0	initialize RAM to 0 on power-on reset
1	initialize RAM to random values on power-on reset

[roms]

cartgroup

0	Apps
1	Games
2	User

cartidx

-1	No cartridge selected
??	index of the cartridge from the appropriate menu

[UserCart0] (and so on up to 99)

name

string	Name of the cartridge. If blank or missing, the cartridge is ignored.

message

string	Compatibility notes, if desired (optional – will show as 'known issues' under Help).

rom0 string (0-15 for a total of 16 ROMs per cartridge)
Contains notes for loading the cartridge. This is a pipe-limited row of data, formatted like so:
T|AAAA|LLLL|filename

T is the single character ROM type. Most carts only use G, C and X types.

C	CPU ROM
D	DSR ROM
E	DSR ROM bank 2 (used for p-code)
G	GROM*
P	P-Code GROM
R	CPU RAM - this is loaded like CPU ROM but is not flagged as read-only!
S	Speech ROM
V	VDP RAM
X	XB Bank 2 ROM (called 'D' in V9T9 notation)
A	AMS memory, RLE encoded (NOT WORKING CORRECTLY)
K	Paste string for the keyboard after boot (to autostart tasks, not a filename)
3	Packed banks with a '379 style decoder. Similar to XB style but more banks and all in one file. Note that this assumes a load at >6000, so you are specifying the offset in the banking space, with bank 0 at >0000, bank 1 at >2000, etc. SuperSpace carts (CRU >400) can be loaded this way for now, too, but will likely split them up later. Max size is 64k.

(* GROM is special - to support multiple GROM bases, you can now load GROMs into different memory bases - up to 16 of them! By default, GROMs load into memory base 0. Append a hex number 1-F after the G to specify a different base (like: G1). The console will detect these and add them to the start menu! Note that not all GROMs will work at alternate bases. Also note, in the real console, the console GROMs appear at all bases, so you can't load alternate bases at less than >6000. Classic99 does not support using this with cartridges that include CPU ROM. Each GROM base is offset 4 from the previous, ie: GROM Read is >9800, >9804, >9808, etc.)

AAAA - load address in hexadecimal

LLLL - length of data in hexadecimal (will override actual size of file)

filename - filename on disk to load

This system supports both raw ROM files, and ROM files with a 6 byte GRAM Kracker style header. The header is detected by checking if the first byte is >00 or >FF, and if the 5th and 6th bytes represent the load address. The header will be ignored if detected - the data in this INI file is considered authoritative.

[video]

FilterMode

0	None
1	2xSAI
2	Super 2xSAI
3	Super Eagle
4	NTSC TV Filter
5	HQ4x

frameskip

0	Number of frames to skip drawing. This may affect interrupts as well. Default 0.

fullscreenmode Note: not all modes are supported on all cards. Classic99 does 16 bit rendering internally.

1	320x240x8

2 640x480x8
3 640x480x16 (default)
4 640x480x32
5 800x600x16
6 800x600x32
7 1024x768x16
8 1024x768x32

heatmapfadespeed
25 Number of pixels updated in heat map per frame. Higher number is faster fade but more CPU is needed to draw it. Default is 25.

hzRate
50 50hz interrupt (PAL)
60 60hz interrupt (NTSC)

MaintainAspect
0 Do not maintain aspect ratio (allow any size window)
1 Force aspect ratio (fix proportions of window)

StretchMode
0 Do not stretch (fastest)
1 DIB - use GDI to stretch
2 DX - use Direct-X to stretch
3 DX Full - use a full-screen Direct-X mode (set by fullscreenmode)

[tvfilter] These settings apply only to the NTSC TV filter
scanlines
0 Don't draw scanlines
1 Draw scanlines

hue
100 Value from 0-200, with 100 being the default setting

saturation
100 Value from 0-200, with 100 being the default setting

contrast
100 Value from 0-200, with 100 being the default setting

brightness
100 Value from 0-200, with 100 being the default setting

sharpness
100 Value from 0-200, with 100 being the default setting

10.2 Example of adding a User Cartridge

These lines will add AtariSoft's Pole Position. The assumption is that you have the files in V9T9 format, named POLEPOSC.BIN, and POLEPOSD.BIN. Store the files into the Classic99\MODS folder. Make sure Classic99 is not running, then edit the Classic99.ini file, adding these lines:

[UserCart0]
name="Pole Position"
ROM0=C|6000|2000|MODS\POLEPOSC.BIN
ROM1=X|6000|2000|MODS\POLEPOSD.BIN

Pole Position is not included - this is just an example! :) Read the [UserCart0] section above for detailed on what the fields mean!

Note that you may have to change the number after 'UserCart' to the first free number not already in the file. The first line sets the name that will appear under Cartridge->User. The second line tells Classic99 that the first ROM to load is POLEPOSC.BIN, indicates that it is type 'C' (cartridge ROM), specifies the load address and size, and the path to the file. It does the same with POLEPOSD.BIN, except that the type is 'X' (bank 2 XB style cartridge).

If you do not know the load address of a ROM file, 6000 is almost always a good guess as this is the base address of the cartridge port for both ROM and GROM. You can get the size from the file if you need to, and the type from the last letter of the filename before BIN: C is cartridge ROM, D is XB bank 2, and G is GROM.

11. Files Included on DSK1 with Classic99

ARC303G - Barry Boone's archiver - use this to extract '@' or .ARK files. Select cartridge Editor/Assembler, select option 5 Run Program File, and enter 'DSK1.ARC303G'

CARDEMO - Simple TI BASIC demonstration by Mike Brent. No cartridge required. Start TI BASIC and enter 'OLD DSK1.CARDEMO'. When it is finished, enter 'RUN'

DEMO - Simple assembly demonstration by Mike Brent. Select cartridge Editor/Assembler. Select Option 5 Run Program File, and enter 'DSK1.DEMO'

DEMP and **DEMQ** - part of DEMO. Do not run directly, DEMO will load them.

SPACEFIGHT - Simple TI BASIC game by Mike Brent - no cartridge required. Start TI BASIC and enter 'OLD DSK1.SPACEFIGHT'. When it is finished, enter 'RUN'. The object is to patrol Earth's orbit and destroy only hostile craft. Keys A, Z, S, and X control speed and shields. Speed increases how often you find other craft in orbit. Shields reduce how much damage you take if attacked. Both increase your energy usage. The alarm will sound to warn of a detected craft, and an image of it will appear on the scanner. You must quickly decide whether to fire your torpedos with the space bar (hint: most hostile craft have weapons). If you destroy a friendly craft, you will lose points. If you do not destroy a hostile craft, it will attack you. You will also occasionally reach a starbase, it will refuel your energy and torpedos (if you don't destroy it!).

STRANGER - Simple TI BASIC text adventure by Mike Brent - no cartridge required. Start TI BASIC and enter 'OLD DSK1.STRANGER'. When it is finished, enter 'RUN'. You will be presented with an on-screen story - type simple two and three word commands (ie: *get rope*, *go north*, etc) and try to survive!

XBDEMO - Simple Extended BASIC demo by Mike Brent. Select cartridge Extended BASIC. Enter 'OLD DSK1.XBDEMO'. When it is finished, enter 'RUN'.

ABOOT - AMS card Boot menu by the SW 99ers. This lets you launch the preconfigured AMS program - TI-Nopoly. Select cartridge Editor/Assembler, select option 5 Run Program File, and enter 'DSK1.ABOOT'. Note that AMS must be enabled in the configuration for this to work.

AEMSSYS, **ABOOTMOD**, **ABOOTMODS**, **AEMSRES1**, **AEMSRES2** - part of ABOOT. Do not run directly, ABOOT will load them.

ASHOE - advanced boot loader for ABOOT - used to retain menu items. If you don't already know it, it probably won't be too useful today. Someday!

AMSTEST4 - AMS card test program. Does not work properly from ABOOT! Select cartridge Editor/Assembler, select option 5 Run Program File, and enter 'DSK1.AMSTEST4'. As this test is slow you may want to set CPU Overdrive to watch it. Note that AMS must be enabled in the configuration for this to work.

TI-NOPOLY - AMS-based Monopoly. This program must be booted from the AMS Boot menu, as described above. Select cartridge Editor/Assembler, select option 5 Run Program File, and enter 'DSK1.ABOOT'. Note that AMS must be enabled in the configuration for this to work. Enter the menu and simply select TI-Nopoly to start it.

You may drop other files into this folder, either from http://www.harmlesslion.com/cgi-bin/showprog.cgi?TI-99/4A or any other site offering files (sometimes called FIAD) for Classic99 or V9T9. Subfolders are also possible with Classic99, the backslash is accessed by pressing FCTN(ALT) – Z (or pressing backslash with the PS2 keyboard emulation).

Here's your excitement: The call was coming from inside the house! GASP!