PC Atari Emulator

Version 2.0

Users Manual

John Dullea (jxd230@psu.edu)

August, 1997

Document reformatting and conversion by Jim Leonard (trixter@hornet.org)

1 What is PCAE?

The PC Atari Emulator, or PCAE, is an MS-DOS program designed to emulate the famous Atari 2600 Video Computer System on PC's compatible with the Intel 80486 CPU or better. It accomplishes this with an emulation engine that is written almost entirely in assembly language, with extra features written in Borland Pascal 7.0 for DOS.

System Requirements

OS:	MS-DOS, though it will run from within a Windows DOS prompt. Some users have successfully run it from within an OS/2 DOS prompt by setting the "Allow access to hardware timer" flag to ON.
CPU:	Intel 80486 or greater, for the BSWAP instruction (though a Pentium is highly recommended for performance purposes)
RAM:	540K free low DOS memory (though 590K or greater is strongly recommended). Also, at least 2Mb of XMS memory is needed to store sound samples (setting a larger sound buffer requires more XMS memory)
Other:	A mouse is required for single-player paddle emulation A 4-button joystick is strongly recommended, ESPECIALLY a Gravis GamePad. Two joysticks are strongly recommended for two-player games To use joysticks to emulate paddles two and three, analog joysticks are required. A special home-built board is necessary for connecting real Atari controllers to the PC.
Sound:	A SoundBlaster or 100% compatible sound card is required for sound support. To get perfect sound from within a Windows DOS prompt the sound card must be compatible with Sound Blasters utilizing DSP versions 2.00 or greater.

Features

- Very fast emulation due to the Pentium-optimized assembly implementation
- Full collision checking
- One paddle is emulated using the mouse, and two other paddles can be emulated using analog joysticks. The paddle assignments can be rotated from the main menu.
- Emulates the Atari Video Touch Pad (for Star Raiders) and Atari Keyboard Controllers via the keyboard.
- Emulates Atari Indy 500 Driving Controllers (for Indy 500) and CBS Booster-Grip (for Omega Race) via the keyboard or joysticks
- Support for two joysticks (four-button joysticks can control select, reset, and both players' fire buttons in all games except CBS 12k games)
- Built-in menu allows easy selection of games
- Support for a game profile file that contains all the cartridge types and controller settings for every game in your library
- On-line help is available in the menu system, integrated debugger, and while playing games
- Built in interactive debugger that supports bank-switched images
- Built-in disassmebler for images 4k or smaller, as well as many 8k, 16k, Super-Chip, and CBS games.
- Support for Atari 8k, Atari 16k, Super-chip, Parker Brothers, CBS RAM-Plus, TigerVision, M-Network, and Pitfall II bank switched cartridges
- Supports 64k cartridges containing 32 independent 2k games (32-in-1 games)
- Supports 64k Brazilian Megaboy cartridge

- Supports extended Super-chip and TigerVision bankswitching specifications
- Support of Starpath Supercharger single-load and multi-load games
- Sound emulation using Ron Fries' TIA Emulation Library on 100% Sound Blaster-compatible sound cards
- Emulation speed can be slowed down to a user-selectable number of frames per second for especially fast computers
- The sound buffer size can be increased for better sound quality or decreased for faster emulation
- The sound playback rate can be varied for compatibility with certain sound cards
- Includes two versions of the emulator, one that runs in standard 320x200x256 VGA mode and one that uses a non-standard 160x200x256 mode for greater speed (might not be compatible with all VGA cards and/or monitors)
- Each executable supports an optional extended VGA resolution for games which utilize more than 200 scan lines (might not be compatible with all VGA cards and/or monitors)
- Support for connecting real Atari 2600 controllers to a PC using a home-built board that connects to a PC joystick port and bidirectional parallel port
- Users with sound cards that are 100% compatible with Sound Blaster cards with DSP versions 2.00 and greater will enjoy improved sound in DOS due to auto-initializing DMA. With such cards, the sound from within Windows 95 should be of the same quality as from within DOS.
- Supports the PAL graphics palette in addition to the NTSC palette
- Allows joystick buttons to be remapped from the main menu
- Built-in vertical autocentering for different games, which can be disabled by the user

Emulation Speed

The emulated speed of games run on the software varies by game and system type. I have found that a Pentium-90 system can run most games at a slightly higher speed than the original Atari console, but there are a few exceptions for which a faster computer might help. I have received e-mails claiming that i486-120 computers can just reach 100% emulated speed using the emulator.

2 Setting It Up

Installing the Software

Most likely you will acquire the emulator via a ZIP archive file that was either downloaded from somewhere or placed on a CD-ROM. The first thing to do is to create a new directory in which to place all the files contained in the archive file. For example, DOS users would type:

```
C> CD\                     ← move to the root directory
C> MD PCAE                 ← create the new directory
C> CD\PCAE                 ← move to the new directory
```

The next thing to do is to place all the files in the archive file into the new directory. There are a number of programs available for extracting a ZIP file, such as PKWare's PKUNZIP for MS-DOS, or the shareware WINZIP for MS-Windows. Symantec's Norton Navigator File Manager for Windows 95 is also a good choice. Using whichever method you choose, extract all the files contained in the ZIP archive into the new directory you just created.

Setting Up Sound Support

PCAE supports sound emulation of the Atari 2600 using 100% Sound Blaster-compatible sound cards. Although using "true-blue" Creative Labs hardware is strongly recommended, there are a great many sound cards that will work with the emulator. In all cases, if you intend to turn on sound support in the emulator, you should first check to see if your sound card is compatible with the emulator and that it is properly configured.

PCAE supports 100% Sound Blaster-compatible sound cards using IRQ settings in the range 0 to 15 and 8-bit DMA settings in the range 0 to 3. To my knowledge, all SB-compatible sound cards use an 8-bit DMA setting in the range 0 to 3, but any that do not will not work with the emulator. To find out or change your card's settings, you should read the installation manual that was included with your hardware. Most sound cards (especially those from Creative Labs) come with a test program that will tell you the current IRQ and DMA settings of your card. Another way to possibly find out the settings is to check them from within the Windows 3.x or Windows 95 Control Panel.

If you have verified that your sound card's IRQ and DMA settings are compatible with PCAE, the next step is to ensure that the BLASTER environment string is properly set. This environment string is the mechanism by which PCAE gets your sound card's settings, and is absolutely necessary for sound emulation to work properly; otherwise, lockups or crashes will likely occur. Usually this line is set automatically by your sound card's installation software, but this is not always the case. You can check to see if it is set by typing "SET" from the MS-DOS prompt:

Example:

```
        C> **SET**
        TEMP=C:\WINDOWS\TEMP
        winbootdir=C:\WINDOWS
        SOUND=F:\SBPRO
        windir=C:\WINDOWS
        COMSPEC=C:\COMMAND.COM
        PATH=C:\;C:\DOS;C:\WINDOWS;C:\WINDOWS\COMMAND;…
                .
                .
                .
        BLASTER=A220 I7 D1                       ← This is the line you're interested in
```

The basic format of the BLASTER environment string is as follows: There should be a field in the form A#, where the number is the sound card's address in hexadecimal. For example, my Creative Labs Sound Blaster Pro is at address 220h. Second, there should be a field in the form I#, where # is the IRQ setting of the card. Finally, there should be a field in the form D#, where # is the 8-bit DMA setting of the card. More advanced cards will probably have additional fields on the line, for such features as 16-bit DMA and MIDI ports.

If this line is present in your environment string list, then, as long as the numbers match the actual hardware settings, there is nothing more to check regarding your sound hardware. However, if the line is not present, you will have to see to it that one is present before running the emulator. This can be done manually or it can be added to a batch file, such as AUTOEXEC.BAT. In either case, the process of adding or changing an environment string is a simple DOS command:

```
        C> SET BLASTER=A220 I7 D1                ← Example of setting the BLASTER line
```

All you do is type the DOS "SET" command as above but add the line to be added after it. "BLASTER=" identifies the name of the environment string and "A220 I7 D1" is the environment string it will refer to.

Please note that environment strings from SET commands do not use up any low DOS memory in the same manner as TSR (terminate-and-stay-resident) files, but they do take up environment string space. Usually the largest environment string you will have is your PATH statement, which is usually set in the AUTOEXEC.BAT file. If you look in this file, you will probably see lines that begin with "PATH" instead of beginning with "SET". Because the PATH environment string is used by everyone, the people at Microsoft included a special command called PATH that sets the PATH environment string. Using "SET PATH=…" instead of "PATH" would yield the same result. At any rate, if you do not have a BLASTER environment string and try to add one, you might run out of environment string space if you have a lot of other environment strings or a few large ones. You can check this by typing "SET" after setting the BLASTER environment string and checking to see if the entire string is present. If it isn't there or is truncated, you might have to either (1) shorten or eliminate another environment string (such as your path) or (2) increase the amount of environment string space. The method for increasing environment string space varies widely with operating systems and is beyond the scope of this document. The only method with which I am familiar is the "/E:xxxx" command line option in MS-DOS COMMAND.COM, which would be specified in CONFIG.SYS as follows:

```
        SHELL=COMMAND.COM /P /E:1024
```

This sets the environment string area to 1024 bytes upon bootup. The /P option is necessary to tell DOS that COMMAND.COM is to be loaded permanently. This should work for those using MS-DOS, Windows 3.x, or those running PCAE after restarting Windows 95 in MS-DOS mode. For Windows 95 MS-DOS prompts, you might not have to change the setting since it has an "Auto" feature, but it is available by exploring to the MS-DOS prompt, right-clicking on it, and selecting "Properties". The initial environment string area size can then be changed by clicking on the "Memory" tab and changing the "Initial environment" setting.

Note: Several users with Pro Audio Spectrum sound cards have expressed problems getting the sound to operate properly. By default, the emulator plays sound at a rate of 31.4 kHz which may be too fast for some cards. There is a feature in the popup menu that allows the playback rate to be set to a lower value which hopefully should help matters (to be discussed in a later section).

Once your BLASTER environment string is set properly, you might want to check to see if your sound card's mixer settings (if any) are set properly. For DOS users, this usually involves a mixer utility that comes with your sound card. For example, my Sound Blaster 16 SCSI-2 has two programs, SB16SET (which is loaded in my AUTOEXEC.BAT) and SB16MIX, which is a TSR that lets me control the mixer settings. This is important because PCAE uses digital sound output for the majority of Atari 2600 sound emulation, but uses FM sound output for the music in Pitfall II: Lost Caverns. For the sound to be properly balanced, the mixer settings for digital and FM sound output should be of equal value. Windows users will probably have similar utilities for their sound cards, and Windows 95 users can try double-clicking on the little speaker icon in the system tray at the bottom right. This should bring up a mixer window that enables control over different settings. FM sound output is usually controlled by the "Midi" slider, and digital output by the "Wave" slider. Be sure also to pay attention to the "Master" slider to ensure that you get sound.

The final issue regarding sound setup is your general memory configuration. PCAE pre-calculates sound samples upon startup and stores them in XMS memory. For buffer sizes of about 4k, about 2Mb of XMS is required. Setting a larger sound buffer size causes the emulator to recalculate the sound samples for the new buffer size and requires more XMS memory. Likewise, smaller buffer sizes relax the XMS requirement. In any case, you will need to have an XMS memory manager loaded to run the emulator with sound support. Managers like HIMEM.SYS provide XMS support and are adequate for this purpose.

To determine whether you have XMS memory, type "MEM" from the DOS prompt, which is a command that comes with MS-DOS and Windows 95. It will display a memory summary that will tell you how much XMS memory, if any, you have available. If none is present, you might have to load an XMS memory manager. The procedure is outside the scope of this document, but for those with MS-DOS, Windows 3.x, or those running PCAE after restarting Windows 95 in MS-DOS mode (NOT merely an MS-DOS prompt) it usually involves loading a driver called HIMEM.SYS into the CONFIG.SYS file on your boot drive:

```
        DEVICE=HIMEM.SYS
```

Your configuration might vary from this, depending on your operating system and the location of your XMS memory manager. Typical directories in which it might reside are \DOS and \WINDOWS. If you are unsure how to load such a driver, you should contact the manufacturer of your operating system.

The amount of XMS memory available in a Windows 95 DOS prompt can be altered by exploring to the MS-DOS prompt, right-clicking on it, and selecting "Properties". Then, clicking on the "Memory" tab will allow configuration of the available memory.

Note for Windows Users

PCAE requires that all its support files reside in the same directory, and that they reside in the same directory from which the program is run. When setting up a Windows 3.1 or Windows 95 shortcut, be sure to specify the PCAE directory as the one in which the program should start. Leaving this entry blank will likely cause the program to abort with an error since it always looks in the current directory for its support files and might not find them.

3 Playing Games

Once the software has been set up and the sound card is configured properly, you are ready to try the emulator with any games that you own. PCAE requires that all games have the extension .BIN. Games can be run in two ways: from the command line using arguments, or using the built-in menu. The easiest way to run games is to place all of them in the same directory, preferably though not necessarily in the same directory as PCAE and use the built-in menu. When trying the emulator for the first time, it is recommended you first use PCAE.EXE instead of PCAE160.EXE, since PCAE.EXE is more compatible with most systems. In this way the non-standard nature of PCAE160's special video mode can be ruled out in the event of problems.

The Command Line

Although not necessary for most users, the command line has the benefit of allowing PCAE to be run from within other game shells or allowing individual games to have Windows shortcuts. The format is as follows:

```
ATARI [path |
       file [BANK | BANK16 | BANKA | BANKM | BANKP | BANKC | BANKSC |
             BANKSP | BANKT | BANK2 | BANK3 | BANKG | PROFILE]
            [32IN1NUMxx]
            [DEBUG]
            [DISASM]
            [FRAMExx]
            [KB | INDY500 | VTP | PADDLE]
```

path	path to a .BIN game directory, e.g. C:\CARTS
file	filename of an individual game

BANK	needed for Atari 8k bankswitched games
BANK16	needed for Atari 16k bankswitched games
BANKA	needed for Activision Decathlon and Robot Tank
BANKM	needed for M-Network bankswitched games
BANKP	needed for 8k Parker Brothers bankswitched games
BANKT	needed for 8k or larger TigerVision bankswitched games
BANKC	needed for 12k CBS RAM-Plus bankswitched games
BANKSC	needed for Atari 16k or larger Super-Chip bankswitched games
BANKSP	needed for Starpath SuperCharger games
BANK2	needed for Pitfall II: Lost Caverns
BANK3	needed for 32-in-1 games
BANKG	needed for Brazilian Megaboy game
PROFILE	Use profile file for bankswitching and controller information

DEBUG	Activate built-in integrated debugger
DISASM	Disassemble a 4k or smaller game to LISTING.ASM
FRAMExx	Limit the number of frames per second to xx for fast computers (or 0 for no limit)
KB	Emulate Atari Keyboard Controllers
INDY500	Emulate Atari Indy 500 Driving Controllers
VTP	Emulate Atari Video Touch Pad
PADDLE	Emulate Atari paddles
32IN1NUMxx	Play 2k game xx (0-31) from 32-in-1 game

Note that the *path* argument is not used with any other command line argument; it is intended for using the built-in menu with game files that are not located in the same directory as the emulator. Some examples of command line usage are:

```
C> PCAE D:
C> PCAE D:\GAMES
C> PCAE PACMAN.BIN
C> PCAE DIGDUG.BIN BANKSC
C> PCAE COMBAT.BIN DISASM
C> PCAE OMEGARAC.BIN BANKC
C> PCAE ROBOTANK.BIN BANKA DEBUG
C> PCAE BNJ.BIN BANKM DISASM DEBUG
C> PCAE INDY500.BIN INDY500 FRAME60
```

Generally it is cumbersome for most users to use the command line in any but the first example unless writing shortcuts to individual games or writing a game shell.

The Main Menu

The main menu is the area from which most games will be run. It provides for easy selection of any game in the directory, as well as configuration of the emulator. In addition, the menu is the area from which the online help reference can be reached.

The menu consists of a text display showing all the .BIN and .PAL files detected in the directory. They will be displayed in color-coded fashion, with different colors denoting different file sizes:

gray	2k and 4k
white	8k
yellow	16k
red	8,448-byte (or multiples thereof) Starpath Supercharger games
green	10,495-byte Pitfall II: Lost Caverns game
black	all others

Files can be selected by using the cursor keys or joystick 1 to move the highlight bar to the desired file. The file size at the top will change to reflect the selected file.

Once a desired file is selected, it can be played by hitting either the Enter key or joystick 1's button 1 if it's color was gray or red. If it was any other color and you are not using the profile feature, a special bankswitching key might have to be pressed instead, depending on the type of game that it is:

If the game profile file has been generated	Enter / joystick 1 button 1
Standard 8k (not any other bankswitched scheme)	Enter / joystick 1 button 1
Standard 16k (not any other bankswitched scheme)	Enter / joystick 1 button 1
12k CBS RAM-Plus	Enter / joystick 1 button 1
16k or larger SuperChip	R
8k Parker Brothers	P
16k M-Network	M
Activision Decathlon or Robot Tank	A
8k or larger TigerVision	T
Pitfall II: Lost Caverns	2
32-in-1 game	3
Brazilian Megaboy game	G

Games that have the .BIN extension will use the NTSC screen palette by default; games with extension .PAL will default to the PAL screen palette.

A note about Pitfall II: Lost Caverns: this game, unlike all other Atari 2600 games, has a special chip that contains 2k of extra data in addition to the standard 8k. It also contains three random number generators and a three-channel square wave generator. To play Pitfall II on PCAE, the ROM file must be 10,495 bytes in size. The first 8k must be the standard 8k dump of the main ROM chip; the next 2048 bytes must consist of the 2k of extra data in the special chip, and the last 255 bytes must be read from one of the random number generators (the three random number generators each have a 255-byte cycle and produce the same values). Such a ROM file encompasses all the data that can be generated from the cartridge; PCAE is designed to automatically read whichever part is appropriate, depending on the addresses accessed.

The Configuration Popup Menu

In addition to the keys that are dedicated to playing games, there are several functions that affect PCAE's general configuration. These are generally related to the controllers that are to be emulated, and the sound and joystick configuration. They are accessible by pressing F10 from the main menu, which will pop-up the configuration menu. A description of the available functions is as follows:

Set frames per second	Limit the emulator to display a specific number of frames per second (e.g. 60 fps)
Build game profile	Automatically generate PCAE.PRO, the game profile file
Use standard controllers	Set the program to emulate Atari joystick controllers
Use paddle controllers	Set the program to emulate Atari paddle controllers
Use keyboard controllers	Set the program to emulate Atari Keyboard Controllers
Use Indy 500 controllers	Set the program to emulate Atari Indy 500 Driving Controllers (used with Indy 500)
Use Video Touch Pad	Set the program to emulate Atari Video Touch Pad (used with Star Raiders)
Change paddle assignments	Changes the paddle emulated by the mouse. The joysticks will always emulate the first two available paddles after the mouse's paddle has been selected.
Use keyboard	Disable PC joysticks and use the keyboard only
Calibrate and use joystick(s)	Enable 1 or 2 PC joysticks and calibrate them
Remap joystick buttons	Display a pop-up menu that allows the joystick buttons to be mapped to player 1 fire, player 2 fire, select, and reset functions
Toggle using real controllers	Instead of using PC joysticks or the keyboard, this will tell the emulator to expect real Atari controllers plugged into a specific home-built board plugged into the PC joystick port and a bidirectional parallel port
Set parallel port	Sets the port number of the parallel port for using real Atari controllers
Calibrate paddles	When using real Atari paddles plugged into the board, select this option to calibrate the paddles for the speed of your PC
Toggle debugger	Toggle using the integrated debugger, which will be invoked when a game is started
Toggle sound	Toggle sound emulation on or off
Set sound options	Adjust the sound buffer size and playback rate
Toggle extended VGA mode	Toggle extended VGA mode (not fully active when using the integrated debugger). PCAE.EXE can optionally display 204 scan lines and PCAE160.EXE can optionally display 215 scan lines.

The Atari 2600 display specification defines a standard viewable area of 160 pixels by 192 scan lines, which was expected to work on all television sets. However, there are a few games which utilize more than this, in fact more than 200 scan lines. For this reason, each executable has the ability of being switched into a non-standard VGA mode that supports extra scan lines. The standard executable, PCAE.EXE, can be switched into a non-standard 320x**204**x256 VGA mode, while PCAE160.EXE will switch from the 160x200x256 mode to 160x**215**x256. When this is done, all subsequent games will run in the extended VGA mode.

In addition to the above commands, the F1 key can be pressed to display summary screens of the command line syntax and menu commands.

The Game Profile File

The game profile file, PCAE.PRO, is a text file the emulator can automatically generate that contains the game type and controller information for each file in the game directory. It is intended to eliminate the need to memorize which bankswitching and controller keys are associated with which game, allowing all games to be played with the Enter key or joystick 1 button 1. It is supplemented by a user-modifiable known game profile file, PCAE.KNW, which contains filenames, game type codes, and controller codes for all games that are not automatically detected by the profile creation process. All Parker Brothers, M-Network, and TigerVision games cannot be automatically detected by the profile creation process, and must be referenced in the known game profile file. In addition, there may be other games which are not correctly identified and must be included in this file. The emulator has no way of autodetecting controller information for each game, and will thus give each game an "unknown type" controller code. This will result in games defaulting to joystick emulation. It is therefore necessary to add entries in the PCAE.KNW file for games the use paddles, Keyboard Controllers, etc.

Note: After changing the PCAE.KNW file, it is necessary to regenerate the profile file (PCAE.PRO) for the changes to take effect.

Each line of text in the known game profile file can be either a valid game reference, a remark, or a blank line. Valid game reference lines should contain the filename of a particular game, one or more spaces, a two-letter code describing the game type, one or more spaces, and a two-letter code describing the controller information. Remarks are not allowed on valid game reference lines. All remark lines should begin with a semicolon (;) as the first character (no leading spaces). The two-letter codes describing game types must be from the following list:

Standard 2k game	2K
Standard 4k game	4K
Standard Atari bankswitched 8k game	8K
Standard Atari bankswitched 16k game	16
Atari Super-chip game	SC
Parker Brothers 8k game	PB
M-Network 16k game	MN
Starpath Supercharger game	SP
CBS 12k RAM-Plus game	CB
TigerVision game	TV
Pitfall II: Lost Caverns	P2
Activision Robot Tank or Decathlon	AC
Unknown	??

Likewise, the two-letter controller codes are as follows:

Joystick	JY
Paddle	PD
Video Touch Pad	VT
Indy 500 Driving Controllers	DR
Keyboard Controllers	KB
Unknown	%%

The automatically-generated profile file, PCAE.PRO, follows the same format as the known game profile file, except that the emulator inserts a variable number of spaces between the filename and the two-letter game type code so that different codes will appear in different columns (for readability). Although the file can be modified, it will be automatically overwritten whenever the auto-generate option is selected from the popup menu; it is meant to be readable by the user and supplemented by the known file, PCAE.KNW, which overrides any settings in the automatically-generated profile file.

What is Bankswitching?

Before discussing bankswitching, it is first necessary to briefly explain a little bit about the Atari 2600 architecture. The VCS uses a processor designated as the 6507, which is a limited version of the more popular 6502. The 6507 processor can address up to 8k bytes of memory. In the 2600, some of this memory area is used for hardware support (e.g. video and audio), as well as for a small amount of RAM in the console. Although much of the address space is unused, there is generally 4k available for game programs. It was not long before companies like Atari and Activision realized that 4k of ROM was not enough to produce more complex games, so they had to devise a way to access more memory even though the CPU in the system could not. They accomplished this with bankswitching—placing higher-capacity ROM chips in their cartridges but only letting the CPU "see" up to 4k of them at a time. For example, a bankswitched 8k game might only let the CPU see either the first 4k or the last 4k, but never both. The switch would be accomplished by special hardware that would detect any CPU access to special "hot" memory addresses, which the manufacturer would designate to the game programmers as areas for switching the bank.

Several game manufacturers used the bankswitching approach for making larger, more complex games, but they tended to use different game sizes and "hot" bankswitching addresses—they had different bankswitching schemes. Some would switch the entire 4k area at a time, others switched smaller 1k areas, and still others included extra RAM in their games. Since each of their cartridges contained specialized hardware to accomplish the bankswitching, there was no need for 2600 users to worry about which scheme was being used by which manufacturer; they simply had to plug the game in and turn the machine on. An emulator, however, does not have any specialized hardware to automatically detect a bankswitching type and accomplish it, so there has to be some way to notify it whether bankswitching is involved and what kind it is.

Regarding Pitfall II: Lost Caverns, the bankswitching idea was taken to an entirely new level. Not only does the game contain 8k of game code using somewhat standard bankswitching, as well as some RAM, it also contains an additional 2k of data accessible by a very special chip. Data can be accessed in standard, bit-reversed, or reversed-nybble formats, and can be "bracketed" such that only data in a certain range is accessible. In addition, it contains a three-channel square wave generator with adjustable phase, frequency, pulse width, and mixing attributes. Just like standard bankswitching, it is necessary to tell the emulator somehow that this game is selected so that it knows to emulate these functions.

When the Game Is Running

Hopefully at this point you have selected a game and are seeing it on the screen. It is recommended that the first games you try are small 2k and 4k games with no bankswitching issues, just to make sure the emulator is running correctly instead of possibly running a bankswitched game with the wrong game setting.

Once the game is visible, PCAE is actually emulating the game code in the file. Since your PC doesn't have the six switches that are located on Atari 2600 consoles for playing a game, the emulator maps certain function keys to those switches. Just like on the actual console, you start by interacting with the game just as you would with a VCS unit—but with the function keys instead of switches. The function key assignments are as follows:

F1	Display a help/status popup window that shows the function keys available and lets you view and change the player difficulty settings, as well as enable or disable vertical autocentering
F2	Reset
F3	Toggle between NTSC and PAL screen palettes
F4	Select
F5	Color/Black-and-white toggle
F6	Player 1 difficulty toggle (A/B)
F7	Player 2 difficulty toggle (A/B)
Esc	Exit game (i.e. Power)
-	Shift screen downward
=	Shift screen upward
F10	Enable/disable vertical autocentering

In addition, Select and Reset are mapped to button 2 of each joystick; if you are using a four-button joystick (e.g. Gravis GamePad) these functions are accessible directly from the joystick as well as the keyboard. The one exception to this is for 12k CBS RAM-Plus games, since Omega Race requires extra buttons for the Booster-Grip joystick add-on that comes bundled with the game. In this case, a four-button joystick is required to play the game.

Vertical autocentering is a feature that attempts to optimally center the screen on your monitor. This is enabled by default, and works well for nearly every game. However, it might be desirable to disable this feature for certain games; in this case, the F10 key can be used to toggle it.

Using Joysticks With PCAE

PCAE will work with either none, one, or two standard PC joysticks. Either four-button or two-button joysticks will work, but it is recommended that one of the joysticks support four buttons to allow easy use of the Atari Select and Reset functions from the joystick instead of the keyboard. To enable joystick support, make sure they are first plugged into your game port. To connect two joysticks, it might be necessary to purchase a joystick “Y” cable to connect both joysticks to a standard joystick port. Some newer game port cards have two ports on them, such as the Gravis GameCard that sells for about $15.00 (US).

When the joystick(s) are connected, select “Calibrate and use joystick(s)” from the popup menu. You will be prompted with calibration questions for joystick 1. At this point it is generally a good idea to center any X/Y trimmer adjustments on the joystick. After you have followed the directions, the same questions will appear for joystick 2. If you have two joysticks connected, you can follow them to calibrate it, but if only one is connected, hit the <Esc> key to notify the emulator that only one joystick is connected. The emulator will turn off support for joystick 2 and the keyboard will be activated for player 2’s functions. In the event that recalibration is desired, the calibrate option can be used to recalibrate all joysticks. This is also necessary when unplugging the second joystick to switch control for player 2 back to the keyboard. If unplugging both joysticks, select “Use keyboard” from the popup menu to switch back to keyboard-only mode.

Joystick ports on Intel PC’s support four joystick buttons; they can be used in various ways, but the most common configurations are (1) one joystick using all four buttons (e.g. Gravis GamePad) or (2) two joysticks, each using two of the four buttons. Whatever the configuration, PCAE supports using two of the joystick buttons for player 1 and player 2 fire, respectively, and the other two buttons for the Atari 2600 “select” and “reset” functions. In addition, PCAE allows these functions to be mapped to different joystick buttons. Selecting Remap Joystick Buttons from the main menu will call up a popup menu displaying the current joystick button assignments. A function can be remapped to a different joystick button by selecting the appropriate function and then pressing the desired joystick button.

Using the Keyboard With PCAE

PCAE has the capability of mapping nearly all controller functions to the keyboard (the only thing it cannot emulate on the keyboard is the paddle, which is emulated via the mouse). When keyboard emulation becomes necessary, the keyboard mappings are as follows:

Joystick and Atari Indy 500 Controllers: (note that Indy 500 controllers have no up or down motion)

Joystick Player 1 fire	<Space> or keypad <Enter>
Joystick Player 1 up	Either up arrow key
Joystick Player 1 down	Either down arrow key
Joystick Player 1 left	Either left arrow key
Joystick Player 1 right	Either right arrow key
Joystick Player 2 fire	normal <Enter>
Joystick Player 2 up	I
Joystick Player 2 down	K
Joystick Player 2 left	J
Joystick Player 2 right	L

Atari Keyboard Controllers and Video Touch Pad: (note that the Video Touch Pad is the right keyboard controller only)

Left Controller			Right Controller		
1	2	3	4	5	6
Q	W	E	R	T	Y
A	S	D	F	G	H
Z	X	C	V	B	N

CBS Booster-Grip keys are emulated with the square bracket keys “[“ and “]” respectively.

Emulating Paddles With the Mouse and Joysticks

If you have a mouse connected and have a DOS mouse driver loaded, the emulator will automatically detect the mouse and use it for games that use a paddle. If you are running PCAE from a Windows DOS prompt, mouse support might already be provided, so it might not be necessary to load a DOS mouse driver. If you have analog joystick(s) connected and calibrated, their horizontal axes can be used to emulate paddles two and three, respectively. Note that joysticks used in this manner must be fully analog (i.e. Gravis GamePads and the like won’t work). The mouse and joystick assignments can be changed with the “Change paddle assignments” option from the main menu. There are four possible configurations (note how the mouse migrates from paddle 1 to paddle 4):

Paddle 1	**Mouse**	Paddle 1	Joystick 1
Paddle 2	Joystick 1	Paddle 2	**Mouse**
Paddle 3	Joystick 2	Paddle 3	Joystick 2
Paddle 4	N/A	Paddle 4	N/A
Paddle 1	Joystick 1	Paddle 1	Joystick 1
Paddle 2	Joystick 2	Paddle 2	Joystick 2
Paddle 3	**Mouse**	Paddle 3	N/A
Paddle 4	N/A	Paddle 4	**Mouse**

Using Real Atari Controllers With PCAE

PCAE supports connecting real Atari 2600 controllers to your PC if you have either built or acquired a special interface board. It connects to your PC's joystick port and a parallel port, which must support bidirectional transfer. Most motherboard parallel ports support this, and are usually configurable for standard, EPP (Enhanced Parallel Port), or ECP (Enhanced Capabilities Port) functionality. The joystick port should be a fully functional one, supporting two PC joysticks. If you have the board, you can use it to get the feel of playing an actual Atari 2600, as well as play games requiring more than one paddle.

To use real controllers, first select "Set parallel port" from the popup menu to choose the port number that matches your bidirectional parallel port. You might have to get this information from your BIOS setup, jumper settings, or Windows configuration, depending on your hardware and software setup. The emulator will display a list of all detected parallel ports and prompt you to choose one. Once this is done, select "Toggle using real controllers" from the popup menu. PC joystick and keyboard support will be turned off, and the board will be polled instead for controller information. The final step is to calibrate your paddle controllers. Connect a set of Atari paddle controllers to the player 1 port of the controller board, and select "Calibrate paddles" from the popup menu. You will be prompted to turn the paddles fully counterclockwise and hit a key. This tells the emulator the full extent of their travel as reported by your PC and will allow both sets of paddles to be connected and properly emulated by the program. It is necessary to calibrate the paddles only once, since the information is saved in the configuration file.

Please note that the current board design that comes with this documentation was tested with Atari joystick controllers, paddle controllers, keyboard controllers, Indy 500 driving controllers, and the CBS Booster-Grip. In all cases except the keyboard controllers, everything worked fine. With the keyboard controllers, however, only the first two columns of each controller worked. Perhaps someone else might design a better interface board that will make them work fully.

Slowing Down the Emulator For Faster PC's

At the time PCAE was written, many people were still using computers with i486 and slower Pentium processors. The assembly implementation was absolutely necessary to achieve a decent emulation speed. Since then, faster Pentium and Pentium Pro processors have become affordable, which presents the problem of games that run far faster on the emulator than on the original Atari 2600. To address this problem, the popup menu contains a feature that can be used to limit the emulator to display a certain number of frames per second. For example, standard NTSC Atari 2600's display at 60 fps, so selecting "Set frames per second" from the popup menu and entering "60" should limit the emulator to the standard NTSC Atari speed. In addition, this feature can be used to make certain games either easier or harder by slowing them down or allowing them to be played faster.

For users running the emulator from within an OS/2 DOS prompt, it may be necessary to select an option allowing DOS programs to access the hardware timer for PCAE to run properly. The frames-per-second slowdown feature uses the hardware timer to determine how much time has elapsed since the last emulated vertical refresh. Based on the number of frames per second it is told to display, it enters a waiting loop until a certain amount of time has elapsed.

In certain rare circumstances, it might be necessary to use different fps values either with certain system configurations or after an upgrade. This is very unusual, but there have been reports of erroneous timer information reaching the emulator somehow. In cases where 60 frames per second results in very slow operation, try half-multiples such as 90, 120, etc.

Configuring the Emulator's Sound Output

There are three features that can be used to configure PCAE's sound output. The first is merely a sound support toggle, which turns all sound on or off via the "Toggle sound" option from the popup menu. The next two features are accessed via the "Set sound options" option; this allows the sound buffer size and playback rate to be adjusted. The emulator simulates the sound output of the 2600 by calculating sound samples at maximum volume and storing them in XMS memory. When a sound is requested, it is loaded into a buffer, adjusted for volume, mixed with the sound output of the second channel (Atari 2600's have two-channel sound), and output. This takes a certain amount of time, and setting smaller buffer sizes will decrease the time it takes to perform this action—resulting in faster emulation. However, smaller buffer sizes also degrade the sound quality, so there may a tradeoff in adjusting this value for slower PC's.

The second parameter that is adjustable along with the buffer size is the sound card's playback rate. By default, the emulator sets the sound card to output the sound at a rate of 31400 Hz, which is the same rate that drives the original Atari 2600 and results in very good sound quality. This seems to work for most sound cards, but there are a few that have had problems getting sound to run. After testing this on a Pro Audio Spectrum 16 sound card, it was found that a slower playback rate would work much better on this card. You can select from four playback rates: 31400 Hz, 22050 Hz, 15700 Hz, and 11025 Hz. It is recommended that you use the highest rate that your sound card supports, since lower rates decrease sound resolution and thus sound quality.

PCAE160: The Other VCS

Included in the PCAE distribution is a second executable, PCAE160.EXE. This program functions nearly exactly as the standard emulator, but uses a non-standard 160x200x256 video mode instead of the usual 320x200x256 (mode 13h). This mode allows the emulator's display code to perform less work to display playfield graphics, resulting in a ten to fifteen percent increase in overall emulation speed. However, since this is a non-standard video mode, it is not compatible with all VGA cards. Specifically, some video cards may display none or only a portion of the screen in this mode, a result of differences in interpreting the End Horizontal Blanking value. Very fast machines (Pentium-100 and faster) should not need this program at all; it is intended for slower machines that need every boost they can get. If, after using PCAE.EXE you find that this one works as well, then you can feel free to use this one exclusively, but it is certainly not expected to be compatible with many (or even a majority) of VGA cards. I have successfully tested it with three ATI video cards, a local-bus Mach 32-based card with 2Mb of VRAM, a Mach 64-based WinTurbo 2MBU/4MB upgraded to 4Mb of VRAM, and the 3D Pro Turbo PC2TV with 8Mb of SGRAM.

4 Development Tools

In addition to the emulation engine, PCAE includes features designed to assist developers of Atari 2600 programs and other Atari 2600 emulators. It contains an integrated debugger that can be run in conjunction with any 2600 program by either using the “DEBUG” command line parameter or using the “Toggle debugger” option from the popup menu. It has a somewhat similar look and feel to Borland’s Turbo Debugger, but with only those features that are basic to program debugging in general and specific to the Atari 2600:

- Display of program code in a code window which can be navigated using the cursor keys
- Display of all 6507 register contents as well as flag bits
- Display of the contents of Atari RAM addresses 80h to FFh
- Displays the contents of the GRP0 and GRP1 player graphics registers
- Displays the current bank number for some bankswitching schemes
- Displays the scan line and horizontal position of the Atari CRT beam
- Displays the screen address that corresponds to the beam position
- Ability to execute unhindered, trace into, trace over, and execute until specific instructions or specific scan lines
- Online command reference and TIA register reference
- Ability to switch the debugger display to 320x400x256 video mode (should be compatible with all VGA cards) to display additional information
- Can display the current contents of the emulated video screen
- Ability to enter multiple bytes of data into the 6507’s addressable range
- Ability to change a 6507 register’s contents
- Ability to save the ROM range 1000h-1FFFh to disk

All debugger commands are invoked with certain keystrokes:

<Cursor keys>	Navigate the code window
D	Display toggle—toggle between standard 320x200x256 mode and 320x400x256 mode
E	Enter one or more bytes of data into the 6507’s address range
G	Go—execute the code unhindered until <Esc> is pressed
H	Run to here—execute until the instruction at the top of the code window
L	Run until a specific scan line
R	Change a 6507 register’s contents
S	Step over—execute until next instruction
T	Trace—execute current instruction only
Z	Save 4k data area 1000h to 1FFFh to disk
<Ins>	Move the highlight to the instruction pointed to by the Program Counter
<Space>	Display the contents of the emulated game screen
<Esc>	Exit the debugger and return to the main menu
<F1>	Display the help reference screens

When in 320x400x256 mode, the debugger also displays the contents of the TIA output registers located at memory locations 00h to 2Ch, as well as the contents of the TIA input registers located at memory locations 00h to 0Dh. In addition, the horizontal positions of the player 1 graphics, player 2 graphics, missile 1 graphics, missile 2 graphics, and ball graphics are displayed. If any of the graphics objects are enabled (if the GRPx registers contain nonzero values or if bit 1 the ENAxx registers is one), then the horizontal position for those objects are displayed in purple instead of black.

Debugger Layout

The largest window, in the upper left corner of the screen, displays the assembly code in the 2600 game program. A yellow highlight bar will be positioned on the next instruction to execute and the code window can be navigated using the cursor keys. To the right of the code window is an area that displays the contents of the 6507 P (flag) register, and right below that is an area that displays the contents of all the 6507 registers in hexadecimal format. Farther down, the beam scan line and horizontal position, its corresponding screen address, bank number, and GRP0 and GRP1 register contents are displayed.

At the bottom of the debugger screen is a data window that displays the contents of the 2600 RAM area, located at memory locations 80h to FFh. When in 320x400x256 mode, between the code window and the main data window are three smaller data windows that display the TIA write registers and read registers, as well as the horizontal positions of all five graphics objects (player 1, player 2, missile 1, missile 2, and ball).

Entering one or more bytes of data

Using the “E” key from the debugger brings up a pop-up window that allows you to enter a target address and one or more bytes of data. The format of the entry should be a hexadecimal address, and equal sign, and one or more hexadecimal bytes, separated by spaces. If a data entry is invalid, that address in the list is skipped. Note that all addresses are bitwise-ANDed with 1FFFh while all data bytes are ANDed with FFh. Also, the debugger does not have the hooks to the code that affects the TIA’s output, and therefore this method cannot be used to change TIA registers (it will only allow addresses in the range (80h to 1FFFh to be changed). Pressing <Esc> at any time aborts the data entry. Finally, be aware that for bankswitched games, all changes made are temporary—when the bank is switched, all changes made are lost.

Examples

1004=00 4 34 AE D3 2F	← Enters these six bytes into 1004h to 1009h
143F=56 – 4E 33	← Enters 56h, 4Eh, and 33h into 143Fh, 1441h, and 1442h

Stopping execution at a certain scan line

Choosing the “L” key brings up a pop-up window that asks for a scan line value. Entering one starts execution of the program, which will halt and return to the debugger when that scan line value is reached. If a number is entered that is never reached (e.g. 300), execution will continue until <Esc> is pressed. You can abort entering a number by hitting <Esc> at any time.

Changing a 6507 register’s contents

Hitting the “L” key opens a pop-up window that asks for a 6507 register and value to be entered. The format should be the register name, an equal sign, and a hexadecimal value to be entered. Hitting <Esc> aborts the process.

Examples

PC=13F3h	← Changes the program counter to 13F3h
A=3	← Changes the accumulator to 03h
P=E2	← Changes the flags register to E2h
X=26	← Changes the X register to 26h
S=9A	← Changes the stack register to 9Ah
Y=71	← Changes the Y register to 71h

Note that data entries are ANDed with FFh to keep the value in the legal byte range.

Saving to Disk

Since the “E” key allows you to interactively change a game file as it runs, the “Z” key allows you to permanently save the final ROM dump to disk. Hitting “Z” brings up a pop-up window that lets you enter a path and filename at which to save the file. Invalid filenames will be ignored, and hitting <Esc> lets you abort the process.

The Disassembler

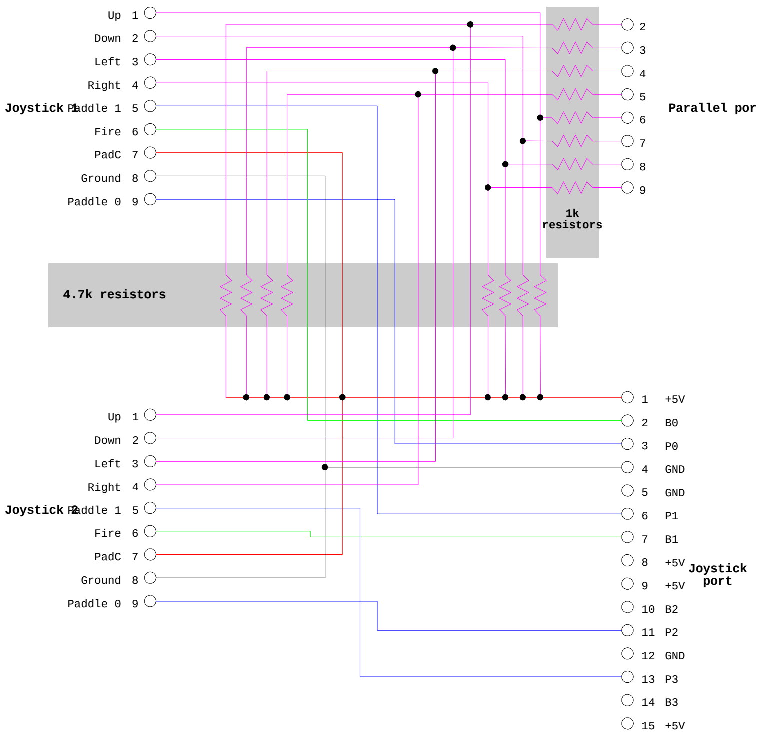
PCAE also includes a disassembler for building 6507 assembler listings of game files that are 4k in size or smaller, as well as many 8k, 16k, Super-Chip, and CBS RAM-Plus games. It is invoked with the “DISASM” command-line option, generating a source file called LISTING.ASM. It automatically detects the game code’s starting point and attempts to differentiate between code and data areas by recursively following all code paths. It is not capable of disassembling games that have other bankswitch types, nor can it follow code paths generated by jump tables. To disassemble bank-switched games, it is important to use the proper bankswitching command-line options to tell the emulator that they are in effect.

5 The Real Controller Board

PCAE supports connecting real Atari 2600 controllers to your PC via a special board that connects to a bidirectional parallel port and a fully functional joystick port. The board I built requires the following parts, though many variations on it can exist:

1 25-pin ribbon cable
1 34-pin IDC connector (connected to the 25-pin ribbon cable)
1 DB25 connector, connected to the 25-pin ribbon cable (plugs into the parallel port)
1 DB15 connector (plugs into the joystick port) I cannibalized an old joystick and cut off the connector.
1 15-pin ribbon cable and IDC connector to connect the DB15 connector to
2 DB9 connectors (Atari controllers plug into them). I used connectors that come with serial I/O cards that have DB9 connectors at one end and IDC connectors at the other end.
4 IDC headers (series of pins that the IDC connectors plug into) (one 26, one 16, and two 10-pin headers)
1 solder-type board
8 1k resistors (I used ½-watt resistors)
8 4.7k resistors (I used ¼-watt resistors)
some wire

Most of these parts can be bought at Radio Shack. The schematic is as follows:



Notes:
1. The 1k resistors and 4.7K resistors serve to protect the bidirectional parallel port from any signals that are being sent by the controllers if it is inadvertently set for write mode. They also serve to cause the default signal to be logic "1", which is interpreted by the Atari as an open connection.

2. Be VERY careful not to short the +5V and ground at the joystick port, and make sure not to connect the connectors backwards!

3. It is only necessary to connect to one of the +5V and ground connections on the joystick port, since the multiple copies are connected internally.

6 Troubleshooting

The computer hangs (or the emulator crashes) when the emulator is run

First, make sure that there is enough low DOS and XMS memory available for the program. To get more low DOS memory, you might have to change your system configuration, such as removing TSR's from either your AUTOEXEC.BAT or CONFIG.SYS files, or perhaps by loading them into upper memory blocks (UMB's). XMS memory must be provided by an XMS memory manager, such as HIMEM.SYS (provided with MS-DOS and Microsoft Windows). There are also third-party memory managers which might help, such as 386MAX and QEMM. You can usually find out how much memory is avaiilable by typing "MEM" from the DOS prompt. The amount of XMS memory required depends on the sound buffer size you have chosen from the main menu; for a 4k buffer size, 2Mb of XMS is required (a 2k buffer size requires 1Mb of XMS, etc.)

If you have met the memory requirements, the next thing to check is if your BLASTER environment string is set properly if you are using sound support. An incorrectly set environment string could cause lockups if sound is enabled in the emulator. You can test this by disabling sound support with the "Toggle sound" option and then attempting to play a game. If the emulator runs properly, this could be an indication of an incorrectly set BLASTER environment string. If this is the case, please see the earlier section on setting the emulator up for help or consult your sound card's documentation.

If both the memory and sound setups are correct but you are still experiencing lockups, check to see if you have a DOS mouse driver loaded if you are not running the emulator from a Windows or OS/2 MS-DOS prompt (i.e. you are in true MS-DOS mode). Not having a DOS mouse driver loaded really shouldn't cause a lockup, but it is recommended (and necessary to emulate paddle games). Common mouse drivers are usually called MOUSE.SYS or MOUSE.EXE.

If you are running the emulator from an OS/2 DOS prompt, be sure to enable access to the hardware timer in the DOS prompt setup. There has been a report from an OS/2 user who found out that the emulator would not run without this enabled.

The emulator runs VERY slowly

The first thing to check is your computer's speed. Benchmarks have shown that, using PCAE.EXE, a Pentium-90 can run most games at about 110% of a normal 2600's speed. A Pentium-60 will run at two-thirds of a P90, and a 486-66 will run at about 55% to 60% of a P60. There have been reports that a 486-120 can run games at about 100% of an Atari 2600. If your computer is slower than a 486-66, expect slow operation. You can find out how fast your PC is emulating the 2600 by running a game with a timer (such as a sports game) and timing how long it takes to count to a certain point. Remember, though, to set the frames-per-second value to 0 to allow the emulator to run as fast as possible before performing benchmarks.

If tests show that the games are not being emulated as fast as they should, the first thing to check is your joystick calibration setup. If you do not have joysticks connected, you should make sure to disable joystick support with the "Use keyboard" option. Otherwise, the emulator will constantly poll the joysticks, which can take a very long time if none are connected. Similarly, if only one joystick is connected, you might want to recalibrate the joystick, making sure to hit the Esc key when asked to calibrate the second joystick. This will tell the emulator that only one joystick is connected and prevent it from constantly polling joystick two. If the above procedures don't help, try changing the frames-per-second value. There have been occurrences where certain video cards incorrectly report vertical refresh information to the emulator, causing it to slow itself down more than it should. Trying higher values for frames-per-second might help.

Finally, if all else fails, check if your computer has a “turbo” feature, usually a button on the case. The effect of the turbo being off should be very noticeable, since ALL software should run slowly. Make sure the turbo is ON, usually indicated by an LED nearby.

The emulator runs more slowly when I use joysticks than when I use the keyboard

It is normal to experience a 10 percent or so degradation of emulation speed when using joysticks with the emulator. This is due to the method required to read analog joysticks. They can only be read by initializing the joystick port and waiting until a certain bit flips or until a time limit has been reached. This occasional waiting loop causes the loss in emulation speed.

The emulator runs, but there is no sound

Sometimes simple solutions are the best; make sure that sound is enabled with the “Toggle sound” option, and check to see that your speakers are connected and on if they have a power switch. If your speakers support optional external power supplies for additional amplification, you might have to set the power switch to “Off” if you are not using any additional power supply for them. Try testing sound with another program to make sure your sound card and speakers are set up properly.

If sound is enabled and your speakers are set up properly, the problem is most likely an incorrectly set BLASTER environment string. See the earlier section on setting the emulator up for help with this, or consult your sound card’s documentation. If there still isn’t sound, check to make sure your sound card’s IRQ and DMA are supported by the emulator.

If your sound card has mixer software or you are running in Windows 95, check your mixer settings to make sure they are correct. For DOS users, you should have mixer software that came with your card, or they might be available from your manufacturer. For example, my Sound Blaster 16 SCSI-2 has a program called SB16MIX.EXE, a DOS TSR that lets me set the individual mixer settings. Windows users should have similar software, and Windows 95 users can try double-clicking on the little speaker icon in the system tray at the bottom right of the screen. You should make sure that the digital and FM outputs are the same value (they are usually called “wave” and “midi”) and that the master volume is not zero or a small value.

Incompatible cards are extremely rare these days, but make sure your sound card is 100% Sound Blaster-compatible, preferably Sound Blaster Pro compatible (a simple AdLib FM card or Creative Labs Game Blaster will not do). In my opinion there is no substitute for a true-blue Creative Labs part, but they tend to command a premium.

I have sometimes had problems after running other software that would cause the sound card to become inaccessible afterwards. Turning off your computer and turning it back on will reset the sound card and might help matters.

I get no music in Pitfall II, the music is very quiet, or is too loud relative to the game sounds

The first thing to check is the mixer sound card settings. This is described in the prior troubleshooting section. You should make sure that the digital and FM settings are equal, and that the master volume is neither small nor zero. Also, you can try turning your computer off and on to reinitialize the sound card.

If you still don’t get sound, perhaps your sound card isn’t 100% Sound Blaster compatible. Check the documentation that came with your sound card to see if it supports FM synthesis. If it does, you can try other software to make sure that the FM is working. For example, Creative Labs cards usually come with a DOS test program, e.g. TESTSBP.EXE, TESTSB16.EXE, etc.

The emulator runs too fast; games are unplayable

From the popup menu, select the “Set frames per second” option. You will be asked to enter a number of frames per second at which the emulator will run. This causes the emulator to limit its speed so as not to exceed the frame rate that you specify. Entering zero will cause the emulator to run as fast as possible. For standard NTSC Atari 2600’s you should enter 60, which should limit the emulator to running at 60 frames per second.

If you had already done this and the emulator still runs too fast, your video card might not be correctly reporting vertical refresh rates to the emulator. Try smaller values, like 45 or 30.

The cursor is uncontrollable from the main menu; it keeps moving by itself

This probably indicates that either (1) you do not have joysticks connected but joystick support is enabled, or (2) you have joystick(s), but they either need to be recalibrated or aren’t connected. If there are no joysticks, disable joystick support with the “Use keyboard” option. The emulator will stop polling the joysticks for cursor movement, and will only use the keyboard. If you have joysticks, make sure they’re connected and recalibrate them with the “Calibrate and use joystick(s)” option (see the earlier section for the calibration instructons).

The menu doesn’t show any games to run

Make sure you are running PCAE from the right directory. If the game files are in a different directory than the emulator, you have to specify the alternate directory from the command line (e.g. PCAE G:\GAMES). Also, make sure that all games have extensions of either .BIN or .PAL, the only extensions the menu displays.

When I try to run the program I get a “Runtime error” message.

Make sure that the emulator is in the **current** directory, not in a different directory than the one you are in. For example, typing “C:\PCAE\PCAE” from a directory other than C:\PCAE will not work, since the emulator looks in the current directory for its support files. Also, make sure none of the support files included in the PCAE distribution are missing. If you are running the emulator from a Windows shortcut, make sure to specify PCAE’s starting directory in the shortcut setup. If you are trying to specify an alternate directory for games (e.g. PCAE G:\GAMES), make sure the alternate directory actually exists (i.e. make sure that G:\GAMES exists and is a directory, not just a file).

The emulator works on one computer, but one of the above problems arises when I copy it to another machine

Different machines have different hardware and software configurations. Make sure there is adequate memory, that the sound setup is correct, and either disable keyboard support in the emulator or recalibrate the joystick. Follow the troubleshooting help for the given problem to attempt to solve it.

When I try to run a certain game, I either get a black screen, garbage on the screen, or it exits immediately

There are three possibilities: (1) the game isn’t supported by the emulator; (2) the game is a different type than the one you have specified, or (3) the game is a bad ROM dump. If the game isn’t supported, there really is nothing you can do; perhaps a future version might support it. For bankswitched games, you can try other bankswitching types for that game size; for example, try Parker Brothers 8k instead of standard Atari 8k. If you have generated the game profile and are trying to run the game by pressing the Enter key, perhaps the game is a different type and needs to be added to the known game profile file. Also, if you have added a game to the known profile file or changed its entry, make sure you have regenerated the standard profile afterwards, since changes to PCAE.KNW will not take effect until PCAE.PRO has been regenerated. If you can establish the proper game type, you can edit the known game file, add the game to the list, and regenerate the profile file from the popup menu. If nothing works, perhaps it is a bad copy of the game.

When PCAE160 is run, all or part of the screen is blank when a game is run

PCAE160 reprograms the VGA registers to attempt to achieve a 160x200x256 screen mode that lets the emulator do less work to display playfield graphics. This results in a ten to fifteen percent speed increase, but, being a non-standard video mode, is not compatible with all VGA cards. If the above symptoms occur, PCAE160 might not be compatible with your VGA card. It has been tested with several cards manufactured by ATI Technologies with good results, but this is no guarantee of compatibility with all brands. At any rate, for machines of equal or faster speed than a Pentium-100, this program should be unnecessary.

I'm having trouble calibrating my joystick(s)

First, make sure they are connected properly, and test them with some other piece of software to ensure that the joysticks and joystick port are working properly. Then, you should examine your joysticks for what are usually known as X/Y trimmer adjustments. These usually take the form of knobs or sliders, and are used for fine adjustment. Before calibrating the joystick, you should make sure the trimmer adjustments are centered (not at either extreme) to allow the fullest range of values to be reported by the joystick.

If centering the trimmers and recalibrating doesn't help, and you have a speed-adjustable joystick port, you might want to examine the speed setting of the port. If you have a driver that sets the speed when you boot up, sometimes simply rebooting or powering down and back up helps, since this will reset the joystick port to your speed setting. It might be necessary to change the speed setting as a last resort, but this usually isn't necessary.

One of the joystick buttons doesn't work, or hitting a joystick button causes more than one thing to happen

You might have a problem with your joystick. Make sure it is fully plugged in, and test it with some other software. If it works fine with other software, check the joystick button mappings by selecting "Remap Joystick Buttons" from the main menu to make sure that you don't have more than one thing mapped to the same joystick button. If the joystick has problems with other software as well, try remapping the buttons to compensate. In this case, you might consider buying either a new joystick or a new game port. You might be able to tell where the problem is by trying a different joystick to see if the problem persists.

I'm trying to use real controllers, but it doesn't work or doesn't work fully

There could be several problems. First, make sure real controllers are enabled by selecting "Toggle using real controllers" from the popup menu. Second, you should make sure you have selected the right parallel port for the emulator; it must be one that supports bidirectional transfer. You can try a different port by selecting "Set parallel port" from the popup menu. Also, make sure the board is properly plugged in and that Atari controllers are properly plugged into the board. If you built the board yourself, make sure it is done properly.

If none of the above steps works, it is possible that your parallel port is not set up properly. Make sure it is enabled by either checking your jumpers or entering your BIOS setup. Also, if your port supports standard, EPP, or ECP modes, try setting it to a different mode. You can also try running the emulator and board on a different PC to see if the parallel port is the problem. Also, make sure the joystick port is a fully functional one that supports two joysticks. You can test this by plugging two joysticks into it using a joystick "Y" cable and making sure that both joysticks work. You also should consider checking to see if your parallel port's port number or IRQ conflicts with other hardware, since some computers will have their sound cards and parallel ports share IRQ settings.

If you're having trouble with keyboard controllers or the Video Touch Pad, please note that the board design that comes with this documentation does not fully support keyboard controllers. In the tests that were run, only the first two columns of each controller worked. Perhaps someone more knowledgeable will be able to design a board that will make them work fully with the emulator.

When I run PCAE, I get an error like “EMM386: Unsupported DMA mode. Press Enter to reboot”

There seems to sometimes be a problem when running PCAE with EMM386, if your sound card is compatible with Sound Blaster DSP version 2.00 or higher (the vast majority of sound cards). The problem is that EMM386 doesn’t always like the auto-initialize DMA mode that the emulator uses to achieve near-perfect sound. The best suggestion is to either run the emulator from within a Windows 3.x or Windows 95 prompt, or remark out the EMM386 line from your CONFIG.SYS file.

7 Acknowledgements

Over the past few years while I have been developing PCAE, I have been lucky enough to receive a lot of help from some gracious people. This should in no way be considered an exhaustive list; I've gotten so much feedback over the past two years from so many people that I could never list everyone—my email folder simply isn't that large, so if you aren't included here, please don't take any offense. I am grateful for all the assistance that has come my way.

Those who helped (not necessarily in any real order)

Matt Conte: First and foremost, for giving PCAE a home! Also for spreading the word, doing lots of beta testing, getting me info and tools I needed, fielding a lot of the email, and generally being really helpful.

Kevin Horton: Helped me with a lot of technical details of the 2600 especially bankswitching issues and Pitfall II deciphering. Also made sure I knew about every kind of bankswiching game he could get his hands on.

Norbert Juffa: Wrote the millisecond timer used in the fps limiting code.

Dan Melton: Wrote the keyboard interrupt service routine.

Ron Fries: Wrote the TIA sound routines and sent them to me.

Dan Boris: Introduced me to Ron Fries' sound routines.

Dave W. (Dave's Video Game Classics): Did lots of testing of PCAE, and supplied a whole bunch of good suggestions to make it better.

Jim Leonard: Provided lots more web space for PCAE, as well as did some very important testing of the final version. Also converted the documentation and schematics to HTML.

Bradford W. Mott and Keith Wilkins: For making available the source code to Stella, another really great 2600 emulator. This has not been an isolated project, and I'd be lying if I said that Stella wasn't helpful. I certainly hope that PCAE's source has been equally helpful.

A lot of others: Many of the features PCAE has today came from people who emailed me with their suggestions. This document is a direct result of all the feedback I've gotten, and is my attempt at making things clearer for everyone.

8 Disclaimer

For the following discussion, this software, the PC Atari Emulator, will be referred to as “PCAE”.

The author of PCAE hereby grants unlimited license for users of PCAE to freely copy and install it for personal use. PCAE is supplied as-is; the author WILL NOT be held responsible for ANY damages that may occur as a direct or indirect result of, or inability to use, PCAE. The author will not be held responsible for damages resulting from the loss of business or productivity due to the use of PCAE. Furthermore, the author will not be held liable for any damages that may be incurred as a result of using non-standard video modes or hardware built for use with PCAE. The author also cannot be held liable regarding the merchantability (or lack thereof) of PCAE. The author grants no warranty whatsoever, and none should be implied.

USE IT AT YOUR OWN RISK.

As stated previously, PCAE may be copied and distributed freely for personal use. It MAY NOT be sold by anyone without prior written consent by the author. It may be distributed as part of a shareware/freeware package as long as not more than $7 US (1996 dollars) is charged for its distribution. PCAE MAY NOT be distributed without this document in its complete form. Any other use of PCAE requires explicit written agreement from the author.

PCAE emulates a commercial game system for which copyrighted software was developed and still exists. The author of PCAE hereby forbids anyone to distribute PCAE in conjunction with any other copyrighted software. In addition, users of PCAE are specifically forbidden from using it in any way with copyrighted software for which they are not in legal ownership. PCAE should IN NO WAY be regarded as condonation of, or an excuse to commit, software piracy, and the author will not be held responsible for the actions of others.