

# Stella

**A multi-platform Atari 2600 VCS emulator**

Release 1.1

## **User's Guide**

February 1999  
Bradford W. Mott  
bwmott@acm.org

# A Brief History of the Atari 2600

In the early 1970's, video arcade games gained commercial success for the first time. The American public was introduced to Pong, Tank, and other interactive video games which populated amusement parks, bars, and arcades. The games were successful enough to create interest for home versions, so in 1975 Atari released Home Pong and it was a smash hit. Other companies such as Magnavox and Coleco followed suit and released their own dedicated console games. Then in 1976, Fairchild Camera and Instrument introduced the Channel F system, the first cartridge based home video game system. The industry recognized that cartridge systems were the future of video gaming, and began development in that direction. In January 1977, RCA released the Studio II, another cartridge based system, although it only projected in black and white and seemed to be focused on educational titles. Then, in October 1977, Atari released the Atari VCS (Video Computer System) with an initial offering of nine games. This system, later renamed the Atari 2600, took the industry by storm and dominated the marketplace for years to come.



Because of oversupply, the Christmas season of 1977 was very rough on the video game industry, and the Atari 2600 was the only system that managed to emerge unscathed. Atari enjoyed strong sales in 1978 and a fantastic holiday season, as Atari released more games such as Outlaw, Spacewar, and Breakout. Internally however, Atari was at odds. Nolan Bushnell, the inventor of pong and founder of Atari,



wound up leaving the company and purchased Pizza Time Theater, which later became the successful Chuck E. Cheese! In 1979 Atari continued their trend and released 12 more games which met with continued success. However, Atari was now facing some stiffer competition from the Mattel Intellivision and the Magnavox Odyssey2.

Atari needed a mega-hit in 1980 in order to squash the competition, and they found it in the home version of a game from Japan called Space Invaders. It was so popular that people were buying the Atari 2600 just so they could play Space Invaders at home. Following that, Atari released Adventure, which was the first video game to contain an Easter Egg - placing an object in a certain area revealed the programmer's name, Warren Robinett.



1980 was important for another reason - the creation of the first ever third party software producer, Activision. The company was formed by four Atari employees who were unsatisfied with the working conditions at the company. They released four games initially: Dragster, Fishing Derby, Checkers and Boxing. The games were very well received by the public, and revealed that the Atari 2600 was capable of better games than Atari themselves had been producing. Atari tried to prevent Activision from selling games, but they failed and Activision grossed \$70 million that year.

By 1981, the video game industry was basically a horse race between the 2600 and the Intellivision. While the Intellivision was technologically superior in some respects, the 2600 continued to lead in sales. Atari released the home version of Asteroids, which was a huge success. Inspired by the success

of Activision, another software development group called Imagic was formed. They would not release any games until 1982 however. Another company, Games by Apollo, was formed in Texas and released several games that year.

Coleco entered the market in 1982 with the release of the graphically superior Colecovision. To combat this new system, Atari produced the 5200, a technologically comparable system. The 2600 dropped \$100 in price in order to remain competitive. Then a company called Arcadia released a peripheral called the Supercharger which played games in an audio cassette medium. This allowed for multiple loads and expanded the 2600's capabilities. Atari released Pac-Man and



E.T. that year, two incredibly hyped games which were critical flops. Although Pac-Man sold many copies, it was considered to be a poor translation of the arcade hit. However, there were many fantastic games produced for the 2600 during this period, and it was still selling strong.



Ever since the inception of Activision, Atari had been fighting to keep third parties from producing cartridges which they felt were stealing profits from them. Finally the issue was settled when Atari agreed to allow third party manufacturing in exchange for a royalty. Suddenly software companies began popping up all over, and 1982 saw releases from companies like Venturevision, Spectravision, Telesys, CBS, 20th Century Fox, US Games, M Network, Tigervision, Data Age, Imagic and Coleco. There was even a company that released a line of X-Rated games for the 2600 called Mystique. The year was financially successful for Atari, however there seemed to be a glut of software. Although there were many quality titles still produced, there was an increasing number of rushed games as manufacturers attempted to cash in on the craze.

More companies jumped on the band wagon in 1983. Zimag, Ultravision, Amiga, and others were also producing games and peripherals. It seemed as if there was just too much product to meet the demand, and as it turned out there was. By the end of the year, companies began folding. US Games, Data Age, Games by Apollo, Telesys and others all closed their doors from poor sales. A video game crash was occurring, and all companies were taking it on the chin.

1984 was a much more subdued year for the Atari 2600, and the price of the system had now dropped to \$40-\$50. Many were saying that the video game industry was dead. However, Atari surprised everyone by announcing the release of the 7800, and also promising more 2600 games with improved graphics and sound. Unfortunately, neither of these things happened in 1984 because Atari sold their home video game division to Jack Tramiel who believed that home computers would replace video game systems. No further mention of the 2600 or 7800 was made that year, and it appeared that they might be dead.

1985 was another very quiet year for Atari and video games in general, and only a few games were released for the 2600. Activision produced Cosmic Commuter and Ghostbusters, but with little fanfare or marketing, these games did not sell well. However, because of the huge game library and cheap price, Atari still sold over a million 2600 consoles in 1985.

There were very few plans for home video game systems by any company in 1986, since the market appeared to be dead. Then, to most people's surprise, Nintendo brought the NES to America and it was a

smash hit, proving that video games still had a place in the US. Atari decided that maybe it would be a good idea to release the 7800 units it had in storage, and produce some more 2600 games. The 7800 was released with only 3 games initially available, although it was compatible with the 2600 library. They also redesigned the 2600 as the 2600 Jr., a machine with the same abilities, but a new look and marketing campaign. It was sold for less than \$50.

Video games were once again selling phenomenally in 1987. Atari released several new titles, including Jr. Pac-Man, and also licensed a number of games from other companies such as Donkey Kong and Q\*Bert. These new titles sold for \$10-\$15. Interestingly, a number of titles began appearing again from third part companies such as Epyx, Froggo, and Exus. It seemed that the 2600 was not dead yet! In 1988, Atari rehired Nolan Bushnell and announced a number of new titles, including Secret Quest, a game written by



Mr. Bushnell himself. Atari continued to manufacture these games even until 1989. However, it was apparent that the 2600, after its introduction over a decade ago, was finally at the end of its run. Although it was still produced and marketed outside of the US, the Atari 2600 finished its run in America. No other console has had such a long history or sold as many systems in the U.S.



Today, the 2600 still has a large number of fans who remember the countless games played over the years, and the years to come. There are even games being produced by hobbyists, some of them quite professionally, being released on newly burnt cartridges with labels and manuals. And the recent trend in retrogaming has brought many more video game fans to rediscover the 2600, and it continues to live on 22 years after its release!

Alexander Bilstein  
February 1999

Stella is a freely distributed multi-platform Atari 2600 VCS emulator; originally developed for Linux by Bradford W. Mott. Stella allows you to enjoy all of your favorite 2600 games once again by emulating the 2600's hardware with software. Stella is written in C++, which allows it to be ported to other operating systems and architectures. Since its original release Stella has been ported to AcronOS, AmigaOS, DOS, FreeBSD, IRIX, Linux, MacOS, OpenStep, OS/2, Unix, and Windows.

## New in Release 1.1

- DOS and Linux versions support real Atari 2600 paddles using a special PC game port adaptor
- Linux version requires the new 1.2.x Linux Joystick API
- Unix versions support the "`-display`" command line option for specifying display to use
- Unix versions support the "`-owncmap`" command line option to install a private colormap
- Greatly improved Supercharger single-load and multi-load game support
- Improved TIA emulation to support the RESPx multi-sprite trick

## Features

- High speed emulation using optimized C++ code
- Includes several "homebrew" games for the Atari 2600
- Supports high quality sound emulation using Ron Fries' TIA Sound Emulation library
- Emulates the Atari 2600 Joystick Controllers using your computer's keyboard or joysticks
- Emulates the Atari 2600 Keyboard Controllers using your computer's keyboard
- Emulates one Atari 2600 Paddle Controller using your computer's mouse
- Emulates four Atari 2600 Paddle Controllers using a special PC game port adaptor
- Emulates the Atari 2600 Driving Controllers using your computer's keyboard or joysticks
- Emulates the CBS Booster-Grip Controller using your computer's keyboard or joysticks
- Supports cartridges using Atari's standard 2K and 4K format
- Supports cartridges using Atari's 8K, 16K, and 32K bank-switching schemes
- Supports cartridges using Activision's 8K bank-switching scheme (Robot Tank and Decathlon)
- Supports cartridges using CBS Electronics' 12K bank-switching scheme
- Supports cartridges using Chris Wilkson's Megacart 128K bank-switching scheme
- Supports cartridges using M-Network's 16K bank-switching scheme
- Supports cartridges using Parker Brothers' 8K bank-switching scheme
- Supports cartridges using Tigervision's bank-switching scheme with up to 512K ROM
- Supports Supercharger single-load and multi-load games
- Supports property file for setting the properties associated with games
- Built-in properties contains settings for over 540 games
- Supports user specified frame rate
- Supports the NTSC and PAL television palettes
- Supports several "undocumented features" of the TIA graphics chip used by some games

- TIA emulation supports full collision checking
- DOS version of Stella works with most emulator game shells such as "Game Menu" by Jim Pragit
- DOS version of Stella supports the standard 320x200 VGA graphics mode as well as the extended 320x240 VGA graphics mode

# 2

# What You Will Need

---

The following sections outline the basic system requirements for running Stella under various operating systems.

## DOS

The DOS version of Stella is designed to work on an IBM-PC or 100% compatible system with the following:

- DOS 5.0 or higher
- 8 MB of RAM
- 256 color VGA graphics card
- 486/120 MHz required; 586/166 MHz highly recommended
- At least 2 megabytes of hard disk space available
- Sound Blaster or 100% compatible sound card required for sound
- Joysticks or gamepads are highly recommended
- Mouse and mouse driver required for paddle emulation using the mouse
- Special adaptor required for using real Atari 2600 paddles
- Emulator game shell highly recommended

## Linux

The Linux version of Stella is designed to work on a Linux Workstation with the following:

- Linux Kernel 2.0.x
- X Window System
- GNU C++ compiler version 2.7.2 and the make utility are required for compiling the Stella source code; GNU C++ compiler version 2.8.1 or later is highly recommended
- 16 MB of RAM required; 32 MB highly recommended
- 8 bit color graphics card required; 16 bit color graphics card highly recommended
- 486/120 MHz required; 586/166 MHz highly recommended
- Open Sound System with a supported sound card required for sound
- Joystick device driver version 1.2.x required for joystick support
- Joysticks or gamepads are highly recommended
- Mouse required for paddle emulation using the mouse
- Special adaptor required for using real Atari 2600 paddles

# Unix

The Unix version of Stella is designed to work on a Unix Workstation with the following:

- Unix operating system
- X Window System
- GNU C++ compiler version 2.7.2 and the make utility are required; GNU C++ compiler version 2.8.1 is highly recommended
- 32 MB of RAM; 64 MB highly recommended
- 8 bit color display required; true color display highly recommended
- High performance CPU
- Open Sound System required for sound

Once you have a Stella distribution you should follow the instructions for your operating system given below. If your operating system isn't listed then see the installation instructions included with the distribution for your system.

## DOS

The DOS version of Stella is distributed as a ZIP archive containing the DOS executable as well as some other files. You should install it as follows:

1. Change directories to the root directory:

```
CD C:\
```

2. Extract the Stella distribution:

```
unzip strelease.zip
```

3. Add the following to your AUTOEXEC.BAT file:

```
SET PATH=%PATH%;C:\STELLA
```

4. If you have a Sound Blaster or 100% compatible sound card make sure the BLASTER environment variable is set in your AUTOEXEC.BAT file. For example:

```
SET BLASTER=A220 I7 D1
```

5. If you have a mouse make sure your mouse driver is loaded in either the CONFIG.SYS file or the AUTOEXEC.BAT file

6. Reboot your system

For the second step you'll need a program for extracting ZIP archive files. You can use PKUNZIP for DOS, the Info-Zip UnZip tool for DOS, or a number of other programs available for extracting ZIP files.

## Linux

The Linux versions of Stella are distributed as a compressed tar file containing the Linux executables as well as some other files. If these pre-built executables do not seem to work on your system then download the Unix source code distribution and build your own executables. You should install the distribution as follows:

1. Extract files from the distribution:

```
tar -zxvf stella-release-linux-arch.tar.gz
```

2. Login as root and change directories to the distribution directory

3. Copy the executables to a system directory:

```
cp xstella /usr/local/bin
```

```
cp stella-sound /usr/local/bin (if it's included)
```

If you have a joystick driver installed, such as `joystick-1.2.13.tar.gz`, you can play games using joysticks. For additional information about the Linux joystick driver see the following web site: <http://atrey.karlin.mff.cuni.cz/~vojtech/joystick/>.

## Unix

The Unix version of Stella is distributed as a compressed tar file containing the C++ source code as well as some other files. The source code can be compiled under most Unix operating systems as well as DOS. The Unix code was developed with the GNU C++ compiler version 2.8.1, however, it should compile with other C++ compilers. The DOS port was developed with DJGPP using GNU C++ version 2.8.1. You should install the distribution as follows:

1. Extract files from the distribution:

```
tar -zxvf stella-release-src.tar.gz
```

2. Change directories to the `stella-release/src/build` directory
3. Edit the makefile to meet your needs
4. Type `make` and follow the on screen instructions to build the `xstella` executable
5. Install `xstella` in a directory that's in your path (e.g. `/usr/local/bin`)
6. Change directories to the `stella-release/src/ui/sound` directory
7. Type `make` and follow the on screen instructions to build the `stella-sound` executable
8. Install `stella-sound` in a directory that's in your path (e.g. `/usr/local/bin`)

Currently, sound is supported using the Open Sound System. Sound has been tested and is known to work under Linux and BSDI. For additional information on OSS see the 4Front Technologies web site at:

```
http://www.4front-tech.com/
```

If you're a programmer and know how sound works on your favorite version of Unix then please take the time to port `stella-sound` to your system.

Stella allows you to play games using ROM images of cartridges and cassettes.

## Cartridges

Most games for the Atari 2600 came on cartridges. A cartridge usually consists of a single Read Only Memory (ROM) chip which contains the data and code for the game. Plugging a cartridge into the Atari 2600 allows the 2600's microprocessor to access the program stored on the cartridge.

In a similar way you must "plug" a copy of a cartridge into Stella when you want to play it. Having a ROM image, BIN file, of the cartridge allows you to do this. A ROM image is a file, which contains the actual data and code read from the cartridge. There are several ways to obtain a ROM image of a cartridge:

1. If you're handy with a soldering iron then you can design and build a device that plugs into the printer port of a PC and read the data from the cartridge
2. You can purchase the Atari 2600 Action Packs by Activision and use their ROM images
3. You can search around the internet and find ROM images to download

**WARNING:** It is illegal to use ROM images of games that you do not actually own since these games are still copyrighted.

## Supercharger Cassettes

Supercharger games were not stored on cartridges instead they were stored on cassette tapes. The Supercharger, which plugged into the Atari 2600's cartridge slot, loaded games into its 6K of Random Access Memory (RAM) using a standard audio cassette player. The Supercharger also supported multi-loading, which allowed games to be broken into several segments and loaded at different times. This was useful for large games which had distinct parts such as role playing games.

Most of the available Supercharger ROM images are stored in 8448 bytes files. However, ROM images of multi-load games are sometimes stored in a set of 8448 byte files. The names of these files have a two character sequence number in them which indicates what load they are. The sequence starts with zero, skips a few numbers and then increments by one.

Stella supports multi-load games, however, the set of ROM images must be combined into a single ROM image file. For example to create a multi-load ROM image file for Survival Island you would do the following under Unix:

```
% cat survivl0.bin survivl6.bin survivl7.bin > survivl.bin
```

Once you have the multi-load ROM image file, `survivl.bin` in this case, you can play the game using it.

# 5

# Starting A Game

Once Stella is installed and you have some ROM images you're ready to start playing. To play a game follow the directions for your operating system.

## DOS

The DOS version of Stella uses command line arguments to specify the game you'd like to play as well as other options. To run Stella use a command line of the following format:

```
stella [options ...] filename.bin
```

### Options

- fps <number> Attempt to display *n* frames per second instead of the default 60 frames per second (if the computer isn't fast enough then it may be unable to display the desired number of frames per second)
- modex Indicates that the 320x240 graphics mode should be used instead of the default 320x200 graphics mode
- paddle <0 | 1 | 2 | 3 | real> Indicates which paddle the mouse should emulate or that real Atari 2600 paddles are being used (defaults to paddle 0)

### Emulator Game Shells

Instead of using the command line to run Stella you may find it easier to use an emulator game shell. An emulator game shell allows you to pick games from a menu without having to type everything at the command line. One such game shell is Jim Pragit's "Game Menu" which can be found at the following web site:

```
http://members.aol.com/jpsoftco/gamemenu.htm
```

There are other game shells available, however, you'll have to search for them.

## Linux and Unix

The Unix version of Stella uses command line arguments to specify the game you'd like to play as well as other options. To run Stella use a command line of the following format:

```
xstella [options ...] filename.bin
```

### Options

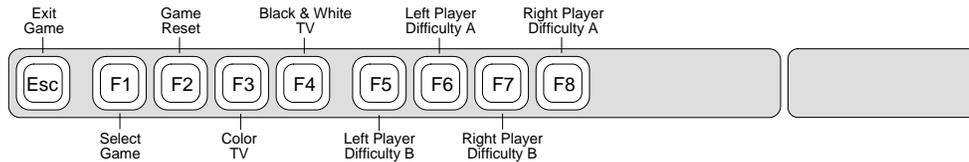
- display <display> Connect to the designated X display
- fps <number> Attempt to display *n* frames per second instead of the default 60 frames per second (if the computer isn't fast enough then it may be unable to display the desired number of frames per second)
- owncmap Install a private colormap
- paddle <0 | 1 | 2 | 3 | real> Indicates which paddle the mouse should emulate or that real Atari 2600 paddles are being used (defaults to paddle 0)

# 6

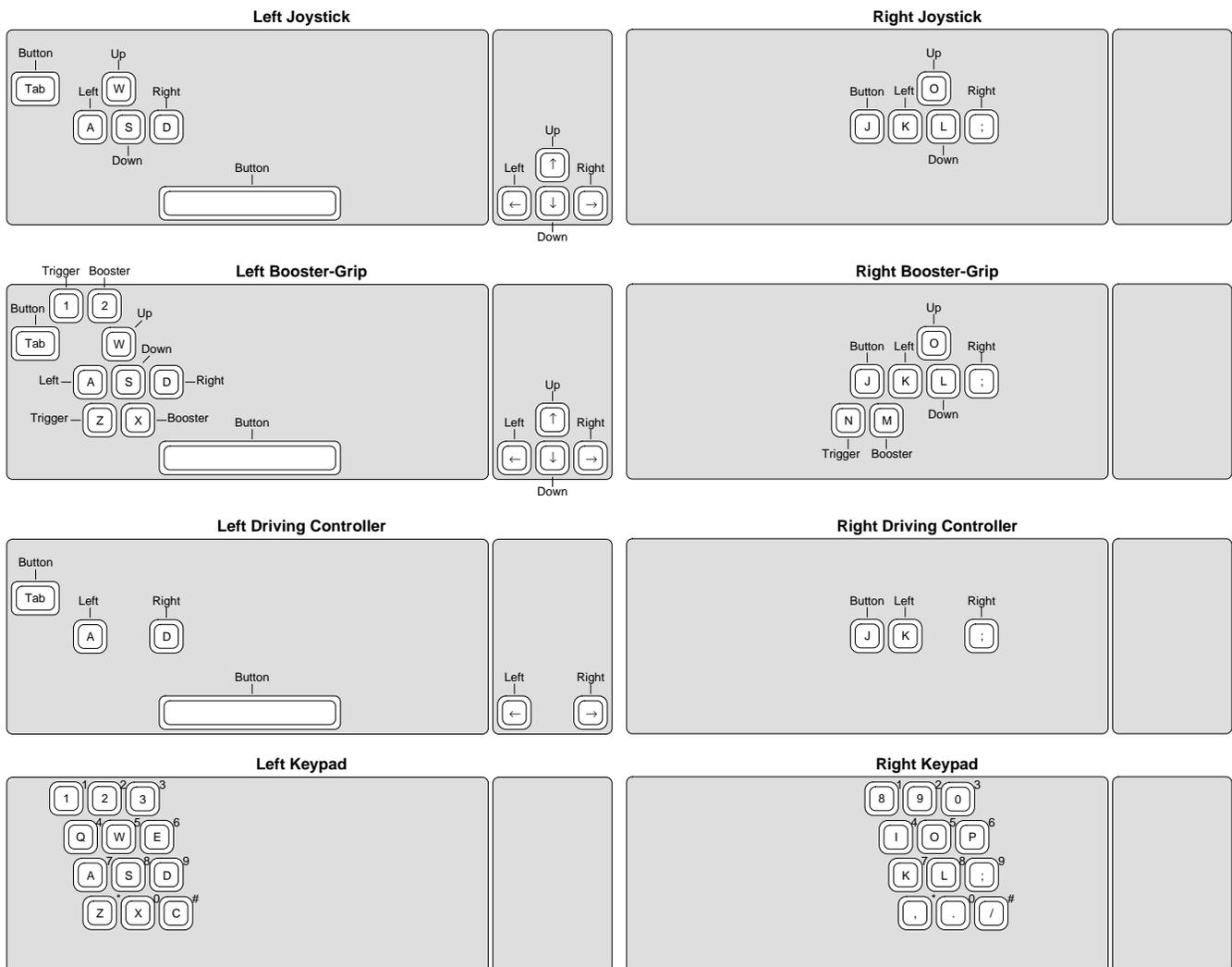
# Keyboard Layout

The Atari 2600 console controls and controllers are mapped to the computer's keyboard as shown in the following figures:

## Console Controls



## Controllers



## Other

The Unix version of Stella allows you to resize the display window using the '=' key.

Stella uses game properties to specify the "best" emulator settings for a game. Stella's built-in properties contain specific settings for over 540 games. If for some reason you need to specify your own properties for a game you can create a property file. The best way to do this is to download the latest `stella.pro` file from the Stella web site, <http://stella.atari.org>, and add your new properties to it.

## DOS

The DOS version of Stella looks for a property file `stella.pro` in the current working directory. If this file is found then the game properties are read from it; otherwise the built-in properties are used.

## Linux and Unix

The Linux and Unix versions of Stella looks for the property file `.stella.pro` in your home directory. If this file is found then the game properties are read from it; otherwise the built-in properties are used.

## Property File

A property file consists of some number of blocks. Each block in the file contains the properties for a single game. For example the general format of a property file is:

```

; Comments
"Cartridge.Name"      "Value"
"Property"            "Value"
" "

; Comments
"Cartridge.Name"      "Value"
"Property"            "Value"
" "

...

; Comments
"Cartridge.Name"      "Value"
"Property"            "Value"
" "
```

Every block in the property file must have a unique value for the `Cartridge.Name` property.

# Properties

Each block in a property file consists of a set of properties for a single game. Stella supports the properties described below.

**Cartridge.FileName:** This property indicates the name of the file containing the ROM image. In general this property is not needed, however, if present Stella uses it while attempting to match a game with its block of properties. If the value of the property matches the filename of the ROM image then Stella uses that block of properties for the game.

**Cartridge.MD5:** This property indicates the MD5 checksum of the ROM image as a string of hexadecimal digits. Stella uses this property while attempting to match a game with its block of properties. If the value of the property matches the MD5 checksum of the ROM image then Stella uses that block of properties for the game. You can use the GNU `md5sum` program, which is included with most Linux distributions, to calculate the MD5 checksum of a ROM image.

**Cartridge.Manufacturer:** This property indicates the game's manufacturer. Currently this property is not being used, however, future versions of Stella will probably allow the user to view this information.

**Cartridge.ModelNo:** This property indicates the manufacturer's model number for the game. Currently this property is not being used, however, future versions of Stella will probably allow the user to view this information.

**Cartridge.Name:** This property indicates the actual name of the game. It is very important that this property's value be unique from all other cartridge names in the property file.

**Cartridge.Note:** This property contains any special notes about playing the game. Currently this property is not being used, however, future versions of Stella will probably allow the user to view this information.

**Cartridge.Type:** This property indicates the bank-switching type for the game. The value of this property must be: Auto-detect, 2K, 3F, 4K, AR, E0, E7, F4SC, F6, F6SC, F8, F8SC, FASC, or FE. For more information about bank-switching see Kevin Horton's 2600 bank-switching document.

**Console.LeftDifficulty:** This property indicates the default difficulty setting for the left player. The value of this property must be A or B.

**Console.RightDifficulty:** This property indicates the default difficulty setting for the right player. The value of this property must be A or B.

**Console.TelevisionType:** This property indicates the default television setting for the game. The value of this property must be Color or BlankAndWhite.

**Controller.Left:** This property indicates what type of controller the left player uses. The value of this property must be Booster-Grip, Driving, Keyboard, Paddles, or Joystick.

**Controller.Right:** This property indicates what type of controller the right player uses. The value of this property must be Booster-Grip, Driving, Keyboard, Paddles, or Joystick.

**Display.Format:** This property indicates the television format the game was designed for. The value of this property must be NTSC or PAL.

**Display.XStart:** This property indicates the horizontal location to start displaying pixels at on a scan-line. The value of this property must be  $n$  such that  $0 \leq n \leq 80$  and  $n$  is divisible by 4.

**Display.Width:** This property indicates the number of pixels to display per scan-line. The value of this property must be  $n$  such that  $80 \leq n \leq 160$  and  $n$  is divisible by 4.

**Display.YStart:** This property indicates the scan-line to start displaying at. The value of this property must be  $n$  such that  $0 \leq n \leq 64$ .

**Display.Height:** This property indicates the number of scan-lines to display. The value of this property must be  $n$  such that  $100 \leq n \leq 256$ .

**Emulation.CPU:** This property indicates the CPU emulation quality. The value of this property must be High or Low.

**Emulation.HmoveBlanks:** This property indicates whether the TIA HMOVE blank bug should be emulated or not. The value of this property must be Yes or No.

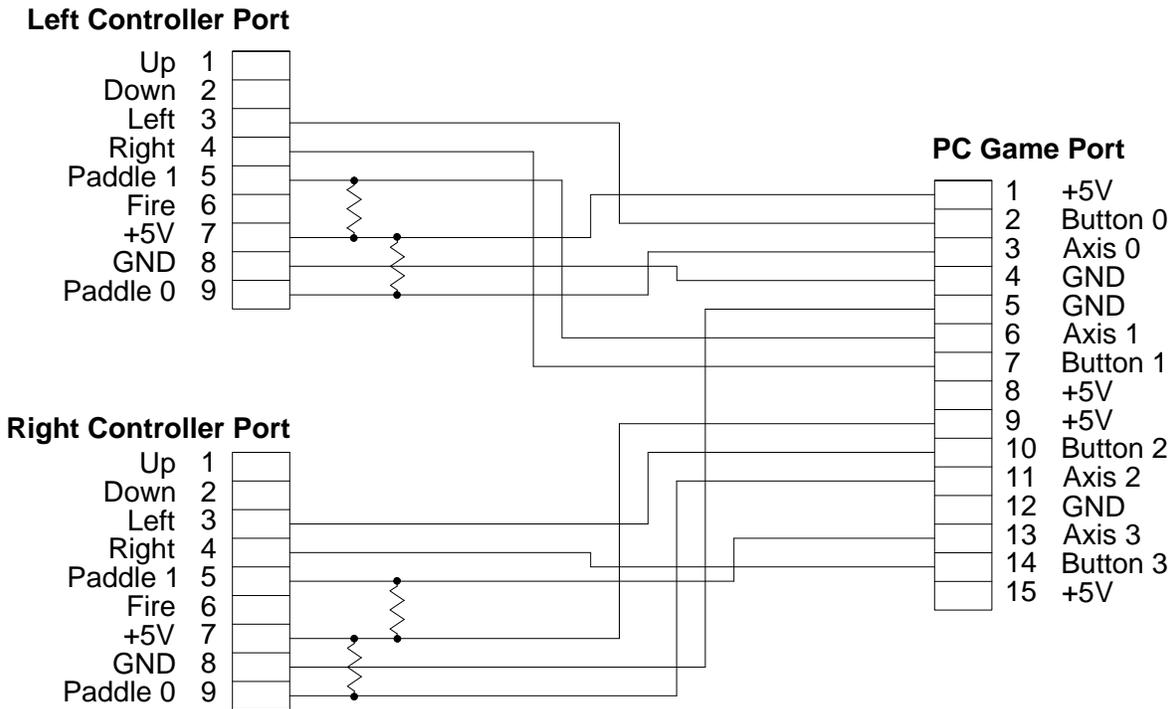
# 8

# Real Paddle Adaptor

The DOS and Linux versions of Stella support real Atari 2600 paddles using a special adaptor which connects to the PC game port. The adaptor requires the following parts:

- 1 DB15 male connector (joystick extension cable plugs into this)
- 2 DB9 male connectors (paddles plug into these)
- 4 115k resistors
- 1 Small project box
- 1 Joystick extension cable
- Some wire or ribbon cable

Most of these parts can be found at Radio Shack. I could not find a DB15 connector there or the 115k resistors, however, if you're willing to sacrifice a joystick extension cable you can do without the DB15 connector and you can always place a 100k resistor and 15k resistor in series to make a 115k resistor. A schematic of the adaptor is shown below:



As you build the adaptor be careful not to short the +5V and GND connections. Although I have built this adaptor and use it on my personal PC I cannot guarantee that the circuit is error free therefore USE IT AT YOUR OWN RISK!

I started developing Stella during the fall of 1995 and since then I've received help from a number of people around the world. Some people have provided technical help while others have offered suggestions and praise. I am forever grateful for all the help and support I have received over the last few years. The following is an incomplete list of the people who have played a part in bringing Stella to you:

David Asbell	Ported release 0.7 of Stella to Java
Christopher Bennett	Helped discover a Supercharger emulation bug in release 1.0 of Stella
Alexander Bilstein	Provided "A Brief History of the Atari 2600" for this manual
Dan Boris	Provided technical information on the Supercharger
Piero Cavina	Allowed "Oystron" to be included in the Stella distribution
Bob Colbert	Allowed "Okie Dokie" to be included in the Stella distribution
Joe D'Andrea	Author of the IRIX port of Stella
Renato Ferreira	Provided code to support private colormaps under Unix
Ron Fries	Author of the awesome TIA Sound library
Aaron Giles	Author of the Power Macintosh version of Stella
Mark Hahn	Allowed "Elk Attack" to be included in the Stella distribution and provided help with the TIA HMOVE blank bug
Kevin Horton	Author of the definitive Atari 2600 bank-switching document
Erik Kovach	Author of the property file for release 0.7, 1.0, and 1.1 of Stella
Daniel Marks	Provided improved keyboard joystick support for Stella release 0.1
David McEwen	Author of the Acorn version of Stella
Jeff Miller	Author of the Windows version of Stella
Dan Mowczan	Provided a Supercharger to help with Stella's development
Jack Nutting	Author of the OpenStep version of Stella
Jim Pragit	Author of the "Game Menu" emulator game shell
John Saeger	Participated in a helpful discussion on understanding the TIA's RESPx multi-sprite trick
Chris Salomon	Provided information and code to help implement Supercharger support
Jason Scott	Organizer of the property file archive for early versions of Stella; now he's helping with the web site
David Shaw	Provided make targets for the BSDI platform
Raul Silva	Helped with design and graphics for an early version of the Stella web site
Chris Snell	Maintained a mirror of the Stella FTP site
Darrell Spice Jr.	Author of the OS/2 version of Stella
Eckhard Stolberg	Provided a description of the TIA bug that produces the star field effect in Cosmic Ark, provided the PAL television palette, and participated in a helpful discussion on understanding the TIA's RESPx multi-sprite trick

Matthew Stroup	Author of the Amiga version of Stella
Joel Sutton	Author of the FreeBSD version of Stella
Greg Troutman	Allowed "This Planet Sucks" to be included in the Stella distribution
Keith Wilkins	Maintained the DOS version of Stella until release 0.7
Jeff Wisnia	Provided technical data sheet for the 6532 RIOT chip
Countless others	Many of the features present in Stella are a result of input from people who have sent me suggestions

This software is Copyright © 1995-1999 by Bradford W. Mott. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

The author hereby grants permission to use, copy, and distribute this software and its documentation for non-commercial purposes, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. If you distribute this software, the entire contents of this distribution must be distributed. The software may be modified for your own purposes, but modified versions may NOT be distributed without prior consent from the author.

If you would like to do something with this software that the license prohibits (such as distributing it with a commercial product, using portions of the source in some other program, etc.), please contact the author.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.