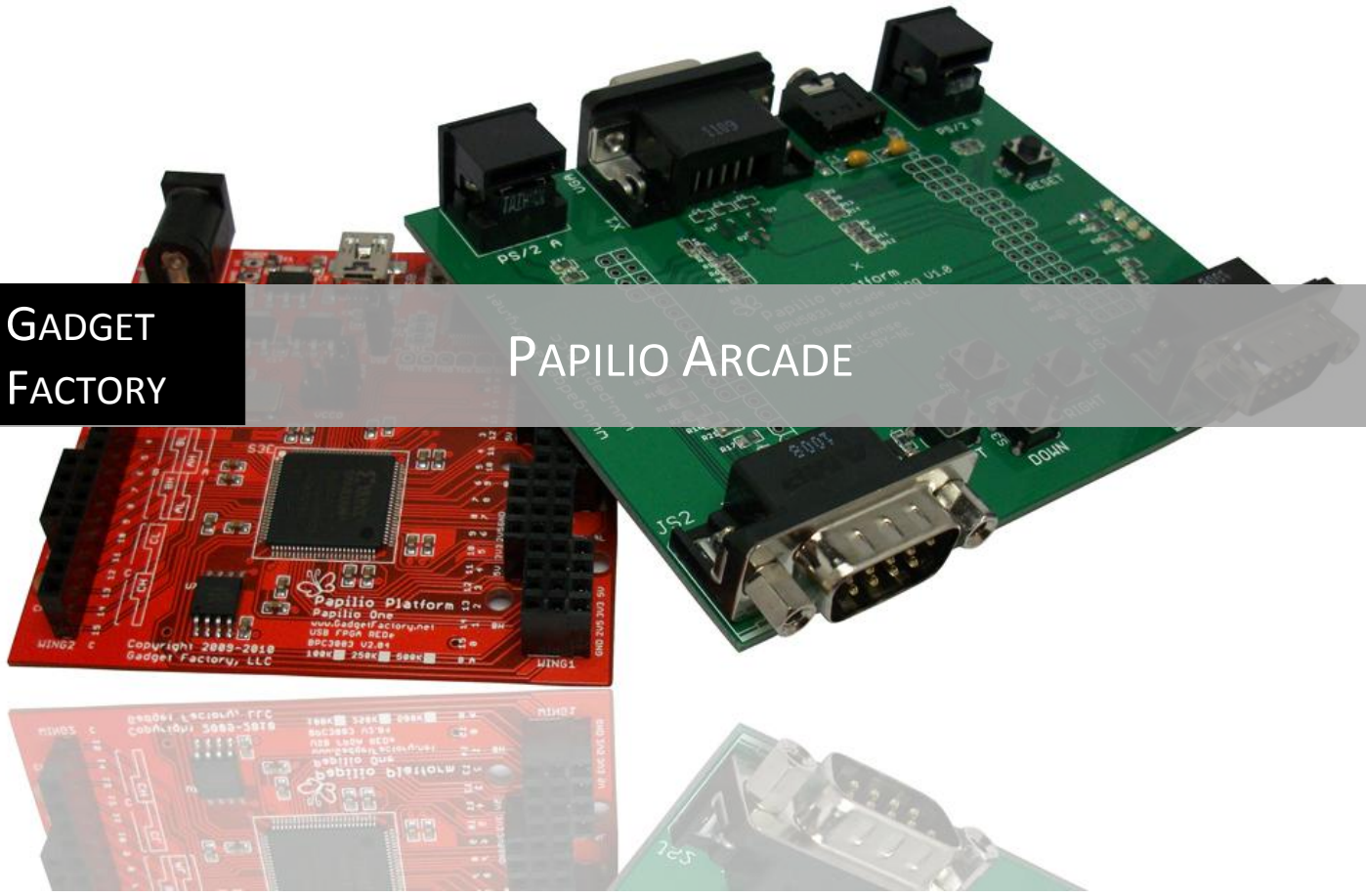
Version 1.2 - November 23, 2011

GADGET FACTORY

PAPILIO ARCADE



Written by | Jack Gassett and the Gadget Factory Team

Volume

1

GADGET FACTORY

Hackable Hardware > Hack | Ware

Papilio Arcade User Guide

GADGET FACTORY

Papilio Arcade User Guide

Boulder, Colorado USA
www.GadgetFactory.net

Table of Contents

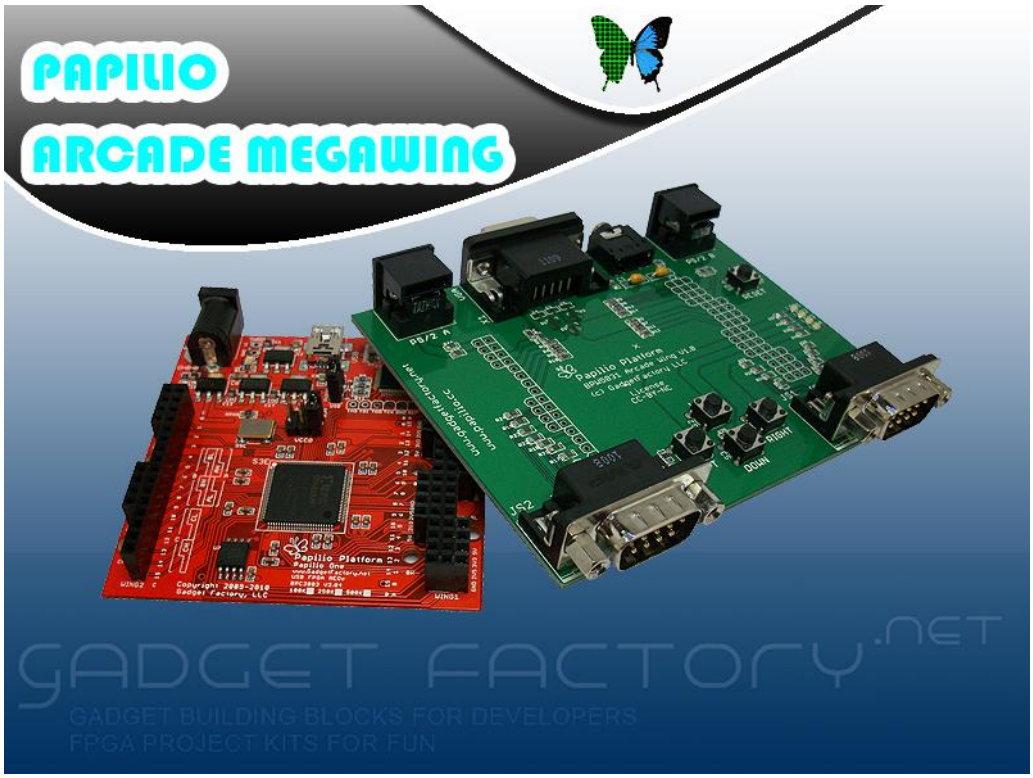
Chapter 1

Introduction and Overview

Remember your first time playing Pac-Man?

Quick, do you remember the first time you played Pac-Man? Chances are that you can still remember where you were and how it made your heart pound. There's something special about classic Arcade games like Pac-Man. The bright yellow cabinet was like a beacon for kids and adults alike.

The amazing thing is that Pac-Man and other classic arcade games are just as compelling today as they were back then. The difference is that we now have the ability to shrink the entire Pac-Man motherboard onto a single FPGA chip! In the process of doing so we can learn how all the pieces of an 8 bit game system fit together and at the end we get the joy of revisiting classic games such as Pac-Man. Papilio Arcade is a great way to start hacking with VHDL, FPGA, and classic 8 bit processor design.



Overview

Welcome to the Papilio Arcade - the ultimate hardware for arcade game hacking. Whether you're a classic game enthusiast or developer, an arcade cabinet hobbyist, a demoscener, or hardware hacker, the Papilio Arcade is rife with possibilities for you to explore. Hack away, and learn through your own experimentation and study using the motherboards from arcade classics like Pac-Man, Frogger, Galaxian, Space Invaders, and more – all in one place, on one board – the Papilio Arcade. Only from Gadget Factory Hack|Ware: Hackable Hardware.

Game Enthusiasts

The Papilio Arcade is based on an FPGA (Field Programmable Gate Array) which allows the user to tell the hardware what to *be* – not just what to *do*. In addition to the Papilio Arcade's ability to *become* an actual motherboard for classic games like Pac-Man and others, since it is FPGA-based it provides endless potential for other classic arcade and homebrew games to be added! There are several games (the same ROMs you use with MAME) that run now and we anticipate many more games to come. The Papilio Arcade App makes it easy to load games onto the hardware. The current games supported are:

- Pac-Man Hardware
  - Pac-Man
  - Gorkans
  - Crush Roller
  - Lizard Wizard
  - Open Source Pong
- Frogger Hardware
  - Frogger
  - Amidar
  - Scramble
  - The End
- Galaxian Hardware
  - Galaxian
- Space Invaders Hardware
  - Sea Earth Invasion
  - Alien Invasion

Arcade Cabinet Hobbyists

The Papilio Arcade hardware does not have the overhead associated with running MAME on a computer. We designed the Papilio Arcade to eliminate the fuss of having your PC hooked up to, or being housed inside of, your arcade cabinet. Ours is a hardware solution – meaning no worries about keymapping, OS problems, or waiting during boot-up. A game is loaded to the Papilio Arcade hardware, and when power is applied it starts up just like the original game motherboard did at the arcades. Just plug in your cabinet and you're ready to play.

Demoscene

For demoscene enthusiasts the FPGA provides the ultimate in flexibility that allows you to create unique demos that have never been seen before. Full source code lets you hack together hardware that has never existed before, use the graphics from Pac-man and the audio from a C64 SID chip.

Homebrew/ Indie Game Developers

The Papilio Arcade is hackable arcade hardware, meaning that what you do with it is up to you. The possibilities are virtually limitless for homebrew and independent game development.. Since the Papilio Arcade can *be* any of the four different hardware motherboards (or any combination of them) the sky really is the limit for the homebrew and indie crowds to develop original games using throwback hardware. We are currently working on a toolchain and "hello world" examples for the different hardware types.

Creating new and unique games is possible with ZPUino - a capable soft processor with everything you need to start writing games using the Arduino IDE. The ZPUino has an integrated VGA controller, YM2149 sound chip, VGA libraries, and code examples to get you started.

Hardware Hackers

Hardware hackers will appreciate that all of the game hardware definitions are completely open source. Full VHDL code is available and can be studied to learn more about classic Arcade hardware. Changes can be hacked into the hardware without any modification to physical motherboards. One single hardware board can become different classic motherboards.

Chapter 2

Hardware

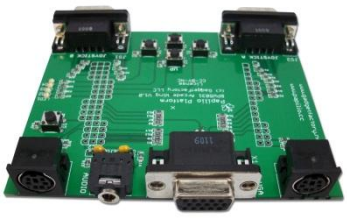
The Papilio Arcade Kit is comprised of the Papilio One 500K core and an application specific Arcade Wing. The Arcade Wing is what provides all of the connectivity to the outside world such as video, audio, and joystick.

There are two options available for the Arcade Wings, an Arcade MegaWing or individual Wings. The Arcade MegaWing provides everything needed in a single unified board and is a more convenient package. Individual Wings can be used to make a Papilio Arcade solution that is more flexible but not as convenient.

Papilio One FPGA Board

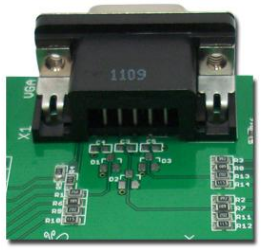
The Papilio One 500K FPGA board is a powerful and flexible platform that provides the core for exciting projects such as the Papilio Arcade. It acts as the empty canvas that classic arcade motherboards can be recreated on. Its flexible nature means that it can be re-used with future Papilio projects by simply purchasing the MegaWing portion of a project.

Arcade MegaWing



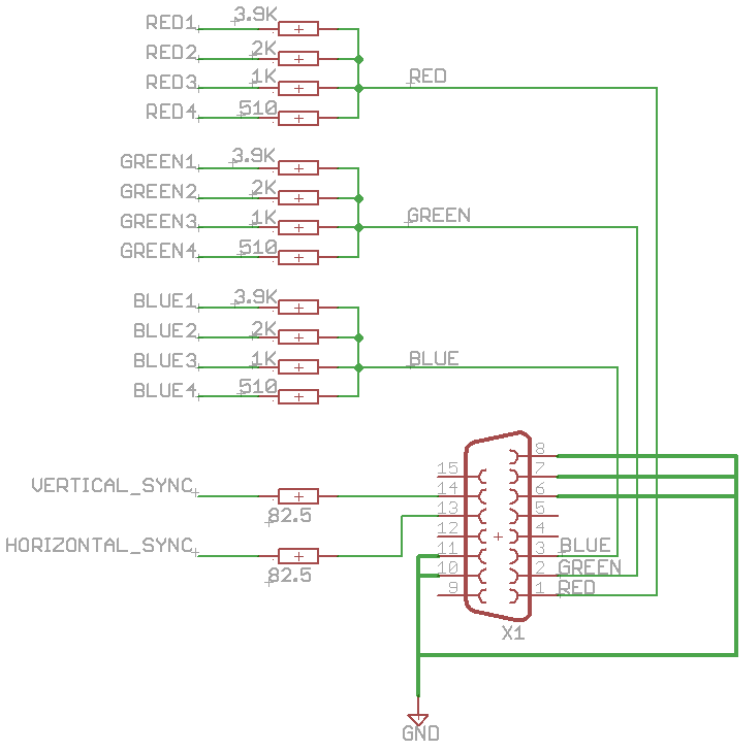
The Arcade MegaWing provides all of the arcade hardware in one convenient and easy to connect circuit board. It snaps into the Papilio One and gives it the necessary hardware resources to communicate with the outside world. Continue reading for more information about each hardware section of the Arcade MegaWing.

MegaWing VGA


VGA Video - 4096 colors

The VGA section of the Arcade MegaWing uses 12 resistors to implement 4K color depth. VGA video is analog in nature so there needs to be some way to vary the RGB (Red, Green, and Blue) signals between 0V and .7V. For each RGB signal the shade, or intensity, of the color is controlled by varying the voltage of the pin between 0 and .7V. The finer control you have over the voltage the more colors you can create. For the Arcade Wing we are able to control 4 different bits per color which allows us to generate 16 different voltage levels between 0 and .7V. This means we can generate 16 shades for each color. If we add all three colors together we have 12 bit video which gives us (2^12=4096) the possibility of 4096 colors.

The usual options for controlling voltage on a pin are to use a DAC, PWM, Delta-Sigma, or a resistor ladder. For the Arcade Wing we wanted to use the lowest cost and easiest method for a hobbyist. PWM and Delta-Sigma were ruled out because we did not think we would be able to vary the voltage with these methods fast enough to keep up with the VGA timing requirements. A DAC would be a good solution but was more than we needed for our modest requirements. A resistor ladder fit the bill perfectly because it is very low cost and easy to implement. The resistor ladder is made up of 4 resistors per RGB color that all connect, in common, to the VGA connector on one side and individually to a digital I/O pin on the other side. The lowest resistor starts at ~500 ohms and each successive resistor doubles in size with the final resistor ending with ~4K ohms. Each digital I/O pin is controlled by the VGA controller inside the FPGA. Each pin can be set to either 0 or 1 with a 1 causing the resistor to contribute its voltage to the final voltage level. The voltage can be stepped up from 0 to the desired output voltage which in this case is .7V. The way the desired output voltage is accomplished is by carefully managing the parallel voltage that the 4 resistors add up to. Using a Parallel Resistance Calculator we see that 500 ohm, 1000 ohm, 2000 ohm, and 4000 ohm give a total resistance of 266 ohm. The final key bit of information is that there is a 75 ohm resistance built into VGA cables, so that means that the point where our 4 resistors come together on our VGA connector forms a Voltage divider circuit. Using a Voltage Divider calculator with 3.3V as the Input Voltage, 266 ohm as R1, and 75 ohm as R2 we end up with an Output Voltage of .73V. The end result is that if all 4 resistors are set to '1' then we will see .73V at the VGA connector. If all 4 resistors are set to '0' then we will see 0V at the VGA connector, any combinations in between will give us evenly stepped voltages between 0 and .73V. The voltages will be evenly stepped because we picked resistors that double in size.

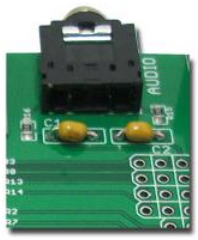


Name	Direction	Function	Papilio One Pin
Red 1	Output	Red Bit 1	P98
Red 2	Output	Red Bit 2	P2
Red 3	Output	Red Bit 3	P3
Red 4	Output	Red Bit 4 (*Note1)	P4
Green 1	Output	Green Bit 1	P68
Green 2	Output	Green Bit 2	P66
Green 3	Output	Green Bit 3	P63
Green 4	Output	Green Bit 4 (*Note1)	P61
Blue 1	Output	Blue Bit 1	P85
Blue 2	Output	Blue Bit 2	P83
Blue 3	Output	Blue Bit 3	P78
Blue 4	Output	Blue Bit 4 (*Note1)	P71
HSync	Output	Horizontal Sync	P95
VSync	Output	Vertical Sync	P94

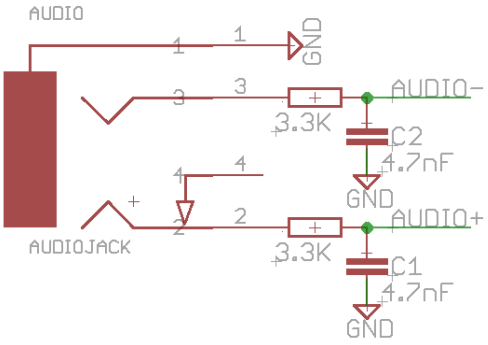
Note 1: Use these bits for 8 color video.

MegaWing Sound

The sound section implements a 1/8th inch stereo jack with a low pass filter. Arcade projects simply implement a Delta-Sigma DAC as outlined in Xilinx App Note 154. A Delta-Sigma DAC allows high quality audio to be implemented with a minimum amount of physical hardware required. The high speed of the FPGA clock allows the FPGA to do the heavy lifting of the Digital to Analog conversion.

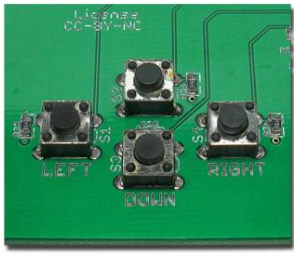


Stereo Audio
Delta Sigma DAC



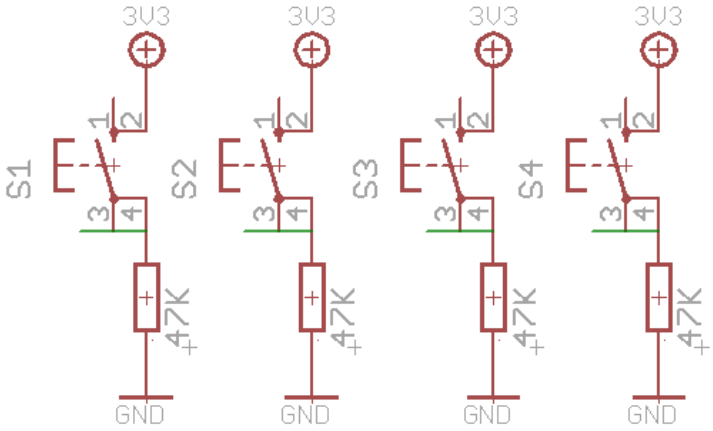
Name	Direction	Function	Papilio One Pin
Audio -	Output	Audio Left	P84
Audio +	Output	Audio Right	P86

MegaWing Directional Buttons



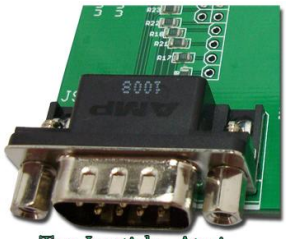
4 Directional Buttons

The Arcade MegaWing provides 4 directional buttons that can be used for general input. They can be used with homebrew games that do not need joysticks or for the coin/start buttons on classic games.



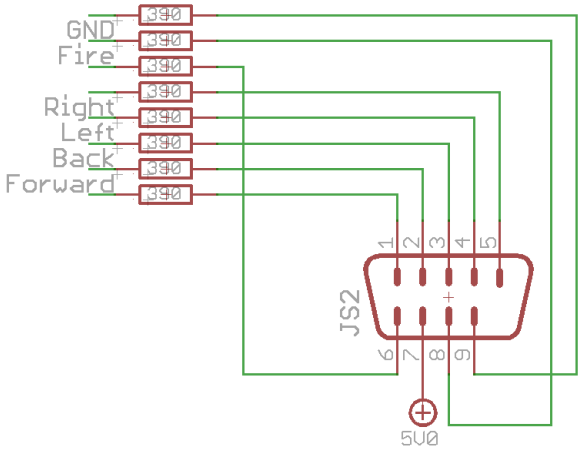
Name	Direction	Function	Alt Function	Papilio One Pin
Left	Input	Left	Coin 1	P58
Up	Input	Up	Start 1	P54
Right	Input	Right	Coin 2	P36
Down	Input	Down	Start 2	P41

MegaWing Joystick



Two Joysticks -Atari, Commodore, Classic

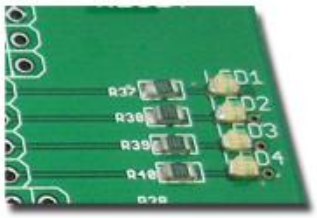
The Papilio Arcade Wing supports Atari 2600, Commodore 64, classic Arcade joystick, and any joystick that uses digital inputs. The board comes with a Male DB9 connector that allows an Atari 2600 joystick to be plugged in and used without any custom wiring. For a custom Arcade style joystick it is necessary to wire the joystick to a female DB9 connector using the diagram below. Each pin of the DB9 connector is protected by a current limiting resistor to provide compatibility with 5V powered joysticks.



Pin	Name	Direction	Function	P1 JA	P1 JB
1	Up	Input	Up Bit 1	P5	P34
2	Down	Input	Down Bit 1	P10	P25
3	Left	Input	Left Bit 1	P11	P22
4	Right	Input	Right Bit 1	P15	P23
5	Fire 2	Input	Fire Bit 2	P17	P33
6	Fire 1	Input	Fire Bit 1	P9	P32
7	5V	N/A	5V	N/A	N/A
8	GND	Input	GND	P12	P18

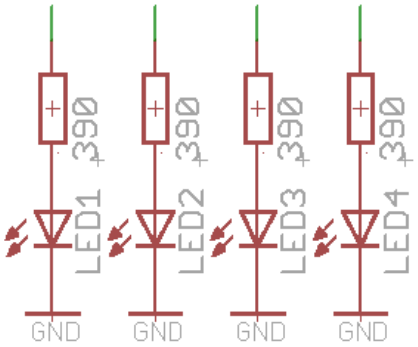
P1 JA is the FPGA pins for Joystick A when connected to a Papilio One.
P1 JB is the FPGA pins for Joystick B when connected to a Papilio One.

MegaWing LED's



4LED'S

The Arcade MegaWing provides 4 LED's for visual feedback of game status.

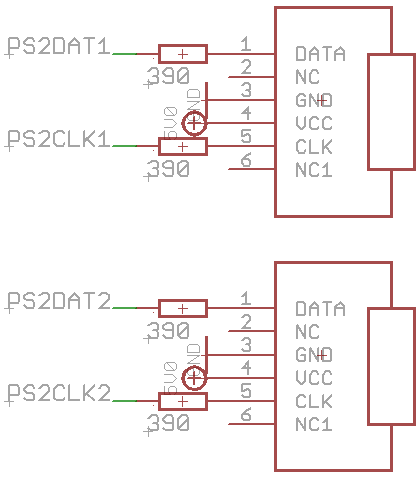


Name	Direction	Typical Function	Papilio One Pin
LED1	Output	Player 1	P57
LED2	Output	Player 2	P53
LED3	Output	Flipped	P40
LED4	Output		P35

MegaWing Keyboard and Mouse

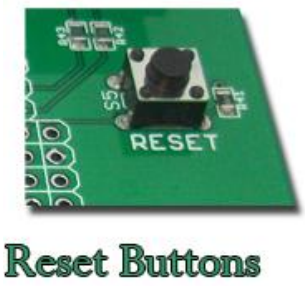
Keyboard and Mouse

The Papilio Arcade MegaWing has two PS/2 ports that can be used to connect a keyboard and mouse.



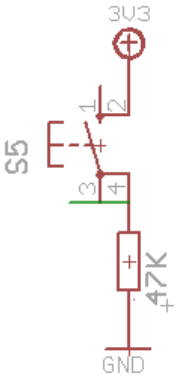
Name	Direction	Function	Papilio One Pin
PS/2 A CLK	Output	Clock	P92
PS/2 A DATA	Input	Data	P91
PS/2 B CLK	Output	Clock	P79
PS/2 B DATA	Input	Data	P70

MegaWing Reset Button



Reset Buttons

The Papilio One has a reset button that resets the game instead of the entire FPGA.



Name	Direction	Function	Papilio One Pin
Reset	Input	Reset	P67

Individual Arcade Wings

The Papilio Arcade can also be built from individual Wings instead of using the Arcade MegaWing. Using individual Wings provides more flexibility since each hardware section can be upgraded independently and the Wings can be used with other projects. The following sections detail the Papilio Wings that can be used to build a Papilio Arcade.

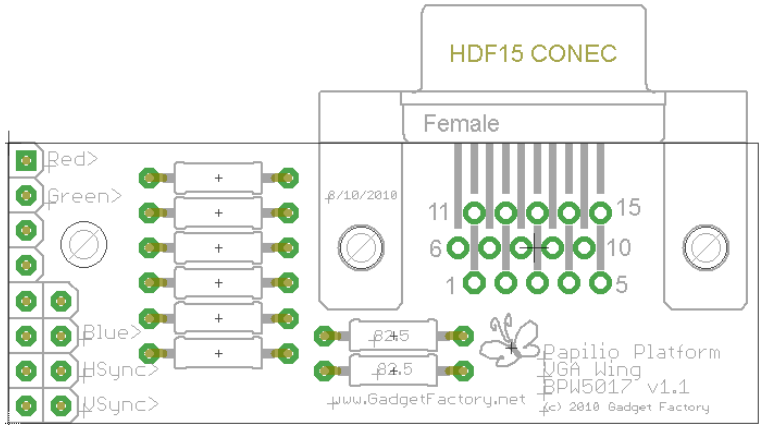
VGA Wing

The VGA Wing provides 8 colors by using three 270 ohm resistors for RGB and two 82.6 ohm resistors for hsync and vsync.

More information such as schematics and pinouts can be found on the VGA Wing homepage.

Pin	Name	Direction	Function
1	Vertical Sync	Output	Vertical Sync
2	Horizontal Sync	Output	Horizontal Sync
3	BLUE1	Output	Blue Bit 1
4	N.C.	Output	Not Connected
5	N.C.	Output	Not Connected
6	N.C.	Output	Not Connected
7	GREEN1	Output	RED Bit 1
8	RED1	Output	RED Bit 1

Table 1 - VGA Wing Pinouts



Joystick Wing

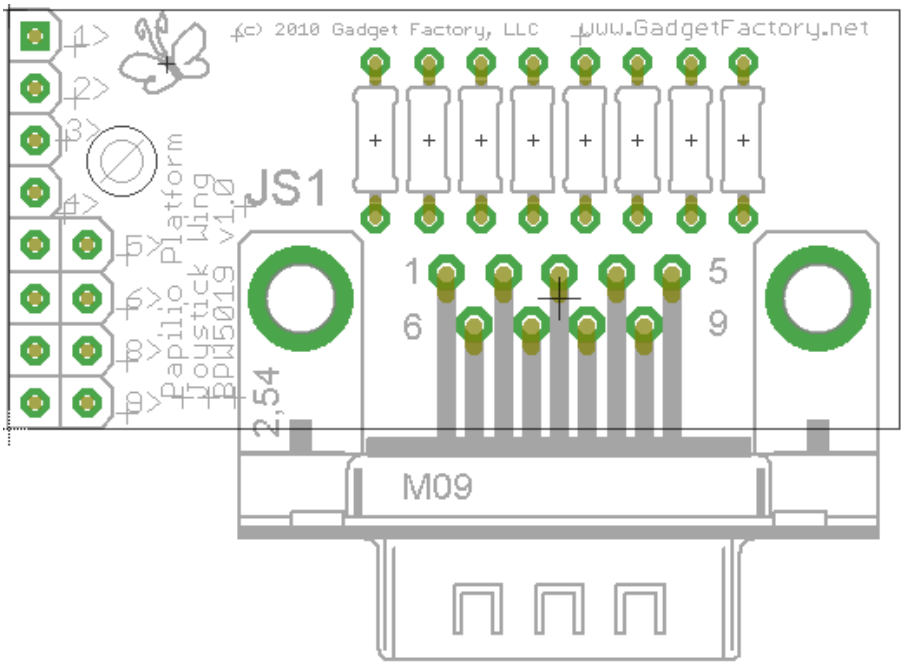
The Joystick Wing provides a DB9 connector that can be directly connected to an Atari 2600 or Commodore joystick.

More information such as schematics and pinouts can be found on the Joystick Wing homepage.

Custom joysticks can be wired to the DB9 connector using the following pinout table:

Pin	Name	Direction	Function
1	Up	Input	Up Bit 1
2	Down	Input	Down Bit 1
3	Left	Input	Left Bit 1
4	Right	Input	Right Bit 1
5	NC	Input	Not Connected
6	Button	Input	Button Bit 1
7	5V	N/A	5V
8	NC	Input	Not Connected

Table 2 - Joystick Wing Pinouts



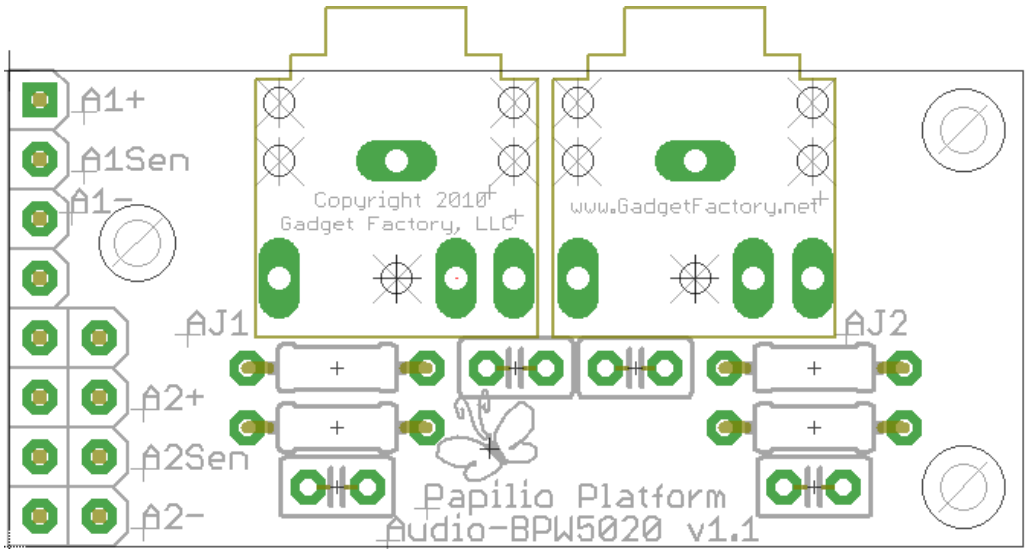
Audio Wing

The Audio Wing provides two 1/8" stereo jacks that capable of outputting CD quality audio.

More information such as schematics and pinouts can be found on the Audio Wing homepage.

Pin	Name	Direction	Function
1	A2-	Output	A2- Bit 1
2	A2Sen	Output	A2Sen Bit 1
3	A2+	Output	A2+ Bit 1
4	N.C.	Output	Not Connected
5	N.C.	Output	Not Connected
6	A1-	Output	A1- Bit 1
7	A1SEN	Output	A1Sen Bit 1
8	A1+	Output	A2+ Bit 1

Table 3 - Audio Wing Pinouts



Chapter 2

Papilio Arcade Quick Start

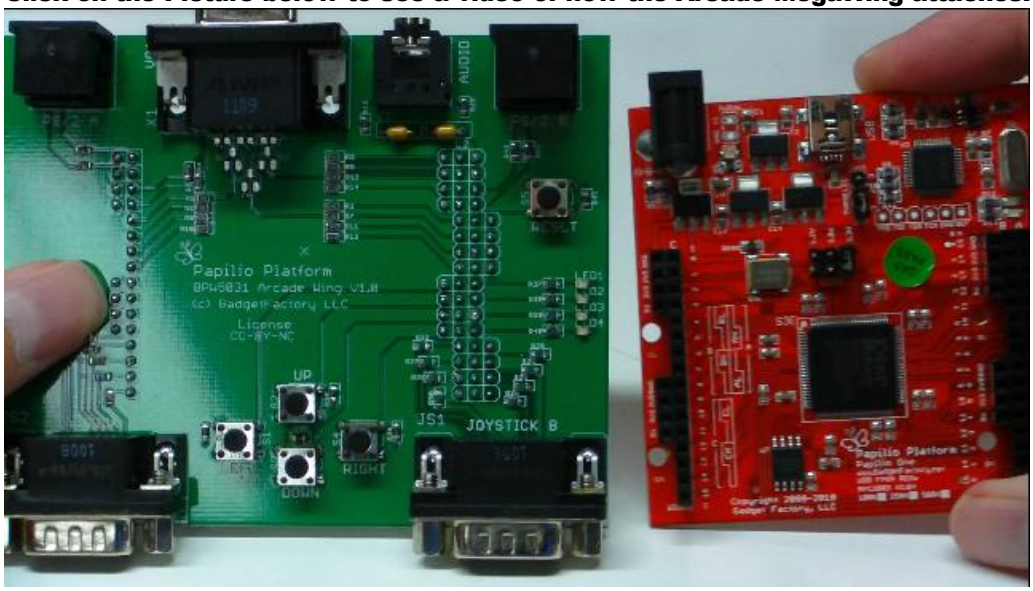
The quickest way to get started with the Papilio Arcade is to load a Home Brew Pong game that has been built on the Pac-Man Hardware. This quick start section will walk you through setting up the hardware, downloading a bit file for the Pong game, and loading it to the Papilio Arcade board. The Quick start guide also gets you started with synthesizing Arcade projects so you can make customizations, add ROM's, and start making homebrew games for classic Arcade hardware.

Setting up the Hardware

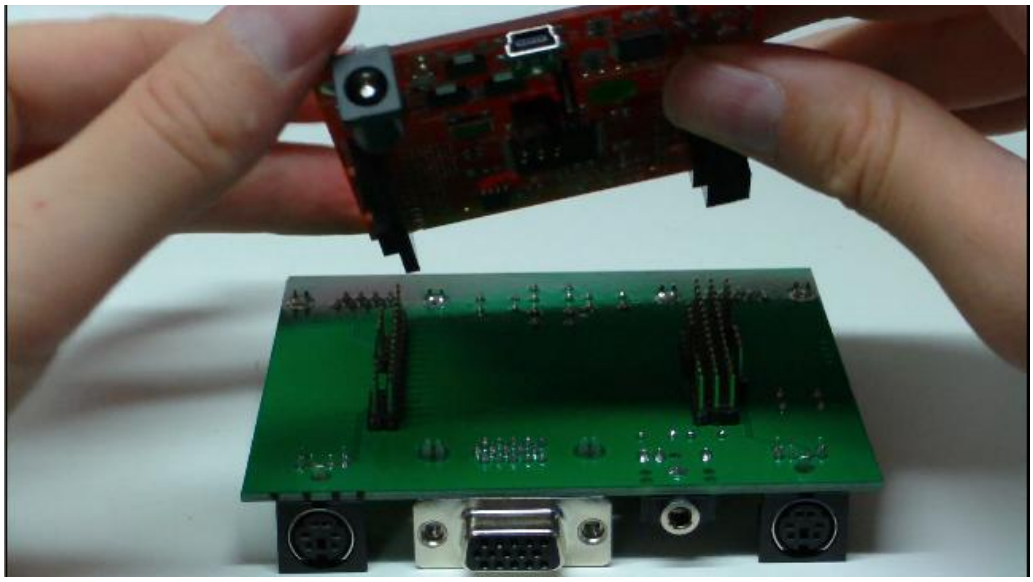
Arcade MegaWing

Setting up the Arcade MegaWing hardware is the easiest solution since everything is integrated into a convenient board that snaps onto the Papilio One 500K.

Click on the Picture below to see a video of how the Arcade MegaWing attaches.



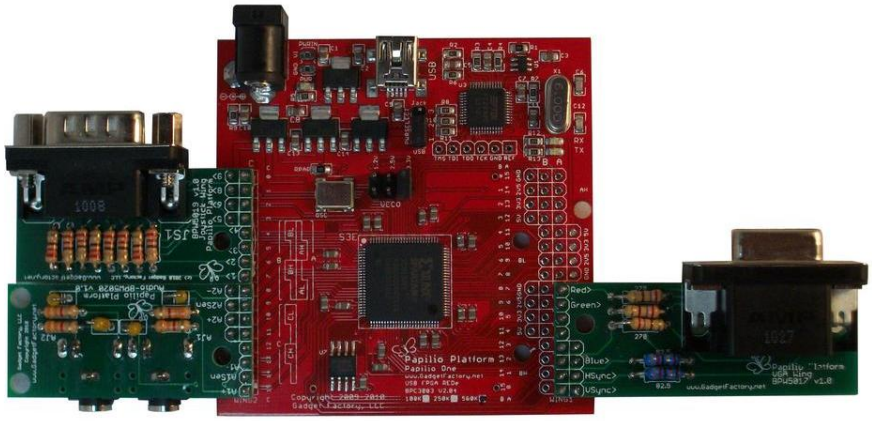
Ensure that there are Female headers soldered onto the Papilio One and Male headers soldered onto the Arcade MegaWing, and then snap the two boards together with the VGA port of the Arcade MegaWing on the top side.



Individual Wings

Individual Wings should be attached according to the following table:

Wing	WingSlot Location
VGA Wing	AL (Row A – Lower 8 bits)
Audio Wing	CL (Row C – Lower 8 bits)
Joystick Wing	CH (Row C – Higher 8 bits)



Loading Pong Demo

The Homebrew Pong Demo game is used as the example for this Quick Start guide since it is Open Source and can be included in a ready to run bit file without any worries about users owning a legal copy of the ROM. It is an example of a ROM running on the Pac-Man hardware. It is possible to run the original Pac-Man ROM, other commercial games written for the Pac-Man hardware, or homebrew games written for the Pac-Man hardware. Homebrew games can be distributed as bit files that can be quickly loaded onto the hardware, but commercial games must have the ROM's supplied by the end user. The provided ROM's must be synthesized into the hardware by the end user and is covered in the synthesize Pac-Man section.

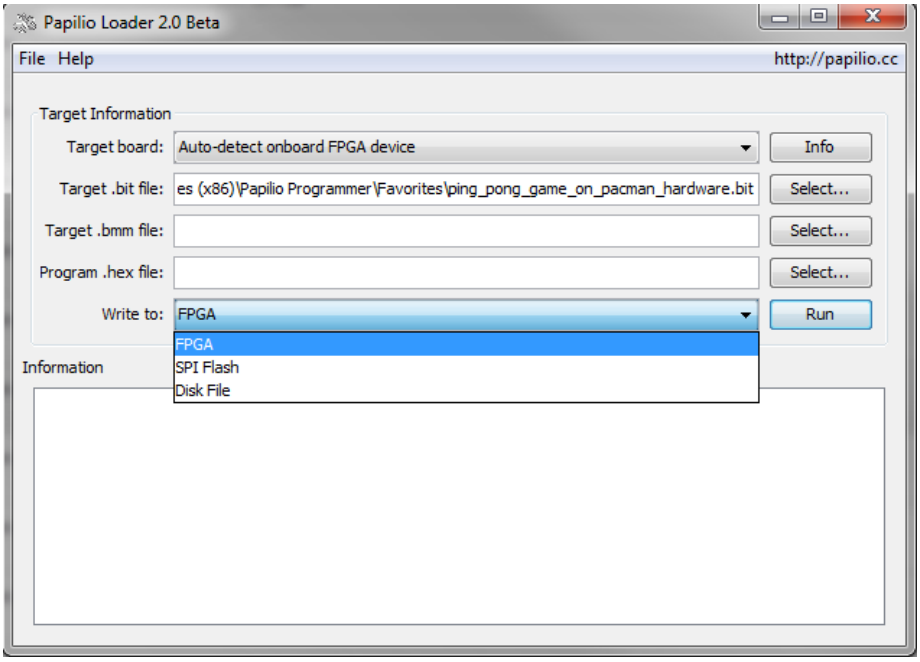
To get started ensure you have the Papilio Loader installed and associated with *.bit files. Follow the Papilio One Quickstart Guide to set this up.

1. Download the Papilio Arcade bitstream file. There are bit files for the different hardware options, be sure to download the one appropriate for your hardware.

2. Plug the Papilio One into the USB port.

3. Double click on the downloaded bit file to load the file to the Papilio Arcade Hardware

4. You have the option to load the bit file directly to the FPGA or to the SPI Flash. Loading the bit file directly to the FPGA is temporary and will be lost when the power is removed. Writing to SPI Flash takes a little bit longer but will cause the bit file to be loaded by the FPGA every time power is cycled.



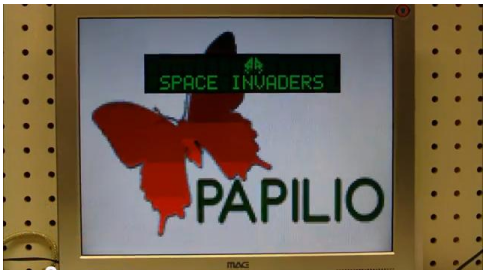
5. Once you have decided on the location to load the bit file press "Run" to load the Pong Homebrew Demo to the hardware. Your loading preferences can be saved under the "File/Preferences" menu.

Loading Zetris

Zetris is a homebrew game that is written as a sketch for the ZPUino soft processor. The ZPUino is a 32 bit processor running at 100Mhz with an integrated VGA display adapter and YM2149 sound processor. Everything is controlled by a sketch and an easy to use VGA library from the Arduino IDE. The fast ZPUino processor gives you plenty of processing power to make cool homebrew games.

Click on the picture below to see a video of Zetris in action.

Take a look at the VGALiquidCrystal library to see other cool things you can do with the ZPUino. The VGALiquidCrystal library allows you to convert Arduino LCD code to run on a VGA monitor by changing one line of code.



Steps to load Zetris

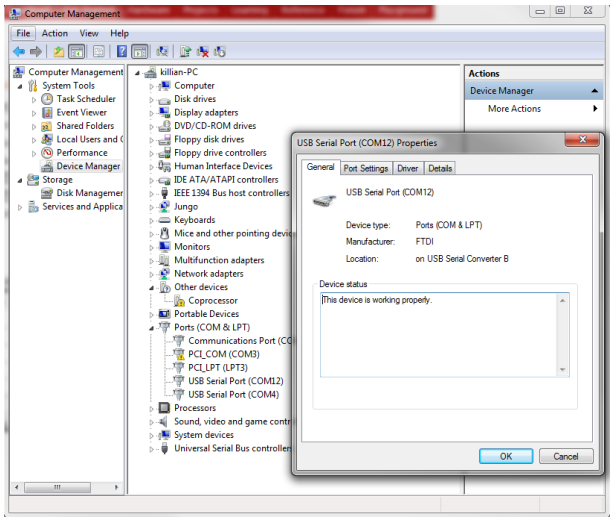
1. Visit Alvaro's ZPUino download page and download the bit file for you hardware type. This quick start guide assumes you are using a Papilio One 500K with the Arcade MegaWing. If you are using something else you need to adjust your downloads accordingly.

2. Download the ZPUino Soft Processor with HQVGA bit file and load it to the Papilio Arcade hardware using the Papilio Loader as outlined in the Papilio One Quickstart Guide.

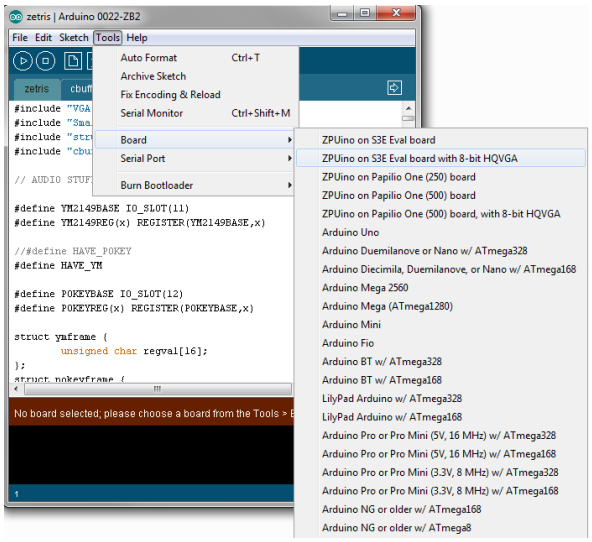
3. Download the ZPUino IDE and unzip to a directory of your choice.

4. Download the Zetris sketch. Zetris Zip or Zetris Source Code on Github

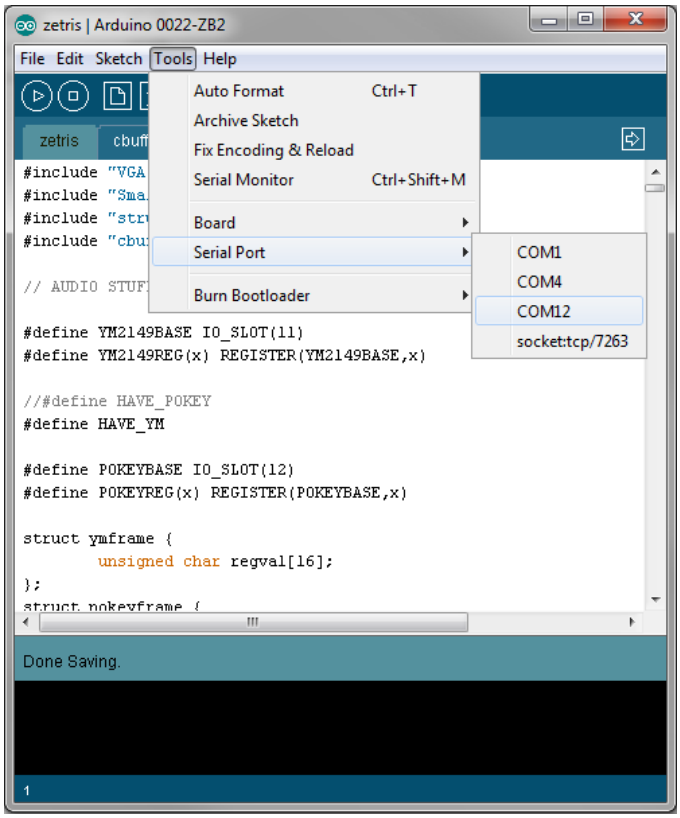
5. Determine which port the second Papilio COM port is connected to. If you are not sure which serial port is associated with the Papilio One then take a look at the Device Management tool in the Control Panel. It should be marked as Serial Converter B in the properties.



6. Open the Zetris sketch in the ZPUino IDE and then set the board type to the ZPUino with HQVGA.



7. Set the serial port to the second serial port of the Papilio that you determined in the previous step.



8. Upload the sketch to the Papilio and you should see Zetris!

Papilio Arcade App

Each arcade hardware type has several games that can be loaded to them. Even though the cabinet and game may have looked totally different the motherboard inside the cabinet could very easily have been Pac-Man, Scramble, Galaxian, or Space Invaders hardware. The Papilio Arcade App allows different video games (ie ROMs) to be loaded/merged into the arcade hardware (ie bit file). The Papilio Arcade app allows you to load ROM's onto the arcade hardware without going through the time consuming synthesis process. The arcade hardware is included with the Papilio Arcade app in the form of a bit file that does not have any code running on it. It is like a Pac-Man motherboard with empty EEPROM chips, there is no copyrighted code running on it and it doesn't do anything. End users can provide homebrew ROMs or commercial ROMs that they own to the application and it will take care of merging the code contained in the ROMs into the blank bit files and loading the result onto the Papilio Arcade hardware. It only takes a couple seconds to load a new game, quite an improvement from the several minutes it takes to synthesize each game.

The Papilio Arcade App is a work on progress and is expected to be available in December of 2011.

Synthesizing Pac-Man from Source Code

The Papilio Arcade project includes pre-synthesized bit files with the Pac-Man hardware running a homebrew pong game. When the user is ready to run the Pac-Man ROMs the end user must have a legal version of the Pac-Man game and must synthesize the Pac-Man ROMs into the Pac-Man hardware design. The following describes the process for the Pac-Man hardware and is very much the same for the other Arcade hardware types.

Download Source

Start out by downloading the VHDL source package that describes the Pac-Man hardware (and many other classic games) from the Gadget Factory Papilio Arcade source code page on GitHub..

Unzip the source into any desired directory.

Provide ROMs

Locate the roms directory and copy *legal* versions of the ROM files that have been obtained from a Pac-Man arcade game. For Pac-Man there are 4 different games that will run on the hardware. Look at the information provided further down in this guide or at the "scripts/build_roms_*.bat" files to determine the ROM files that are needed by each game. Place the necessary ROM files in the directory that corresponds with the game that is going to be synthesized into the Pac-Man Hardware. The available games are:

- Pac-Man
- Crush Roller
- Gorkans
- Lizard Wizard

Navigate to the scripts directory and run the "build_roms_*.bat" file that corresponds to the game that you are synthesizing in. This will convert the supplied ROM files into the VHDL files needed to synthesize the game into the Pac-Man Hardware.

Change Settings

There are several settings that can be controlled at the time of synthesis. To modify the following setting open the source/pacman.vhd file.

Free Play

To require a coin to play or to set Freeplay where simply pressing the start button is required search for and set the following line:

```
dipsw_reg(3 downto 2) <= "10"; -- pacman (3)
```

Set the "10" portion of the line according to Table 1.

TABLE 3

Mode	Setting
1 Coin = 1 Play	"11"
1 Coin = 2 Play	"01"
2 Coin = 1 Play	"10"
Freeplay	"00"

Upright/Cocktail table

Search for the following line:

```
in1_reg(7) <= '0'; -- table
```

Set the '0' portion of the line to either '1' or '0' according to Table 2.

TABLE 4

Mode	Setting
Cocktail Table	'0'
Upright	'1'

Number of lives per play

To set the number of lives per play search for the following line:

```
dipsw_reg(3 downto 2) <= "10"; -- pacman (3)
```

Set the "10" portion of the line according to Table 3.

TABLE 5

Mode	Setting
1 Packman	"11"
2 Packmen	"01"
3 Packmen	"10"
5 Packmen	"00"

Points until extra life

To set the number of lives per play search for the following line:

```
dipsw_reg(3 downto 2) <= "10"; -- pacman (3)
```

Set the "10" portion of the line according to Table 4.

TABLE 6

Mode	Setting
Bonus at 10,000	"11"
Bonus at 15,000	"01"
Bonus at 20,000	"10"
No Bonus	"00"

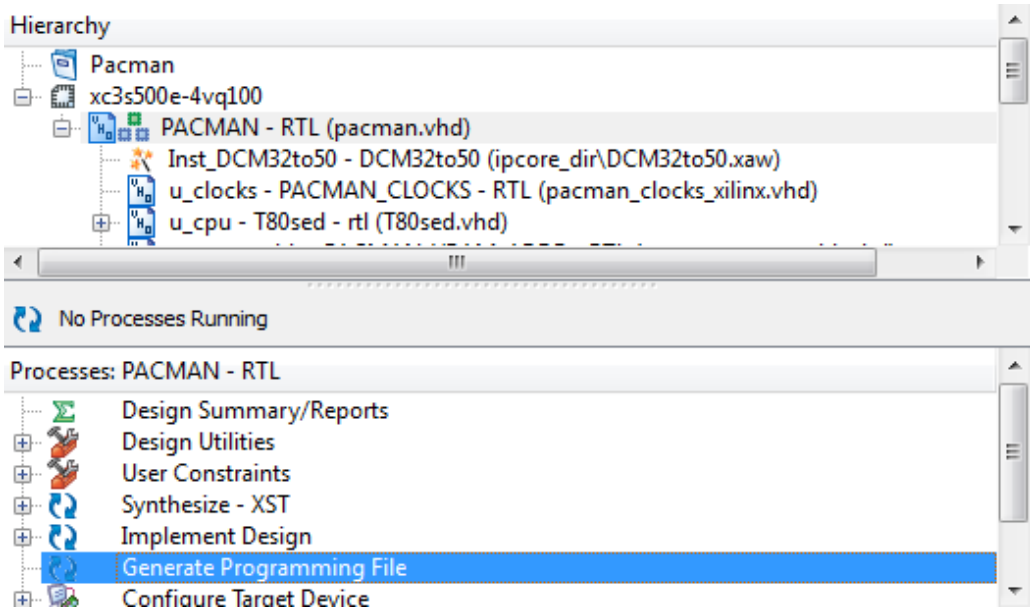
Synthesize

Once the ROM files are generated and all the desired settings are set then it is time to synthesize the design. It can take quite some time to synthesize the design so be prepared to wait a while.

- Open the ISE project file from the following path. There are multiple ISE projects, choose the project that corresponds to the hardware you have. For example, if you are using the Arcade MegaWing with the P1 500K then you would select:

build\pacman_p1_500k_MegaWing.xise

- Make sure the top level Pacman module is selected and then double click on "Generate Programming File".



Load Bit Filie

Once the design is synthesized there will be a pacman.bit file generated in the build directory.

Ensure you have the Papilio Loader installed and associated with *.bit files. Follow the Papilio One Quickstart Guide to set this up.

- Double click on the newly synthesized build/pacman.bit file.

Chapter 3

Arcade Hardware Types

Overview

There are several Open Source HDL projects that describe the hardware required to implement classic arcade games on an FPGA. These designs allow for learning, exploration, and a deep understanding of how simple 8 bit systems work. Learning the ins and outs of these classic arcade games is not only fun, but it provides a very solid foundation for building your own systems, whether it be games or gadgets.

There are currently four hardware types that run on the Papilio Arcade hardware. What this means is that we have VHDL source code that defines all of the chips that were on the original motherboards for these hardware types. The code has been modified to run on the Papilio Arcade hardware and the following is currently provided:

- Full source code released under an Open Source license.
- A Xilinx ISE project configured for the Papilio Arcade. The project can be opened in ISE Webpack and directly modified and synthesized to customize the game type's hardware.
- Scripts to transform ROMs into VHDL files needed by the Xilinx ISE project. These scripts let you synthesize the code contained in the game ROMs into the hardware types.
- Pre-synthesized bit files for each hardware type, these bit files are either empty or contain a homebrew ROM. Bit files can be loaded onto the hardware and tell the Papilio to become the hardware type defined by the VHDL.
- Scripts to merge ROMs into the pre-synthesized bit file without requiring a full synthesis. Synthesizing a design can take around five minutes, these scripts allow game ROMs to be merged into existing bit files in a few seconds.

Each game type can typically run several different games. It was a common practice to use popular game motherboards to develop new games. Popular games like Pac-man often had other games, like Gorkans, running on the same hardware type. In addition, homebrew games can run on each hardware type and we hope to see many more homebrew games available.

The Papilio Arcade App and the scripts to merge ROMs are meant to allow different games, defined by ROMs, to be loaded into the hardware types, defined by bit files.

Source Code

The source code for all arcade Hardware types can be downloaded from the Papilio Arcade source code page on GitHub, or the latest zip file can be downloaded directly.

PAC-MAN Hardware

Cabinet image courtesy of http://www.classicgaming.com/mame32qa/

Pac-Man Hardware Specifications

- Platform — *NAMCO 8-bit PCB*
- CPU — *Z80A at 3.072 MHz*
- ROM — *16K in four, 4K chips*
- RAM — *~2K*
- Display — *Raster*
  - Orientation — *Vertical*
  - Resolution — *224x288*
  - Colors — *16*
  - Attributes — *Eight 16x16 hardware sprites*
  - Refresh rate — *60.61 Hz*
- Sound — *Monophonic 3-voice waveform sound generator chip*
- Controls — One *4-way leaf joystick, 1P/2P buttons*
- Models — *Upright, and Cocktail*

Pac-Man Hardware Source Code

Pac-Man Source code on GitHub.

Pac-Man Game

Game Image

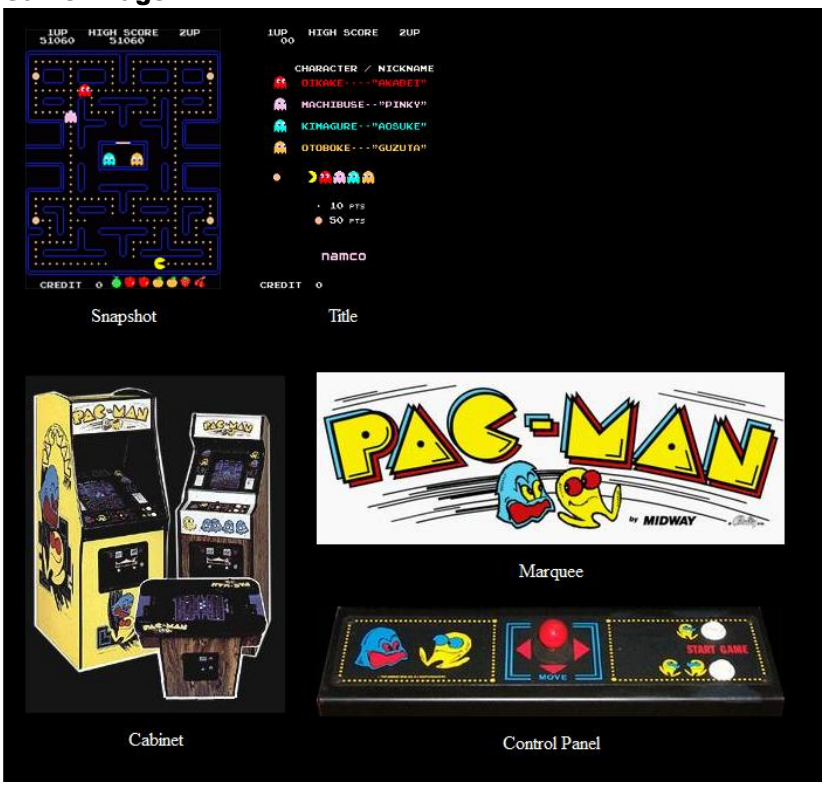


Figure 1 - Image courtesy of mamedb.com

Game Database Link

http://www.mamedb.com/game/pacman

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
pacman*/scripts/build_roms_pacman.bat
```

Merging the games ROM into a bit file

1. Provide legal ROMs in the following directory:

```
pacman*/roms/pacman
```

2. Run the following script:

```
pacman*/scripts/merge_roms_pacman.bat
```

Crush Roller Game

Game Image

Figure 2 - Image courtesy of mamedb.com

Game Database Link

http://www.mamedb.com/game/crush2

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
pacman*/scripts/build_roms_crush2.bat
```

Merging the games ROM into a bit file

1. Provide legal ROMs in the following directory:

```
pacman*/roms/crush2
```

2. Run the following script:

```
pacman*/scripts/merge_roms_crush2.bat
```

Gorkans Game

Game Image

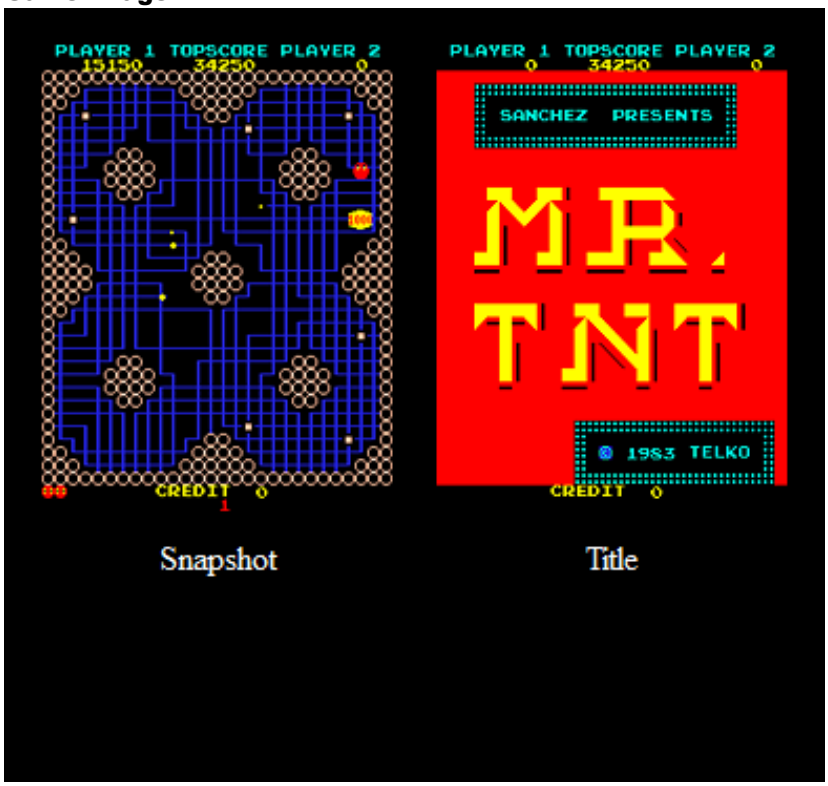


Figure 3 - Image courtesy of mamedb.com

Game Database Link

http://www.mamedb.com/game/gorkans

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
pacman*/scripts/build_roms_gorkans.bat
```

Merging the games ROM into a bit file

1. Provide legal ROMs in the following directory:

```
pacman*/roms/gorkans
```

2. Run the following script:

```
pacman*/scripts/merge_roms_gorkans.bat
```

Lizard Wizard Game

Game Image

Figure 4 - Image courtesy of mamedb.com

Game Database Link

http://www.mamedb.com/game/lizwiz

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
pacman*/scripts/build_roms_lizwiz.bat
```

Merging the games ROM into a bit file

At this time it is not possible to merge Lizard Wizard, the game must be synthesized in.

Frogger Hardware

Cabinet image courtesy of http://www.classicgaming.com/mame32qa/

Frogger Hardware Specifications

- CPU:
  - Z80 at 3.072 MHz
  - Z80 at 1.78975 MHz
- Sound – AY-3-8910A at 1.789750 MHz
- Display
  - Resolution – 768 x 224
  - Orientation – Vertical
  - Refresh Rate – 60.61 Hz

Frogger Hardware Source Code

Frogger Source code on GitHub.

Synthesis Note for Frogger Hardware

The available source code is the hardware image for both the Frogger and Scramble. They only have a minor difference in hardware configuration. There is a constant at the scramble_top.vhd that controls which configuration to select.

```
constant I_HWSEL_FROGGER  : boolean := TRUE;
```

The current source code is configured for Frogger hardware. Just set the I_HWSEL_FROGGER to FALSE to select the Scramble configuration.

Amidar Game

Game Image

Figure 5 - Image courtesy of mamedb.com

Game Database Link

http://www.mamedb.com/game/amidars

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
scramble*/scripts/build_roms_scramble.bat
```

Merging the games ROM into a bit file

At this time it is not possible to merge Amidar, the game must be synthesized in.

Sega Frogger Game

Game Image

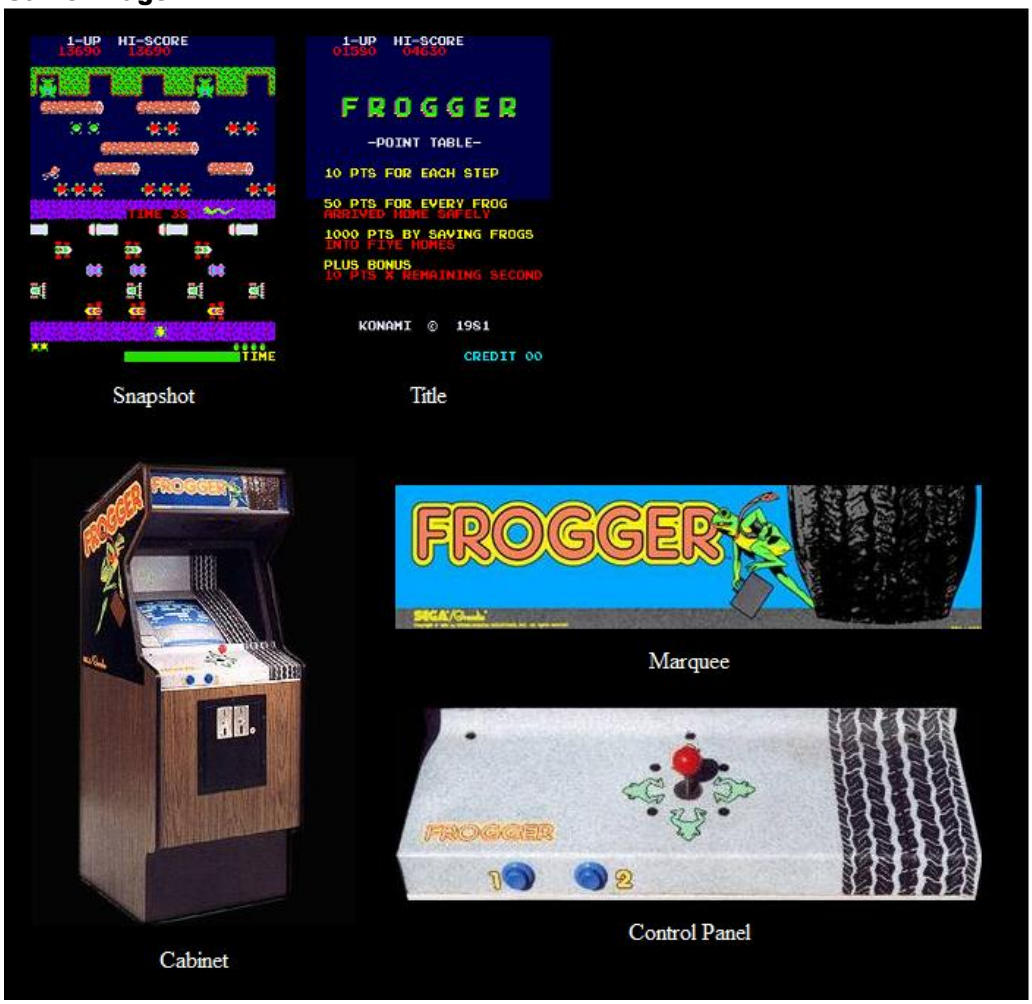


Figure 6 - Image courtesy of mamedb.com

Game Database Link

http://www.mamedb.com/game/froggers2

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
scramble*/scripts/build_roms_frogger.bat
```

Merging the games ROM into a bit file

At this time it is not possible to merge Frogger, the game must be synthesized in.

Stern Scramble Game

Game Image

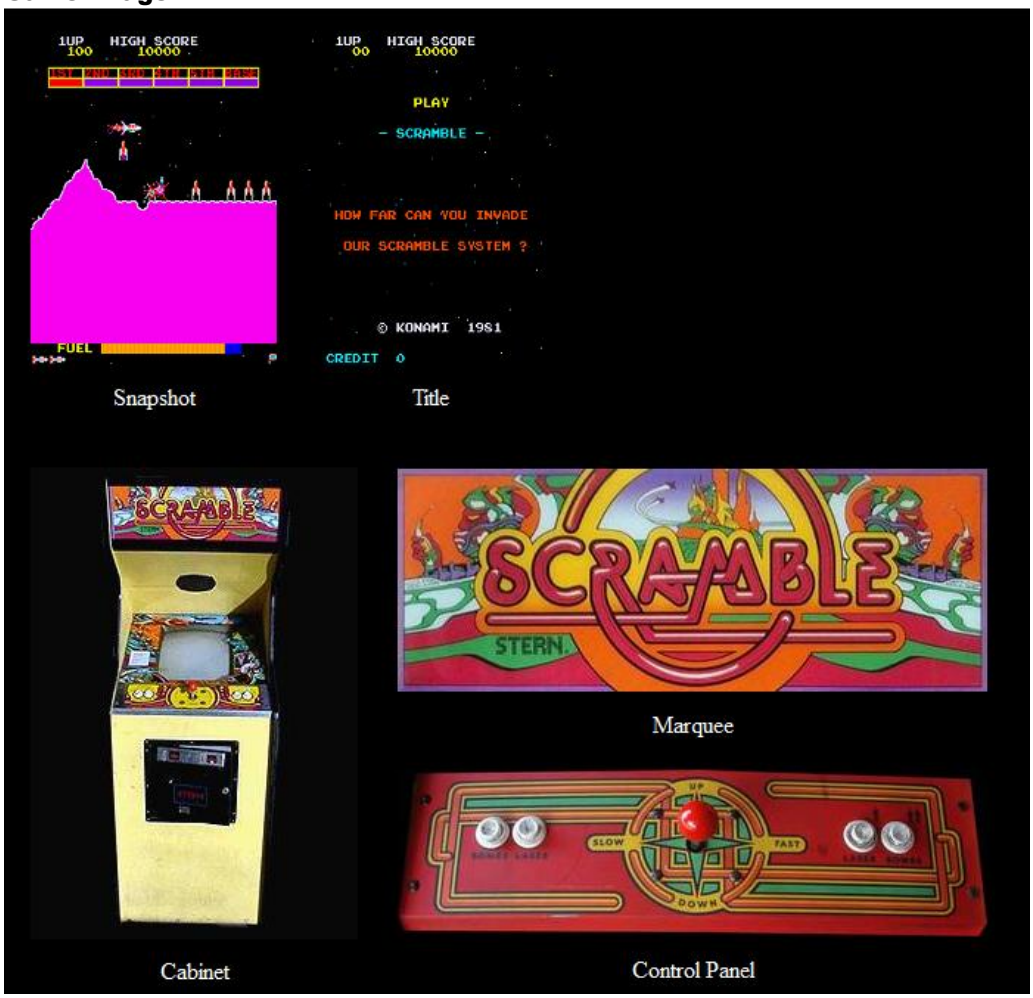


Figure 7 - Image courtesy of mamedb.com

Game Database Link

http://www.mamedb.com/game/scrambles

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
scramble*/scripts/build_roms_scramble.bat
```

Merging the games ROM into a bit file

At this time it is not possible to merge Scramble, the game must be synthesized in.

Stern The End Game

Game Image

Figure 8 - Image courtesy of mamedb.com

Game Database Link

http://www.mamedb.com/game/theends

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
scramble*/scripts/build_roms_stern_the_end.bat
```

Merging the games ROM into a bit file

At this time it is not possible to merge The End, the game must be synthesized in.

The End Game

Game Image

Figure 9 - Image courtesy of mamedb.com

Game Database Link

http://www.mamedb.com/game/theend

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
scramble*/scripts/build_roms_the_end.bat
```

Merging the games ROM into a bit file

At this time it is not possible to merge The End, the game must be synthesized in.

Crazy Kong Game – Not Working Yet

Game Image

Figure 10 - Image courtesy of mamedb.com

Game Database Link

http://www.mamedb.com/game/ckongs

This game should work on the Frogger hardware type, if anyone gets this game working please update us in the forum.

Galaxian Hardware

Cabinet image courtesy of http://www.classicgaming.com/mame32qa/

Galaxian Hardware Specifications

- CPU:
  - Z80 at 3.072 MHz

Galaxian Hardware Source Code

Galaxian Source code on GitHub.

Galaxian Game

Game Image

Figure 11 - Image courtesy of mamedb.com

Game Database Link

http://www.mamedb.com/game/galaxian

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
galaxian*/scripts/build_roms_galaxian.bat
```

Merging the games ROM into a bit file

At this time it is not possible to merge The End, the game must be synthesized in.

Space Invaders Hardware

Cabinet image courtesy of http://www.classicgaming.com/mame32qa/

Space Invaders Hardware Specifications

- CPU:
  - 8080 Processor

Space Invaders Hardware Source Code

Invaders Source code on GitHub.

Space Invaders Screen Note

The Space Invaders screen is rotated 180 degrees in the opposite direction as the rest of the games. We are working on a method to flip the screen so the VGA monitor does not need to be manually rotated.

Sea Earth Invasion Game

Game Image

Figure 12 - Image courtesy of mamedb.com – NOTE: The snapshot in this image does not match the game.

Game Database Link

http://www.mamedb.com/game/searthin

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
invaders*/scripts/build_roms_searthin.bat
```

Merging the games ROM into a bit file

At this time it is not possible to merge The End, the game must be synthesized in.

Alien Invasion Game

Game Image

Figure 13 - Image courtesy of mamedb.com – NOTE: The snapshot in this image does not match the game.

Game Database Link

http://www.mamedb.com/game/alieninvp2

Synthesizing the game

Follow the Quick Start Guide for synthesizing games but provide ROMs that match the Mame database entry above and use the following script to include this games ROMs:

```
invaders*/scripts/build_roms_alieninv.bat
```

Merging the games ROM into a bit file

At this time it is not possible to merge The End, the game must be synthesized in.

Commodore 64 SID Audio Chip

The Papilio Arcade can recreate the Commodore 64 SID audio chip!

For more information watch the video and visit the project page.

Chapter 4

Homebrew Games

Overview

Homebrew games can be created with the ZPUino soft processor or using development environments for the classic arcade hardware types.

ZPUino Games

The ZPUino is a 100Mhz, 32-bit soft processor with an integrated VGA controller and LM2149 audio chip. Games can be written as Arduino sketches and use the VGA Library to interface with the VGA controller. LM2149 audio tracks can be integrated into projects using the smallFS library.

Zetris is a Tetris clone that provides a solid code example of how to make homebrew games using the ZPUino soft processor.

For more information look at the ZPUino Quickstart Guide.

Pac-Man Hardware Games

The ultimate goal is to setup programming environments with hello world examples for each of the available hardware types. This is a work in progress, below are some links to projects that can be used to make a programming environment for the Pac-Man hardware type.

http://umlautllama.com/projects/pacdocs/

http://www.csh.rit.edu/~jerry/arcade/pacman/

http://www.ascotti.org/programming/pie/hardware.htm

http://www.ascotti.org/programming/pie/pie.htm

http://fms.komkon.org/EMUL8/HOWTO.html

Chapter 5

Conclusion and Future Direction

Conclusion

Thank you for choosing the Papilio Arcade. We believe that you'll find an exciting, expandable, hackable world of classic arcade goodness just waiting to be explored and exploited – all in one place, all on one board. The Papilio Arcade is a powerful, stable, and expandable hardware design for custom arcade games and demos that are simply not possible with static motherboards. Welcome to the Papilio Arcade!

Future Direction

The future direction of the Papilio Arcade is to continue porting more hardware types and to provide development environments them that can be used to make homebrew games. We want to make this the ultimate Arcade hacking hardware for homebrew and demoscene enthusiasts.

Other Papilio Uses

The Papilio One is a flexible FPGA board that can be used for purposes other than the Papilio Arcade. Some of the other uses that are either available or in the works are:

Existing

- "Sump" Logic Analyzer – A 32 channel 200Mhz Open Source Logic Analyzer.
- Arduino IDE Development Environment – Write your own sketches for the Papilio One using an ATMega103 compatible soft processor.

Appendix A - Attribution

- FPGA Arcade has done a tremendous job of making many Open Source Arcade projects available on his website. All of the current arcade hardware types come from the FPGA Arcade website.
- Alex – Alex ported many of the arcade hardware types to the Papilio and did a general cleanup of the source code structure.

Appendix B – Related Links

Great Overview of Pac-Man.

Good Technical info

Papilio Arcade MegaWing Test Plan

User Guides for many Arcade Games

http://www.jeffsromhack.com/toolbox/index.htm

http://home.comcast.net/~jpittman2/pacman/pacman_opmanual.pdf