1 Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Installation</td>
<td>3</td>
</tr>
<tr>
<td>1.2 Quick Start</td>
<td>5</td>
</tr>
<tr>
<td>1.3 User Guide</td>
<td>8</td>
</tr>
<tr>
<td>1.4 Contribute</td>
<td>34</td>
</tr>
</tbody>
</table>
Open source **ecosystem for open FPGA boards**. It was inspired by PlatformIO.

Apio (pronounced [a.pjo]) is a **multiplatform toolbox**, with static pre-built packages, project configuration tools and easy command interface to verify, synthesize, simulate and upload your **verilog** designs.

Apio is used by Icestudio.

**Source code:** [https://github.com/FPGAwars/apio](https://github.com/FPGAwars/apio)
1.1 Installation

Apio is written in Python and works on Linux (+ARM), Mac OS X, Windows.

Contents

- Installation
  - System requirements
  - Install Apio
  - Install FTDI drivers
  - Install Serial drivers

1.1.1 System requirements

Operating System  Linux (+ARM), Mac OS X or Windows

Python Interpreter  Python 2.7, Python 3.5+

Attention: Windows Users: Please Download the latest Python and install it. DON’T FORGET to select Add python.exe to Path feature on the “Customize” stage, otherwise Python Package Manager pip command will not be available.
Terminal Application  All commands below should be executed in Command-line application (Terminal). For Mac OS X and Linux OS - Terminal application, for Windows OS – cmd.exe application.

Access to Serial Ports (USB/UART)  Windows Users: Please check that you have correctly installed USB driver from board manufacturer.

Linux Users: Ubuntu/Debian users may need to add own “username” to the “dialout” group if they are not “root”, doing this issuing a `sudo usermod -a -G dialout $USER`.

1.1.2 Install Apio

The latest stable version of Apio may be installed or upgraded via Python Package Manager (pip) as follows:

```
$ pip install -U apio
```

If `pip` command is not available run `easy_install pip`.

Note that you may run into permissions issues running these commands. You have a few options here:

- Run with `sudo` to install Apio and dependencies globally
- Specify the `pip install --user` option to install local to your user
- Run the command in a `virtualenv` local to a specific project working set.

Note:  Debian users can also install the application and its packages by executing the following commands. (These packages may not be updated).
1.1.3 Install FTDI drivers

For boards with a FTDI interface.

```
$ apio drivers --ftdi-enable
```

To revert the FTDI drivers configuration.

```
$ apio drivers --ftdi-disable
```

1.1.4 Install Serial drivers

For boards with a Serial interface.

```
$ apio drivers --serial-enable
```

To revert the Serial drivers configuration.

```
$ apio drivers --serial-disable
```

1.2 Quick Start

Once apio has been installed and the drivers have been correctly configured is time to start playing with your FPGA!

1.2.1 Install packages

```
$ apio install --all
```

1.2.2 Create a project

Go to your project’s directory or try the examples

```
$ apio examples -d leds
$ cd leds
```

Configure your board

Find your board in the list
$ apio boards --list

Supported boards:

<table>
<thead>
<tr>
<th>Board</th>
<th>FPGA</th>
<th>Type</th>
<th>Size</th>
<th>Pack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cat-board</td>
<td>iCE40-HX8K-CT256</td>
<td>hx</td>
<td>8k</td>
<td>ct256</td>
</tr>
<tr>
<td>TinyFPGA-B2</td>
<td>iCE40-LP8K-CM81</td>
<td>lp</td>
<td>8k</td>
<td>cm81</td>
</tr>
<tr>
<td>TinyFPGA-BX</td>
<td>iCE40-LP8K-CM81</td>
<td>lp</td>
<td>8k</td>
<td>cm81</td>
</tr>
<tr>
<td>alhambra-ii</td>
<td>iCE40-HX4K-TQ144</td>
<td>hx</td>
<td>8k</td>
<td>tq144:4k</td>
</tr>
<tr>
<td>blackice</td>
<td>iCE40-HX4K-TQ144</td>
<td>hx</td>
<td>8k</td>
<td>tq144:4k</td>
</tr>
<tr>
<td>blackice-ii</td>
<td>iCE40-HX4K-TQ144</td>
<td>hx</td>
<td>8k</td>
<td>tq144:4k</td>
</tr>
<tr>
<td>fpga101</td>
<td>iCE40-UP5K-SG48</td>
<td>up</td>
<td>5k</td>
<td>sg48</td>
</tr>
<tr>
<td>go-board</td>
<td>iCE40-HX1K-VQ100</td>
<td>hx</td>
<td>1k</td>
<td>vq100</td>
</tr>
<tr>
<td>iCE40-HX8K</td>
<td>iCE40-HX8K-CT256</td>
<td>hx</td>
<td>8k</td>
<td>ct256</td>
</tr>
<tr>
<td>iCE40-UP5K</td>
<td>iCE40-UP5K-SG48</td>
<td>up</td>
<td>5k</td>
<td>sg48</td>
</tr>
<tr>
<td>iCEBreaker</td>
<td>iCE40-UP5K-SG48</td>
<td>up</td>
<td>5k</td>
<td>sg48</td>
</tr>
<tr>
<td>iCEBreaker-bitsy</td>
<td>iCE40-UP5K-SG48</td>
<td>up</td>
<td>5k</td>
<td>sg48</td>
</tr>
<tr>
<td>iceblink-hx1k</td>
<td>iCE40-HX1K-VQ100</td>
<td>hx</td>
<td>1k</td>
<td>vq100</td>
</tr>
<tr>
<td>icestick</td>
<td>iCE40-HX1K-TQ144</td>
<td>hx</td>
<td>1k</td>
<td>tq144</td>
</tr>
<tr>
<td>icezum</td>
<td>iCE40-HX1K-TQ144</td>
<td>hx</td>
<td>1k</td>
<td>tq144</td>
</tr>
<tr>
<td>icoboard</td>
<td>iCE40-HX8K-CT256</td>
<td>hx</td>
<td>8k</td>
<td>ct256</td>
</tr>
<tr>
<td>kefir</td>
<td>iCE40-HX4K-TQ144</td>
<td>hx</td>
<td>8k</td>
<td>tq144:4k</td>
</tr>
<tr>
<td>upduino</td>
<td>iCE40-UP5K-SG48</td>
<td>up</td>
<td>5k</td>
<td>sg48</td>
</tr>
<tr>
<td>upduino2</td>
<td>iCE40-UP5K-SG48</td>
<td>up</td>
<td>5k</td>
<td>sg48</td>
</tr>
</tbody>
</table>

Create an apio.ini file with your board

$ apio init --board icestick

1.2.3 Process the project

Verify

Check your verilog code using Icarus Verilog

$ apio verify

Simulate

Simulate your test bench using Icarus Verilog and GTKWave
Note: GTKWave must be installed.

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debian</td>
<td><code>apt-get install gtkwave</code></td>
</tr>
<tr>
<td>Mac OSX</td>
<td><code>brew install gtkwave</code></td>
</tr>
<tr>
<td>Windows</td>
<td><code>apio install gtkwave</code></td>
</tr>
</tbody>
</table>

Build

Syntesize your project using Icestorm Tools

$ apio build

Upload

Connect your FPGA board and upload the bitstream using Icestorm Tools

$ apio upload

All the leds should turn on after 3 seconds

1.2. Quick Start
Congrats! Now You have your fully open source FPGA toolchain ready!

1.3 User Guide

Contents

- User Guide
  - Usage
  - Options
  - Project Commands
  - Setup Commands
  - Utility Commands

1.3.1 Usage

```
apio [OPTIONS] COMMAND [ARGS]
```

You can execute just `apio` to see the help:

```
$ apio
Usage: apio [OPTIONS] COMMAND [ARGS]...

Options:
  --version  Show the version and exit.
  --help     Show this message and exit.

Project commands:
  build      Synthesize the bitstream.
  clean      Clean the previous generated files.
```

(continues on next page)
lint  Lint the verilog code.
sim  Launch the verilog simulation.
time  Bitstream timing analysis.
upload  Upload the bitstream to the FPGA.
verify  Verify the verilog code.

Setup commands:
drivers  Manage FPGA boards drivers.
init  Manage apio projects.
install  Install packages.
uninstall  Uninstall packages.

Utility commands:
boards  Manage FPGA boards.
config  Apio configuration.
examples  Manage verilog examples.
raw  Execute commands using Apio packages.
system  System tools.
upgrade  Check the latest Apio version.

1.3.2 Options

--version
Show the version of Apio.

1.3.3 Project Commands

apio build

<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>• apio build</td>
</tr>
<tr>
<td>– Usage</td>
</tr>
<tr>
<td>– Description</td>
</tr>
<tr>
<td>– Options</td>
</tr>
<tr>
<td>– Examples</td>
</tr>
</tbody>
</table>

Usage

apio build [OPTIONS]

Description

Synthesize the bitstream: generates a **bin** file from a **verilog** and a **pcf** files.
Required packages: scons, icestorm.

Options

-b, --board
Select a specific board.

--fpga
Select a specific FPGA.

--size --type --pack
Select a specific FPGA size, type and pack.

-p, --project-dir
Set the target directory for the project.

-v, --verbose
Show the entire output of the command.

--verbose-yosys
Show the yosys output of the command.

--verbose-arachne
Show the arachne output of the command.

Note: All available boards, FPGAs, sizes, types and packs are showed in apio boards

Examples

1. Process the leds example

$ apio build
[] Processing icezum
--------------------------------------------------------------------------------------
˓→
yosys -p "synth_ice40 -blif hardware.blif" -q leds.v
arachne-pnr -d 1k -P tq144 -p leds.pcf -o hardware.asc -q hardware.blif
icepack hardware.asc hardware.bin
================================== [SUCCESS] Took 0.72 seconds.
˓→

apios clean

Contents

- apio clean
  - Usage
Usage

apio clean [OPTIONS]

Description

Clean the previous generated files: blif, asc, bin, rpt and out.
Required packages: scons.

Options

-p, --project-dir
Set the target directory for the project.

Examples

1. Clean the leds example

$ apio clean
Removed hardware.blif
Removed hardware.asc
Removed hardware.bin
Removed hardware.out
======================================================================== [SUCCESS] Took 0.17 seconds
========================================================================

apio lint
Usage

apio lint [OPTIONS]

Description

Lint the **verilog** code. It is agnostic of the FPGA. It does not use the **pcf** file.

**Required packages:** scons, verilator.

Options

- **-a, --all**
  Enable all warnings, including code style warnings.

- **-t, --top**
  Set top module.

- **--nostyle**
  Disable all style warnings.

- **--nowarn**
  Disable specific warning(s).

- **--warn**
  Enable specific warning(s).

- **-p, --project-dir**
  Set the target directory for the project.

Examples

1. Lint the **leds example**

   ```bash
   apio lint
   verilator --lint-only -I/path/to/share leds.v
   =========================================================================== [SUCCESS] Took 0.20 seconds
   ```

2. Lint the **leds example** with all the options

   ```bash
   apio lint --all --top leds --nostyle --nowarn PINMISSING,WIDTH --warn DECLFILENAME,
   DEFPARAM
   verilator --lint-only -I/path/to/share -Wall -Wno-style -Wno-PINMISSING -Wno-WIDTH -
   -Wwarn-DECLFILENAME -Wwarn-DEFPARAM --top-module leds leds.v
   =========================================================================== [SUCCESS] Took 0.20 seconds
   ```
apio sim

Contents

- `apio sim`
  - Usage
  - Description
  - Options
  - Examples

Usage

`apio sim [OPTIONS]`

Description

Launch the verilog simulation using GTKWave from a verilog test bench.

Required packages: scons, iverilog.

![GTKWave screenshot]

Note: GTKWave must be installed.
Debian  |  apt-get install gtkwave
Mac OSX |  brew install gtkwave
Windows |  apio install gtkwave

Options

-p, --project-dir
Set the target directory for the project.

Examples

1. Simulate the leds example

```bash
$ apio sim
iverilog -B /path/to/lib/ivl -o leds_tb.out -D VCD_OUTPUT=leds_tb /path/to/vlib/
   -system.v leds.v leds_tb.v
vvp -M /path/to/lib/ivl leds_tb.out
VCD info: dumpfile leds_tb.vcd opened for output.
End of simulation
gtkwave leds_tb.vcd leds_tb.gtkw

GTKWave Analyzer v3.3.66 (w)1999-2015 BSI
[0] start time.
[1000] end time.
WM Destroy
================================== [SUCCESS] Took 1.96 seconds
================================== 
```

apio time

Contents

- apio time
  - Usage
  - Description
  - Options
  - Examples

Usage

```
apio time [OPTIONS]
```
Description

Bitstream timing analysis: generates a \texttt{rpt} file with a topological timing analysis report, from a \texttt{verilog} and a \texttt{pcf} files. Required packages: \texttt{scons, icestorm}.

Options

\texttt{-b, --board}
Select a specific board.

\texttt{--fpga}
Select a specific FPGA.

\texttt{--size --type --pack}
Select a specific FPGA size, type and pack.

\texttt{-p, --project-dir}
Set the target directory for the project.

\texttt{-v, --verbose}
Show the entire output of the command.

\texttt{--verbose-yosys}
Show the yosys output of the command.

\texttt{--verbose-arachne}
Show the arachne output of the command.

Note: All available boards, FPGAs, sizes, types and packs are showed in \texttt{apio boards}

Examples

1. Timing analysis for the \texttt{leds} example

```
$ apio time
[ ] Processing icezum
=============================================
\_\_\_\_\_\_
[\ldots]
\// Reading input .asc file..
\// Reading 1k chipdb file..
\// Creating timing netlist..
\// Timing estimate: 0.24 ns (4161.98 MHz)
=============================================
\_\_\_\_\_\_
\[SUCCESS\] Took 1.10 seconds.
\_\_\_\_\_\_

$ cat hardware.rpt
icetime topological timing analysis report
=============================================
```

(continues on next page)
Warning: This timing analysis report is an estimate!
Info: max_span_hack is enabled: estimate is conservative.

Report for critical path:
------------------------

pre_io_13_11_0 (PRE_IO) [clk] -> PADOUT: 0.240 ns
0.240 ns io_pad_13_11_0_din

Total number of logic levels: 0
Total path delay: 0.24 ns (4161.98 MHz)

apio upload

Contents

- apio upload
  - Usage
  - Description
  - Options
  - Examples

Usage

apio upload [OPTIONS]

Description

Upload the bitstream to the FPGA. It builds the project if required.
It also performs an automatic discovery and validation of the FTDI chip depending on the selected board.
Required packages: scons, system, icestorm.

Note: FTDI driver configuration must be done before upload. More information in apio drivers.

Options

-b, --board
Select a specific board.
--serial-port
Select a specific serial port. You can check the available serial devices with the command `apio system --lsserial`.

`--ftdi-id`  
Select a specific FTDI index. You can check the available FTDI indexes with the command `apio system --lsftdi`. This numerical index is provided by `libftdi1`, that is different from `libftdi0`.

`-s, --sram`  
Perform SRAM programming. Only available for `iceprog` compatible boards.

`-p, --project-dir`  
Set the target directory for the project.

`-v, --verbose`  
Show the entire output of the command.

`--verbose-yosys`  
Show the yosys output of the command.

`--verbose-arachne`  
Show the arachne output of the command.

**Note:** All available boards, FPGAs, sizes, types and packs are showed in `apio boards`.

**Examples**

1. Upload the `leds example`

```bash
$ apio upload
[] Processing icezum
------------------------------------------------------------------------------------------------------------------------
---
[...] iceprog -d i:0x0403:0x6010:0 hardware.bin init..
cdone: high reset..
cdone: low flash ID: 0x20 0xBA 0x16 0x10 0x00 0x00 0x23 0x85 0x32 0x13 0x00 0x54 0x00 0x29.. 0x10 0x06 0x15 0x51 0x62
file size: 32220 erase 64kB sector at 0x00000.. programming.. reading..
VERIFY OK cdone: high Bye.
============================================================= [SUCCESS] Took 1.96 seconds
```
apio Documentation, Release 0.4.0

apio verify

Usage

apio verify [OPTIONS]

Description

Verify the verilog code. It is agnostic of the FPGA. It does not use the pcf file.

Required packages: scons, iverilog.

Options

-p, --project-dir
Set the target directory for the project.

Examples

1. Verify the leds example

   $ apio verify
   iverilog -B /path/to/lib/ivl -o hardware.out -D VCD_OUTPUT= /path/to/vlib/system.v
   -leds.v
   ======================================= SUCCESS Took 0.17 seconds
   =======================================

1.3.4 Setup Commands

apio drivers

Contents

• apio drivers
Usage

apio drivers [OPTIONS]

Description

Enable/Disable the FTDI drivers.

- Linux: add the rules file. It may require a reboot or to uplug and reconnect the board.
- Mac OSX: configure FTDIUSBSerialDriver and AppleUSBFTDI keys and install libftdi.
- Windows: open zadig to replace the current driver by libusbK. It requires to uplug and reconnect the board.

This command requires the driver package (only for Windows).

Note: More information in Install FTDI drivers

Options

--ftdi-enable
Enable FPGA drivers.

--ftdi-disable
Disable FPGA drivers.

--serial-enable
Enable Serial drivers.

--serial-disable
Disable Serial drivers.

Examples

1. Enable the FTDI drivers on Linux

$ apio drivers --ftdi-enable
Configure FTDI drivers for FPGA
[sudo] password for user:
FTDI drivers enabled
Unplug and reconnect your board
2. Disable the FTDI drivers on Linux

```
$ apio drivers --ftdi-disable
Revert FTDI drivers configuration
[sudo] password for user:
FTDI drivers disabled
Unplug and reconnect your board
```

3. Enable the Serial drivers on Linux

```
$ apio drivers --serial-enable
Configure Serial drivers for FPGA
[sudo] password for user:
Serial drivers enabled
Unplug and reconnect your board
```

4. Disable the Serial drivers on Linux

```
$ apio drivers --serial-disable
Revert Serial drivers configuration
[sudo] password for user:
Serial drivers disabled
Unplug and reconnect your board
```

### apio init

#### Contents

- apio init
  - Usage
  - Description
  - Options
  - Examples

#### Usage

```
apio init [OPTIONS]
```

#### Description

Manage apio projects. In addition to the code, an apio project may include a configuration file `apio.ini` and a Scons script `SConstruct`.

#### Options

- `-s`, `--scons`
Create a default SConstruct file. This file can be modified and it will be used instead of the default script.

-\texttt{b, --board}

Create a configuration file with the selected board. This will be the default board used in \texttt{apio build}, \texttt{apio time} and \texttt{apio upload} commands.

-\texttt{p, --project-dir}

Set the target directory for the project.

-\texttt{y, --sayyes}

Automatically answer YES to all the questions.

\textbf{Examples}

1. Create a SConstruct file.

\begin{verbatim}
$ apio init --scons
Creating SConstruct file ...
File 'SConstruct' has been successfully created!
\end{verbatim}

2. Create an \texttt{apio.ini} file with the icezum board

\begin{verbatim}
$ apio init --board icezum
Creating apio.ini file ...
File 'apio.ini' has been successfully created!
\end{verbatim}

\textbf{apio install}

\begin{verbatim}
Contents
  • \texttt{apio install}
    − Usage
    − Description
    − Options
    − Examples
\end{verbatim}

\textbf{Usage}

\texttt{apio install [OPTIONS]}

\textbf{Description}

Install packages. Automatically installs the latest version of the package. Also other versions can be installed using the following notation: \texttt{package@version}.

Available packages
### Package Installation

<table>
<thead>
<tr>
<th>Package</th>
<th>Installation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>drivers</td>
<td>apio install drivers</td>
<td>Drivers tools (only for Windows)</td>
</tr>
<tr>
<td>examples</td>
<td>apio install examples</td>
<td>Verilog basic examples, pinouts, etc</td>
</tr>
<tr>
<td>gtkwave</td>
<td>apio install gtkwave</td>
<td>Simulation viewer. <a href="https://www.gtkwave.org">GTKWave project</a> (only for Windows)</td>
</tr>
<tr>
<td>icestorm</td>
<td>apio install icestorm</td>
<td>iCE40 FPGA synthesis, place &amp; route and configuration tools. <a href="https://github.com/-veri%E5%8D%8A%E5%B2%9B">Icestorm project</a></td>
</tr>
<tr>
<td>iverilog</td>
<td>apio install iverilog</td>
<td>Verilog simulation and synthesis tool. <a href="https://www.icarus.com/">Icarus Verilog project</a></td>
</tr>
<tr>
<td>scons</td>
<td>apio install scons</td>
<td>A software construction tool. <a href="https://scons.org/">Scons project</a></td>
</tr>
<tr>
<td>system</td>
<td>apio install system</td>
<td>Tools for listing the USB devices and retrieving information from the FTDI chips</td>
</tr>
<tr>
<td>verilator</td>
<td>apio install verilator</td>
<td>Verilog HDL simulator. <a href="https://www.verilator.com/">Verilator project</a></td>
</tr>
</tbody>
</table>

### Options

- **-a, --all**
  Install all packages.

- **-l, --list**
  List all available packages.

- **-f, --force**
  Force the packages installation.

- **-p, --platform**
  Set the platform [linux, linux_x86_64, linux_i686, linux_armv7l, linux_aarch64, windows, windows_amd64, windows_x86, darwin] (Advanced).

### Examples

1. **Install system and icestorm packages:**

   ```sh
   $ apio install system icestorm
   Installing system package:
   Download tools-system-linux_x86_64-1.1.0.tar.gz
   Downloading [####################################] 100%
   Unpacking [####################################] 100%
   Package 'system' has been successfully installed!
   Installing icestorm package:
   Download toolchain-icestorm-linux_x86_64-1.11.0.tar.gz
   Downloading [####################################] 100%
   Unpacking [####################################] 100%
   Package 'icestorm' has been successfully installed!
   ```

2. **Install examples package version 0.0.11**

   ```sh
   $ apio install examples@0.0.11
   Installing examples package:
   Download apio-examples-0.0.11.zip
   Downloading [####################################] 100%
   Unpacking [####################################] 100%
   Package 'examples' has been successfully installed!
   ```
3. Show all available packages

```
$ apio install --list

Installed packages:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>examples</td>
<td>Verilog examples</td>
<td>0.0.11</td>
</tr>
<tr>
<td>icestorm</td>
<td>Icestorm toolchain</td>
<td>1.11.0</td>
</tr>
<tr>
<td>system</td>
<td>System tools</td>
<td>1.1.0</td>
</tr>
</tbody>
</table>

Not installed packages:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iverilog</td>
<td>Icarus Verilog toolchain</td>
</tr>
<tr>
<td>scons</td>
<td>Scons tool</td>
</tr>
<tr>
<td>verilator</td>
<td>Verilator toolchain</td>
</tr>
</tbody>
</table>
```

4. Install and update all packages

```
$ apio install --all

Installing examples package:
Already installed. Version 0.0.11
Installing icestorm package:
Already installed. Version 1.11.0
Installing iverilog package:
Download toolchain-iverilog-linux_x86_64-1.2.0.tar.gz
Downloading [####################################] 100%
Unpacking [####################################] 100%
Package 'iverilog' has been successfully installed!
Installing scons package:
Download scons-3.0.1.tar.gz
Downloading [####################################] 100%
Unpacking [####################################] 100%
Package 'scons' has been successfully installed!
Installing system package:
Already installed. Version 1.1.0
Installing verilator package:
Download toolchain-verilator-linux_x86_64-1.0.0.tar.gz
Downloading [####################################] 100%
Unpacking [####################################] 100%
Package 'verilator' has been successfully installed!
```

5. Install the drivers package for windows in a linux platform

```
$ apio install drivers --platform windows

Installing drivers package:
Download tools-drivers-windows-1.1.0.tar.gz
Downloading [####################################] 100%
Unpacking [####################################] 100%
Package 'drivers' has been successfully installed!
```
apio uninstall

## Contents

- apio uninstall
  - Usage
  - Description
  - Options
  - Examples

## Usage

```bash
apio uninstall [OPTIONS]
```

## Description

Uninstall packages. Before uninstalling a package, a confirmation is requested.

### Available packages

<table>
<thead>
<tr>
<th>Package</th>
<th>Installation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>drivers</td>
<td>apio install drivers</td>
<td>Drivers tools (only for Windows)</td>
</tr>
<tr>
<td>examples</td>
<td>apio install examples</td>
<td>Verilog basic examples, pinouts, etc</td>
</tr>
<tr>
<td>gtkwave</td>
<td>apio install gtkwave</td>
<td>Simulation viewer. GTKWave project (only for Windows)</td>
</tr>
<tr>
<td>icestorm</td>
<td>apio install icestorm</td>
<td>iCE40 FPGA synthesis, place &amp; route and configuration tools. Icestorm project</td>
</tr>
<tr>
<td>iverilog</td>
<td>apio install iverilog</td>
<td>Verilog simulation and synthesis tool. Icarus Verilog project</td>
</tr>
<tr>
<td>scons</td>
<td>apio install scons</td>
<td>A software construction tool. Scons project</td>
</tr>
<tr>
<td>system</td>
<td>apio install system</td>
<td>Tools for listing the USB devices and retrieving information from the FTDI chips</td>
</tr>
<tr>
<td>verilator</td>
<td>apio install verilator</td>
<td>Verilog HDL simulator. Verilator project</td>
</tr>
</tbody>
</table>

## Options

- `-a`, `--all`
  Uninstall all packages.

- `-l`, `--list`
  List all installed packages.

- `-p`, `--platform`
  Set the platform `[linux_x86_64, linux_i686, linux_armv7l, linux_aarch64, windows, darwin]` (Advanced).
Examples

1. Uninstall examples package

```bash
$ apio uninstall examples
Do you want to continue? [y/N]: y
Uninstalling examples package:
Package 'examples' has been successfully uninstalled!
```

2. Uninstall the drivers package for windows in a linux platform

```bash
$ apio uninstall drivers --platform windows
Do you want to continue? [y/N]: y
Uninstalling drivers package:
Package 'drivers' has been successfully uninstalled!
```

1.3.5 Utility Commands

apio boards

Contents

- apio boards
  - Usage
  - Description
  - Options
  - Examples

Usage

```
apio boards [OPTIONS]
```

Description

Show FPGA boards information.

All supported boards:

HX1K
- IceZUM Alhambra
- Nandland Go board
- iCEstick Evaluation Kit

HX8K
- Alhambra II

1.3. User Guide 25
• BlackIce
• BlackIce II
• CAT board
• icoBOARD 1.0
• Kéfir I
• icle40-hx8k breakout board

LP8K
• TinyFPGA B2
• TinyFPGA BX

UP5K
• UPduino v1.0
• UPduino v2.0
• iclebreaker
• iclebreaker bitsy
• FPGA 101 workshop badge board
• icle40 UltraPlus breakout board

Note: All supported FPGAs are shown in Project IceStorm web page

Options

-l, --list
List all supported boards.

-f, --fpga
List all supported FPGAs.

Examples

1. Show all available boards

$ apio boards --list

Supported boards:

<table>
<thead>
<tr>
<th>Board</th>
<th>FPGA</th>
<th>Type</th>
<th>Size</th>
<th>Pack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cat-board</td>
<td>ice40-hx8k-ct256</td>
<td>hx</td>
<td>8k</td>
<td>ct256</td>
</tr>
<tr>
<td>TinyFPGA-B2</td>
<td>ice40-lp8k-cm81</td>
<td>lp</td>
<td>8k</td>
<td>cm81</td>
</tr>
<tr>
<td>TinyFPGA-BX</td>
<td>ice40-lp8k-cm81</td>
<td>lp</td>
<td>8k</td>
<td>cm81</td>
</tr>
<tr>
<td>alhambra-ii</td>
<td>ice40-hx4k-tq144</td>
<td>hx</td>
<td>8k</td>
<td>tq144:4k</td>
</tr>
<tr>
<td>blackice</td>
<td>ice40-hx4k-tq144</td>
<td>hx</td>
<td>8k</td>
<td>tq144:4k</td>
</tr>
</tbody>
</table>
2. Show all available FPGAs

```
$ apio boards --fpga
```

Supported FPGAs:

<table>
<thead>
<tr>
<th>FPGA Type</th>
<th>Size</th>
<th>Pack</th>
</tr>
</thead>
<tbody>
<tr>
<td>iCE40-HX1K-CB132</td>
<td>hx</td>
<td>1k</td>
</tr>
<tr>
<td>iCE40-HX1K-TQ144</td>
<td>hx</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-HX1K-VQ100</td>
<td>vx</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-HX4K-BG121</td>
<td>hx</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-HX4K-CB132</td>
<td>hx</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-HX4K-TQ144</td>
<td>hx</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-HX8K-BG121</td>
<td>hx</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-HX8K-CB132</td>
<td>hx</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-HX8K-CM225</td>
<td>hx</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-HX8K-CT256</td>
<td>hx</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-LP1K-CB121</td>
<td>lp</td>
<td>1k</td>
</tr>
<tr>
<td>iCE40-LP1K-CB81</td>
<td>lp</td>
<td>1k</td>
</tr>
<tr>
<td>iCE40-LP1K-CM121</td>
<td>lp</td>
<td>1k</td>
</tr>
<tr>
<td>iCE40-LP1K-CM36</td>
<td>lp</td>
<td>1k</td>
</tr>
<tr>
<td>iCE40-LP1K-CM49</td>
<td>lp</td>
<td>1k</td>
</tr>
<tr>
<td>iCE40-LP1K-CM81</td>
<td>lp</td>
<td>1k</td>
</tr>
<tr>
<td>iCE40-LP1K-QN84</td>
<td>lp</td>
<td>1k</td>
</tr>
<tr>
<td>iCE40-LP1K-SWG16TR</td>
<td>lp</td>
<td>1k</td>
</tr>
<tr>
<td>iCE40-LP384-CM36</td>
<td>lp</td>
<td>384</td>
</tr>
<tr>
<td>iCE40-LP384-CM49</td>
<td>lp</td>
<td>384</td>
</tr>
<tr>
<td>iCE40-LP384-QN32</td>
<td>lp</td>
<td>384</td>
</tr>
<tr>
<td>iCE40-LP4K-CM121</td>
<td>lp</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-LP4K-CM225</td>
<td>lp</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-LP4K-CM81</td>
<td>lp</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-LP8K-CM121</td>
<td>lp</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-LP8K-CM225</td>
<td>lp</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-LP8K-CM81</td>
<td>lp</td>
<td>8k</td>
</tr>
<tr>
<td>iCE40-UP3K-UWG30</td>
<td>up</td>
<td>5k</td>
</tr>
<tr>
<td>iCE40-UP5K-SG48</td>
<td>up</td>
<td>5k</td>
</tr>
<tr>
<td>iCE40-UP5K-UWG30</td>
<td>up</td>
<td>5k</td>
</tr>
</tbody>
</table>
## Usage

`apio config [OPTIONS]`

## Description

Apio configuration commands.

## Options

- `-l, --list`
  List all configuration parameters.

- `-v, --verbose [0|1]`
  Verbose mode: 0 General, 1 Information.

- `-e, --exe [default|native]`
  Configure executables: `default` selects apio packages, `native` selects native binaries (except system package).

**Note:** In **debian** systems, if `/etc/apio.json` defines a new `APIO_PKG_DIR`, this new path will be used to load the packages.

<table>
<thead>
<tr>
<th>Mode</th>
<th>default</th>
<th>native</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/etc/apio.json</code></td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Load installed packages</td>
<td>Yes</td>
<td>Yes *</td>
</tr>
<tr>
<td>Check installed packages</td>
<td>Yes</td>
<td>Yes **</td>
</tr>
</tbody>
</table>

* load `APIO_PKG_DIR` from `/etc/apio.json`

** Suggest message `apt-get install apio-[pkg]`
Examples

1. Show all configuration parameters

$ apio config --list
Executable mode: default
Verbose mode: 0

2. Enable native mode for executable binaries

$ apio config --exe native
Executable mode updated: native

3. Enable verbose mode 1

$ apio config --verbose 1
Verbose mode updated: 1

apiaio examples

Contents

- apio examples
  - Usage
  - Description
  - Options
  - Examples

Usage

`apio examples [OPTIONS]`

Description

Manage verilog examples: https://github.com/FPGAwars/apio-examples
This command requires the examples package.

Options

- `-l, --list`
List all available examples.
- `-d, --dir`
Copy the selected example directory.
-f, --files
Copy the selected example files.

-p, --project-dir
Set the target directory for the examples.

-n, --sayno
Automatically answer NO to all the questions.

Examples

1. Show all available examples

```bash
$ apio examples --list
[ ... ]
leds
-------------------------------------------------------------
 Vulcan
Verilog example for Turning all the leds on (for the icestick/icezum boards)
wire
-------------------------------------------------------------
 Vulcan
Verilog example on how to describe a simple wire
[ ... ]
```

2. Copy the `leds` example files

```bash
$ apio examples --files leds
Copying leds example files ...
Example files 'leds' have been successfully created!

$ ls
leds.pcf  leds_tb.gtkw  leds_tb.v  leds.v
```

3. Copy the `leds` example directory

```bash
$ apio examples --dir leds
Creating leds directory ...
Example 'leds' has been successfully created!

$ tree leds
leds
    info
    leds.pcf
    leds_tb.gtkw
    leds_tb.v
    leds.v
```

```
Contents

- apio raw
  - Usage
  - Description
  - Argument
  - Examples

Usage

```plaintext
apio raw "[CMD]"
apio raw ‘[CMD]’
```

Description

Execute commands using Apio packages.

Argument

cmd

Command to be executed using installed Apio packages.

Examples

1. Run yosys (package installed with Apio)

```
$ apio raw 'yosys'
```

2. Generate a verilog diagram with yosys

```
$ apio raw 'yosys -p "read_verilog leds.v; show" -q'
```

1.3. User Guide 31
apio system

Contents

- apio system
  - Usage
  - Description
  - Options
  - Examples
Usage

apio system [OPTIONS]

Description

System tools: https://github.com/FPGAwars/tools-system
This command requires the system package.

Options

--lsftdi
List all connected FTDI devices.

--lsusb
List all connected USB devices.

--lsserial
List all connected Serial devices.

-i, --info
Show system information.

Examples

1. List connected FTDI devices

$ apio system --lsftdi
Number of FTDI devices found: 1
Checking device: 0
Manufacturer: Mareldem, Description: IceZUM Alhambra v1.1 - B01-020

2. List connected USB devices

$ apio system --lsusb
1d6b:0003 (bus 3, device 1)
04ca:7049 (bus 2, device 4) path: 8
8087:0a2a (bus 2, device 3) path: 7
138a:0017 (bus 2, device 2) path: 6
0403:6010 (bus 2, device 69) path: 2
1d6b:0002 (bus 2, device 1)
8087:8001 (bus 1, device 2) path: 1
1d6b:0002 (bus 1, device 1)

3. List connected Serial devices

$ apio system --lsserial
Number of Serial devices found: 2
/dev/ttyUSB1

(continues on next page)
4. Show system information

$ apio system --info
Platform: linux_x86_64

apio upgrade

Contents

- apio upgrade
  - Usage
  - Description
  - Examples

Usage

apio upgrade [OPTIONS]

Description

Check latest Apio version in https://pypi.python.org/pypi/apio.

Examples

1. Check the Apio version

$ apio upgrade
You're up-to-date!
Apio 0.4.0 is currently the newest version available.

1.4 Contribute

1.4.1 Support a new board

In order to support a new board based on FPGA Lattice iCE40 family, follow these steps:

1. Find your FPGA name in fpgas.json. This file contains all FPGAs supported by the Icestorm project.
2. Find or add your programmer in `programmers.json`.

```
"iceprog": {
    "command": "iceprog",
    "args": "-d 1:0x${VID}:0x${PID}:${FTDI_ID}"
}
```

```
"icoprog": {
    "command": "export WIRINGPI_GPIOMEM=1; icoprog",
    "args": "-p "
}
```

```
"tinyprog": {
    "command": "tinyprog",
    "args": "--pyserial -c ${SERIAL_PORT} --program",
    "pip_packages": [ "tinyprog" ]
}
```

NOTE: if your programmer uses a python package, add this package and its version range to `distribution.json`.

```
"pip_packages": {
    "blackiceprog": ">=2.0.0,<3.0.0",
    "litterbox": ">=0.2.1,<0.3.0",
    "tinyfpgab": ">=1.1.0,<1.2.0",
    "tinyprog": ">=1.0.21,<1.1.0"
}
```

3. Add your board to `boards.json` with the following format:

```
"icezum": {
    "name": "IceZUM Alhambra",
    "fpga": "iCE40-HX1K-TQ144",
    "programmer": { 
        "type": "iceprog"
    },
    "usb": { 
        "vid": "0403",
        "push": "0x01"
    }
}
```

(continues on next page)
"icoboard": {
  "name": "icoBOARD 1.0",
  "fpga": "iCE40-HX8K-CT256",
  "programmer": {
    "type": "icoprog"
  },
  "platform": "linux_armv7l"
}

"TinyFPGA-BX": {
  "name": "TinyFPGA BX",
  "fpga": "iCE40-LP8K-CM81",
  "programmer": {
    "type": "tinyprog"
  },
  "usb": {
    "vid": "1d50",
    "pid": "6130"
  },
  "tinyprog": {
    "desc": "TinyFPGA BX"
  }
}
Symbols

-fpga
  apio-build command line option, 10
  apio-time command line option, 15
-ftdi-disable
  apio-drivers command line option, 19
-ftdi-enable
  apio-drivers command line option, 19
-ftdi-id
  apio-upload command line option, 17
-lsftdi
  apio-system command line option, 33
-lsserial
  apio-system command line option, 33
-lsusb
  apio-system command line option, 33
-nostyle
  command line option, 12
-nowarn
  command line option, 12
-serial-disable
  apio-drivers command line option, 19
-serial-enable
  apio-drivers command line option, 19
-serial-port
  apio-upload command line option, 16
-size -type -pack
  apio-build command line option, 10
  apio-time command line option, 15
-verbose-arachne
  apio-build command line option, 10
  apio-time command line option, 15
  apio-upload command line option, 17
-verbose-yosys
  apio-build command line option, 10
  apio-time command line option, 15
  apio-upload command line option, 17
-version
  apio command line option, 9
  -warn
  apio command line option, 12
-a, -all
  apio-install command line option, 22
  apio-uninstall command line option, 24
  command line option, 12
-b, -board
  apio-build command line option, 10
  apio-init command line option, 21
  apio-time command line option, 15
  apio-upload command line option, 16
-d, -dir
  apio-examples command line option, 29
-e, -exe [default|native]
  apio-config command line option, 28
-f, -files
  apio-examples command line option, 29
-f, -force
  apio-install command line option, 22
-f, -fpga
  apio-boards command line option, 26
-i, -info
  apio-system command line option, 33
-l, -list
  apio-boards command line option, 26
  apio-config command line option, 28
  apio-examples command line option, 29
  apio-install command line option, 22
  apio-uninstall command line option, 24
-n, -sayno
  apio-examples command line option, 30
-p, -platform
  apio-install command line option, 22