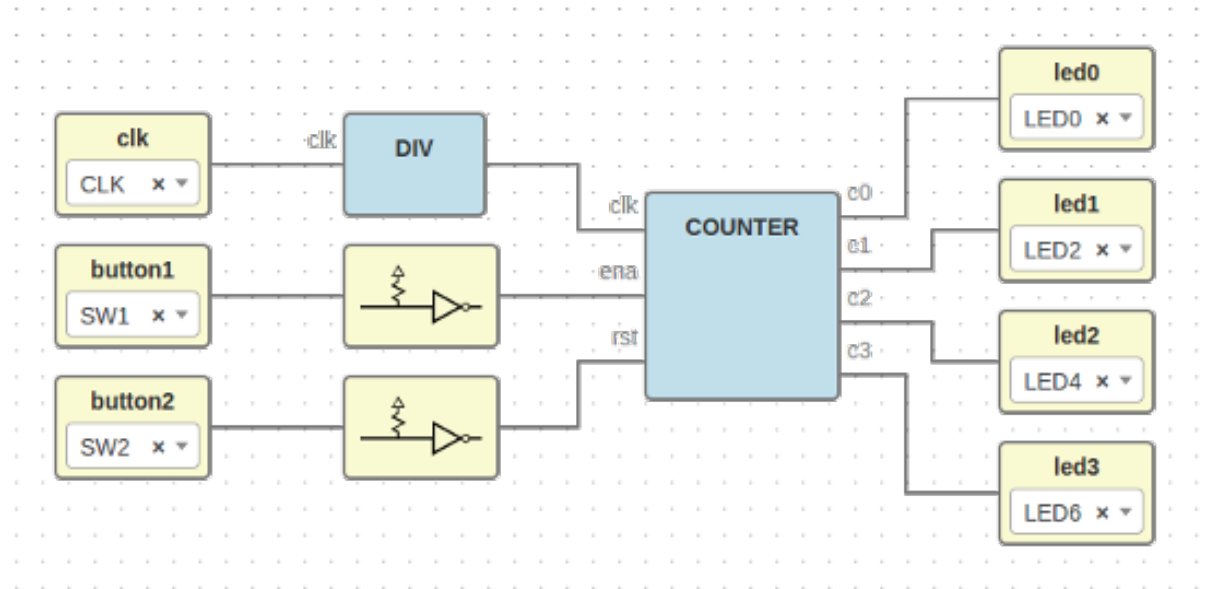


Workshop - Open FPGA tools



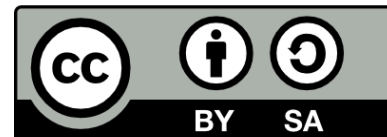
Jesús Arroyo Torrens

<https://github.com/Jesus89>



OSHWDem 16
November 5, 2016
Museo Domus, A Coruña

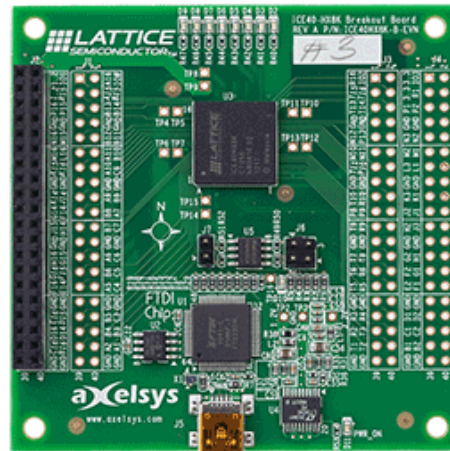
<https://github.com/FPGAwards/workshops>



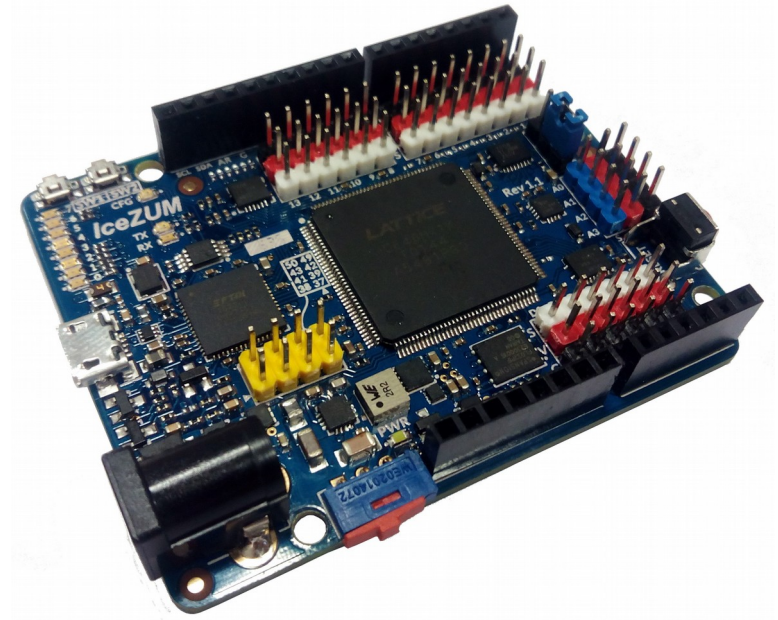
Open FPGA boards



[IceStick](#)



[iCE40-HX8K Breakout Board](#)



[IceZUM Alhambra](#)

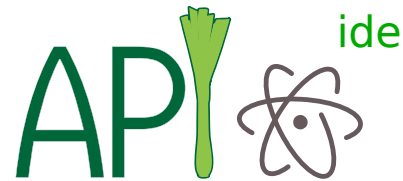
Open FPGA toolchains

Icestorm

Iverilog

GTKWave

Open FPGA toolchains

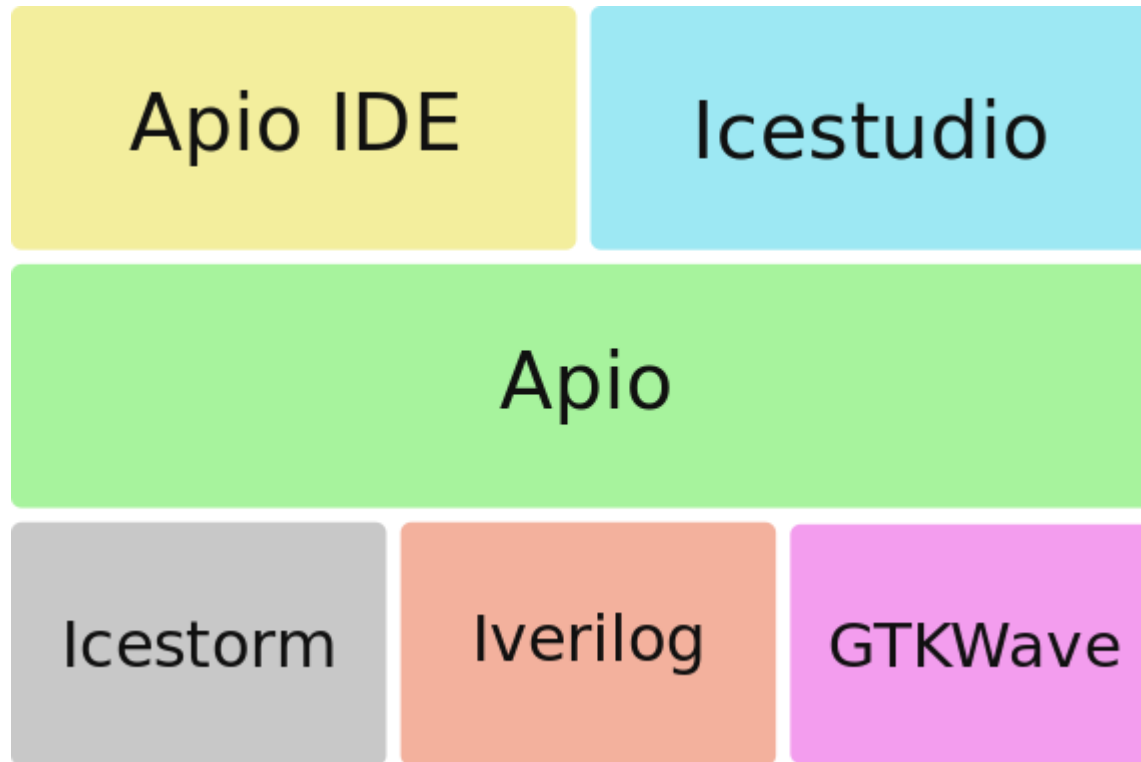


Icestorm

Iverilog

GTKWave

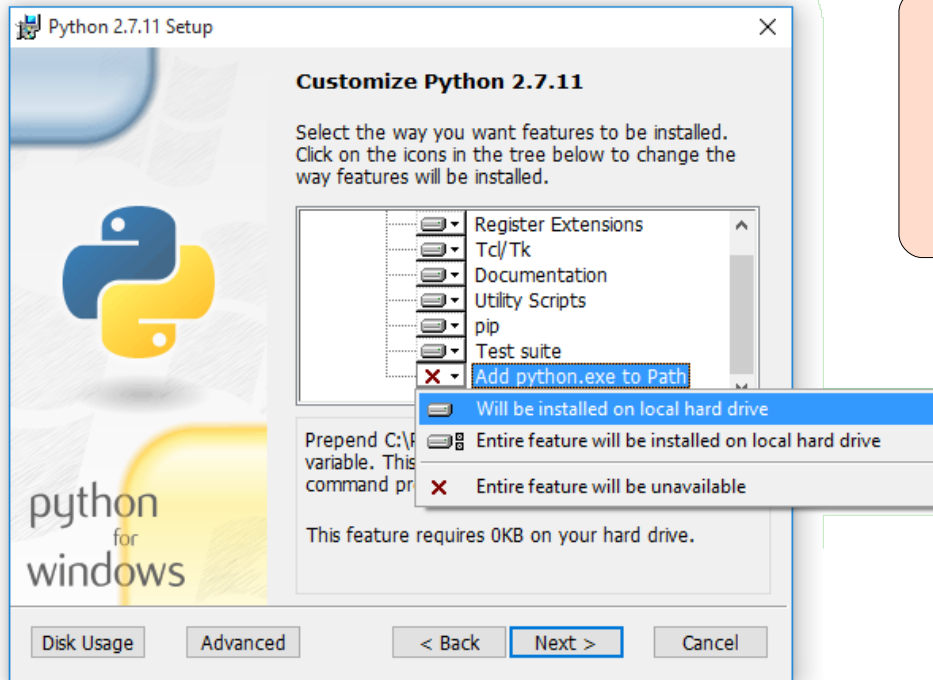
Open FPGA stack tools



Requirements

1. Python 2.7

<https://www.python.org>



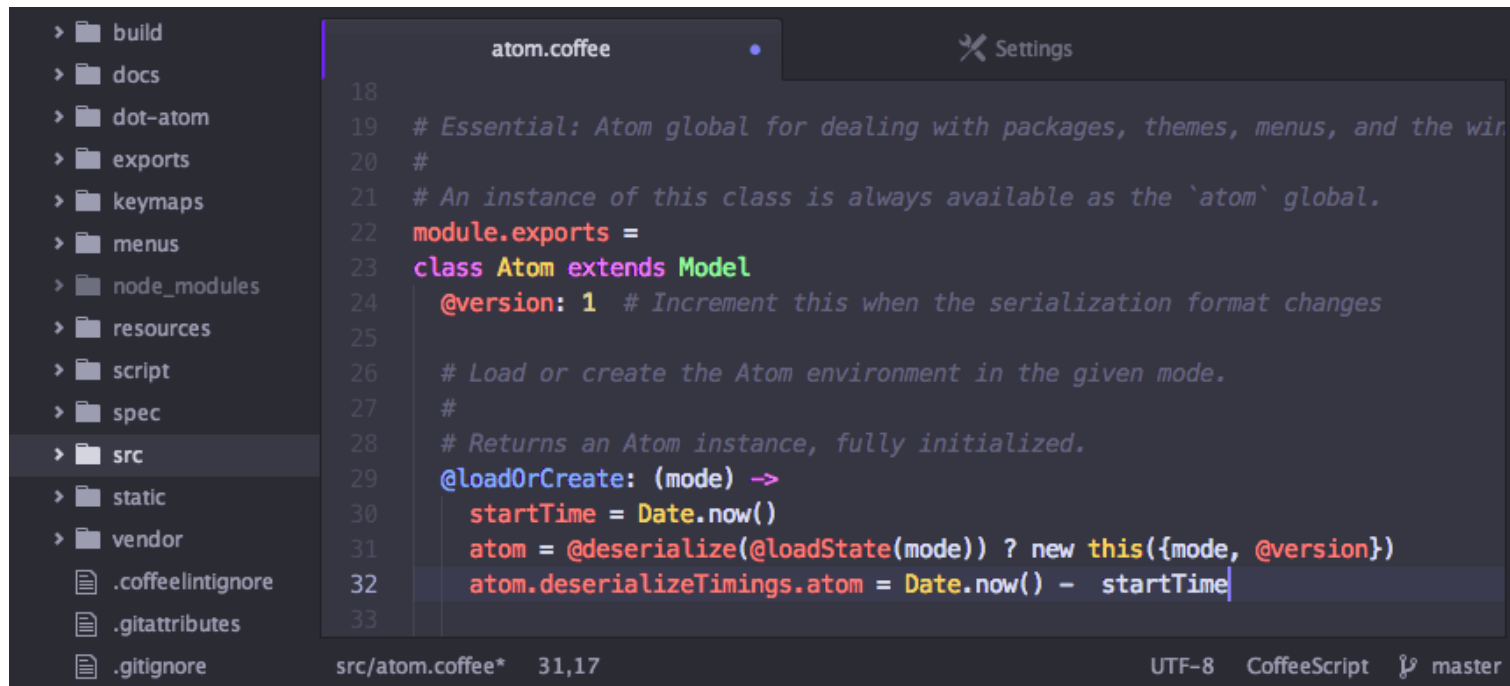
*Windows Users: DON'T FORGET to select **Add python.exe to Path** feature on the "Customize" stage.*

Check installation: open a console and type **python**

Requirements

2. Atom Editor

<https://atom.io>



```
18
19 # Essential: Atom global for dealing with packages, themes, menus, and the win
20 #
21 # An instance of this class is always available as the `atom` global.
22 module.exports =
23 class Atom extends Model
24   @version: 1 # Increment this when the serialization format changes
25
26   # Load or create the Atom environment in the given mode.
27   #
28   # Returns an Atom instance, fully initialized.
29   @loadOrCreate: (mode) ->
30     startTime = Date.now()
31     atom = @deserialize(@loadState(mode)) ? new this({mode, @version})
32     atom.deserializeTimings.atom = Date.now() - startTime
33
```

src/atom.coffee* 31,17 UTF-8 CoffeeScript master

Requirements

3. Apio packages*

<https://github.com/FPGAwars/workshops/releases>

- Copy *apio-dir.zip*
- Copy *install.py*
- Open console and execute

```
$ python install.py
```

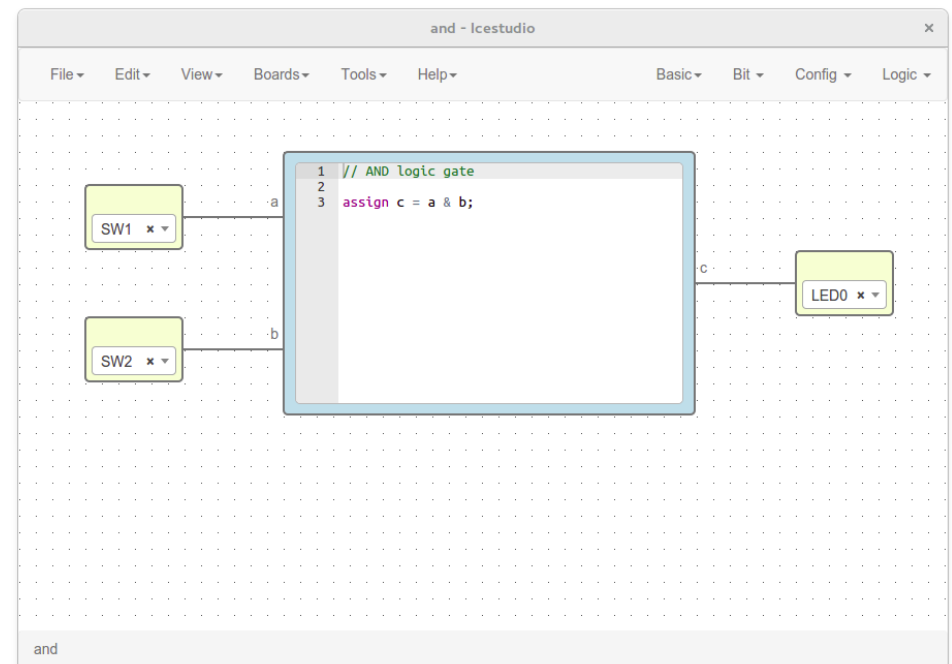
```
Insert the zip filename: apio-dir-linux64.zip
```

```
Success: .apio dir updated!
```

* (optional) equivalent to *Icestudio Install toolchain* or `apio install -all`.
It is used to save time downloading the packages

Icestudio

<https://github.com/FPGAwards/icestudio>



Experimental graphic editor for open FPGAs. Created with HTML and JS

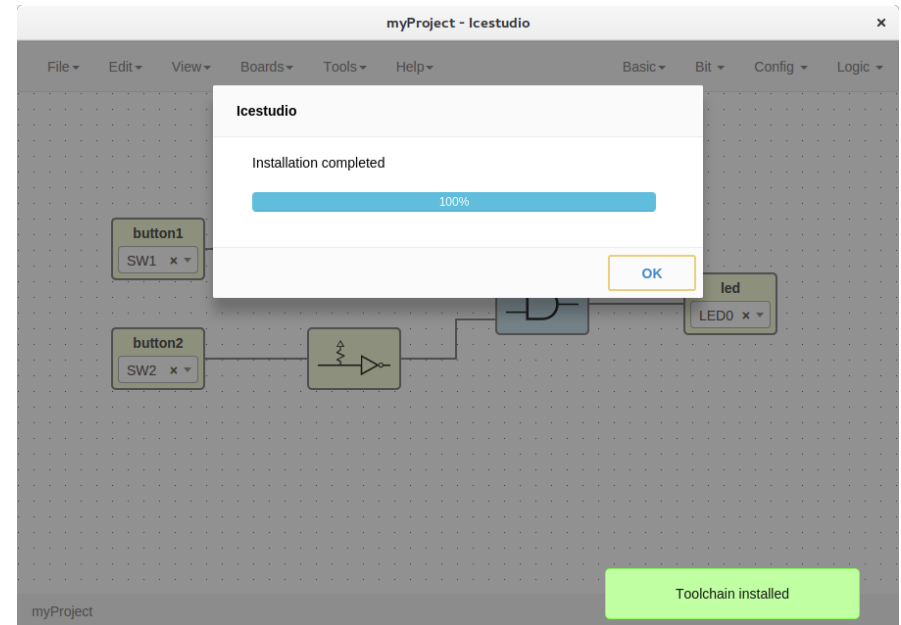
Icestudio

1. Install

- Copy *icestudio-0.2.2-rc.zip*
- Unzip the file
- Execute *icestudio*

2. Setup

- Install toolchain
Tools > Install/Upgrade toolchain
- Install drivers
Tools > Enable drivers

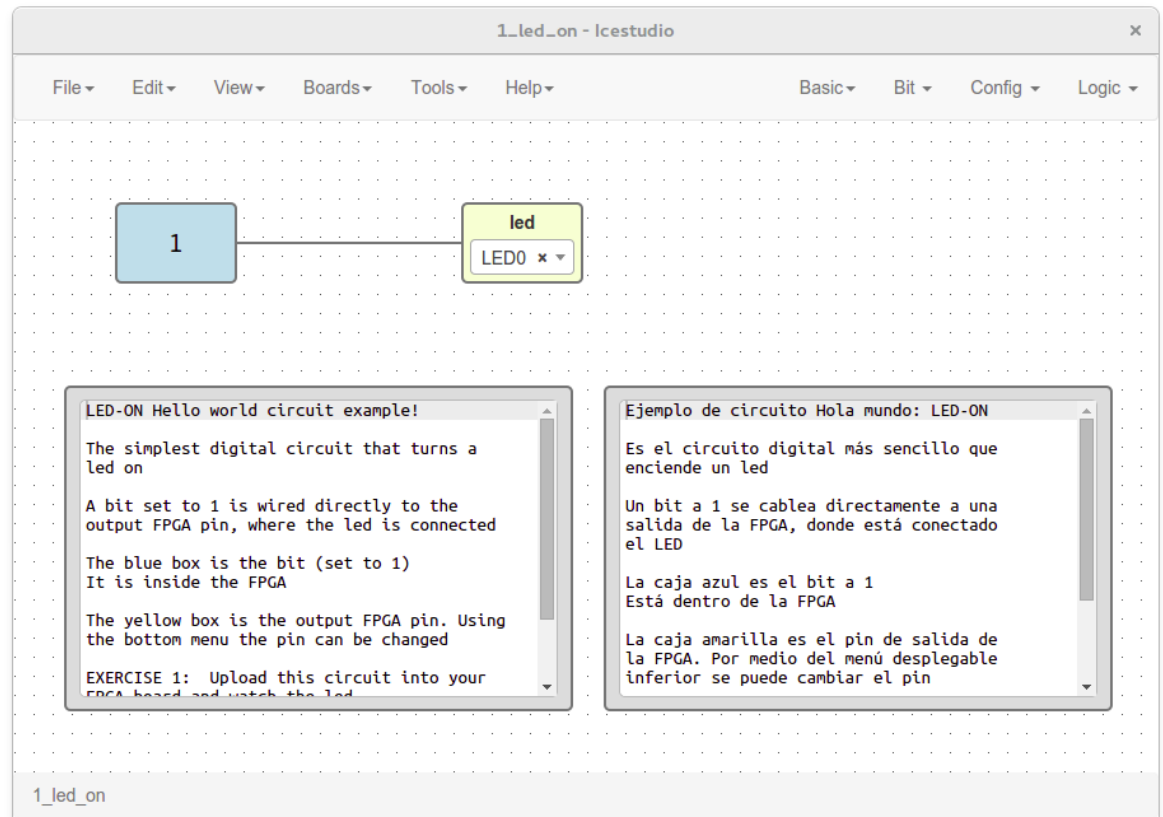


Drivers configuration requires administrator permissions
Follow the instructions in each OS

Icestudio

3. Hello, world!

- Load example
File > Examples
- Select board
Board > IceZUM
- Select I/O pin
Edit the combo
- Upload bit stream
Tools > Upload

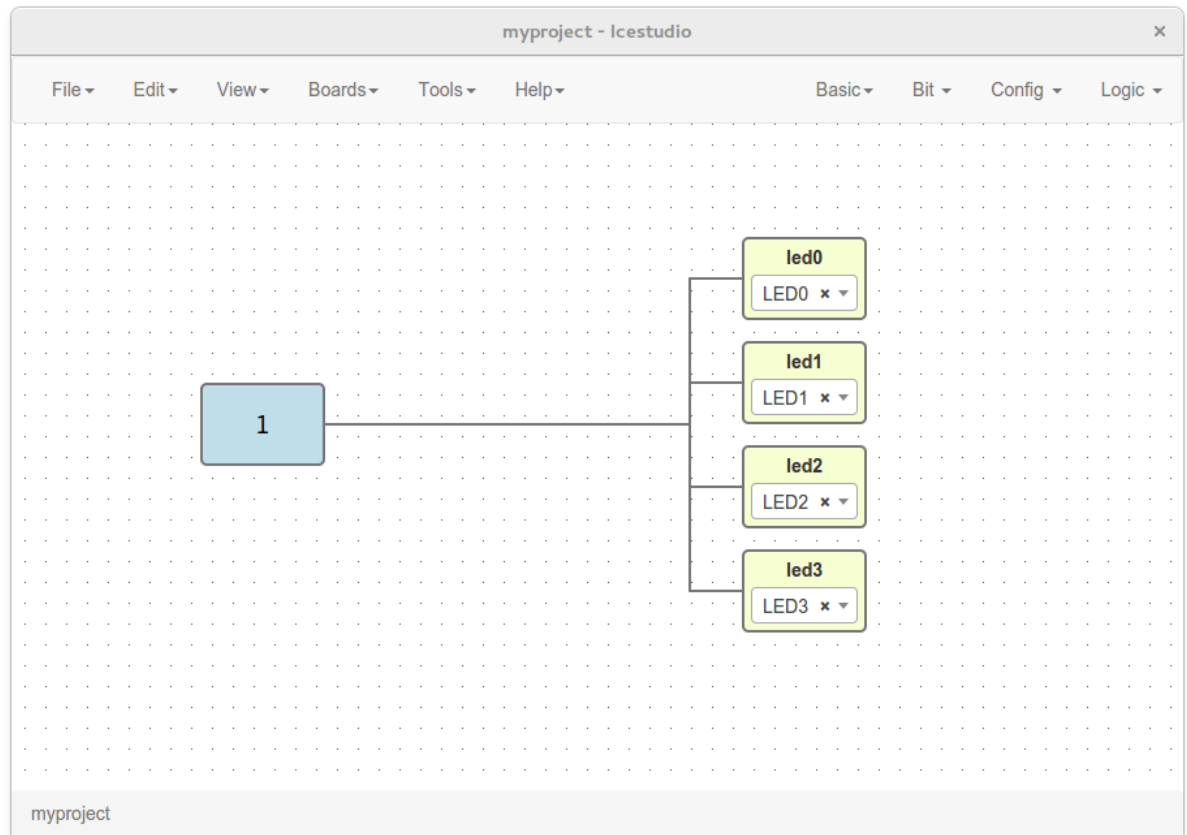


Inspect a block and edit a I/O label using *double-click*

Icestudio

4. More leds on

- Create a project
File > New project
- Add blocks
Bit > 1
Basic > Output
- Connect wires
- Upload bit stream
Tools > Upload

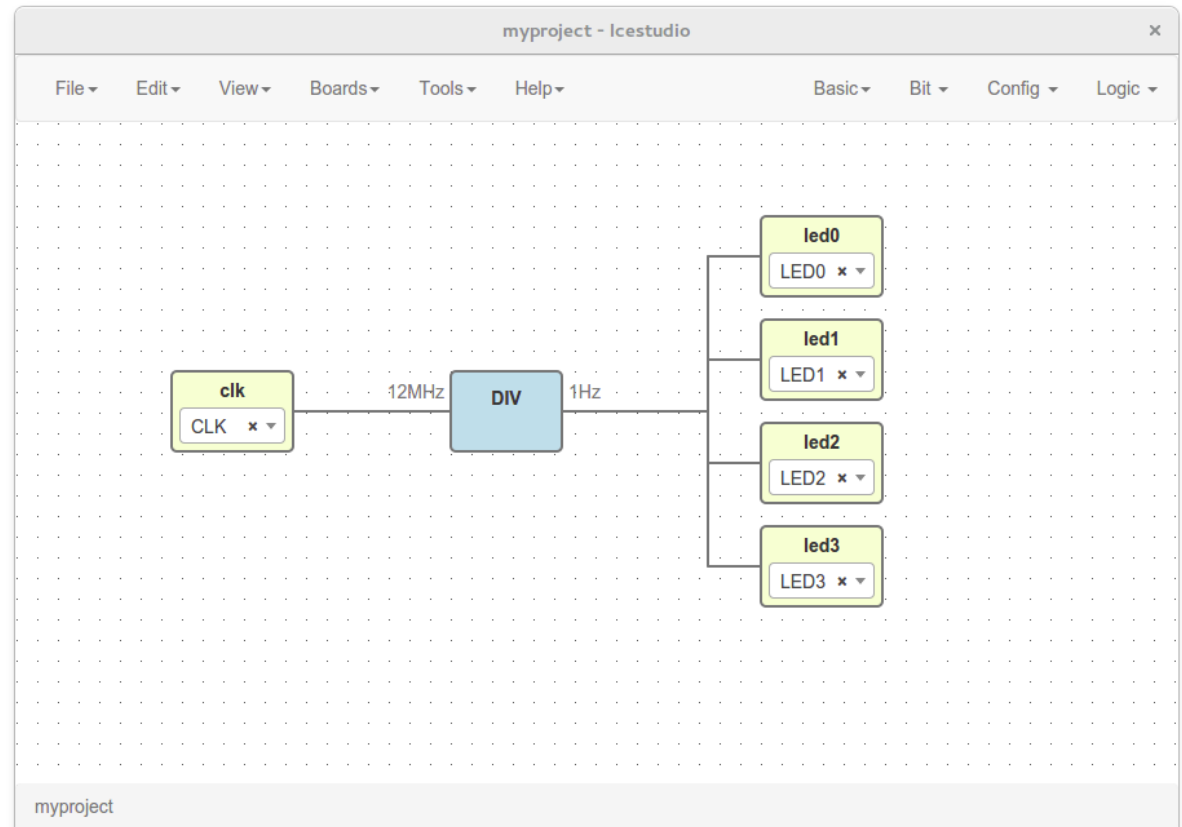


Multiple I/O blocks can be created e.g. "*led0_led1_led2*"

Icestudio

5. Blink

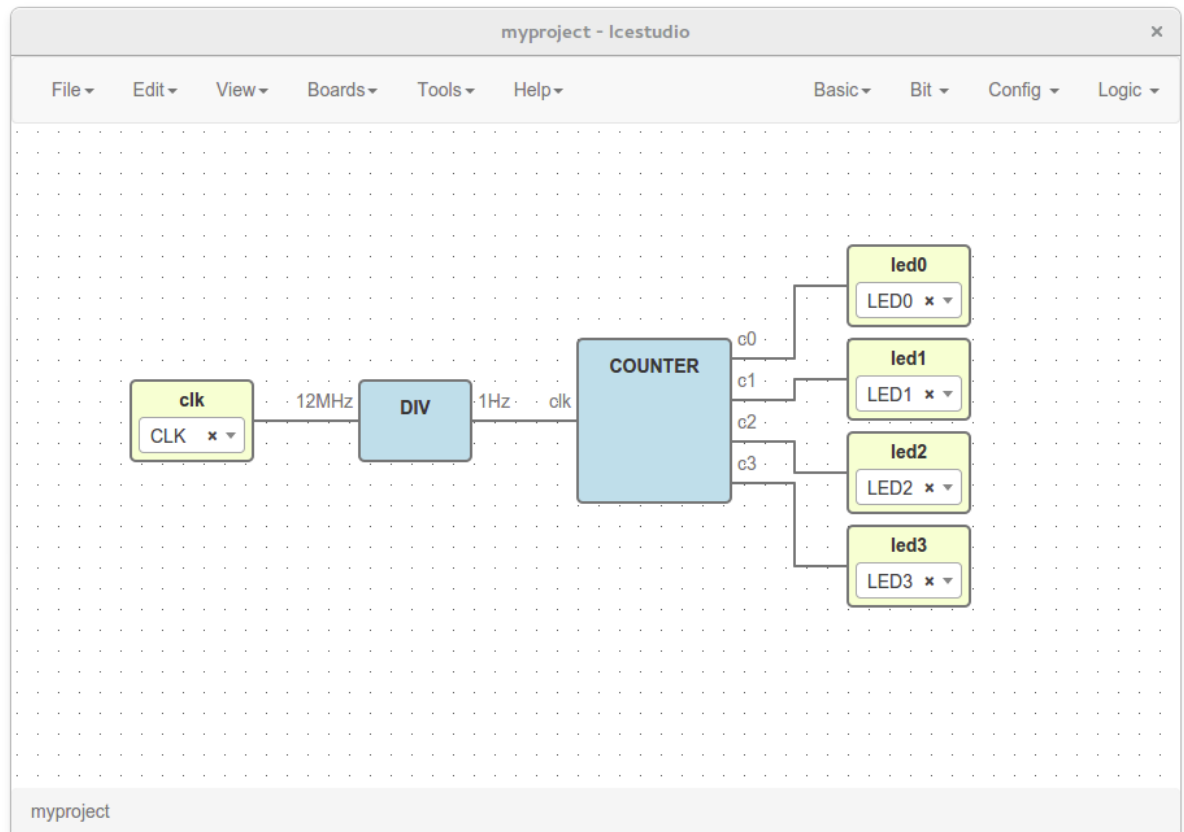
- Import DIV block
File > Import block
- Add clock input
Basic > Input
- Connect wires
- Upload bit stream
Tools > Upload



Icestudio

5. Counter

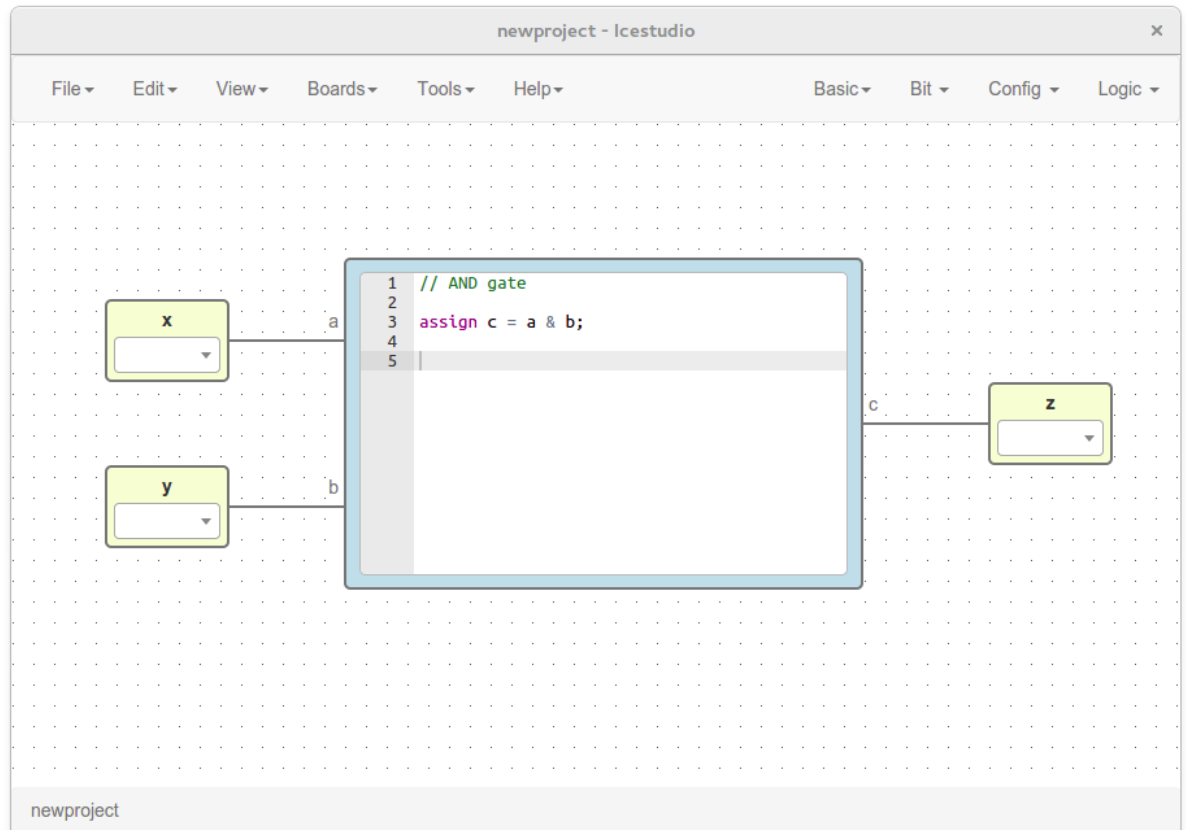
- Import counter
File > Import block
- Reconnect wires
- Upload Bit stream
Tools > Upload



Icestudio

6. Let's code

- Create a project
File > New project
- Add blocks
Basic > Code
Basic > Input
Basic > Output
- Connect wires
- Verify the design
Tools > Verify

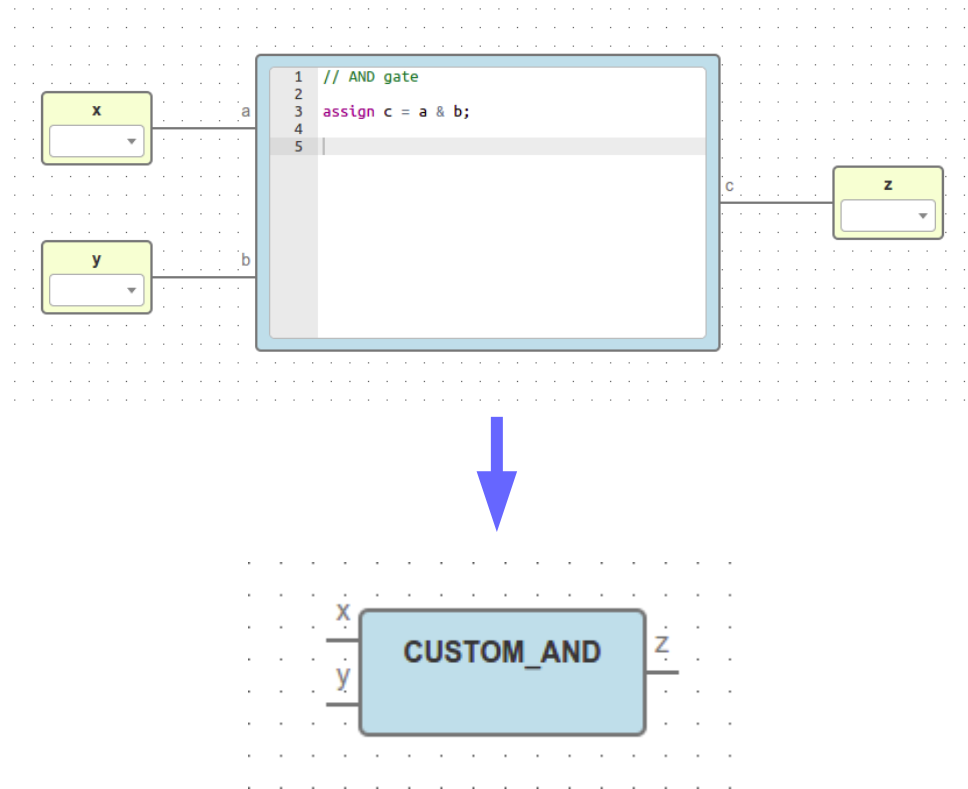


Code block ports can be created "in_out" e.g. "a,b,c"

Icestudio

7. Custom block

- Load a project
File > Open project
- Save project as block
File > Export as block



Input / Output pins will become in/out block ports

Apio

<https://github.com/FPGAwards/apio>



```
Terminal
jesus@ThinkPad ~
$ apio
Usage: apio [OPTIONS] COMMAND [ARGS]...

Experimental micro-ecosystem for open FPGAs

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

Code commands:
  build      Synthesize the bitstream.
  clean      Clean the previous generated files.
  sim        Launch the verilog simulation.
  time       Bitstream timing analysis.
  upload     Upload the bitstream to the FPGA.
  verify     Verify the verilog code.

Environment commands:
  boards     Manage FPGA boards.
  config     Apio configuration.
  drivers    Manage FPGA drivers.
  examples   Manage verilog examples.
  init       Manage apio projects.
  install    Install packages.
  system     System tools.
  uninstall  Uninstall packages.
  upgrade    Check the latest Apio version.

jesus@ThinkPad ~
$
```

Experimental open source ecosystem for open FPGAs. Created with Python

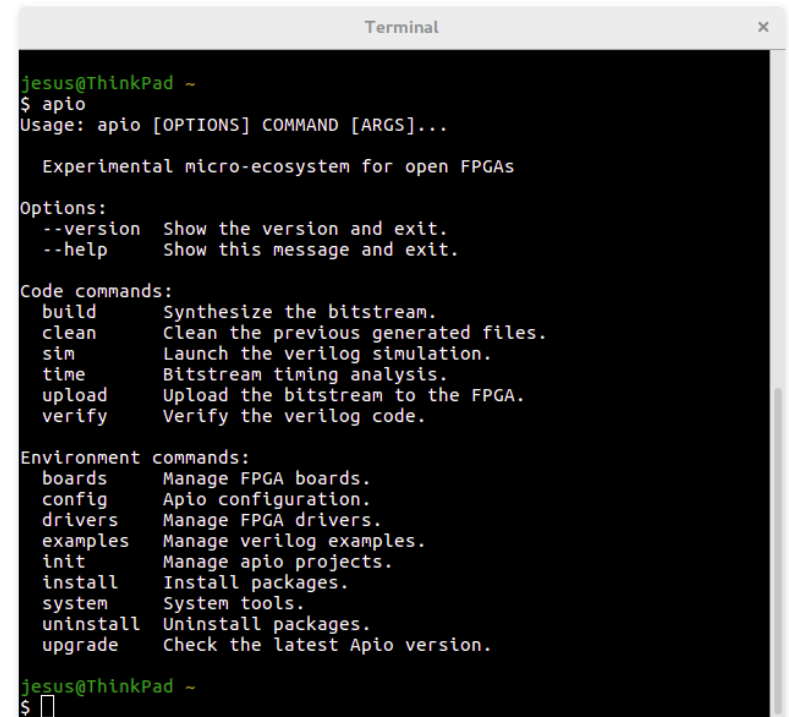
Apio

1. Install

- Open the console and execute
`$ pip install -U apio`
- Check apio
`$ apio`

2. Setup

- Install toolchain
`$ apio install --all`
- Install drivers
`$ apio drivers --enable`



```
Terminal
jesus@ThinkPad ~
$ apio
Usage: apio [OPTIONS] COMMAND [ARGS]...

  Experimental micro-ecosystem for open FPGAs

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

Code commands:
  build      Synthesize the bitstream.
  clean      Clean the previous generated files.
  sim        Launch the verilog simulation.
  time       Bitstream timing analysis.
  upload     Upload the bitstream to the FPGA.
  verify     Verify the verilog code.

Environment commands:
  boards     Manage FPGA boards.
  config     Apio configuration.
  drivers     Manage FPGA drivers.
  examples   Manage verilog examples.
  init       Manage apio projects.
  install    Install packages.
  system     System tools.
  uninstall  Uninstall packages.
  upgrade    Check the latest Apio version.

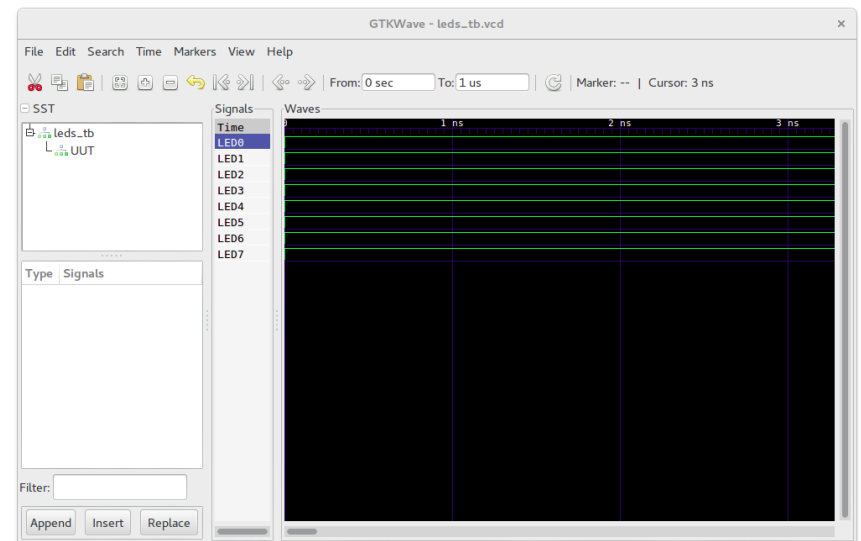
jesus@ThinkPad ~
$
```

Drivers configuration requires administrator permissions
Follow the instructions in each OS

Apio

3. Hello, world!

- Load example
`$ apio examples -d icezum/leds`
- Move to example
`$ cd icezum/leds`
- Verify and simulate
`$ apio verify $ apio sim`
- Build and upload
`$ apio build $ apio upload`
- Time analysis and clean
`$ apio time $ apio clean`



Apio IDE

<https://github.com/FPGAwards/apio-ide>



The screenshot shows the Apio IDE interface with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'leds' with files: .sconsign.dblite, apio.ini, info, leds_tb.gtkw, leds_tb.v, leds.pcf, and leds.v. The code editor displays the contents of 'leds.v', which is a Verilog module for 8 LEDs. The status bar at the bottom indicates 'ApioBuild leds.v 1:1' and '2 updates'.

```
leds.v
1 // *****
2 // *****
3 // *****
4 // *****
5 // *****
6 module leds(output wire LED0,
7             output wire LED1,
8             output wire LED2,
9             output wire LED3,
10            output wire LED4,
11            output wire LED5,
12            output wire LED6,
13            output wire LED7);
14
15    assign LED0 = 1'b1;
16    assign LED1 = 1'b1;
17    assign LED2 = 1'b1;
18    assign LED3 = 1'b1;
19    assign LED4 = 1'b1;
20    assign LED5 = 1'b1;
21    assign LED6 = 1'b1;
22    assign LED7 = 1'b1;
23
24 endmodule
25
```

Experimental open source ecosystem for open FPGAs. Created with Python

Apio IDE

1. Install

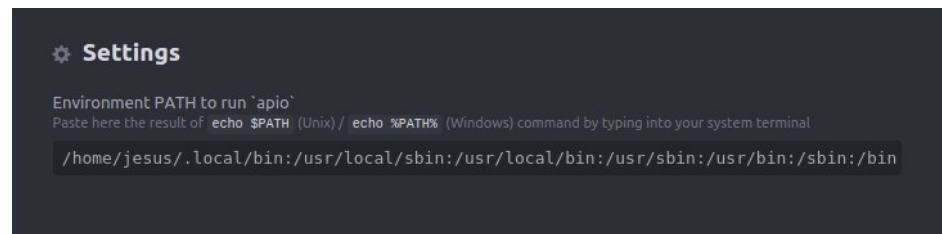
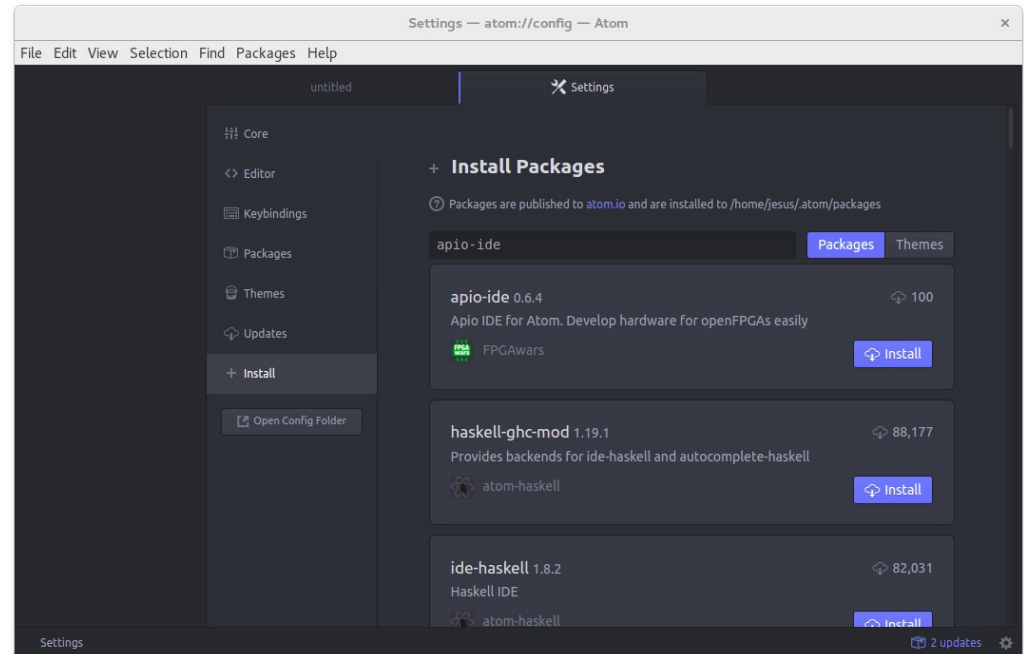
- Open Atom
- Edit > Preferences
- + Install: apio-ide
- Retart Atom

2. Setup

- Sometimes is required to add

apio-ide > Settings > Env PATH

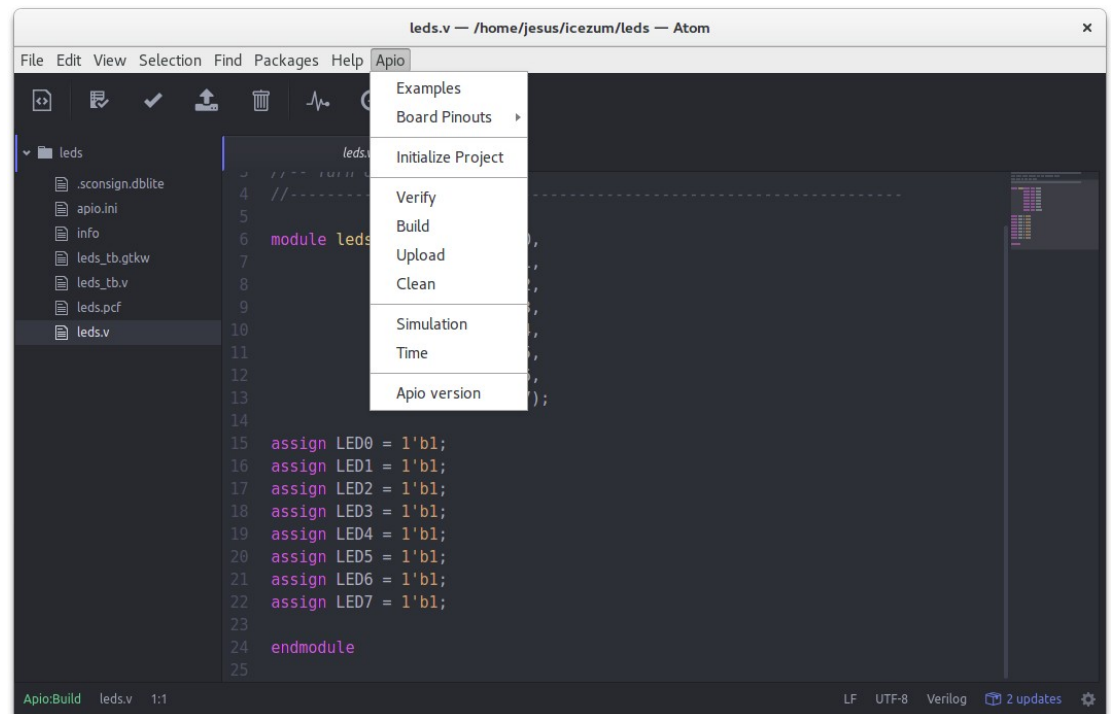
to let Atom find apio



Apio IDE

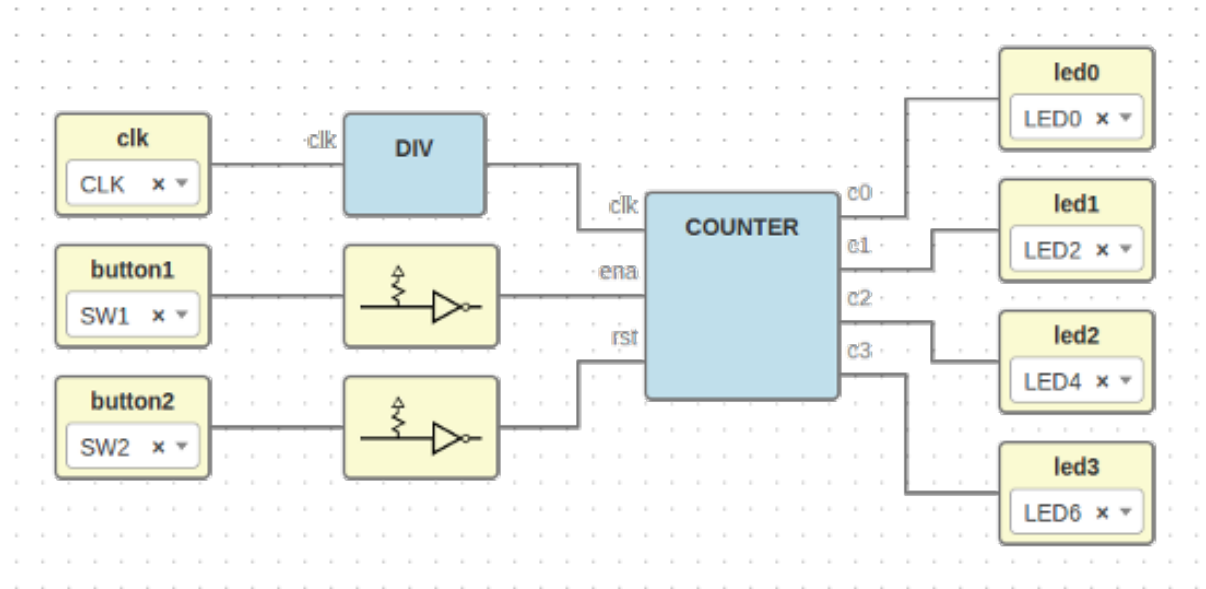
3. Hello, world!

- Open a new window
File > New Window
- Load icezum/leds
File > Open Folder...
- Load and edit leds.v
- Upload bit stream
Toolbar > Upload
Apio menu > Upload



All **apio code commands** are implemented in the GUI

Workshop: open FPGAs tools



Jesús Arroyo Torrens

<https://github.com/Jesus89>



OSHWDem 16
November 5, 2016
Museo Domus, A Coruña

<https://github.com/FPGAwards/workshops>

