

Rodrigo A. Melo

# Free and Open Source Software for FPGA development

Virtual | Feb | 2021



**Joint ICTP-IAEA School on FPGA-based  
SoC and its Applications for Nuclear and**



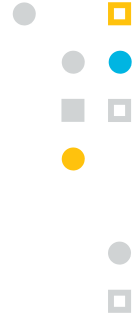
**Related Instrumentation | (smr 3562)**



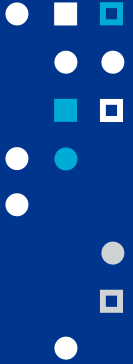
Ministerio de  
Desarrollo Productivo  
Argentina

# Outline

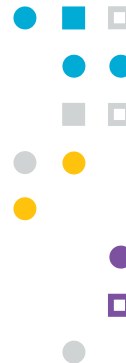
- 1 Introduction
- 2 General-purpose
- 3 Simulation
- 4 Frameworks and methodologies for testing and verification
- 5 Implementation
- 6 Other tools/projects
- 7 Boards
- 8 Final words



# Introduction



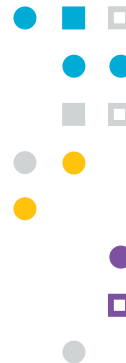
# What means FOSS?



- **F**ree (as freedom) and **O**pen **S**ource (you can access the source code) **S**oftware (programs).
- Solves the disambiguation between **free software** and **open-source software**.
- Anyone is freely licensed to **use, copy, study, and change** the software.



# Why to use FOSS?



## General

- Personal control, customization and freedom
- Privacy and security
- Low or no costs (solutions and support)
- Quality, collaboration and efficiency
- High level of flexibility and open-standards adherence
- Innovation

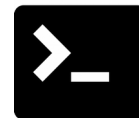
## Particular for FPGA development

- Vendor-independence
- Lightweight tools (size and speed)

# General-purpose

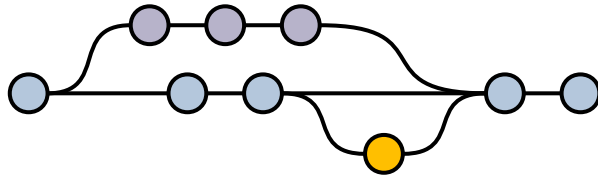


- Aka shell, terminal, console, bash, etc.
- Most projects provide one or more *Command-line interface (CLI)* tool/s.
- It is common for Linux distributions.
- You can use the *Windows Subsystem for Linux (WSL)*, which is a compatibility layer for running Linux binary executables (probably with some limitations).



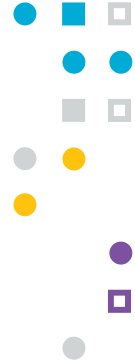
```
$ git init
Initialized empty Git repository in /tmp/tmp.IMBYSY7R8Y/.git/
$ cat > README << 'EOF'
> Git is a distributed revision control system.
> EOF
$ git add README
$ git commit
[master (root-commit) e4dcc69] You can edit locally and push
to any remote.
 1 file changed, 1 insertion(+)
 create mode 100644 README
$ git remote add origin git@github.com:cdown/thats.git
$ git push -u origin master
```

- Is a distributed version control system.
- It was created in 2005 by Linus Torvalds, the Linux kernel creator, for development of the Linux kernel.
- Is the de facto standard for FOSS projects.
- Allows you to deal with a software repository (repo), managing versions and multiple users.

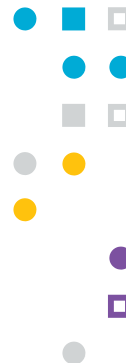


These platforms, which are tools provided around Git, are the nowadays online CV.

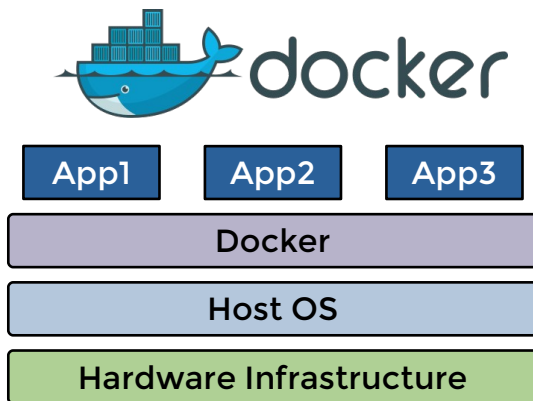
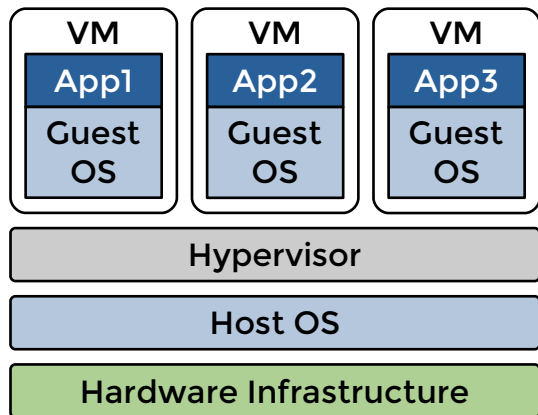
*“Take Concurrent Versions System (CVS) as an example of what not to do; if in doubt, make the exact opposite decision”* Linux Torvald, 2007







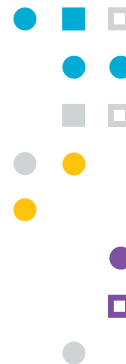
Docker uses OS-level virtualization to deliver software in packages called containers, which are isolated one from another and bundle their own software, libraries and configuration files.



✓  
All containers are run by a single OS Kernel and therefore uses fewer resources than VM.

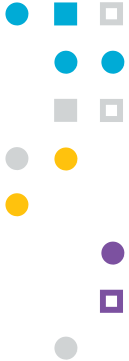


# Continuous integration, Delivery and Deployment (CI/CD)

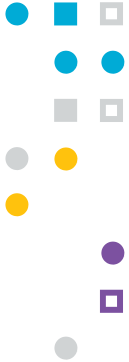


- Is to automatically perform an action based on a repository event (push, merge, cron, etc).
- **Continuous Integration:** run linters, unit and/or integration tests, **Hardware-in-the loop** simulation.
- **Continuous Delivery:** build binaries, documentation, packages, etc.
- **Continuous Deployment:** build and install in production.

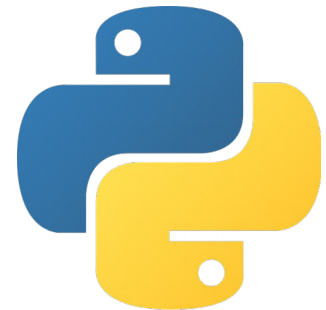




- Is a build automation tool.
- A Makefile contains a set of directives (targets, dependencies and rules) which are used by make to generate a target/goal.
- It works upon the principle that files only needs to be recreated if their dependencies are newer than the file being re/created.
- There are other newer tools such as CMake and Scons, but make is definitively the building tool, and sometimes part of the execution, in the FPGA ecosystem.

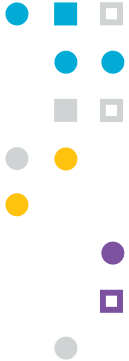


- Is an interpreted, high-level and general-purpose programming language (one of the most used in general, and the main in certain fields such as Machine/Deep Learning).
- A lot of its libraries are written in C (performance).
- Easy to read and learn.
- Most FOSS FPGA tools are written in Python, or C/C++ with a Python binding/wrapper.
- There are several HDL languages based on Python.
- Is also being used as verification language.



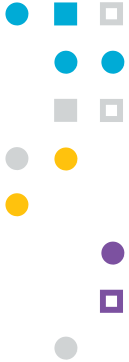
# Simulation





# GHDL

- Analyzer, compiler, simulator and (experimental) synthesizer for VHDL.
- Full support for the 1987, 1993, 2002 versions of the IEEE 1076 VHDL standard, and partial for the latest 2008 revision. Partial support of PSL (Property Specification Language).
- It generates binaries to perform a simulation.
- Can write waveforms to VCD or GHW (recommended for VHDL) files.



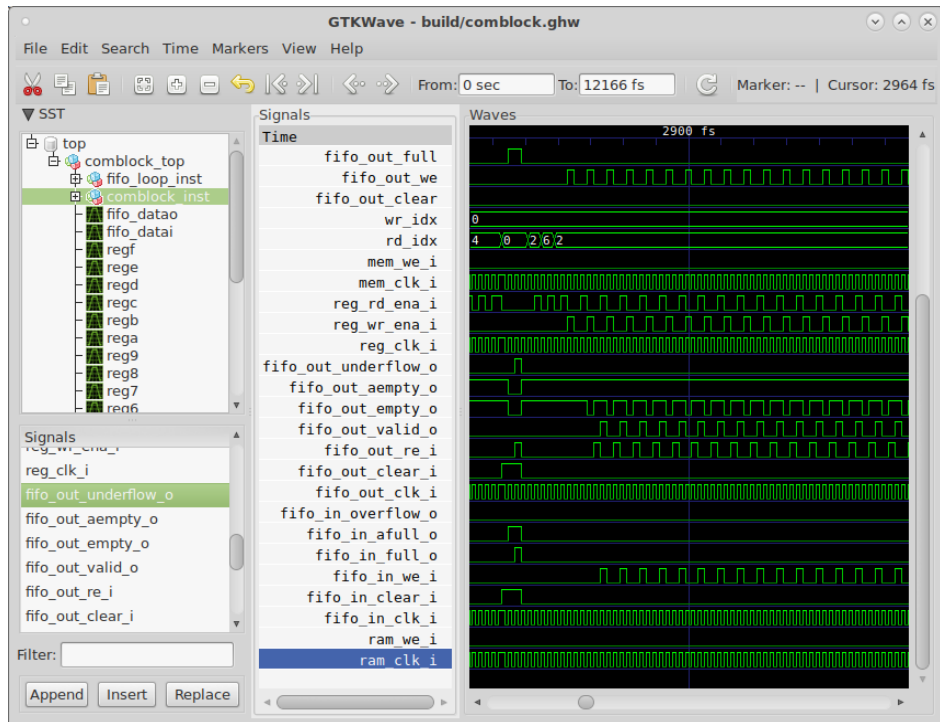
- Verilog (IEEE-1364) simulator.
- It generates an intermediate file format which is after executed by a command.



- Verilog/SystemVerilog simulator.
- Compiles into multithreaded C++.
- Performs lint code-quality checks.



GTKWave is a fully featured wave viewer which reads LXT, LXT2, VZT, FST, and GHW files as well as standard Verilog VCD/EVCD files and allows their viewing

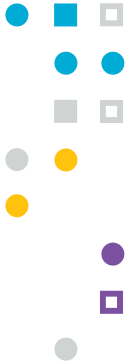






# Frameworks and methodologies for testing and verification

# HDL based frameworks/methodologies



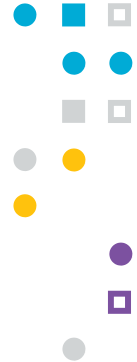
- **OSVVM**: Open Source VHDL Verification Methodology
- **UVVM**: Universal VHDL Verification Methodology
- **VUnit**: unit testing framework for VHDL/SystemVerilog
- **SVUnit**: unit testing framework for Verilog/SystemVerilog

 OSVVM

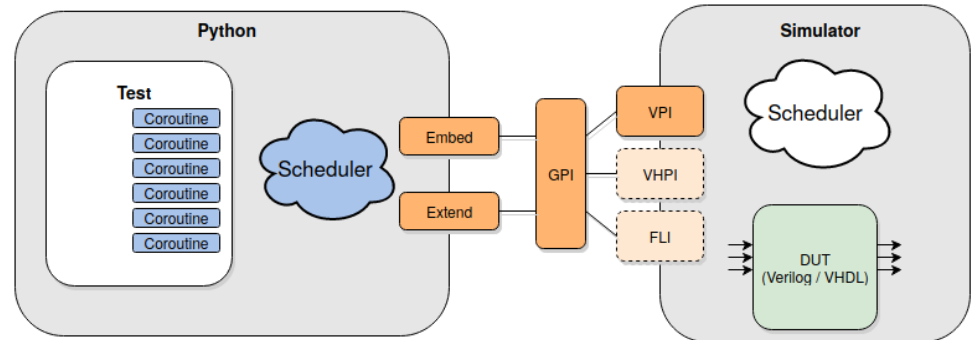
 UVVM



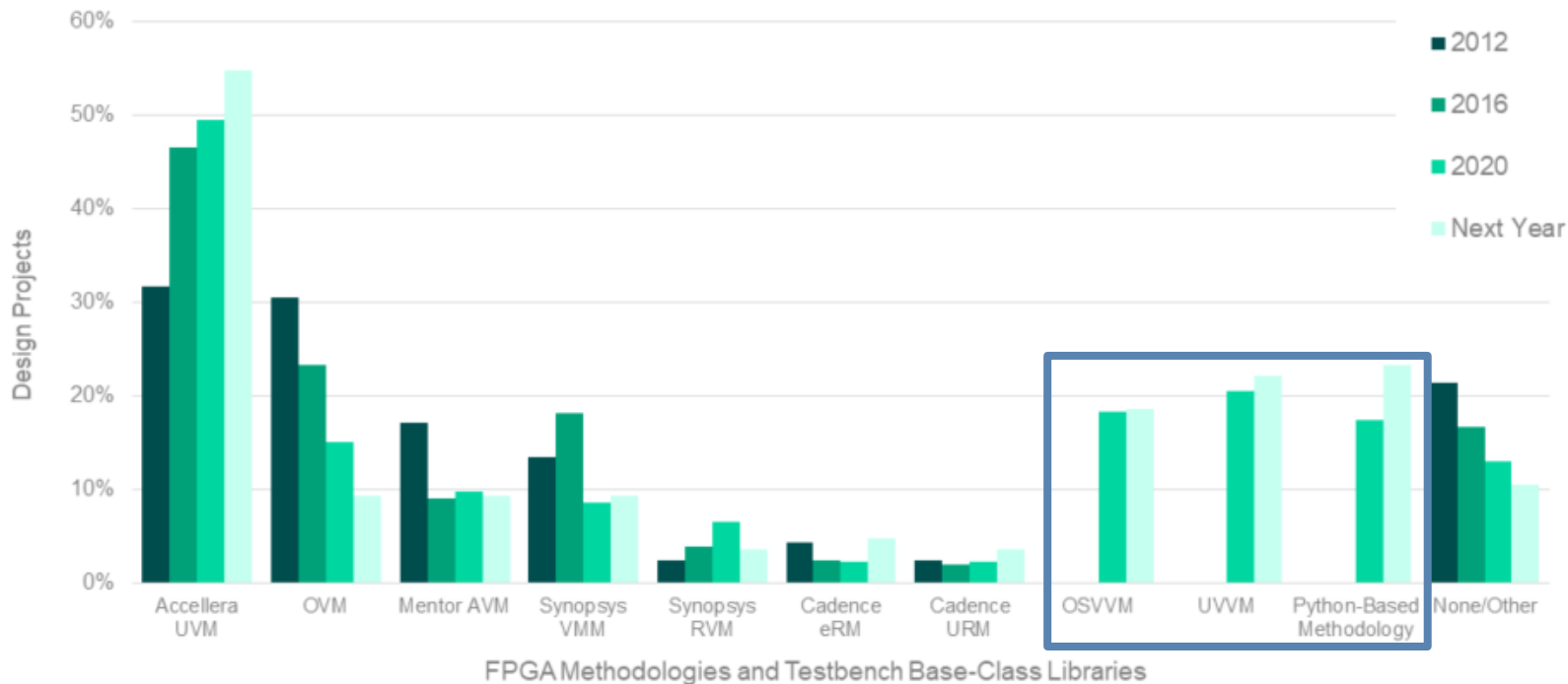
 SVUnit

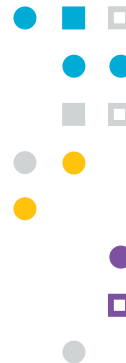


- **cocotb**: Coroutine Co-simulation Test Bench
- A coroutine based cosimulation library for writing VHDL and Verilog testbenches in Python
- Supported simulators: **ghdl**, **iverilog**, **verilator**, Synopsys VCS, Aldec Riviera-PRO, Aldec Active-HDL, Mentor Questa, Mentor ModelSim, Cadence Incisive, Cadence Xcelium, Tachyon DA CVC.



# Verification trends





- **SymbiYosys (sby)**: front-end driver program for **Yosys-based** formal hardware verification flows.
- Formal Verification is the act of proving the correctness of intended algorithms underlying a system with respect to a certain formal specification or property, using formal methods of mathematics (assumptions and assertions).
- Supports Verilog (free), VHDL and SystemVerilog (through verific with a license).

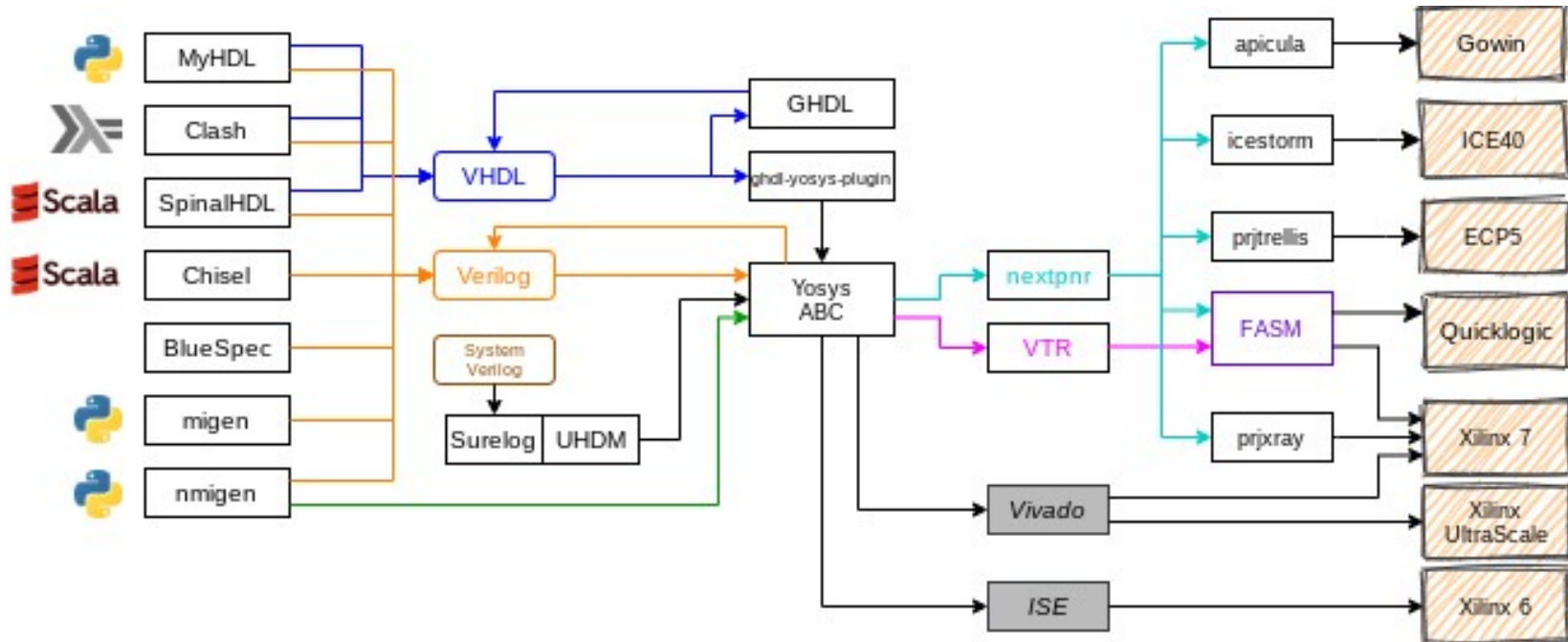


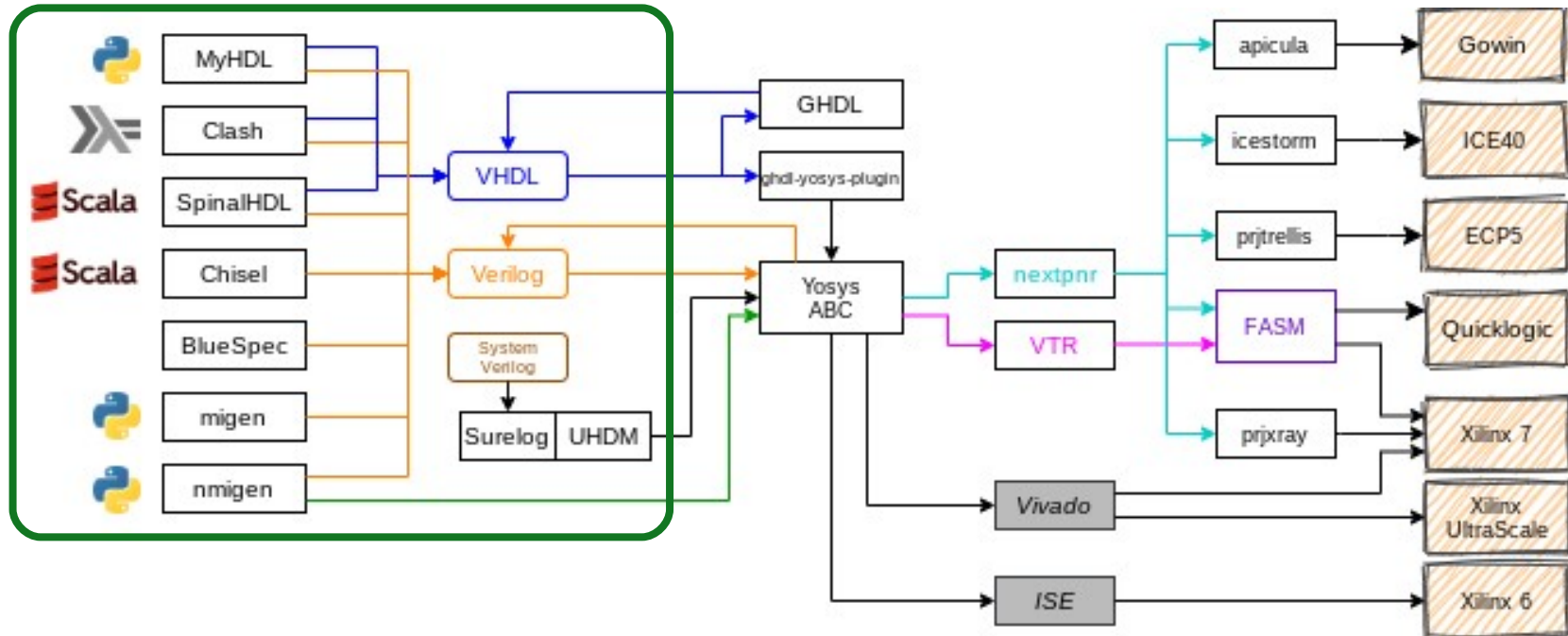
# Implementation

## (HDL-to-Bitstream)

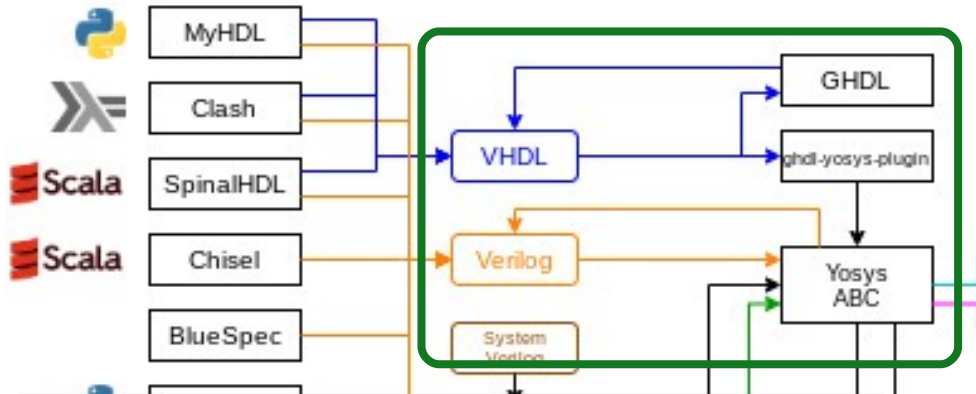


# HDL-to-Bitstream main related tools





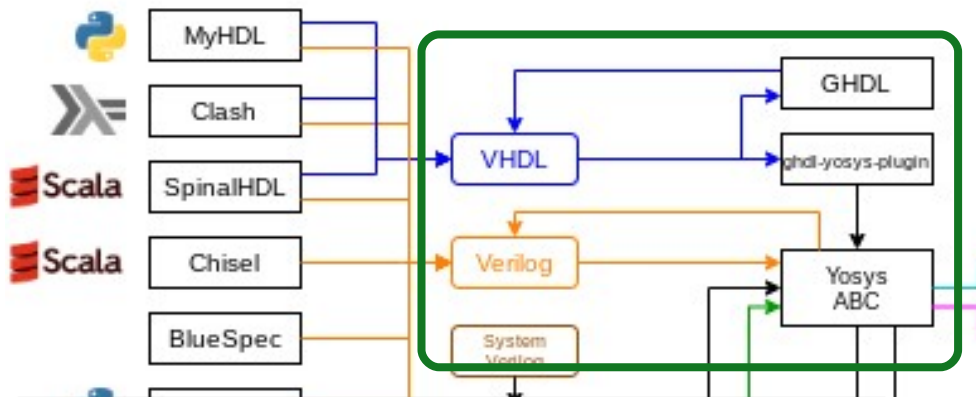




Is to convert an abstract specification of a circuit (being an HDL a common input) into a design implementation in terms of the basic blocks supported by the chosen technology (being a netlist the output).

## Yosys Open SYNthesis Suite

Is a framework for RTL synthesis tools. It currently has extensive Verilog-2005 support and provides a basic set of synthesis algorithms for various application domains. *It was the first useful FOSS synthesizer.* Supports devices from Lattice (iCE40 and ECP5), Xilinx (Series 7, Ultrascale, and others), Gowin, Achronix, Intel, Microsemi, etc.

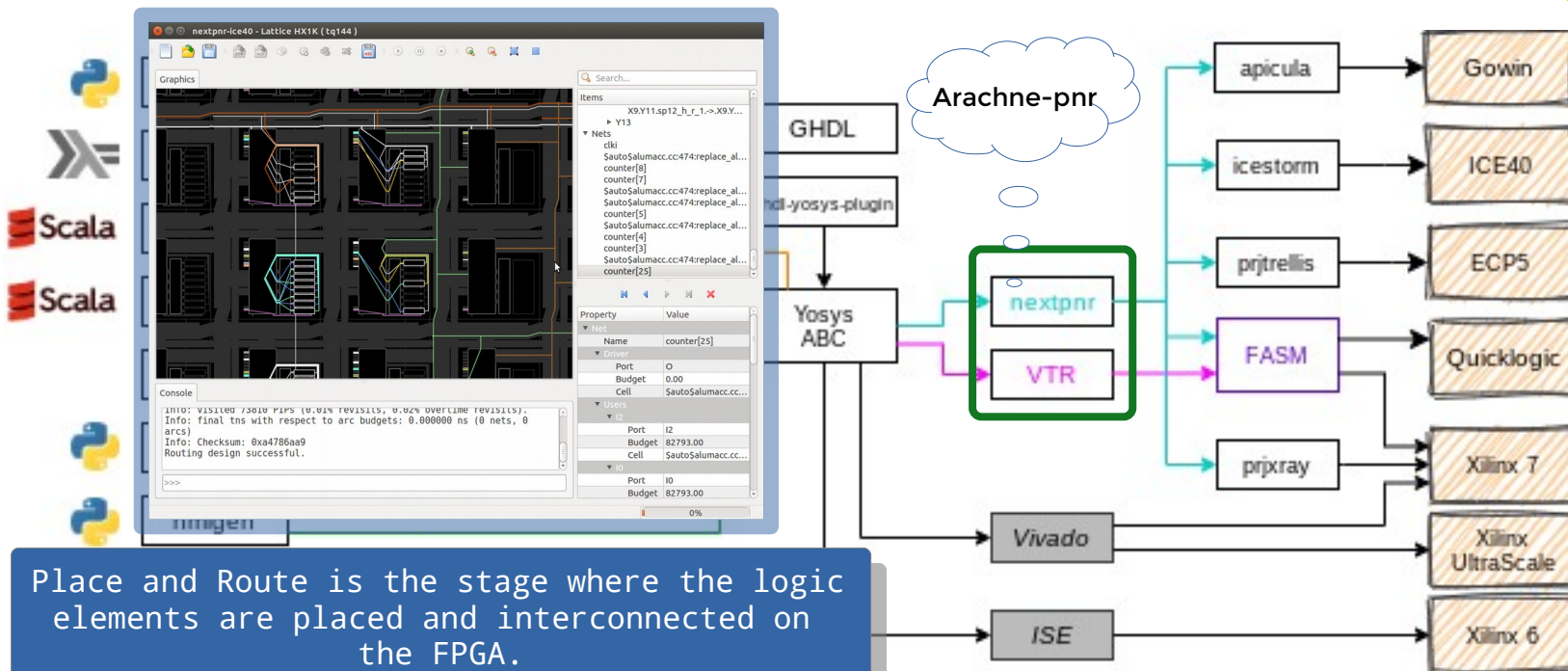


Is to convert an abstract specification of a circuit (being an HDL a common input) into a design implementation in terms of the basic blocks supported by the chosen technology (being a netlist the output).

# GHDL

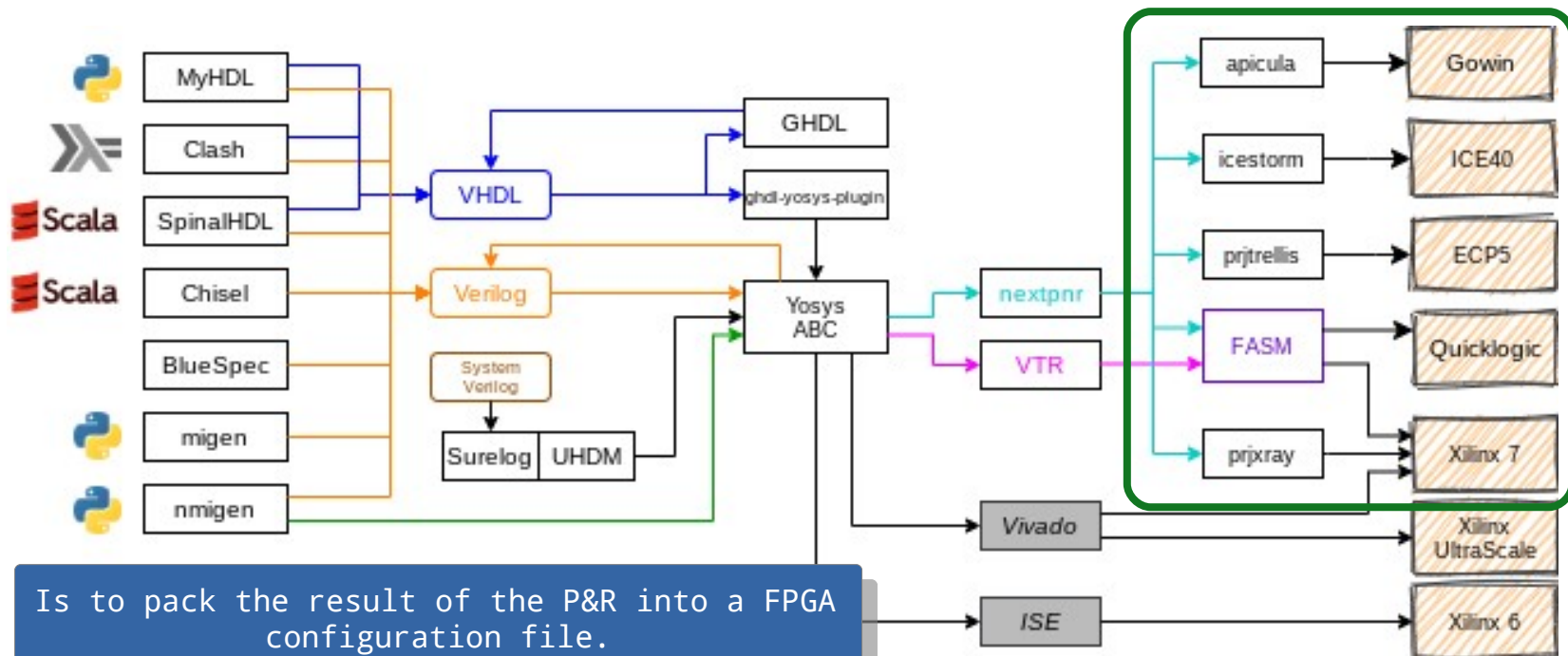
**GHDL:** the open-source analyzer, compiler, simulator and (experimental, general purpose) synthesizer for VHDL.

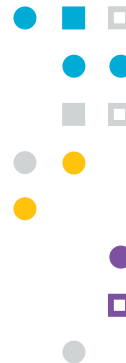
**ghdl-yosys-plugin:** VHDL synthesis, based on GHDL and Yosys.



Place and Route is the stage where the logic elements are placed and interconnected on the FPGA.

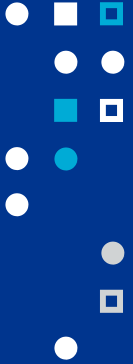
# Bitstream Generation

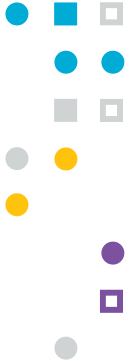





- **OpenOCD:** Free and Open On-Chip Debugging, In-System Programming and Boundary-Scan Testing
- **UrJTAG:** universal JTAG library, server and tools.
- **icestorm:** programmer of the IceStorm project (FTDI-based programmers).
- **ecpprog:** programmer for the Lattice ECP5 series (FTDI-based programmers).
- **openFPGALoader:** universal utility for programming FPGA.
- **dfu-util:** Device Firmware Upgrade Utilities (intended to download/upload firmware to devices connected over USB).

# Others tools/projects

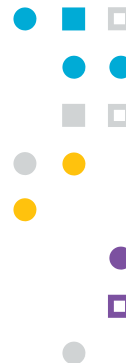




- **HDLmake:** tool for generating multi-purpose makefiles for FPGA projects. 
- **edalize:** a Python Library for interacting with EDA tools (was part of FuseSoC, now its build backend).
- **PyFPGA:** A Python package to use FPGA development tools programmatically.



- Supports Synthesis, Implementation, Bitstream generation and Programming from Python.
- Supports **ISE**, **Vivado**, **Quartus**, **Libero-SoC** and open-source tools (**Yosys**, **GHDL**, **ghdl-yosys-plugin**, **nextpnr**, **icestorm**, **trellis**).
- **Helpers:** hdl2bit, prj2bit & bitprog.



- **PoC (Pile of Cores Library):** a library of free, open-source and platform independent IP cores.
- **FuseSoC:** package manager and build abstraction tool (edalize) for FPGA/ASIC development.
- **Litex:** a Migen/MiSoC based SoC builder that provides the infrastructure to easily create Cores/SoCs
- **OpenCores and LibreCores:** collections of IP-cores.

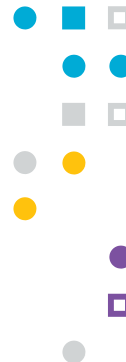


✓ There are lot of FOSS projects at **GitHub** and **GitLab**.



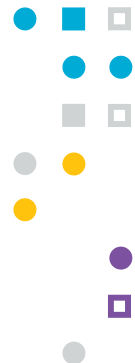


- **Leon 3 (Gaisler):**
  - Is a synthesisable VHDL model of a 32-bit processor compliant with the SPARC V8 architecture.
  - GNU GPL license for research and education.
  - Distributed as part of the GRLIB.
- **OpenRISC:**
  - Specification OpenRISC 1000 (32/64 bits)
  - The flagship implementation, the OR1200, is written in Verilog.
  - Distributed as part of OpenRISC Reference Platform System-on-Chip (ORPSoC).



- **Leon-3 (Gaisler):**
  - ~~Is a synthesisable VHDL model of a 32-bit processor compliant with the SPARC V8 architecture.~~
  - ~~GNU GPL license for research and education.~~
  - ~~Distributed as part of the GRLIB.~~
- **OpenRISC:**
  - ~~Specification OpenRISC 1000 (32/64 bits)~~
  - ~~The flagship implementation, the OR1200, is written in Verilog.~~
  - ~~Distributed as part of OpenRISC Reference Platform System-on-Chip (ORPSoC).~~

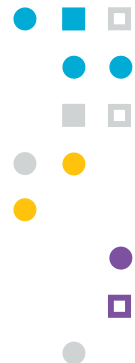
# Softcores - RISC V



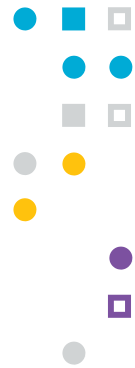
The image displays a comprehensive list of RISC-V softcore providers and partners, organized in a grid. The logos include:

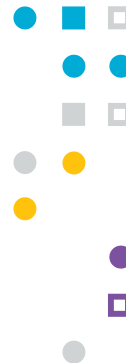
- Row 1:** Aril Inc, ANDES, antmicro, aselsan, ALLWINNER, Alibaba.com, AdaCore, adafruit, Airavath Foundation, APEXMIC, GRADIENT, ASHLING.
- Row 2:** Baylibre, Berkeley Architecture Research, BITMAIN, bluespec, BOSTON UNIVERSITY, BAE SYSTEMS, cadence, cortus, DOVER MICROSYSTEMS.
- Row 3:** CE2A, CEVA, CLEMSON, CloudBEAR, COBHAM, csem, Codasip, Collins Aerospace, CRYPTAPE, CPSEC, GALOIS.
- Row 4:** DRAPER, dxcorr, EMBECOSM, Esperanto Technologies, ESPRESSIF, ETH zürich, expresslogic, oculus, FORTH, FUTUREWEI, galois.
- Row 5:** GHELIA, Golden Gate University, Google, GOWIN, GREEN WAVES, HENSOLDT, Hewlett Packard Enterprise, HEX-Five Security, HITACHI, Honeywell.
- Row 6:** HORTONWORKS, HUAWEI, huami, IAR SYSTEMS, IBM, ICE, IDT, Imagination, imperas, imt., infineon, Ingenic, Inria, inside secure, miru bangladesh.
- Row 7:** inspur, ISCAS, INTRINSIC ID, Intrinsic, KU LEUVEN, LATTICE, LAUTERBACH, lowRISC, MEDIATEK, Mellanox, Mentor, Minima.
- Row 8:** MIT CSAI, Microsemi, MICROCHIP, MINRES, Micron, Morse Micro, Mosaik, NTNU, NHI-TEXE, NUCLEI, NVIDIA, NUS, NOKIA, NORDEC, REALTEK.
- Row 9:** Ockam, onespin, OPENHW, ORION, pango, PerlXLab, PRINCETON UNIVERSITY, qamcom, QUALCOMM, QuickLogic, Rambus, Rumble Development.
- Row 10:** NERVOS, NETRONOME, NORTHROP GRUMMAN, NP, SEMORIVE, SECURE RF, SEAGATE, SANCIPS, SAMSUNG, SEGGER, emidynamic, SHIELD.
- Row 11:** SiFive, SIEMENS, SILICON LABS, SK hynix, SHC, SONY, SoundAI, sparkfun ELECTRONICS, SILEX INSIGHT, Logic, REM, Red Hat, SRI International, SYNOPSIS, SUSTech.
- Row 12:** ST, surecore, Symbiotic EDA, Syntacore, Syntrox, SYZEXION, Technology, Telink, TrustKernel, THALES.
- Row 13:** TIMESILICON, Tortuga Logic, TRINAMIC, Think Silicon, UBC, UBILITE, ultraSOC, 云知声, Unisound, 東京大学, University of BRISTOL, 立功科技.
- Row 14:** UPPSALA UNIVERSITET, KU, VectorBlox, VenSilicon, 创景科技, Western Digital, WIND RIVER, WIT, XILINX, XtremeEDA, ZeroPoint, 信大捷安, VENTANA MICRO.

# Softcores - RISC V



# Softcores - RISC V





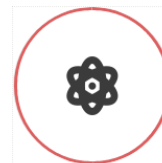
```

41  state next <= ST_S2;
42  end if;
43  when others=> -- ST_S3 =>
44    state_next <= ST_S1;
45  end case;
46  and process p_comb;
47
48
49  process(input1, input2, state_reg)
50  begin
51    state_next <= state_reg; -- default state_next
52    case state_req is
53      when s0 =>
54        if input>3 then -- if (input1 = '01') then
55          state_next <= s2;
56        elsif input=4 then -- add all the required conditionstions
57          state_next <= s1;
58        else -- remain in current state
59          state_next <= s0;
60        end if;
61      when s1 =>
    
```

State machine diagram

```

graph TD
    s0((s0)) -- "input=0" --> s2((s2))
    s0 -- "input=1" --> s1((s1))
    s0 -- "input=2" --> s0
    s1 -- "input=0" --> s0
    s1 -- "input=1" --> s1
    s1 -- "input=2" --> s0
    s2 -- "input=0" --> s0
    s2 -- "input=1" --> s1
    s2 -- "input=2" --> s2
    
```



Atom



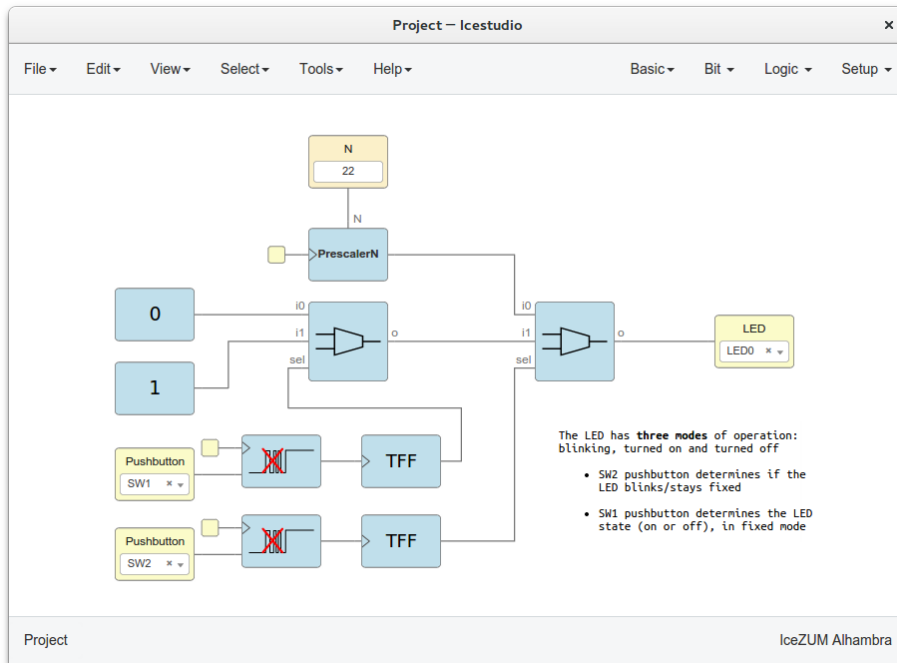
VSCode



VERILATOR



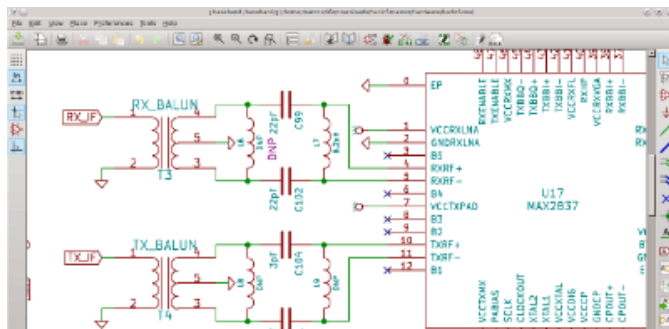
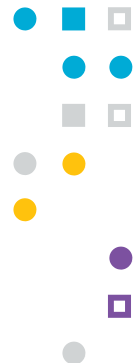
ICARUS



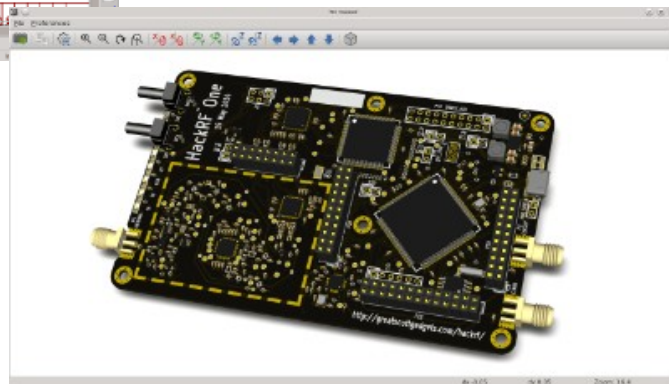
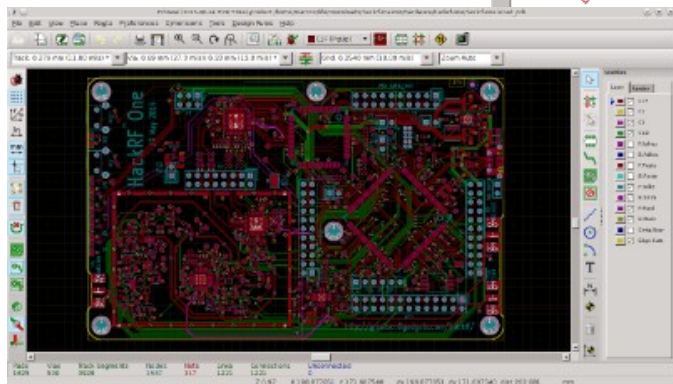
# Boards (Open Hardware)

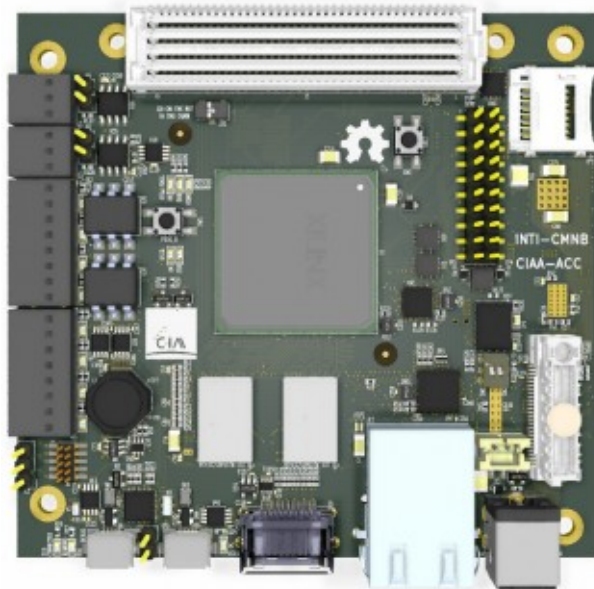






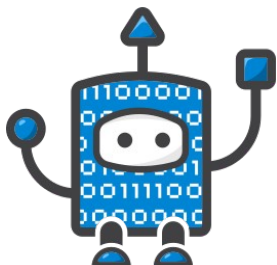
  
open source  
hardware



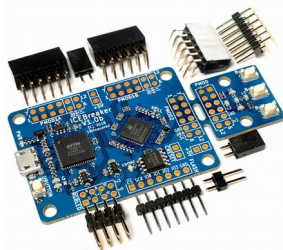


**12 Layers!!! Based on a Zynq-7030**

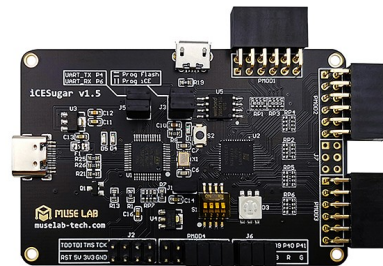
# Some ICE40 based boards



**FOMU**



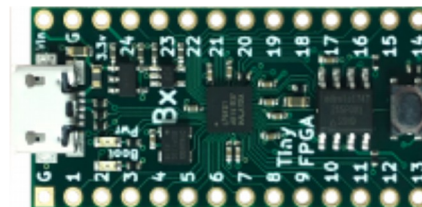
**iCEBreaker**



**iCESugar**

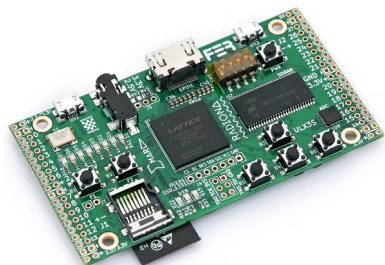


**IceZUM Alhambra / Alhambra II**

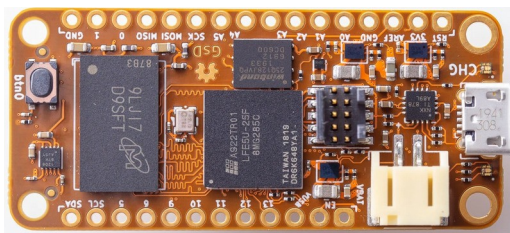


**TinyFPGA BX**

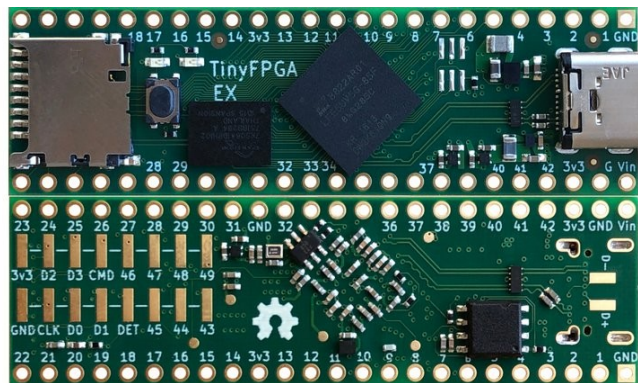
# Some ECP5 based boards



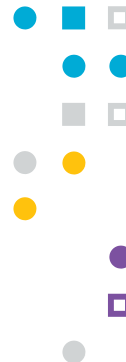
**ULX3S**



**OrangeCrab**

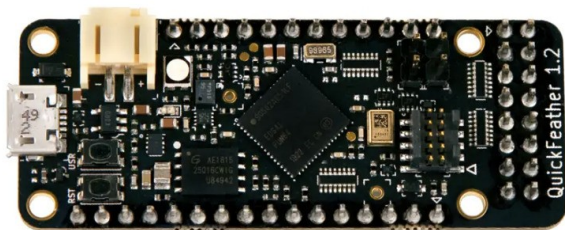
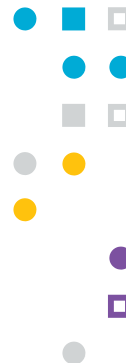


**TinyFPGA EX**





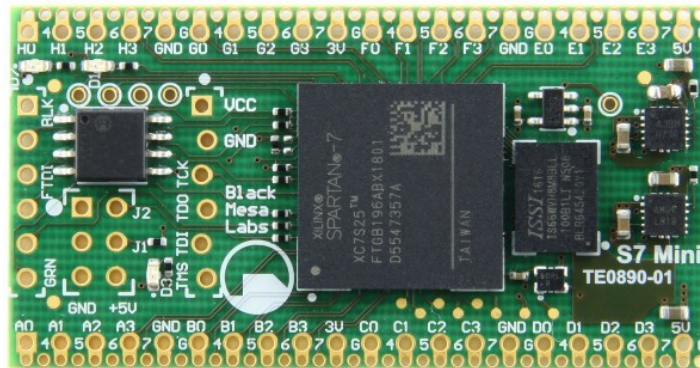
# Some other boards



**QuickFeather – EOS S3**



**SymbiFlow**



**S7 mini – Spartan 7**





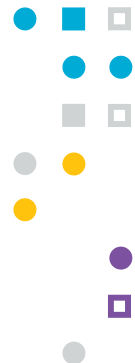
Instituto  
Nacional  
de Tecnología  
Industrial

**INTI**

# Final words



# How to find projects and be updated? Organizations



**El Correo Libre**  
 \* Monthly newsletter

**SymbiFlow**  
 The GCC for FPGAs

Yosys

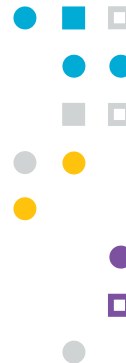
- |                   |                     |
|-------------------|---------------------|
| nextpnr<br>ICE40  | Project<br>IceStorm |
| nextpnr<br>ECP5   | Project<br>Trellis  |
| nextpnr<br>others | Other<br>projects   |
| VTR               | Project<br>X-Ray    |



YosysHQ

- |        |
|--------|
| SBY    |
| MCY    |
| nMigen |
| others |

# How to find projects and be updated? People



**Tim 'mithro' Ansell**



**mithro**



**@mithro**

## SymbiFlow

Verilog, Synthesis, Yosys

**Unai Martinez-Corral**



**umarcor**

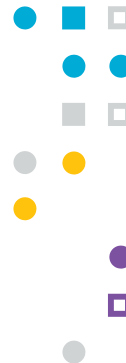
## GHDL



VHDL, Simulation, GHDL



# How to find projects and be updated? hdl/awesome



## Awesome resources for Hardware Description

Tools, frameworks, IP cores, libraries and more!

Welcome to Awesome HDL, a community effort for maintaining a curated list of resources for Hardware Description, such as [Tools](#), [IP Core Libraries](#) and [Collections](#), [Frameworks](#) and [more!](#) Read all [about it](#) and see [how to contribute](#).

### Categories

Curated mid-sized taxonomy

### Tags

Loose taxonomy

### All items

No filter

## pyFPGA

A Python package to use FPGA development tools programmatically



Maintained by: Rodrigo Alejandro Melo  
Licensed under: GPL-3.0-or-later  
Since 2019

PyFPGA is a Python Class for vendor-independent FPGA development. It allows using a single project file and programmatically executing synthesis, implementation, generation of bitstream and/or transference to supported boards.

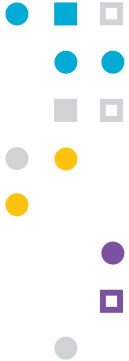
- The workflow is command-line centric.
- It's friendly with Version Control Systems and Continuous Integration (CI).

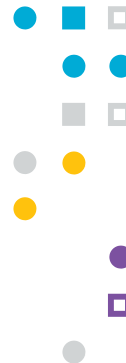
<https://github.com/hdl/awesome>

<https://hdl.github.io/awesome>

# How to obtain updated tools

- Several projects provide Dockerfiles or nightly builds to test its development.
- YosysHQ/fpga-toolchain: Multi-platform nightly builds of open source FPGA tools.
- hdl/MINGW-packages: Electronic design automation (EDA) package recipes for MinGW-w64 (MSYS2).
- hdl/containers: Building and deploying container images for open source electronic design automation (EDA).
- ghdl/docker: Scripts to build and use docker images including GHDL.





- Install Docker following the [instructions for your OS](#).
- Run: `$DOCKER <CONTAINER> <TOOL> <OPTIONS>`

```
$ DOCKER="docker run --rm -v $HOME:$HOME -w $PWD"
```

```
$ $DOCKER ghdl/ghdl:buster-mcode ghdl --version
GHDL 2.0.0-dev (v1.0.0-13-gad9906d3) [Dunoon edition]
  Compiled with GNAT Version: 8.3.0
  mcode code generator
  Written by Tristan Gingold.
```

```
Copyright (C) 2003 - 2021 Tristan Gingold.
GHDL is free software, covered by the GNU General Public License. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
$ $DOCKER hdlc/yosys yosys --version
Yosys 0.9+3894 (git sha1 eff18a2b, clang 7.0.1-8+deb10u2 -fPIC -Os)
```



The first run of a tool, or when outdated, will produce the **pull** of a new image from Dockerhub.



[rmelo@inti.gov.ar](mailto:rmelo@inti.gov.ar)



[rodrigoalejandromelo](https://www.linkedin.com/in/rodrigoalejandromelo)



[@rodrigomelo9ok](https://twitter.com/rodrigomelo9ok)



[rodrigomelo9](https://plus.google.com/rodrigomelo9)



[rodrigomelo9](https://www.youtube.com/rodrigomelo9)





# Thank you

This work is licensed under CC BY 4.0



If you want to know more about  
INTI, we wait for you at

 INTIArg

 @INTIargentina

 INTI

 @intiargentina

 canalinti

[www.inti.gob.ar](http://www.inti.gob.ar)

[consulta@inti.gob.ar](mailto:consulta@inti.gob.ar)

0800 444 4004

