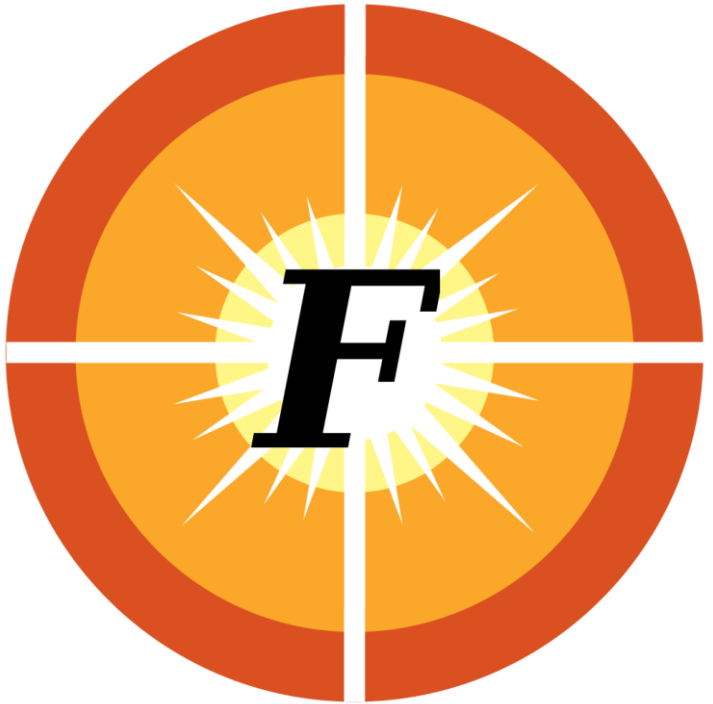
A CPU is Only as Good as its Ecosystem:

Turning RISC-V CPUs into Systems with FuseSoC



Olof Kindgren

Qamcom Research & Technology

FOSSi Foundation

Spring 2022 RISC-V week - Paris 2022.05.05

Who am I?

ORConf

Portland Oregon

LATCH-UP

May 4-5 2019

Award-winning

FOSSi Foundation

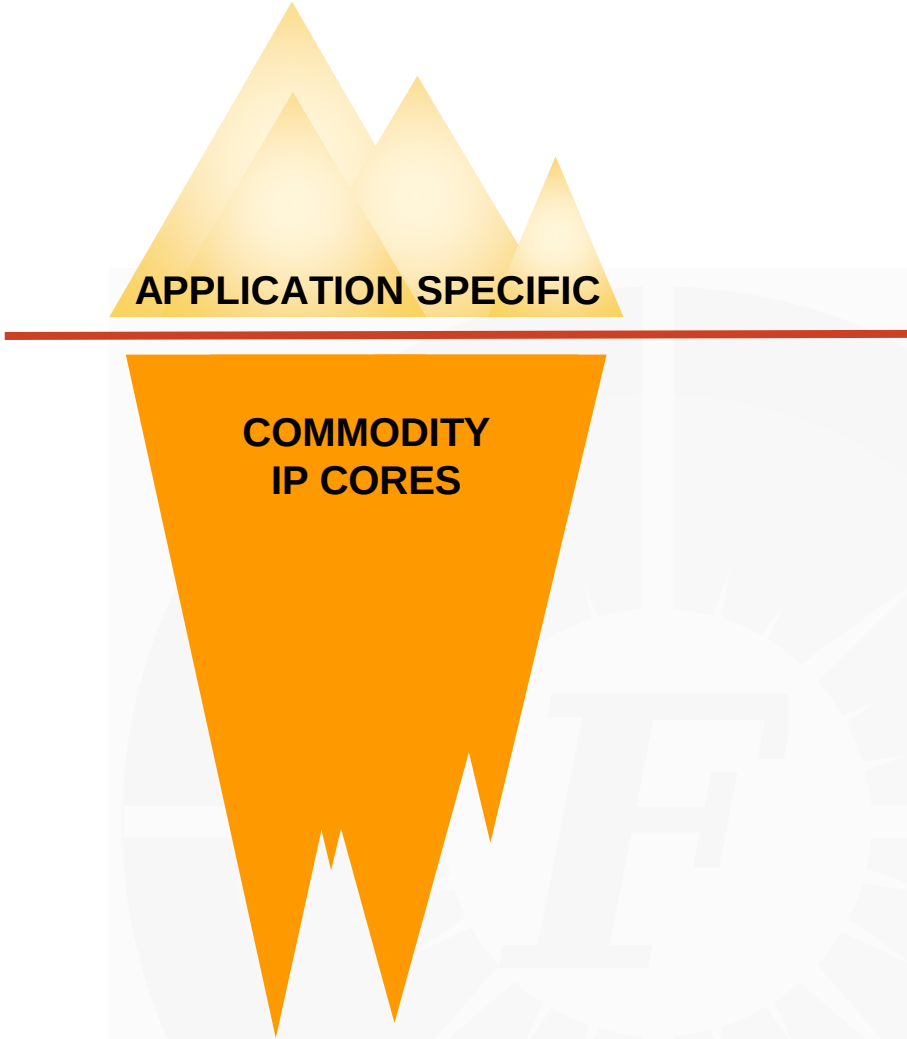
qamcom

IP cores

Traditional HDL designs are built from IP cores.

Ideally, a product should be built upon a foundation of existing commodity IP cores with the value-add on top.

This is how software products are normally developed.

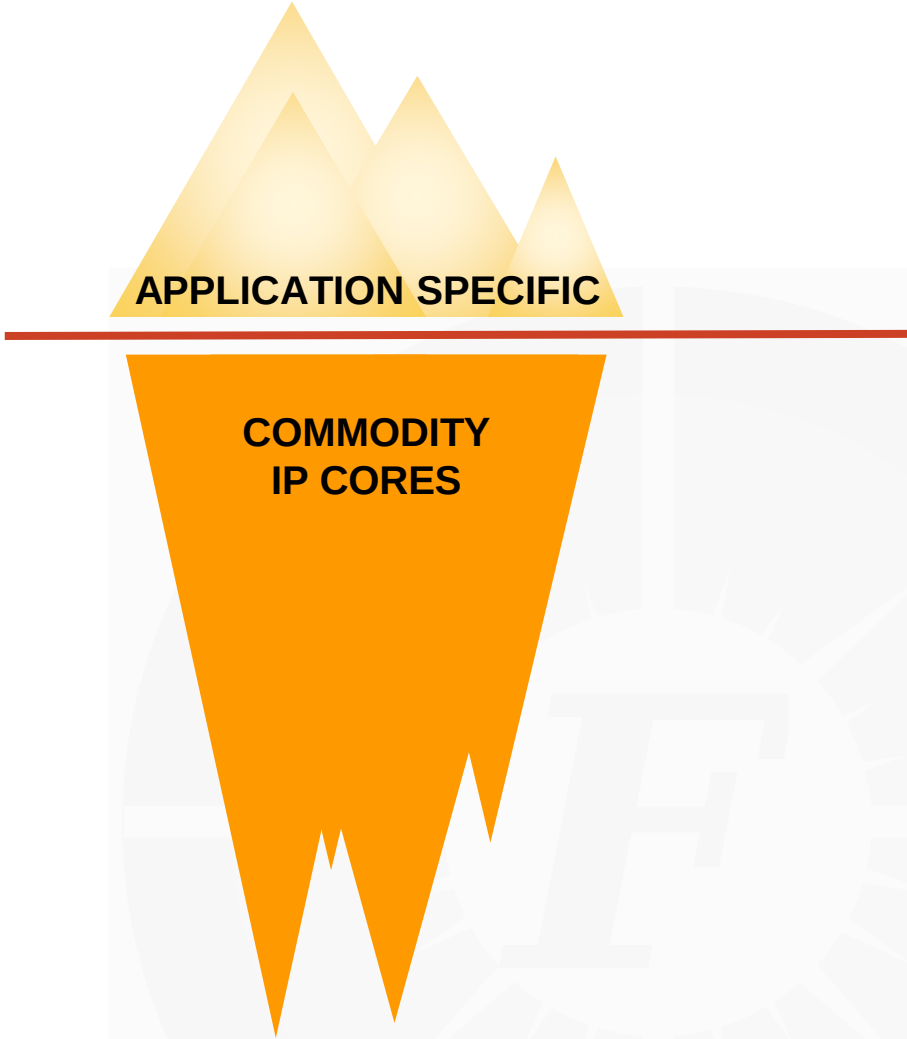


IP cores

IP cores come from four sources.

A non-trivial design normally use a mix of these.

- In-house developed
- 3rd party proprietary
- Platform-provided
- 3rd party open source



What is FuseSoc?

What is FuseSoc?

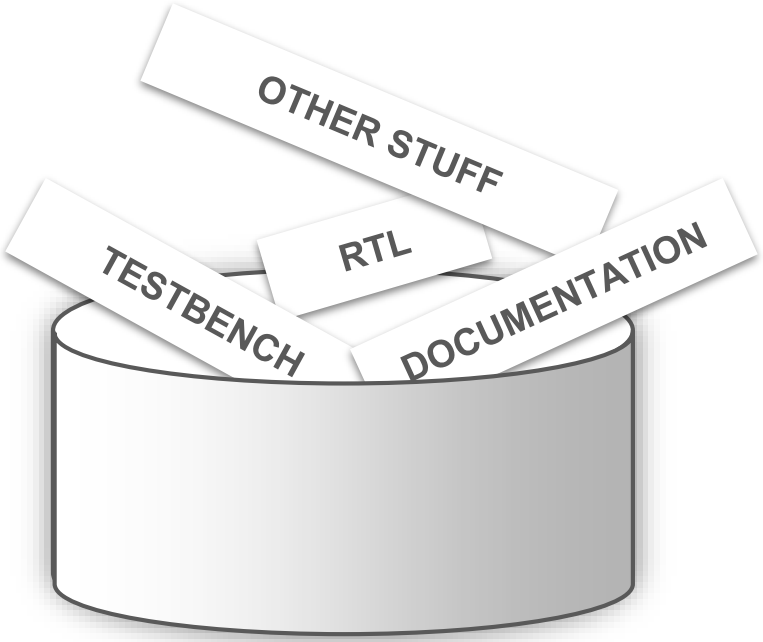
FuseSoC is a package manager…

What is FuseSoc?

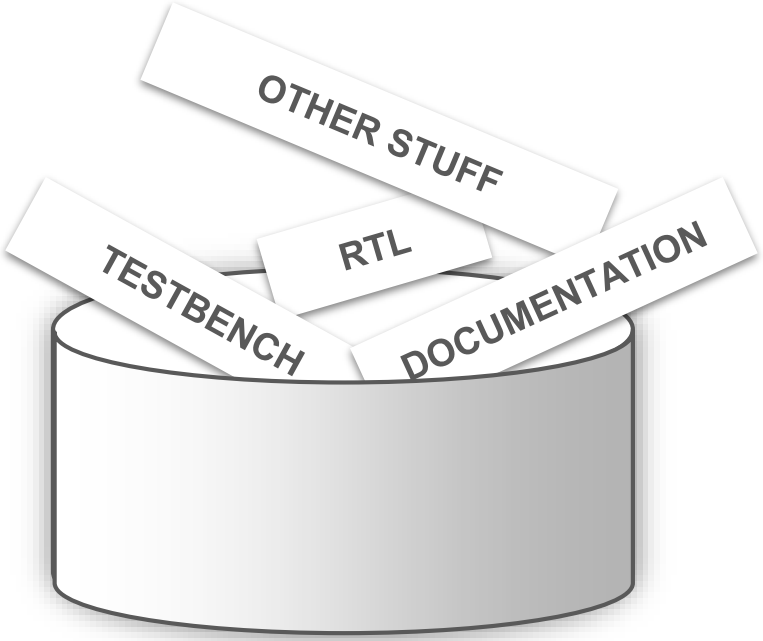
FuseSoC is a package manager…

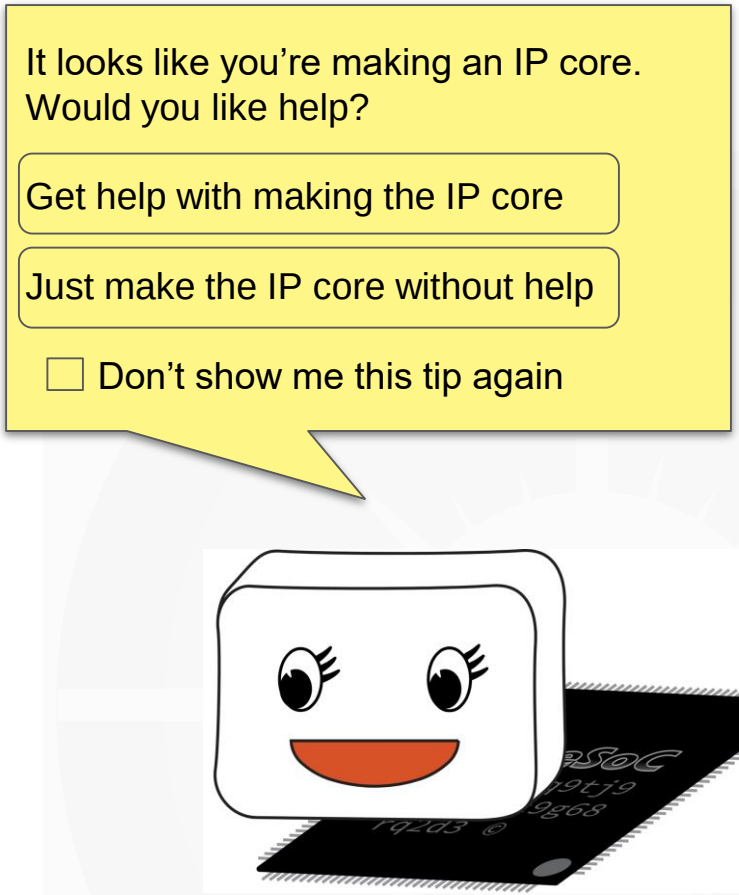
...and a build tool abstraction for HDL

What is a core?



What is a core?



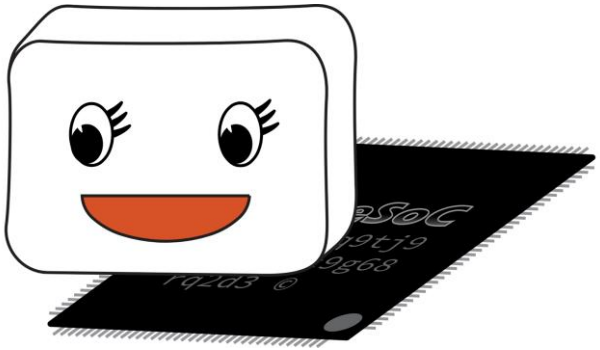


Core description files

```
...
targets:
  nexys_a7:
    default_tool: vivado
    filesets: [rtl, nexys_files]
    tools: [vivado: {part : xc7a100tcsg324-1}
    toplevel: corey_top
  tb:
    default_tool: modelsim
    filesets: [rtl, tb]
    toplevel: corey_tb
    tools:
      modelsim:
        vlog_options: [-timescale=1ns/1ns]
      xsim:
        xelab_options: [--timescale_1
```

Core description files describe properties of the core that the EDA tools need e.g.

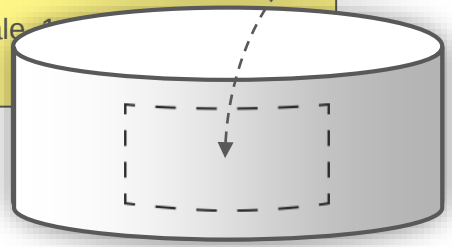
- files
- parameters
- tool options



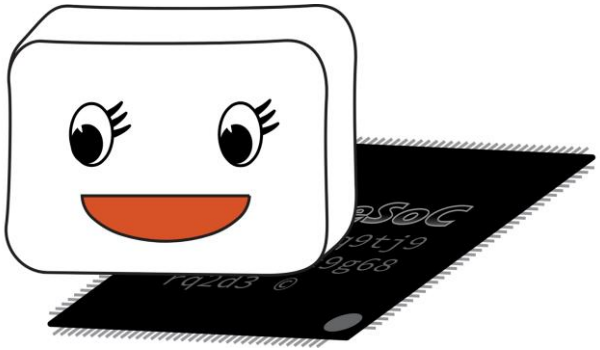
Core description files

```
...
targets:
  nexys_a7:
    default_tool: vivado
    filesets: [rtl, nexys_files]
    tools: [vivado: {part : xc7a100tcsg324-1}
    toplevel: corey_top
  tb:
    default_tool: modelsim
    filesets: [rtl, tb]
    toplevel: corey_tb
    tools:
      modelsim:
        vlog_options: [-timescale=1ns/1ns]
      xsim:
        xelab_options: [--timescale
```

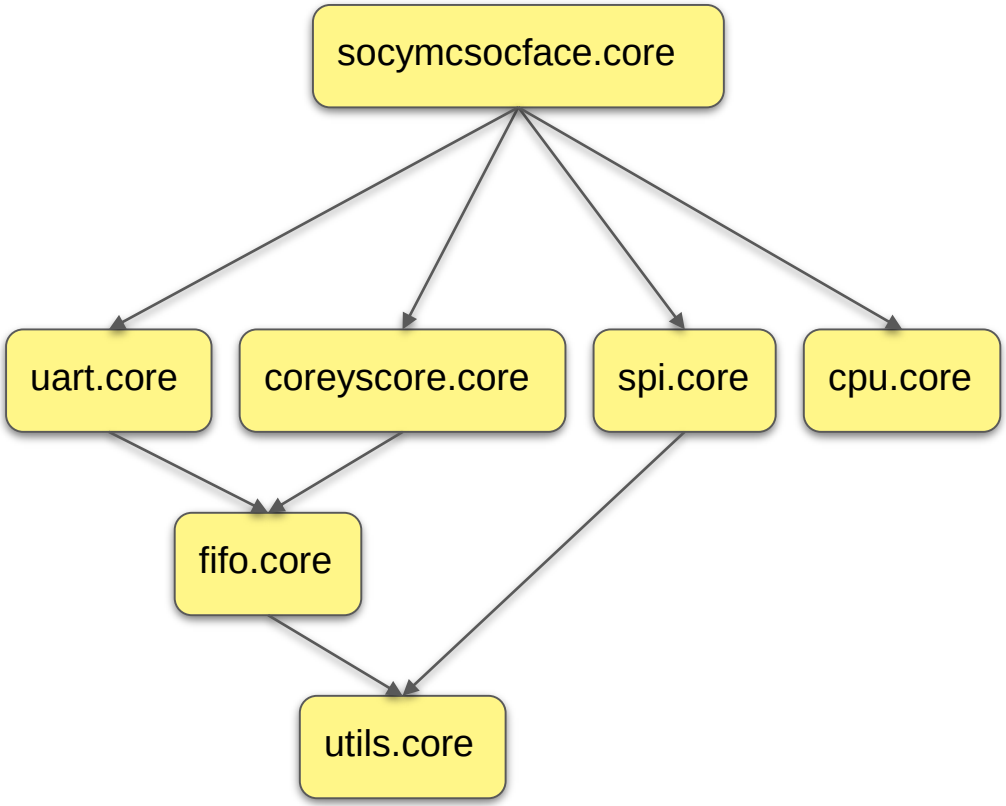
Coreyscore.core



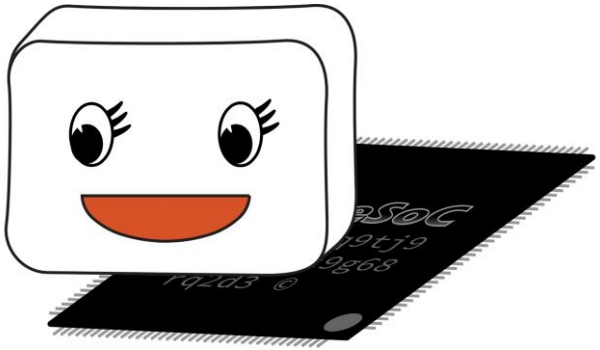
Core description files can be stored separately from the core, in which case they contain info on how FuseSoC can find the core files.



What is a core?



Cores list their immediate dependencies which forms dependency trees. FuseSoC resolves the dependencies to find a set of versions that satisfies all requirements.



What is a core?

uart.core

spi.core

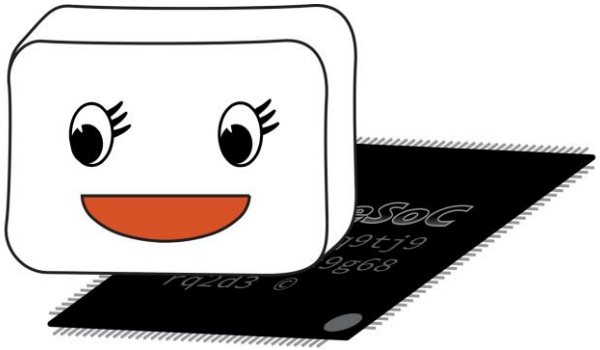
fifo.core

utils.core

coremensandiego.core

coreyscore.core

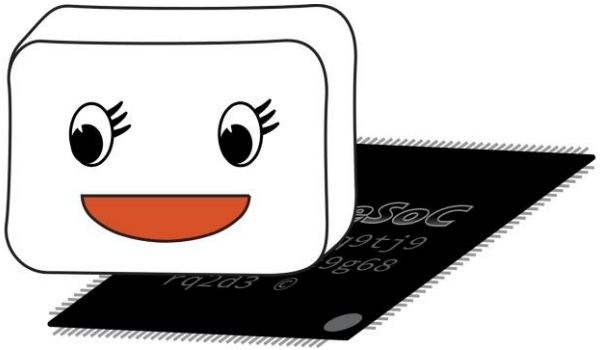
Core description files can be grouped into core libraries for easier management.



What is a core?

```
...
targets:
  nexys_a7:
    default_tool: vivado
    filesets: [rtl, nexys_files]
    tools: [vivado: {part : xc7a100tcsg324-1}
    toplevel: corey_top
  tb:
    default_tool: modelsim
    filesets: [rtl, tb]
    toplevel: corey_tb
    tools:
      modelsim:
        vlog_options: [-timescale=1ns/1ns]
      xsim:
        xelab_options: [--timescale
```

Targets describes ways to use the core and its dependencies. Some targets, like a simulation target, can support several tools.



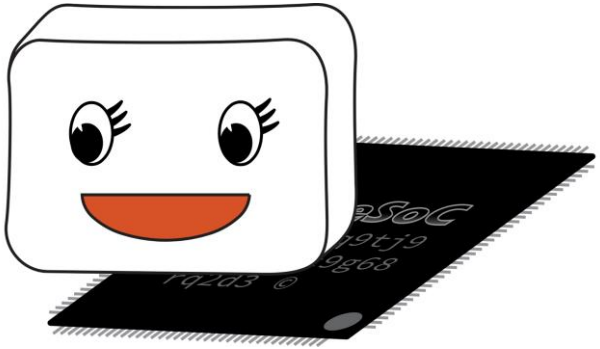
FUSESOC RUN…

… --target=sim ---tool=verilator corey

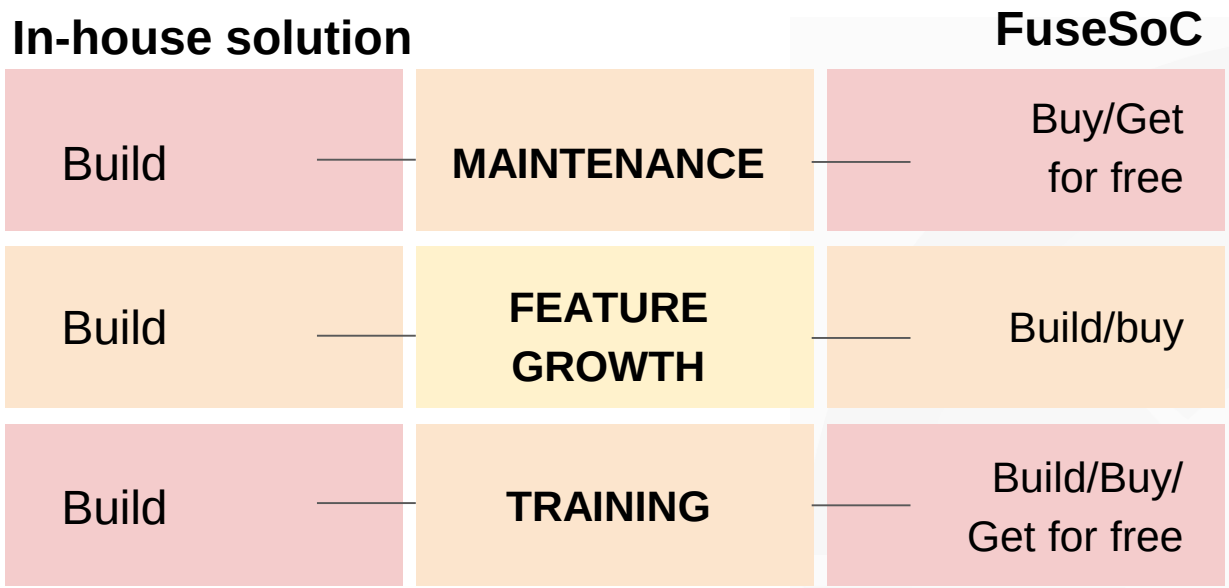
… --target=nexys_a7 corey

… --target=de0_nano corey

… --target=sim --tool=icarus corey



Build or buy (or get for free)



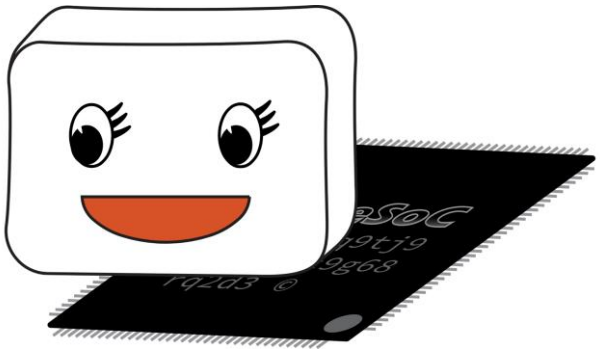
Existing open source libraries

fusesoc-cores	77
orpsoc-cores	100
openpiton	66
opentitan	129
optimsoc	102

+ many smaller libraries e.g. CVA6, picorv32, PULP, serv, microwatt, SweRV

+ many proprietary libraries

Most prominent open source silicon projects already use or have started looking at using FuseSoC



Tomorrow

- Increased adoption
- Expanded base library
- Additional EDA tool support
- Public package pool (à la pypi)

Future

- Industry standard
- World-class documentation

Future

- Industry standard
- World-class domination

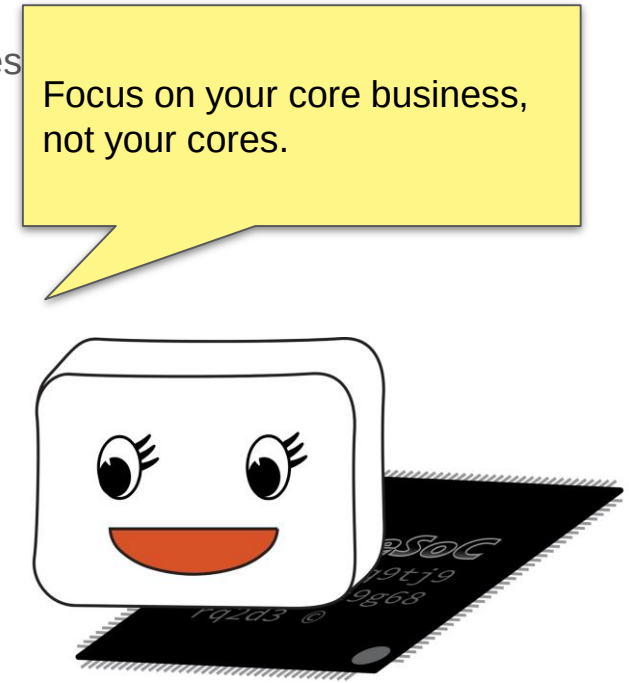
Why use FuseSoc?

Increase reuse

- Target different tools and devices with the same core description files
- Share cores between different working groups
- Reuse cores between projects

Reduce cost

- Lower maintenance and on-boarding costs
- Battle-proven base functionality that can be easily extended

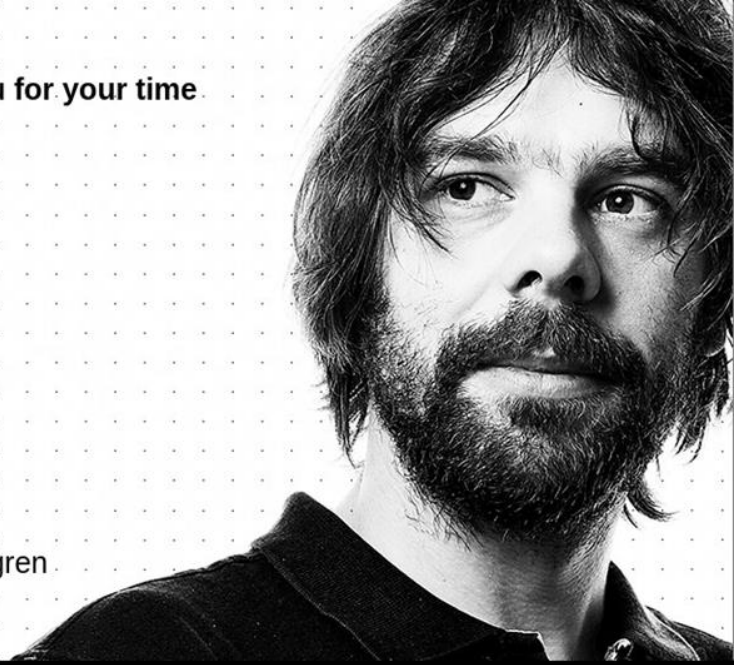


SERV

Observer

CORE SCOBE

Thank you for your time



https://github.com/olofk/serv
https://github.com/olofk/observer
https://github.com/olofk/corescore

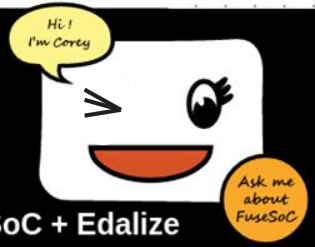
https://www.linkedin.com/in/olofkindgren

https://twitter.com/OlofKindgren

qamcom

RISC-V® AMBASSADOR

https://riscv.org/risc-v-ambassadors
https://www.qamcom.com



FuseSoC + Edalize

https://github.com/olofk/fusesoc
https://github.com/olofk/edalize

FOSSi Foundation

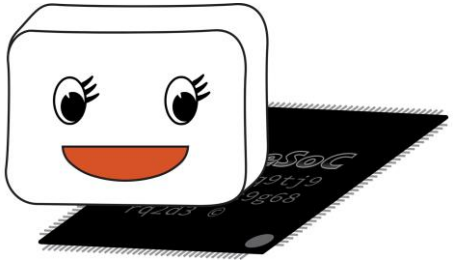
CHIPS ALLIANCE

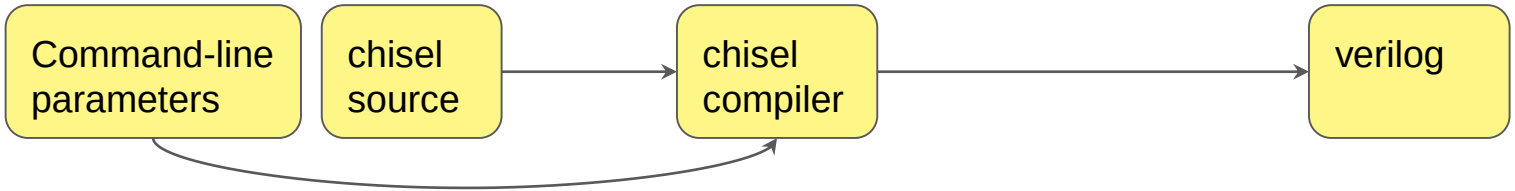
https://fossi-foundation.org
https://chipsalliance.org

Backup slides

Core description files (generators)

EDA tools only speak (system)verilog (and VHDL). All other file types must be generated before being sent to the EDA tool

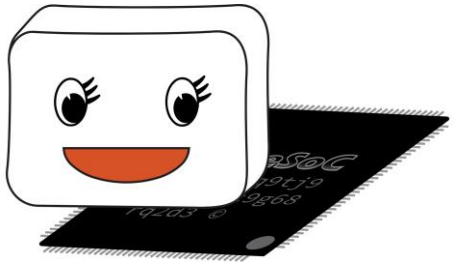


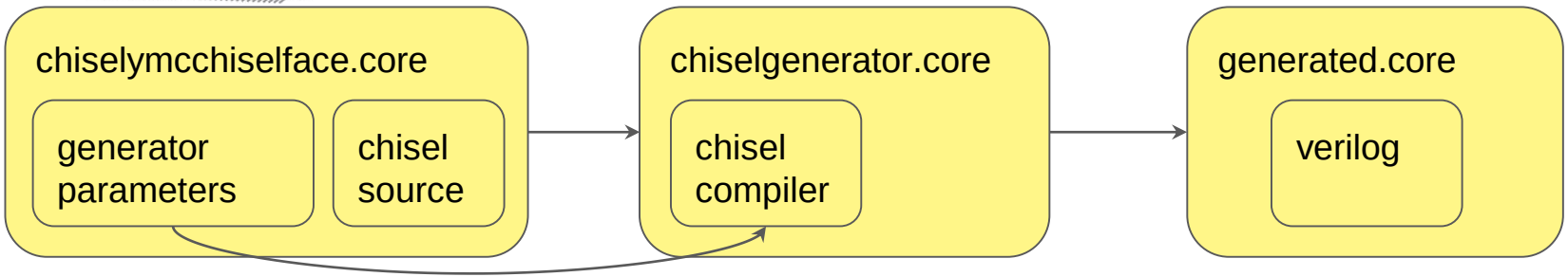


Core description files (generators)

EDA tools only speak (system)verilog (and VHDL). All other file types must be generated before being sent to the EDA tool.

Generators can be shared between cores

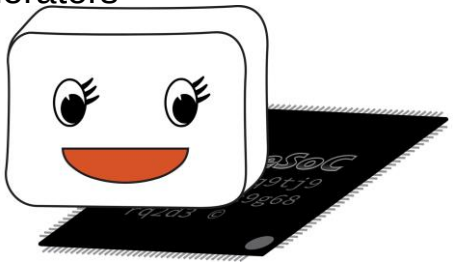


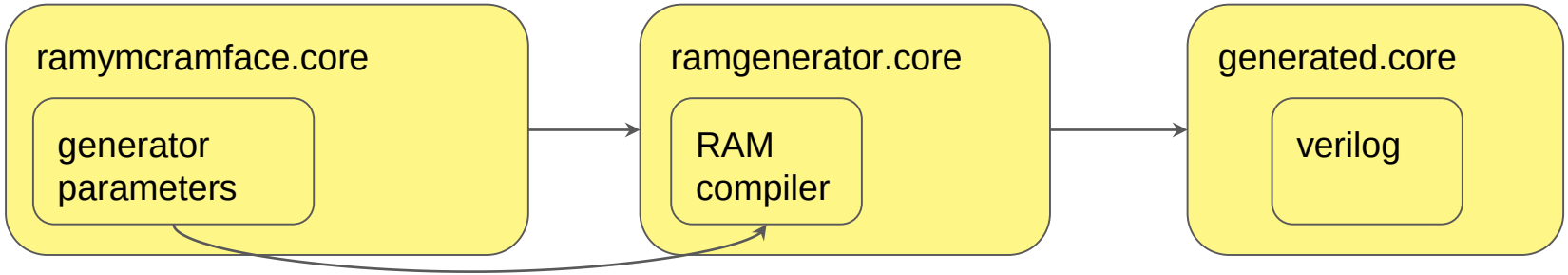


Core description files (generators)

Other examples:

C => $readmemh files
IP-XACT toplevels => verilog
CPU configuration => VHDL/SV packages/defines
memory maps => interconnect
generators





Core description files (the other stuff)

Parameters

- Tool-agnostic descriptions of `define, parameters, plusargs, generics…
- Controllable from command-line

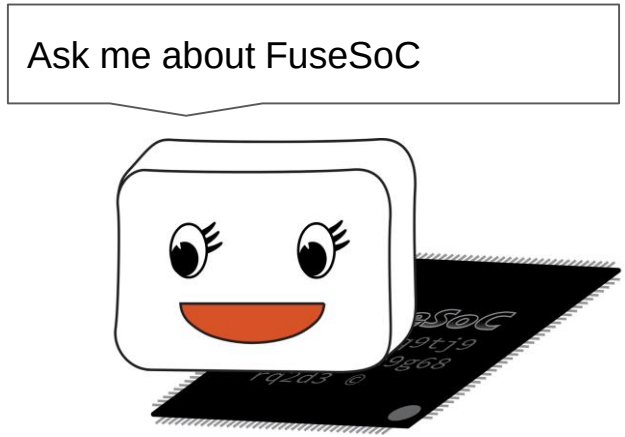
Script hooks

- Inject custom scripts before and after setup, build and run

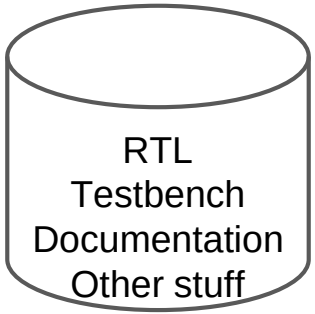
VPI

- Tool-agnostic way of compiling VPI libraries
- No DPI yet (but coming)

Probably more stuff I forgot



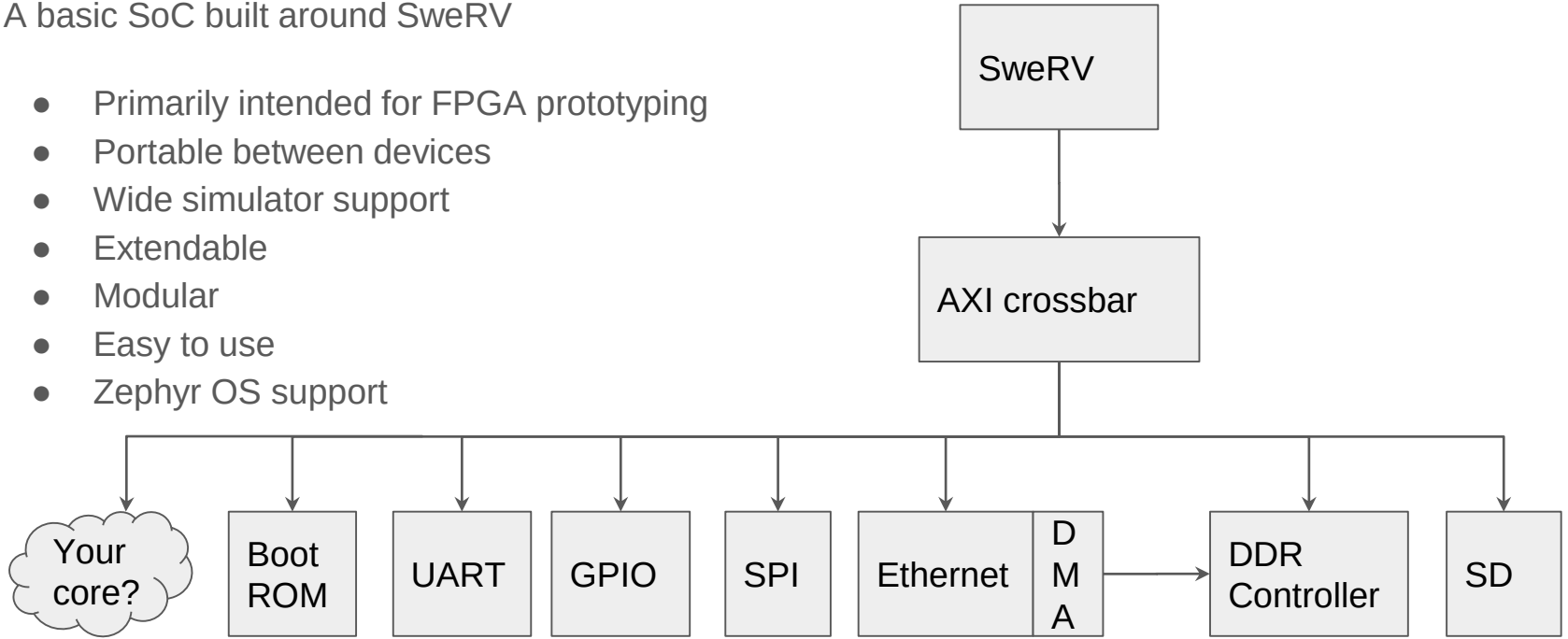
SweRV



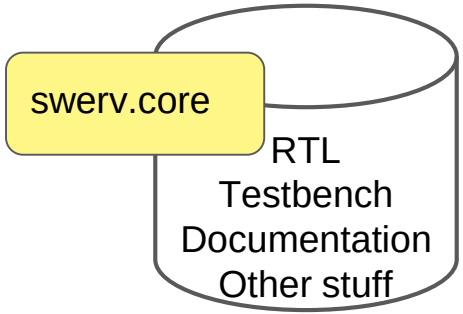
SweRVolf

A basic SoC built around SweRV

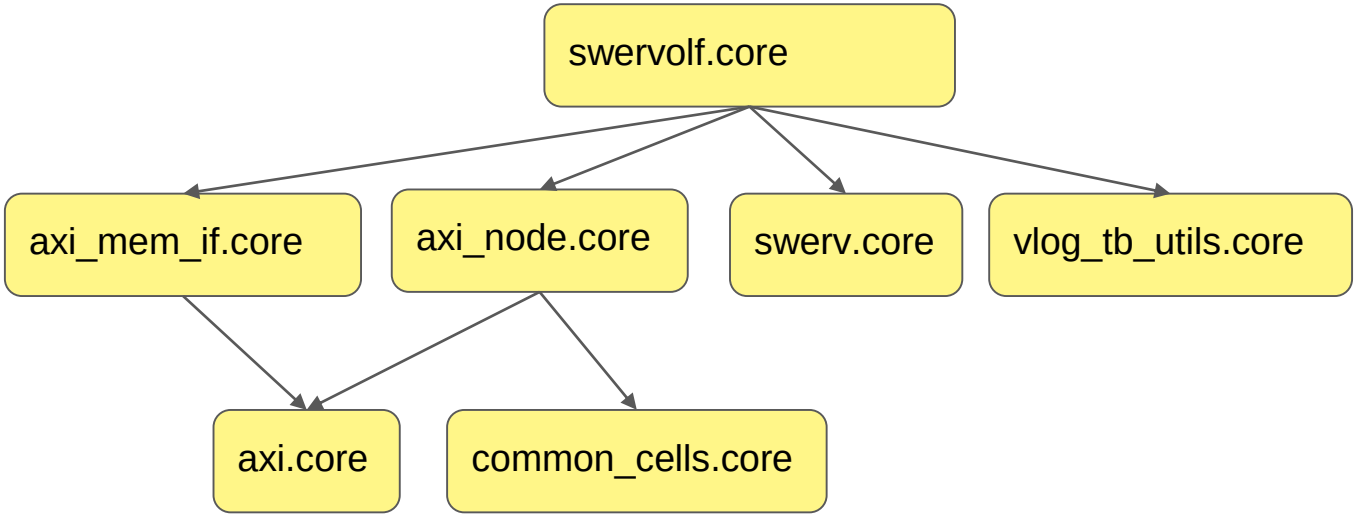
- Primarily intended for FPGA prototyping
- Portable between devices
- Wide simulator support
- Extendable
- Modular
- Easy to use
- Zephyr OS support



SweRV



SweRVolf dependencies



Core description files (generators)