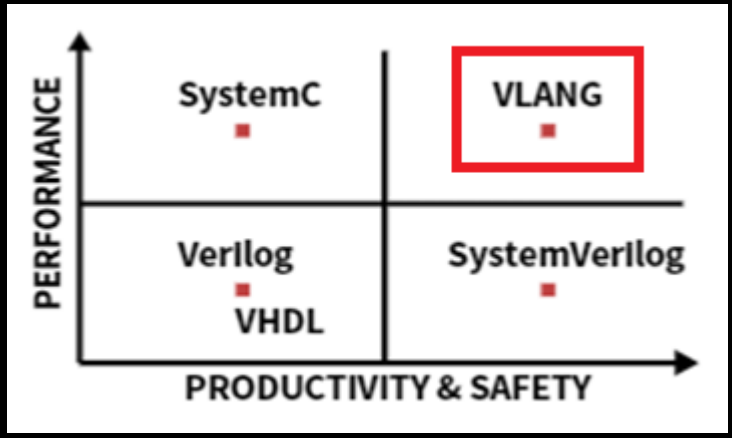
DConf Online 2022

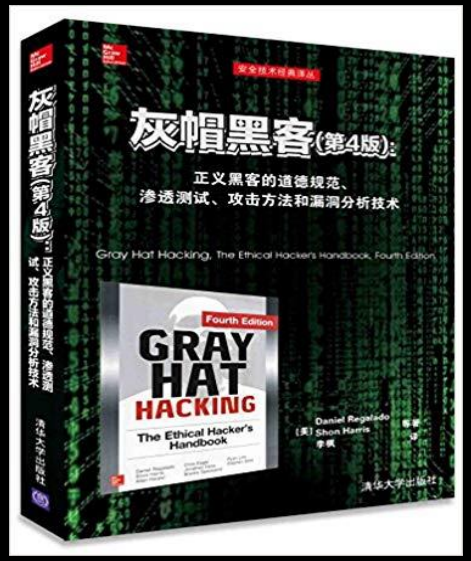
-based
Next Generation Verification Language



Feng Li (李枫)
hkli2013@126.com
Dec 17, 2022

Who Am I

- An indie developer from China
- The main translator of the book «Gray Hat Hacking The Ethical Hacker's Handbook, Fourth Edition» (ISBN：9787302428671) & «Linux Hardening in Hostile Networks, First Edition» (ISBN: 9787115544384)



- Pure software developmement for ~15 years(~11 years on Mobile dev)
- Actively participating Open Source Communities
- https://github.com/XianBeiTuoBaFeng2015/MySlides
- Recently, focus on infrastructure of Cloud/Edge Computing, AI, IoT, Programming Languages & Runtimes, Network, Virtualization, RISC-V, EDA, 5G/6G…

Agenda

I. Background
- Tech Stack
- Testbed

II. Architecture & Design
- Vlang
- ESDL
- EUVM
- Summary

III. Vlang on ARM
- Porting to RPi4
- More bug fixing and testing

IV. The future

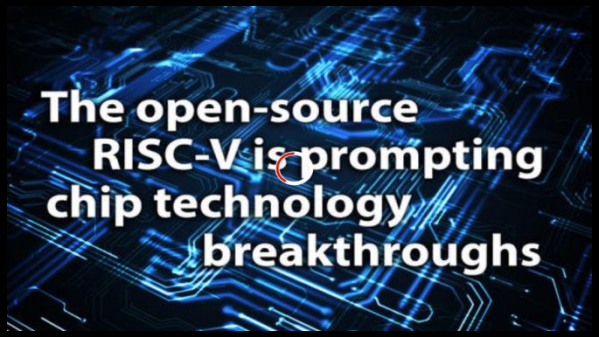
V. Wrap-up

I. Background

1) Tech Stack

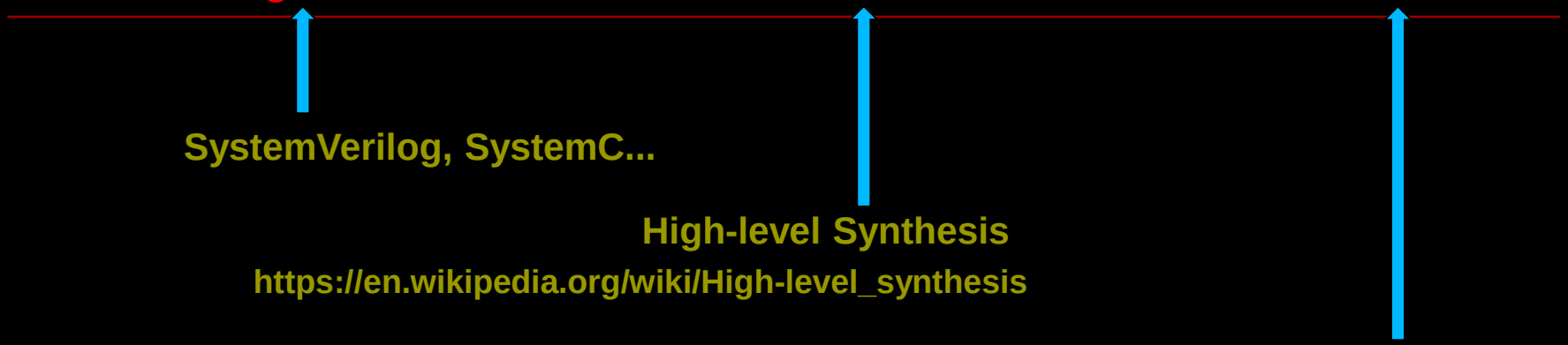
1.1 FOSS EDA

- https://en.wikipedia.org/wiki/Comparison_of_EDA_software
- https://semiwiki.com/wikis/industry-wikis/eda-open-source-tools-wiki/
- https://fossi-foundation.org/
- https://ieeexplore.ieee.org/document/9398963
- https://ieeexplore.ieee.org/document/9398960
- https://ieeexplore.ieee.org/document/9336682
- https://ieeexplore.ieee.org/document/9105619
- …



1.2 Evolution of HDLs

- https://en.wikipedia.org/wiki/Hardware_description_language
- https://hdl.github.io/awesome/items/
- Verilog/VHDL → HLS → eDSL



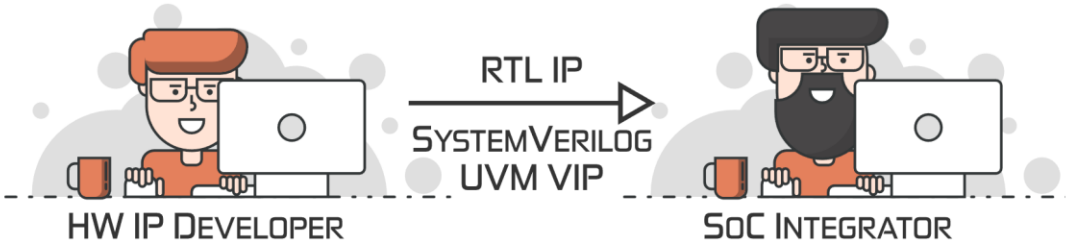
Typical eDSLs

- Haskell as host
  Bluespec, Clash...
- Scala as host
  Chisel, SpinalHDL...
- Python as host
  Amaranth/FHDL, PyGears...
- Finally convert to Verilog/VHDL
  https://en.wikipedia.org/wiki/Source-to-source_compiler

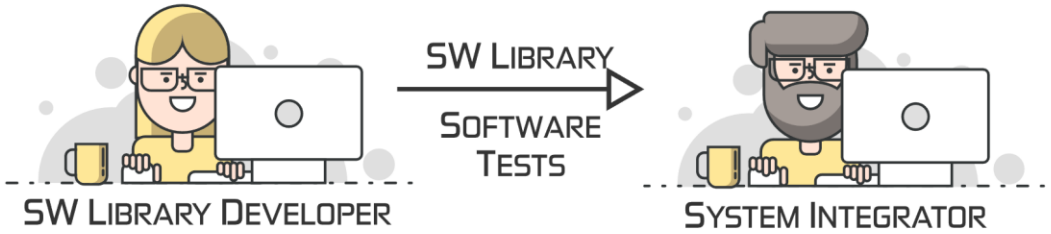
1.3 HW Verification

What is it

How does it impact the Hardware Verification domain? Traditionally, hardware and software are designed and developed in isolation. In an ASIC hardware design scenario, an IP team codes the RTL, and verifies it using SystemVerilog powered UVM testbenches before handing off the IP to the SoC integration team. An essential part of the IP -> SoC handoff is the UVM test suit that is required to be run at the SoC or at a subsystem level to make sure that the IP integration is seamless.



A similar work-flow is undertaken by software teams. A software IP (or library) developer passes on a set of tests to the application development team to make sure that the SW IP works without glitches in the application/system development environment.

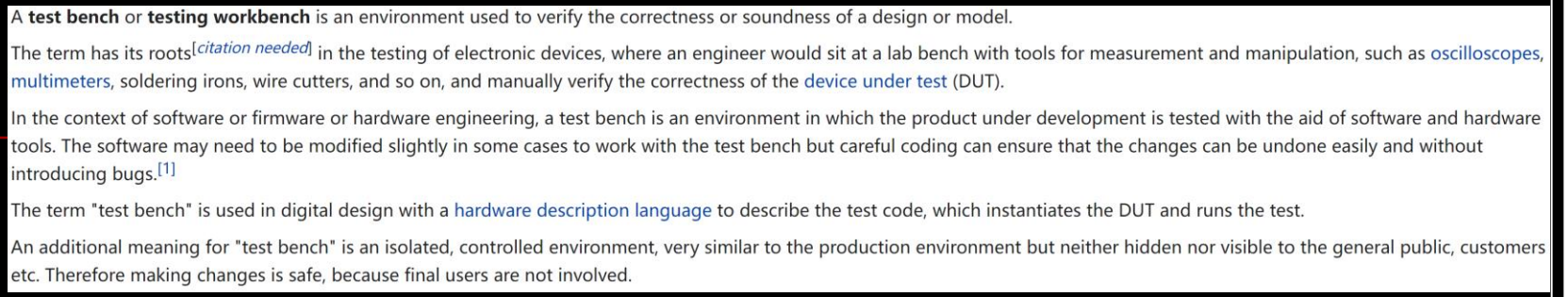


FPGA based Hardware Accelerator technology requires a complete re-look at how verification is done today, and how it needs to evolve. Since a hardware accelerator integrates tightly with the processor, a lot more hardware-software coverification is obviously required.

Source: http://uvm.io/blog/2019/04/accelerated-uvm/

Terminologies

- **https://en.wikipedia.org/wiki/Test_bench**



A **test bench** or **testing workbench** is an environment used to verify the correctness or soundness of a design or model.

The term has its roots[*citation needed*] in the testing of electronic devices, where an engineer would sit at a lab bench with tools for measurement and manipulation, such as oscilloscopes, multimeters, soldering irons, wire cutters, and so on, and manually verify the correctness of the device under test (DUT).

In the context of software or firmware or hardware engineering, a test bench is an environment in which the product under development is tested with the aid of software and hardware tools. The software may need to be modified slightly in some cases to work with the test bench but careful coding can ensure that the changes can be undone easily and without introducing bugs.[1]

The term "test bench" is used in digital design with a hardware description language to describe the test code, which instantiates the DUT and runs the test.

An additional meaning for "test bench" is an isolated, controlled environment, very similar to the production environment but neither hidden nor visible to the general public, customers etc. Therefore making changes is safe, because final users are not involved.

- **https://docs.cocotb.org/en/stable/glossary.html#term-VPI**
- **…**

Hardware Verification Language (HVL)

- **https://en.wikipedia.org/wiki/Hardware_verification_language**
- **https://en.wikipedia.org/wiki/SystemVerilog**
- **…**

1.3.1 Methodologies

- https://github.com/ben-marshall/awesome-open-hardware-verification
- …

1.3.1.1 UVM

- https://en.wikipedia.org/wiki/Universal_Verification_Methodology

The **Universal Verification Methodology** (UVM) is a standardized methodology for verifying integrated circuit designs. UVM is derived mainly from the OVM (Open Verification Methodology) which was, to a large part, based on the eRM (e Reuse Methodology) for the e Verification Language developed by Verisity Design in 2001. The UVM class library brings much automation to the SystemVerilog language such as sequences and data automation features (packing, copy, compare) etc., and unlike the previous methodologies developed independently by the simulator vendors, is an Accellera standard with support from multiple vendors: Aldec, Cadence, Mentor Graphics, Synopsys, Xilinx Simulator(XSIM).

History [edit]

In December 2009, a technical subcommittee of Accellera — a standards organization in the electronic design automation (EDA) industry — voted to establish the UVM and decided to base this new standard on the Open Verification Methodology (OVM-2.1.1),[1] a verification methodology developed jointly in 2007 by Cadence Design Systems and Mentor Graphics.

On February 21, 2011, Accellera approved the 1.0 version of UVM.[2] UVM 1.0 includes a Reference Guide, a Reference Implementation in the form of a SystemVerilog base class library, and a User Guide.[2]

Definitions [edit]

- Agent - A container that emulates and verifies DUT devices
- Blocking - An interface that blocks tasks from other interfaces until it completes
- DUT - Device under test, what you are actually testing
- DUV - Device Under Verification
- Component - A portion of verification intellectual property that has interfaces and functions.
- Transactor - see component
- Verification Environment Configuration - those settings in the DUT and environment that are alterable while the simulation is running
- VIP - verification intellectual property

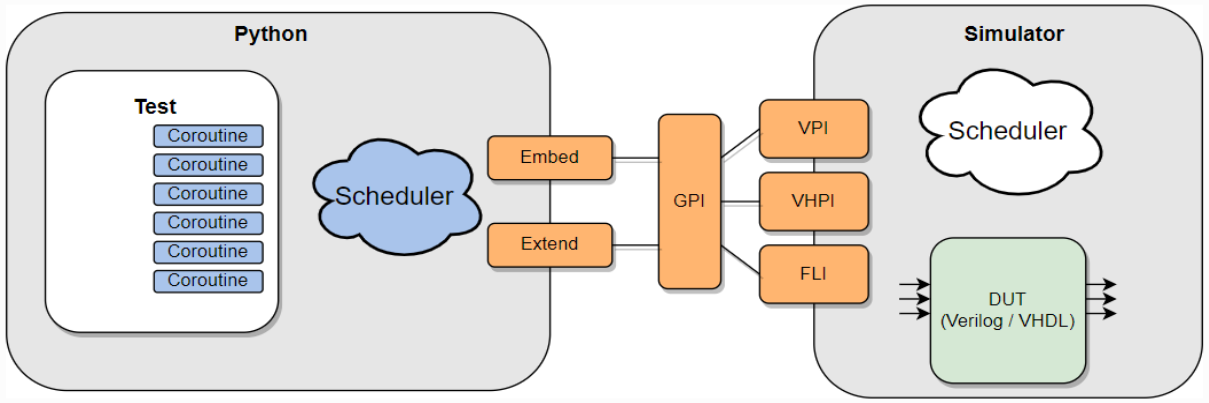
…

- https://ieeexplore.ieee.org/document/9195920
- https://www.accellera.org/community/uvm/
- The latest version of UVM is 1.2, and 2.0 is on the way.
  https://www.accellera.org/downloads/standards/uvm
- …

1.3.1.2 Cocotb

- https://www.cocotb.org/
  A Coroutine based Cosimulation TestBench environment for verifying VHDL and Verilog RTL using Python.
- https://github.com/cocotb/cocotb
- How it works

A typical cocotb testbench requires no additional RTL code. The Design Under Test (DUT) is instantiated as the toplevel in the simulator without any wrapper code. cocotb drives stimulus onto the inputs to the DUT (or further down the hierarchy) and monitors the outputs directly from Python. Note that cocotb can not instantiate HDL blocks - your DUT must be complete.



A test is simply a Python function. At any given time either the simulator is advancing time or the Python code is executing. The `await` keyword is used to indicate when to pass control of execution back to the simulator. A test can spawn multiple coroutines, allowing for independent flows of execution.

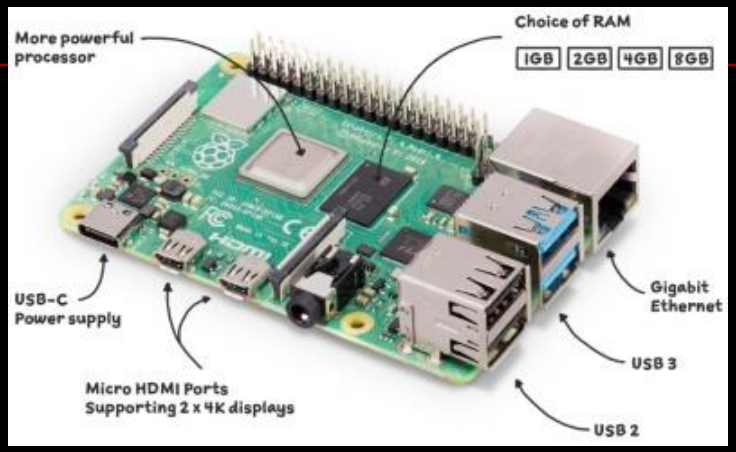
Source: https://docs.cocotb.org/en/stable/

- https://www.reddit.com/r/FPGA/comments/v8fd63/cocotb_in_python_vs_uvm_in_systemverilog/
- ...

2) Testbed

2.1 Raspberry Pi 4(8GB LPDDR4) with Fedora 37

- HW env



SW env

```
[mydev@fedora /]$ uname -a
Linux fedora 6.0.11-300.fc37.aarch64 #1 SMP PREEMPT_DYNAMIC Fri Dec 2 20:14:38 UTC 2022 aarch64 aarch64 aarch64 GNU/Linux
[mydev@fedora /]$
[mydev@fedora /]$ free -m
               total        used        free      shared  buff/cache   available
Mem:            7826         563        2011          31        5251        6876
Swap:           7825           0        7825
[mydev@fedora /]$
```

```
[mydev@fedora /]$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/aarch64-redhat-linux/12/lto-wrapper
Target: aarch64-redhat-linux
Configured with: ../configure --enable-bootstrap --enable-languages=c,c++,fortran,objc,obj-c++,ada,go,d,lto --prefix=/usr --mandir=/usr/share/man --infodi
r=/usr/share/info --with-bugurl=http://bugzilla.redhat.com/bugzilla --enable-shared --enable-threads=posix --enable-checking=release --enable-multilib --w
ith-system-zlib --enable-__cxa_atexit --disable-libunwind-exceptions --enable-gnu-unique-object --enable-linker-build-id --with-gcc-major-version-only --e
nable-libstdcxx-backtrace --with-linker-hash-style=gnu --enable-plugin --enable-initfini-array --with-isl=/builddir/build/BUILD/gcc-12.2.1-20221121/obj-aa
rch64-redhat-linux/isl-install --enable-gnu-indirect-function --build=aarch64-redhat-linux --with-build-config=bootstrap-lto --enable-link-serialization=1
Thread model: posix
Supported LTO compression algorithms: zlib zstd
gcc version 12.2.1 20221121 (Red Hat 12.2.1-4) (GCC)
[mydev@fedora /]$
[mydev@fedora /]$ clang -v
clang version 15.0.4 (Fedora 15.0.4-1.fc37)
Target: aarch64-redhat-linux-gnu
Thread model: posix
InstalledDir: /usr/bin
Found candidate GCC installation: /usr/bin/../lib/gcc/aarch64-redhat-linux/12
Selected GCC installation: /usr/bin/../lib/gcc/aarch64-redhat-linux/12
Candidate multilib: .;@m64
Selected multilib: .;@m64
[mydev@fedora /]$
```

```
[mydev@fedora /]$ ldc2 -v
binary    /opt/MyWorkSpace/MyProjs/Languages/D/Compiler/LDC/Official/ldc-master/build/Install/bin/ldc2
version   1.30.0-git-5fd86e5 (DMD v2.100.1, LLVM 15.0.4)
config    /opt/MyWorkSpace/MyProjs/Languages/D/Compiler/LDC/Official/ldc-master/build/Install/etc/ldc2.conf (aarch64-redhat-linux-gnu)
Error: No source files
[mydev@fedora /]$
[mydev@fedora /]$ dub --version
DUB version 1.23.0, built on Jul 27 2022
[mydev@fedora /]$
```

2.1.1 Fedora

- A Linux distribution developed by the community-supported Fedora Project which is sponsored primarily by Red Hat.
- https://en.wikipedia.org/wiki/Fedora_(operating_system)
- https://getfedora.org/
- https://alt.fedoraproject.org/alt/
- https://spins.fedoraproject.org/
- https://fedoraproject.org/wiki/Architectures/ARM
- https://fedoramagazine.org/
- https://silverblue.fedoraproject.org/
- https://fedoraproject.org/wiki/Changes/RaspberryPi4
- Developer friendly!

2.1.1.1 Fedora 37

- https://fedoraproject.org/wiki/Releases/37/ChangeSet
- https://www.phoronix.com/news/Fedora-37-Released

Raspberry Pi 4

- **Fedora 37 To Offer Official Support On Raspberry Pi 4 Devices**
  https://www.phoronix.com/news/Raspberry-Pi-4-Fedora-37

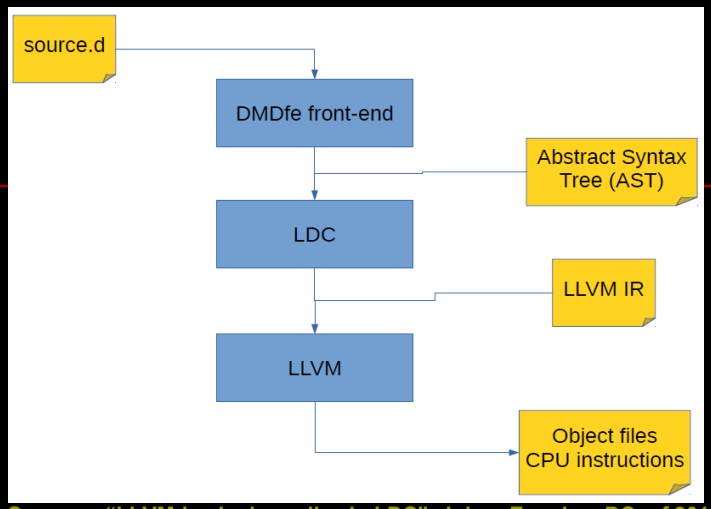
A month ago there was the Fedora 37 change proposal for Fedora to officially support the Raspberry Pi 4, including its accelerated Broadcom graphics and to better advertise Fedora for the Raspberry Pi. The Fedora Engineering and Steering Committee (FESCo) has now signed off on this "official" support for the Raspberry Pi 4.

The Raspberry Pi 4 to date hasn't been a significant focus for Fedora Workstation due to various patches not being upstreamed-- most notably, waiting on the open-source 3D graphics bits to be upstreamed in the kernel. Now though that those upstream bits are coming together, Fedora 37 will be focusing on advertising its support for the Raspberry Pi 4 Model B as well as the Raspberry Pi 400 and Raspberry Pi CM4 compute module.

With the open-source OpenGL and Vulkan support, these latest Raspberry Pi boards are suitable for Fedora Workstation usage. The latest milestones there are V3DV just having crossed Vulkan 1.2 and Raspberry Pi 4 V3D DRM/KMS driver support in Linux 6.0. However, there still are upstream issues surrounding WiFi support for the Raspberry Pi 400, the Raspberry Pi CM4 not necessarily being very suitable for desktop use but more edge/IoT/embedded, and some device support such as around audio may be problematic.

> ***The Raspberry Pi 4 is a widely available, reasonably priced device. It has worked well in Fedora for some time in IoT and Server use cases, and now with a fully accelerated graphics stack available it's a great device from a price-per-performance perspective, and it has a wide ecosystem, so fully supporting this in Fedora makes a compelling case.***

2.1.2 LDC



Source: "LLVM-backed goodies in LDC", Johan Engelen, DConf 2018.

- **Artifacts of the latest official LDC release for AArch64**

ldc-1.30.0-src.zip	10.3 MB	21 Jul 2022
ldc2-1.30.0-android-aarch64.tar.xz	30.1 MB	21 Jul 2022
ldc2-1.30.0-android-armv7a.tar.xz	27.6 MB	21 Jul 2022
ldc2-1.30.0-freebsd-x86_64.tar.xz	17.3 MB	21 Jul 2022
ldc2-1.30.0-linux-aarch64.tar.xz	51.3 MB	21 Jul 2022
ldc2-1.30.0-linux-x86_64.tar.xz	62.3 MB	21 Jul 2022
ldc2-1.30.0-osx-arm64.tar.xz	65.5 MB	21 Jul 2022
ldc2-1.30.0-osx-universal.tar.xz	136 MB	21 Jul 2022
ldc2-1.30.0-osx-x86_64.tar.xz	72.8 MB	21 Jul 2022
ldc2-1.30.0-windows-multilib.7z	50.1 MB	21 Jul 2022
ldc2-1.30.0-windows-multilib.exe	53.1 MB	21 Jul 2022
ldc2-1.30.0-windows-x64.7z	36 MB	21 Jul 2022
ldc2-1.30.0-windows-x86.7z	35.1 MB	21 Jul 2022
ldc2-1.30.0.sha256sums.txt	1.31 KB	21 Jul 2022

https://github.com/ldc-developers/ldc/releases

```
[mydev@fedora LDC]$ tree -L 2 1.30.0
1.30.0
├── bin
│   ├── ddemangle
│   ├── dub
│   ├── dustmite
│   ├── ldc2
│   ├── ldc-build-runtime
│   ├── ldc-profdata
│   ├── ldc-prune-cache
│   ├── ldmd2
│   ├── rdmd
│   └── reggae
├── etc
│   ├── bash_completion.d
│   └── ldc2.conf
├── import
│   ├── __builtins.di
│   ├── core
│   ├── etc
│   ├── importc.h
│   ├── ldc
│   ├── object.d
│   └── std
├── lib
│   ├── ldc_rt.dso.o
│   ├── libdruntime-ldc.a
│   ├── libdruntime-ldc-debug.a
│   ├── libdruntime-ldc-debug-shared.so
│   ├── libdruntime-ldc-debug-shared.so.100
│   ├── libdruntime-ldc-debug-shared.so.100.1
│   ├── libdruntime-ldc-lto.a
│   ├── libdruntime-ldc-shared.so
│   ├── libdruntime-ldc-shared.so.100
│   ├── libdruntime-ldc-shared.so.100.1
│   ├── libldc_rt.asan.a
│   ├── libldc_rt.builtins.a
│   ├── libldc_rt.fuzzer.a
│   ├── libldc_rt.lsan.a
│   ├── libldc_rt.msan.a
│   ├── libldc_rt.profile.a
│   ├── libldc_rt.tsan.a
│   ├── libldc_rt.xray.a
│   ├── libldc_rt.xray-basic.a
│   ├── libldc_rt.xray-fdr.a
│   ├── libldc_rt.xray-profiling.a
│   ├── libphobos2-ldc.a
│   ├── libphobos2-ldc-debug.a
│   ├── libphobos2-ldc-debug-shared.so
│   ├── libphobos2-ldc-debug-shared.so.100
│   ├── libphobos2-ldc-debug-shared.so.100.1
│   ├── libphobos2-ldc-lto.a
│   ├── libphobos2-ldc-shared.so
│   ├── libphobos2-ldc-shared.so.100
│   ├── libphobos2-ldc-shared.so.100.1
│   └── LLVMgold-ldc.so
├── LICENSE
└── README
```

build LDC from src on RPi4

https://wiki.dlang.org/Building_LDC_from_source

Run the following commands to configure and build LDC and its default libraries:

```
# Generate Ninja build files
mkdir build-ldc && cd build-ldc # using a fresh new build dir is highly recommended whenever re-invoking CMake
cmake -G Ninja ../ldc \
  -DCMAKE_BUILD_TYPE=Release \
  -DCMAKE_INSTALL_PREFIX=$PWD/../install-ldc \
  -DLLVM_ROOT_DIR=$PWD/../install-llvm \                    # not needed if using a distro LLVM package
  -DD_COMPILER=$PWD/../ldc2-1.17.0-linux-x86_64/bin/ldmd2 # not needed if host D compiler (ldmd2/dmd/gdmd) is found in PATH or if building 0.17 ltsmaster

# Build; use -j<N> to limit parallelism if running out of memory. The binaries end up in bin/.
ninja
# Optional: install LDC to the CMAKE_INSTALL_PREFIX directory above
ninja install
```

On master branch, last commit: 5fd86e59841ccbcf52969412dd9efed3f2bd2c76

outputs:

```
[mydev@fedora ldc-master]$ pwd
/opt/MyWorkSpace/MyProjs/Languages/D/Compiler/LDC/Official/ldc-master
```

```
[mydev@fedora ldc-master]$ tree -L 1 build/Install/
build/Install/
├── bin
├── etc
├── include
└── lib
```

```
[mydev@fedora ldc-master]$ tree build/Install/bin
build/Install/bin
├── ldc2
├── ldc-build-runtime
├── ldc-profdata
├── ldc-prune-cache
└── ldmd2
```

```
[mydev@fedora ldc-master]$ tree build/Install/lib
build/Install/lib
├── ldc_rt.dso.o
├── libdruntime-ldc.a
├── libdruntime-ldc-debug.a
├── libdruntime-ldc-debug-shared.so → libdruntime-ldc-debug-shared.so.100
├── libdruntime-ldc-debug-shared.so.100 → libdruntime-ldc-debug-shared.so.100.1
├── libdruntime-ldc-debug-shared.so.100.1
├── libdruntime-ldc-shared.so → libdruntime-ldc-shared.so.100
├── libdruntime-ldc-shared.so.100 → libdruntime-ldc-shared.so.100.1
├── libdruntime-ldc-shared.so.100.1
├── libphobos2-ldc.a
├── libphobos2-ldc-debug.a
├── libphobos2-ldc-debug-shared.so → libphobos2-ldc-debug-shared.so.100
├── libphobos2-ldc-debug-shared.so.100 → libphobos2-ldc-debug-shared.so.100.1
├── libphobos2-ldc-debug-shared.so.100.1
├── libphobos2-ldc-shared.so → libphobos2-ldc-shared.so.100
├── libphobos2-ldc-shared.so.100 → libphobos2-ldc-shared.so.100.1
└── libphobos2-ldc-shared.so.100.1
```

sudo dnf install dub…

II. Architecture & Design

1) Vlang

1.1 Overview

- https://github.com/coverify-org/vlang-docs

Next Generation Verification Language that base on 

Vlang Features at a Glance

Multicore Vlang enables concurrent programming. End user can fine-tune the number of concurrently running threads at module level. Vlang also enables concurrency at a higher abstraction by allowing multiple simulators running in parallel.

Constrained Randomization Full blown and efficient. Concurrency enabled.

UVM Compliance Word-to-word translation of SystemVerilog UVM. More efficient and user-friendly due to generic programming.

Object Oriented Programming Support for function/operator overloading.

Safety and Productivity Automatic Garbage Collection. Exception Handling. Unittests.

Systems Programming Allows low level access to hardware resources. Allows embedded assembly language.

Interface with other Languages Full blown C++ interface. VHPI/VPI bindings with VHDL and SystemVerilog.

Licensing Provided free under open source boost license. Vlang UVM library is available under Apache2 license.

Verification with Vlang

Even as the chip complexity keeps increasing, we continue to rely on same old RTL methodology to design our chips. As a result the abstraction gap between the design and the specification is increasing exponentially.

Vlang attempts to bridge this gap by providing you a high productivity and high performance verification environment.

Higher Productivity means that you take less time in building your verification infrastructure and higher performance means that your regression runs much faster.

If you have ESL as part of your SoC development flow, there are additional reasons that you should use Vlang to verify your ESL models. Vlang supports much better integration with C++ compared to SystemVerilog. Vlang is ABI compatible with C/C++. Vlang also allows you to call any method (including virtual methods) on C++ objects right from Vlang without any boilerplate code. In comparison SystemVerilog DPI interface is limited to C language. As a result any interface between SystemC and SystemVerilog tends to be highly inefficient.

Yet another advantage is that Vlang is free and open source just like your SystemC simulator.

Source: https://fdocuments.in/document/vlang-flyer.html?page=2 (Date Post 28-Dec-2015)

Initial Design

- Vlang is a DSL built on top of Dlang

Introduction to Next Generation Verification Language - Vlang

Puneet Goel and Sumit Adhikari
Coverify Systems Technology, India and NXP Semiconductors, Germany
email: puneet@coverify.com and sumit.adhikari@nxp.com

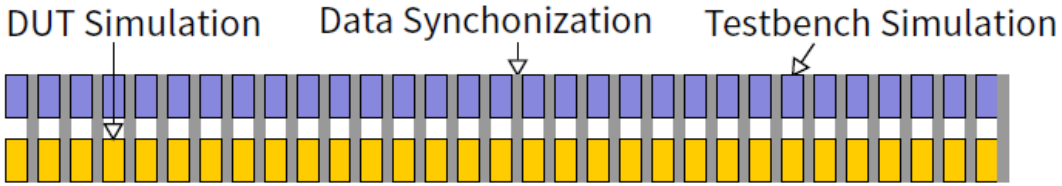
***Abstract*—IP/SoC verification is a fundamental problem in design cycle which needs support from a suitable and powerful language which must include the latest findings in computer science. State-of-the-art verification languages are decade old, closed source, not software domain, single paradigm, type unsafe, advocate code boilerplate, single core and not supportive to generic programming. This forces verification engineer to use decade old language concepts, using a HW domain language to build a software called test-bench using a single incomplete programming paradigm called object orientation. Furthermore, incomplete support for object orientation added with no support for generative programming forces the user to write redundancy which could be avoided. In this article we solve this age old verification problem by introducing a novel open source verification language called Vlang [1]. Vlang is built on top of D Programming Language [2], and consequently it inherits support for parallel, generic, generative, and functional programming paradigms in addition to Object Oriented Programming. Vlang is ABI compatible with C/C++, creates single executable with SystemC [3], [4] and integrates with System Verilog [5] seamlessly through VPI/DPI. Currently Vlang supports UVM-1.1d standard as in built methodology library.**

Source: https://dvcon-proceedings.org/wp-content/uploads/introduction-to-next-generation-verification-language-vlang.pdf (DVCon Europe 2014)

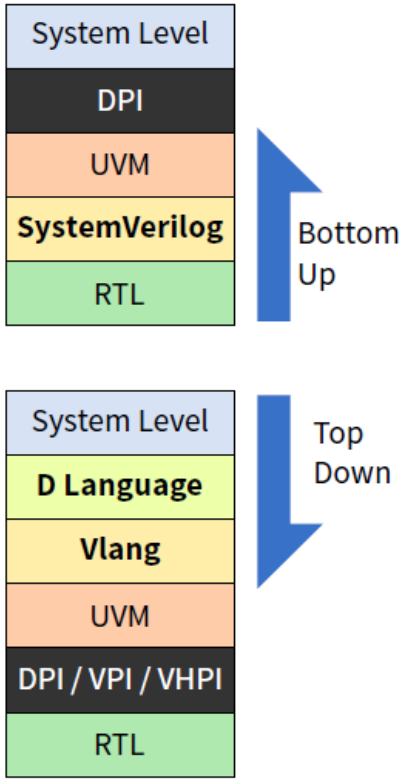
Top Down Verification Stack

System Verilog integrates tightly with RTL;
Vlang with System Level

- Vlang is built on top of D Programming Language which provides ABI compatibility with C/C++
- Vlang interfaces with RTL using DPI
  - DPI overhead is compensated by parallel execution of testbench
- **Vlang offers zero communication overhead when integrating with Emulation Platforms and with Virtual Platforms**

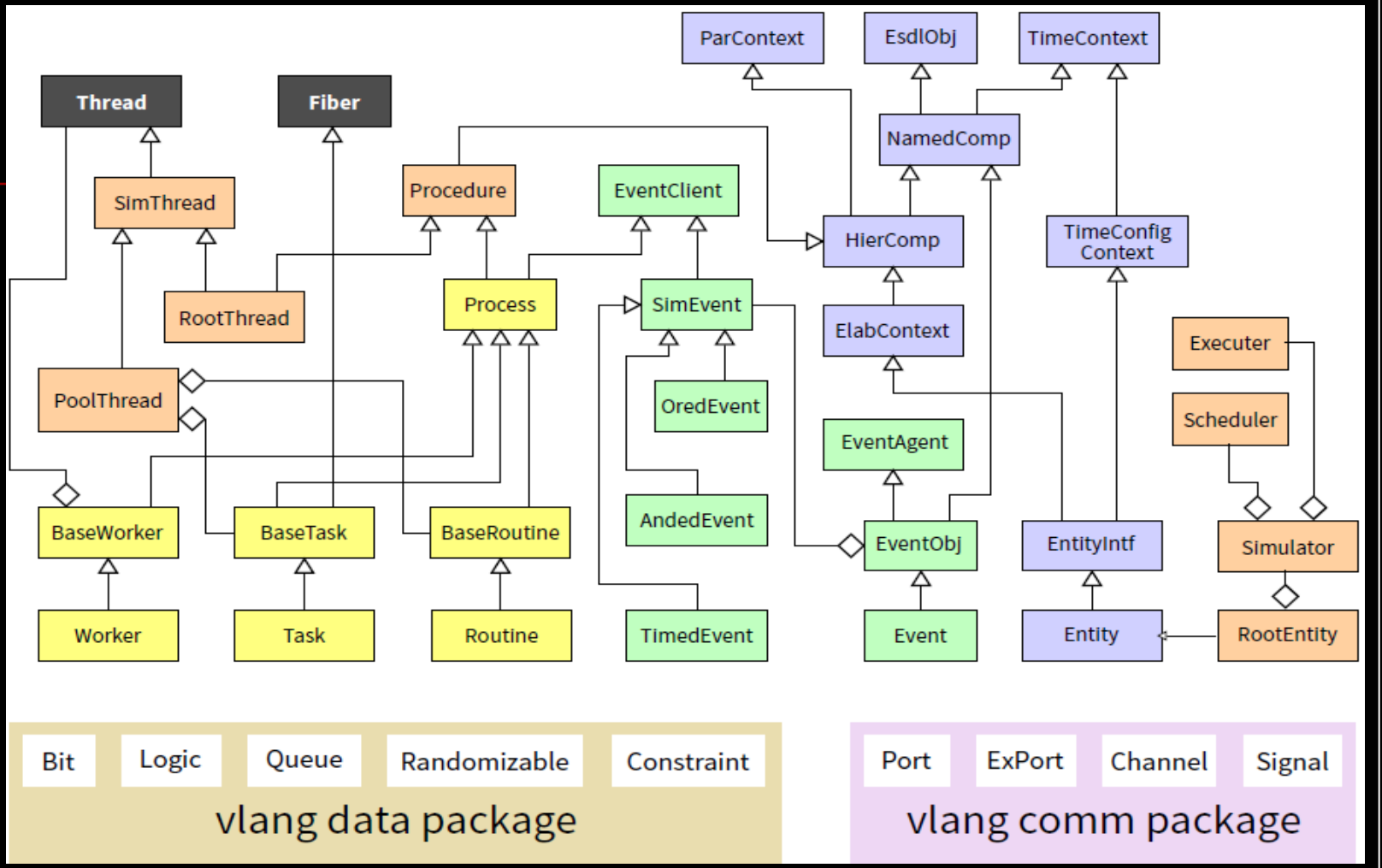


Vlang Cosimulation with Virtual/Emulation Platforms



Source: https://dvcon-proceedings.org/wp-content/uploads/introduction-to-next-generation-verification-language-vlang-presentation.pdf (DVCon Europe 2014)

Core Infrastructure



Source: https://dvcon-proceedings.org/wp-content/uploads/introduction-to-next-generation-verification-language-vlang-presentation.pdf (DVCon Europe 2014)

Processes

- Processes and Forks (and for that matter everything else) in Vlang is an object
  - You can pass an Event, a Process, or a Fork as an argument to a function
- Joining a fork can be done flexibly with the Fork object
- Forks and Processes can be suspended, disabled, aborted etc

```
void frop() {
  Fork zoo = fork
    ({ // fork1
       foo();
     },
     { // fork2
       bar();
     }).joinAny();
  // Some Code
  zoo.join();
  zoo.abortTree();
}
void foo() {
  auto proc = Process.self();
  proc.abortTree();
}
void bar() {
  // wait for fork branch
  Fork ff = Fork.self();
  ff.joinAny();
  // ....
}
```

Source: https://dvcon-proceedings.org/wp-content/uploads/introduction-to-next-generation-verification-language-vlang-presentation.pdf (DVCon Europe 2014)

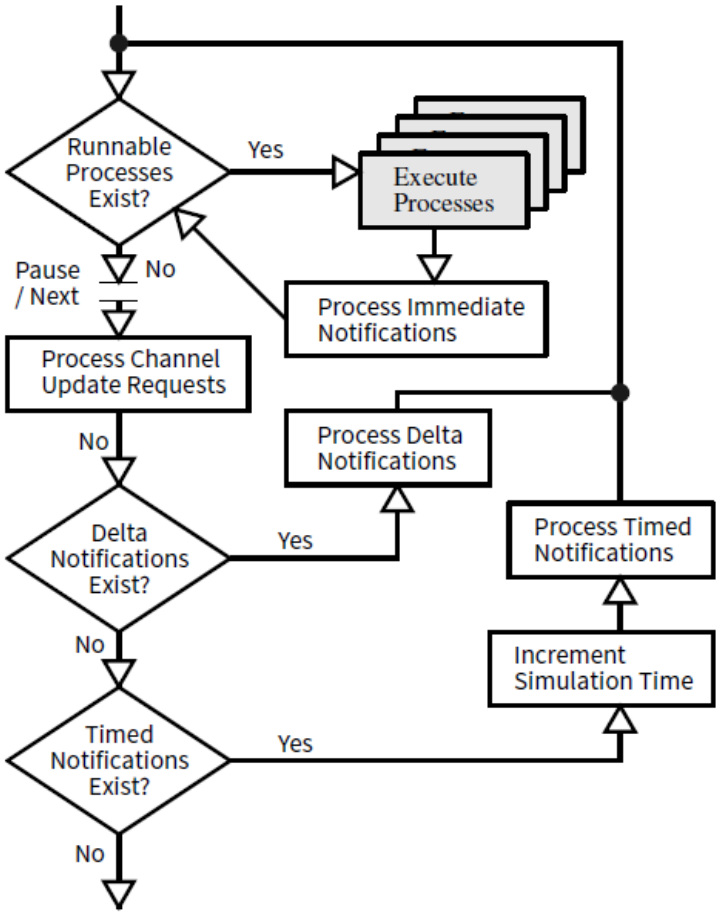
Constrained Randomization

- System Verilog style *Constrained Randomization*
- Class needs to be derived from `Randomizable`
- **`mixin Randomization` magic makes `randomize` polymorphic**
- Within UVM (and that is what matters) all the ugliness is gone
- Support for common arithmetic, logical and comparison operators
- Support for array and dynamic array randomization
- Support for `if-else`, `foreach`
  - Can be freely nested

```
class Foo: Randomizable {
  mixin Randomization;
  @rand!8 byte[] foo;
  @rand Logic!12 baz;
}
class Bar: Foo {
  mixin Randomization;
  @rand ubyte[8] bar;
  Constraint! q{
    foo.length > 2; // array
    baz[0..8] == 16;
  } cstFooLength;
  Constraint! q{
    foreach(i, f; bar) f <= i;
    foreach(i, f; foo) {
      if(i > 4) /*condition*/
        f + i < 32 && f > 16;
    }
  } cstFoo;
}
void main() {
  Foo randObj = new Bar();
  for (size_t i=0; i!=10; ++i) {
    randObj.randomize();
  }
}
```

Source: https://dvcon-proceedings.org/wp-content/uploads/introduction-to-next-generation-verification-language-vlang-presentation.pdf (DVCon Europe 2014)

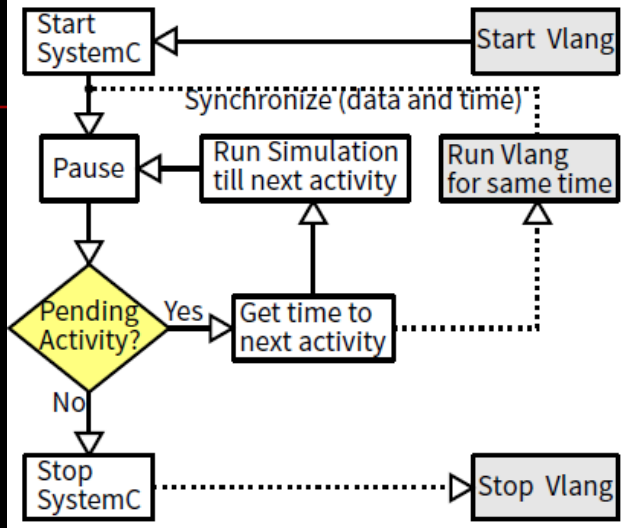
Multicore UVM



- Vlang simulator is multicore capable
- Vlang implementation of Multicore UVM takes advantage of the fact that there is minimal interaction between different `uvm_agents`
- Vlang provides an abstraction `ParContext` to manage parallelization
- The uvm constructs (like `uvm_objection`), that are shared between the components, are *synchronized* in the UVM base library implementation
  - In Vlang port of UVM, a `uvm_component` implements `ParContext`

Source: https://dvcon-proceedings.org/wp-content/uploads/introduction-to-next-generation-verification-language-vlang-presentation.pdf (DVCon Europe 2014)

Interfacing with SystemVerilog and SystemC



- Vlang simulator can be fully synchronized with SystemC and SystemVerilog

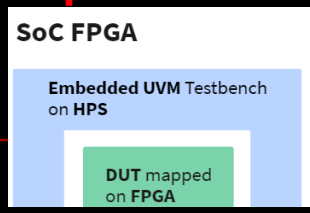
- With Systemc, Vlang can lock at delta cycle level

```
int main(int argc, char* argv[]) {
  initEsdl();          // initialize vlang
  int screslut =
    sc_core::sc_elab_and_sim(argc, argv);
  finalizeEsdl();      // stop vlang
  return 0;
}
int sc_main( int argc, char* argv[]) {
  sc_set_time_resolution(1, SC_PS);
  top = new SYSTEM("top");
  sc_start( SC_ZERO_TIME );
  while(sc_pending_activity()) {
    sc_core::sc_time time_ =
      sc_time_to_pending_activity();
    // start vlang simulation for given time
    esdlStartSimFor(time_.value());
    sc_start(time_);
    // wait for vlang to complete time step
    esdlWait();
  }
  return 0;
}
```

Source: https://dvcon-proceedings.org/wp-content/uploads/introduction-to-next-generation-verification-language-vlang-presentation.pdf (DVCon Europe 2014)

1.2 Embedded UVM

- http://uvm.io/

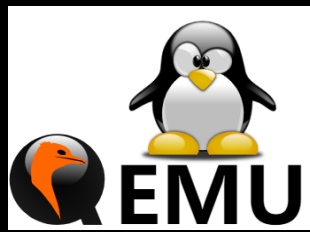


Spawn your own Emulation Platform for $100

Create your own SoCFPGA based Emulator for $100 and upto 100X speedup, with an **Embedded UVM** testbench running on HPS and DUT mapped on FPGA.

Run UVM Tests with Vivado and with GHDL

Opensource and Free **IEEE UVM 1.0** port complete with Constrained Reandomization, released under Apache2/Boost license.

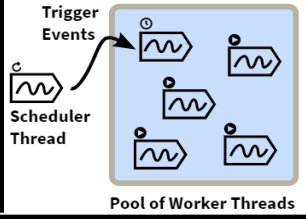


Co-Simulate your DUT with Device Drivers

LLVM powered native compilation on ARM and other embedded processors, with runtime Footprint small enough to run UVM on Raspberry PI and Beaglebone.

Deploy UVM Testbenches on Software Stack

LLVM powered native compilation on ARM and other embedded processors, with runtime Footprint small enough to run UVM on Raspberry PI and Beaglebone.

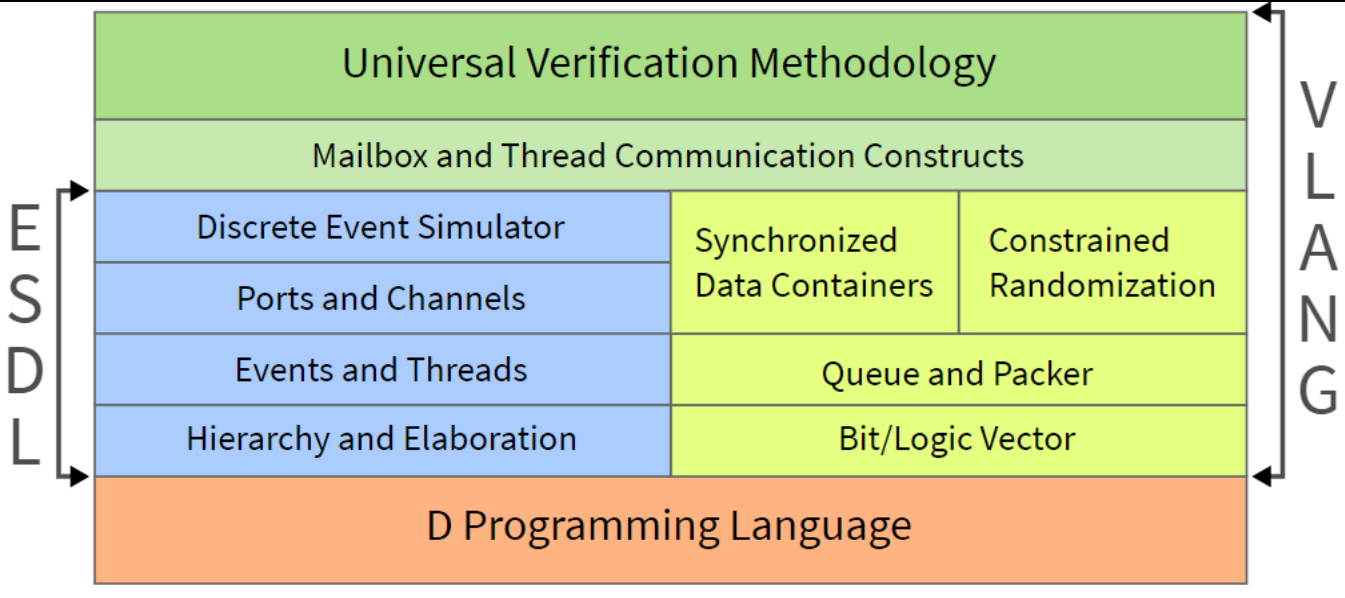


Scale your Testbench to Multicore Servers

The first and yet the only UVM implementation that lets your testbench run on multiple cores. Lets your testbench scale on Multicore server machine.

Vlang is built on top of the D Programming Language (Why D?). At the core of Vlang is a system specification language called **Electronic System Description Language** (ESDL). ESDL is much like SystemC, but is written from scratch in D Language and has many improvisations including ability to run simulation on a multicore system.

is a Discrete Event Simulator
and the associated constructs. Vlang also defines hardware data types that make it convenient for the verification engineer to model bit/logic vectors and signals. Also included in the core is a BDD based constraint solver and a glue library that allows constraints to be declared as part of a data transaction object. On top of the core layer, Vlang implements a port of the Universal Verification Methodology (UVM).



Source: http://uvm.io/docs/core-concepts/architecture-overview/

Design Elaboration

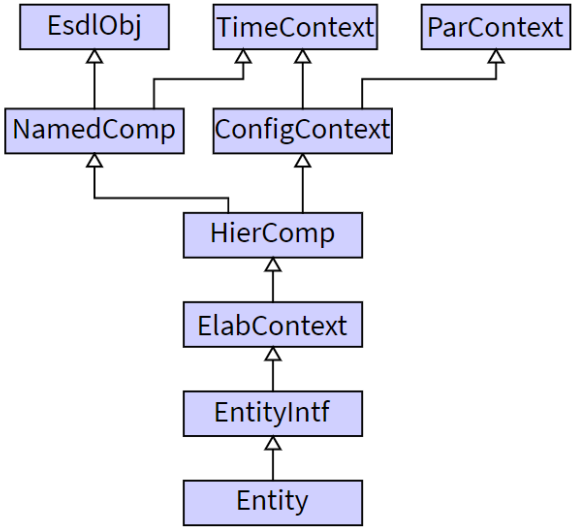
Elaboration in ESDL happens in phases. A component in ESDL is called an Entity. The diagram below shows the base class hierarchy of an Entity. To make elaboration of a design possible, each Entity in the design must have auxiliary code that reflects on the composition of the component. This is done by adding the following line as part of the Entity class declaration:

```
mixin Elaboration;
```

That is it. Normally you will not be required to add any other line of code in the entity to facilitate its elaboration. The Elaboration mixin would add the required code to the Entity. For example the code added by the mixin would:

- create a build function that will construct (new) all the component entities declared in the body of the current entity
- call build function for those component entities as well.

An ESDL *Entity* is analogous to `module` in SystemVerilog or an `sc_module` in SystemC. ESDL chose to use Entity because `module` is a keyword in D Programming Language.



Source: http://uvm.io/docs/core-concepts/hierarchy/

- **Notion of Time**

Just like SystemVerilog, time comes in ESDL in two forms. You can specify time in absolute form complete with a unit. In this form, the value of time is stored as a 64-bit value and the unit is stored as a byte enum. Since they are clubbed together in form of a structure, Time specified in this form takes 128 bits (16 bytes) of data.

More often, you would want to look at time as an integral value. ESDL uses a wrapper struct named SimTime for such integral modeling of time. SimTime gets the absolute sense only in conjunction with time precision used by the simulation. Since ESDL allows multiple simulator instances running on parallel threads, time precision is tagged with a simulator instance.

Source: http://uvm.io/docs/core-concepts/hierarchy/

- **http://uvm.io/faq/**

2) ESDL

2.1 Overview

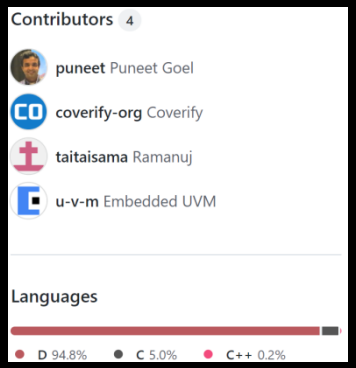
- https://github.com/coverify/esdl

Electronic System Description Language.

This package has the base simulator and the data modules, which include the constrained randomization unit. UVM package is available separately because it comes with a different opensource license.

- Src

Stats (on master branch, last commit: 33f946e10a46776d735b4c7c1d374367048ceb42)




```
[mydev@fedora esdl-master]$ tree -L 5 -d .
.
├── examples
│   ├── data
│   ├── rand
│   │   └── misc
│   ├── sim
│   └── vcd
└── src
    ├── esdl
    │   ├── base
    │   ├── data
    │   ├── intf
    │   │   ├── btor
    │   │   ├── verilator
    │   │   │   └── cpp
    │   │   └── z3
    │   │       └── api
    │   ├── posix
    │   │   └── sys
    │   │       └── net
    │   ├── rand
    │   ├── solver
    │   ├── sync
    │   ├── sys
    │   └── vcd
    └── unstd
        └── memory
```

```
[mydev@fedora esdl-master]$ tokei
===============================================================================
 Language            Files        Lines         Code     Comments       Blanks
===============================================================================
 C Header                3         3512         1173         2129          210
 C++                     2          127           64           41           22
 D                     135        93611        60477        21318        11816
 JSON                    2           33           33            0            0
 Markdown                1           12            0            8            4
===============================================================================
 Total                 143        97295        61747        23496        12052
===============================================================================
[mydev@fedora esdl-master]$
```

3) EUVM

3.1 Overview

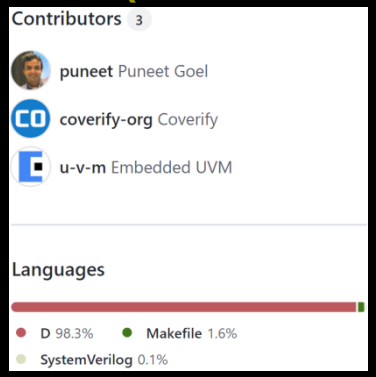
- https://github.com/coverify/euvm

  Embedded UVM (depends on ESDL)

  D language port of IEEE standard 1800.2 2020-1.0 UVM.

- Src

Stats (on master branch, last commit: f62139c470bf79d39c4a0b14f68bd76fb6a445b1)




```
[mydev@fedora euvm-master]$ tokei
===============================================================================
 Language            Files        Lines         Code     Comments       Blanks
===============================================================================
 D                     200        86933        47056        26936        12941
 JSON                    2           42           42            0            0
 Makefile               37         1343          401          653          289
 Markdown                1            5            0            4            1
 SystemVerilog           1           65           51            3           11
===============================================================================
 Total                 241        88388        47550        27596        13242
===============================================================================
[mydev@fedora euvm-master]$
```

```
[mydev@fedora euvm-master]$ tree -L 3 .
.
├── dub.json
├── dub.selections.json
├── LICENSE
├── README.md
├── src
│   ├── uvm
│   │   ├── base
│   │   ├── comps
│   │   ├── dap
│   │   ├── dpi
│   │   ├── meta
│   │   ├── package.d
│   │   ├── reg
│   │   ├── seq
│   │   ├── tlm1
│   │   ├── tlm2
│   │   └── vpi
│   └── uvm_pkg.d
├── tests
│   ├── 00basic
│   │   ├── 00hello
│   │   ├── 01compfail
│   │   ├── 02runfail
│   │   ├── 03error
│   │   ├── 06plusargs
│   │   ├── 07toolargs
│   │   ├── 10post_test
│   │   ├── 20subgroup
│   │   ├── 25typename
│   │   └── 90Mantis
│   ├── 01report
│   │   ├── 00message
│   │   ├── 02server
│   │   ├── 10catcher
│   │   ├── 20severity
│   │   ├── 30handler
│   │   └── 40macros
│   ├── 04vlang
│   │   └── 00set_local
│   └── test_defines.mk
└── verilog
    └── uvm_dpi_utils.sv
```

4) Summary

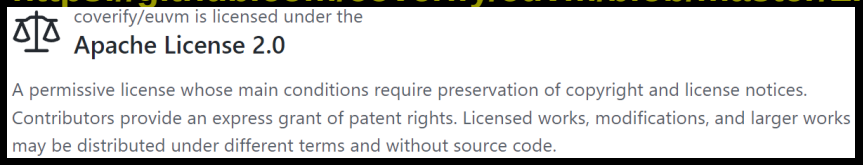
- ESDL/EUVM only support UVM 1.0;
- Project ESDL/EUVM(https://github.com/coverify/) are lacking of documents and test results;
- The latest all-in-one release EUVM v1.0-beta25 targets X64 only:

https://github.com/coverify/euvm/releases


```
[mydev@fedora euvm-1.0-beta25]$ file lib/libuvm-ldc-shared.so.0.0.2
lib/libuvm-ldc-shared.so.0.0.2: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, BuildID[sha1]=6a
305cb6c939de04bc7de1473d21808b3d311ba2, not stripped
[mydev@fedora euvm-1.0-beta25]$ file lib/libesdl-ldc-shared.so.0.0.2
lib/libesdl-ldc-shared.so.0.0.2: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, BuildID[sha1]=b
140692473dd80de8003680cf8e1865db26bed01, not stripped
[mydev@fedora euvm-1.0-beta25]$
```

Apache2/Boost license:

https://github.com/coverify/euvm/blob/master/LICENSE




```
[mydev@fedora euvm-1.0-beta25]$ cat ./LICENSE.txt
The following license applies to Software that is:

1. the executables in the bin directory
2. the libraries in the lib directory
3. the supplied source code

LDC, the D compiler is covered under the license file LICENSE_LDC.txt,
Z3, an SMT Solver by Microsoft is covered under the license file LICENSE_Z3.txt,
Embedded UVM port of IEEE UVM 2017-1.0 Reference Implementation is covered under the license file LICENSE_UVM.txt

The ESDL part of the package is covered under Boost Software License - Version 1.0.
```

v1.0-beta25:

```
[mydev@fedora euvm-1.0-beta25]$ tree -L 1 .
.
├── bin
├── etc
├── import
├── lib
├── LICENSE_LDC.txt
├── LICENSE.txt
├── LICENSE_UVM.txt
├── LICENSE_Z3.txt
├── README_LDC.txt
└── utils
```

```
[mydev@fedora euvm-1.0-beta25]$ tree bin
bin
├── ldc2
├── ldc-build-runtime
├── ldc-profdata
├── ldc-prune-cache
└── ldmd2
```

```
[mydev@fedora euvm-1.0-beta25]$ tree etc
etc
├── bash_completion.d
│   └── ldc2
└── ldc2.conf
```

```
[mydev@fedora euvm-1.0-beta25]$ tree utils
utils
├── patchelf
└── setup.sh
```

```
[mydev@fedora euvm-1.0-beta25]$ tree -L 2 -d import
import
├── core
│   ├── gc
│   ├── internal
│   ├── stdc
│   ├── stdcpp
│   ├── sync
│   ├── sys
│   └── thread
├── esdl
│   ├── base
│   ├── data
│   ├── intf
│   ├── posix
│   ├── rand
│   ├── solver
│   ├── sync
│   ├── sys
│   └── vcd
├── etc
│   ├── c
│   └── linux
├── ldc
├── std
│   ├── algorithm
│   ├── container
│   ├── datetime
│   ├── digest
│   ├── experimental
│   ├── format
│   ├── internal
│   ├── math
│   ├── net
│   ├── range
│   ├── regex
│   ├── uni
│   └── windows
└── uvm
    ├── base
    ├── comps
    ├── dap
    ├── dpi
    ├── meta
    ├── reg
    ├── seq
    ├── tlm1
    ├── tlm2
    └── vpi
```

```
[mydev@fedora euvm-1.0-beta25]$ tree lib
lib
├── ldc_rt.dso.o
├── libdruntime-ldc.a
├── libdruntime-ldc-debug.a
├── libdruntime-ldc-debug-shared.so
├── libdruntime-ldc-debug-shared.so.2.0.99
├── libdruntime-ldc-debug-shared.so.99
├── libdruntime-ldc-shared.so
├── libdruntime-ldc-shared.so.2.0.99
├── libdruntime-ldc-shared.so.99
├── libesdl-ldc.a
├── libesdl-ldc-debug.a
├── libesdl-ldc-debug-shared.so
├── libesdl-ldc-debug-shared.so.0
├── libesdl-ldc-debug-shared.so.0.0
├── libesdl-ldc-debug-shared.so.0.0.2
├── libesdl-ldc-shared.so
├── libesdl-ldc-shared.so.0
├── libesdl-ldc-shared.so.0.0
├── libesdl-ldc-shared.so.0.0.2
├── libldc-jit-rt.a
├── libldc-jit.so
├── libldc-jit.so.2.0.99
├── libldc-jit.so.99
├── libldc_rt.asan.a
├── libldc_rt.builtins.a
├── libldc_rt.fuzzer.a
├── libldc_rt.msan.a
├── libldc_rt.profile.a
├── libldc_rt.tsan.a
├── libldc_rt.xray.a
├── libldc_rt.xray-basic.a
├── libldc_rt.xray-fdr.a
├── libldc_rt.xray-profiling.a
├── libphobos2-ldc.a
├── libphobos2-ldc-debug.a
├── libphobos2-ldc-debug-shared.so
├── libphobos2-ldc-debug-shared.so.2.0.99
├── libphobos2-ldc-debug-shared.so.99
├── libphobos2-ldc-shared.so
├── libphobos2-ldc-shared.so.2.0.99
├── libphobos2-ldc-shared.so.99
├── libuvm-ldc.a
├── libuvm-ldc-debug.a
├── libuvm-ldc-debug-shared.so
├── libuvm-ldc-debug-shared.so.0
├── libuvm-ldc-debug-shared.so.0.0
├── libuvm-ldc-debug-shared.so.0.0.2
├── libuvm-ldc-shared.so
├── libuvm-ldc-shared.so.0
├── libuvm-ldc-shared.so.0.0
├── libuvm-ldc-shared.so.0.0.2
├── libz3.a
├── libz3.so
└── LLVMgold-ldc.so
```

```
[mydev@fedora euvm-1.0-beta25]$ du -sh
989M    .
[mydev@fedora euvm-1.0-beta25]$
```

- **Anyway, ESDL/EUVM provides a good reference of design and code base for us.**

III. Vlang on ARM

1) Porting to RPi4

1.1 Issues and workarounds

1.1.1 ESDL on RPi4

- We could meet the following error against “sourceLibrary” when trying to build the latest code of project ESDL directly on RPi4:

```
[mydev@fedora esdl-master]$ which ldc2
/opt/MyWorkSpace/MyProjs/Languages/D/Compiler/LDC/Official/ldc-master/build/Install/bin/ldc2
[mydev@fedora esdl-master]$
[mydev@fedora esdl-master]$ which dub
/usr/bin/dub
[mydev@fedora esdl-master]$
[mydev@fedora esdl-master]$ dub build
Main package must not have target type "sourceLibrary". Cannot build.
[mydev@fedora esdl-master]$
```

```
[mydev@fedora esdl-master]$ bat dub.json
       File: dub.json
   1   {
   2       "name": "vlang",
   3       "description": "Hardware Verification DSL",
   4       "license": "Boost",
   5       "copyright": "Copyright © 2012-2015 Coverify Systems Technology",
   6       "authors": [
   7       "Puneet Goel"
   8       ],
   9       "targetPath": "lib",
  10       "configurations": [
  11       {
  12           "name": "vlangsourcelib",
  13           "targetType": "sourceLibrary",
  14           "excludedSourceFiles": ["source/app.d"]
  15       },
  16       {
  17           "name": "vlangdynamiclib",
  18           "targetType": "dynamicLibrary",
  19           "dflags": ["-fPIC"],
  20           "excludedSourceFiles": ["source/app.d"]
  21       },
  22       {
  23           "name": "vlangstaticlib",
  24           "targetType": "staticLibrary",
  25           "excludedSourceFiles": ["source/app.d"]
  26       }
  27       ]
  28   }
[mydev@fedora esdl-master]$
```

apply a workaround and rebuild:

```
[mydev@fedora esdl-master]$ git diff
diff --git a/dub.json b/dub.json
index 978816b..917239f 100644
--- a/dub.json
+++ b/dub.json
@@ -10,7 +10,7 @@
     "configurations": [
        {
            "name": "vlangsourcelib",
-           "targetType": "sourceLibrary",
+           "targetType": "library",
            "excludedSourceFiles": ["source/app.d"]
        },
        {
[mydev@fedora esdl-master]$
[mydev@fedora esdl-master]$ dub build
Performing "debug" build using ldc2 for aarch64, arm_hardfloat.
vlang ~master: building configuration "vlangsourcelib"...
src/unstd/casts.d(60,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(88,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(222,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(54,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(177,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(220,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/esdl/base/alloc.d(5,8): Error: unable to read module `mallocator`
src/esdl/base/alloc.d(5,8):        Expected 'esdl/experimental/allocator/mallocator.d' or 'esdl/experimental/allocator/mallocator/package.d' in one of the following import paths:
import path[0] = src/
import path[1] = /opt/MyWorkSpace/MyProjs/Languages/D/Compiler/LDC/Official/ldc-master/build/Install/include/d
ldc2 failed with exit code 1.
[mydev@fedora esdl-master]$
```

temporarily replace the module “mallocator" from ESDL with what in Phobos and rebuild:

```
[mydev@fedora esdl-master]$ git diff
diff --git a/dub.json b/dub.json
index 978816b..917239f 100644
--- a/dub.json
+++ b/dub.json
@@ -10,7 +10,7 @@
     "configurations": [
        {
            "name": "vlangsourcelib",
-           "targetType": "sourceLibrary",
+           "targetType": "library",
            "excludedSourceFiles": ["source/app.d"]
        },
        {
diff --git a/src/esdl/base/alloc.d b/src/esdl/base/alloc.d
index 2eabe47..0b5f80f 100644
--- a/src/esdl/base/alloc.d
+++ b/src/esdl/base/alloc.d
@@ -2,7 +2,7 @@ module esdl.base.alloc;

 import std.experimental.allocator.typed: TypedAllocator;
 // import std.experimental.allocator.gc_allocator : GCAllocator;
-import esdl.experimental.allocator.mallocator: Mallocator;
+import std.experimental.allocator.mallocator: Mallocator;
 // import std.experimental.allocator.mmap_allocator : MmapAllocator;

 alias EsdlAllocator = TypedAllocator!(Mallocator);
[mydev@fedora esdl-master]$
[mydev@fedora esdl-master]$ dub build
Performing "debug" build using ldc2 for aarch64, arm_hardfloat.
vlang ~master: building configuration "vlangsourcelib"...
src/unstd/casts.d(60,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(88,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(222,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(54,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(177,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(220,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(16,8): Error: unable to read module `array`
src/unstd/memory/weakref.d(16,8):        Expected 'unstd/array.d' or 'unstd/array/package.d' in one of the following import paths:
import path[0] = src/
import path[1] = /opt/MyWorkSpace/MyProjs/Languages/D/Compiler/LDC/Official/ldc-master/build/Install/include/d
ldc2 failed with exit code 1.
[mydev@fedora esdl-master]$
```

comment the unused module "array" and rebuild:

```
[mydev@fedora esdl-master]$ git diff
diff --git a/dub.json b/dub.json
index 978816b..917239f 100644
--- a/dub.json
+++ b/dub.json
@@ -10,7 +10,7 @@
     "configurations": [
         {
             "name": "vlangsourcelib",
-            "targetType": "sourceLibrary",
+            "targetType": "library",
             "excludedSourceFiles": ["source/app.d"]
         },
         {
diff --git a/src/esdl/base/alloc.d b/src/esdl/base/alloc.d
index 2eabe47..0b5f80f 100644
--- a/src/esdl/base/alloc.d
+++ b/src/esdl/base/alloc.d
@@ -2,7 +2,7 @@ module esdl.base.alloc;

 import std.experimental.allocator.typed: TypedAllocator;
 // import std.experimental.allocator.gc_allocator : GCAllocator;
-import esdl.experimental.allocator.mallocator: Mallocator;
+import std.experimental.allocator.mallocator: Mallocator;
 // import std.experimental.allocator.mmap_allocator : MmapAllocator;

 alias EsdlAllocator = TypedAllocator!(Mallocator);
diff --git a/src/unstd/memory/weakref.d b/src/unstd/memory/weakref.d
index 9a258ac..24bb571 100644
--- a/src/unstd/memory/weakref.d
+++ b/src/unstd/memory/weakref.d
@@ -13,7 +13,7 @@ import core.exception;
 import core.memory;
 import core.atomic;

-import unstd.array;
+//import unstd.array;
 import unstd.casts;
 import unstd.memory.allocation;

[mydev@fedora esdl-master]$
[mydev@fedora esdl-master]$ dub build
Performing "debug" build using ldc2 for aarch64, arm_hardfloat.
vlang ~master: building configuration "vlangsourcelib"...
src/unstd/casts.d(60,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(88,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(222,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(54,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(177,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(220,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(18,8): Error: unable to read module `allocation`
src/unstd/memory/weakref.d(18,8):        Expected 'unstd/memory/allocation.d' or 'unstd/memory/allocation/package.d' in one of the following import paths:
import path[0] = src/
import path[1] = /opt/MyWorkSpace/MyProjs/Languages/D/Compiler/LDC/Official/ldc-master/build/Install/include/d
ldc2 failed with exit code 1.
[mydev@fedora esdl-master]$
```

comment the unused module "allocation" and rebuild:

```
[mydev@fedora esdl-master]$ git diff
diff --git a/dub.json b/dub.json
index 978816b..917239f 100644
--- a/dub.json
+++ b/dub.json
@@ -10,7 +10,7 @@
     "configurations": [
         {
             "name": "vlangsourcelib",
-            "targetType": "sourceLibrary",
+            "targetType": "library",
             "excludedSourceFiles": ["source/app.d"]
         },
         {
diff --git a/src/esdl/base/alloc.d b/src/esdl/base/alloc.d
index 2eabe47..0b5f80f 100644
--- a/src/esdl/base/alloc.d
+++ b/src/esdl/base/alloc.d
@@ -2,7 +2,7 @@ module esdl.base.alloc;

 import std.experimental.allocator.typed: TypedAllocator;
 // import std.experimental.allocator.gc_allocator : GCAllocator;
-import esdl.experimental.allocator.mallocator: Mallocator;
+import std.experimental.allocator.mallocator: Mallocator;
 // import std.experimental.allocator.mmap_allocator : MmapAllocator;

 alias EsdlAllocator = TypedAllocator!(Mallocator);
diff --git a/src/unstd/memory/weakref.d b/src/unstd/memory/weakref.d
index 9a258ac..f919c83 100644
--- a/src/unstd/memory/weakref.d
+++ b/src/unstd/memory/weakref.d
@@ -13,9 +13,9 @@ import core.exception;
 import core.memory;
 import core.atomic;

-import unstd.array;
+//import unstd.array;
 import unstd.casts;
-import unstd.memory.allocation;
+//import unstd.memory.allocation;


 /**
[mydev@fedora esdl-master]$
[mydev@fedora esdl-master]$ dub build
Performing "debug" build using ldc2 for aarch64, arm_hardfloat.
vlang ~master: building configuration "vlangsourcelib"...
src/unstd/casts.d(60,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(88,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(222,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(54,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(177,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(220,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
libstdc++ std::__cxx11::basic_string is not yet supported; the struct contains an interior pointer which breaks D move semantics!
[mydev@fedora esdl-master]$
```

The final outputs:

```
[mydev@fedora esdl-master]$ tree lib
lib
└── libvlang.a
```

```
[mydev@fedora esdl-master]$ tree .dub
.dub
├── build
│   ├── vlangdynamiclib-debug-linux.posix-aarch64.arm_hardfloat-ldc_2100-3420C13DF86B711EA86F53B1DA6FBC5F
│   └── vlangsourcelib-debug-linux.posix-aarch64.arm_hardfloat-ldc_2100-227CD0E0F8F4DE9B6F10F9D94EDADED0
│       └── libvlang.a
└── obj
    ├── bvec.o
    ├── esdl.base.alloc.o
    ├── esdl.base.cmdl.o
    ├── esdl.base.comm.o
    ├── esdl.base.core.o
    ├── esdl.base.o
    ├── esdl.base.rand.o
    ├── esdl.data.bstr.o
    ├── esdl.data.bvec.o
    ├── esdl.data.charbuf.o
    ├── esdl.data.o
    ├── esdl.data.packed.o
    ├── esdl.data.packer.o
    ├── esdl.data.queue.o
    ├── esdl.data.sync.o
    ├── esdl.data.time.o
    ├── esdl.data.vector.o
    ├── esdl.intf.btor.api.o
    ├── esdl.intf.btor.btor.o
    ├── esdl.intf.btor.o
    ├── esdl.intf.btor.types.o
    ├── esdl.intf.fli.o
    ├── esdl.intf.hal.o
    ├── esdl.intf.mhpi.o
    ├── esdl.intf.mraa.o
    ├── esdl.intf.o
    ├── esdl.intf.systemc.o
    ├── esdl.intf.verilator.o
    ├── esdl.intf.verilator.trace.o
    ├── esdl.intf.verilator.verilated.o
    ├── esdl.intf.vhpi.o
    ├── esdl.intf.vpi.o
    ├── esdl.intf.vtap.o
    ├── esdl.intf.z3.api.o
    ├── esdl.intf.z3.api.z3_algebraic.o
    ├── esdl.intf.z3.api.z3_api.o
    ├── esdl.intf.z3.api.z3_ast_containers.o
    ├── esdl.intf.z3.api.z3_fixedpoint.o
    ├── esdl.intf.z3.api.z3_fpa.o
    ├── esdl.intf.z3.api.z3_optimization.o
    ├── esdl.intf.z3.api.z3_polynomial.o
    ├── esdl.intf.z3.api.z3_rcf.o
    ├── esdl.intf.z3.api.z3_spacer.o
    ├── esdl.intf.z3.api.z3_types.o
    ├── esdl.intf.z3.o
    ├── esdl.intf.z3.z3.o
    ├── esdl.o
    ├── esdl_pkg.o
    ├── esdl.posix.sys.net.if_.o
    ├── esdl.rand.agent.o
    ├── esdl.rand.base.o
    ├── esdl.rand.cover.o
    ├── esdl.rand.cstr.o
    ├── esdl.rand.domain.o
    ├── esdl.rand.expr.o
    ├── esdl.rand.func.o
    ├── esdl.rand.meta.o
    ├── esdl.rand.misc.o
    ├── esdl.rand.o
    ├── esdl.rand.objx.o
    ├── esdl.rand.parser.o
    ├── esdl.rand.pred.o
    ├── esdl.rand.proxy.o
    ├── esdl.rand.vecx.o
    ├── esdl.solver.base.o
    ├── esdl.solver.buddy.o
    ├── esdl.solver.dist.o
    ├── esdl.solver.mono.o
    ├── esdl.solver.o
    ├── esdl.solver.obdd.o
    ├── esdl.solver.z3expr.o
    ├── esdl.solver.z3.o
    ├── esdl.sync.barrier.o
    ├── esdl.sys.o
    ├── esdl.sys.sched.o
    ├── esdl.vcd.parse.o
    ├── unstd.casts.o
    └── unstd.memory.weakref.o
```

Testing

```
[mydev@fedora esdl-master]$ dub test
Generating test runner configuration 'vlang-test-vlangsourcelib' for 'vlangsourcelib' (library).
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Performing "unittest" build using ldc2 for aarch64, arm_hardfloat.
vlang ~master: building configuration "vlang-test-vlangsourcelib"...
src/unstd/casts.d(60,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(88,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(222,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(54,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(177,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(220,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
/tmp/dub_test_root_2246384d_e04d_4cdb_8385_86ae9921d662.d(65,15): Error: module `bvec` from file src/esdl/vcd/bvec.d must be imported with 'import bvec;'
ldc2 failed with exit code 1.
[mydev@fedora esdl-master]$
```

add the missing module path and retest:

```
[mydev@fedora esdl-master]$ git diff src/esdl/vcd/bvec.d
diff --git a/src/esdl/vcd/bvec.d b/src/esdl/vcd/bvec.d
index 92c55af..60b3e34 100644
--- a/src/esdl/vcd/bvec.d
+++ b/src/esdl/vcd/bvec.d
@@ -7,6 +7,7 @@
 // Authors:   Puneet Goel <puneet@coverify.com>

 // This file is part of esdl.
+module esdl.vcd.bvec;

 import std.bitmanip;

[mydev@fedora esdl-master]$
```

```
[mydev@fedora esdl-master]$ dub test
Generating test runner configuration 'vlang-test-vlangsourcelib' for 'vlangsourcelib' (library).
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Excluding package.d file from test due to https://issues.dlang.org/show_bug.cgi?id=11847
Performing "unittest" build using ldc2 for aarch64, arm_hardfloat.
vlang ~master: building configuration "vlang-test-vlangsourcelib"...
src/unstd/casts.d(60,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(88,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/casts.d(222,1): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(54,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(177,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
src/unstd/memory/weakref.d(220,2): Deprecation: usage of the `body` keyword is deprecated. Use `do` instead.
libstdc++ std::__cxx11::basic_string is not yet supported; the struct contains an interior pointer which breaks D move semantics!
src/esdl/data/bstr.d(131,16): Error: undefined identifier `LogicArray`
src/esdl/data/bstr.d(269,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(326,11): Error: struct `BitString` does not overload ()
src/esdl/data/bstr.d(442,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(478,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(479,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(547,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(548,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(25,7): Error: template instance `esdl.data.bstr.BitString!true` error instantiating
src/esdl/data/bstr.d(131,16): Error: undefined identifier `LogicArray`
src/esdl/data/bstr.d(269,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(326,11): Error: struct `BitString` does not overload ()
src/esdl/data/bstr.d(442,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(478,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(479,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(547,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(548,23): Error: struct `BitArray` does not overload ()
src/esdl/data/bstr.d(26,7): Error: template instance `esdl.data.bstr.BitString!false` error instantiating
src/esdl/data/bvec.d(3779,7): Error: overloads `(_bvec!(true, true, 1LU) other)` and `(_bvec!(true, true, 1LU) other)` both match argument list for `opAssign`
src/esdl/data/bvec.d(3779,7): Error: forward reference to template `opAssign`
ldc2 failed with exit code 1.
[mydev@fedora esdl-master]$
```

still working on fix all the issues in ESDL testing…

1.1.2 EUVM on RPi4

- It’s sure to meet the following errors when trying to build EUVM directly for missing dependency “vlang(ESDL)”:

```
[mydev@fedora euvm-master]$ which ldc2
/opt/MyWorkSpace/MyProjs/Languages/D/Compiler/LDC/Official/ldc-master/build/Install/bin/ldc2
[mydev@fedora euvm-master]$
[mydev@fedora euvm-master]$ which dub
/usr/bin/dub
[mydev@fedora euvm-master]$
[mydev@fedora euvm-master]$ dub -v build
Using dub registry url 'https://code.dlang.org/'
Refreshing local packages (refresh existing: true)...
Looking for local package map at /var/lib/dub/packages/local-packages.json
Looking for local package map at /home/mydev/.dub/packages/local-packages.json
Looking for local package map at /opt/MyWorkSpace/MyProjs/Languages/D/HW-EDA/HDL/Vlang/Official/euvm-master/.dub/packages/local-packages.json
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Determined package version using GIT: lumars 1.5.0+commit.1.ga12c005
Determined package version using GIT: vlanguvm ~master
Refreshing local packages (refresh existing: false)...
Looking for local package map at /var/lib/dub/packages/local-packages.json
Looking for local package map at /home/mydev/.dub/packages/local-packages.json
Looking for local package map at /opt/MyWorkSpace/MyProjs/Languages/D/HW-EDA/HDL/Vlang/Official/euvm-master/.dub/packages/local-packages.json
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
  Missing dependency vlang ~master of vlanguvm
Refreshing local packages (refresh existing: false)...
Looking for local package map at /var/lib/dub/packages/local-packages.json
Looking for local package map at /home/mydev/.dub/packages/local-packages.json
Looking for local package map at /opt/MyWorkSpace/MyProjs/Languages/D/HW-EDA/HDL/Vlang/Official/euvm-master/.dub/packages/local-packages.json
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
Ignoring version specification (≥0.0.0) for path based dependency ../../
  Missing dependency vlang ~master of vlanguvm
Checking for missing dependencies.
Selected package vlang ~master doesn't exist. Using latest matching version instead.
Search for versions of vlang (1 package suppliers)
No versions for vlang for registry at https://code.dlang.org/ (fallbacks registry at https://codemirror.dlang.org/, registry at https://dub.bytecraft.nl/, registry at https://code-mirror.dlang.io/)
Nothing found for vlang
Failed to find any versions for package vlang, referenced by vlanguvm ~master
[mydev@fedora euvm-master]$
```

add the path of project ESDL that we built previously and rebuild:

```
[mydev@fedora euvm-master]$ bat dub.json
```

File: dub.json

```
{
    "name": "vlanguvm",
    "description": "UVM Port for Vlang",
    "license": "Apache 2.0",
    "copyright": "Copyright © 2012-2015 Coverify Systems Technology",
    "authors": [
    "Puneet Goel"
    ],
    "dependencies": {
    "vlang": "*"
    },
    "targetPath": "lib",
    "configurations": [
    {
        "name": "vlanguvmdynamiclib",
        "targetType": "dynamicLibrary",
        "dflags": ["-fPIC"],
        "excludedSourceFiles": ["source/app.d"]
    },
    {
        "name": "vlanguvmstaticlib",
        "targetType": "staticLibrary",
        "excludedSourceFiles": ["source/app.d"]
    },
    {
        "name": "vlanguvmsourcelib",
        "targetType": "sourceLibrary",
        "excludedSourceFiles": ["source/app.d"]
    }
    ],
    "excludedSourceFiles": [
    "src/uvm/reg/*",
    "src/uvm/tlm2/*",
    "src/uvm/vpi/uvm_hdl.d"
    ]
}
```

```
[mydev@fedora euvm-master]$
```

```
[mydev@fedora euvm-master]$ git diff
diff --git a/dub.json b/dub.json
index dfddcc6..d51056c 100644
--- a/dub.json
+++ b/dub.json
@@ -7,7 +7,10 @@
         "Puneet Goel"
     ],
     "dependencies": {
-        "vlang": "*"
+        "vlang": {
+            "version": "~master",
+            "path": "../esdl-master"
+        }
     },
     "targetPath": "lib",
     "configurations": [
[mydev@fedora euvm-master]$
```

```
[mydev@fedora euvm-master]$ dub build
Selected package vlang ~master doesn't exist. Using latest matching version instead.
Performing "debug" build using ldc2 for aarch64, arm_hardfloat.
vlang ~master: target for configuration "vlangsourcelib" is up to date.
vlanguvm ~master: building configuration "vlanguvmdynamiclib"...
ldc2: Unknown command line argument '-fPIC'.  Try: 'ldc2 --help'
ldc2: Did you mean '-I'?
ldc2 failed with exit code 1.
[mydev@fedora euvm-master]$
```

remove the "-fPIC" dflag and rebuild:

```
[mydev@fedora euvm-master]$ git diff
diff --git a/dub.json b/dub.json
index dfddcc6..d94ba0f 100644
--- a/dub.json
+++ b/dub.json
@@ -7,14 +7,16 @@
         "Puneet Goel"
     ],
     "dependencies": {
-        "vlang": "*"
+        "vlang": {
+            "version": "~master",
+            "path": "../esdl-master"
+        }
     },
     "targetPath": "lib",
     "configurations": [
         {
             "name": "vlanguvmdynamiclib",
             "targetType": "dynamicLibrary",
-            "dflags": ["-fPIC"],
             "excludedSourceFiles": ["source/app.d"]
         },
         {
diff --git a/dub.selections.json b/dub.selections.json
index 11aa124..1387208 100644
--- a/dub.selections.json
+++ b/dub.selections.json
@@ -1,6 +1,6 @@
 {
         "fileVersion": 1,
         "versions": {
-                "vlang": "~master"
+                "vlang": {"path":"../esdl-master"}
         }
 }
[mydev@fedora euvm-master]$
```

```
[mydev@fedora euvm-master]$ dub build
Performing "debug" build using ldc2 for aarch64, arm_hardfloat.
vlang ~master: target for configuration "vlangsourcelib" is up to date.
vlanguvm ~master: building configuration "vlanguvmdynamiclib"...
src/uvm/base/uvm_registry.d(239,10): Error: class `uvm.base.uvm_registry.uvm_object_registry!(uvm_table_printer, "uvm.ba
ences
src/uvm/base/uvm_object_defines.d(209,19): Error: template instance `uvm.base.uvm_registry.uvm_object_registry!(uvm_tabl
src/uvm/base/uvm_registry.d(203,5): Error: template instance `uvm.base.uvm_registry.uvm_registry_common!(uvm_object_regi
ject_creator, uvm_table_printer, "uvm.base.uvm_printer.uvm_table_printer")` error instantiating
src/uvm/base/uvm_object_defines.d(209,19):        instantiated from here: `uvm_object_registry!(uvm_table_printer, "uvm.
src/uvm/base/uvm_registry.d-mixin-245-mixin-245(249,31): Error: no property `_m_inst` for type `uvm.base.uvm_registry.uv
_object_registry.uvm_scope`
src/uvm/base/uvm_object_defines.d(118,3): Error: mixin `uvm.base.uvm_printer.uvm_table_printer.uvm_object_essentials!voi
nter")` error instantiating
src/uvm/base/uvm_printer.d(1070,3): Error: mixin `uvm.base.uvm_printer.uvm_table_printer.uvm_object_essentials!void` err
ldc2 failed with exit code 1.
[mydev@fedora euvm-master]$
```

working on the workarounds for these issues…

2) More bug fixing and testing

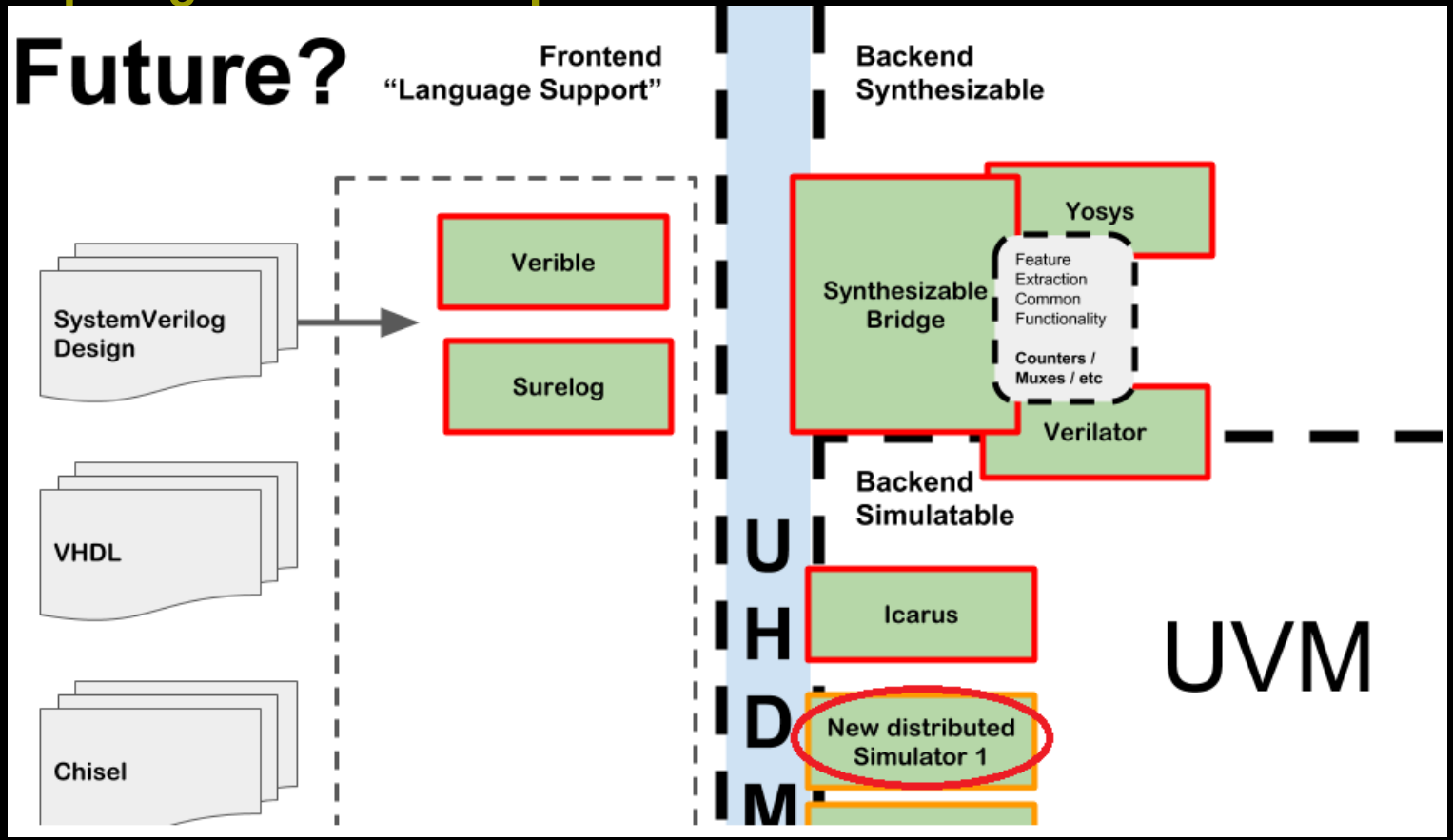
- As you saw before, it is obviously that there's still a lot of work left to be done to make ESDL/EUVM really does work on ARM devices…

IV. The future

1) Next Generation RTL Simulator

Distributed Simulation

- https://github.com/chipsalliance/UHDM

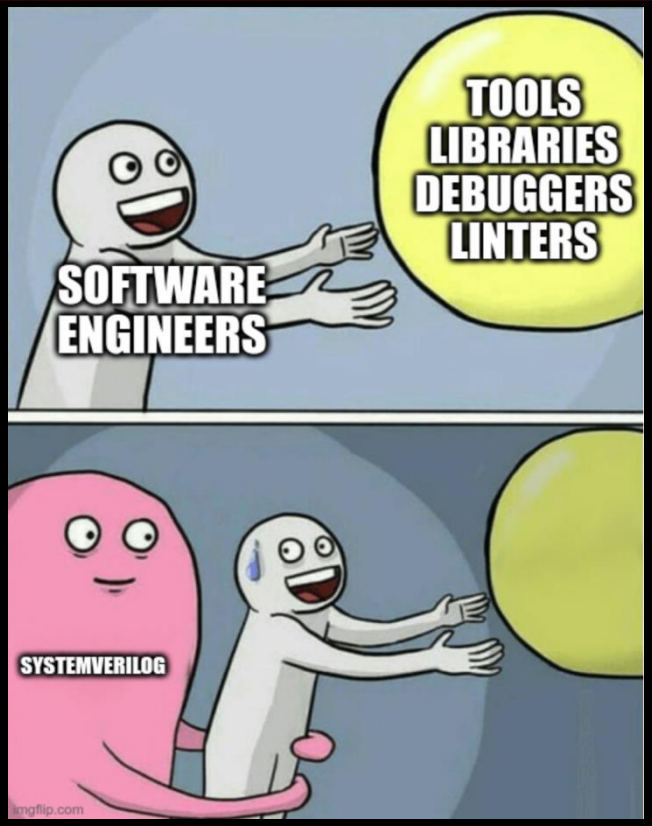


...

2) Beyond UVM

2.1 It's time to thank UVM and say goodbye?

- https://olofkindgren.blogspot.com/2022/10/its-time-to-to-thank-uvm-and-say-goodbye.html



…

2.2 Cocotb with 

- **Code Stats(master branch)**

  (master branch, last commit: d2b8dab8f5baffbc6cc53a6c44068011e5532089)

```
[mydev@fedora cocotb]$ git branch
* master
[mydev@fedora cocotb]$ git log -1
commit d2b8dab8f5baffbc6cc53a6c44068011e5532089 (HEAD → master, origin/master, origin/HEAD)
Author: Tomasz Hemperek <themperek@users.noreply.github.com>
Date:   Fri Nov 25 10:13:12 2022 +0100

    Python runner improvements and documentation  (#3103)

    Refactor and add documentation of Python runner
[mydev@fedora cocotb]$
[mydev@fedora cocotb]$ tokei
===============================================================================
 Language            Files        Lines         Code     Comments       Blanks
===============================================================================
 Autoconf                1            5            5            0            0
 C Header               21        13247         8818         3183         1246
 Coq                     9         1207         1101            0          106
 C++                    16         9057         6703         1027         1327
 Makefile               64         1973          755          897          321
 Module-Definition       4          282          282            0            0
 PowerShell              1           15            8            5            2
 Python                128        22584        16801         1983         3800
 ReStructuredText       33         4460         3063            0         1397
 SVG                     5          128          126            2            0
 SystemVerilog          27         1535          968          292          275
 Plain Text              3          484            0          425           59
 TOML                    1           71           47           13           11
 VHDL                   52         4785         3197          912          676
 XML                     4          337          337            0            0
 YAML                    1            9            6            2            1
-------------------------------------------------------------------------------
 Markdown                6          681            0          498          183
 ├─ Python               2           24           14            6            4
 ├─ SystemVerilog        1           14            9            1            4
 (Total)                             719           23          505          191
===============================================================================
 Total                 376        60860        42217         9239         9404
===============================================================================
[mydev@fedora cocotb]$
```

- **Trying to replace the C++ code in Cocotb with**  **…**

V. Wrap-up

- A New Golden Age for FOSS EDA!
-  is already showing its potential for the next generation system programming, including the HW-SW collaboration, but still has a long way to go…
- Many D-based open-source projects may not be well-tested outside X86, while we should pay more attention on ARM and RISC-V…
  First, let's enhance LDC to better support for ARM, RISC-V, Wasm, eBPF and SPIR-V (it seems that porting of LDC to RISC-V is on the way…)!
- For more materials, you may refer to our previous talks "Will D be a better system programming language" at OpenInfra Days China 2022(Online) and "First exploration of D for HW-SW co-designed system" at 1st OSEDA Workshop China 2022(Online).
  And you may also look forward to our upcoming follow-ups "Cocotb: a Swiss Army Knife for hardware verification" at PyCon China 2022, "Revisiting D as a better system programming language", and "Rethinking D for HW-SW co-designed system" in the near future.

Q & A

Reference

Slides/materials from many and varied sources:

- http://en.wikipedia.org/wiki/
- http://www.slideshare.net/

- https://en.wikipedia.org/wiki/SystemVerilog
- https://en.wikipedia.org/wiki/Device_under_test
- https://en.wikipedia.org/wiki/Coroutine
- https://bitsbytesgates.blogspot.com/
- https://chipsalliance.org/workshops-meetings/
- http://testbench.in/
- https://github.com/kennyalive/Language-Arena
- https://thume.ca/2019/07/14/a-tour-of-metaprogramming-models-for-generics/
- https://users.rust-lang.org/t/dlang-adds-a-borrowchecker-called-the-ob-system-for-ownership-borrowing/42872/11
- https://wiki.dlang.org/LDC_CUDA_and_SPIRV
- https://github.com/hdl/bazel_rules_hdl
- https://blog.devgenius.io/d-c-evolution-programming-on-android-9bdf9aa1b67d
- …