UHDM

Universal **H**ardware **D**ata **M**odel

github.com/alainmarcel/UHDM

Interface in between parsers (SureLog…) and tools

For every SystemVerilog Object in the IEEE 1800-2017 Object Model, this tool creates:

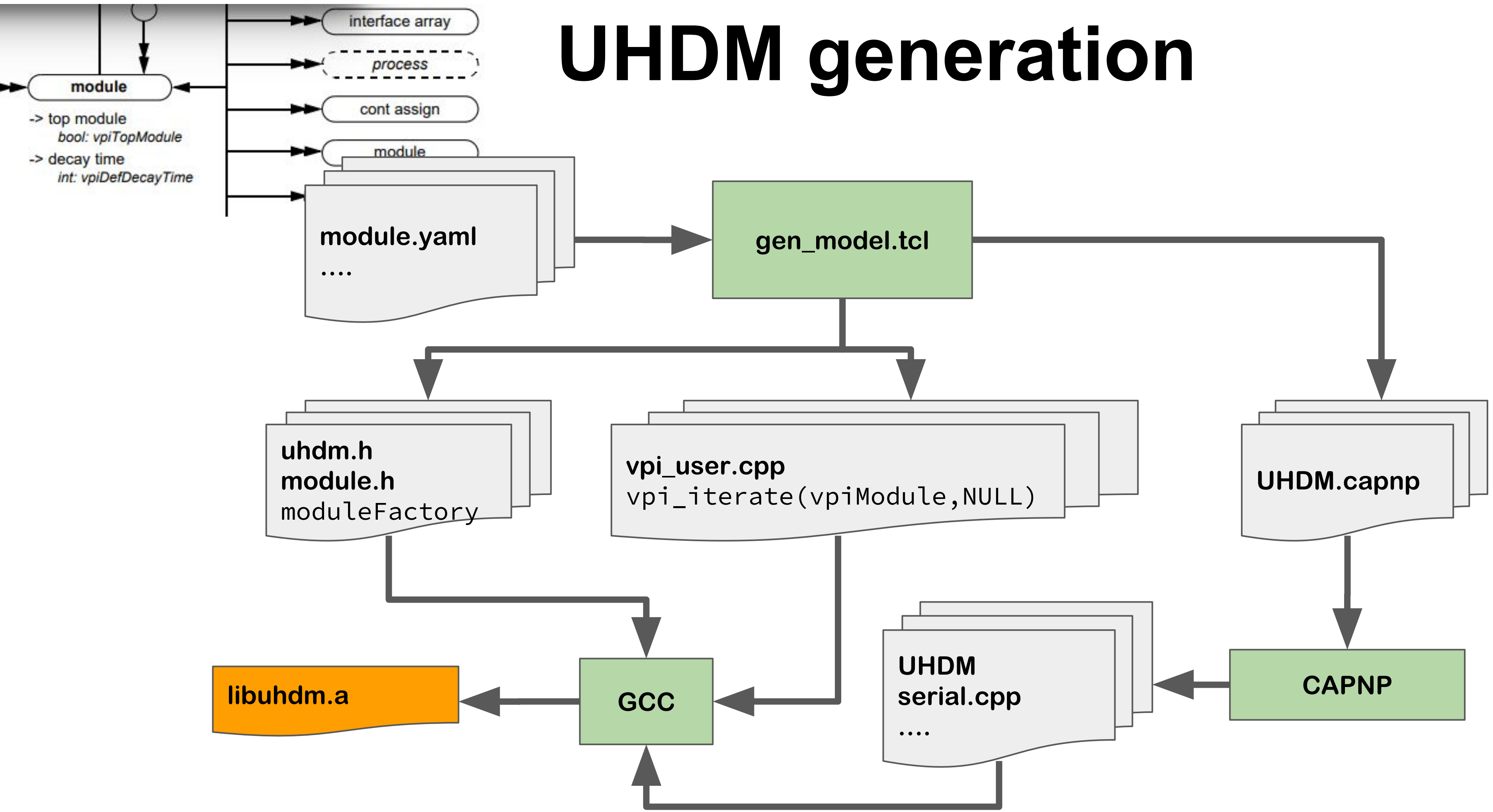
- a C++ class,
- corresponding VPI access (vpi_iterate, vpi_get, vpi_handle…)
- Serialization using Cap'n Proto mechanism
- Walker (Decompiler) producing human readable output
- Listener Design Pattern
- Optional Elaborator/Uniquification (pre or post serialization)

How?

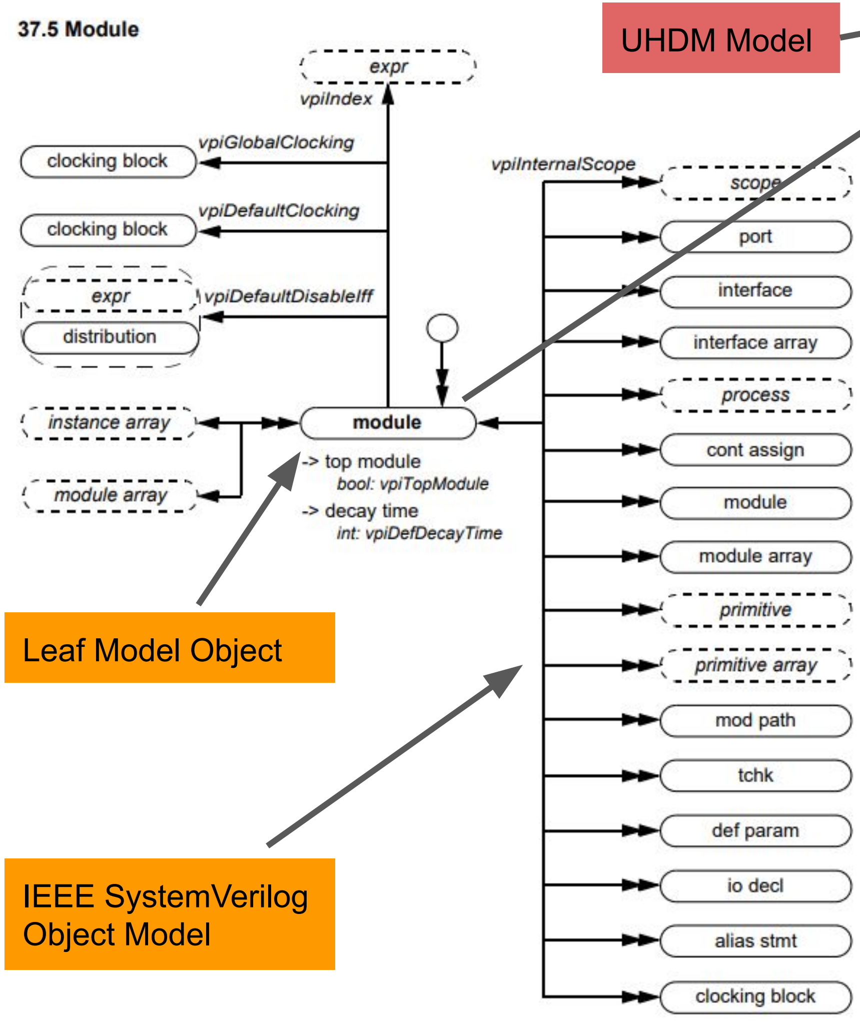
- The Object Model is captured in YAML-like form
- A script generates all the code automatically.
- In-memory data model is read/write even after deserialization

This material resulted (in part) from work supported by Google. The content is solely the responsibility of the authors and does not necessarily represent the official views of Google.

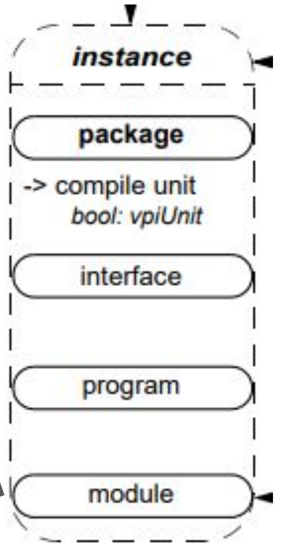
UHDM generation



UHDM Model



```
- obj_def: module
  - extends: instance
  - property: index
    name: index
    vpi: vpiIndex
    type: unsigned int
    card: 1
  - property: type
    name: vpiModule
    vpi: vpiType
    type: unsigned int
    card: 1
  - property: top_module
    name: top module
    type: bool
    vpi: vpiTopModule
    card: 1
  - property: decay_time
    name: decay time
    type: int
    vpi: vpiDefDecayTime
    card: 1
  - obj_ref: global_clocking
    name: global clocking
    vpi: vpiGlobalClocking
    type: clocking_block
    card: 1
  - obj_ref: default_clocking
    name: default clocking
    vpi: vpiDefaultClocking
    type: clocking_block
    card: 1
  - class_ref: expr_dist
    name: expr distribution
    vpi: vpiDefaultDisableIff
```

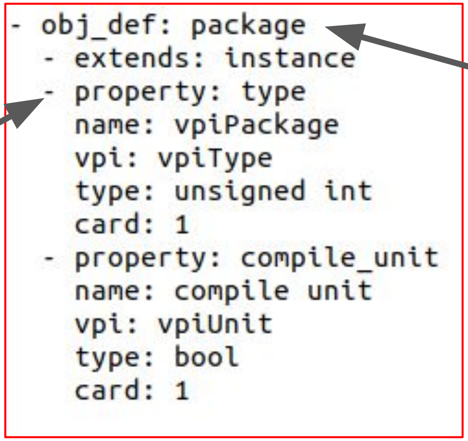


Leaf Model Object

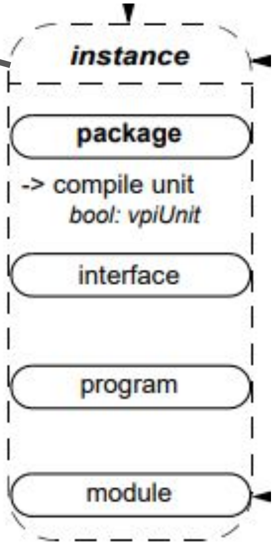
IEEE SystemVerilog Object Model

The “instance” diagram shows that “module” inherits from “instance”

All objects have a type field matching their name by default (camelization), but model writer can override

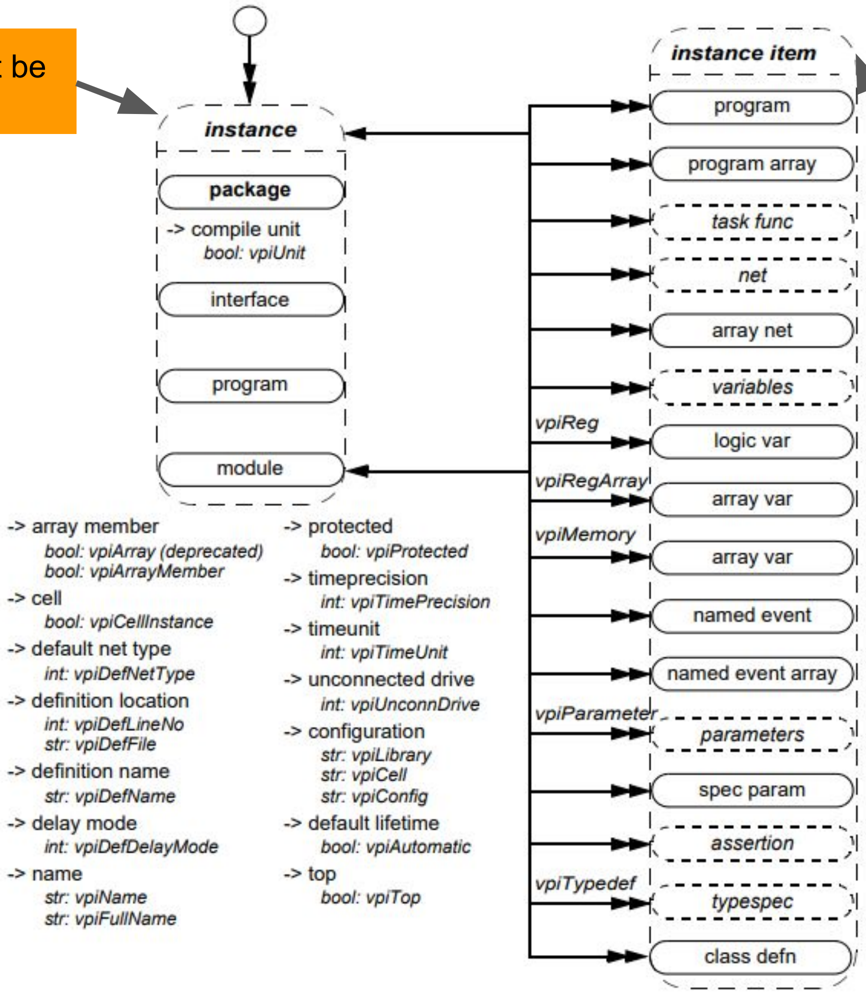


```
- obj_def: package
  - extends: instance
  - property: type
    name: vpiPackage
    vpi: vpiType
    type: unsigned int
    card: 1
  - property: compile_unit
    name: compile unit
    vpi: vpiUnit
    type: bool
    card: 1
```



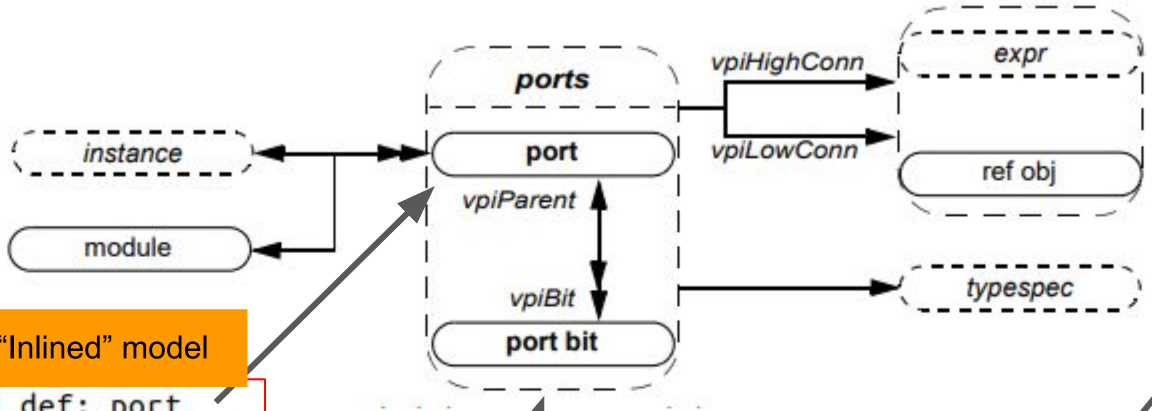
Virtual Object (Class), cannot be instantiated

```
class_def: instance
- extends: scope
- property: definition_name
  name: definition name
  vpi: vpiDefName
  type: string
  card: 1
- property: array_member
  name: array member
  vpi: vpiArrayMember
  type: bool
  card: 1
- property: cell
  name: cell
  vpi: vpiCellInstance
  type: bool
  card: 1
- property: default_net_type
  name: default net type
  vpi: vpiDefNetType
  type: int
  card: 1
- property: definition_location_file
  name: definition location file
  vpi: vpiDefFile
  type: string
  card: 1
- property: definition_location_line
  name: definition location line
  vpi: vpiDefLineNo
```



Named Group (think void*)

```
- group_def: instance_item
  - obj_ref: program
  - obj_ref: program_array
  - class_ref: task_func
  - class_ref: net
  - class_ref: array_net
  - class_ref: variables
  - obj_ref: logic_var
  - obj_ref: array_var
  - obj_ref: named_event
  - obj_ref: named_event_array
  - class_ref: parameters
  - obj_ref: spec_param
  - class_ref: assertion
  - class_ref: typespec
  - class_ref: class_defn
```



Leaf “Inlined” model

```
- obj_def: port
  - extends: ports
  - obj_ref: bits
    name: bits
    vpi: vpiBit
    type: port_bit
    card: any
```

```
- class_def: ports
  - property: index
    name: index
    vpi: vpiPortIndex
    type: unsigned int
    card: 1
 .....
  - property: explicitly_named
    name: name
    vpi: vpiExplicitName
    type: string
    card: 1
  - group_ref: high_conn
    name: high conn
    vpi: vpiHighConn
    type: expr_ref_obj_group
    card: 1
  - group_ref: low_conn
    name: low conn
    vpi: vpiLowConn
    type: expr_ref_obj_group
```

Unnamed group (we give a representative name)

```
- group_def: expr_ref_obj_group
  - class_ref: expr
  - obj_ref: ref_obj
```

UHDM APIs

Populate UHDM:

```
std::vector<vpiHandle> build_designs (Serializer& s) {
  std::vector<vpiHandle> designs;
  design* d = s.MakeDesign();
  d->VpiName("design1");

  //-----------------------------------------
  // Module definition M1 (non elaborated)
  module* m1 = s.MakeModule();
  {
    m1->VpiDefName("M1");
    m1->VpiParent(d);
    m1->VpiFile("fake1.sv");
    m1->VpiLineNo(10);
  }
```

VPI:

```
unsigned int objectType = vpi_get(vpiType, obj_h);
 if (objectType == vpiModule) {
   if (const int n = vpi_get(vpiTopModule, obj_h)) ...
   vpiHandle itr = vpi_iterate(vpiPort,obj_h);
   while (vpiHandle obj = vpi_scan(itr) ) {
     ...
     release_handle(obj);
   }
```

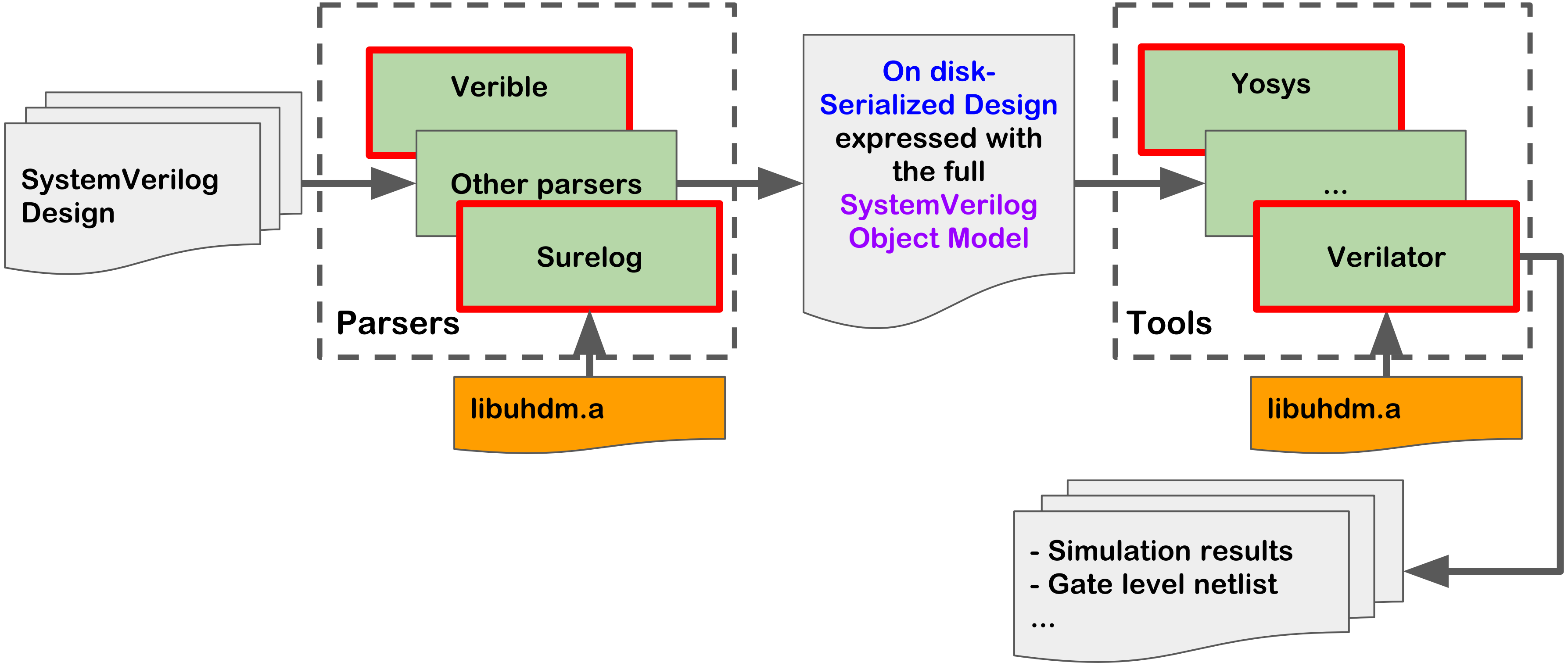
Main:

```
Serializer serializer;
const std::vector<vpiHandle>& designs = build_designs(serializer);

cout << "DUMP Design content (Pre elab):\n";
visit_designs(designs);

ElaboratorListener* listener = new ElaboratorListener(&serializer, true);
listen_designs(designs,listener);

MyVpiListener* listener = new MyVpiListener();
listen_designs(designs,listener);
```

Custom Listener:

```
class MyVpiListener : public VpiListener {
protected:
  void enterModule(const module* object, const BaseClass* parent,
                   vpiHandle handle, vpiHandle parentHandle) override {
    const char* const parentName = vpi_get_str(vpiName, parentHandle);
    std::cout << "Module: " << object->VpiName()
              << ", parent: " << ((parentName != nullptr) ? parentName : "") << std::e
```

UHDM in the Compiler flow



SureLog

First parser supporting UHDM database: **SureLog**

SystemVerilog Parser + Preprocessor

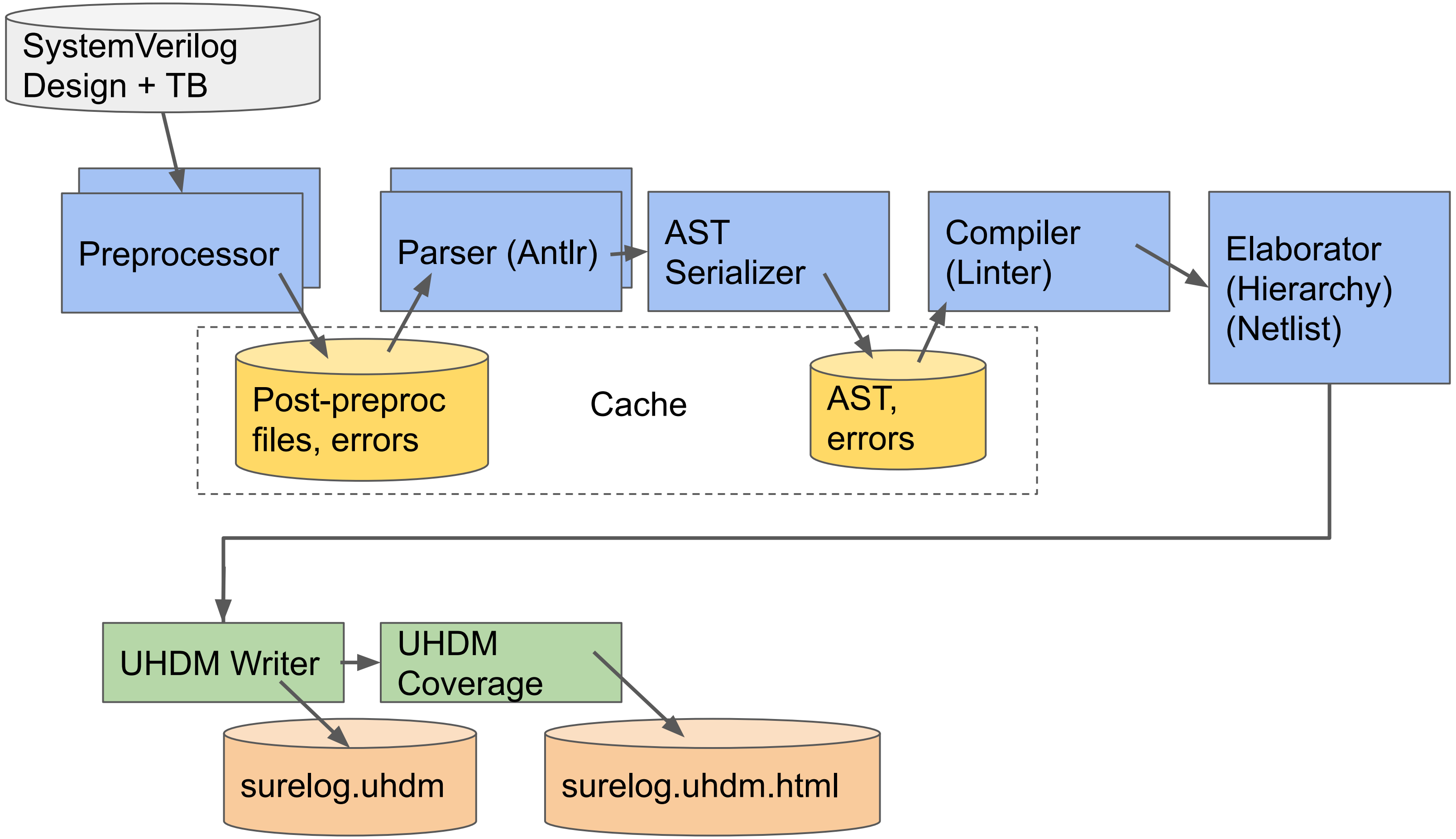
github.com/alainmarcel/Surelog

Written in C++

ANTLR with multithreading + incremental (persistent ASTs)

Semantic checks, Datatype binding, Design & UVM Elaboration, Python API, UHDM on-disk compiled output

SureLog Architecture



SureLog AST and UHDM dump example

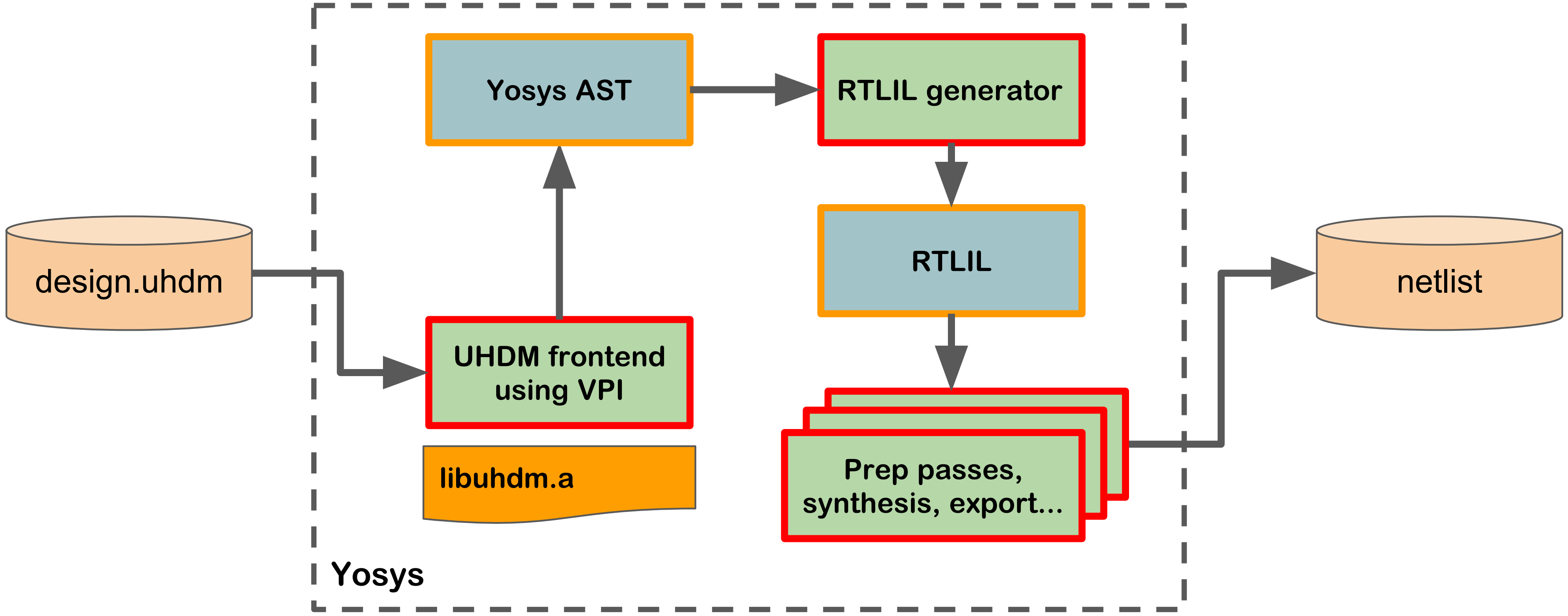
```
Test:
module top(input a);
endmodule

AST:
n<> u<0> t<Null_rule> p<14> s<13> l<1>
n<> u<1> t<Module_keyword> p<10> s<2> l<1>
n<top> u<2> t<StringConst> p<10> s<9> l<1>
n<> u<3> t<PortDir_Inp> p<6> s<5> l<1>
n<> u<4> t<Data_type_or_implicit> p<5> l<1>
n<> u<5> t<Net_port_type> p<6> c<4> l<1>
n<> u<6> t<Net_port_header> p<8> c<3> s<7> l<1>
n<a> u<7> t<StringConst> p<8> l<1>
n<> u<8> t<Ansi_port_declaration> p<9> c<6> l<1>
n<> u<9> t<List_of_port_declarations> p<10> c<8> l<1>
n<> u<10> t<Module_ansi_header> p<11> c<1> l<1>
n<> u<11> t<Module_declaration> p<12> c<10> l<1>
n<> u<12> t<Description> p<13> c<11> l<1>
n<> u<13> t<Source_text> p<14> c<12> l<1>
n<> u<14> t<Top_level_rule> l<1>
```

```
UHDM dump:

|uhdmtopModules:
\_module: work@top (work@top) tests/ScratchPad.sv:1:
  |vpiDefName:work@top
  |vpiName:work@top
  |vpiPort:
  \_port: (a), line:1, parent:work@top
    |vpiName:a
    |vpiDirection:1
    |vpiLowConn:
    \_ref_obj:
      |vpiActual:
      \_logic_net: (work@top.a), line:1, parent:work@top
        |vpiName:a
        |vpiFullName:work@top.a
  |vpiNet:
  \_logic_net: (work@top.a), line:1, parent:work@top
```

UHDM-Yosys

https://github.com/alainmarcel/uhdm-integration

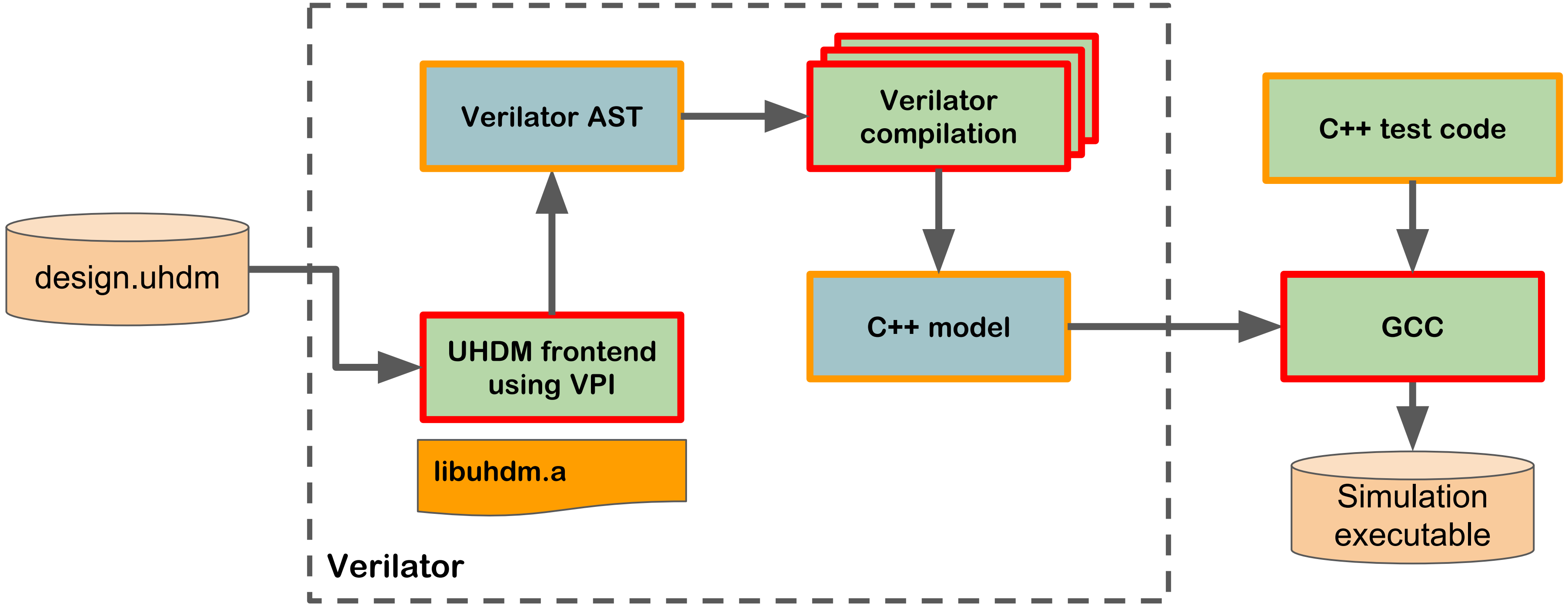
https://github.com/antmicro/yosys/tree/uhdm-yosys



UHDM-Verilator

https://github.com/alainmarcel/uhdm-integration

https://github.com/antmicro/verilator/tree/uhdm-verilator

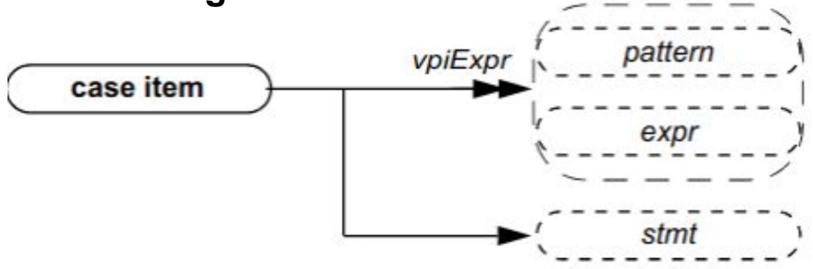


UHDM-Verilator

https://github.com/alainmarcel/uhdm-integration

https://github.com/antmicro/verilator/tree/uhdm-verilator

VPI diagram

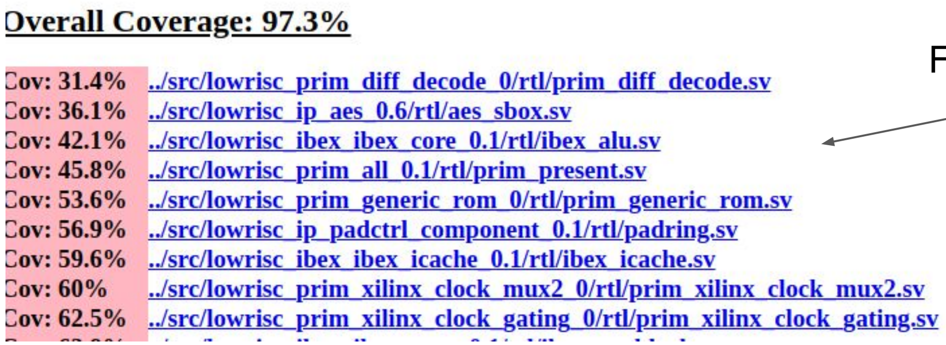


- Frontend walks the design tree (based on vpi_visitor example in UHDM)
- Functions provided for retrieving one-to-many, one-to-one relations
- UHDM AST is translated into Verilog AST, replacing Verilator parser
- The rest of Verilator flow is not modified
- Same idea for Yosys

Verilator UHDM frontend

```
case vpiCaseItem: {
        AstNode* expressionNode = nullptr;
        visit_one_to_many({vpiExpr}, obj_h, visited, top_nodes,
            [&](AstNode* item){
              if (item) {
                if (expressionNode == nullptr) {
                  expressionNode = item;
                } else {
                  expressionNode->addNextNull(item);
                }
              }
            });
        AstNode* bodyNode = nullptr;
        visit_one_to_one({vpiStmt}, obj_h, visited, top_nodes,
            [&](AstNode* node){
              bodyNode = node;
            });
        return new AstCaseItem(new FileLine("uhdm"),
                               expressionNode,
                               bodyNode);
      }
```

UHDM, Yosys, Verilator Coverage



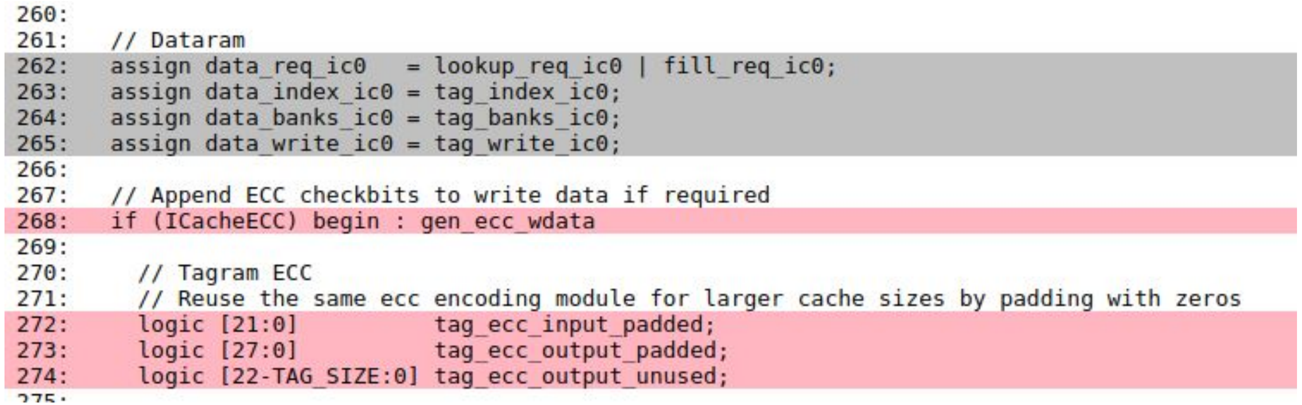
Overall Coverage: 97.3%

Cov: 31.4%	../src/lowrisc_prim_diff_decode_0/rtl/prim_diff_decode.sv
Cov: 36.1%	../src/lowrisc_ip_aes_0.6/rtl/aes_sbox.sv
Cov: 42.1%	../src/lowrisc_ibex_ibex_core_0.1/rtl/ibex_alu.sv
Cov: 45.8%	../src/lowrisc_prim_all_0.1/rtl/prim_present.sv
Cov: 53.6%	../src/lowrisc_prim_generic_rom_0/rtl/prim_generic_rom.sv
Cov: 56.9%	../src/lowrisc_ip_padctrl_component_0.1/rtl/padring.sv
Cov: 59.6%	../src/lowrisc_ibex_ibex_icache_0.1/rtl/ibex_icache.sv
Cov: 60%	../src/lowrisc_prim_xilinx_clock_mux2_0/rtl/prim_xilinx_clock_mux2.sv
Cov: 62.5%	../src/lowrisc_prim_xilinx_clock_gating_0/rtl/prim_xilinx_clock_gating.sv

File level coverage and ranking

Each stage expresses its coverage relatively to the previous stage data structure:
UHDM cov ≈ UHDM nodes / AST nodes (source lines)
Yosys cov ≈ Yosys nodes / UHDM nodes
Verilator cov ≈ Verilator nodes / UHDM nodes

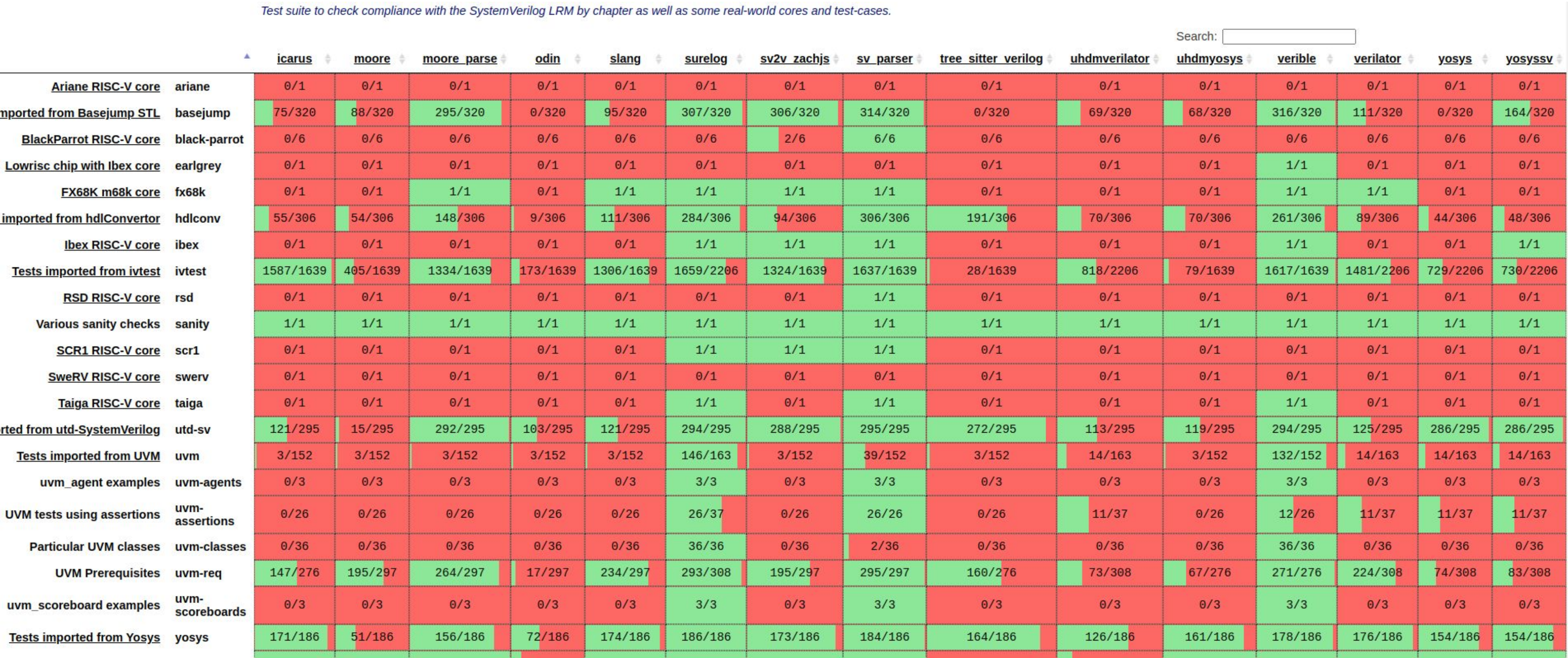
Detailed line coverage



```

  // Dataram
  assign data_req_ic0   = lookup_req_ic0 | fill_req_ic0;
  assign data_index_ic0 = tag_index_ic0;
  assign data_banks_ic0 = tag_banks_ic0;
  assign data_write_ic0 = tag_write_ic0;

  // Append ECC checkbits to write data if required
  if (ICacheECC) begin : gen_ecc_wdata

    // Tagram ECC
    // Reuse the same ecc encoding module for larger cache sizes by padding with zeros
    logic [21:0]             tag_ecc_input_padded;
    logic [27:0]             tag_ecc_output_padded;
    logic [22-TAG_SIZE:0] tag_ecc_output_unused;
```

sv-tests

https://symbiflow.github.io/sv-tests/

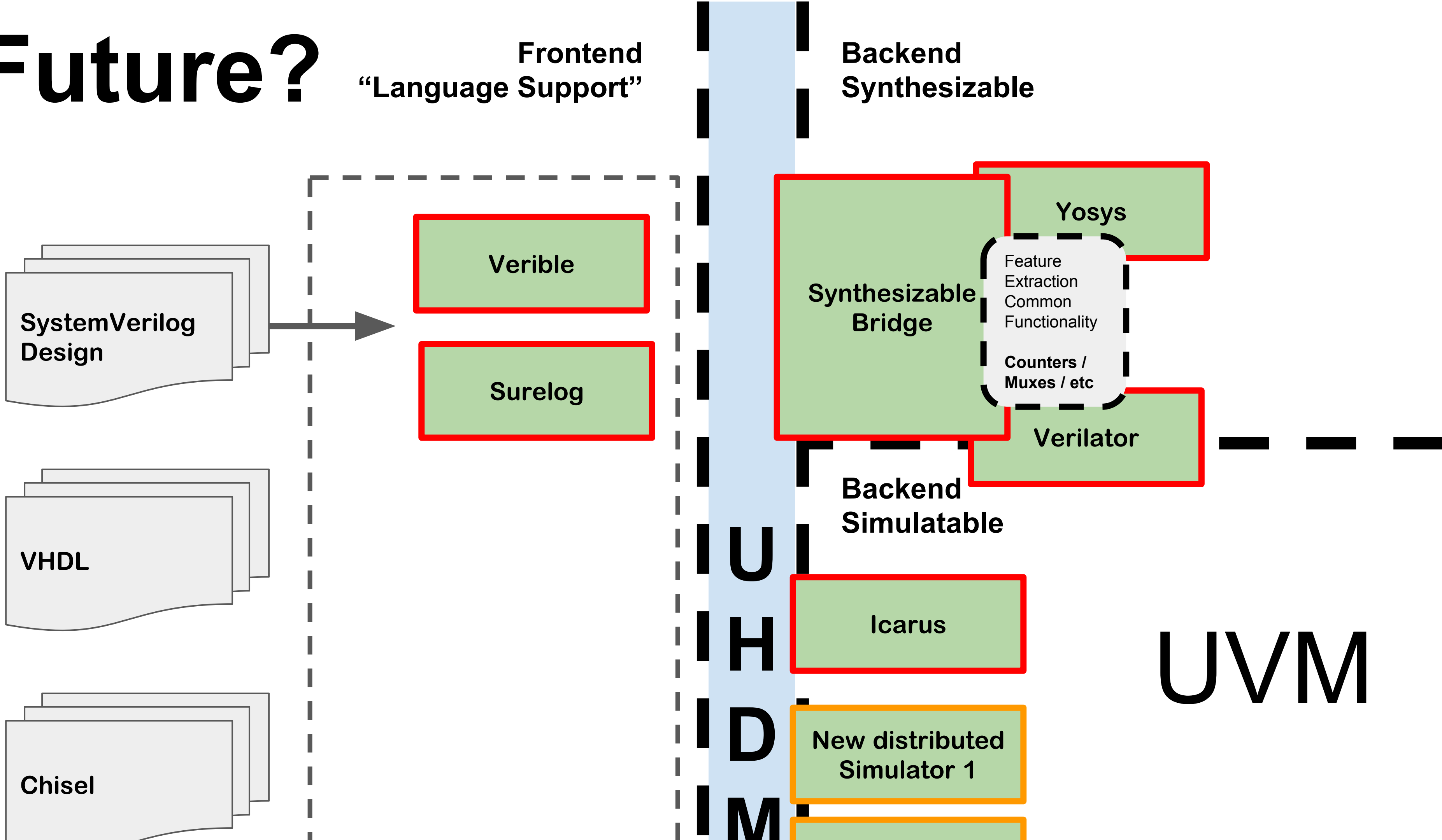


Test suite to check compliance with the SystemVerilog LRM by chapter as well as some real-world cores and test-cases.

Search:

		icarus	moore	moore_parse	odin	slang	surelog	sv2v_zachjs	sv_parser	tree_sitter_verilog	uhdmverilator	uhdmyosys	verible	verilator	yosys	yosyssv
Ariane RISC-V core	ariane	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
mported from Basejump STL	basejump	75/320	88/320	295/320	0/320	95/320	307/320	306/320	314/320	0/320	69/320	68/320	316/320	111/320	0/320	164/320
BlackParrot RISC-V core	black-parrot	0/6	0/6	0/6	0/6	0/6	0/6	2/6	6/6	0/6	0/6	0/6	0/6	0/6	0/6	0/6
Lowrisc chip with Ibex core	earlgrey	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	1/1	0/1	0/1	0/1
FX68K m68k core	fx68k	0/1	0/1	1/1	0/1	1/1	1/1	1/1	1/1	0/1	0/1	0/1	1/1	1/1	0/1	0/1
imported from hdlConvertor	hdlconv	55/306	54/306	148/306	9/306	111/306	284/306	94/306	306/306	191/306	70/306	70/306	261/306	89/306	44/306	48/306
Ibex RISC-V core	ibex	0/1	0/1	0/1	0/1	0/1	1/1	1/1	1/1	0/1	0/1	0/1	1/1	0/1	0/1	1/1
Tests imported from ivtest	ivtest	1587/1639	405/1639	1334/1639	173/1639	1306/1639	1659/2206	1324/1639	1637/1639	28/1639	818/2206	79/1639	1617/1639	1481/2206	729/2206	730/2206
RSD RISC-V core	rsd	0/1	0/1	0/1	0/1	0/1	0/1	0/1	1/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
Various sanity checks	sanity	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1
SCR1 RISC-V core	scr1	0/1	0/1	0/1	0/1	0/1	1/1	1/1	1/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
SweRV RISC-V core	swerv	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
Taiga RISC-V core	taiga	0/1	0/1	0/1	0/1	0/1	1/1	0/1	1/1	0/1	0/1	0/1	1/1	0/1	0/1	0/1
rted from utd-SystemVerilog	utd-sv	121/295	15/295	292/295	103/295	121/295	294/295	288/295	295/295	272/295	113/295	119/295	294/295	125/295	286/295	286/295
Tests imported from UVM	uvm	3/152	3/152	3/152	3/152	3/152	146/163	3/152	39/152	3/152	14/163	3/152	132/152	14/163	14/163	14/163
uvm_agent examples	uvm-agents	0/3	0/3	0/3	0/3	0/3	3/3	0/3	3/3	0/3	0/3	0/3	3/3	0/3	0/3	0/3
UVM tests using assertions	uvm-assertions	0/26	0/26	0/26	0/26	0/26	26/37	0/26	26/26	0/26	11/37	0/26	12/26	11/37	11/37	11/37
Particular UVM classes	uvm-classes	0/36	0/36	0/36	0/36	0/36	36/36	0/36	2/36	0/36	0/36	0/36	36/36	0/36	0/36	0/36
UVM Prerequisites	uvm-req	147/276	195/297	264/297	17/297	234/297	293/308	195/297	295/297	160/276	73/308	67/276	271/276	224/308	74/308	83/308
uvm_scoreboard examples	uvm-scoreboards	0/3	0/3	0/3	0/3	0/3	3/3	0/3	3/3	0/3	0/3	0/3	3/3	0/3	0/3	0/3
Tests imported from Yosys	yosys	171/186	51/186	156/186	72/186	174/186	186/186	173/186	184/186	164/186	126/186	161/186	178/186	176/186	154/186	154/186

Future?


Frontend
“Language Support”
Backend
Synthesizable
SystemVerilog
Design
VHDL
Chisel
Verible
Surelog
UHDM
Synthesizable
Bridge
Yosys
Feature
Extraction
Common
Functionality
Counters /
Muxes / etc
Verilator
Backend
Simulatable
Icarus
New distributed
Simulator 1
UVM