Portable Stimulus versus UVM: What's the Difference?

John Aynsley, Doulos

Portable Stimulus versus UVM: What's the Difference?

- What is PSS?
- What does it look like to use PSS with UVM?
- PSS versus UVM
- PSS Semantics

Copyright © 2018 by Doulos

DOULOS

What is the Portable Test and Stimulus Standard (PSS)?

The next step beyond constrained random

Copyright © 2018 by Doulos

DOULOS

What is the Portable Test and Stimulus Standard (PSS)?

Constrained random on steroids

Copyright © 2018 by Doulos

DOULOS

What is the Portable Test and Stimulus Standard (PSS)?

An Accellera standard

Copyright © 2018 by Doulos

DOULOS

What is the Portable Test and Stimulus Standard (PSS)?

A new language, not SystemVerilog or C++

Copyright © 2018 by Doulos

DOULOS

What is the Portable Test and Stimulus Standard (PSS)?

Written as DSL or C++

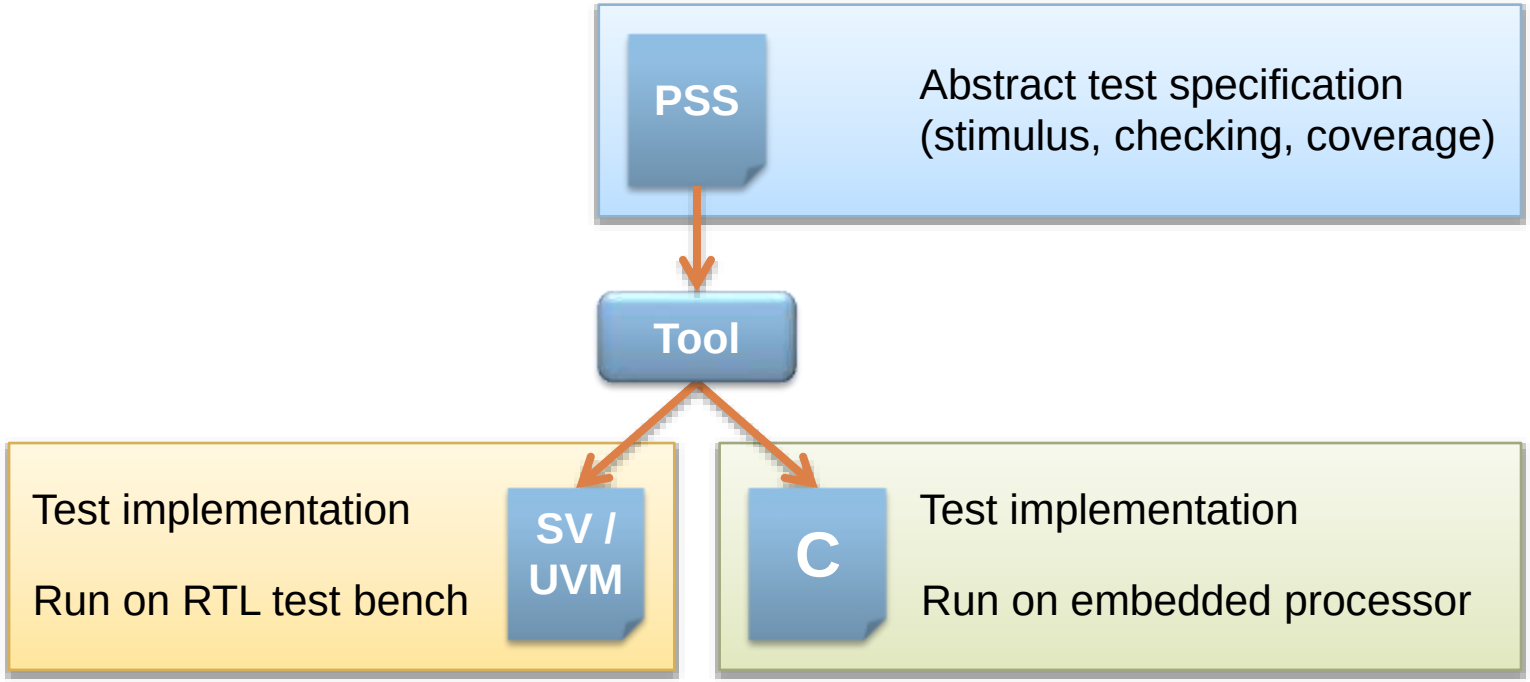
```
component uart_c {
  action configure {
    rand modes_e mode;
    constraint {mode != A};
}
```

```
class uart_c : public component { ...
  class configure : public action { ...
    rand_attr<modes_e> mode{"mode"};
    constraint c {mode != modes_e::A};
  };
  type_decl<configure> configure_decl;
};
```

Copyright © 2018 by Doulos

DOULOS

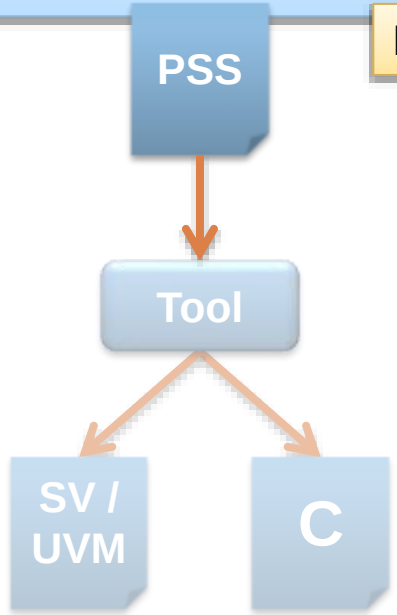
What is the Portable Test and Stimulus Standard (PSS)?



Copyright © 2018 by Doulos

Stakeholders

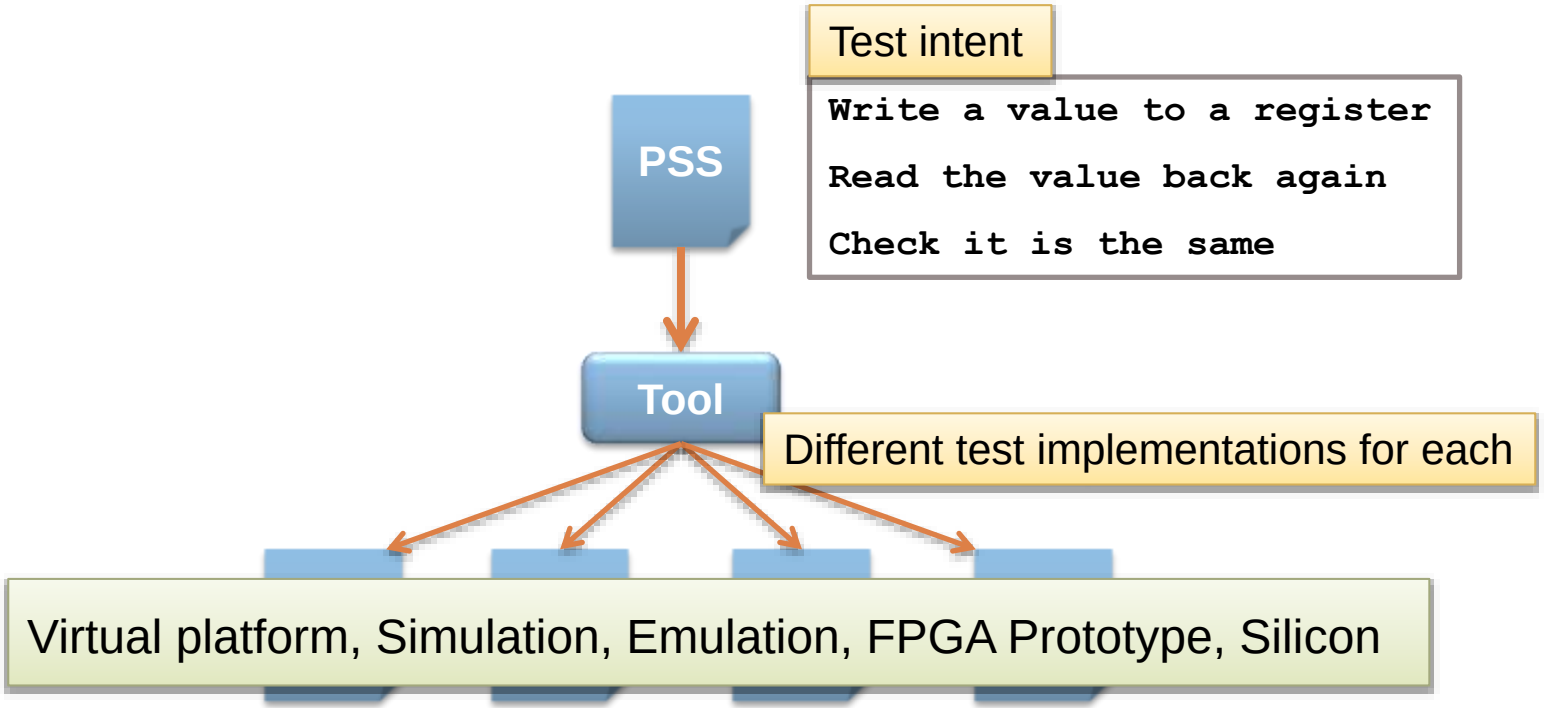
Architect, HW Developer, SW Developer, Verification Engineer, SW Test, Post-Silicon Validation



Copyright © 2018 by Doulos

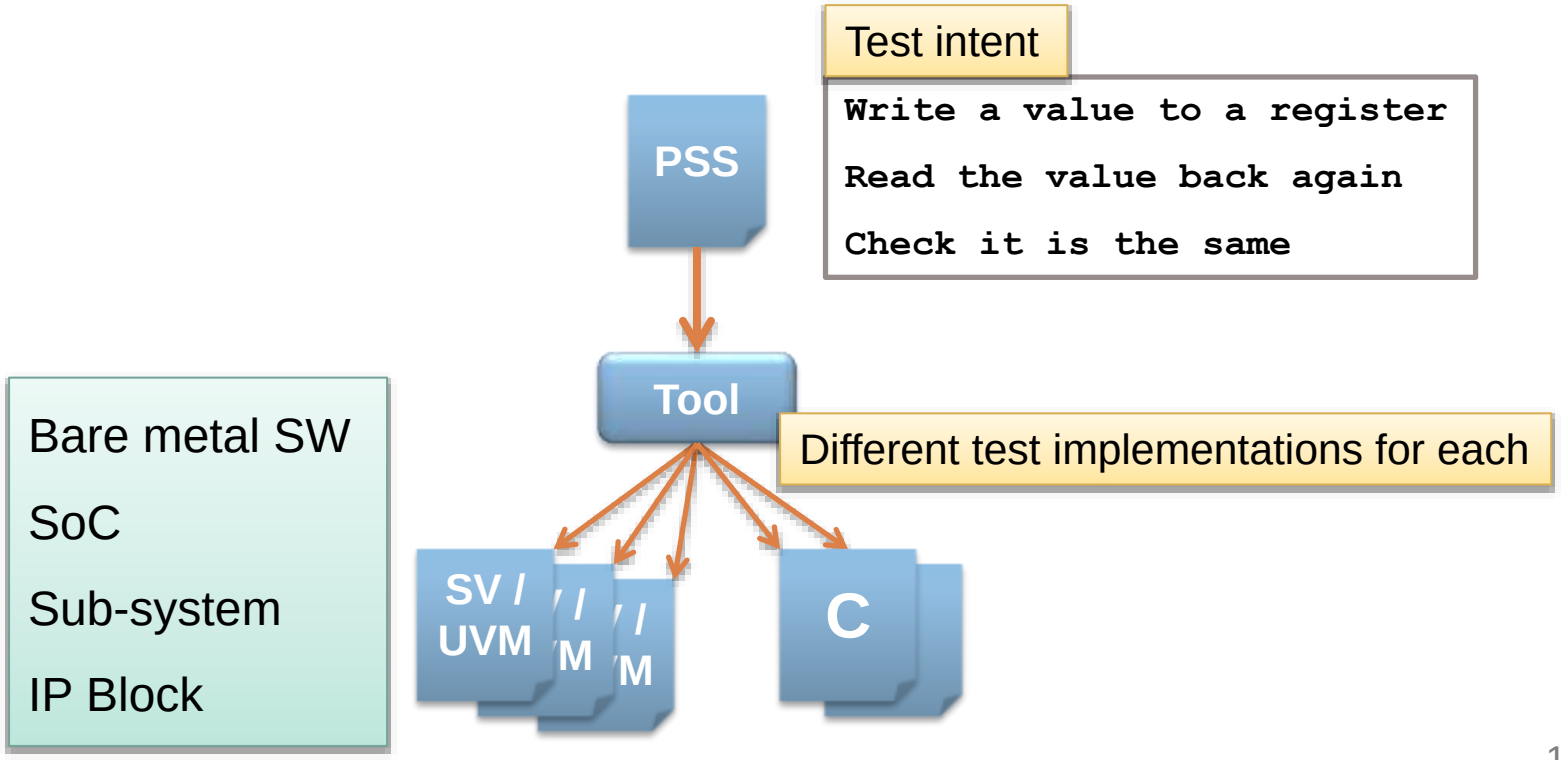
DOULOS

Platforms



Copyright © 2018 by Doulos

Scope, Vertical Reuse



Copyright © 2018 by Doulos

PSS First Release

Digital	✓	Hardware verification	✓
Power	✓	System validation	✓
Analog	✗	Middleware, O/S, applications	✗
AMS	✗		
		Multicore	✓
		Cache coherency	✓
		Performance	✓

Copyright © 2018 by Doulos

DOULOS

Portable Stimulus versus UVM: What's the Difference?

What is PSS?

What does it look like to use PSS with UVM?

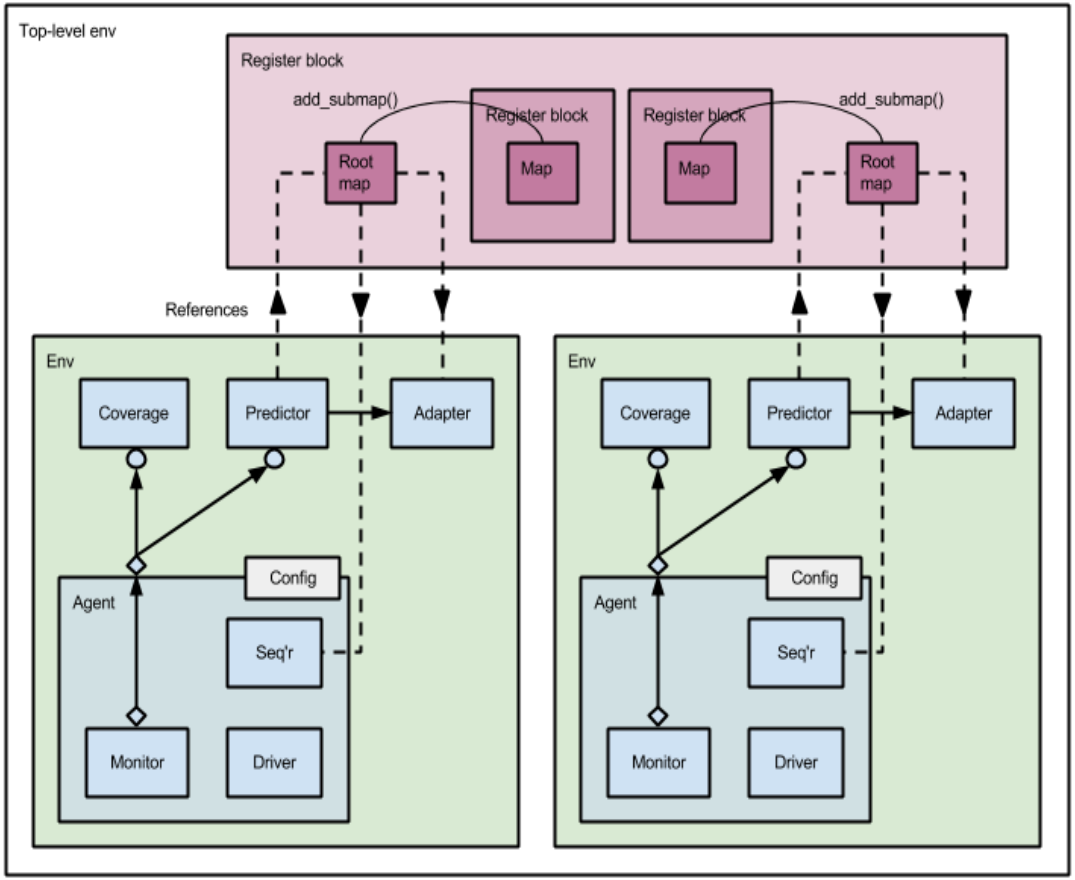
PSS versus UVM

PSS Semantics

Copyright © 2018 by Doulos

DOULOS

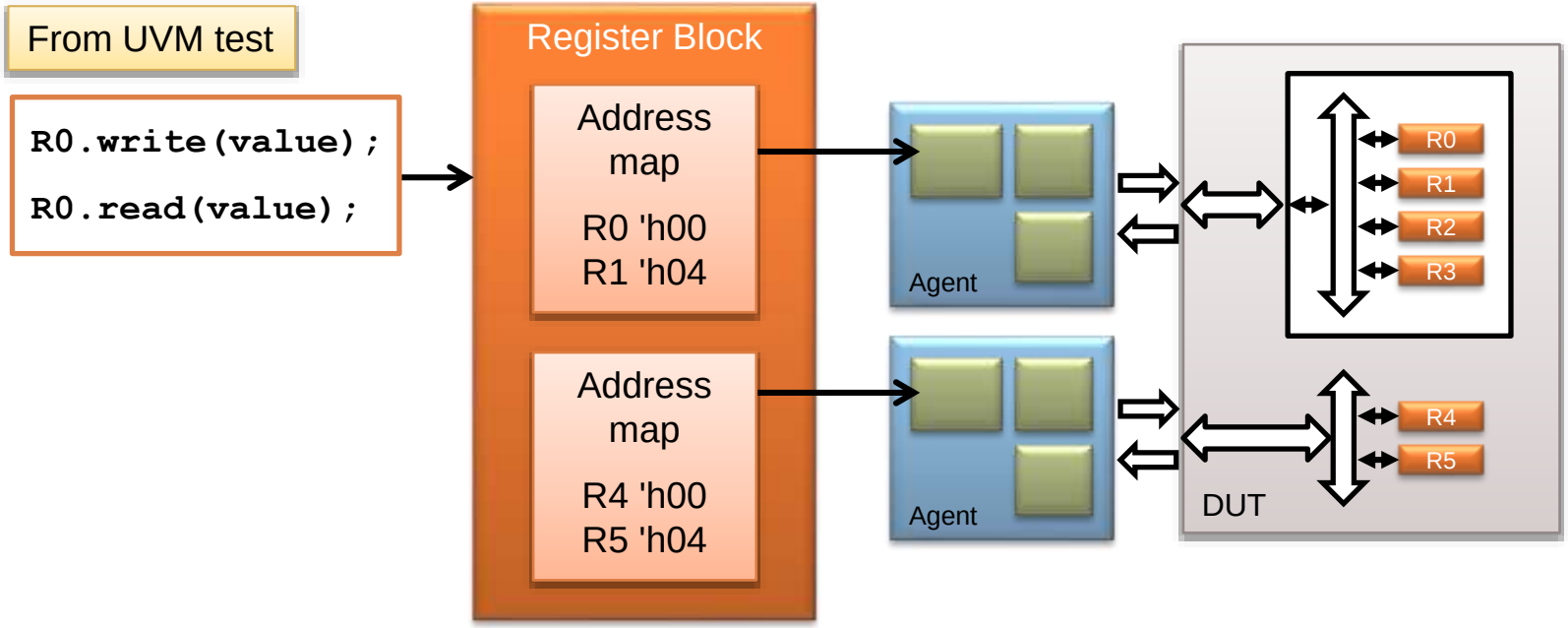
Native UVM – Register Model



Copyright © 2018 by Doulos

DOULOS

UVM Register Layer



Copyright © 2018 by Doulos

DOULOS

Native UVM Register Test

```
class bus_env_reg_seq extends bus_env_default_seq;

  `uvm_object_utils(bus_env_reg_seq)

  function new(string name = "");
    super.new(name);
  endfunction

  task body;
     regmodel.R0.write(status, .value('hab), .parent(this));
     assert(status == UVM_IS_OK);

     regmodel.R0.read(status, .value(data), .parent(this));
     assert(data == 8'hab);
  endtask

endclass
```

Copyright © 2018 by Doulos

DOULOS

Expose Individual Sequences

```
class write_seq extends bus_env_default_seq;
  ...
  task body;
     regmodel.R0.write(status, .value(value), .parent(this));
  endtask
endclass

class read_seq extends bus_env_default_seq;
  ...
```

```
class test_base extends uvm_test;
  ...
  task write(byte value);        // Runs write_seq
    ...
  task read(output byte value); // Runs read_seq
    ...
endclass
```

Details are tool-specific

Copyright © 2018 by Doulos

DOULOS

Call UVM Functions from PSS

```
buffer data_buff_s {
  rand bit [7:0] data;
}
```

```
action write_data {
  output data_buff_s dout;

  exec body SV = """
    write( {{dout.data}} );
  """
}

action read_data {
  input data_buff_s din;

  exec body SV = """
    read( {{din.data}} );
  """
}
```

Copyright © 2018 by Doulos

DOULOS

A PSS Test Scenario

```
component my_reg {
  action write_data {...}
  action read_data {...}

  action write_then_read {
    activity {
      do write_data;
      do read_data;
    }
  }
}
```

Copyright © 2018 by Doulos

DOULOS

Generated UVM Code

PSS

```
component my_reg {
  action write_data {...}
  action read_data {...}

  action write_then_read {
    activity {
      do write_data;
      do read_data;
    }
  }
}
```

UVM

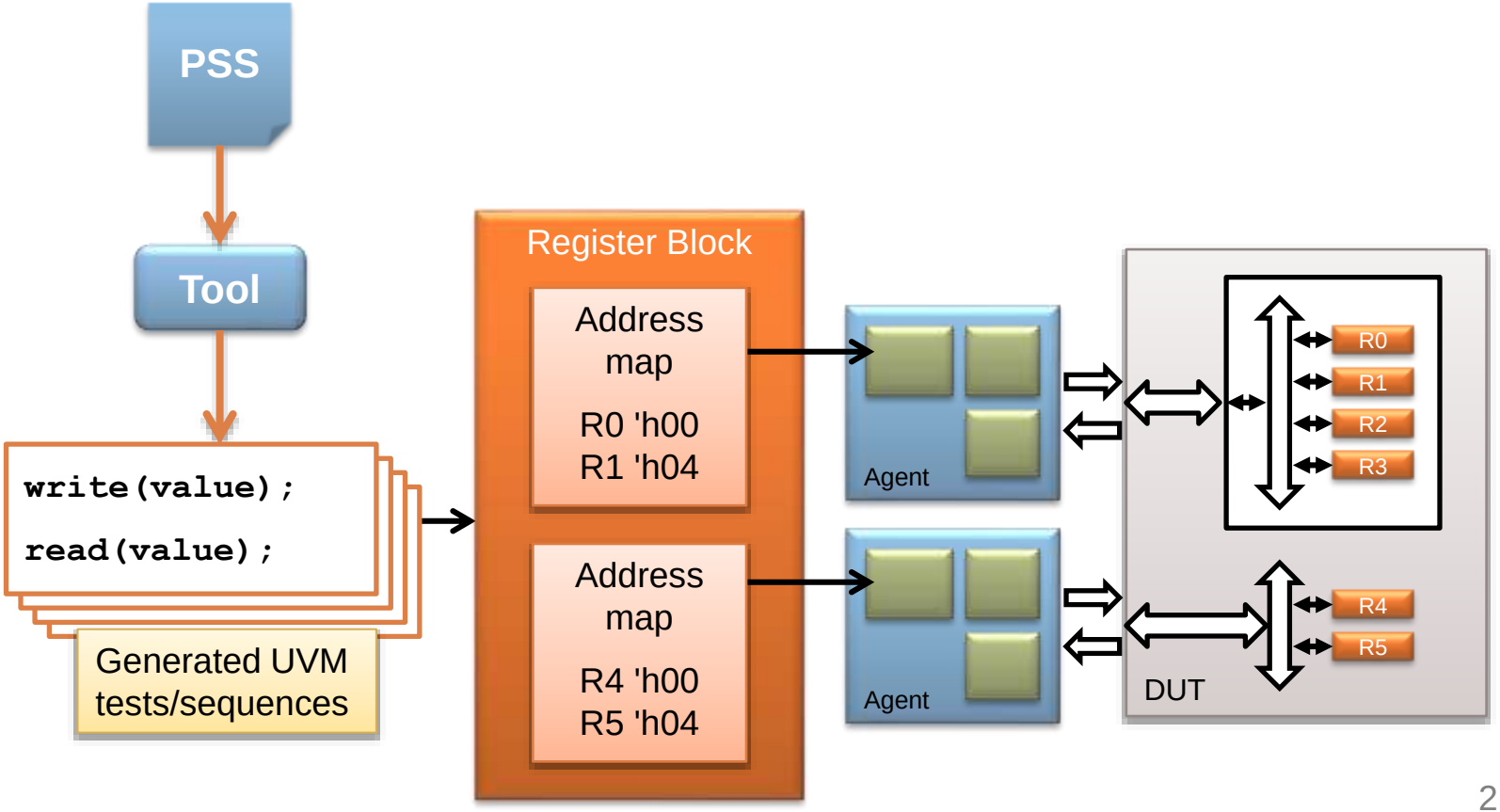
```
class bus_env_reg_seq extends ...
  `uvm_object_utils(bus_env_reg_seq)
  ...
  task body;
     write('hab);
     read(data);
  endtask

endclass
```

Details are tool-specific

Copyright © 2018 by Doulos

DOULOS

PSS + UVM



Copyright © 2018 by Doulos

Portable Stimulus versus UVM: What's the Difference?

What is PSS?

What does it look like to use PSS with UVM?

PSS versus UVM

PSS Semantics

Copyright © 2018 by Doulos

DOULOS

Generated UVM Code

PSS

```
component my_reg {
  action write_data {...}
  action read_data {...}

  action write_then_read {
    activity {
      do write_data;
      do read_data;
    }
  }
}
```

Constraints (declarative)

UVM

```
class bus_env_reg_seq extends ...
  `uvm_object_utils(bus_env_reg_seq)
  ...
  task body;
     write('hab);
     read(data);
  endtask

endclass
```

Procedural

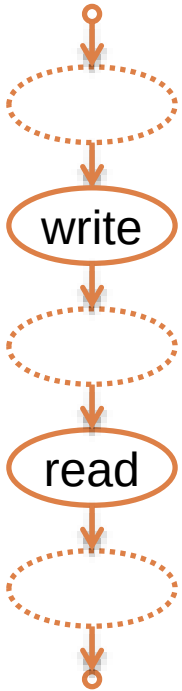
Copyright © 2018 by Doulos

DOULOS

PSS versus UVM

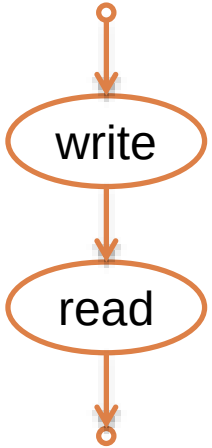
PSS

```
activity {
  do write_data;
  do read_data;
}
```



UVM

```
task body;
  write(...);
  read(...);
```



Copyright © 2018 by Doulos

DOULOS

Procedural versus Constraints

SystemVerilog

Procedural	Constraints
`int v;`	`int v;`
`initial v = 2;`	`constraint c1 { v == 2; }`
`initial v = ??`	`constraint c2 { v > 3; v < 8; }`

Copyright © 2018 by Doulos

DOULOS

Constraints are Composable

Constraints

```
constraint c2 { v > 3; v < 8; }
```

```
constraint c3 { v > 5; v < 12; }
```

Composition

```
constraint c4 { v > 5; v < 8; }
```

Procedural

```
initial
  v = 6
```

```
initial
  v = 7
```

Contradiction!

Copyright © 2018 by Doulos

DOULOS

PSS versus UVM

In PSS, everything is a constraint (almost)!

Copyright © 2018 by Doulos

Portable Stimulus versus UVM: What's the Difference?

What is PSS?

What does it look like to use PSS with UVM?

PSS versus UVM

PSS Semantics

Copyright © 2018 by Doulos

DOULOS

More Actions

```
buffer data_buff_s {
  rand bit [7:0] data;
}
```

Buffer must be written before it can be read

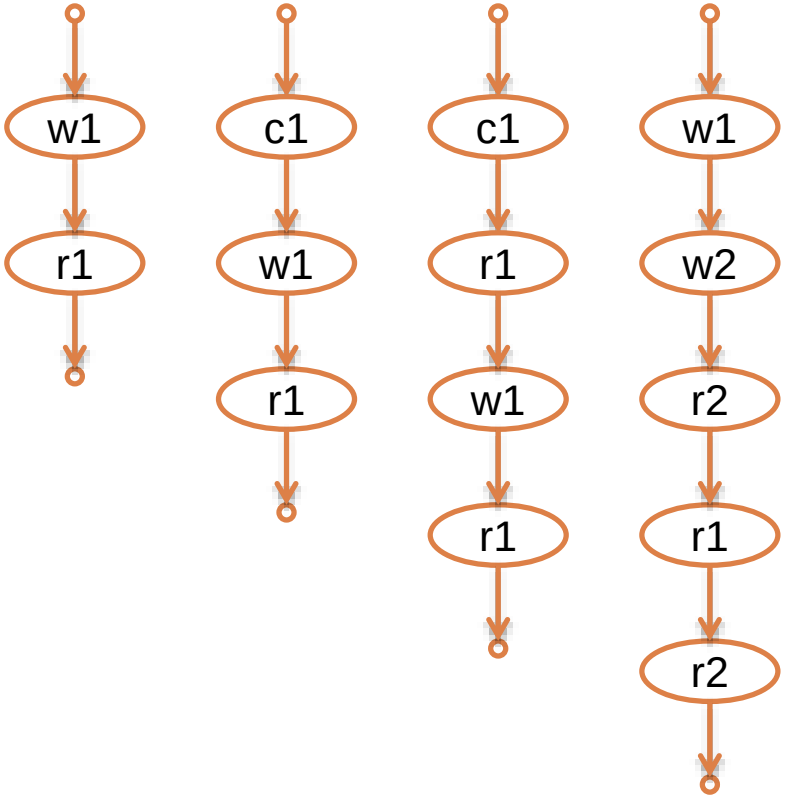
```
action write_data {
  output data_buff_s dout;
  ...
}

action read_data {
  input data_buff_s din;
  ...
}

action clear_data {
  output data_buff_s dout;
  ...
}
```

Copyright © 2018 by Doulos

DOULOS

Possible Schedules

```
write_data w1, w2;
read_data  r1, r2;
clear_data c1, c2;

activity {
  w1;
  r1;
}
```

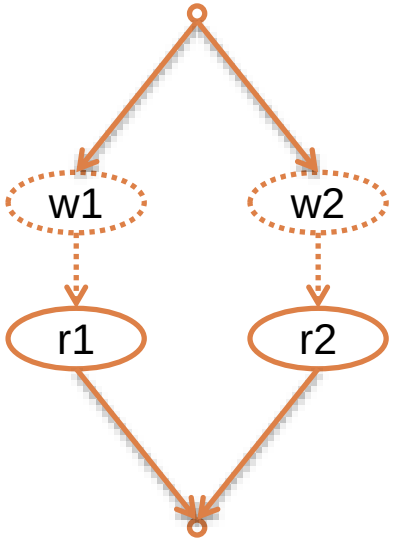


Copyright © 2018 by Doulos

DOULOS

Further Activities - Select

```
write_data w1, w2;
read_data  r1, r2;

activity {
  select {
    r1;
    r2;
  }
}
```

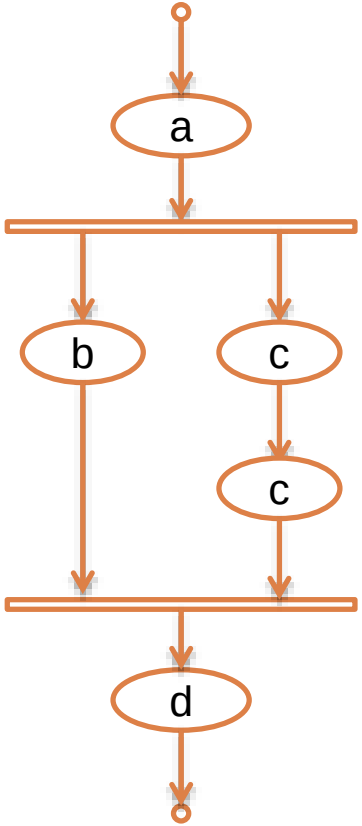


Copyright © 2018 by Doulos

DOULOS

Further Activities – Parallel, Repeat

```
activity {
  a;
  parallel {
    b;
    repeat (2) {
      c;
    }
  }
  d;
}
```



Copyright © 2018 by Doulos

DOULOS

Flow Objects and Resources

buf	Buffer must be written before being read
str	Stream must be written and read in parallel
sta	State get initialized, cannot be written concurrently
rsc	Resource can be locked, shared, and pooled

Copyright © 2018 by Doulos

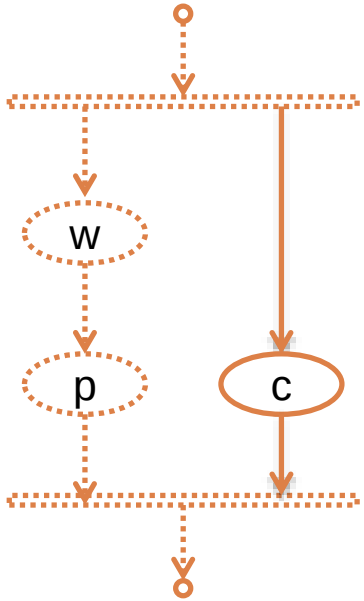
DOULOS

Inferred Actions

```
buffer data_buff_s {...}
stream data_stream_s {...}

action produce_data {
  input  data_buf_s    din;
  output data_stream_s dout;
...}

action consume_data {
  input  data_stream_s din;
...}
```

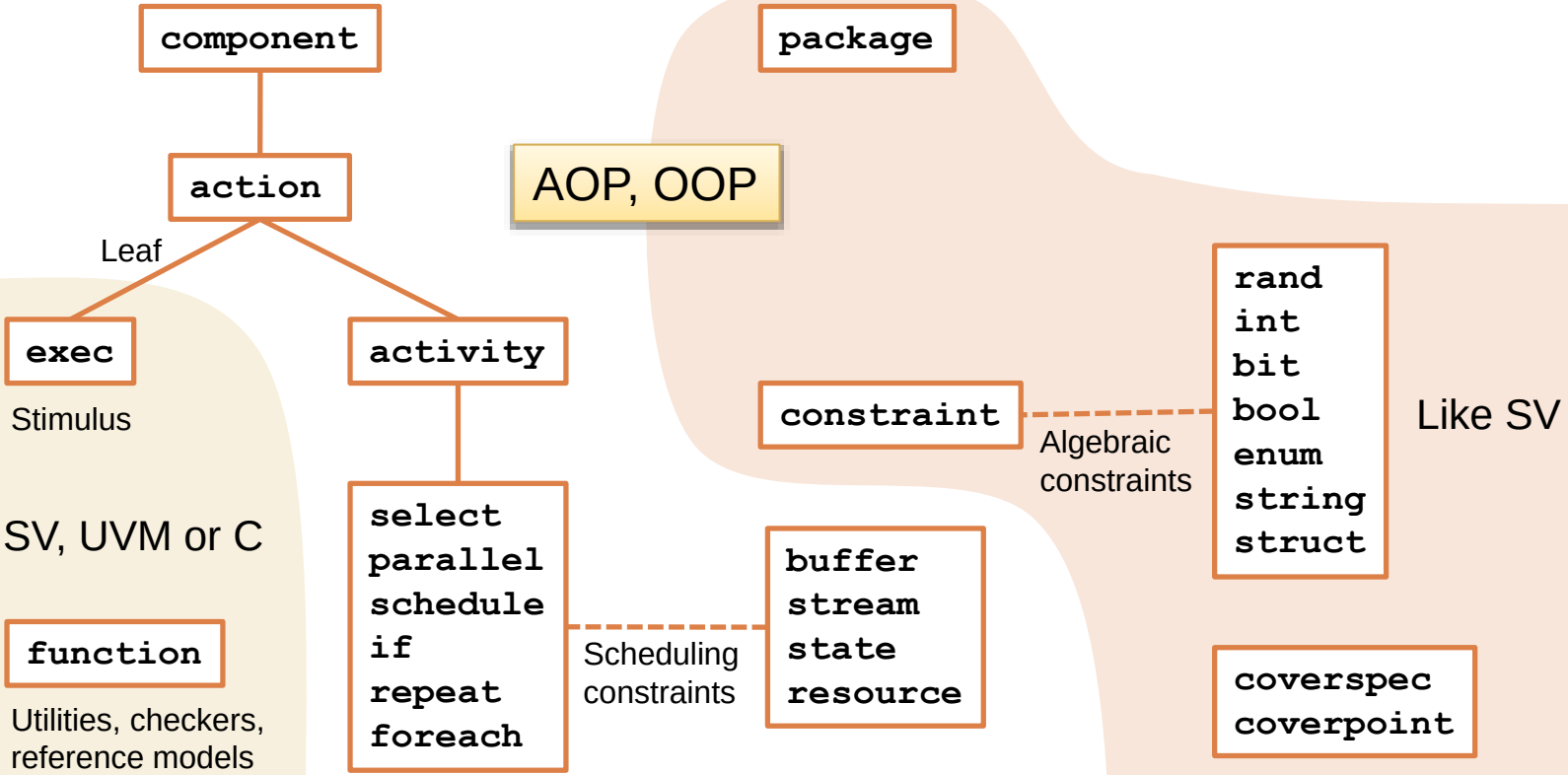
```
write_data   w;
produce_data p;
consume_data c;
```

```
activity {
  c;
}
```



Copyright © 2018 by Doulos

PSS as a Language



Copyright © 2018 by Doulos

DOULOS

Summary

PSS	UVM
For HW, SW, and System test	For hardware verification
Architects, HW, SW, post-silicon	For UVM experts
A new language	SystemVerilog Class Library
Constraints only	Large, complex, complete
Generates tests in C or UVM	Standalone

Copyright © 2018 by Doulos

DOULOS

DOULOS

Delivering KnowHow www.doulos.com

SoC Design & Verification

» SystemVerilog » UVM
» SystemC » TLM-2.0 » Arm

FPGA & Hardware Design

» VHDL » Verilog » Perl » Tcl
» Xilinx » Intel (Altera)

Embedded Software

» Emb C/C++ » Emb Linux
» Yocto » RTOS » Security » Arm

Python & Deep Learning