

VMM Standard Library Overview

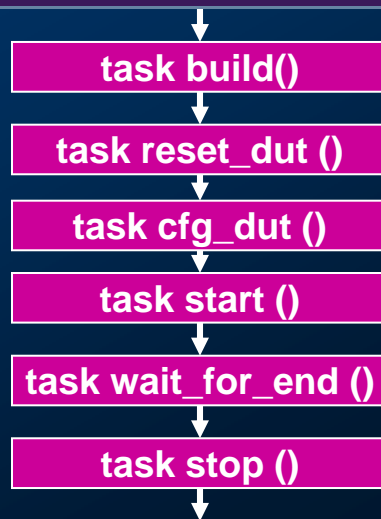
Robert Freeman
Verification Group
Synopsys Inc

November 2005

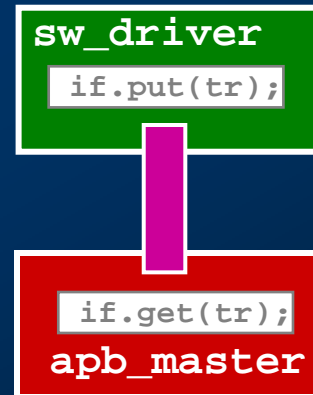


VMM Standard Library

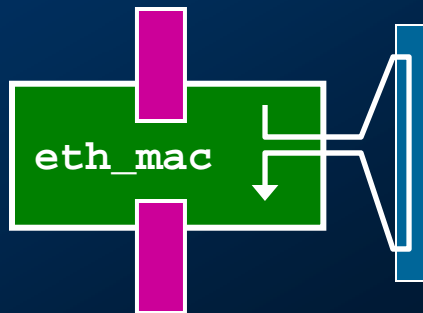
Organizational Management



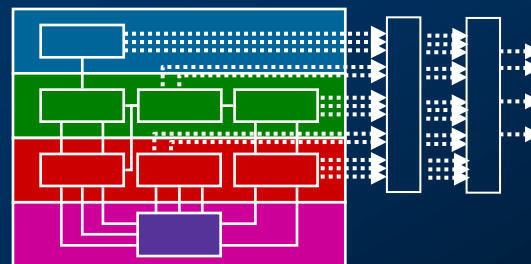
Transaction-Level Channel Interface



Reusable Transactors

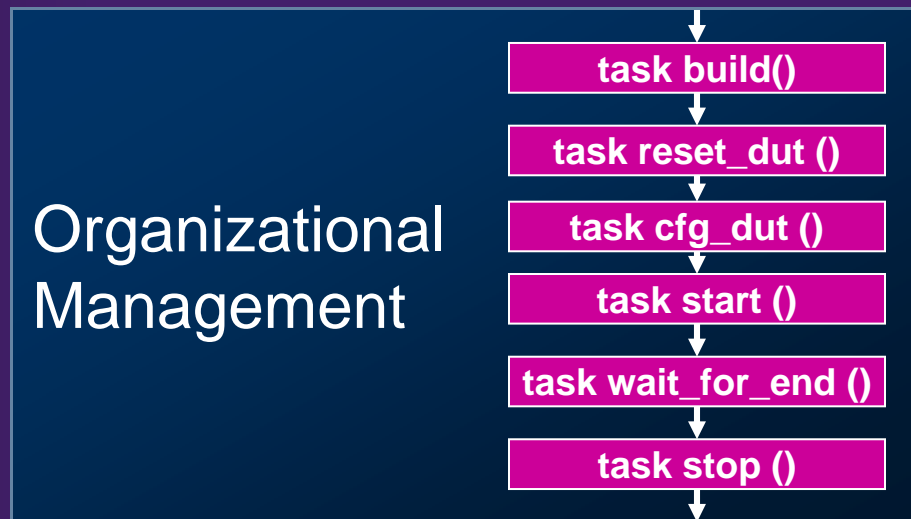


Messaging Service



Organizational Management

vmm_env



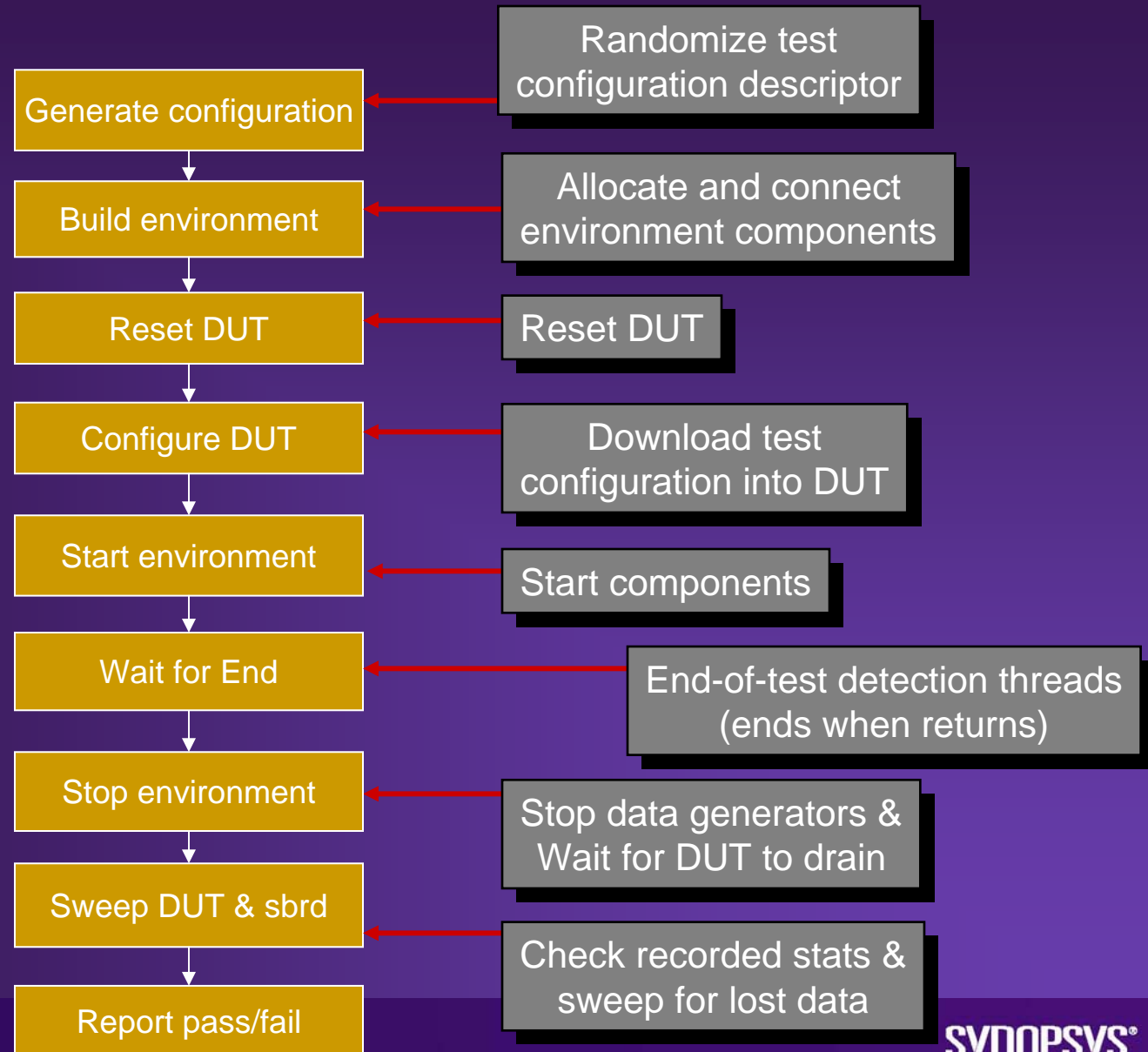
Organizational Management

vmm_env

- A Standardized Testbench Format
 - Enforces organizational rules
 - Scales well
- Environment Creation takes less time
- Testbench Integration is simplified
- Configurations are easy to specify and test
- Quickly make specific tests
 - without clobbering anyone else's project work
- Enables IP reuse

Organizational Management

vmm_env



Organizational Management

vmm_env

- Simplest test

```
program test (...);  
    verif_env env = new;  
  
    env.run();  
endprogram
```

Run() knows to follow the steps as explained on prev slide

- Initial debug, trivial test

```
program test (...);  
    verif_env env = new();  
  
    env.gen_cfg();  
    env.randomized_cfg.run_for_n_frames = 1;  
    env.run();  
endprogram
```

Organizational Management

vmm_env

- Corner-case test

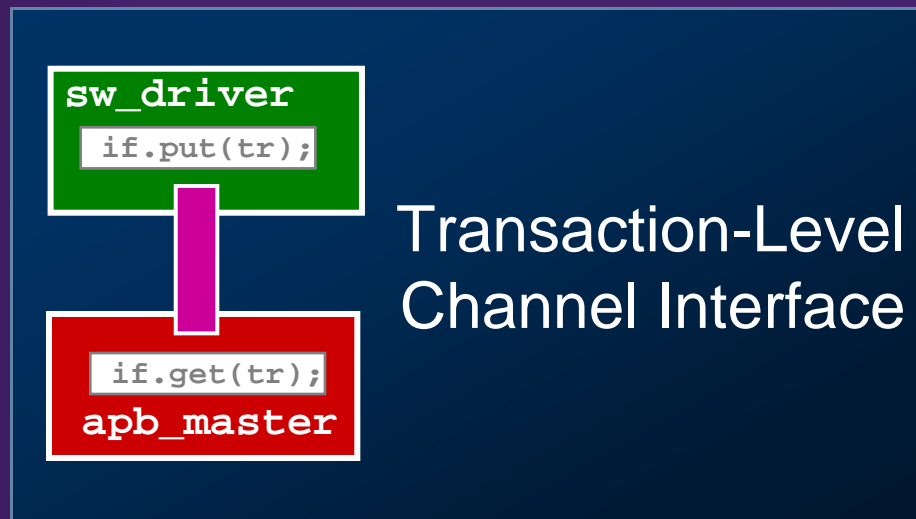
```
class my_eth_frame extends eth_frame ;
    bit [47:0] mac_address;
    constraint one_port_only {
        da == mac_address;
    }
endclass

program test (...);
    verif_env env = new();

    env.build();
    begin
        my_eth_frame my_fr = new();
        my_fr.mac_address = env.randomized_cfg.mac_address[0];
        env.src[0].randomized_fr = my_fr;
    end
    env.run();
endprogram
```

Data and Dataflow Management

vmm_data



Data Management

vmm_data

- Establishes a standard format for representing data and transactions
 - Encapsulates data and transactions naturally
 - Standardized method names, behavior for all transactions
- One component handles the same data
 - Irregardless if it is randomized, directed or modified data
- Code is easier to use
- Used as the basis for other VMM components

Data Management

vmm_data

da	sa	len	data	fcs
48 bits	48 bits	16 bits	<i>len</i> x 8 bits	32 bits

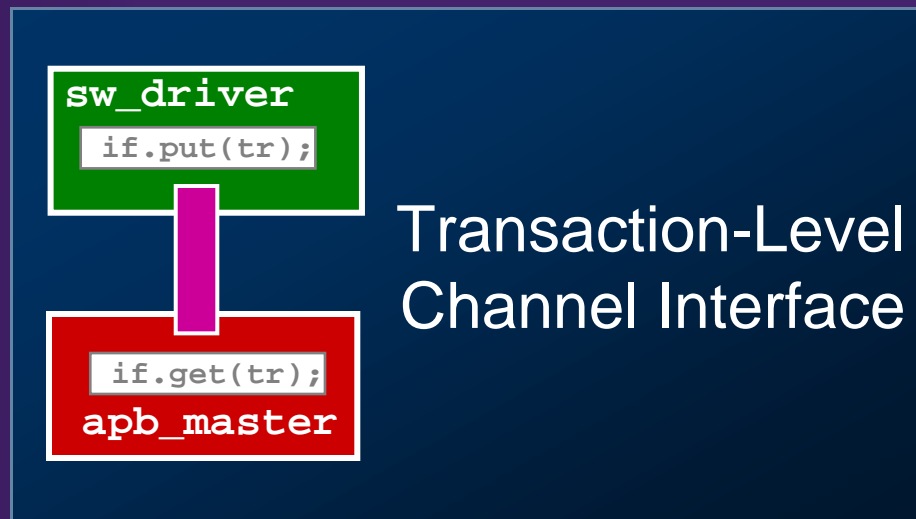
```
class eth_mac_frame extends vmm_data;
  // actual data
  rand bit [47:0] da;
  rand bit [47:0] sa;
  rand bit [15:0] len;
  rand bit [ 7:0] data[$];
  rand bit [31:0] fcs;

  // controls randomization
  constraint legal_spec {
    sa[41:40] == 2'b0;
    len      in {46:1500};
    data.size() == len;
    fcs      == 32'h0000_0000;
  }

  // some of the many built in methods
  virtual task display(string prefix = "");
  virtual function vmm_data allocate();
  virtual function bit compare(vmm_data to);
endclass
```

Data and Dataflow Management

vmm_channel



Dataflow Management

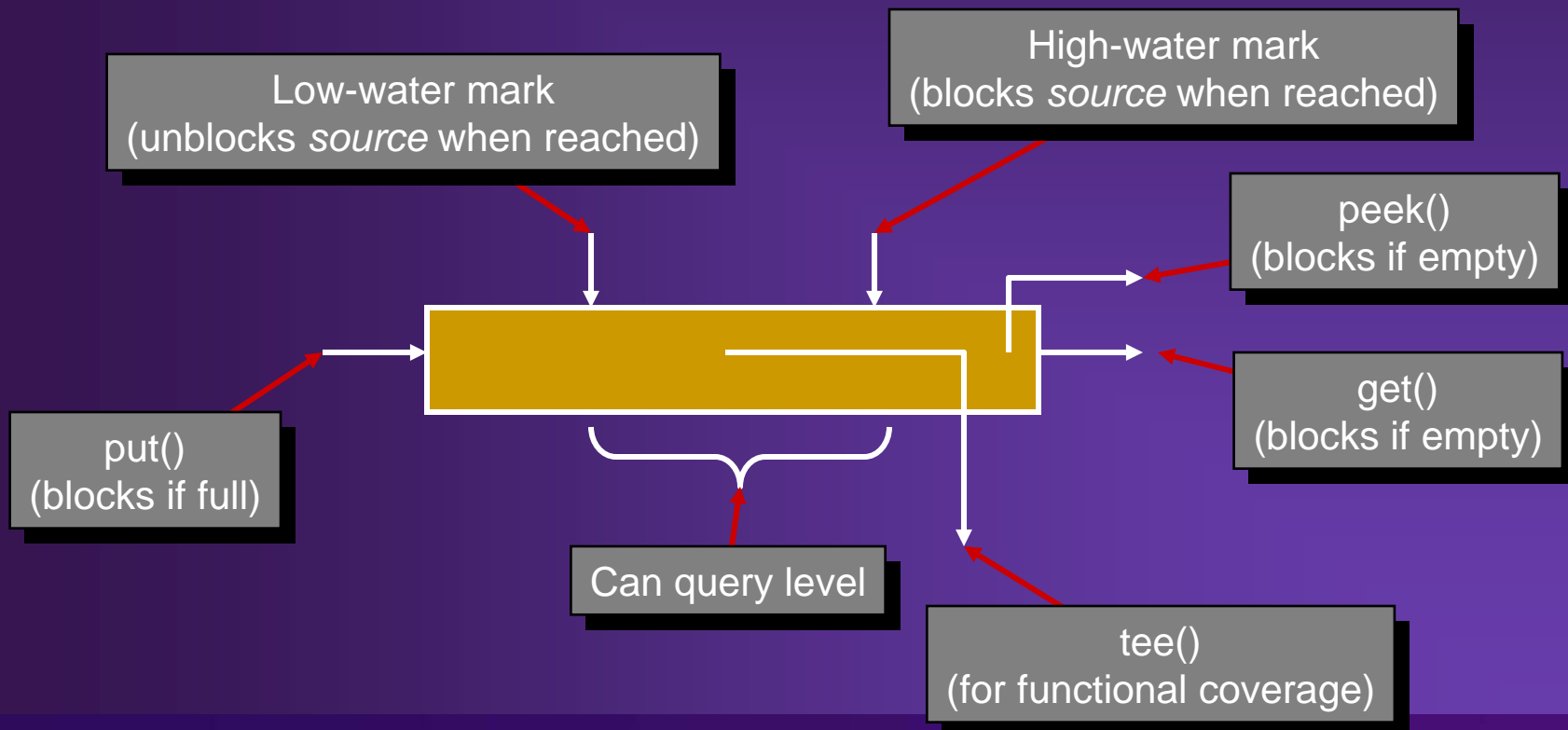
vmm_channel, vmm_broadcast, vmm_scheduler

- Simplifies environment
 - Establishes links between testbench components
 - Decouples components
 - Implements flow control in both receiver and sender
 - Channel depth configuration and blocking put
 - Broadcast allows controlled copying of transactions
 - Scheduler allows controlled merging of transactions
- Increases reuse
 - Well defined channel interface between components
 - Eliminates interface dependency between components
 - Allows for Plug and play of components

Dataflow Management

vmm_channel

- Strongly typed, self-regulated transaction interface object
 - Supports out-of-order submission and extraction



Dataflow Management

vmm_channel

- Carries only *vmm_data* derivatives
- Declared using macro
 - In same file that declares data model
 - Defines new class

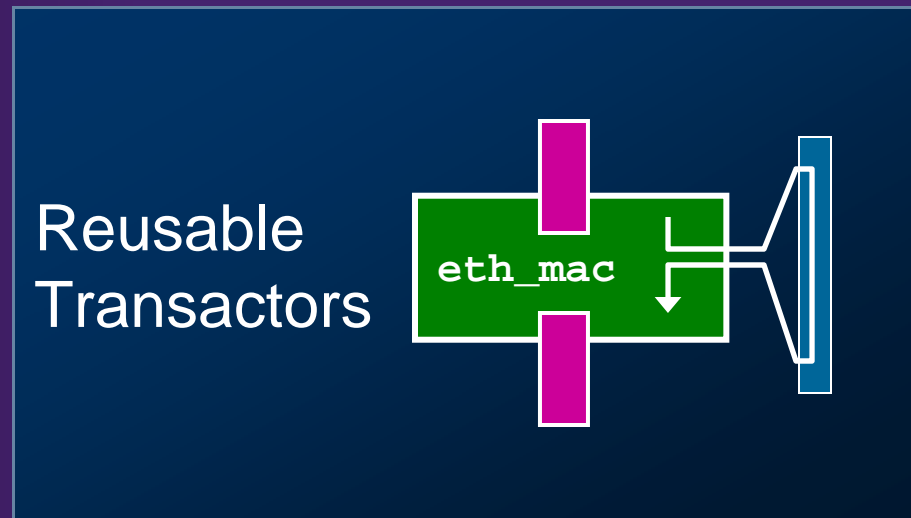
```
class eth_mac_frame extends vmm_data ;  
  ...  
endclass  
vmm_channel(eth_mac_frame)
```

```
eth_mac_frame_channel ch = new(...);  
fork  
  while (1) begin  
    eth_mac_frame fr = new();  
    ch.put(fr);  
  end  
  while (1) begin  
    eth_mac_frame fr = ch.get();  
  end  
join
```

Both threads now
self-regulated

Transaction Management

vmm_xactor



Transaction Management

vmm_xactor

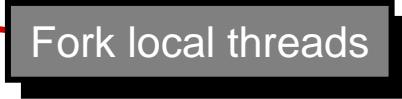
- A vmm_xactor is any process that creates, transforms or collects data
 - Stimulus Generators
 - Bus Functional Model (BFM)
 - Abstraction Layer
- Establishes a uniform use model for all components:
 - Configuration
 - dataflow through vmm_channel
 - Start/stop/reset
 - Easy modification to behavior through callbacks
- Easy to substitute Transaction Level Model with Bus Functional Model
 - vmm_channel dataflow management
- Easy to reuse as IP

Transaction Management

vmm_xactor

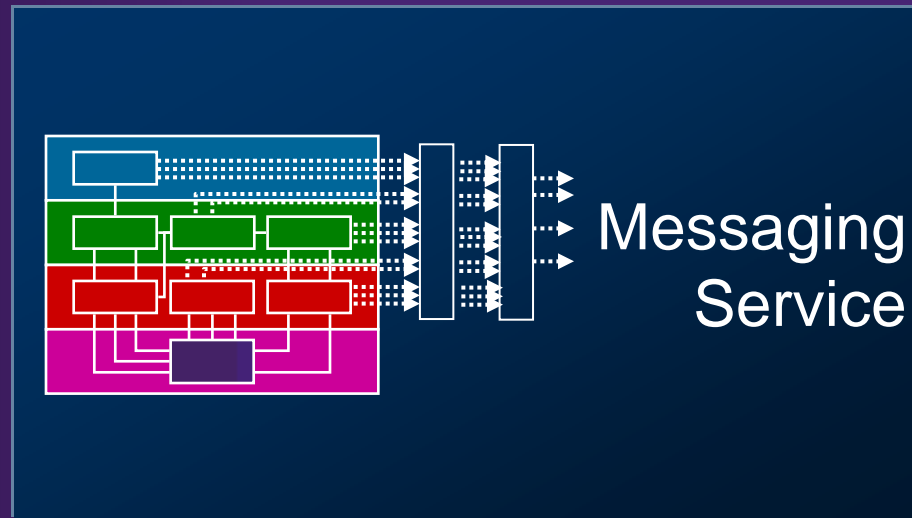
- Standard transactor management methods
 - *start_xactor()*
 - Forks *main()*
 - *stop_xactor()*
 - Blocks *wait_if_stopped()*
 - *reset_xactor()*
 - Aborts *main()*
- Implement functionality in *main()*
 - Can be blocking or nonblocking

```
class mii_phy extends vmm_xactor ;
  virtual task start_xactor() ;
    super.start_xactor();
    ...
  endtask
  virtual task stop_xactor() ;
    super.stop_xactor();
    ...
  endtask
  virtual task reset_xactor(...);
    super.reset_xactor(...);
    ...
  endtask
  virtual task main() ;
    ...
  endtask
endclass
```



Messaging Management

vmm_log



Messaging Management

vmm_log

- Log files and output are easier to read
 - All messages have a similar format
 - Standard severity levels
 - Report files are easier to post-process
- Log files and output are smaller and easier to manage
 - Built in methods to filter messages
 - Log files contain only messages of interest
- Messages can be promoted/demoted
 - Consistent method for promoting messages as errors
 - Consistent method for demoting errors to warnings

Messaging Management

vmm_log

- Both Classes and Macros are offered:
 - Macros offer default type or severity

```
`vmm_error(vmm_log log, string msg);  
`vmm_fatal(vmm_log log, string msg);  
`vmm_warning(vmm_log log, string msg);  
`vmm_note(vmm_log log, string msg);  
`vmm_trace(vmm_log log, string msg);  
`vmm_debug(vmm_log log, string msg);  
`vmm_verbose(vmm_log log, string msg);
```

- Example:

```
vmm_verbose(log, "Checking byte that was just received");  
if (byte != expect) begin  
    vmm_error(log, "Bad data: failure in XX BFM");  
end
```

VMM Standard Library Resources

- VMM book
 - Complete specification for VMM Standard Library
 - More info at <http://www.vmm-sv.org>
- Verification Guild
 - <http://www.verifictionguild.com>
- EDA tool support
 - VCS
 - Pioneer-NTB

