UNIVERSITEIT GENT

FACULTY OF ENGINEERING

Building a better VHDL testing environment

Joren Guillaume

FEA
Ghent University

Thesis presentation

Outline

Outline

VHDL

VHDL

- VHSIC Hardware Description Language
- Used for describing digital and mixed-signal systems
- Developed by U.S. Department of Defense
  - Document → Simulate → Synthesize

Outline

Testing VHDL

Test benches

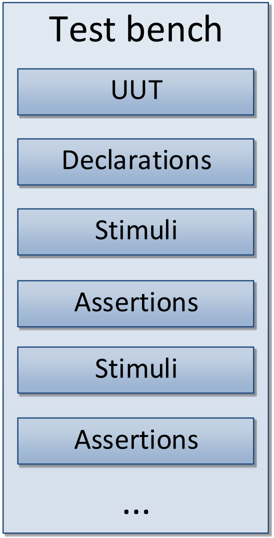
- Unit Under Test (UUT)
- Signal drivers, stimuli & processes
- Assertions and output tracking
  - Comparison to desired result
    - Manual or automated
  - Wave-check

Coverage reports

- Functional coverage
- Code coverage

Problems

- Non-standardized process
- Single point of failure



Testing VHDL - assertions example

D-flipflop

- d: delay, input
- q: output

```
...
assert q = '0'
report "Wrong output value at startup" severity FAILURE;
d <= '1';
      WAIT FOR clk_period;
       assert q = '1'
report "Wrong output value at first test" severity FAILURE;
...
```

Outline

Software development techniques

Unit testing

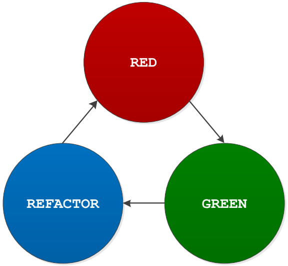
- Unit = smallest behaviour in code
- Test failure → exact location

Test First Development

- Create unit test before the code
- Work from how the code will behave

Test Driven Development

- TFD & short development cycle
- All behaviour is tested
- Proven to significantly reduce errors



Outline

VHDL testing framework

Standardized testing framework

- Based on previously mentioned software techniques
- Cross platform
- At the core: Python script
- Utility library
- Continuous Integration (CI) systemn

Outline

Utility library

Bitvis utility library

- Compatible with all VHDL versions
- Expands VHDL functions
  - Easy value checking
  - String handling & random generation
  - Formatted & automated console output
- Quick & uniform coding

Outline

Continuous Integration

Hudson-CI

- Centralized, automated testing
- Revision control integration (e.g. Git)
- Very customizable (many modules)
- Displays statistics
- Standardized test reports (JUnit)

Outline

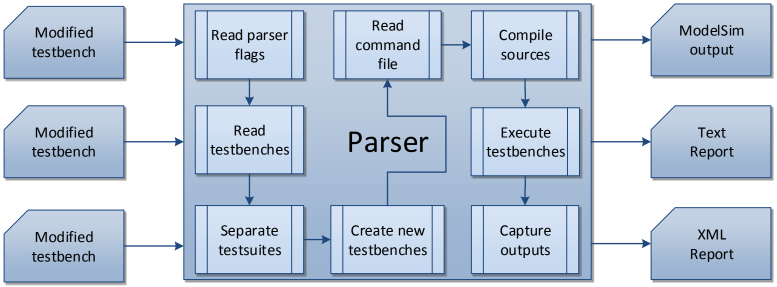
Python script

Features

- Customizable process
  - Command-line arguments
  - Multiple supported methods
- Multiple useful outputs
  - ModelSim simulation output
  - processed text-based report
  - processed XML (JUnit) report
- Automated clean up

Script workflow

Specialized python script



Outline

Preparing the test bench

Preparing the test bench:

1. Import the utility library
2. Decide on separation method
   - Line by line → No editing test bench
   - Start/Stop
   - Partitioned (recommended)
3. Sort tests into groups accordingly
4. Create dependencies file if needed

Modified test bench example

Old test bench:

```
...
assert q = '0'
report "Wrong output value at startup" severity FAILURE;
d <= '1';
      WAIT FOR clk_period;
      assert q = '1'
report "Wrong output value at first test" severity FAILURE;
...
```

Modified test bench:

```
...
--Test 1
check_value(q = '0', FAILURE, "Wrong output value at startup");
write(d, '1', "DFF");
check_value(q = '1', FAILURE, "Wrong output value at first test");
...
--End 1
...
```

Running the job

Running the job:

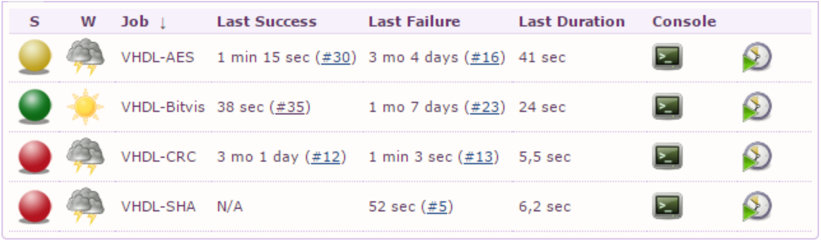
1. Create new job at Hudson-CI
2. Optional: set for import from revision control source
3. Set correct parser flags
4. Set for timed or manual build
5. Build & check results

```
python src\testbench_parser.py -m partitioned -l sim\tb_dff_r.vhd
```

Outline

Results

- Multiple open-source projects converted
- Tested with Git, Hudson-CI & Bitvis



S	W	Job ↓	Last Success	Last Failure	Last Duration	Console	
🟡		VHDL-AES	1 min 15 sec (#30)	3 mo 4 days (#16)	41 sec		
🟢		VHDL-Bitvis	38 sec (#35)	1 mo 7 days (#23)	24 sec		
🔴		VHDL-CRC	3 mo 1 day (#12)	1 min 3 sec (#13)	5,5 sec		
🔴		VHDL-SHA	N/A	52 sec (#5)	6,2 sec		

→ Successful runs where code faultless
→ Partially completed test-runs even with severe errors
→ Unsuccessful runs only due to code faults

Outline

Commentaries on developing VHDL

- An industry stuck in 1993
  - Outdated practices
  - "Don't fix what isn't broken"
- Hardware engineers are not software developers
  - No software development experience
  - No fresh ideas: taught by seniors
- VHDL has no reflection or introspection
  - Could make test benches even more compact

Outline

Future work

- Wider tool support
  - Extensive support for current tool (ModelSim)
  - Greater variety of tools
- Lexical analysis
  - Full code analysis
  - Smart test bench generation
  - Automated partitioning
- Adapted CI tool
  - Specific needs of VHDL development
  - Integration with coverage

Outline

Demo

Demo