


UNIVERSITEIT GENT

FACULTY OF **ENGINEERING**

# Building a better VHDL testing environment

Joren Guillaume

FEA
Ghent University

Thesis presentation

# Outline


1. Situating
   - Testing VHDL
   - Proposed solution
2. The framework
   - Utility library
   - Continuous Integration
   - Python script
   - Using the framework
3. Concluding
   - Results
   - Future work
4. Demo

# Outline


1. Situating
   - Testing VHDL
   - Proposed solution
2. The framework
   - Utility library
   - Continuous Integration
   - Python script
   - Using the framework
3. Concluding
   - Results
   - Future work
4. Demo


UNIVERSITEIT GENT

# Testing VHDL

VHDL

- VHSIC Hardware Description Language
- Develop hardware
- Tested with test benches
  - Output tracking

Problems

- Non-standardized process
- Single point of failure
- Time consuming

# Software development techniques

Unit testing

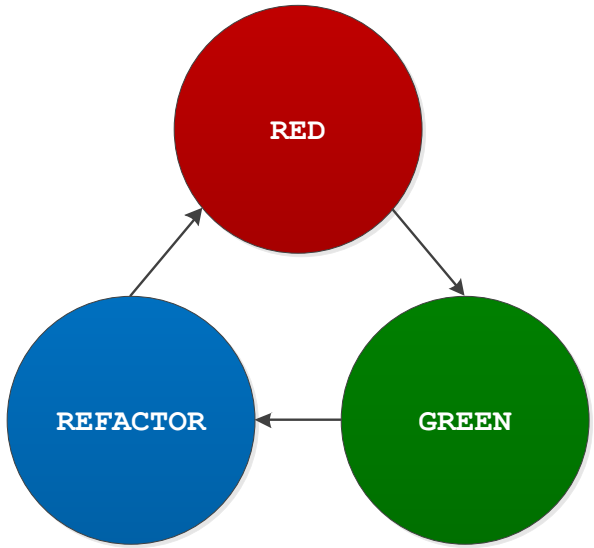
- Unit = smallest behaviour in code
- Test failure → exact location

Test First Development

- Create test before the code
- How the code will behave?

Test Driven Development

- TFD & short development cycle
- Proven to significantly reduce errors



# Outline


1. Situating
   - Testing VHDL
   - Proposed solution
2. The framework
   - Utility library
   - Continuous Integration
   - Python script
   - Using the framework
3. Concluding
   - Results
   - Future work
4. Demo

# VHDL testing framework

Python script

1. Identify and group tests
2. Separate groups into new test benches
3. Compile sources
4. Compile and execute new test benches
5. Capture and process results

# Preparing test benches

- Use Bitvis utility library
  - Faster coding
  - Improved readability
- Separate independent tests
  - Line by line
  - Start/Stop
  - Partitioned
- Create commands file

# Modified test bench example

D flip-flop

- 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
```

# Processing and compilation

Python script:

1. Read command line arguments
2. Read modified test bench
3. Group tests into new test benches

ModelSim:

4. Compile source code
5. Compile test benches

# Execution and results

ModelSim:

1. Execute separated test benches

Python script:

2. Capture ModelSim output
3. Process results
   - Text report
   - XML report

# Automation

Hudson-CI

- Source at revision control system
- Create job per project
- Get latest version from RC
- Timed script execution
- Result progress (XML)

# Outline


1. Situating
   - Testing VHDL
   - Proposed solution
2. The framework
   - Utility library
   - Continuous Integration
   - Python script
   - Using the framework
3. Concluding
   - Results
   - Future work
4. Demo


UNIVERSITEIT GENT

# Results

- Multiple open-source projects converted

| 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 |  |  |

# Future work

- Wider, better tool support
- Lexical analysis
  - Automated partitioning
  - Smart test bench generation
- Adapted CI tool
  - Specific needs of hardware development

# Thanks for your attention

Questions?

# Outline


1. Situating
   - Testing VHDL
   - Proposed solution
2. The framework
   - Utility library
   - Continuous Integration
   - Python script
   - Using the framework
3. Concluding
   - Results
   - Future work
4. Demo

# Demo