Software for embedded systems

Verification Lab.
(Assertion mining)

Samuele Germiniani
samuele.germiniani@univr.it

Alessandro Danese
alessandro.danese@univr.it

January 19, 2020

Simple-platform case study

How to download the simple platform:

```
git clone https://github.com/SamueleGerminiani/simple_platform.git
```

Environment set-up:

1. cd simple_platform
2. source env_setup.sh

Makefile menu

How to open the Makefile menu (terminal):

1. cd questa.simulation
2. make

Makefile menu

Once started, the following menu should appear on the terminal

```
================================================================================

 USAGE: make RECEPIE|TARGET
 Authors: Alessandro Danese (alessandro.danese@univr.it)
          Samuele Germiniani (samuele.germiniani@univr.it)

 --- RECIPES ------------------------------------------------------------------
 simulation              => Performs: clean, compile_s, and run_s
 mining_bl_master        => Performs: clean, and assertion mining for buslayer (master)
 mining_bl_slave_0       => Performs: clean, and assertion mining for buslayer (slave_0)
 mining_bl_slave_1       => Performs: clean, and assertion mining for buslayer (slave_1)
 mining_camellia         => Performs: clean, and assertion mining for camellia
 mining_transmitter      => Performs: clean, and assertion mining for transamitter
 ABV                     => Performs: clean, and Assertion-based Verification
 faultC_bl_master        => Performs: clean, and fault coverage for buslayer (master)
 faultC_bl_slave         => Performs: clean, and fault coverage for buslayer (slave)
 faultC_camellia         => Performs: clean, and fault coverage for camellia
 faultC_transmitter      => Performs: clean, and fault coverage for transmitter

 --- TARGETS ------------------------------------------------------------------
 check_faults            => Performs: fault-coverage with faults.txt file
 compile_s               => Compilings DUT
 run_s                   => Runs simulation.

 --- ADMINISTRATIVE TARGETS ---------------------------------------------------
 help                    => Displays this message.
 clean                   => Removes all intermediate and log files.

================================================================================
```

Mining assertions for the simple platform

In "simple_platform/mining/a-team/tests/", there are 3 mining configuration files ready to be used:

- bl_master/mining_bl_master.xml
- camellia/mining_bl_master.xml
- serial_transmitter/mining_serial_transmitter.xml

Together with the configuration file there are three additional files:

- trace.variables : it contains the list of variables in the submodule
- trace.mangrove : it contains the values for each variable/signal in the submodule
- $< name\_of\_submodule > .vcd$: it contains the vcd of the submodule for the current simulation

Mining assertions for the simple platform

To mine assertions for one of the three submodules, type
make *mining_* < *name_of _submodule* >
The command will execute the following operations:

1. Compile the simple-platform
2. Simulate the simple-platform and generate the vcd file
3. Move the vcd file in the correct submodule directory
4. Execute the mining for the given submodule

Don't forget to complete the configuration file (.xml file) before executing the mining!

Exercise 1

In the previous lesson we wrote assertions for the master_bus_layer submodule of the simple platforms. These are some of those assertions:

- $(!busy \ \&\& \ request) \ |-> \ nexttime[1](busy);$
- $(!busy \ \&\& \ request) \ |-> \ nexttime[1](wb\_we \ == \ \$past(write, 1));$

Complete the bl_master configuration file with propositions and templates to enable the miner of automatically mine the above assertions.
Warning: the current implementation of the assertion miner does not support the $past operator as a template but only as an manually defined proposition!

Exercise 2

Read the camellia specifications and complete the camellia configuration file with propositions and templates to enable the assertion mine to extract the assertions. If you gave the correct templates and propositions, the mined assertions should be similar to what you would manually write by reading the specifications.

Exercise 3

The serial_transmitter does not have a specification file!
Try to briefly write its specifications, use the other specification files as a guideline.
You can easily do it by inspecting the code and by mining assertions.