系統晶片驗證 (SoC Verification)

109 學年下學期 電機系電子所選修課程 943 U0250

Final Project [Formal Verification Research and Development]

(Due: 9:00pm, Thursday, July 01, 2021)

[Note] 請已決定題目以及分組完成之組別，至此 Google Form 填寫分組資訊。

0. Objectives

1. Learn how to conduct advanced formal verification research or development.

2. You are expected to conduct reference work survey, design algorithms, implement your final project (i.e. write codes) and demonstrate results with reasonable experiments.

1. Project Description

As last year, there are three options for this semester's final project:

1.1. PDR Implementation

Improve PDR from a given very simple (and maybe problematic) implementation. This is a one-person project. More project description and reference code can be found under the "final_PDR" directory.

1.2. CAD Contest

Participate in the 2021 CAD Contest (http://iccad-contest.org/2021/tw/). Here are the rules for choosing CAD Contest as your final project:

1. You can choose from Problems A ~ C, although Problem A is recommended. However, in your final report and implementation, you need to demonstrate how formal verification techniques are used in your algorithm.
2. You can have as many team members as it is allowed by the contest rules, and it's OK to have team members outside this class, or from this class but will not claim it for the final project. Either way, in your final report, you need to state clearly the responsibilities and contributions of each team member.
3. You need to register for the contest. Please note that the deadline for registration is 05/10 (Mon). Therefore, even if you haven't decided whether to take the CAD contest as your final project, register it first anyway!

1.3. Self-Defined Problem

You are also encouraged to define your own problem for the final project. Here are the rules to define your own problem as your final project:

1. Any field of application is fine. However, it should be a real and practical problem, and utilization of formal verification techniques is a MUST.
2. Implementation (i.e. write codes) and experimental results are expected. You cannot just do paper survey.
3. Any number of team members is fine, although less team members may imply bigger challenges and thus may result in slightly higher score. You should state clearly the responsibilities and contributions of each team member in your final report.
4. You need to project description in the Google form (at least 300 words). You are welcome to discuss with the instructor in advance for any potential ideas.

2. General Rules

Here are some general rules that apply to all types of projects.

1. Make sure your code can be compiled and executed in Linux environment.

2. Utilization of other's codes/libraries is allowed. However, you should state it clearly in your report. For those not declared, we will assume they are written by you. Do not copy and paste other's codes from either classmates, previous students, or any other sources. Any act of plagiarism will lead to severe deduction of your point and we will report this to the departmental office.

3. You cannot use this same project for other class(es), unless it is an improvement from some previous work. And in that case, you should state it clearly and point out what the improvement is. I will check with other professors for this.

4. (For CAD Contest and Self-Defined Project) However, it is OK to have teammates from other classes, or in this class but do not plan to claim it for final project. Again, just state clearly your contribution in this project for this class.

3. Project Submission

Rename the directory to "<GroupID>_final" before compressing it. Name the report file "<GroupID>_report.pdf" in PDF format and save it in the root directory of this final project (i.e. "<GroupID>_final"). Besides, you should compress it under Linux workstation by the command:

```
tar zcvf <GroupID>_final.tgz <GroupID>_final
```

Submission deadline is 9pm, Thursday, 07/01, 2021. Late penalty is high (deducing 1/3 of your points and above). Please be in time.

There will be NO in-class presentation!

3.1 PDR Implementation

3.1.1 Submission Check List

☐ README: Describe (i) What you submit. (2) How to compile your project. (3) What the testcases are and how to test them.

☐ Written Report

- Must be in English.
- Two-column 6-page limit. Please follow the IEEE format.
- Please specify your contact information in the author list. I may need to contact you in case there is any problem about your project source code or report.
- The content should include: (i) Abstract: briefly describe your improvements and results, (ii) One improvement idea in one section, (iii) Verification results of the vending machine design, (iv) Experimental results of the HWMCC benchmark, comparing to the built-in V3 PDR engine (and engines from other homework), (v) Disclaimer: on the use of 3rd party's codes and improvement from your previous work, and (vi) Optionally, provide your feedbacks and suggestions to this class.

☐ Source code

- Make sure it can be compiled by a single "make" command in the project root directory.
- Remove the unnecessary files (e.g. core dumps, *.o) by "make clean" or so, before compressing and submitting it.

☐ Testcases: All the testcases you have tested and dofiles/scripts to test them.

3.1.2 Grading Criteria

○ Technical Depth/Algorithm: 20%

Don't need to repeat the PDR algorithm that was taught in class. Just describe how you improve the baseline solver and how effective they are.

○ Program Performance: 60%

Tested by the provided and hidden cases on a Linux workstation.

- Counter-example trace implementation: 5%

  Pay attention to the format as specified in the problem description file.

- Written Report: 15%

3.2 CAD Contest

3.2.1 Submission Check List --- One submission per group

- ☐ README: Describe (i) What you submit. (2) How to compile your project. (3) What the testcases are and how to test them.

- ☐ Written Report
  - Must be in English.
  - Two-column 6-page limit. Please follow the IEEE format.
  - Please specify the contact information of all the team members in the author list. I may need to contact you in case there is any problem about your project source code or report.
  - The content should include: (i) Abstract: briefly describe your algorithms, achievements and results, (ii) Algorithms: pseudo codes and/or flow charts are expected, (iii) Experimental results: comparison with others is a plus, (iv) Teamwork: how the tasks are divided among team members, and whether there is any team member that is NOT in this class, (v) Disclaimer: on the use of 3rd party's codes and improvement from your previous work, and (vi) Optionally, provide your feedbacks and suggestions to this class.
  - NOTE: For the fairness of the competition, the report will only be accessible to me (Ric). If you are not comfortable in submitting thru Ceiba, please feel free to submit it to me directly (thru e-mail or messenger).

- ☐ Source code
  - Make sure it can be compiled by a single "make" command in the project root directory.
  - Remove the unnecessary files (e.g. core dumps, *.o) by "make clean" or so, before compressing and submitting it.

- ☐ Testcases: All the testcases you have tested and dofiles/scripts to test them.

3.2.2 Grading Criteria

- Technical Depth/Algorithm: 50%

  Don't need to repeat the problem definition/description as in the contest website. I am very familiar with this problem. However, please clearly describe how you

analyze this problem and present your algorithms with pseudo codes or flowcharts. Please note that I will read your report for the grading purpose only and will NOT share it with anyone else.

- Program Performance: 35%

  Tested by the public cases from the contest website, and maybe some other resources from the instructor.

- Written Report: 15%

3.3 Self-Defined Problem

3.3.1 Submission Check List --- One submission per group

☐ README: Describe (i) What you submit. (2) How to compile your project. (3) What the testcases are and how to test them.

☐ Written Report

- Must be in English.
- Two-column 6-page limit. Please follow the IEEE format.
- Please specify the contact information of all the team members in the author list. I may need to contact you in case there is any problem about your project source code or report.
- The content should include: (i) Abstract: briefly describe the problem, achievements and results, (ii) Algorithms: pseudo codes and/or flow charts are expected, (iii) Experimental results: comparison with others is a plus, (iv) Teamwork: how the tasks are divided among team members, and whether there is any team member that is NOT in this class, (v) Disclaimer: on the use of 3rd party's codes and improvement from your previous work, and (vi) Optionally, provide your feedbacks and suggestions to this class.
- NOTE: For the fairness of the competition, the report will only be accessible to me (Ric). If you are not comfortable in submitting thru Ceiba, please feel free to submit it to me directly (thru e-mail or messenger).

☐ Source code

- Make sure it can be compiled by a single "make" command in the project root directory.
- Remove the unnecessary files (e.g. core dumps, *.o) by "make clean" or so, before compressing and submitting it.

☐ Testcases: All the testcases you have tested and dofiles/scripts to test them.

3.2.2 Grading Criteria

- Technical Depth/Algorithm: 50%

  Don't need to repeat the problem definition/description as in the contest website. I am very familiar with this problem. However, please clearly describe how you analyze this problem and present your algorithms with pseudo codes or flowcharts. Please note that I will read your report for the grading purpose only and will NOT share it with anyone else.

- Program Performance: 35%

  Tested by the public cases from the contest website, and maybe some other resources from the instructor.

- Written Report: 15%

4. What you should turn in

All the files in this project (including written report) should be turned in by 9pm, Thursday, 07/01, 2021. However, please make sure it can be compiled by a single "make" command in the project root directory. Please remove the unnecessary files (e.g. core dumps, *.o) by "make clean" or so, before compressing and submitting it. Otherwise, it may be too big to submit on ceiba. Remember to rename the directory to "<GroupID>_final" before compressing it (that is, you can keep the same directory structure as original). Besides, you should compress it under Linux workstation by the command:

```
tar zcvf <GroupID>_final.tgz <GroupID>_final
```