# Logic CAD of VLSI Design - 046880

## **Topic**: SAT - based Gate Level Formal Equivalence Checker

**Introduction**

Formal Equivalence Verification (FEV) provides a key technology that enables comparing the RTL to the synthesized gate-level or multiple different gate-level implementations to each other. In recent years, BDDs which were the cornerstone of formal models and FEV was replaced by SAT solvers to resolve the capacity limitations of BDDs.

**In this assignment**

1. You will learn how to use and integrate your code with Minisat which is an award-winning SAT solver (using API)
2. You will learn how to generate a .cnf file for a specific circuit.
3. You will implement a "Formal Equivalence Verifier (FEV)" which compares two gate-level Verilog netlists for the design.

**Detailed Tasks,**

1. Implement a "Formal Equivalence Verifier" This program:
   a. Reads two sets of Verilog files, analyzes them to obtain the Primary Inputs and Outputs (PIs and POs) of the design.
   b. Match the PIs and POs between the two circuits (by name).
   c. Formulate a CNF for a system comparing each pair of outputs and call Minisat to prove the matching.
   d. Generates a **.cnf** file (named spec-cell.cnf) which represents the circuit that is used to compare the two Verilog files. The file will allows running the MiniSat solver on it and get UNSAT if the two circuits are equivalent and SAT and assignment, if not.

The program should use the MiniSat API for the comparison. At the end the program should print if the circuits are equivalence, and if not – which **input assignment** will show it.

**Detailed Requirements**:

a. Correctness
b. When mismatches are found the code should clearly state the input vector that causes the mismatch (one example for such input, even if there are more).
c. Keep your code efficient.
d. Keep the code well documented and with good readability.

**[Dry]** Provide documentation for your code and answer the next questions in your report:

a. What is the basic code structure/high-level algorithm?
b. Provide an exact explanation of the comparison algorithm.
c. How are constant being handled?
d. How are XOR gates being supported? Show the detailed Tseytin development if you use it.

## Stage A – Build and write your own programs with HCM

1. **cd HCM**
2. **make** – not needed if you followed the workshop
3. **cd wet03**
4. Write your code in HW3ex1.cc file.
5. **make**
   NOTE: the tests for this exercise are automatic!
   If your submission will not compile on the virtual machine – it will be grated ZERO!

## Stage B – Test your own programs with HCM

After finishing Stage A, make sure you run the following command from wet03 directory.

Now you are ready to generate the required output files for self-testing before submitting your work.

To generate TopLevel1355.cnf and TopLevel0409.cnf files run the following lines in the Virtual machine, in "wet03" directory.

- **./gl_verilog_fev -s TopLevel1355 stdcell.v c1355.v -i TopLevel1356 stdcell.v c1356.v**
- **./gl_verilog_fev -s TopLevel0409 stdcell.v c0409.v -i TopLevel0410 stdcell.v c0410.v**

You should test your code with **SAT and UNSAT comparison cases** based on the following benchmark circuits: **c1355.v** and **c0409.v** and get **UNSAT** and **SAT** respectively.

## Stage C – Create your submission

After finishing Stage B, make sure your wet03 directory include only the .cc files and your final report.
Run the following command from wet03/ directory -

1. **cd ..**
2. **tar czf wet03_<id1>_<id2>.tar.gz wet03**
   (<id1> and <id2> are the id numbers of the students)
3. Upload wet03_<id1>_<id2>.tar.gz to Moodle
   (Example of the file name: wet03_123456789_147852369.tar.gz)

- Make sure that only the .cc your report files are in the folder !

**Note: Bold lines are execution commands for the Linux environment (LUX).**
**Note**: The following will be taken into consideration when grading your assignment:

a. The correct modeling of the behavior of the circuit.
b. Clear answers to all dry part questions.
c. The quality of the code and the documentation.

| | |
|---|---|
| Topic | SAT- based Gate Level Formal Equivalence Checker |
| Submission date | |
| Exercise owner | Yoac Cohen - yoavneta1@technion.ac.il |
| Given Code files | HW3_ex1.cc<br>Makefile |
| Given Input Verilog files | c1355.v<br>c1356.v<br>c0409.v<br>c0410.v |
| Given Extra Verilog files | stdcell.v |
| Given Output files for self-testing | - |
| Files to submit | **HW3_ex1.cc**<br>**HW3.pdf** |
| Submission format | **wet03_<id1>_<id2>.tar.gz** |
| NOTE: the tests for this exercise are automatic!<br>wrong submission format will be grated ZERO! | |

Questions about this WET exercise should be posted on Moodle. You required to follow the answers of the course staff and write your code according to the answers.

**Good luck!**