Homework Assignment 4
Total Points: 50

Professor Karem A. Sakallah

EECS 598-002: Formal Verification of Hardware & Software Systems

Assigned: February 13, 2024
Due: February 20, 2024

Guidelines

- The College of Engineering Honor Code applies to all work in this course.
- The due date is firm. Follow submission instructions (at the end).

Objectives

This assignment explores the application of SAT solvers (using the PySAT Python wrapper) to verify functional equivalence between two combinational circuits or two CNF formulas.

1 [Equivalence Checking of Combinational Circuits] (30 Points)

Figure 1 defines the inputs and outputs of two circuits that generate the carry signals in two 4-bit adder designs, a ripple-carry adder (RCA) and a carry-lookahead adder (CLA).

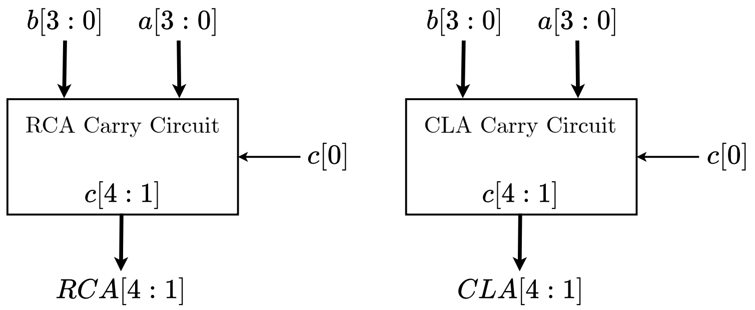

Figure 1: Inputs and outputs of the carry logic in 4-bit ripple-carry and carry-lookahead adders

The equations that specify how the carry signals are computed in each of these designs are shown in Figure 2. To confirm that these two sets of equations produce identical carry signals:

- Use the Tseitin transformation to create a CNF file, named **h4p1.cnf**, that encodes these equations and the required comparison logic. The CNF formula should be UNSAT if the two designs are equivalent.

- Create a Python file, named **h4p1.py**, to read this CNF file and solve it using any of the available solvers. Instrument your Python file to produce descriptive output that documents your verification run. This should include the name of the solver used, and relevant statistics such as run time, number of decisions and conflicts. Save this output in a file named **h4p1.txt.**

RCA Equations

$$RCA[i+1] = (a[i] \wedge b[i]) \vee (c[i] \wedge (a[i] \oplus b[i])) \text{ for } i \in [0,3]$$

CLA Equations

Bit propagate and generate equations:

$$p[i] = (a[i] \vee b[i]) \quad \text{for } i \in [0,3]$$
$$g[i] = (a[i] \wedge b[i]) \quad \text{for } i \in [0,3]$$

Group propagate and generate equations:

$$P[1,0] = p[1] \wedge p[0]$$
$$P[3,2] = p[3] \wedge p[2]$$
$$P[3,0] = P[3,2] \wedge P[1,0]$$
$$G[1,0] = g[1] \vee (p[1] \wedge g[0])$$
$$G[3,2] = g[3] \vee (p[3] \wedge g[2])$$
$$G[3,0] = G[3,2] \vee (P[3,2] \wedge G[1,0])$$

Carry equations:

$$CLA[1] = g[0] \vee (p[0] \wedge c[0])$$
$$CLA[2] = G[1,0] \vee (P[1,0] \wedge c[0])$$
$$CLA[3] = g[2] \vee (p[2] \wedge CLA[2])$$
$$CLA[4] = G[3,0] \vee (P[3,0] \wedge c[0])$$

Figure 2: RCA and CLA Carry Equations

2 [Equivalence Checking of CNF Formulas] (20 Points)

The files **h4p2f.cnf** and **h4p2g.cnf** are the CNF encoding of two Boolean functions f and g. Create a Python file, **h4p2.py** that checks if these two CNF formulas are equivalent. Save the output of your run in **h4p2.txt**.
Hint: Use PySAT's negate method.

Submission Instructions

1. Create a directory named <your uniqname>_hw4

2. Place in the directory the following Python files corresponding to each of the problems or problem parts:

   - Problem 1: h4p1.py, h4p1.txt
   - Problem 2: h4p2.py, h4p2.txt

3. Write a short description explaining any issues you encountered and suggestions for how such verification tasks can be automated further and save it in **h4.pdf**.

4. Zip the entire directory using "zip -r <your uniquename>_hw4.zip <your uniqname>_hw4" and upload to Canvas.