

DAILY ASSESSMENT REPORT

Date:	05-August-2020	Name:	Swastik R Gowda
Course:	VLSI CAD PART 1	USN:	4AL17EC091
Topic:	Week 3	Semester & Section:	6 th Sem 'B' Sec
Github Repository:	swastik-gowda		

FORENOON SESSION DETAILS

Image of session

Famous: Reduce-Expand-Irredundant Loop

- Famous tool **ESPRESSO** for 2-level minimization
 - Started at IBM, finished at Berkeley
 - Brayton, Hachtel, McMullen, Sangiovanni-Vincentelli, **Logic Minimization Algorithms for VLSI Synthesis**, Kluwer Academic Press, 1984, is the reference here
 - Richard L. Rudell, (1986-06-05), "Multiple-Valued Logic Minimization for PLA Synthesis", Memo No. UCB/ERL M86-65 (U. California Berkeley M.S. Thesis)
 - Also, Giovanni DeMicheli, **Synthesis and Optimization of Digital Circuits**, McGraw Hill, 1994

Slide 14. © 2013, R.A. Rutenbar

Summary

- 2-level logic synthesis uses **heuristics** to find **good** solutions
 - Not "best", but instead "good enough"
 - Minimal (not minimum), prime, irredundant
 - Famous idea: iterative improvement – **reduce-expand-irredundant loop**
 - All done with PCN cubelists, covering matrices, and URP ideas
- But... not every piece of logic is implemented in 2-level form
- Next: **Multi-level logic**

Slide 26. © 2013, R.A. Rutenbar

Report – Report can be typed or hand written for up to two pages.

ESPRESSO1 is a 2-level logic optimization tool developed by researchers at the University of California at Berkeley. Early versions of the idea were developed at the T.J. Watson Research Labs; the final version of the tool was developed at Berkeley.

Rick Rudell's Master's Thesis2 from 1986 is a very complete explanation of the development and software implementation of the tool. ESPRESSO is a hugely successful optimizer: it is the tool that pioneered the reduce-expand-irredundant heuristic, and the ideas are built into most modern logic synthesis systems.

ESPRESSO has many options: you can instruct ESPRESSO "how hard" to work to optimize your logic. However, we will run ESPRESSO in its standard, default mode, which will suffice for our teaching purposes. As we described in lecture, you can start a 2-level optimization problem with a partially specified truth table (TT) with input don't cares.

Let's look at a small example of an ESPRESSO input file, shown below. (Note: we add the //comments for clarity; but you cannot put these comments in the actual ESPRESSO input file! Don't add these comments! ESPRESSO will not like this!).

.i 4 // There are 4 inputs

.o 1 // There is 1 output

// ESPRESSO can optimize several functions at the same time

// These are .ill w x y z e the names of the inputs

.ob f // This is the name of the output

0-11 1 // One line of the TT. Order does not matter. "-" = input don't care

01-1 1 // More TT

1011 1 // More TT

1111 - // More TT. We explicitly tell ESPRESSO that f is don't care here.