

WEEK #1: January 19

Introduction.

Course overview, modern computer-aided digital design, Boolean representations.

Exact 2-Level Logic Minimization.

The Quine-McCluskey method (basics).

Heuristic 2-Level Logic Minimization.

Introduction to the “espresso” algorithm. Basic Boolean representation and terminology.

WEEK #2: January 26

Heuristic 2-Level Logic Minimization.

Multiple-output functions. Local search, iterative improvement algorithms.

Overview of key espresso steps: *expand, irredundant, reduce.*

Details of expand: cube order, expansion direction.

WEEK #3: February 2

Heuristic 2-Level Logic Minimization.

Details of irredundant and reduce. The Unate Recursive Paradigm.

Cofactoring. Fast tautology checking (introduction): basic termination rules.

WEEK #4: February 9

Heuristic 2-Level Logic Minimization.

Shannon decomposition: recursive divide-and-conquer.

Advanced techniques: exploiting properties of unate functions and covers.

Fast tautology checking (conclusion): final flow, advanced termination rules.

The containment problem. Fast complementation method.

WEEK #5: February 16

Heuristic 2-Level Logic Minimization.

Generating all essentials without generating all primes.

Final iterative espresso loop.

Advanced optimizations: make-sparse; avoiding local minima with last-gasp/super-gasp.

Multi-Level Optimization: Algebraic Techniques.

Motivation. Circuit modelling: logic network graphs. Approximate cost models: area, delay.

Overview of logic transforms: extraction, substitution, collapse, simplify, decomposition.

WEEK #6: February 23

Multi-Level Optimization: Algebraic Techniques.

Introduction to the UC Berkeley “SIS” CAD tool environment. Designer scripts.

Algebraic vs. Boolean models. Network collapse with fast eliminate.

Algebraic division, single- and multi-cube extraction.

Foundations: kernels, co-kernels. Brayton/McMullen’s Fundamental Theorem.

WEEK #7: March 1

Multi-Level Optimization: Algebraic Techniques.

Rudell’s rectangle covering formulation. Substitution, decomposition.

Technology Mapping: Basics.

VLSI cell libraries, cell characterization. Exact vs. heuristic solutions.

Overview of heuristic tech map: decomposition, partitioning, matching/covering.

Subject and pattern graphs.

WEEK #8: March 8

Technology Mapping: Basics.

Tree-based matching and covering algorithms. Introduction to dynamic programming.
Targeting different cost functions: area, delay, power.
Inverter-pair heuristic. Simple delay-oriented mapping (load-independent).

WEEK OF MARCH 12–MARCH 16: SPRING BREAK.**WEEK #9: March 22**

Technology Mapping: Advanced.

Delay-oriented mapping (load-dependent): load binning, exploiting drive strengths.
Power-oriented mapping: using stochastic models, area/delay tradeoffs.
Recent techniques for LUT-based FPGA's.

Physical Design Basics: Partitioning and Placement/Routing.

Partitioning of large-scale circuits: Kernighan-Lin divide-and-conquer method.
Place-and-route: problem formulation, introduction to simulated annealing techniques.

WEEK #10: March 29

System-Level Optimization: Retiming.

Optimizing area and clock cycle time by repositioning registers.
Graph-based models. Leiserson/Saxe's method.

WEEK #11: April 5

Architectural Synthesis: Register-Transfer Level Design.

Specifying systems using control-dataflow graphs (CDFG's).
Overview of scheduling: resource-constrained, latency-constrained. Chaining techniques.
Architectural Synthesis: CAD Optimization.
Exploring system-level cost tradeoffs: latency, area and power. Detailed case study.
Optimal scheduling algorithms: resource-constrained, time-constrained, force-directed.

WEEK #12: April 12

Architectural Synthesis: CAD Optimization.

Optimal resource sharing: registers, function units, buses.
Binary Decision Diagrams (BDDs).
Introduction to modern compact Boolean data structures.
Canonicity, basic variable ordering, properties. Reduced ordered BDD's (ROBDD's).

WEEK #13: April 19

Binary Decision Diagrams (BDDs).

The REDUCE and APPLY operations.
Cofactor and complementation operations. Multi-output BDD's.
Applications: checking circuit equivalence, multi-level logic synthesis.
Advanced techniques: dynamic variable ordering using "sifting" heuristics.

WEEK #14: April 26

Satisfiability (SAT) Solvers.

Constraint satisfaction problems.
Search techniques: satisfying assignments, conflict-driven learning, backtracking.
Applications: routing, circuit equivalence.
Multi-Level Optimization: Advanced Boolean Techniques.
Boolean simplification. Exploiting local don't-cares (CDC's, ODC's).

FINAL EXAM: Wednesday, May 2, 4:10-7:00pm (location TBA)