

**California Institute of Technology
Department of Computer Science
Electronic Design Automation**

CS286.5b, Winter 2000

Assignment #2

Tuesday, February 8

Due: Friday, March 3, 11:59pm.

Problem

Devise and implement an algorithm for recursive partition that minimizes energy consumption. Benchmark your implementation on representative circuits and compare to a conventional (io-minimizing) alternative (provided).

Assumptions and Details

- Use HSRAppr infrastructure for netlist reading and handling.
- Energy target for optimizing is net switching on wires.
- Energy is modeled with an energy cost for each transition on each partition level which a net crosses.

Collaboration

Each student is expected to do his/her own work. For this project, you are free (and encouraged) to discuss basic strategies and approaches with your fellow classmates or others, but implementations, analysis, and writeups should always be the work of the individual. If you get advice or insights from others that significantly influenced your work, please acknowledge this in your writeups.

Writeup

Writeup should be in an electronically readable format (HTML or PDF preferred) and should include:

1. Basic problem formulation (including optimization criteria and cost model)
2. Outline of solution, including algorithm description

3. Instructions on how to run your optimizer (arguments, options, etc.)
4. Results and comparison
5. Lessons and recommendations

There is no page limit (one way or the other), but I'd guess typical writeups might be 7-10 equivalent pages. Just focus on getting the key ideas across.

Turnin

1. Pointer to your implementation (source and executable)
2. Pointer to your writeup

Additional Information

Base for support code is `/ufs/students5/andre/EDA/`. You'll want to make your own copy of the `copy_this` directory as a starting point. It includes:

- `ENVIRONMENT` which you will need to modify appropriately (and you may want to adapt it to your shell)
- `my_powerpart.java` to use as a base for optimization
- `Makefile` for running java compiler plus an example invocation of the existing `my_powerpart` once built.

You may want to grab other classes from `HSRAppr` to modify according to your needs.

Examples live in `iwls93/flowmap_4lut`. I have generated switching probabilities for almost everything in `asub1k` and those live in `asub1k.activity`. All of the directories starting with "a" are subsets of `area_mapped`. When I'm developing, I start working through the small examples (smaller sets) and work up. The activities in `asub1k.activity` are thus a superset of the designs in `atiny` and `asmall`.

Technology files live in `tech` and define the energy cost of each level. Primarily, there is a flat cost `uniform.tech` and an exponential cost `exp.tech`.

I'll set you up with accounts on `mycroft` and, at a minimum, you can run java there. The code as is should certainly run on a modern linux machine. It should run (maybe with small tweaks) on Solaris or Windows. I was unable to make it run on the VLSI lab NetBSD machines this weekend. The primary portability issue (aside from the existence of a JVM and java compiler) is that the spectral ordering code uses native libraries.