Suggestions and Resources for Current and Prospective Graduate Students

Zhiyang Ong

June 18, 2017

Abstract

This is a list of resources for current and prospective graduate students.

Contents

1 Preliminary Ph.D. Thesis Proposal

The full Ph.D. dissertation proposal is approximately 20-50 pages with appropriate tradeoffs between what others have done, what the student has done, and what the student will do.

Things to be included in my preliminary Ph.D. thesis proposal (preliminary research proposal):

1. This may also be known as the "PhD plan"
2. This project proposal shall be about 5-10 pages
3. This can or should be completed in cooperation with my main supervisor(s)
4. Abstract of the research proposal: include field for keywords to indicate the scope of the project
5. Project description (0.5-1 page):
   (a) **Introduction:**
      i. provide a general introduction to what the thesis is all about
      ii. it is not just a description of the contents of each section
      iii. briefly summarize the research problem/question (I shall be stating the research problem/question in detail later)
      iv. also, provide a summary for some of the reasons why it is a worthwhile question
      v. `For the Ph.D. thesis ONLY: give an overview of my main results`
      vi. this is a bird's-eye view of the answers to the main questions answered in the thesis
      vii. set up the basis of the research

   (b) **Background Information (optional):**
      i. a brief section giving background information may be necessary, especially if your work spans two or more traditional fields
      ii. that means that your readers may not have any experience with some of the material needed to follow your thesis, so you need to give it to them
      iii. a different title than that given above is usually better; e.g., "A Brief Review of Frammis Algebra."

   (c) **Research Question or Problem Statement:**
      i. Problem formulation.
      ii. The problem statement should be comprehensible by faculty in the department who are not in the particular specialty of the student and the advisor.
      iii. Preliminary considerations:
         A. What is the considered scope, or focus of my Ph.D. research?
         B. What's my vision? And, what's my research topic?
         C. Which issues will I be considering?
         D. Is my proposed research topic well-defined and specific?
         E. Is it novel?
         F. If this research topic is competently investigated, is it significant enough to be worthy of a Ph.D.?
      iv. What problem(s) am I solving?:
         A. What is the problem?
         B. provide a clear explanation of the problem(s) to be worked on
         C. Succinctly state the research problem and its significance:
            - use a concise statement of the problem that my research will attempt to solve
            - omit vague terms in my problem statement... for example, do not say "develop a Zylon algorithm capable of handling very large scale problems in reasonable time"; specify "large scale" and "reasonable time" by describing what I mean in numerical figures, or numerical scale (e.g. ultra-deep submicron technologies, nanoscale architectures, exabyte data storage, or THz imaging)
            - definition of the problem space

- list keyword to define your topic
- state your research topic as a problem/question
- think about the significant terms, concepts, and keywords that describe your topic.
- these terms will become the (search) keys while searching for information about your subject in library catalogs, online databases, and other resources.

v. **What is so interesting about this/(these) problem(s)?**
   A. Why is the problem important?
   B. Have I identified a worthwhile problem/question that has not been solved/answered?
   C. Is this open problem indicated in a number of the papers that I have read?
   D. for open problems (unresolved issues), provide a justification, by direct reference to a literature review, to indicate that my research problem is unsolved
   E. else, indicate that the problem has not been solved adequately, or its solutions may not be as effective in the near future as technologies improve
   F. alternatively, say... given existing architectural scaling trends (or process technology scaling trends), this problem would remerge since current solutions do not scale well (with respect to architectural scaling trends, or process technology scaling trends). Hence, these solutions may not be able to adequately handle BLAH ISSUES with advanced architectures (or process technologies). In this case, more extensive justification for the selection of this research problem can be given later in the literature review section
   G. analyze the information read during my literature review, and indicate how each class/category of current approaches fails; i.e., current approaches can handle only small problems, and/or takes too much time

vi. Discussion of why it is worthwhile to solve this problem:
   A. What are my motivations for solving this problem?
   B. How would my work extend the cutting edge in this field?
   C. Why is it useful to have a solution to this problem?
   D. Who would care if I have a solution to this problem? R&D teams in high-tech companies??? University research labs???
   E. What are the wider implications of my research?

vii. To highlight this section (i.e., Problem Statement), try to include the term “`Problem Statement`” in a fairly specific section title. For example, “The Large-Scale Zylon Algorithm Problem.”

(d) **Review of the State of the Art:**
   i. **Survey of the state of the art (2-5 pages approx.); 5-10 pages in a typical thesis proposal, but is longer in the “prior work” section of a Ph.D. dissertation**
   ii. this section shall be longer if there exists rich literature for my research topic
   iii. make ample references to existing literature in the field
   iv. this literature review section can serve as the nucleus for such a chapter in my Ph.D. dissertation; since the chapter in Ph.D. dissertation is longer, it pays to do a extensive literature review early on for my research proposal and keep up to date with advancements in the field
   v. What has been done so far on the problem?
   vi. How does it relate to previous work in the field?
   vii. Include a brief introduction to my topic, and introduce key concepts
   viii. What is the cutting edge in my field? State-of-the-art solution(s) of the problem.
   ix. *That is, review the state of the art relevant to my thesis ...* `What has been done so far on the problem?`
   x. Again, a different title is probably appropriate; e.g., “State of the Art in Zylon Algorithms.”

xi. Determine a collection of seminal papers (fundamental references) that can be considered real milestones in the field of interest

xii. What are the seminal papers in this field, and for my research topic? Have I read, reviewed, and referenced them?

xiii. For my research topic, are there seminar classes offered by any university that provides a list of papers that I should read?

xiv. What other issues are involved in this topic?

xv. Present (/perform critical analysis of) the major ideas in the state of the art right up to, but not including, my own personal brilliant ideas

xvi. Organize this section by idea(s), and not by author or by publication.

xvii. For example, if there have been three important main approaches to Zylon Algorithms to date, you might organize subsections around these three approaches, if necessary:
   A. 3.1 Iterative Approximation of Zylons
   B. 3.2 Statistical Weighting of Zylons
   C. 3.3 Graph-Theoretic Approaches to Zylon Manipulation

xviii. Determine if related research might be published under different keywords

xix. The lit review can be difficult, and I may feel lost and don't know where I'm going. This is because I'm trying to do several things concurrently; I'm trying to learn about this whole field, and get a conceptual framework of how to map out this area of research. So, Fight On!!!

(e) **A very clear statement of the question is essential to proving that you have made an original and worthwhile contribution to knowledge. To prove the originality and value of your contribution, you must present a thorough review of the existing literature on the subject, and on closely related subjects. Then, by making direct reference to your literature review, you must demonstrate that your question (a) has not been previously answered, and (b) is worth answering. Describing how you answered the question is usually easier to write about, since you have been intimately involved in the details over the course of your graduate work.**

(f) **How will I be addressing these issues?:**

i. Suggested/proposed approaches, including initial approach

ii. What are my goals/objectives [or aims] of this research project?

iii. **Proposal for innovative research contribution for the thesis work with a clear and detailed description of the research directions and objectives (2-5 pages approx.)**

iv. What would be my contribution(s) of the thesis on the problem?

v. Would my contribution(s) be original and non-trivial?

vi. Is my proposed approach concrete, clearly explained, and of justifiable promise?

vii. How would I position this research project in the international context of research in the field?:
   A. Has this ever been done before? Or am I merely extending current work for larger problems (bigger inputs)?
   B. How many people are working on this research topic, or in a similar topic?
   C. What would make this project more significant than theirs?
   D. Where would I rank in comparison to them? Are we the best research team in this topic? if not, top 3? (or top 5?) Top 10?? Top 20??? Insignificant???

viii. Proposed research methodology/solution (or method):
   A. What main techniques, experiments, and trials would I be using?
   B. include detailed descriptions of what I plan to do:
      - theorems to formulate and prove
      - systems to build

    - experiments to perform
  C. what is my exposition/exposé of the way chosen to reach my goals
  D. for the aforementioned problem(s) to be attacked, what are my approaches for attacking them?
    - details for my approaches do not need to be fully worked out; indeed, they shouldn't be at this stage
    - however, there needs to be enough information for the committee to estimate the likelihood that the approach will work, and to estimate the difficulty of completing the thesis
  E. **What would be my original and useful contribution(s) to the body of knowledge in formal verification, or electronic design automation?**
  F. What is the novelty and groundbreaking character of the proposed research?
  G. Use `divide and conquer` to break the problem into subproblems
  H. For each subproblem that I would be solving, come up with a set of objectives for the subproblem
  I. Is each objective numerically quantifiable? Can I measure properties (or physical quantities, or modal properties – such as verifiability) of the algorithm, circuit, methodology, or system? If so, how should I quantify them? What metrics would I be using?
  J. What is the scientific basis for my evaluation of the experimental results? Or rather, how do I plan to evaluate my experimental results?
  K. Verify if meeting this objectives would solve this problem
  L. Have I convinced the examiners that my proposed solutions can solve the problem that I set for myself?
  M. Are my solutions effective and efficient? Can I further improve its computational time and space complexity? Are my solutions scalable for larger problems (bigger inputs), and for larger parallel computers (bigger server farm, `Linux` cluster, GPGPU computing platform, or bigger processor (64-bit processors and/or multi- and many-core processors))?
  N. Have I benchmarked my EDA tool [for VLSI formal verification]? Does it outperform the other EDA tools in the public domain?
  O. Have I also convinced the examiners that I have foreseen potential roadblocks, such as blind alleys and dead ends? Have I taken appropriate measures to avoid them, and documented this in my proposal? Have I listed contingency plans for potential roadblocks in my proposal? Indicate that I have a reasonable risk/change management process?
  P. Have I shown that my proposed solutions and research directions are relevant to solving the problem?

ix. Indicate how the techniques and ideas of my reference papers play a role in the directions that I am taking

x. Accomplished tasks and milestones met:
  A. If I have solved some of the subproblems, summarize the results
  B. relevant publications (conference/journal papers, or technical reports) may be attached as appendices

xi. Desired/Expected outcomes:
  A. Requirements and deliverables of the Ph.D. research. And, evaluation method for the verification of the requirements and deliverables. E.g., "we will use the XY benchmark and compare to the figures in Z's article."
  B. What are my expected results in relation to the current international status of re-

search in this field?

C. What is the impact and potential for promoting scientific innovation, both inside and outside the field?

D. What are the deliverables?:
   - A SMT-solver?
   - And, a MPSoC formal verification tool based on that SMT-solver?
   - A combined SMT/CSP solver that is integrated with automated reasoning and bit precision reasoning?
   - A MPSoC formal verification tool based on that combined system?
   - What will the MPSoC formal verification tool verify? All the subsequently mentioned circuits, subsystems, and systems?
   - Benchmark circuits for the MPSoC components and system-level/behavioral models of MPSoCs that I want to test my formal verification tools with??? **Do I have to develop such benchmarks? Or are these benchmarks available?**
   - Software test automation framework that facilitates regression testing; this is also known as the test harness or automated test framework. It includes a driver script, or startup script, and a set of executable test suites; a test suite is a collection of test cases, and each class and module in the software shall have a test suite.
   - Software documents for each software project/tool, based on IEEE Software Document Definitions:
     – Software Requirements Specification (SRS), IEEE 830
     – Software Project Management Plan (SPMP), IEEE 1058
     – Software Design Description (SDD), IEEE 1016 – Software Design Document in my SEP (Software Engineering & Project) class at the University of Adelaide ... Includes software architecture design in UML, and pseudo code for SAT solver and formal verification tool
     – Software Configuration Management Plan (SCMP), IEEE 828
     – Software Quality Assurance Plan (SQAP), IEEE 730; see http://readyset.tigris.org/nonav/templates/qa-plan.html
     – Software Test Documentation (STD), IEEE 829
     – Software Validation & Verification Plan (SVVP), IEEE 1012
     – User manual
     – Application programming interface (API) and call graph for my software, in the following formats: PDF, man pages, and LaTeX

E. What is the significance and importance of my work?

6. Who is/(are) my proposed supervisor(s)?
7. Scheduled plan for classes taken in this first 3 semesters
8. Project timeline: work schedule (or working plan), with milestones (e.g., literature review, software implementation, software testing, tool analysis, and writing) and targets:
   (a) A deliverable is different from a project milestone in the sense that a milestone is a way of checking to what extent you are making progress to an outcome. A milestone can be completing the planning phase of the project while a deliverable includes things like design documents, testing scripts, requirement documents etc.
   (b) indicate previous coursework and research experience that prepared me for research in this field
   (c) indicate what will I do in preparation for Ph.D. research in the program, prior to its commencement; **intern in a VLSI formal verification group (or another EDA group) at National Taiwan University?** *Develop SMT solver* **TrojanSMT Lite***???* *Develop tool for HWMCC???* **Design Automation Summer School??? ARTIST Summer School in Europe 2009? Check if I can use research in this phase for my Ph.D. dissertation!!!**

(d) list specific milestones, and a timeline for when I expect to meet each milestone; the timeline should schedule writing up of results along the way, and indicate possible conferences or journals to which the work should be submitted

(e) Completion of comprehensive literature review of my research problem

(f) Completion of each subproblem that will help me solve my research problem

(g) for each subproblem, include the following milestones/tasks:

   i. brief literature review to keep myself to date with contemporary solutions to subproblem

   ii. either select one of the best solution to implement, tweak the "best" existing solution, or develop a new solution

   iii. the development of any solution shall include the following tasks:

      A. development of an algorithm or heuristic

      B. if an algorithm is developed, mathematically prove that it works; if necessary, use axioms lemmas ... Also, does this algorithm have any corollary?

      C. if a heuristic is developed, indicate if a metaheuristic is used; if so, which metaheuristic is used

   iv. implementation of the solution

   v. verify, test, and validate the solution

   vi. benchmark this solution

   vii. analyze this solution and draw conclusions from my experimental results; this should also involve a statistical analysis of my experimental results

   viii. write up my solution and findings in a technical report and/or conference paper; this shall form a section or chapter in my Ph.D. dissertation

(h) When developing and updating this timeline, for each task that I assume will take time t to complete, plan for a duration of $1.5\times t$ to complete the task

(i) Course credits in **ICT doctoral courses** to be completed: $\geq$15 in 1st year, & $\geq$9 in 2nd year; $\geq$24 in the first two academic years

(j) Completion/Progress of classes that are required for the degree program

(k) Suggested timeframe for the **qualifying examination (Quals)**

(l) Complete a draft of my Ph.D. thesis proposal by the end of my first year, get the draft reviewed before the 3rd week of the second year, make corrections to the proposal in the next fortnight, and repeat review-correct process (2 weeks for review, 2 weeks for correction) no more than thrice ... `Else, it may be an indication that I would have chosen a bad research problem`

(m) Pass my qualifying examination before the 18th month mark

(n) Pass my qualifying examination before I commence my first internship. This lifts a burden off me, and brings me closer towards graduation. It can also bring a focus to my research to a specialized (sub-)topic, and help me find an internship that better suits my research interests; if I can use work from my research internship for my thesis, this may allow me to knock of some milestones (solve some subproblems) and bring me closer to graduation.

(o) Tasks to work on for a thesis on SMT-based formal verification of MPSoCs:

   i. develop SMT solver **TrentoSMT Lite**: develop parser for input boolean function of predicates, implement SAT solver engine, and develop output generator

   ii. develop *tool* to transform a representation of a VLSI circuit into a boolean function of predicates (so that I can verify it with the SMT solver)

   iii. VLSI circuits to verify:

      A. control unit: *break it into modules for the 5-stage pipelined MIPS architecture???*

      B. data path unit:

         • multiplexer
         • adder
         • comparator
         • multiplier

- encoder/decoder
- floating-point unit (FPU)
- bus architecture: bus network topology / bus network architecture

C. memory components:
- register files
- level 1 (L1) cache
- level 2 (L2) cache
- level 3 (L3) cache
- memory controller, or memory management unit (MMU)

D. analog components (timing sources): oscillators and phase-locked loops

E. mixed-signal components: ADC & DAC

F. communication fabric of the MPSoC:
- crossbar switch
- cover different network topologies

G. system-level designs of multiprocessors:
- homogeneous multicore processors (SMP)
- heterogeneous MPSoCs (AMP) that include:
  - general-purpose processor cores
  - digital signal processors (DSPs)
  - graphics processing units (GPUs)
  - network processor / network processing unit (NPU)
  - cryptographic accelerator / crypto-processor

iv. plan and schedule these tasks on a development roadmap at 3, 6, and 12 month intervals, as well as the entire 3-year Ph.D. program

(p) Include conferences, workshops, and symposiums that I plan to attend:

i. ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages (POPL 201X) – Jan
ii. Asia and South Pacific Design Automation Conference (ASPDAC) – Jan
iii. International Conference on High Performance Embedded Architectures & Compilers (HiPEAC 201X) – Jan
iv. Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools (RAPIDO'1X) – Jan
v. Workshop on Computer Architecture and Operating System co-design (CAOS) – Jan
vi. HiPEAC Workshop on Design for Reliability (DFR'1X) – Jan
vii. Workshop on Interconnection Network Architectures: On-Chip, Multi-Chip (INA-OCMC) – Jan
viii. International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems (TAU Workshop) – Feb / Timing issues in EDA, performance optimization, delay minimization, clock domains, asynchronous VLSI
ix. Euromicro International Conference on Parallel, Distributed and Network-Based Computing: Special Session on On-Chip Parallel and Network-Based Systems – Feb
x. International Symposium on High-Performance Computer Architecture (HPCA-1X, or HPCA-2X) – Feb
xi. International Conference on Architecture of Computing Systems (ARCS) – Feb/Mar
xii. Workshop on Dependability and Fault Tolerance – Mar
xiii. Workshop on Many Cores – Mar
xiv. International Workshop on Multi-Core Computing Systems (MuCoCoS) – Mar
xv. International Symposium on Quality Electronic Design (ISQED) – Mar
xvi. Workshop on Synthesis And System Integration of Mixed Information technologies (SASIMI 201X) – Mar
xvii. International Test Synthesis Workshop (ITSW) – Mar
xviii. International Conference on Simulation Tools and Techniques (SIMUTools 201X) – Mar
xix. ACM Symposium on Applied Computing (SAC 201X) – Mar

xx. Design, Automation and Test in Europe (DATE) – Mar/Apr
xxi. Predictability and Performance in Embedded Systems Workshop (PPES 2011) – Mar/Apr
xxii. International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 201X) – Mar/Apr
xxiii. International Symposium on Physical Design (ISPD) – Apr
xxiv. IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS 201X) – Apr
xxv. International Symposium on Code Generation and Optimization (CGO 201X) – Apr
xxvi. IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 201X, which is part of Cyber-Physical Systems week / CPSWeek) – Apr
xxvii. International Conference on Hybrid Systems: Computation and Control (HSCC'1X) – Apr
xxviii. Southern Programmable Logic Conference (SPL201X) – Apr
xxix. International conference on Design & Technology of Integrated Systems in nanoscale era (DTIS'1X) – Apr
xxx. International ICST Conference on Practice and Theory of Algorithms in (Computer) Systems (TAPAS 201X) – Apr
xxxi. International Conference on Thermal, Mechanical and Multiphysics Simulation and Experiments in Micro-Electronics and Micro-Systems (EuroSimE 201X) – Apr
xxxii. IEEE International Systems Conference 201X (SysCon 201X) – Apr
xxxiii. International ICST Conference on Nano-Networks (Nano-Net 201X) – Apr
xxxiv. International Workshop on Formal Methods for Globally Asynchronous Locally Synchronous Design (FMGALS) – Apr
xxxv. Annual Electronic Design Process Symposium (EDPS 201X) – Apr
xxxvi. IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS) – Apr
xxxvii. International Symposium on VLSI Design, Automation and Test (VLSI-DAT) – Apr
xxxviii. The European Future Technologies Conference and Exhibition (FET1X / $f\mathrm{ET}^{1X}$; biennial event) – Apr/May
xxxix. Great Lakes Symposium on VLSI (GLSVLSI) – Apr/May
xl. International Review of Progress in Applied Computational Electromagnetics (ACES) – Apr
xli. Annual IEEE/SEMI® Advanced Semiconductor Manufacturing Conference (ASMC) - International Symposium on Semiconductor Manufacturing (ISSM) ... new conference in 2010: (ASMC-ISSM) – May
xlii. IEEE VLSI Test Symposium (VTS 201X) – May
xliii. International Symposium on Embedded Systems Design and Applications (ESDA 201X, in Romania) – May
xliv. IEEE Computer Society Annual Symposium on VLSI (ISVLSI) – May
xlv. IEEE International Symposium on Circuits and Systems (ISCAS 201X) – May
xlvi. Reconfigurable Architectures Workshop (RAW 201X) – May
xlvii. International Symposium on Asynchronous Circuits and Systems (ASYNC) – May
xlviii. ACM Great Lakes Symposium on VLSI (GLVLSI) – May
xlix. International Symposium on Networks-on-Chip (NOCS) – May
l. International ICST Conference on Performance Evaluation Methodologies and Tools (Valuetools 201X) – May
li. ACM International Conference on Computing Frontiers (Computing Frontiers 201X) – May
lii. IEEE International Symposium on Multiple-Valued Logic (ISMVL 201X) – May // `http://www.lsi-cad.com/ismvl/`
liii. International Workshop on Post-Binary ULSI Systems (ULSI 201X) – May
liv. Reed-Muller 201X Workshop – May // `http://www.lsi-cad.com/RM/index.html`

lv. IEEE European Test Symposium (ETS '11) – May
lvi. ACM Symposium on Theory of Computing (STOC 201X) – May/Jun
lvii. ACM Symposium on Parallelism in Algorithms and Architectures (SPAA '1X): – May/Jun
lviii. ACM International Conference on Measurement and Modeling of Computer Systems (ACM SIGMETRICS 201X) – May/Jun
lix. International ACM Symposium on High-Performance Parallel and Distributed Computing (HPDC 201X) – Jun
lx. Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 201X) – May/Jun
lxi. ACM SIGACT/SIGMOBILE International Workshop on Foundations of Mobile Computing (DIALM-POMC 201X) – Jun/Sep
lxii. International Supercomputing Conference (ISC'10) – May/Jun
lxiii. International Symposium on Memory Management (ISMM 201X) – Jun
lxiv. Annual IEEE International Mixed-Signals, Sensors, and Systems Test Workshop (IMS3TW'1X) – Jun
lxv. International Symposium on Computer Architecture (ISCA 201X) – Jun
lxvi. IEEE Symposium on Computer Arithmetic (ARITH) – Jun
lxvii. International Symposium on Rapid System Prototyping (RSP) – Jun
lxviii. International Symposium on Symbolic and Algebraic Computation (ISSAC 201X) – Jun
lxix. International Symposium on Formal Methods (FM201X; e.g., FM2009, FM2010) – Jun / Nov
lxx. International Conference on Application and Theory of Petri Nets and Other Models of Concurrency (Petri Nets) – Jun
lxxi. ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES 201X) – Jun
lxxii. 201X Symposium on VLSI Technology (part of the 201X Symposia on VLSI Technology and Circuits) – Jun
lxxiii. 201X Symposium on VLSI Circuits (part of the 201X Symposia on VLSI Technology and Circuits) – Jun
lxxiv. International Conference Mixed Design of Integrated Circuits and System (MIXDES 201X) – Jun
lxxv. International Conference on Computer Aided Verification (CAV) – Jun/Jul
lxxvi. Hardware Verification Workshop (HWVW'1X) – Jun/Jul
lxxvii. Workshop on Formal Verification of Analog Circuits (FAC) – Jun/Jul
lxxviii. Workshop on Constraints in Formal Verification (CFV) – Jun/Jul
lxxix. Platforms for Analysis, Design and Verification of Embedded Systems (PADVES) – Jun/Jul
lxxx. International Conference on Theory and Applications of Satisfiability Testing (SAT) – Jun/Jul
lxxxi. Design Automation Conference (DAC) – Jun/Jul
lxxxii. International Workshop on Logic and Synthesis (IWLS) – Jun/Jul
lxxxiii. International Workshop on Bio-Design Automation (IWBDA 201X) – Jun/Jul
lxxxiv. ACM/IEEE International Conference on Formal Methods and Models for Codesign (MEMOCODE) – Jun/Jul
lxxxv. IEEE International Workshop on Design for Manufacturability and Yield 201X (DFM&Y) – Jun/Jul
lxxxvi. International Workshop on System Level Interconnect Prediction (SLIP) – Jun/Jul
lxxxvii. IEEE Symposium on Application Specific Processors (SASP) – Jun/Jul
lxxxviii. IEEE International Conferences on Embedded Software and Systems (ICESS-1X) – Jun/Jul
lxxxix. International Conference on Application of Concurrency to System Design (ACSD) – (Jun/)Jul

xc. Euromicro Conference on Real-Time Systems (ECRTS) – Jul
xci. IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP 201X) – Jul
xcii. International Conference on Tests & Proofs (TAP 201X) – Jul
xciii. International Symposium on Signals, Circuits and Systems (ISSCS 201X, in Romania) – Jul
xciv. International Symposium on Software Testing and Analysis (ISSTA 201X) – Jul
xcv. International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (IC-SAMOS) – Jul
xcvi. International Workshop on Systems, Architectures, Modeling, and Simulation (SAMOS) – Jul
xcvii. Metaheuristics International Conference (MIC 201X) – Jul
xcviii. International Workshop on First-Order Theorem Proving (FTP 201X) – Jul
xcix. Conference on Automated Deduction (CADE) – Jul/Aug
c. International Workshop on Satisfiability Modulo Theories (SMT) – Jul/Aug
ci. International Joint Conference on Automated Reasoning (IJCAR 201X) – Jul/Aug
cii. Pragmatics of Decision Procedures in Automated Reasoning (PDPAR 201X) – Jul/Aug
ciii. IEEE Symposium on Logic in Computer Science (LICS 201X) – Aug
civ. International Static Analysis Symposium (SAS 201X) – Aug
cv. International Conference on Theorem Proving in Higher Order Logics (TPHOLs) – Aug
cvi. International Forum on Application-Specific Multi-Processor SoC (MPSoC) – Aug
cvii. IEEE International Midwest Symposium on Circuits and Systems (MWSCAS 201X) – Aug
cviii. EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA) – Aug
cix. International Symposium on Low Power Electronics and Design (ISLPED) – Aug
cx. European Conference on Circuit Theory and Design (ECCTD'201X) – Aug
cxi. IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 201X) – Aug
cxii. Workshop on Highly Parallel Processing on a Chip (HPPC 201X) – Aug
cxiii. Euro-Par 2010 – Aug/Sep
cxiv. Euromicro Symposium on Digital System Design: Architectures, Methods and Tools (DSD) – Aug/Sep
cxv. Symposium on Integrated Circuits and Systems Design (SBCCI) – Aug/Sep
cxvi. IEEE International Behavioral Modeling and Simulation Conference (BMAS) – Sep
cxvii. IEEE/IFIP International Conference on VLSI and System-on-Chip (VLSI-SoC 201X) – Sep
cxviii. International Workshop on Power And Timing Modeling, Optimization and Simulation (PATMOS 201X) – Sep
cxix. International Symposium on Embedded Multicore Systems-on-Chip (MCSoC-n), where $n \in \mathbb{N}$ – Sep
cxx. Annual Conference of the Italian Operational Research Society (AIRO201X) – Sep
cxxi. International Symposium on Frontiers of Combining Systems (FroCoS) – Sep
cxxii. EACSL Annual Conference on Computer Science Logic (CSL) – Sep
cxxiii. IEEE Custom Integrated Circuits Conference (CICC) – Sep / Compact models for active and passive devices, behavioral modeling, signal-integrity modeling and simulation. System and circuit simulation. Parasitic extraction and reduction. Simulation techniques for analog, RF, and mixed-signal circuits. Package modeling. Process variation, statistical, and reliability modeling. Compact models for extreme environment operation. SOI and multiple gate device modeling.
cxxiv. European Microwave Integrated Circuits Conference (EuMIC) – Sep
cxxv. European Microwave Conference (EuMC) – Sep
cxxvi. International Symposium on Symbolic and Numeric Algorithms for Scientific Computing

(SYNASC 201X, in Romania) – Sep

cxxvii. IEEE International Conference on 3D System Integration (3D IC) – Sep
cxxviii. International Conference on Simulation of Semiconductor Process and Devices (SISPAD 201X) – Sep
cxxix. IEEE East-West Design & Test Symposium (EWDTS) – Sep
cxxx. Forum on specification & Design Languages (FDL 201X) – Sep
cxxxi. System, Software, SoC and Silicon Debug (S4D) Conference – Sep
cxxxii. Scientific Computing in Electrical Engineering conference (SCEE 201X) – Sep
cxxxiii. International Test Synthesis Workshop (ITSW 201X) – Sep
cxxxiv. EMC Europe 201X – International Symposium on EMC and International Wroclaw Symposium on EMC – Sep // http://www.emceurope.org/2010/dates.html
cxxxv. NVIDIA GPU Technology Conference, including the `NVIDIA Research Summit`; see http://www.nvidia.com/object/gpu_tech_conf_research_summit.html – Sep/Oct
cxxxvi. International Conference on Parallel Architectures and Compilation Techniques (PACT 201X) – Sep/Oct
cxxxvii. International Symposium on System-on-Chip (SoC) – Oct
cxxxviii. Haifa Verification Conference (HVC) – Oct
cxxxix. International Workshop on Thermal Investigations of ICs and Systems (THERMINIC 201X) – Oct
cxl. International Microsystems, Packaging, Assembly and Circuits Technology Conference (IMPACT 201X) – Oct
cxli. International Conference on Compilers, Architecture, and Synthesis for Embedded Systems (CASES, which is part of Embedded Systems Week / ESWeek) – Oct
cxlii. International Conference on Hardware/Software Codesign and System Synthesis [or International Conference on Hardware-Software Codesign and System Synthesis, CODES+ISSS, which is part of Embedded Systems Week / ESWeek] – Oct
cxliii. International Conference on Embedded Software (EMSOFT, which is part of Embedded Systems Week / ESWeek) – Oct
cxliv. Workshop on Compiler-Assisted System-On-Chip Assembly 201X (CASA 201X, which is part of Embedded Systems Week) – Oct
cxlv. Workshop on Software Synthesis (WSS'1X, which is part of Embedded Systems Week) – Oct
cxlvi. International Conference on Computer Design (ICCD) – Oct
cxlvii. IEEE Compound Semiconductor IC Symposium (CSICS) – Oct
cxlviii. International Test Conference (ITC 201X) – Oct / Nov
cxlix. International Conference on Computer-Aided Design (ICCAD) – Nov
cl. IEEE International Workshop on Test and Validation of High Speed Analog Circuits (TVHSAC 201X) – Nov
cli. IEEE International High Level Design Validation and Test Workshop (HLDVT) – Nov
clii. IEEE International Workshop on Testing Three-Dimensional Stacked Integrated Circuits (3D-TEST 201X)
cliii. International Workshop on Parallel and Distributed Methods in verifiCation (PDMC) – Nov
cliv. International Conference on Formal Methods in Computer-Aided Design (FMCAD) – Nov
clv. International Workshop on Microprocessor Test and Verification (MTV) – Dec
clvi. IEEE Circuits and Systems Society Forum on Emerging and Selected Topics (CAS-FEST) – Dec
clvii. IEEE International Conference on Electronics, Circuits, and Systems (ICECS 201X) – Dec
clviii. Annual IEEE/ACM International Symposium on Microarchitecture (Micro-4X) – Dec
clix. International Workshop on Network on Chip Architectures (NoCArc 201X) – Dec

clx. International Conference on Formal Engineering Methods (ICFEM) – Dec
clxi. IEEE Real-Time Systems Symposium (RTSS 201X) – Dec
clxii. IEEE Pacific Rim International Symposium on Dependable Computing (PRDC'1X) – Dec
clxiii. **To learn about more "Call for Papers," see**:
- Computing Research Association: Resources: Conference List for NRC ([probably refers to National Research Council]) Rankings: http://www.cra.org/resources/conferencelist/
- *WikiCFP*: http://www.wikicfp.com/
- *edacentrum's edaCalendar*: https://secure.edacentrum.de/edakalender/index.php?lang=en
- *Researchr*: http://researchr.org/conferencecalendar/
- *Conference Alerts*: http://conferencealerts.com/
- *SemreX CFP*: http://grid.hust.edu.cn:8080/call/index.jsp
- Department of Computer Engineering, Tallinn University of Technology: http://ati.ttu.ee/calls_for_papers.php?show=by_deadline
- Jonas Fritzin, "List of VLSI / IC Conferences and Workshops," Division of Electronic Devices, Department of Electrical Engineering (ISY), Linköpings universitet: http://www.ek.isy.liu.se/~fritzin/conf.html
- Mathematical Optimization Society: conferences in mathematical optimization and mathematical programming; http://www.mathprog.org/?nav=meetings

clxiv. See http://electronicsystemlevel.com/phpBB/viewforum.php?f=10 regarding `call for papers` of conferences related to electronic system-level (ESL) design and verification. [**Defunct.**]
clxv. Intel Logic Verification Symposium, Haifa???
clxvi. Intel Formal Verification Symposium, Haifa???
clxvii. Intel DTS Annual Symposium, Haifa???
clxviii. Intel European Research and Innovation Conference (ERIC). Available at: http://www.intel.com/corporate/education/emea/event/irc/index.htm; last accessed on September 17, 2010.
clxix. Also, check out International Association of Science and Technology for Development (IASTED) conferences; `these conferences and symposiums are of lower quality`
clxx. See http://www.mos-ak.org/ for events on compact modeling by MOS-AK.
clxxi. WARNING: Pay attention to rules about simultaneous submission and prior publication. For example, look at ACM's policy at the following reference. ACM, "Policy on Prior Publication and Simultaneous Submissions," ACM, New York, NY, May 21, 2009. Available online at: http://www.acm.org/publications/policies/sim_submissions; last accessed on September 27, 2010.

(q) Include programming contests that I plan to participate in:
i. International Microwave Symposium: Student Design Competitions – Jan (includes AMS circuit simulation, and AMS/RF EDA); http://ims2011.org/Technical_Program/Student_Design_Competitions.html
ii. QBF solvers competition – Jan
iii. ISPD Programming Contest – Feb/Mar
iv. TAU Workshop's Power Grid Simulation Contest – Mar
v. SAT Competition – Mar
vi. Memocode 201X HW/SW Co-design contest – Mar
vii. Max-SAT Evaluation – Mar
viii. IEEE Programming Challenge at IWLS – Apr
ix. IEEE Computer Society Simulator Design competition [Develop simulator for processor] – Apr ... See http://www.computer.org/portal/web/competition

x. Pseudo-Boolean Competition – Apr
xi. *Apple Design Awards – Apr/May; also see WWDC 201X Student Scholarship (applications are due in mid-April)*
xii. Satisability Modulo Theories Competition – Jul
xiii. Hardware Model Checking Competition (HWMCC) – Jun/Jul
xiv. CADathlon @ ICCAD – Nov
xv. Argonne National Laboratory, ANL; Mathematics and Computer Science Division:
 A. J. H. Wilkinson Prize for Numerical Software (for developers of numerical software): http://www.mcs.anl.gov/research/opportunities/wilkinsonprize/index.php
xvi. Society for Industrial and Applied Mathematics, SIAM:
 A. SIAM/ACM Prize in Computational Science and Engineering: http://www.siam.org/prizes/sponsored/cse.php. [For developers of mathematical and computational tools and methods for the solution of science and engineering. Or, for developers of computational science and engineering software.]

(r) Include summer schools/programs or other short-term workshops that I can attend:
 i. To attend a Summer School, obtain permission of my advisor and follow the mission procedure presented in the Student Career informative system.
 ii. After having attended the Summer School, follow the procedure regarding Summer Schools credit recognition presented in the Student Career informative system.
 iii. Summer schools that may interest me:
  A. EPFL Summer Schools:
   - Nanoelectronic Circuits and Tools: http://si.epfl.ch/page12409.html
   - Workshop and Nano-scale Systems and Technologies: http://si.epfl.ch/page16490.html
  B. First International SAT/SMT Solver Summer School 2011: http://people.csail.mit.edu/vganesh/summerschool/index.html
  C. ARTIST Summer School — See http://www.artist-embedded.org/artist/Applications,1685.html — **APPLY BY MAY 15, 2009**
  D. CompArch Summer School on Parallel Programming and Architectures
  E. HiPEAC International Summer School on Advanced Computer Architecture and Compilation for Embedded Systems (Acaces) — See http://www.hipeac.net/summerschool/index.php?page=home — **APPLY ASAP**
  F. NSF-SIGDA-SRC Design Automation Summer School
  G. COMSON International Summer School on "Modelling and Optimization in Micro- and Nano- Electronics – MOMiNE 2009"; see http://momine2009.unical.it/ ... **APPLY BY JUNE 30**
  H. TACC Summer Supercomputing Institute: http://www.tacc.utexas.edu/summer-institut
  I. RISC Summer 201X
  J. RISC/SCIEnce Training School in Symbolic Computation
  K. Summer School on Algebraic Analysis and Computer Algebra
  L. CRA-W/CDC Summer Schools:
   - CRA-W/CDC Computer Architecture Summer School
   - See CRA-W/CDCComputerArchitectureSummerSchool for a list of summer schools covering different topics in computer science. This summer schools are organized/co-organized by the Computer Research Association's Committee on the Status of Women in Computing Research (CRA-W).
  M. Summer School of Information Engineering, Bressanone (BZ), Italia: http://www.dei.unipd.it/ssie
  N. lists of summer schools:

- Uppsala University, Department of Computer Systems, Formal Methods and Software Technology group: http://user.it.uu.se/~bengt/Info/summer-schools.shtml
- Pervasive Adaptation (PERADA)-ASSYST Summer School on Adaptive Socio-Technical Pervasive Systems: http://www.perada.eu/other-summer-schools

O. EURO-DOTS (European Doctoral Training Support in Micro/Nano-electronics): http://www.eurodots.org/ and http://www.eurodots.org/Downloads.asp (Public deliverables of EURO-DOTS).

P. Santa Fe Institute: Complex Systems Summer School (http://www.santafe.edu/education/schools/complex-systems-summer-schools/), and Global Sustainability Summer School (http://www.santafe.edu/education/schools/global-sustainabilit

Q. New England Complex Systems Institute (NECSI): NECSI Summer and Winter School (http://www.necsi.edu/education/school.html)

R. Wolfram Research: Wolfram Science Summer School 201X, http://www.wolframscience.com/summerschool/

(s) Internship(s):
   i. Include details of any (foreseen) collaboration with other departments or research institutions (e.g., ITC-irst Trento) that I am collaborating with for my Ph.D. research and my internships, which is required for the CS Ph.D. at Uni Trento
   ii. Intel Haifa, or elsewhere: say Intel Strategic CAD Labs (SCL), IBM Haifa Research Lab (HRL), Cadence Research Laboratories (CRL), or the Institute for Formal Models and Verification (FMV) at Johannes Kepler Universität (JKU). Other research groups that I can intern at include the VLSI formal verification groups at Stanford University (Stanford), Carnegie Mellon University (CMU), University of Texas at Austin (UT Austin), and the formal verification groups at the Technion's Department of Computer Science and the William Davidson Faculty of Industrial Engineering & Management.
   iii. At the University of Trento's CS Ph.D. program, I have to complete internships overseas for: 2× 15 course credits. Since I have to complete 60 course credits a year, it means each internship shall last for 3 months. Thus, I have to complete 2× 3-month internships overseas.

(t) Speed boosts for developing SMT solvers, SMT-based formal verification tools, combined systems for VLSI formal verification:
   i. plan for hacking sessions with labmates (throw hacker parties):
      A. coding sprints (1/2 or 2/3 days)
      B. hackathon (few days to a week)
      C. codefest (hackathon for Linux developers)
   ii. stock up caffeinated drinks: coffee, chai/tea, energy drinks (Rockstar/ROCKST★R and Red Bull)
   iii. plan for all-nighters around long weekends, so that I have time to recover

(u) Suggested timeframe for writing the [remainder/bulk] of my Ph.D. dissertation; save about 3-6 months for this

(v) Suggested timeframe for the `Doctoral Thesis examination`; leave about 4-8 weeks to arrange a suitable date for the defense, when all members of the thesis committee can show up, and another 4-8 weeks to make revisions to my dissertation (if necessary)

(w) The Academic year begins on November, 1st and ends on October, 31st.

(x) summary of timeline:
   i. 1^{st} year Exploration research plan; Literature study
   ii. 2^{nd} Execution of research; Publication
   iii. 3^{rd} Execution of research; Publication

iv. 4th Complete research; PhD Thesis
v. In my 2nd and 3rd years, supervise master students and interns; tutorship (TA)
vi. Pass Quals in 3 months; absolute deadline to pass Quals (end of 9th month)

9. (Explanatory) budget / funding plan:
   (a) check out scholarships at http://scholarship.bursa-lowongan.com/
   (b) suggestions on how to get funding: http://www.aaai.org/Library/Funding/funding-library.php
10. Estimate of required scientific, computing, and material resources:
    (a) What resources will your research require?
11. References (max 20 for Uni. Trento)
12. This proposal will be revised through several rounds of iteration with the Ph.D. advisor(s)
13. Judge my progress based on deliverables and performance/quality of the FV tool
14. Evaluate my research proposal to assess the following:
    (a) **Ask myself the following:**
        i. Have I solved the worthwhile problem, or answered the question?
        ii. Has my thesis served as a formal document that proves that I have made an original and useful contribution to knowledge in formal verification, or EDA? That is, has my dissertation served its sole purpose?
    (b) **Does my Ph.D. thesis help the examiners on my thesis committee answer the following questions?**
        i. **What is this student's research question?**
        ii. **Is it a good question? (Has it been answered before? Is it a useful question to work on?)**
        iii. **Did the student convince me that the question was adequately answered?**
        iv. **Has the student made an adequate contribution to knowledge?**
        v. If your thesis does not provide adequate answers to the few questions listed above, you will likely be faced with a requirement for major revisions or you may fail your thesis defence outright.
    (c) Merit/Quality of the research [How innovative is this? Radically innovative???]
    (d) Originality of the research
    (e) Applicability of the proposed research: How useful would this research be?
    (f) How reasonable is the budget?

Resources for writing Ph.D. thesis proposals (and other research proposals):

1. The University of Manchester:
   (a) The University of Manchester, "Advice on Writing a Research Proposal," School of Computer Science, The University of Manchester. Available at: http://www.cs.manchester.ac.uk/phd/proposal/; last accessed on September 2, 2010.

2 Considerations about Any Ph.D. Program

Stuff to consider when pursuing a degree program, and when selecting classes or specialization track (if any):

- Take into account your interests and strengths

- Theory versus programming/projects (practical work)
- Descriptive system courses versus more abstract material
- Depth versus breath
- Career plans

2.1 Questions about the Ph.D. program for Professor or Administrator

Questions to ask about the Ph.D. program (ask a professor or administrator):

- Does this Ph.D. position include any teaching responsibilities? Or, do Ph.D. students have teaching responsibilities?
- Quals (Qualifying Exam): Does the qualifying exam only include the research proposal? Do I also have to pass an oral examination and/or a written examination, which tests the breath and depth of my technical knowledge in my research topic? How many attempts am I allowed to have? Or rather, how many times can I attempt to take the qualifying exam?
- Do I have to take the Italian intensive and extensive language courses under CIAL?
- Can I do research overseas at Intel Haifa for two semesters, or Intel Haifa and CMU/Berkeley?
- What is the grading policy for classes? What is the weightage for midterms (midterm examination), finals (final examination), term/research paper, papers, projects (presentations and reports)
- Does the university have a dead week, swotvac, or study days before finals? Reading week???
- Does the uni/program have a finals week for Ph.D. students?
- Does the university have a spring/fall recess, or spring/fall break?
- The CS Ph.D. program at the University of Trento is a sandwich PhD program, which requires me to do an internship at a company or another academic institution
- How can I get data on student outcomes such as internship placement, attrition rates, indebtedness and types of employment?
- Can I complete all my coursework requirements in my first year? That is, can I take 24 course credits for ICT doctoral courses in my first year, and take no classes in my second and third years?
- Can I take my quals in my first year?
- What is the average number of Ph.D. graduates per year for this university? Remember USC confers $>$300 doctoral degrees in 2009, and Stanford conferred $\approx$700 (or about 670)

2.2 Questions about the Ph.D. program for Current Grad Students

Questions to ask about the Ph.D. program (ask a grad student):

- How do grad students feel about their advisors, thesis committee members, and department staff?
- How do grad students feel about stress, pressure, and the research-life (work-life) balance
- What are the feelings about the University and the research lab that grad students work in as opposed to their lab mates that they work with?
- How much is the university thought by their faculty, staff, and students to put back into society in general, and the local community in particular?
- To what extent do the grad students feel that they are stretched and challenged by their research

3 Statement of Purpose

Advice for writing the statement of purpose, or the short “mission statement”:

1. include a short ($\frac{1}{2}$ page to two pages) general description of what kinds of project I would like to tackle (taking into account the research goals as described on the web page of the research lab that I want to join)
2. I must have some background in one or more of the research areas of the lab that I want to join
3. a statement that you have to write:
   (a) Why do I want to get a Ph.D. degree?
   (b) What interests me in Computer Science?
   (c) What topics do I want to work on?
   (d) What do I plan to do after I get a Ph.D.?
4. this statement is limited to two pages (in a font that can be read without magnifying glass)
5. a good estimate is no more than 110 lines of (at most) 75 characters each
6. if I participated in some research project as an undergraduate, then mention this fact in my statement (and my CV):
   (a) What did I learn?
   (b) Why was my project interesting?
   (c) What was difficult about it?
   (d) What kind of programming projects have I done?
   (e) I indicated an interest in VLSI formal verification – what is my experience, building software?
7. identify a research topic, pursue the research, and then report and disseminate the results
8. a student must spend some time surveying the field, and may very well pursue a few dead ends ... Better to do this as a student than in my first job.
9. Other advice:
   (a) list my short- and long- term goals, area of interest, why do I wanna pursue this degree
   (b) What are some of your most influential past experiences?:
      i. Note major turning points in your life, travel experiences, important people, life lessons, etc.
      ii. Note past successes/ failures.
      iii. This is related to the previous point, but make sure to jot down major successes, how you accomplished them, and major failures, what you learned.
      iv. Do not be afraid of writing about your failures.
      v. It is sometimes from our failures that we learn the most, and they can make for quite compelling essays.
      vi. Why is your perspective valuable?
      vii. Think about what it is that makes your specific perspective valuable and why you would be a valuable contributor to the programs to which you are applying.
   (c) Thesis idea and funding possibilities.:
      i. If you could do any type of research what would it be?
      ii. Can you imagine an ideal project that you could spend a few years on, particularly if you are looking to earn a Ph.D.?
      iii. What are some organizations that are funding this type of research?
      iv. Remember: Universities are businesses:
         A. You need to be able to present a clear ”business plan” of what it is you are going to do and how you are going to fund it.
      v. What is it you most want to do?:

   A. This is different than long and short term goals.
   B. This is number one.
   C. What drives you?
   D. What gets you up in the morning?
   E. In many ways, this is the hardest question to answer.

10. See http://graduate-schools.suite101.com/article.cfm/brainstorming_ideas for ideas to brainstorm about, while writing my statement of purpose

4 Advice for Getting Letters of Recommendation

Advice for getting letters of recommendation:

1. use http://www.interfolio.com/fh_recletters.html to get my professors at USC and Adelaide Uni to send letters of recommendation to graduate programs that I am applying to; this is suggested by Shayna, email her at pregrad@usc

5 Issues about the Ph.D. program and its Internship

5.1 Issues That May Concern Me Regarding Internship Programs

Issues that may concern me regarding internship programs:

- Does it have a lightweight IP (intellectual property) program?
- Can I use the work from my internship in my thesis?

5.2 Questions To Dwell Upon While Considering Internship Opportunities

Questions to dwell upon while considering internship opportunities:

1. What would the quality of the internship experience be like? What can I expect to gain from the internship experience?
2. What preferences do I have for a research internship?:
   (a) **What type of environment do I work in?**
   (b) multicultural/diverse, inclusive, and people-friendly environment
   (c) work in a team of really, really smart people ... they should be sufficiently smart enough to overcome bigotry
   (d) do research and development in my Ph.D. research area/topic
   (e) develop new algorithms, heuristics, or methodologies
   (f) implement EDA tools, and/or design VLSI circuits and systems
   (g) verify, test, and validate EDA tool(s), and/or VLSI circuit(s) and system(s)
   (h) technical documentation for:
      i. requirements
      ii. specifications
      iii. design
      iv. implementation
      v. verification
      vi. testing

        vii. validation
        viii. user manuals
    (i) (some) freedom in setting up or customizing my software development, or IC design environment; choice of:
        i. project management software/tools: e.g., project planning (document/maintain Gantt or Pert charts); document management; resource management; change management; risk management plan; quality management; and project portfolio management
        ii. collaborative software: collaborative project management tools, and collaborative management tools
        iii. requirements management tools
        iv. design modeling tools: UML [design] tools
        v. integrated development environments (IDEs)
        vi. text editors
        vii. documentation generators
        viii. software building tools (compile, link): compilers, and build automation tools
        ix. code analysis, inspection, and referencing tools; e.g., performance analyzer, profiler, or automated code review software
        x. test automation tools; including test suites
        xi. debuggers; e.g., memory debugger
        xii. problem reporting/tracking tools: bugtracker tool or bug tracking system; engineering change order (ECO) management system; or issue tracking system
        xiii. refactoring tools
        xiv. configuration management tools
        xv. source code generation tools
        xvi. compiler generator, or compiler-compiler: lexical analyzers; and parser generator that creates a parser, interpreter, or compiler
        xvii. *GNU toolchain*

3. Where would I be interning?:
    (a) Do I have a preference for the location of the internship?
    (b) Think about:
        i. availability of Protestant church
        ii. weather
        iii. political and socioeconomic stability: absence of civil unrest, riots, and war
        iv. vulnerability to natural calamities:
            A. avalanches
            B. blizzards
            C. bushfires and wildfires
            D. droughts
            E. earthquakes
            F. famines
            G. floods
            H. heat waves
            I. landslides
            J. pandemics
            K. storms (dust storms or sandstorms, firestorms, hailstorms, ice storms, snowstorms, tornadoes, and tropical cyclones)
            L. tsunamis
            M. volcanic eruptions
        v. availability and quality of health care
        vi. presence of allergens in the physical environment: e.g., dust mites, animal dander, and pollens

vii. personal safety issues, and crime rate
viii. availability, extent, and quality of public transport (including the frequency of buses/-trains)
ix. availability of sporting facilities or trails to run
x. availability of entertainment outlets, and entertainment activities:
    A. art galleries/museums
    B. cinemas
    C. concert halls
    D. festivals (art, film, and/or jazz festivals)
    E. shopping malls
    F. theatres
xi. national/federal and state/local government policies
xii. effectiveness of the local/national police
xiii. life/medical insurance
xiv. immigration issues: work visa, permanent residency, and citizenship
xv. opportunities for postdoc research, and (expected) availability of permanent full-time positions
xvi. What are the common local languages used? English???

4. What are the living costs of the city/town where I am interning?:
    (a) rental housing cost
    (b) cost of dining in local restaurants
    (c) cost of groceries
    (d) transportation costs (public transport system, and cabs/taxicabs/taxis)
    (e) cost of common entertainment activities (e.g., movies)
    (f) cost of travel to the location of the internship from my university
5. needs of loved ones, such as those of my significant other (if I have one then)
6. occupational, health, and safety issues of workplace
7. policies against discrimination in the workplace
8. non-disclosure agreement (NDA): extent (previous/current IP owned???), and expiry of NDA
9. What is the quality of the supervision? What is the work or corporate (company) culture like?
10. **Try to answer as many of these questions as possible if and when I go for the on-site interview, or ask the human resource personnel if and when I get the internship offer**
11. Do I have time for the internship application?

5.3 Notes and Advice About Internships

Notes and advice about internships:

1. To deal with scheduling difficulties with internship supervisor/manager, try arranging weekly appointments and reschedule when necessary. Else, try phone meetings, or inform the internship director
2. Attend meetings with an agenda, so that I can get what I want out of the meeting from the meeting
3. Do not fear about making mistakes, and being faulted
4. Do not be sidelined, and ignored by your supervisor/manager
5. Try to pick your supervisor, so that I may have a better chance to click with that person
6. Ask questions relevant to the internship program, such as on specific training models or philosophies, Try to determine the unique aspects of the internship program from the corporate web page, such as seminars and research opportunities

7. figure out what kinds of careers the internship is preparing its interns for, and how I fit into those goals ... Do I really want that kind of professional experiences two, five, and ten years after the internship? If so, why? Then try to convey that in the internship interview(s).
8. Questions that I may be asked during the internship interview:
   (a) Describe the research problem that you are working on. What have you done this far? Or, what progress have you made thus far?
   (b) Describe a problem where you [encountered such a problem (e.g., teammate conflict)].
   (c) Why are you interested in this site?
   (d) What can you bring to this internship?
   (e) What are your long-term career goals?
   (f) Tell me about your dissertation and how you came up with the topic.
9. Talk to current interns and ask them the following questions to find out about the inner workings and day-to-day activities of the internship program, and determine whether the site fits my needs:
   (a) **If possible, try and schedule time to meet with current interns in private**
   (b) Does the program offer what it claims?
   (c) What is a typical work day like?
   (d) How do the supervisors/faculty typically treat interns?
   (e) Do supervisors take active interest in your work?
   (f) What are the program's strengths and weaknesses?
   (g) Are the equipment and office space up to par?
   (h) What is the city like and how easily did you find housing?
   (i) Does the internship leave time for personal (or family) obligations?

5.4 Determine the Costs of the Ph.D. Program

Determine the costs of the Ph.D. program:

- academic costs (academic enrolment fees + cost of specific services)
- living costs
- medical costs: cost of medical/dental insurance, medical consultation, and medical treatment
- costs for memberships to professional organizations (such as IEEE and SIGDA):
  - Note: I am a lifetime professional member of ACM, and a lifetime subscriber to the ACM digital library
- cost of travel to conferences, programming contests, summer schools, symposiums, and workshops:
  - make sure that I apply for student travel grants
  - determine if I can register for conferences, symposiums, and workshops as a student
  - refund/reimbursement make take some time, and I may have to cover the costs initially
- cost of travel to institutions for (invited/job) talks:
  - refund/reimbursement make take some time, and I may have to cover the costs initially

5.5 Notes And Advice For Being A Graduate Student

Notes and advice for being a graduate student:

1. Don't treat my techniques as golden hammers. They can't save the world, and solve all of its problems.
2. Focus on my research and its originality. And, don't pick a mentor who bogs me down with too many courses, obligations, and prerequisites
3. Find what I am passionate about, and pursue it regardless of what others expect me to do. Don't try to be a clone of my mentor or anyone else, find my own unique path and passions
4. Once I identify my passion, and have a job consistent with it, work my butt off and produce, produce, produce.
5. Don't be bashful about what I have to offer as a research engineer/scientist – I've had great training.
6. Remember that I am more than my job; I don't live forever, so have some fun!
7. Find a mentor whose commitment is to my professional development, and who will help me achieve what I are interested in
8. Have a "cuddle group" – a group of friends, colleagues, and classmates that can support me and who I can talk to about the challenges of being an intern, postdoc, or new faculty member or research engineer/scientist
9. Get involved in the community – volunteer at a domestic violence shelter, or advocate for policies or laws that matter to me ... make a difference!
10. Always ask what is the main takeaway of this paper, report, thesis, or project.
11. Motivation: doing excellent research gives me a better chance to get a research position in industry, or get a tenure-track position in academia immediately after completing my Ph.D.; otherwise, I may have to settle for postdoc positions, and avoid being mired in postdoctoral research for too long (and the constant scramble for grant money) ... postdoc positions that require me to carry out too many duties can also be snares along the way
12. Establish a excellent research reputation (quality and quantity of publication record, and winning programming or IC design contests), and grant history to help me get a good research position after graduation
13. When I use LaTeX to typeset any document, such as my dissertation or grant proposal, store each chapter/section/subsection of the document in a new file. This allows me to easily typeset only a chapter, section, or subsection of the document, and disseminate that chapter, section, or subsection to others for review. When people look at an individual chapter, section, or subsection, they may be more wiling to pay greater attention to the details and give me feedback on the document. Use these feedback when I update (correct and extend) my document
14. Read my manuscript aloud to peers or have them read it to me – it's amazing what errors you'll catch
15. Attend as many research seminars and colloquiums/colloquia as I can
16. Determine who the authorities (compare with key opinion leaders / KOLs, or opinion leaders) are in my field, and what are its seminal publications (or seminal/classic papers). That is, who are the true leaders (experts) in my field? Which publications are highly influential in my field? Available online at: http://www.quora.com/How-do-I-find-the-seminal-papers-of-an-academic-field; last accessed on September 25, 2010.
17. Use Web 2.0 technologies, such as *MyEdu* (http://www.myedu.com/), to select my classes, plan my class timetable, check my progress towards graduation, and organize my calendar.
18. Put videos of my talks at http://videolectures.net/ or YouTube; e.g., I can put up videos of my talks at conferences, Quals, and my Ph.D. oral defense

19. Talk to rising seniors/juniors about research topics that fascinate them. I may get insights into research trajectories/topics that are relevant to my trajectory/topics, but am currently ignorant of. Do that for outstanding rising sophomores as well.
20. Do the heavy-lifting in my research: writing journal and conference papers, and software development.
21. Revise Q-Q plots, so that I can use Q-Q plots to compare SAT/SMT solvers and EDA tools with a given set of benchmarks. See http://en.wikipedia.org/wiki/Q-Q_plot.
22. Some things to pay attention to:
    (a) "**Autodidacticism** (also autodidactism) is self-directed learning that is related to but different from informal learning. In a sense, autodidacticism is 'learning on your own' or 'by yourself', and an autodidact is a self-teacher. Autodidacticism is a contemplative, absorptive procession. Some autodidacts spend a great deal of time reviewing the resources of libraries and educational websites. One may become an autodidact at nearly any point in one's life. While some may have been informed in a conventional manner in a particular field, they may choose to inform themselves in other, often unrelated areas." Reference: Wikipedia contributors, "Autodidacticism," in *Wikipedia, The Free Encyclopedia: Learning methods*, Wikimedia Foundation, San Francisco, CA, July 5, 2012. Available online at: http://en.wikipedia.org/wiki/Autodidacticism; last accessed on July 19, 2012.
        i. "Autodidact: a person who has learned a subject without the benefit of a teacher or formal education; a self-taught person." Reference: Dictionary.com, "Autodidact," IAC, Oakland, CA, July 18, 2012. Available online at: http://dictionary.reference.com/browse/autodidact; last accessed on July 19, 2012.

    (b) "**Intellectual need** is a specific form of intrinsic motivation; it is a desire to learn something. It has been recognized as critical in effective education and learning. Intellectual need arises when someone poses a question to themselves or others, either out of curiosity or to solve a specific problem. Intellectual need is often greatest when there is a hole in an otherwise well-connected web of knowledge. Merely understanding a question and being unable to answer it is not sufficient to create intellectual need–intellectual need arises when a person believes the question to be interesting or important, and usually this involves fitting the question into a framework of well-understood ideas." ... "Giving students a new technique to solve a problem will not be effective if the students are already able to solve the problem through other easier or more enjoyable techniques, because they will have no intellectual need for the new technique." Reference: Wikipedia contributors, "Intellectual need," in *Wikipedia, The Free Encyclopedia: Learning*, Wikimedia Foundation, San Francisco, CA, April 11, 2012. Available online at: http://en.wikipedia.org/wiki/Intellectual_need; last accessed on July 19, 2012.

    (c) "**Latent inhibition** is a technical term used in classical conditioning to refer to the observation that a familiar stimulus takes longer to acquire meaning (as a signal or conditioned stimulus) than a new stimulus." Reference: Wikipedia contributors, "Latent inhibition," in *Wikipedia, The Free Encyclopedia: Learning*, Wikimedia Foundation, San Francisco, CA, June 24, 2012. Available online at: http://en.wikipedia.org/wiki/Latent_inhibition; last accessed on July 19, 2012.

23. **Map out my career, and keep updating that map!!!**
24. **Make a list of research labs in my research area where I can intern or be a postdoc; keep the list updated!!!**
25. "Cognitive dissonance is the term used in modern psychology to describe the state of holding two or more conflicting cognitions (e.g., ideas, beliefs, values, emotional reactions) simultaneously."

Reference: Wikipedia contributors, "Cognitive dissonance," in *Wikipedia, The Free Encyclopedia: Social psychology*, Wikimedia Foundation, San Francisco, CA, September 23, 2012. Available online at: http://en.wikipedia.org/wiki/Cognitive_dissonance; last accessed on October 1, 2012.

26. When helping others, beware of the "Dunning-Kruger effect"; see http://en.wikipedia.org/wiki/Dunning-Kruger_effect. Ditto for cranks, crackpots, and kooks. Be aware of self-serving bias and group-serving bias.
    (a) "The Dunning–Kruger effect is a cognitive bias in which unskilled individuals suffer from illusory superiority, mistakenly rating their ability much higher than average. This bias is attributed to a metacognitive inability of the unskilled to recognize their mistakes."
    (b) Kruger and Dunning proposed that, for a given skill, incompetent people will:
        i. tend to overestimate their own level of skill;
        ii. fail to recognize genuine skill in others;
        iii. fail to recognize the extremity of their inadequacy;
        iv. recognize and acknowledge their own previous lack of skill, if they are exposed to training for that skill
    (c) crank:
        i. "An ill-tempered, grouchy person."
        ii. "An unbalanced person who is overzealous in the advocacy of a private cause."
        iii. "An eccentric or odd person, esp someone who stubbornly maintains unusual views"
        iv. Reference: Dictionary.com, "crank," IAC, Oakland, CA, 2012. Available online at: http://dictionary.reference.com/browse/crank?s=t&ld=1087; last accessed on July 25, 2012.
    (d) crackpot:
        i. "A person who is eccentric, unrealistic, or fanatical."
        ii. Reference: Dictionary.com, "crackpot," IAC, Oakland, CA, 2012. Available online at: http://dictionary.reference.com/browse/crackpot?s=t; last accessed on July 25, 2012.
    (e) kook:
        i. "An eccentric, strange, or foolish person."
        ii. "An insane person."
        iii. Reference: Dictionary.com, "kook," IAC, Oakland, CA, 2012. Available online at: http://dictionary.reference.com/browse/kook?s=t; last accessed on July 25, 2012.
    (f) Reference: Wikipedia contributors, "DunningKruger effect," in *Wikipedia, The Free Encyclopedia: Social psychology*, Wikimedia Foundation, San Francisco, CA, July 20, 2012. Available online at: http://en.wikipedia.org/wiki/DunningKruger_effect; last accessed on July 25, 2012.
27. Group-serving bias:
    (a) "Group-serving bias is identical to self-serving bias except that it takes place between groups rather than individuals, under which group members make dispositional attributions for their group's successes and situational attributions for group failures, and vice versa for outsider groups."
    (b) Reference: Wikipedia contributors, "Group-serving bias," in *Wikipedia, The Free Encyclopedia: Cognitive biases*, Wikimedia Foundation, San Francisco, CA, July 7, 2012.
28. lemming:
    (a) "Any member of a group given to conformity or groupthink, especially a group poised to follow a leader off a cliff."

   (b) Reference: Wiktionary contributors, "Lemming," Wiktionary, Wikimedia Foundation, San Francisco, CA, August 29, 2012. Available online at: http://en.wiktionary.org/wiki/lemming; last accessed on October 2, 2012.

29. Beware of the reality distortion field (RDF):
    (a) "The RDF was said by Andy Hertzfeld to be Steve Jobs' ability to convince himself and others to believe almost anything with a mix of charm, charisma, bravado, hyperbole, marketing, appeasement and persistence. RDF was said to distort an audience's sense of proportion and scales of difficulties and made them believe that the task at hand was possible. While RDF has been criticized as anti-reality, those close to Jobs have also illustrated numerous instances in which creating the sense that the seemingly impossible was possible led to the impossible being accomplished (thereby proving that it had not been impossible after all). Similarly, the optimism which Jobs sowed in those around him contributed to the loyalty of his colleagues and fans."
    (b) Reference: Wikipedia contributors, "Reality distortion field," in *Wikipedia, The Free Encyclopedia: Steve Jobs*, Wikimedia Foundation, San Francisco, CA, July 14, 2012. Available online at: http://en.wikipedia.org/wiki/Reality_distortion_field; last accessed on July 18, 2012.

30. Rube Goldberg machine, contraption, invention, device, or apparatus:
    (a) "A Rube Goldberg machine, contraption, invention, device, or apparatus is a deliberately over-engineered or overdone machine that performs a very simple task in a very complex fashion, usually including a chain reaction."
    (b) "A comically involved, complicated invention, laboriously contrived to perform a simple operation"
    (c) References:
        i. Wikipedia contributors, "Rube Goldberg machine," in *Wikipedia, The Free Encyclopedia: Mechanisms*, Wikimedia Foundation, San Francisco, CA, July 17, 2012. Available online at: http://en.wikipedia.org/wiki/Rube_Goldberg_machine; last accessed on July 26, 2012.
        ii. Rube Goldberg Inc., *The Official Rube Goldberg Website*, Rube Goldberg Inc., Westport, CT, 2012. Available online at: http://www.rubegoldberg.com/; last accessed on July 26, 2012.

31. Self-efficacy:
    (a) "Self-efficacy is the term used to describe how one judges one's own competence to complete tasks and reach goals."
    (b) Wikipedia contributors, "Self-efficacy," in *Wikipedia, The Free Encyclopedia: Positive psychology*, Wikimedia Foundation, San Francisco, CA, July 9, 2012. Available online at: http://en.wikipedia.org/wiki/Self-efficacy; last accessed on July 27, 2012.

32. Self-serving bias:
    (a) "A self-serving bias, sometimes called a self-serving attributional bias, refers to individuals attributing their successes to internal or personal factors but attributing their failures to external or situational factors. This bias is a mechanism for individuals to protect or enhance their own self-esteem."
    (b) Reference: Wikipedia contributors, "Self-serving bias," in *Wikipedia, The Free Encyclopedia: Cognitive biases*, Wikimedia Foundation, San Francisco, CA, July 18, 2012. Available online at: http://en.wikipedia.org/wiki/Self-serving_bias; last accessed on July 25, 2012.

33. Somebody Else's Problem (also known as Someone Else's Problem or SEP):

   (a) "Somebody Else's Problem is a condition where individuals/populations of individuals choose to decentralize themselves from an issue that may be in critical need of recognition. Such issues may be of large concern to the population as a whole but can easily be a choice of ignorance at an individualistic level."
   (b) Reference: Wikipedia contributors, "Somebody Else's Problem," in *Wikipedia, The Free Encyclopedia: Group processes*, Wikimedia Foundation, San Francisco, CA, July 12, 2012. Available online at: `http://en.wikipedia.org/wiki/Somebody_Else%27s_Problem`; last accessed on July 25, 2012.

34. survivorship bias:
   (a) Wikipedia contributors, "Survivorship bias," in *Wikipedia, The Free Encyclopedia: Sampling (statistics)*, Wikimedia Foundation, San Francisco, CA, September 4, 2012. Available online at: `http://en.wikipedia.org/wiki/Survivorship_bias`; last accessed on September 14, 2012.

35. Test my EDA tools with benchmark circuits, and use them to design VLSI circuits and systems. Send the designed VLSI circuits and systems to the following for manufacturing:
   (a) MOSIS (Metal Oxide Semiconductor Implementation Service, MOSIS Integrated Circuit Fabrication Service): `http://www.mosis.com/`
   (b) Circuits Multi-Projets® (or Multi-Project Circuits®), CMP: `http://cmp.imag.fr/`
   (c) IMEC: `http://www2.imec.be/be_en/home.html`

36. Information specific to U.S.-based graduate students:
   (a)

37. National Postdoctoral Association:
   (a) Publications & Resources — RCR for Postdocs:
      i. Peer review: `http://www.nationalpostdoc.org/publications/rcr/116-rcr-toolkit-peer-`

38. "If you're proposing a new statistical method, you should absolutely post code on the web for carrying out the technique. This doesn't have to be commercial-level code, and could even be production code with limited comments, but it should be enough that a knowledgeable statistician and programmer can figure out what you did, including all the detailed steps." – Prof. Christopher Paciorek; see `http://www.biostat.harvard.edu/~paciorek/research/webPhilosophy/webPhilosophy.html`, last viewed on June 25, 2010.

---

39. **Andrew Kahng**
40. "Andrew Kahng–Crafting the Future of EDA," By Peggy Aycinena, in DACeZine, Volume 4, Issue 2, Oct 16, 2008:
   (a) `http://www.dac.com/newsletter/shownewsletter.aspx?newsid=53`
   (b) Initiatives such as OpenAccess, plus the fact that core synthesis, timing, global placement, sizing, etc. technologies are getting long in the tooth, mean that many genies are out of the bottle and easier to match or improve on. So, new EDA companies can become serious players very fast – for instance, Atoptech in the back end – and it is easier than ever for customers to switch vendors in and out.
   (c) With so much focus on polishing or re-implementing the current scope of "cash cow" subflows, not only does the EDA market fail to grow in system-level or die-package or embedded software or analog/mixed-signal design areas, but existing technologies become ever more commoditized and interchangeable.

(d) The big vendors have not figured out how to grab that big brass ring of royalty-based revenue – although a couple of smaller vendors have – so EDA remains a rental (and, service) business.

(e) If were going to solve the ESL, embedded software, multicore, analog, DFM, etc. challenges, we have to move beyond yet another SPICE parser, yet another shapes database, yet another back-end data model, and so on. Interoperability has proven impossible to incent, so perhaps customers must collectively create their own EDA development entity that generates and supports production-worthy infrastructure, to which EDA vendors can plug in or snap on.

(f) Will EDA companies ever improve their understanding of electronic system design, the chip design process, and software quality, so as to merit "partner" rather than "supplier" standing in the semiconductor supply chain?

(g) **"It's an old rant of mine is that tools shouldn't just be gigantic Swiss army knives. They must actively comprehend and define methodologies."**

(h) most exciting growth areas going forward for young researchers in EDA:
   i. "Multicore! System architectures, how to design them, how to program them, etc. These issues will continue to bring challenges for many, many years to come. And, any list of buzz-phrases has to include system-level optimization, enablement of embedded software design, and verification."
   ii. "Plus, we still require a science of design up through the system scale. EDA has been an incredible mélange of device physics, graph algorithms, heuristic optimization, computer and network architecture, etc. Its culture is largely one of incremental improvement of point optimizations, without much global design context. We need better frameworks and we need better experimental and evaluation methodologies!"
   iii. non-traditional technologies for computing still demand research on feasibility – which requires design and optimization – as well as scalable manufacturing

(i) advice for new grad students:
   i. My generic advice to students is to de-specialize! To be creative and aware, to learn physics, chemistry or biology – and, optimization – to learn how to program parallel and heterogeneous systems, and to always understand the context for what you do, and why you do it.
   ii. If I were going to grad school today – beyond depth in some research focus, I would seek breadth, breadth, and more breadth – not just in CAD and VLSI, but allied areas as well.
   iii. The world will always need better and broader connections between the science or art of design and the underlying technology options and application needs. Hugo De Man, in his memorable DAC 2000 keynote address, explained to us what a 21st-century design engineer should be able to do. Reference: Hugo De Man, "System Design Challenges in the Post-PC Era," Keynote address, 37^{th} ACM/IEEE Design Automation Conference, Los Angeles, June 2000.
   iv. Around that time, my friend Ranko Scepanovic was telling me how he was working on "from C to OPC." It would have been great to have perfected such a toolkit during graduate school rather than piecing it together afterward.
   v. Of course, all graduate students need to work on their abilities to: 1) see big pictures and their choke points; 2) create practically relevant abstractions and problem formulations; and 3) innovate and validate impactful solutions to these problems.

(j) "Given the massive changes in the geo-political and geo-economic landscape, do you see the geographic focus for cutting-edge CAD tool work remaining here in North America for the foreseeable future? If not, where has it moved to, or will it move to?" – Peggy Aycinena:
   i. "I think we will inevitably see multiple foci, colored by regional investment: the EU and system-level design; migration of semiconductor production to Asia; migration of chip

implementation services to India and China; etc. Well also see other factors at work, such as the macro trends and crossroads we were discussing earlier." – Andrew Kahng

ii. "One example of a non-U.S. focal point: Taiwan. Taiwan schools are very strong in the CADathlon held at ICCAD each year. They win the placement and global routing contests at ISPD. And, they can be expected to do very strong research in areas such as DFM. On the commercial side, we see the rapidly increasing visibility of SpringSoft." – Andrew Kahng

iii. "I believe, and I hope, that the U.S. university system and the continued attractiveness of North American schools to students from China, Korea, etc. will help keep the largest focal point here for a long time. But as you and I once discussed in an earlier conversation about 'brain drains,' I think we are seeing that flows of engineering talent in general, and CAD tool innovation specifically, are starting to move away from the U.S." – Andrew Kahng

(k) The world is evolving so quickly in so many ways. We see an unprecedented global economic context, consolidation of technologies and businesses, emergence of new product classes and markets, massive shifts toward cleantech and health/bio – kind of a hurricane of macrotrends. But at the eye of this hurricane, more than ever before, our economic and societal health depends on technology agility and a continuing stream of innovative semiconductor-based products that bring compelling benefit to mankind (and, compelling value to consumers). To this end, I believe that EDA serves as a nexus that joins the product markets and driving applications, the designers, the semiconductor and packaging technologies, and the mathematical, engineering and algorithmic foundations of IC CAD. And in this context, we need DAC to be a heartbeat not only for the EDA industry and its users, but for the broader design chain as well.

---

41. **Francesco Stefanni**
42. Advice from Francesco Stefanni, a CS Ph.D. student in the Electronic System Design (ESD) research group that I am part of at the University of Verona:
    (a) [February 22, 2011 / Tuesday] Use my travel budget for major conferences, rather than for workshops and low(er)-quality conferences as well as Ph.D. forums. That is, use my travel budget wisely . . . This is because a good track record of research publications, including high-quality conference papers, is a large factor of what would help me get a good research position rather than merely networking with people from the industry and academic in these forums.

---

43. **Henry Petroski**
44. Reference: Henry Petroski, "Welcoming Summer: For postgraduate researchers, a season of challenge," in *Prism: Refractions*, American Society for Engineering Education, Washington, D.C., Summer, 2012. Available online at: http://www.prism-magazine.org/summer12/refractions.cfm; last accessed on July 27, 2012.
    (a) "**For me, one of the true joys of academic life has always been the change of pace that summer brings.** It has not been, as it is in Porgy and Bess, that the living is easy. Indeed, **there has been many a summertime when I worked longer hours on harder problems but without the comic relief of faculty meetings**."
    (b) "When I was in graduate school in Urbana, Illinois, where I had my first real taste of academic freedom, summers initially meant studying foreign languages to satisfy requirements that are

now as rare as prices in textbook catalogs. Whereas during the fall and spring semesters I took all of my classes on the engineering end of campus, during the summer I trekked beyond the student union to where French and German were taught – in buildings that housed not laboratories but theaters."

(c) "For those of us teaching assistants who were fortunate enough to pass the translation tests, summers next came to mean concentrated work on a research project. **It was here that I first learned what it meant to live and breathe a problem daily – and dream of its solution nightly – without the benefit of an answer key, instructors manual, or expert to consult. Now we students were expected to be the experts to whom others would come regarding the slice of engineering science that we had carved out for ourselves."**

(d) **"I lived and breathed a problem by day, dreamt a solution by night."**

(e) **"The loneliness of the long-distance runner has been written about, but little has been said about the loneliness of the long-suffering doctoral student. Day after day and night after night, we worked away on problems whose solution seemed then even more elusive than a balanced checkbook. Summer was a time to make progress on a problem that would define us, at least for a while, but progress came slowly and sullenly."**

(f) **"With the end of my research in sight, summer came to mean the time to write up the dissertation and also write up papers based on it. I had a sense of urgency to submit my first paper before someone else might submit something similar enough to make all of my work null and void. After all, what I was working on was pushing the frontier of my field, something other graduate students were doing also. And the field was somewhat crowded."**

(g) "**Becoming young faculty members brought new summer pressures, at first mostly centered on writing proposals for research extending our dissertation work. Soon, we were advised to break away from that and come up with fresh ideas.** When we were fortunate enough to get some of those funded, summer was often the time of year when the serious research promised in the proposals had to be done."

(h) **"Summer also is, or should be, a time for reflection. Where have we been and where are we going, not only with our research but also with our teaching? Summer is a time to take stock and, on the eve of the new academic year, to make resolutions."**

(i) "Nonacademic friends and relatives envy us our 'summers off,' but that's probably because they do not appreciate how busy our summers actually are. And they are short, as we realize in early August, when the new semester looms like a massive homework assignment. There are syllabi to organize and lectures to prepare. Little time is left for doing research or writing up results. That has to wait for next summer."

---

45. **Sachin Sapatnekar**
46. "DACeZine Profile: Sachin Sapatnekar," by Peggy Aycinena, in DACeZine, Volume 4, Issue 3, Dec. 3, 2008:
    (a) `http://www.dac.com/newsletter/shownewsletter.aspx?newsid=64`
    (b) For an engineer, the basic skill that needs to be mastered is problem solving. Take a physical problem, abstract it as a mathematical model, solve the abstraction, and map it back to reality. A really good abstraction achieves the necessary balance between the realism of the model and the tractability of the abstracted problem. Of course, this isnt just an EDA

concept – its true of most areas of engineering, and this is what I urge my students to learn. The rest of what should be considered revolves around the technology itself – something thats critically important to comprehend, but is ever changing. I emphasize to my students that 10 years from today, they will be working on a completely different area than their current research, and they must learn the skills that make them adaptable.

(c) Ive found the students who do work in this area to be highly motivated. EDA has always been a rich source of solution techniques that have made a broader impact – think BDDs, verification, model-order reduction, and so on. Our problems are large in size, challenging in nature, and require great ingenuity to solve. This isnt going to change anytime soon!

---

47. **Sara Vinco**
48. Advice from Sara Vinco, a CS Ph.D. student in the Electronic System Design (ESD) research group that I am part of at the University of Verona:
    (a) [January 20, 2011 / Thursday] When preparing summaries for research publications that Prof. Franco Fummi has asked me to do, do the following:
        i. Summarize the research paper in 1-2 sentence(s).
        ii. Next, write a short paragraph to summarize the paper. In this short paragraph, talk about what the paper, thesis, or report is about. In addition, highlight the things that are important about this paper.
        iii. Sara has written a state of the art summary of at least 70 pages.

---

49. **W. Robert Daasch**
50. [From http://web.cecs.pdx.edu/~daasch/general.cgi?1+2+3#1, by Prof. W. Robert Daasch] A helpful reality check to track and evaluate the progress of any research project are four practical questions. Each question demands an answer sooner rather than later. To a limited extent these questions address the age-old (and generally unhelpful) discussion about a research effort being pure research or applied research. If the answer to any of the questions goes much beyond 100 words it is likely time to stop and think rather than do.:
    (a) What is the problem to be solved?
    (b) What are the system observables?
    (c) Do the current research results describe the past behavior or predict the future behavior of the system?
    (d) What is the difference between making progress and going in circles?

5.6 Notes and Advice For Grant Application

Notes and advice for grant application:

1. learn how to write grants, which is an essential skill for researchers
2. the sooner I learn the vital craft of grant-writing, the better off I'll be: Grant money can help finance my education and, indeed, all of my future research
3. grant-writing is integral to graduate students who plan to go into academia when they finish school, but no one teaches you how to do it
4. advice for grant applications:
    (a) start early: turn my research proposal/plan into a grant application
    (b) get educated about the grant-writing process, and seek guidance from others

(c) learn about where can I get grants, how can I learn which grants are available, how difficult or easy are the grants to get, and what the grants will pay for

(d) Involve others:
   i. find professors in my area who have written successful grants and garner their advice
   ii. get mentors, related experts, and fellow students to read drafts as I proceed
   iii. don't hesitate to go outside your department

(e) Follow your interests:
   i. while I might be tempted to seek research dollars in a trendy funding area, stick with my most basic loves
   ii. the topics that are of genuine interest to me will be around a lot longer than the research fads of the moment
   iii. don't be shy:
      A. get advice from my mentor and others on the best people and institutes to contact
      B. don't hesitate to ask them for information and advice
      C. the one exception is that I can't contact reviewers when my application is under review
   iv. follow the rules, such as formatting guidelines
   v. take the advice:
      A. if my application is returned with suggestions to revise it, follow reviewers' advice
      B. my reviewers will highlight strengths and areas for improvement, so when I resubmit my grant, make sure the areas for improvement have become strengths
      C. that doesn't mean I have to agree with everything the reviewers say, but I must clearly articulate in my revision why I disagree
   vi. market my plan:
      A. make my message very clear in a way that helps them learn why my project is so important
      B. at the same time, don't brag or otherwise sell myself too aggressively
      C. I want to present myself as a thoughtful, serious fellow scientist
   vii. be realistic:
      A. reviewers frequently critique young researchers' proposals for being overly ambitious
      B. my research plan should be conceptually tight, be well-linked to the literature, have clearly articulated hypotheses, and allow me to answer the questions the research hypotheses raise
      C. my proposal should be airtight, anticipating any questions the reviewer might have about decisions I made in designing my study
      D. reviewers are looking not just for the quality of research, but also at the quality of the applicant – the way I think and the way I work
   viii. persist:
      A. the advice "Never give up; never surrender!" applies to every phase of the grant application process, whether it's enlisting others' help, crafting my grant, or revising my proposal.
      B. do not be discouraged if my application is sent back for a rewrite: It is extremely common for grants to get accepted on the second or third round
      C. many applicants who apply for F31s research grants, and have to revise them end up securing them
      D. if I plan to garner research funds as my career goes on, I'll likely be doing plenty of rewriting and resubmitting

5. resources for writing grant proposals and applying for grants:
   (a) Alfred P. Sloan Foundation: `http://www.sloan.org/apply/page/6`
   (b) American Psychological Association: *gradPSYCH* Magazine:

   i. go to NIH's web page to view the section on writing a topnotch grant application; i.e., see http://www.apa.org/gradpsych/2004/04/grant-resources.aspx

(c) *American Association for the Advancement of Science, AAAS*: http://sciencecareers.sciencemag.org/tools_tips/how_to_series/how_to_get_funding and http://sciencecareers.sciencemag.org/funding

(d) Catalyst Foundation:
   i. http://www.catalyst-foundation.org/call-for-proposals/
   ii. Funds interdisciplinary research, especially those related to analog/RF and mixed-signal ICs

(e) The Mathematical Association of America, *Tips for Writing Successful Grant Proposals*, The Mathematical Association of America. Available at: http://www.maa.org/programs/grantwriting/; last accessed on September 2, 2010.

(f) European Commission:
   i. http://cordis.europa.eu/fp7/home_en.html
   ii. Find calls at:
      A. http://cordis.europa.eu/fp7/dc/index.cfm?fuseaction=UserSite.FP7CallsPage
      B. http://cordis.europa.eu/fp7/dc/index.cfm
      C. Information and Communication Technologies (ICT): Information Society Technologies (IST), Future and Emerging Technologies (FET):
         - http://cordis.europa.eu/fp7/dc/index.cfm?fuseaction=UserSite.FP7ActivityCal id_activity=3
         - http://cordis.europa.eu/fp7/ict/participating/calls_en.html
         - http://cordis.europa.eu/fp7/ict/home_en.html
         - **EU's Seventh Framework Programme (FP7): ICT Program**, http://cordis.europa.eu/fp7/ict/
         - **Future and Emerging Technologies (FET) Young explorers** (for postdocs who have completed their Ph.D.s no more than 6 years ago): http://cordis.europa.eu/fp7/ict/fet-open/ye_en.html

(g) *Research*:
   i. http://www.researchresearch.com/ [Link to this page from University of Gothenburg: http://www.gu.se/english/research/scholarships/ResearchResearch/]
   ii. Provides information on funding opportunities, as well as a list of job vacancies (research positions)

(h) UNESCO: http://www.unesco.org/en/venice

(i) University of Wisconsin-Madison:
   i. Grant Proposal Writing: http://researchguides.library.wisc.edu/content.php?pid=16143&sid=108666

(j) Montana State University, "Some Reasons Proposals Fail," Office of Sponsored Programs, Montana State University. Available at: http://www.montana.edu/wwwvr/osp/reasons.html; last accessed on September 2, 2010.

(k) OpenContent Wiki, "Grant Writing and Project Management 2010". Available at: http://opencontent.org/wiki/index.php?title=Grant_Writing_and_Project_Management_2010; last accessed on September 4, 2010. [Also available as: David Wiley, "IP&T 682, Grant Writing and Project Management (Instructional Psychology and Technology 682)," Department of Instructional Psychology and Technology, David O. McKay School of Education, Brigham Young University.

(l) Semiconductor Research Corporation – (anticipated) Calls for GRC White Papers: http://www.src.org/compete/

(m) CATRENE (Cluster for Application and Technology Research in Europe on NanoElectronics):

i. Project Calls – CATRENE Calls for Project Proposals: http://www.catrene.org/web/calls/local_index.php
ii. Resources and guidelines for project proposals: http://www.catrene.org/web/calls/prepare_pp.php

(n) Association for Computing Machinery (ACM):
i. Sanjeev Arora, Boaz Barak, and Luca Trevisan, "Funding Opportunities and Tips," in *Theory Matters Wiki: Theoretical Computer Science (TCS) Advocacy Wiki*, SIGACT Committee for the Advancement of Theoretical Computer Science, ACM Special Interest Group on Algorithms and Computation Theory (SIGACT), Association for Computing Machinery, February 25, 2010. Available at: http://theorymatters.org/pmwiki/pmwiki.php?n=Main.FundingOpportunities; last accessed on September 15, 2010.

(o) The Meadows Foundation:
i. http://www.mfi.org/
ii. Provides grants for educational institutions: http://www.mfi.org/display.asp?link=Z0WR3A

(p) Broadcom Foundation:
i. http://www.broadcomfoundation.org/
ii. Funding areas: http://www.broadcomfoundation.org/about/#funding_areas
iii. Can seeking funding for research projects to support research in science, technology, engineering, and mathematics (STEM), particularly for underrepresented minorities.
iv. Geographic Focus: http://www.broadcomfoundation.org/about/#geographic_focus
v. Funding Guidelines: http://www.broadcomfoundation.org/apply/#funding_guidelines

(q) NYSTAR, New York State Foundation for Science, Technology and Innovation:
i. NYSTAR Funding Initiatives: http://www.nystar.state.ny.us/initiatives.htm

(r) The European Mathematical Society:
i. European funding:
A. Seventh Research Framework Programme: http://www.euro-math-soc.eu/eu_funding.html

(s) European Science Foundation:
i. http://www.esf.org/
ii. May be merging funding resources with European Heads of Research Councils (EUROHORCs) and ECRP (European collaborative research programme). Chloe Kembery, "European Science Foundation cancels research funding calls," in *European Science Foundation: Media Centre: Press Releases*, November 25, 2010. Available online at: http://www.esf.org/media-centre/press-releases/ext-single-news.html?tx_ttnews[tt_news]=636&cHash=5e6d50f55aff7ec5f4b9853f08528de1; last accessed on December 13, 2010.
iii. European Science Foundation. *Grants, Calls, and Applications.* Available at: http://www.esf.org/research-areas/physical-and-engineering-sciences/grants-calls-and-ap
html; last accessed on September 2, 2010.
iv. Calls and Funding:
A. http://www.esf.org/activities/calls-and-funding.html
B. Latest calls: http://www.esf.org/activities/calls-and-funding/latest-calls.html

(t) European Research Council (ERC):
i. http://erc.europa.eu/
ii. Grants:

- http://erc.europa.eu/index.cfm?fuseaction=page.overView&topicID=118
- How to Get Funding: http://erc.europa.eu/index.cfm?fuseaction=page.display&topicID=498
- Calls for proposals: http://erc.europa.eu/index.cfm?fuseaction=page.display&topicID=497
- ERC Starting Independent Researcher Grant: http://erc.europa.eu/index.cfm?fuseaction=page.display&topicID=65

iii. Useful Information on ERC Funding: http://erc.europa.eu/index.cfm?fuseaction=page.display&topicID=23
iv. Funding Strategy and Methodology: http://erc.europa.eu/index.cfm?fuseaction=page.display&topicID=24

(u) Wellcome Trust:
i. Funding (for activities, including research, that are related to medical science and technologies): http://www.wellcome.ac.uk/Funding/index.htm
ii. Managing a grant: http://www.wellcome.ac.uk/Managing-a-grant/index.htm

6. It's no trivial item to say in my first job interview, "I already have experience bringing in research dollars"
7.

5.7 Advice for Teaching Assistants (TAs)

Advice for graduate teaching assistants (GTAs), or simply TAs:

1. Prepare for the classes with ample time to spare; prepare early, and know my stuff well ... if I don't know my stuff well, my students can/will assume that I do not know what I'm doing, and they'll not pay attention to the discussion session that I am leading ... think of the poor, inept TAs that I had
2. Talk to TAs and faculty who have taught the same course that I will teach, and ask them for copies of their syllabi
3. Sit in on the class that I'm going to teach
4. Find out which faculty in the department are known for being good teachers, and ask them if I can watch them teach
5. Attend conference workshops and seminars, and my department's formal GTA programs ... such as a three-day teaching orientation before the school year begins
6. Observe other GTAs teaching
7. Know my job requirements, and expectations from the course instructor and/or fellow TAs ... For example, do I have to come to class occasionally and record grades? Do I have to grade a hundred term papers and answer students' e-mails?How much time per week am I expected to work?
8. Structure weekly or biweekly meetings where I can talk with the professor about student considerations, and how I think the class is responding to the teacher overall
9. Budget time into my schedule to complete my teaching tasks
10. New TAs often underestimate how long it takes to design and grade tests, or answer student questions
11. Develop an organization system that works for me and stick with it: use LaTeXto typeset (lecture/study/class) notes, presentation slides, assignments (homework handouts), project descriptions, and examinations.
12. Pay attention to student epistemologies:

- (a) Some students (such as myself) learn just enough material and practice problems repetitively to get by in their engineering degree program without having a sound understanding of the fundamentals in science (including computer science) and mathematics.
- (b) That is, students may favor descriptive knowledge (also known as declarative knowledge or propositional knowledge) without having a deep understanding of the fundamentals in science (including CS) and mathematics.
- (c) Hence, I need to develop ways to help students (including myself) to acquire a deep understanding of the fundamentals in science (including CS) and mathematics.

13. For online resources, see bullet point "Advice on teaching" in §7.5

5.8 Notes on EDA Software Development

Notes on EDA software development:

1. **Software Environment**
2. do not spend too much time setting up the software development environment:
   - (a) if so-called "better" software engineering tools for build automation, configuration management, documentation generation, and automated regression testing take a long time to set up, use substitute tools that are easier to learn and use
   - (b) When I have free time later, I can learn how to use them and integrate them into my software development environment
   - (c) See §??. **Seriously, don't bother with all the fancy stuff until I have a significant code base.**
   - (d) **Remember what Alicia Lowery said. "Don't spend so much time automating stuff, such that you hardly have time for what you're supposed to do"**
3. do not spend too much time on setting up experimental environments to the detriment of spending time on innovating:
   - (a) focus on advancing state-of-the-art in evaluation methodologies to off-load ourselves to be more innovative
4. avoid vendor lock-in: use free and open-source software as far as possible; and choose proprietary software that uses open standards and file formats, rather than those that don't

---

5. **Computer-Aided Software Engineering (CASE) Tools**
6. configuration management tools
7. data modeling tools
8. model transformation tools
9. refactoring tools/browsers:
   - (a) **Learn to use refactoring tools!!!**
   - (b) techniques that allow for more abstraction:
     - i. Encapsulate Field (field encapsulation, or data hiding) – force code to access the field with getter and setter methods
     - ii. Generalize Type (type generalization) – create more general types to allow for more code sharing
     - iii. Replace type – checking code with State/Strategy
     - iv. Replace conditional with polymorphism
   - (c) techniques for breaking code apart into more logical pieces:

i. Extract Method, to turn part of a larger method into a new method. By breaking down code in smaller pieces, it is more easily understandable. This is also applicable to functions.
ii. Extract Class moves part of the code from an existing class into a new class.

(d) techniques for improving names and location of code:
   i. Move Method or Move Field – move to a more appropriate Class or source file
   ii. Rename Method or Rename Field - changing the name into a new one that better reveals its purpose
   iii. Pull Up – in OOP, move to a superclass
   iv. Push Down – in OOP, move to a subclass
   v. Shotgun surgery – in OOP, when a single change affects many classes. In this case, all affected code is kept in a single class, so that the change is made only in one place.

(e) avoid “cut and paste coding”

(f) **beware of technical debt (or design debt or code debt)** :
   i. building up technical/design debt by not refactoring regularly would eventually force a rewrite of the software, which would consume much more time than regularly refactoring the software
   ii. as technical/design debt accumulates, the software architecture tends toward a big ball of mud (no distinguishable software architecture); regular refactoring of the code can avoid this
   iii. avoid these:
      A. spaghetti code: source code that has a complex and tangled control structure, especially one using many GOTOs, exceptions, threads, or other “unstructured” branching constructs
      B. ravioli code: a type of computer program structure, characterized by a number of small and (ideally) loosely-coupled software components
      C. lasagna code: a type of program structure, characterized by several well-defined and separable layers, where each layer of code accesses services in the layers below through well-defined interfaces

10. source code generation tools
11. Unified Modeling Language modeling tools:
   (a) **Find an appropriate UML tool to use for my software projects!!!** :
      i. `ArgoUML`
      ii. `Eclipse UML 2 Tools`
      iii. `BoUML`
      iv. `Dia`
      v. `Acceleo`
      vi. `Umbrello UML Modeller`
      vii. `TopCased`
   (b) alternative modeling languages and tools:
      i. flowchart
12. 3 categories of CASE tools:
   (a) tools support only specific tasks in the software process
   (b) workbenches support only one or a few activities
   (c) environments support (a large part of) the software process
   (d) *workbenches and environments are generally built as collections of tools*
13. see `http://www.tigris.org/servlets/ProjectList` for a list of open source software engineering tools

14. **Integrated Development Environments (IDEs)**
15. use integrated development environments (IDEs) for:
    (a) source code editing:
        i. `Aquamacs Emacs` and `GNU Emacs/XEmacs` serve as decent editors for `SPICE` with syntax highlighting
    (b) autocomplete
    (c) compiler and/or interpreter:
        i. `javac`
        ii. `GCC`
        iii. `Intel` provides a free non-commercial version of their `C++` compiler for `Linux` which gets 20% better performance on numerically intensive code than gcc by using MMX and SSE instructions; see http://software.intel.com/en-us/intel-sdp-home/ ... Old link: http://developer.intel.com/software/products/compilers/clin/noncom.htm
        iv. use `distcc` to speed up compilation of source code by using distributed computing over a computer network. With the right configuration, `distcc` can dramatically reduce a project's compilation time. This works for `C`, and `C++`
    (d) build automation tools:
        i. `make`
        ii. the "makedepend" feature in `GNU Make` allows some source code dependency management, and incremental build processing
        iii. **Familiarize myself with the GNU build system (also known as Autotools)**
        iv. `CMake` is a decent substitute for `make`:
            A. http://www.cmake.org/
            B. Also, use *pkg-config* with this. See http://pkg-config.freedesktop.org/wiki/.
        v. *Waf*
        vi. *SCons*
        vii. *Boost.Build*
    (e) configuration management (CM):
        i. `Mercurial`
        ii. `Git`
        iii. `Darcs`: distributed revision control system **Try this with Trac**
        iv. `Concurrent Versions System (CVS)`
        v. `Subversion (SVN)`; see http://www.centerstageproject.com/wiki/index.php/svn_tutorial/contents.html
        vi. `GNU arch`
        vii. `Bazaar`; see http://bazaar-vcs.org/
        viii. `Fossil`: http://en.wikipedia.org/wiki/Fossil_(software)
        ix. `Codeville`: http://en.wikipedia.org/wiki/Codeville
        x. `ArX`: http://en.wikipedia.org/wiki/ArX_(revision_control)
        xi. `SVK`: http://en.wikipedia.org/wiki/SVK
        xii. check out "open source software hosting facilities":
            A. see http://en.wikipedia.org/wiki/Comparison_of_free_software_hosting_facilitie
            B. `Google Code`; see http://code.google.com/
            C. `GitHub`; see http://github.com/
            D. `SourceForge`; see http://sourceforge.net/
            E. `GNU Savannah`; see http://savannah.gnu.org/
            F. `Tigris.org`; see http://www.tigris.org/
        xiii. organize the collection of source code files into a directory tree (**source tree**)
        xiv. CM tool is used to manage the **code base**
        xv. carry out `continuous integration` to reduce integration times; that is, by reducing the changes/differences between the repository code and the developer's copy (of the repository), the amount of work required by the developer prior to the submission of

their own changes is smaller; hence, continuous integration is used to avoid "integration hell," where the aforementioned difference is huge; try, "nightly builds," or submit a revision/build every couple of hours (prior to lunch/dinner/sleep); **Automated Continuous Integration**

xvi. advantages of continuous integration:
   A. when unit tests fail, or a bug is discovered, developers might revert the codebase back to a bug-free state, without wasting time debugging
   B. integration problems are detected and fixed continuously – no last minute hiatus before release dates
   C. early warning of broken/incompatible code
   D. early warning of conflicting changes
   E. immediate unit testing of all changes
   F. constant availability of a "current" build for testing, demo, or release purposes
   G. the immediate impact of checking in incomplete or broken code acts as an incentive to developers to learn to work more incrementally with shorter feedback cycles

xvii. use automated regression testing and test-driven development to avoid problems with configuration management

xviii. tools to facilitate continuous integration:
   A. see http://en.wikipedia.org/wiki/Continuous_integration

(f) debugging:
   i. `GDB: The GNU Project Debugger`

(g) GUI design and development

(h) check the code coverage

(i) static code analysis

(j) class browsing

(k) performance analysis (or profiling)

16. learn to take advantage of the IDE's toolchain and API browser
17. *check if my text editors allow me to print (and save as PDF files) syntax highlighted:*
   (a) `SPICE` decks
   (b) `Verilog` code
   (c) `Verilog-AMS` code
   (d) `SystemVerilog`™ code
   (e) `SystemC`™ code
   (f) `SystemC-AMS`™ code
   (g) `VHDL` code
   (h) `VHDL-AMS` code
18. **Try Xcode and Eclipse; they are great IDEs**

---

19. **Software Analysis**
20. analyze the EDA software:
   (a) analyze the verification/test code coverage for the software using code coverage tools:
      i. EMMA (code coverage tool); see http://emma.sourceforge.net/ and http://www.eclemma.org/
   (b) CPU and memory usage
   (c) load capabilities
   (d) use professional tools for performing such audits.
21. `OpenC++`: a software tool to parse and analyze C++ source code; see http://opencxx.sourceforge.net/

22. `Flex++`: a parser generator for creating language parsing programs; `Flex` primarily generates C code to be compiled, as opposed to `C++` libraries and code. `Flex++` is used for generating C++ code and classes. The `Flex++` classes and code require a `C++` compiler to create lexical and pattern-matching programs. `Flex` defaults to generating a parsing scanner in `C` code. The `Flex++` generated `C++` scanner includes the header file `FlexLexer.h`, which defines the interfaces of the two `C++` generated classes. See http://flex.sourceforge.net/
23. `Instruments`® is a performance analyzer and visualizer for `Mac OS X`®:
    (a) Use this tool for performance analysis of my software (EDA tools) on my `Mac Book Pro`® and `PowerBook`®
    (b) Instruments of change: `Instruments` includes analysis instruments in six main categories
        i. **User events:** Tracks the exact time of user interactions, such as mouse clicks.
        ii. **CPU and processes:** Monitors activity, sampling, load graphs, and threads.
        iii. **Memory:** Tracks garbage collection, object allocations, and leaks.
        iv. **File activity:** Monitors disk activity, reads and writes, and file locks.
        v. **Network activity:** Measures and records network traffic.
        vi. **Graphics:** Interprets the inner workings of the `OpenGL` driver.
    (c) Tracks that track:
        i. Now you can visually compare many analysis instruments side by side over time.
        ii. Thats because `Instruments` creates a time-based record of your entire application run, storing information such as CPU load, network and file activity, and memory allocations as separate tracks of data.
        iii. These tracks are displayed synchronized over time, allowing you to quickly identify application events – like what was happening with the disk just before the CPU usage spiked.
        iv. `Instruments` gives you a complete picture of your application, so you can better understand cause-and-effect relationships and make changes to improve performance.
    (d) Record, replay, repeat:
        i. Instruments works with new `Universal Access features` in `Leopard` to record a user-interactive run of your application, making it easy to replay the same behavior over and over.
        ii. That way, you can create ad hoc application tests anytime you like.
        iii. Record the applications behavior in `Instruments`, change your code, and rerun the tests to see the effect of your changes.
        iv. Run your application in a template and `Instruments` visually compares the runs side by side, making it clear where your code changes affect performance or memory consumption.
    (e) See http://www.apple.com/macosx/developertools/instruments.html
    (f) Built-in instruments can track:
        i. User events, such as keyboard keys pressed and mouse moves and clicks with exact time.
        ii. CPU activity of processes and threads.
        iii. Memory allocation and releasing of memory regions, garbage collection, and memory leaks.
        iv. File reads, writes, locks.
        v. Network activity and traffic.
        vi. Graphics and inner workings of `OpenGL`.
24. carry out performance engineering, which process includes:
    (a) performance optimization (and performance verification)
    (b) performance analysis
    (c) performance tuning
    (d) performance testing, including load testing, stress testing, endurance testing (soak testing), and spike testing

   (e) [*repeat if necessary*]

25. **learn to use performance analysis tools**

---

26. **Software Testing**
27. automate regression testing; 'nuff said:
    (a) develop a software test automation framework (STAF) that is a cross-platform, distributed software test environment
28. software testing:
    (a) functional software testing:
        i. black box testing:
            A. take an external perspective of the test object to derive test cases
        ii. white box testing:
            A. also known as:
                - clear box testing
                - glass box testing
                - transparent box testing
                - translucent box testing
                - structural testing
            B. use an internal perspective of the system to design test cases based on internal structure
            C. includes:
                - Control flow testing
                - Data flow testing
                - Branch Testing
        iii. grey box testing: having access to internal data structures and algorithms for purposes of designing the test cases, but testing at the user, or black-box level
        iv. user acceptance testing (UAT)
        v. smoke test: a preliminary to further testing, which should reveal simple failures severe enough to reject a prospective software release. In this case, the smoke is metaphorical.
    (b) non-functional software testing:
        i. performance testing
        ii. stability testing
        iii. usability testing:
            A. performance:
                - speed
                - memory capacity
            B. accuracy:
                - Does it give me the right answer?
                - Or only a vague answer?
            C. error tolerance of software:
                - Can it tolerate any user errors (mistakes)?
                - If so, what kind of mistakes are tolerated?
                - How many mistakes can it tolerate?
            D. (ease of) recall:
                - After a significant period of non-use (absence of usage), how much can the user remember about "using the software"?
                - Can the user remember how to use it?
                - Or, at least, most of the software functions?
            E. emotional response:
                - How does the person feel about the tasks completed?

- Is the person confident, stressed?
- Would the user recommend this system to a friend?

iv. security testing
v. internationalization and localization

(c) levels of software testing:
   i. unit testing
   ii. module testing
   iii. integration testing (check for bugs at the interfaces of the modules and caused by modular interaction)
   iv. system testing
   v. system integration testing (validation by user in their computing environment/platform)

(d) regression testing

(e) carry out alpha and beta testing before shipping the final version of software

(f) artifacts of the software testing process (testing artifacts):
   i. Test case: A test case in software engineering normally consists of a unique identifier, requirement references from a design specification, preconditions, events, a series of steps (also known as actions) to follow, input, output, expected result (postcondition), and actual result.
   ii. Test script:
      A. a set of instructions that will be performed on the system under test to test that the system functions as expected
      B. the test script is the combination of a test case, test procedure, and test data
   iii. Test data: Multiple sets of values or data are used to test the same functionality of a particular feature. All the test values and changeable environmental components are collected in separate files and stored as test data.
   iv. Test suite:
      A. a collection of test cases that are intended to be used to test a software program to show that it has some specified set of behaviours
      B. contains detailed instructions or goals for each collection of test cases, and information on the system configuration to be used during testing
      C. executable test suite: test suite integrated with test harness
   v. Test plan:
      A. **it documents the strategy that will be used to verify and ensure that a product or system meets its design specifications and other requirements**
      B. a test specification that lets the developers (dev) know what test cases will be used by the software quality assurance (QA) team
      C. a systematic approach to testing a system, such as a machine or software
      D. the plan typically contains a detailed understanding of what the eventual workflow will be
      E. it includes one or more of the following:
         - Design Verification, or Compliance test: to be performed during the development or approval stages of the product, typically on a small sample of units
         - Manufacturing/Production test: to be performed during preparation or assembly of the product in an ongoing manner for purposes of performance verification and quality control
         - Acceptance/Commissioning test: to be performed at the time of delivery or installation of the product
         - Service and Repair test: to be performed as required over the service life of the product
         - Regression test: to be performed on an existing operational product, to verify that existing functionality didn't get broken when other aspects of the environment are

changed (e.g., upgrading the platform on which an existing application runs)

F. three major elements of a test strategy, which should be described in the test plan:
  - Test Coverage:
    - states what requirements will be verified during each stage of the product life
    - derive test coverage from design specifications and other requirements, such as safety standards or regulatory codes
    - each requirement or specification of the design would have one or more corresponding means of verification
    - test coverage for different product life stages may overlap
    - test coverage also feeds back into the design process, since the product may have to be designed to allow test access; see Design For Testability (DFT)
  - Test Methods:
    - they state how test coverage will be implemented
    - they may be determined by standards, regulatory agencies, or contractual agreement, or created ad hoc
    - they also specify test equipment to be used in the performance of the tests and establish pass/fail criteria
    - test methods used to verify hardware design requirements can range from very simple steps, such as visual inspection, to elaborate test procedures that are documented separately
  - Test Responsibilities:
    - include what organizations will perform the test methods, and at each stage of the product life
    - allow test organizations to plan, acquire, or develop test equipment and other resources necessary to implement the test methods for which they are responsible
    - indicate what data will be collected, and how that data will be stored and reported (often referred to as "deliverables")
    - one outcome of a successful test plan should be a record or report of the verification of all design specifications and requirements as agreed upon by all parties

vi. Test harness: The software, tools, samples of data input and output, and configurations

vii. Traceability matrix: A table that correlates requirements (or design documents) to test documents. It is used to change tests when the source documents are changed, or to verify that the test results are correct.

viii. Testbed:

A. A platform for experimentation for large development projects.

B. Testbeds allow for rigorous, transparent and replicable testing of scientific theories, computational tools, and other new technologies.

C. A sandbox is a testing (or virtual) environment that isolates untested code changes and experimentation from the production environment or repository, in the context of software development including Web development and revision control, and by extension in web-based editing environments including wikis.

ix. Test oracle:

A. a mechanism used to determine whether a test has passed or failed

B. to determine whether the software passes the test case for input(s) of a given test case, it compares the output(s) of the system under test to the expected output(s)

C. oracles are always separate from the system under test

D. common oracles include:
  - specifications and documentation
  - a substitute software for carrying out the same function, while using different algorithm(s) and heuristic(s)
  -

(g) phases in a software testing cycle:
   i. requirements analysis and design phase: QA team work with developers in determining what aspects of a design are testable, and the required parameters for those tests to work
   ii. test planning: test strategy, test plan, and testbed creation
   iii. test development: test procedures, test scenarios, test cases, test datasets, and test scripts
   iv. test execution: QA team execute/run the software based on the test plan, and carry out bug reporting (and/or feature request – missing feature?)
   v. test reporting: provide test reports on the test effort, and indicate whether the software is ready for release based on selected metrics indicated in the test plan
   vi. test result analysis, or defect analysis: the development team and the users decide what defects should be treated, fixed, rejected (i.e., false positive), or deferred (with respect to making a decision)
   vii. retesting the resolved defects: once a defect has been dealt with by the development team, it is retested by the testing team
   viii. regression testing: it is common to have a small test program built of a subset of tests, for each integration of new, modified or fixed software, in order to ensure that the latest delivery has not ruined anything, and that the software product as a whole is still working correctly
   ix. test closure: once the test meets the exit criteria, the activities such as capturing the key outputs, lessons learned, results, logs, and documents related to the project are archived and used as a reference for future projects

(h) Scenario testing uses scenario tests (or scenarios) that are based on a hypothetical story to help a person think through a complex problem or system for a testing environment:
   i. The ideal scenario has five key characteristics: it is:
      A. a story that is
      B. motivating
      C. credible
      D. complex, and
      E. easy to evaluate
   ii. These tests are usually different from test cases; test cases cover a single step, whereas scenarios cover a number of steps
   iii. Test suites and scenarios can be used together for complete system testing

(i) All-pairs testing or pairwise testing is a combinatorial software testing method that, for each pair of input parameters to a system (typically, a software algorithm), tests all possible discrete combinations of those parameters. Using carefully chosen test vectors, this can be done much faster than an exhaustive search of all combinations of all parameters, by "parallelizing" the tests of parameter pairs. The number of tests is typically *O(nm)*, where n and m are the number of possibilities for each of the two parameters with the most choices.

29. use `Expect`, which is a Unix automation and testing tool, to facilitate automated regression testing
30. a good bug report includes a stack trace and a set of reduced test cases
31. before filing a bug report:
   (a) verify that it exists in the latest build
   (b) next, verify it hasn't been filed already
   (c) next, file the bug using the bug reporting template
32. keep refining my `software quality assurance` process, along with my software development process
33. `GM seed` (or GM build): GM stands for Gold Master. It's the finalized firmware before public release.

34. **Debugging, Including Memory Debugging**
35. do bug scrubbing whenever I find a bug:
    (a) don't put off bug scrubbing until a later date, because I probably won't do it then and procrastinate again
    (b) compound bugs/faults are harder to resolve
    (c) fault masking makes debugging considerably harder
36. use `Valgrind`, `Insure++`, and `IBM Rational Purify` for:
    (a) memory debugging
    (b) memory leak detection
    (c) memory corruption detection (buffer overflows)
    (d) software profiling (software performance analysis)
    (e) e.g., use runtime analysis to detect memory corruption detection (buffer overflows) and memory leak detection
37. **learn to use GDB very well for debugging**
38. **learn to use software formal verification tools**
39. **learn to use bug tracking systems, ECO management systems, or issue tracking systems**; also, see bug databases:
    (a) `Bugzilla`
    (b) `Trac`
    (c) `Atlassian Jira`
    (d) `LibreSource`
    (e) `SharpForge`
40. check out distributed bug trackers, such as "DisTract" and "Bugs Everywhere"
41. **improve the way I perform dynamic memory management**
42. **Boehm-Demers-Weiser garbage collector**, **Boehm GC**, or **Boehm garbage collector**; see `http://www.hpl.hp.com/personal/Hans_Boehm/gc/`, `http://developers.sun.com/solaris/articles` `libgc.html`, `http://en.wikipedia.org/wiki/Boehm_garbage_collector`, or `http://sourceforge.net/projects/bdwgc/`
43. Unusual software bugs: `http://en.wikipedia.org/wiki/Unusual_software_bug`
44. Links:
    (a) `http://www.debugging-guide.com/tools.html`
    (b) `http://www.debugging-guide.com/links.html`

---

45. **Formal Technical Reviews & Software Reviews**
46. software review:
    (a) "A process or meeting during which a software product is [examined by] project personnel, managers, users, customers, user representatives, or other interested parties for comment or approval"
    (b) software product:
        i. any technical document or partial document, produced as a deliverable of a software development activity
        ii. may include documents such as:
            A. contracts
            B. project plans and budgets

C. requirements documents
D. specifications
E. designs
F. source code
G. user documentation
H. support and maintenance documentation
I. test plans
J. test specifications
K. standards
L. and any other type of specialist work product

(c) include:

i. Software peer reviews are conducted by the author of the work product, or by one or more colleagues of the author, to evaluate the technical content and/or quality of the work
ii. Software management reviews are conducted by management representatives to evaluate the status of work done and to make decisions regarding downstream activities
iii. Software audit reviews are conducted by personnel external to the software project, to evaluate compliance with specifications, standards, contractual agreements, or other criteria

(d) values of software reviews:

i. defect detection process: makes it less costly to find bugs than software testing or field use
ii. defect prevention process: identify and remove process inadequacies that encourage defects
iii. Early and frequent reviews of small work samples can identify systematic errors in the authors' work processes, which can be corrected before further erroneous work is done. This can help the authors improve their skills and significantly reduce the time it takes to write a high-quality technical document, and dramatically decrease the error rate in using the document in downstream processes.

47. formal technical review:

(a) also known as:

i. software inspection
ii. code inspection
iii. code review
iv. code walkthrough
v. engineering peer review
vi. product peer review
vii. software peer review
viii. technical peer review

(b) "peer review of any work product by trained individuals who look for defects using a well defined process"

(c) "designer or programmer leads members of the development team and other interested parties through a software product, and the participants ask questions and make comments about possible errors, violation of development standards, and other problems"

(d) "a team of qualified personnel ... examines the suitability of the software product for its intended use and identifies discrepancies from specifications and standards. Technical reviews may also provide recommendations of alternatives and examination of various alternatives"

(e) "A code review can be done as a special kind of inspection in which the team examines a sample of code and fixes any defects in it. In a code review, a defect is a block of code which does not properly implement its requirements, which does not function as the programmer intended, or which is not incorrect but could be improved (for example, it could be made more readable or its performance could be improved). In addition to helping teams find and fix bugs, code reviews are useful for both cross-training programmers on the code being reviewed and for helping junior developers learn new programming techniques."

(f) "The goal of the inspection is to identify defects"

(g) "The purpose of a technical review is to arrive at a technically superior version of the work product reviewed, whether by correction of defects or by recommendation or introduction of alternative approaches. While the latter aspect may offer facilities that software inspection lacks, there may be a penalty in time lost to technical discussions or disputes which may be beyond the capacity of some participants."

(h) roles of participants:
   i. Moderator / Walkthrough Leader / Review Leader: can be the team manager/leader (preferable)
   ii. Inspectors / Technical Staff / Reviewer
   iii. Author
   iv. Reader
   v. Recorder/Scribe
   vi. *Decision Maker:* can be the team manager/leader (preferable), or the author (bad idea); this may also be carried out by the Review Leader
   vii. *Management Staff*
   viii. *Customer, or User Representatives*

(i) steps in this process:
   i. entry evaluation:
      A. the Review Leader uses a standard checklist of entry criteria to ensure that optimum conditions exist for a successful review
      B. Is the artifact ready for review?
      C. Here, the artifact refers to:
         - software
         - IC design
         - software/IC module
         - document
   ii. management preparation:
      A. management staff manages resources (time, materials, and tools) required/needed for the review
      B. they also manages how the reviews shall be conducted according to policies, standards, or other relevant criteria
   iii. planning [the review]:
      A. the moderator staff plans the review/inspection
      B. the Review Leader identifies or confirms the objectives of the review
      C. the Review Leader also organizes a team of Reviewers
      D. in addition, the Review Leader ensures that the team is equipped with all necessary resources for conducting the review
   iv. overview meeting (or "overview of review procedures"):
      A. The Review Leader ensures that all Reviewers understand:
         - the review goals
         - the review procedures
         - the materials available to them
         - the procedures for conducting the review
      B. the author presents the background information of the artifact, which helps the reviewers acquire the necessary knowledge to carry out the review and put things in perspective

v. [Individual] preparation:
   - A. each Reviewer prepares for group examination of the artifact by examining it carefully for anomalies (potential/possible defects), the nature of which will vary with the type of review and its goals
   - B. each inspector shall record all identified "potential/possible defects"
   - C. possible defect: Is this a defect? Can this be a defect?
   - D. potential defect:
     - Can this lead to a defect? Or, can it become a defect later on?
     - If so, identify this as a "potential defect," so that it can be discussed at the "inspection meeting"; at the "inspection meeting," the Reviewers can suggest modifications that can prevent failures/errors from occurring
   - E. defects can be major or minor

vi. inspection meeting (or "group examination"):
   - A. during this meeting the reader reads through the artifact, part by part, and the inspectors point out the defects for every part
   - B. consolidate the findings of the Reviewers (from their individual preparation)
   - C. arrive at a consensus regarding the status of the artifact (or activity) being reviewed, and come up with an action plan addressing each defect; the defects in the action plan include potential defects

vii. rework:
   - A. the author(s) make(s) changes to the artifact/process to correct the errors/defects or resolve the failures
   - B. this is carried out according to the action plan(s) from the inspection meeting
   - C. if the action plan cannot be adhered to, the author(s) should indicate why and bring that up to the Review Leader and committee of Reviewers; if necessary, another review may be held to modify the action plan (if necessary, repeat the process)

viii. follow-up:
   - A. the Review Leader checks that the author has made changes according to the action plan
   - B. if changes are not made according to the action plan, go back to the `rework` stage/phase

ix. The process is terminated by the moderator when it satisfies some predefined exit criteria:
   - A. all action plans must have been adhered to and completed
   - B. the Review Leader also checks that documents used in this review been kept in the repository under a change management (or configuration management) process

(j) use a software repositories based on `Subversion` with `Trac`, `Mercurial`, or `GIT` to allow people to collaboratively review code; `specific tools for collaborative code review can facilitate the code review process`:

i. `Trac`:
   - A. an open source, web-based project management (roadmap & milestones) and bug-tracking tool
   - B. `Trac` allows hyperlinking information between a computer bug database, revision control and wiki content
   - C. `Trac` also serves as a web interface to a version control system like:
     - `Mercurial`
     - `Git`
     - `Subversion`
     - `Bazaar`
     - `Darcs`: **Try this with Trac**
   - D. RSS Feeds
   - E. **If I can draw Gantt charts with this, USE THIS!!!**

ii. `Mercurial`: a cross-platform, distributed revision control tool; ∃ a plugin for Eclipse

   (k) use "automated code review software" to facilitate the code review process
   (l) categories of code review process:
      i. formal code review
      ii. lightweight code review (includes pair programming)
   (m) be objective
   (n) do not make this a scoring process: "Oh, I found more bugs in your code"
   (o) they differ from:
      i. management reviews:
         A. conducted by management representatives, rather than by colleagues
         B. conducted for management purposes (e.g., promotion, bonuses, and incentive pay), rather than for technical evaluation
      ii. software audit reviews:
         A. conducted by personnel external to the project
         B. they evaluate compliance with specifications, standards, contractual agreements, or other criteria

---

48. **Coding Standards, Code Documentation Guidelines, and Coding Style Guide**
49. Documentation generators that I am using:
   (a) Doxygen (for C++, C, Java, Python, Fortran, and VHDL)
   (b) DOC++ (for C/C++ and Java)
   (c) ROBODoc (for Ruby)
   (d) RDoc (for Ruby)
   (e) Sphinx (for Python): `http://sphinx.pocoo.org/`
   (f) Javadoc (for Java)
   (g) Haddock (for Haskell)
   (h) DocBook (semantic markup language) that can produce documents in various formats. **Note: I am not sure if it can parse source code, and churn out the appropriate documentation (e.g., API) in PDF, as man pages, and in HTML. While I can use it for documentation of my projects, I may as well use LaTeX since I would have to use LaTeX for my research publications anyway.**
50. My coding style should follow that of `Javadoc`, which is similar to that used by Doxygen. My indent style would be the `1TBS` variant of the `K&R` style, which is an abbreviation of "`The One True Brace Style`". It is also equivalent to the `Kernel Normal Form style` (or `BSD KNF style`).
51. Tags that I would use in my comments are:
   (a) @author *Author's_Name*: indicate the author (*Author's_Name*) of the file/function
   (b) @version *X.Y*: indicate the version (*X.Y*) of the file
   (c) @section *SECTION_NAME*: indicate the section (*SECTION_NAME*) of the file, which can be: *LICENSE* or *DESCRIPTION*
   (d) @param x: indicate the parameter (x) of the constructor or function
   (e) @exception *Exception_Name*, or @throws *Exception_Name*: an exception that a function/method can throw
   (f) @return *Return_Statement*: indicate the return (type and) action of the function
   (g) @see *reference*: a link to another element in the documentation; e.g., @see *Class_Name*, or @see *Class_Name#member_function_name*
   (h) @since *X.Y: Month-Day-Year*: This functionality has been added since version *X.Y* (and on the date *Month-Day-Year*)

(i) @deprecated *description*: Describe an outdated function/method, and indicate when the function/method has deprecated
(j) "@link ... *URL*... @endlink" is used to include hyperlinks in the generated documentation for Doxygen
(k) #### IMPORTANT NOTES: Notes that are critical for helping the reader understanding assumptions and decisions made while developing the software
(l) @todo(<message>, <version>) (or #### TO BE COMPLETED): Task to be finished at a later time
(m) #### TO BE FIXED: Task to be debugged at a later time
(n) @migration(<message>, <version>): Code is being migrated to another function/method, or class.
(o) See `http://www.stack.nl/~dimitri/doxygen/commands.html` for more information of tags that are recognized by `Doxygen`.
(p) @pre (or @precondition): Precondition(s) of the function.
(q) @assert: Assertion(s) of the function.
(r) @post (or @postcondition): Postcondition(s) of the function.

52. The order of tags in different sections of the code is given as follows:
    (a) Headers/Interfaces and Classes: @version, @author, and @see
    (b) Constructors: @param, @exception, and @see
    (c) Functions/Methods: @param, @return, @exception, and @see
    (d) Variables can use the @see tags.
    (e) The @deprecated tag can be used for headers/interfaces, classes, constructors, functions/methods, and variables.
53. Since I am using Doxygen for generating documentation, I can use LaTeX to provide richer markup.
54. Use `ROBODoc` and `RDoc` for `Ruby`, and the latter for `Tcl` and `Perl`. The latter also generates documentation in LaTeX, `HTML`, and `UNIX man pages`.
55. For a suggested coding style for `Python` scripts, see `http://www.python.org/dev/peps/pep-0008/`.
56. For a suggested coding style for `Ruby` scripts, see `http://www.caliban.org/ruby/rubyguide.shtml`.
57. Document the known bugs for each function/method.
58. `GNU coding standards`; see `http://www.gnu.org/prep/standards.html` and `http://en.wikipedia.org/wiki/GNU_Coding_Standards`; also see `http://www.ronancrowley.com/Eclipse_GNU_Style.xml` for "`Eclipse` Code Style Formatter for `GNU Coding Standards`"
59. FMG Programming Guidelines (FPG); see `http://www-ti.informatik.uni-tuebingen.de/~fmg/guidelines.html` ... Also, see `http://www.boost.org/development/requirements.html` and `http://cm.bell-labs.com/cm/cs/tpop/`
60. `Gnits Standards`: a collection of standards and recommendations for programming, maintaining, and distributing software; see `http://www.gnu.org/software/womb/gnits/`
61. `Mozilla` Coding Style Guide; see `https://developer.mozilla.org/en/Mozilla_Coding_Style_Guide` or `http://www.mozilla.org/hacking/mozilla-style-guide.html`
62. `Linux` Kernel Coding Style (or Documentation/CodingStyle in the `Linux` Kernel source tree); see `http://lxr.linux.no/source/Documentation/CodingStyle`
63. code documentation guidelines;
    see `http://www-ti.informatik.uni-tuebingen.de/~fmg/guidelines.html`
64. The `NetBSD` source code style guide; see `ftp://ftp.netbsd.org/pub/NetBSD/NetBSD-current/src/share/misc/style`

65. use a good **documentation generator**, such as `Doxygen` or `Javadoc`, to generate API documentation, end-user guide (or user manual), and call graphs from commented source code files; special tags (or metadata/annotations) within source code comments are often used to process documentation
66. terms used interchangeably to describe this:
    (a) coding standards
    (b) code documentation guidelines
    (c) coding style guide
    (d) programming style
    (e) code conventions
67. this covers:
    (a) the layout of the source code; code appearance; the visual appearance of source code; code formatting
    (b) indent style
    (c) the use of white space around operators and keywords
    (d) naming conventions:
        i. the style and spelling of user-defined identifiers, such as function, procedure, and variable names
        ii. the capitalization or otherwise of keywords and variable names
        iii. common elements:
            A. length of identifiers
            B. letter case and numerals
            C. multiple-word identifiers:
                - delimiter-separated words
                - letter-case separated words
        iv. examples of naming conventions include:
            A. CamelCase / "camel case" / medial capitals
            B. Hungarian notation
    (e) the use and style of comments:
        i. block comments (or prologue comments)
        ii. line comments (or inline comments)
        iii. code descriptions: comments can be used to summarize code or to explain the programmer's intent
        iv. algorithmic description: comments may contain an explanation of the methodology; such explanations may include diagrams and formal mathematical proofs
        v. resource inclusion:
            A. logos, diagrams, and flowcharts consisting of ASCII art constructions can be inserted into source code formatted as a comment
            B. copyright notices can be embedded within source code as comments
            C. Binary data may also be encoded in comments through a process known as binary-to-text encoding, although such practice is uncommon and typically relegated to external resource files
        vi. debugging: a common developer practice is to comment out a code snippet, so that it will not be executed in the final program
    (f) the use or avoidance of particular programming constructs (such as GOTO statements)
    (g) left-hand comparison style: place constants or expressions to the left in any comparison
    (h) declarations
68. see `http://en.wikipedia.org/wiki/Programming_style` for examples of programming styles:

(a) http://google-styleguide.googlecode.com/svn/trunk/cppguide.xml
(b) http://en.wikibooks.org/wiki/C%2B%2B_Programming/Code_Style
(c) http://www.possibility.com/Cpp/CppCodingStandard.html
(d) http://www.state-machine.com/doc/AN_QL_Coding_Standard.pdf
(e) http://geosoft.no/development/cppstyle.html
(f) http://www.qhull.org/road/road-faq/xml/cpp-guideline.xml
(g) http://java.sun.com/docs/codeconv/
(h) http://www.ambysoft.com/essays/javaCodingStandards.html
(i) http://geosoft.no/development/javastyle.html
(j) http://perldoc.perl.org/perlstyle.html
(k) http://rpa-base.rubyforge.org/wiki/wiki.cgi?GoodAPIDesign
(l) http://www.caliban.org/ruby/rubyguide.shtml
(m) http://www.python.org/peps/pep-0008.html
(n) http://httpd.apache.org/dev/styleguide.html
(o) http://drupal.org/coding-standards
(p) http://pear.php.net/manual/en/standards.php
(q) http://www.chris-lott.org/resources/cstyle/
(r) TIOBE Software BV, *TIOBE Coding Standard Methodology*: http://www.tiobe.com/index.php/content/paperinfo/HowToUseCodingStandards.html
(s) Virginia Tech:
  i. Department of Computer Science:
    A. Clifford A. Shaffer, "Elements of Programming Style: Department of Computer Science, Computer Program Documentation Standards: Version 1.3," for the *Data Structures and File Processing* class, Department of Computer Science, Virginia Tech (Virginia Polytechnic Institute and State University). Available online at: http://ei.cs.vt.edu/~cs2604/Standards/Standards.html; last accessed on September 30, 2010.
(t) books:
  i. B.W. Kernighan and P.J. Plauger. The Elements of Programming Style. McGraw-Hill, New York, NY, second edition, 1982. [Also, see the following summary. Tae Yano, "Quote from 'The Elements of Programming Style' by Kernighan and Plauger," Computer Science Department, School of Computer Science, Carnegie Mellon University. Available online at: http://www.cs.cmu.edu/~taey/pub/element_of_p_style.txt; last accessed on September 30, 2010.] [9]

69. certain tags are used in comments to assist in indexing common issues:
  (a) examples of tag conventions include:
    i. FIXME to mark potential problematic code that requires special attention and/or review
    ii. NOTE to document inner workings of code and indicate potential pitfalls
    iii. TODO to indicate planned enhancements
    iv. XXX to warn other programmers of problematic or misguiding code
  (b) There is a risk that tags accumulate over time; it is advisable to include the date and the tag owner in the tag comment to ease tracking
70. source code is not always self-explanatory
71. comments that are superfluous, excessive, or difficult to maintain MUST be omitted
72. the level of detail and description of comments may vary considerably; **good comments clarify intent**
73. do not include offensive comments in the source code

74. place the comments in-context, so that the comment is relevant to the code adjacent to it

---

75. **Hybrid Software Involving Multiple Computer Languages**
76. use the `Swig` software tool and/or the `Boost Python C++` library to integrate `C++` and `Python` code, and to provide seamless operability between them
77. `C++/Tcl` is a software programming library interface which allows the integration of `C++` into `Tcl` and vice versa:
    (a) the `C++/Tcl` library offers the following features:
        i. Support for both extending `Tcl` with `C++` modules and embedding `Tcl` in `C++` applications
        ii. Possibility to expose free `C++` functions as commands in `Tcl`
        iii. Possibility to define classes and class member functions, visible in `Tcl` in the style similar to `SWIG` wrappers
        iv. Possibility to manipulate `Tcl` lists and objects from the `C++` code
78. `Tcl/Java` is a project to bridge `Tcl` and `Java`:
    (a) The `Tcl/Java` project's goal is to make integrating the `Java` platform and the `Tcl` scripting language as easy as possible
    (b) `Tcl/Java` consists of two packages, `Jacl` and `Tcl Blend`
    (c) `Jacl`:
        i. a `Tcl` interpreter that is written in `Java`
        ii. a self-contained implementation of a `Tcl` interpreter, written entirely in `Java`
        iii. run `Tcl` within the `JVM`
        iv. `Jacl` also includes features that facilitate communication between a `Java` interpreter and a `Tcl` interpreter
        v. `Jacl` is typically used to incorporate scripting functionality into an existing `Java` application
        vi. doesn't require the user to deal with the complexities of native code that come with `Tcl Blend`
        vii. **Jacl** is used to integrate **Tcl** scripts into **Java** applications
    (d) `Tclblend`:
        i. call `Java` class libraries from the `C` runtime using `JNI` and an embedded `JVM`
        ii. `Tcl Blend` is a `Tcl` extension that makes use of `JNI` to facilitate communication between a `Java` interpreter and a `Tcl` interpreter
        iii. `Tcl Blend` is typically used to load a `Java` interpreter into an existing `Tcl` process, so that functionality implemented in `Java` can be accessed via `Tcl`
        iv. One can also load `Tcl Blend` and `Tcl` into a `Java` process, which is a great way to add scripting functionality to an existing `Java` application
        v. Because `Tcl Blend` is a normal `Tcl` extension, one can use it with other popular `Tcl` extensions like `Tk`, `Expect`, and `Itcl`
        vi. **Tcl Blend is used to call/invoke Java methods on Java objects that are allocated in Tcl scripts**
    (e) `Tcl Blend` and `Jacl` define both a `Tcl` API and a `Java` API that make it easy to call `Java` code from `Tcl` or call `Tcl` code from `Java`
    (f) For example, one could allocate a `Java` object in a `Tcl` script and interactively invoke `Java` methods on the object
    (g) It is also easy to use the supplied API to evaluate a `Tcl` procedure from a `Java` method or implement `Tcl` procudures in `Java`
    (h) The flexible API and wealth of implementation options provided by the `Tcl/Java` project make integrating `Tcl` and `Java` easy

79. also, see `Jython` and `JRuby`

---

80. **Concurrent Programming and Scalability**
81. Does the EDA application scale (well)?:
    (a) Does it scale for larger input netlists (or system-level or behavioral RTL design models)?
82. Also, does it scale well for parallel computers with more processor nodes/cores? ... Scale my EDA tool for:
    (a) multicore processors
    (b) many-core processors
    (c) 64-bit processors
    (d) GPGPU computing platform / GPU accelerators
    (e) FPGA implementation
    (f) Linux cluster
    (g) server farm???
    (h) grid computing
    (i) distributed computing
    (j) hardware accelerator / accelerator boards
83. learn about different APIs for concurrent programming, such as:
    (a) `POSIX Threads`; see http://developer.apple.com/mac/library/samplecode/PThreadSorts/index.html for an example of using POSIX threads (pthreads) to sort pictures.
    (b) `OpenMP`
    (c) `MPI`
    (d) `UPC`
    (e) `Intel Threading Building Blocks`
    (f) `Boost.Thread`
    (g) `OpenCL` (Open Computing Language) framework: for **stream processing**; see http://www.khronos.org/registry/cl/, http://www.khronos.org/opencl/sdk/1.0/docs/man/xhtml/, and http://www.khronos.org/opencl/
    (h) `Global Arrays`
    (i) `Charm++`
    (j) `Cilk`
    (k) `CUDA`:
        i. previously "Compute Unified Device Architecture"
        ii. program in "C for CUDA" (C with NVIDIA extensions)
        iii. Third party wrappers are also available for:
            A. `Java`
            B. `Python`
            C. `Fortran`
            D. `Jacket`: A CUDA-engine for MATLAB
84. GPGPU computing vs using multicore/manycore processors:
    (a) options for GPGPU computing include `NVIDIA`'s `CUDA` technology
    (b) options for multicore include `IBM`'s `Cell BE`
    (c) "the typical barrier to adopting GPUs is simple, you have to re-write your code"
    (d) however, as multi-/many-core processors scale in terms of the number of processor cores, software that run on such multiprocessors need to be rewritten to keep each core busy
    (e) note that rewriting code can be very costly in terms of time

(f) rewriting code for GPGPU computing or multi-/many-core processors probably requires software refactoring of the EDA software architecture
(g) explore the benefits of GPGPU computing vs those of multi-/many-core processors the way a software architect would
(h) see `http://www.chipdesignmag.com/payne/2009/04/13/gpu-versus-multicore-in-eda/`

85. techniques to speed up a computer program:
    (a) automatic partitioning
    (b) load balancing
    (c) event synchronization
    (d) memory optimization
86. **make sure that I provide multi-threading options to my EDA tools**
87. See `http://www.cilk.com/` to develop multicore-enabled software applications
88. get compilers that can compile code for parallel computing platforms, such as multicore processors
89. learn dataflow programming, and apply it to concurrent computing to make data flow concurrently
90. learn about data-parallel programming
91. Maurice Herlihy and Nir Shavit, "The Art of Multiprocessor Programming," Morgan Kaufmann Publishers, Burlington, MA, 2008

---

92. **Minimizing (Peak) Memory Usage**
93. "Peak memory usage of a linux/unix process". Available online at: `http://stackoverflow.com/questions/774556/peak-memory-usage-of-a-linux-unix-process`; last accessed on July 30, 2011.
94. Google "measuring peak memory in C++"

---

95. **Real-Time Collaboration in Software Development**
96. use real-time collaborative editors so that my teammates and I can edit the same file at the same time; *non-real-time collaborative editors do not allow editing of the same file at the same time, so they are similar to revision control systems*
97. for real-time collaboration in software development, try:
    (a) `PaintChats`: facilitate group online chat sessions, and sharing of a whiteboard; see `http://en.wikipedia.org/wiki/Paint_chat`
    (b) *Coccinella*: It allows me to scribble something on a window, my mates' virtual whiteboard, and share my diagrams/words with other people. This is good for collaborating with others in a geographically distributed environment. See `http://thecoccinella.org/about`.
    (c) *Scriblink* "is a free digital sketchpad that users can share online in real time": `http://www.scriblink.com/`
    (d) *imagination at work* [General Electric Company]: `http://www.imaginationcubed.com/`
    (e) *TypeWith.me*: `http://typewith.me/`
    (f) *EtherPad* is a "web-based realtime collaborative document editor": `http://etherpad.com/` and `http://code.google.com/p/etherpad/`. Also, see `http://ietherpad.com/`.
    (g) *Etherpad* "is a hosted web service that allows really real-time document collaboration for groups of users": `http://etherpad.org/`
    (h) *Sync.in* "is a web based word processor for people to collaborate in real-time"; this is only free for public documents: `http://sync.in/`
    (i) *PiratePad*: `http://piratepad.net/front-page/` and `http://piratepad.net/UuYiC8Cy59`

(j) *Jarnal*, which "is an open-source application for notetaking, sketching, keeping a journal, making a presentation, annotating a document - including pdf - or collaborating using a stylus, mouse or keyboard": `http://www.dklevine.com/general/software/tc1000/jarnal.htm` or `http://jarnal.wikispaces.com/`. Compare this to "Microsoft Windows Journal".

(k) *PBworks*, which provides a wiki to support collaboration: `http://pbworks.com/`

(l) collaborative real-time editors

(m) computer supported cooperative work (CSCW)

(n) collaborative software (or groupware, or workgroup support systems)

(o) Collaborative Working Environment (CWE)

(p) `ACE`: a collaborative editor is a platform-independent, collaborative real-time editor; see `http://sourceforge.net/projects/ace`

(q) **Gobby — free real-time collaborative editor**

(r) `LibreSource`: a collaborative development platform for open-source software, groupware, community interaction, electronic archiving, and Web publishing. See `http://dev.libresource.org/`

(s) *Dimdim*:
  i. Is a service that lets anyone host and attend online meetings, demos and webinars via a web browser
  ii. Enables documents, web pages, whiteboards, audio, and video to be shared
  iii. Enables events to be recorded
  iv. Needs no software installation
  v. `http://www.dimdim.com/`

(t) *Dropbox*:
  i. Allows me to share files with others
  ii. It also allows others to transfer files to me
  iii. `https://www.dropbox.com/home#:::`

(u) `GForge`:
  i. a free software fork of the web-based project-management and collaboration software
  ii. provides facilities for:
    A. Source Code Management (SCM) via CVS and Subversion
    B. customizable Trackers
    C. Task Managers
    D. Document Managers
    E. Forums
    F. Mailing Lists
  iii. `http://gforge.org/gf/`

(v) electronic meeting system (EMS):
  i. a type of computer software that facilitates group decision-making within an organization
  ii. equivalent to:
    A. Group Support Systems (GSS)
    B. Group Decision Support Systems (GDSS)

---

98. **GUI Development**
99. use a GUI builder (graphical user interface builder or GUI designer) to help me create GUIs; examples of such tools are:

(a) `Qt` toolkit (pronounced as "cute") and `Qt Creator`; see `http://qtsoftware.com/`

(b) `GTK+`, or The `GIMP` Toolkit; see `http://www.gtk.org/`

(c) `GNUstep`; see `http://www.gnustep.org/`

(d) `wxWidgets`; see http://www.wxwidgets.org/
(e) `Tcl/Tk`; see http://www.tcl.tk/
(f) `OpenGL` (Open Graphics Library):
   i. a standard specification that defines a cross-language, cross-platform API for writing applications that produce 2D and 3D computer graphics
   ii. See http://www.khronos.org/opengl/, http://www.opengl.org/registry/, and http://www.opengl.org/sdk/
   iii. compare `OpenGL Utility Toolkit (GLUT)` with `freeglut`: http://freeglut.sourceforge.net/
(g) `OpenGL User Interface Library (GLUI)`:
   i. a `C++` user interface library based on the `OpenGL Utility Toolkit (GLUT)`, which provides controls such as buttons, checkboxes, radio buttons, and spinners to `OpenGL` applications
   ii. It is window- and operating system independent, relying on `GLUT` to handle all system-dependent issues, such as window and mouse management
   iii. It lacks the features of a more full-fledged GUI toolkit such as `Qt`, `wxWidgets`, or `FLTK`, but it has a very small footprint and is extremely easy to use
(h) `cairo`: 2D graphics library; see http://cairographics.org/. It seems that `cairo` involves using functional programming to create the graphics/GUI.
(i) `pycairo` is a set of `Python` bindings for the `cairo` graphics library; see http://cairographics.org/pycairo/
(j) `Motif`; see http://www.openmotif.org/ and http://www.opengroup.org/motif/ — I do not prefer to use `Motif` because GUI development with `Motif` requires a decent, if not good, Motif GUI IDE. An example of a decent Motif GUI Builder would be the proprietary X-Designer. Note that `Motif` is a proprietary software; the cost of its source code is A LOT (US$2,000 – US$17,000), and its object code costs ≈US$30.
(k) `Interface Builder` (for both `Carbon` and `Cocoa` applications); see http://developer.apple.com/tools/interfacebuilder.html — Avoid working with this, since I would be locked in to Apple Inc. and be forced to work on Mac OS X platforms.
(l) `XForms`; see http://savannah.nongnu.org/projects/xforms/

100. **Figure out how to develop interactive GUI using Java Swing**:
   (a) I used `Java Swing` to develop the GUI for my honors project at the University of Adelaide, and the simulator for the micromouse project as part of an extracurricular activity with IEEE@USC
   (b) I was unable to use the "`events manager`" or threads to handle user activities, and get the GUI to respond appropriately

101. things that I want to build in my GUI:
   (a) use drag-and-drop, instead of clicking (point-and-click & double-click) and scrolling in GUI development
   (b) pull-down menu or pop-up menu
   (c) menu bar
   (d) dialog box
   (e) tooltip
   (f) splash screen
   (g) about box or about dialog

102. use `ActiveState Komodo` tools, such as `Komodo IDE` and `Komodo Edit`, to develop (web) applications on the Mozilla platform, using dynamic programming languages

---

103. **Security, Encoding/Decoding, & Encryption/Decryption**
104. Botan:
    (a) a BSD-licensed cryptographic library written in C++
    (b) provides a wide variety of cryptographic algorithms, formats, and protocols
105. use a hardware dongle or the FlexLM license to protect my EDA software; dongles cut down on illegal copies

---

106. **Other Libraries and Packages**
107. `Boost C++ Libraries`: extend the functionality of `C++`:
    (a) see `http://boost.org/`
    (b) see `http://www.codeproject.com/vcpp/stl/boostsmartptr.asp`
    (c) see `http://shoddykid.blogspot.com/2008/07/getting-started-with-boost.html`
108. `Tcllib` is a collection of packages available for the `Tcl` programming language; it has packages for:
    (a) Mathematics
    (b) Data structures
    (c) Text processing
    (d) File formats
    (e) Hashes, checksums, and encryption
    (f) Documentation tools
    (g) Benchmark tools
    (h) Networking
    (i) Terminal control
    (j) CGI programming
    (k) Grammars and finite automata
    (l) TKLib
    (m) Transfer module
109. `dlib C++ Library`: a cross platform open source software library written in the `C++` programming language; it has functions for:
    (a) Numerical Algorithms:
        i. Arbitrary-precision arithmetic
        ii. Basic mathematical functions
        iii. Complex numbers
        iv. Linear algebra
        v. Random number generation
    (b) Machine Learning:
        i. Bayesian networks
        ii. Support Vector Machine Classification
        iii. Relevance Vector Machine Classification and Regression
        iv. Online Regression
        v. Online Novelty Detection
        vi. Online SVM classification
        vii. Kernel Clustering
        viii. Neural Networks

ix. Radial Basis Function Networks

(c) A simple Web Server
(d) Checksum Computation
(e) Command line interface Parsing
(f) Container Classes
(g) Data Compression
(h) Graphical user interface Support
(i) Image Processing
(j) log4j style logger
(k) Multithreading:
    i. Futures
(l) N-tuples
(m) Object Serialization
(n) TCP networking
(o) Unit Testing
(p) XML Parsing

110. `ScientificPython` is a `Python` library for common tasks in scientific computing; see http://dirac.cnrs-orleans.fr/plone/software/scientificpython/
111. `ACM Collected Algorithms (CALGO)`; see http://calgo.acm.org/

---

112. **Design and Analysis of Software Architecture and Algorithms**
113. learn and apply design patterns to software development; use them for:
    (a) object-oriented design (including architectural patterns???)
    (b) architectural design: architectural patterns:
        i. Architectural patterns are software patterns that offer well-established solutions to architectural problems in software engineering.
        ii. It gives description of the elements and relation type together with a set of constraints on how they may be used.
        iii. An architectural pattern expresses a fundamental structural organization schema for a software system, which consists of subsystems, their responsibilities and interrelations.
        iv. In comparison to design patterns, architectural patterns are larger in scale.
        v. Even though an architectural pattern conveys an image of a system, it is not an architecture as such.
        vi. An architectural pattern is rather a concept that captures essential elements of a software architecture.
        vii. Countless different architectures may implement the same pattern and thereby share the same characteristics.
        viii. Furthermore, patterns are often defined as something "strictly described and commonly available".
        ix. One of the most important aspects of architectural patterns is that they embody different quality attributes. For example, some patterns represent solutions to performance problems and others can be used successfully in high-availability systems. In the early design phase, a software architect makes a choice of which architectural pattern(s) best provide the system's desired qualities.
    (c) fundamental patterns
    (d) creational patterns:
        i. These patterns have to do with class instantiation.
        ii. They can be further divided into class-creation patterns and object-creational patterns.

iii. While class-creation patterns use inheritance effectively in the instantiation process, object-creation patterns use delegation to get the job done.
iv. creational design patterns deal with object creation mechanisms to avoid design problems or added design complexity during object creation

(e) structural patterns:
   i. These concern class and object composition.
   ii. They use inheritance to compose interfaces and define ways to compose objects to obtain new functionality.
   iii. structural design patterns are design patterns that ease the design by identifying a simple way to realize relationships between entities

(f) behavioral patterns:
   i. Most of these design patterns are specifically concerned with communication between objects.

(g) concurrency patterns: design patterns that deal with multi-threaded programming paradigm

(h) see "Design Patterns: Elements of Reusable Object-Oriented Software," by Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides (collectively known as the "Gang of Four", or GoF for short)

---

114. **Software Development Methodology**
115. **software development methodology**: a framework that is used to structure, plan, and control the process of developing software; it is based on a software development philosophy and a set of tools, models, and methods to facilitate software development
116. $\nexists$ a software development methodology that can be suitably used by all projects
117. a software development methodology shall be chosen based on various technical, organizational, project, and team considerations
118. familiarize myself with various software development processes and methodologies, such as:
   (a) object oriented development methodologies, including:
      i. object-oriented analysis and design (OOAD)
      ii. object-oriented programming
   (b) agile methodologies:
      i. Extreme Programming / eXtreme Programming (XP):
      ii. Scrum
      iii. Agile Unified Process (AUP)
   (c) heavyweight software development processes:
      i. Rational Unified Process (RUP)
      ii. cleanroom software engineering
   (d) generic software development approaches:
      i. waterfall: linear framework type
      ii. prototyping: iterative framework type
      iii. incremental: combination of linear and iterative framework types
      iv. spiral: combination of linear and iterative framework types
      v. rapid application development (RAD): iterative framework type
119. software development life cycle (SDLC):
   (a) phases in the SDLC:
      i. *initiation*
      ii. *system concept development*
      iii. planning

iv. [requirements gathering and] analysis
v. design
vi. *development* [building and/or coding]
vii. *integration and test*
viii. implementation
ix. [operations and] maintenance
x. *deployment / installation / disposition*
xi. *iterate the process*

(b) management control domains:
i. planning and organization: project definition; user requirements definition; and system requirements definition
ii. acquisition and implementation: user requirements definition; system requirements definition; analysis and design; and system build/prototype/pilot
iii. delivery and support: analysis and design; system build/prototype/pilot; implementation and training; and sustainment
iv. monitoring: project definition; user requirements definition; system requirements definition; analysis and design; system build/prototype/pilot; implementation and training; and sustainment

120. SDLC models:
(a) sequential or big-design-upfront (BDUF) models:
- Waterfall model: Stepwise refinement; also see “Sashimi model” (modified waterfall model)

(b) iterative and incremental development (IID):
- agile software development:
  - eXtreme Programming (XP) — see elaboration below
  - Scrum
  - Crystal Clear
  - Feature Driven Development (FDD)
  - values captured by the Agile Manifesto:
    - `individuals and interactions` over processes and tools
    - `working software` over comprehensive documentation
    - `customer collaboration` over contract negotiation
    - `responding to change` over following a plan
  - principles behind the Agile Manifesto:
    - customer satisfaction by rapid, continuous delivery of useful software
    - working software is delivered frequently (weeks rather than months)
    - working software is the principal measure of progress
    - even late changes in requirements are welcomed (this does not mean code and run. Instead removing an existing feature or moving a deadline forward to accommodate late/unplanned feature requests)
    - close, daily cooperation between business people and developers
    - face-to-face conversation is the best form of communication (Co-location)
    - projects are built around motivated individuals, who should be trusted
    - continuous attention to technical excellence and good design
    - simplicity
    - self-organizing teams
    - regular adaptation to changing circumstances
  - Agile Modeling (AM): http://www.agilemodeling.com/
- Spiral model: incremental model
- Rational Unified Process (RUP), which is a subset of the Unified Software Development Process (or Unified Process)

- lean software development
- Goal-Driven Software Development Process (GDP)
- Adaptive Software Development
- *Wheel And Spoke Model* (sort of)
- From Toyota Production System (TPS): Muda, :
  - i. Muda is a traditional Japanese term for an activity that is wasteful and doesn't add value or is unproductive, etymologically none or un-useful in practice or others.
  - ii. Mura is traditional general Japanese term for unevenness, irregularity, or inconsistency in physical matter or human spiritual condition.
  - iii. Muri (unreasonable) is a Japanese term for overburden, unreasonableness, or absurdity. Muri can be avoided through standardized work.
  - iv. Three types of waste associated with TPS: muda, mura, and muri.
  - v. Also, see lean manufacturing, lean software development, and total quality management (TQM).

(c) rapid application development (RAD); include the derivative "Dynamic Systems Development Method (DSDM)"; that is, DSDM is based on RAD

(d) V-model: also see Dual Vee Model

(e) Chaos model: Chaos strategy (extends the stepwise refinement strategy)

121. eXtreme Programming (XP):
  (a) Compare this with "flexible product development" outside the software industry
  (b) goals of XP:
    - i. reduce the cost of change (the main aim of XP)
    - ii. An attempt to reconcile humanity and productivity
    - iii. A mechanism for social change
    - iv. A path to improvement
    - v. A style of development
    - vi. A software development discipline

  (c) values of XP:
    - i. Communication
    - ii. Simplicity
    - iii. Feedback
    - iv. Courage
    - v. Respect

  (d) XP is defined by its rules, rather than just by its rules and practices:
    - i. some XP practices are fuzzy in definition
    - ii. you can be doing XP without following all the practices
    - iii. the Rules of Play are what make XP unique
    - iv. following the Rules of Play is Extreme Programming
    - v. following the Rules of Play and the Rules of Engagement is Extreme Software Development

  (e) rules of engagement:
    - i. collaboration between business people and developers: they must work together daily throughout the project
    - ii. primary priority is customer satisfaction : the customer must set and continuously adjust the objectives and priorities based on estimates, and other information provided by the developers (or other members) of the team; objectives are defined in terms of what must be accomplished, not how will things be accomplished
    - iii. deliver working software frequently: the frequency can range in the order of several weeks/months; preferences are given to the shorter time scale (Timeboxing)

- iv. working software: the primary measure of progress
- v. global awareness: At any point, any member of the team must be able to measure the teams progress towards the customers objectives and the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly
- vi. the team must act as an **effective social network**, which means:
    - A. honest communication leading to continuous learning and an emphasis on person-to-person interaction, rather than documentation
    - B. minimal degrees of separation from what is needed by the team to make progress and the people/resources that can meet those needs
    - C. alignment of authority and responsibility

(f) rules of play:

- i. **continuous testing**: work produced must be continuously validated/by testing
- ii. **clearness and quality of code**: all code written for potential use in the software product must clearly express every concept and have clarity, contain no duplication and no superfluous parts and pass all the unit tests.
- iii. **common vocabulary**: there is a sketch of the product guide all development activities with a simple shared story of how the whole system works; so everyone involved could grasp the essence of the project in a term universally understood
- iv. **everybody has the authority**: everybody has the authority to do any task, and at least two people have the understanding necessary to do any task

(g) principles that form the basis of XP:

- i. feedback
- ii. assume simplicity
- iii. embrace change

(h) practices of XP:

- i. fine scale feedback:
    - A. pair programming
    - B. planning game
    - C. test-driven development
    - D. whole team
- ii. continuous process:
    - A. continuous integration
    - B. refactoring or design improvement
    - C. small releases
- iii. shared understanding:
    - A. coding standards
    - B. collective code ownership
    - C. simple design
    - D. system metaphor
- iv. programmer welfare:
    - A. sustainable pace
- v. modularity (facilitate refactoring and simplify design)
- vi. bottom-up (unit testing, module testing, integration testing, and automated regression testing)
- vii. incremental design (regular builds/releases)

(i) resources

- i. http://princetonacm.acm.org/downloads/extreme_agile.html
- ii. http://xprogramming.com
- iii. http://www.extremeprogramming.org/
- iv. http://c2.com/cgi/wiki?ExtremeProgramming
- v. http://www.industrialxp.org/

vi. http://ootips.org/xp.html

122. software prototyping:
    (a) throwaway/rapid prototyping, or close ended prototyping: costly activity, prototype is not used again, or to facilitate the development of the software product
    (b) evolutionary/breadboard prototyping

123. process models:
    (a) CMMI (and CMM):
        i. Capability Maturity Model Integration (CMMI)'s areas of interest:
            A. CMMI for Development (CMMI-DEV): Product and service development
            B. CMMI for Services (CMMI-SVC): Service establishment, management, and delivery
            C. CMMI for Acquisition (CMMI-ACQ): Product and service acquisition
        ii. levels of process maturity (in CMMI) for an organization:
            A. Level 5: `Optimizing`:
                - Focus: continuous process improvement
                - process areas:
                    - organizational innovation and deployment
                    - causal analysis and resolution
            B. Level 4: `Quantitatively Managed`:
                - Focus: quantitative management
                - process areas:
                    - organizational process performance
                    - quantitative project management
            C. Level 3: `Defined`:
                - Focus: process standardization
                - process areas:
                    - requirements development
                    - technical solution
                    - product integration
                    - verification
                    - validation
                    - organizational process focus
                    - organizational process definition
                    - organizational training
                    - integrated project management
                    - risk management
                    - decision analysis and resolution
            D. Level 2: `Managed`:
                - Focus: basic project management
                - process areas:
                    - requirements management
                    - project planning
                    - project monitoring and control
                    - supplier agreement and management
                    - management and analysis
                    - process and product quality assurance
                    - configuration management
            E. Level 1: `Initial`:
                - Focus: competent people and heroics
                - competent people and heroics
        iii. levels of process maturity (in CMM) for an organization:
            A. Initial (`ad hoc`, `chaotic`, heroic) the starting point for use of a new process — (Level 1)

B. `Repeatable` (project management, process discipline) the process is used repeatedly — (Level 2)
C. `Defined` (institutionalized) the process is defined/confirmed as a standard business process — (Level 3)
D. `Managed` (quantified) process management and measurement takes place — (Level 4)
E. `Optimizing` (process improvement) process management includes deliberate process optimization/improvement — (Level 5)

iv. Key Process Areas (KPAs) for characterizing CMM maturity level:
A. Goals
B. Commitment
C. Ability
D. Measurement
E. Verification

(b) ISO/IEC 15504, also known as SPICE (Software Process Improvement and Capability Determination):

i. capability levels and process attributes:
A. Optimizing process (Level 5)
B. Predictable process (Level 4)
C. Established process (Level 3)
D. Managed process (Level 2)
E. Performed process (Level 1)
F. Incomplete process (Level 0)

ii. process dimension: process categories
A. engineering
B. management
C. organizational
D. customer-supplier and/or acquisition supply
E. support
F. operations

iii. process attributes:
A. Process Performance
B. Performance Management
C. Work Product Management
D. Process Definition
E. Process Deployment
F. Process Measurement
G. Process Control
H. Process Innovation
I. Process Optimization
J. four-point (N-P-L-F) rating scale for process attributes:
- Not achieved (0 - 15%)
- Partially achieved (>15% - 50%)
- Largely achieved (>50%- 85%)
- Fully achieved (>85% - 100%)

---

124. **Programming Paradigm**
125. use declarative programming to improve the productivity of software development

---

126. **Advice and Tips on Computer Programming**
127. when writing computer programs, place the constants on the left-hand side (LHS) when comparing a variable and a constant; this will avoid confusion with assignment statements, where the

constants are always on the right-hand side (RHS)

---

128. **Project Life Cycle and Software Release Life Cycle**
129. product life cycle phase and project life cycle:
    (a) inception
    (b) first development
    (c) major release
    (d) minor release
    (e) bug fix release
    (f) maintenance
    (g) obsolescence
130. software release life cycle:
    (a) software release life cycle: composed of different stages that describe the stability of a piece of software, and the amount as the development process proceeds
    (b) software release: the distribution (whether public or private) of an initial or upgraded version of a computer software product
    (c) release stages of software development (in the development and testing period):
        i. Pre-alpha release:
            A. nightly builds
            B. development release
        ii. release development stage, near the end of a release cycle:
            A. stability of code:
                - unstable release
                - stable release
            B. Alpha release; e.g., 3.0.a, 3.0-a, 3.0.0
            C. Beta release:
                - closed beta
                - open beta
                - Beta 1 release; e.g., 3.0.b1, 3.0.b, 3.0-b1, 3.0.1
                - Beta 2 release (with some bug fixes); e.g., 3.0.b2, 3.0.b2, 3.0-b2
                - *Beta 3 release*
            D. release candidate (RC):
                - Gamma release; e.g., 5.0rc1, 5.0-rc, 5.0.2
                - Delta release; e.g., 5.0rc2, 5.0-rc2
                - ... till *Omega release*
                - `Gold master (GM) release / GM seed` – Apple, Inc.
                - A release is called code complete when the development team agrees that no entirely new source code will be added to this release:
                    – There may still be source code changes to fix defects
                    – There may still be changes to documentation and data files, and to the code for test cases or utilities
                    – New code may be added in a future release
            E. *RTM: Release to Marketing / Release to Manufacturing*
            F. *General Availability*
            G. *production release, live release, commercial distribution, or Gold release*; e.g., 5.0r, 5.0-r, 5.0.3
            H. commercial distribution with many bug fixes; e.g., 5.2r5, 5.2-r5
            I. 1.0, 2.0, 3.0, ... ; X.0, where $X \in \mathbb{N} = \{1, 2, 3, ...\}$
            J. Version numbers (sequence-based identifiers for software versioning):
                - sequence-based identifiers for software versioning: each software release is assigned a unique identifier that consists of one or more sequences of numbers or letters

- schemes vary widely in areas such as::
  - the quantity of sequences
  - the attribution of meaning to individual sequences
  - the means of incrementing the sequences
- examples of schemes:
  - major.minor[.build[.revision]]
  - major.minor[.maintenance[.build]]
- increment the first number (**major number**), when significant changes are made to the features of the software
- increment the second number (**minor number**) for minor changes:
  - even second number are stable releases
  - odd second number are unstable releases
  - distribute release notes with these releases; see `http://en.wikipedia.org/wiki/Release_notes`
- increment the third number (**revision number**) for:
  - bug fixes
  - change to the user interface (UI)
  - change to the documentation
- increment the fourth number (**trivial version number**) for `very minor changes`
- Other software versioning schemes:
  -

(d) In software development, a rolling release approach refers to a continuously developing software system, as opposed to one with versions that must be reinstalled over the previous versions... A rolling release is typically implemented using small and frequent updates.

---

131. **Software Portability**
132. for `Ruby` programs, Jruby is a compiler that can generate Java bytecode; likewise, Jython can do the same for Python programs; this is because Java bytecode instructions are machine independent, while conventional processor dependent machine code isn't
133. write cross-platform (or multi-platform) software: write once, compile anywhere (WOCA)

---

134. **Software Metrics**
135. learn to develop metrics for your goals and objectives, determine the error tolerance, make appropriate trade-offs, and measure them

---

136. **Software Engineering Practices**
137. pair programming
138. programming productivity
139. software reuse
140. systems integration

---

141. **Software Engineering Resources**
142. software engineering textbooks: Stephen R. (Steve) Schach, Ian Sommerville, Roger Pressman

---

143. **Guidelines for Algorithm Design**
144. Whenever you are asked to design a new algorithm, there are a number of things you should do. This document itemizes your responsibilities as an algorithm designer. In particular, you are expected to do each of the following tasks whenever writing up a solution to design problem in CMSC-441 or CMSC-641:

(a) Give your algorithm a name.
(b) State the input/output specifications of the algorithm.
(c) Write your algorithm in clear HIGH-LEVEL pseudocode. Your pseudocode should be detailed enough that a programmer could easily translate it into a high-level programming language; however, your pseudocode should not be so detailed as to look like low-level C code. Write your code as clearly and in as high a level as possible.
(d) Briefly and intuitively explain how your algorithm works. State the design strategy; do not explain your pseudocode line-by-line.
(e) Prove that your algorithm works correctly.
(f) State and prove the resource usage of your algorithm, including its running time and memory usage.
(g) State and prove any significant special properties of your algorithm.
(h) Extra notes:
    i. Justify the steps/approach of my algorithm. Why did I do that? How is this beneficial?
    ii. Elementary steps should be used in the pseudocode. Complex sequences of elementary steps should be wrapped as functional calls.
    iii. Describe unambiguously each procedure with a finite number of steps to solve my research problems.
    iv. Use examples to adequately explain my proposed techniques.
(i) Alan T. Sherman, "Algorithm's Policy," *Prof. Alan Theodore Sherman's web page: Courses: How To's and Other Generic Course Documents*, Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, February 15, 1997. Reference: Available online at: http://www.csee.umbc.edu/~sherman/Courses/documents/algorithms_policy.html; last accessed on February 8, 2012.

---

145. **General Advice and Random Notes**
146. learn how to write production-quality code, and develop industrial-grade EDA tools:
    (a) What is production-quality code?
    (b) How do I write production quality code?
    (c) What makes an EDA tool industrial-grade?
    (d) How do I develop industrial-grade EDA tools?
    (e) e.g., develop "leading-edge industrial-grade VLSI Physical Design tools"; see http://vlsicad.eecs.umich.edu/BK/PDtools/; last viewed June 21, 2009 @ 0645 hrs
147. keep working on improving my development speed:
    (a) the faster I can develop software, the faster I can try and implement new algorithms/heuristics to solve research problems, and the more publications I can churn out
    (b) also, development speed is dependent on the choice of computer language used:
        i. programming languages:
            A. `C++`
            B. `Java`™
            C. `MATLAB`®
            D. `C#`
            E. `Fortran` (previously `FORTRAN`)
            F. `C`
            G. METAPOST
            H. visual programming language (VPL):
                - `Simulink`®
                - `LabVIEW`™

I. assembly language:
   - `MIPS`
   - `DLX`
   - `Motorola 68000` / Motorola 68k / Motorola 680x0

J. curly bracket programming languages

ii. scripting languages:
   A. `Ruby`
   B. `Python`: off-side rule
   C. `Perl`
   D. `Tcl` (Tool Command Language)
   E. `UNIX shell scripts`:
      - `Bash`: Bourne-again shell
      - `Bourne shell (sh)`
      - `C shell (csh)`
      - `tcsh`

   F. `AWK`
   G. `sed`
   H. `PHP`
   I. `SenseTalk`™
   J. `AppleScript`®

iii. hardware description languages (HDLs):
   A. `Verilog`
   B. `Verilog-AMS`
   C. `SystemVerilog`™
   D. `SystemC`™
   E. `SystemC-AMS`™
   F. `VHDL`
   G. `VHDL-AMS`
   H. See the respective Language Reference Manuals (LRMs)s for more details of these HDLs

iv. query languages:
   A. `SQL`

v. modeling languages:
   A. `UML`, which is also a specification language and architecture description language, ADL

vi. markup language:
   A. LaTeX
   B. `HTML`
   C. `XML` (Extensible Markup Language)

vii. declarative language:
   A. `OCL`: Object Constraint Language

148. use macros to speed things up; write/develop them for my text editor and/or my (computer) program

149. use `Automator` on `Mac OS`® X to implement point-and-click (or drag-and-drop) creation of sequential workflows for automating repetitive tasks:
   (a) Automator enables the repetition of tasks across a wide variety of programs
   (b) Automator uses `AppleScript`® and `Cocoa`®

150. How should I measure programming productivity?
   (a) amount of code that can be created or maintained per programmer (often measured in KLOCs per day)
   (b) detecting and avoiding errors (through techniques like Six Sigma management, Zero Defects coding, and Total Quality Management)

   (c) software cost estimation (cost being a direct consequence of productivity)

151. AVOID the use of anti-patterns (or antipatterns) AT ALL COSTS
152. never ever knowingly join a death march project:
   (a) In the software development and software engineering industries, a death march is a dysphemism for a project that is destined to fail.
   (b) Usually it is a result of unrealistic or overly optimistic expectations in scheduling, feature scope, or both, and often includes lack of appropriate documentation, or any sort of relevant training.
   (c) The knowledge of the doomed nature of the project weighs heavily on the psyche of its participants, as if they are helplessly watching the team as it marches into the sea.
   (d) Often, the death march will involve desperate attempts to right the course of the project by asking team members to work especially grueling hours, weekends, or by attempting to "throw (enough) bodies at the problem" with varying results, often causing burnout.
153. Brooks's law is a principle in software development which says that "adding manpower to a late software project makes it later":
   (a) "ramp up" time: It takes some time for the people added to a project to become productive
   (b) Communication overheads increase as the number of people increase
154. detect **code smells**, and make appropriate corrections promptly
155. avoid "development hell," where the software is stuck in the implementation phrase for a considerably long time; if I am in charge if such a project, should it be shelved?
156. beware of the **software Peter principle**:
   (a) The software Peter principle is used in software engineering to describe a dying project which has little by little become too complex to be understood even by its own developers.
   (b) It is well known in the industry as a silent killer of projects, and by the time the symptoms arise it is often too late to do anything about it.
   (c) Good managers can avoid this disaster by establishing clear coding practices where unnecessarily complicated code and design is avoided.
   (d) The Peter Principle is the principle that "In a Hierarchy, Every Employee Tends to Rise to His Level of Incompetence."
157. The Dilbert Principle refers to a 1990s satirical observation by Dilbert cartoonist Scott Adams stating that companies tend to systematically promote their least-competent employees to management (generally middle management), in order to limit the amount of damage they're capable of doing:
   (a) In the Dilbert strip of February 5, 1995 Dogbert says that "leadership is nature's way of removing morons from the productive flow."
158. Parkinson's Law is the adage first articulated by Cyril Northcote Parkinson as the first sentence of a humorous essay published in The Economist in 1955: Work expands so as to fill the time available for its completion.
159. Parkinson's Law of Triviality (also known as the bicycle shed example, and by the expression colour of the bikeshed) is C. Northcote Parkinson's 1957 argument that organisations give disproportionate weight to trivial issues.
160. Student syndrome refers to the phenomenon that many people will start to fully apply themselves to a task just at the last possible moment before a deadline. This leads to wasting any buffers built into individual task duration estimates.
161. reasons to rewrite a function/method, or class:

(a) When the source code to be able to extend an existing program is not available.
(b) When the source code is available under an incompatible license.
(c) When the code cannot be adapted to a new target platform.
(d) When the existing code has become too difficult to handle and extend.
(e) When the task of debugging the existing code seems too complicated.
(f) When the programmer finds it difficult to understand the source code.

162. the contents of a readme (or read me) file may contain:
    (a) configuration instructions
    (b) installation instructions; can be placed in the file `install.txt`
    (c) operating instructions
    (d) a file manifest
    (e) copyright and licensing information; can be placed in the file `license.txt`
    (f) contact information for the distributor or programmer
    (g) known bugs; can be placed in the file `bugs.txt`
    (h) troubleshooting
    (i) credits; can be placed in the file `authors.txt`
    (j) acknowledgments; can be placed in the file `thanks.txt`
    (k) a changelog; can be placed in the file `news.txt` or `changelog.txt`
163. Prefactoring is the application of past experience to the creation of new software systems. Its relationship to its namesake refactoring is that lessons learned from refactoring are part of that experience.
164. for each software I develop/co-develop, consider providing support for encryption/decryption of source files, and other files
165. goals of each software project:
    (a) high quality software
    (b) meet or exceed customer expectations
    (c) reach completion [of the project] within time and cost estimates
    (d) work effectively and efficiently in the current and planned Information Technology infrastructure
    (e) is inexpensive to maintain
    (f) cost-effective to enhance
166. become a subject matter expert:
    (a) A Subject Matter Expert (SME) is a person who is an expert in a particular area.
    (b) In software engineering environments, the term is used to describe professionals with expertise in the field of application but without technical project knowledge.

---

167. **Checklists**
168. **When comparing my formal verification (or EDA) tools/solutions to others, do other tools/solutions provide comparable or better capabilities?:**
    (a) **Do others need a set of tools to mach what my tool can do?**
    (b) **Do my tools have better performance?**
    (c) **Do my tools have better memory capacity/usage?**
    (d) **Can my tools handle bigger inputs (such as design models or netlists)?**
    (e) **Do my tools scale better for a bigger parallel computing platform?**

5.9 Important Considerations for Developing EDA Tools

Important considerations for developing EDA tools:

1. How can I turn on/off different parts of the code?
   (a) Can I use this to facilitate debugging? In particularly, I want to know if I can use different namespaces for different parts of a C++ program to facilitate debugging.
2. Within a C++ class, use:
   (a) Copy constructor [14, pp. 5-6 of the Introduction]:
      i. Initialize object with a different object of the same type
      ii. Widget(const Widget& rhs);
      iii. When called, we are "passing-by-value"

   (b) Copy assignment operator [14, pp. 5-6 of the Introduction]:
      i. Copy the value from one object to another object of the same type
      ii. Widget& operator=(const Widget& rhs);
      iii. When called, we "pass-by-reference-to-const"

   (c) Function objects [14, pp. 6 of the Introduction]:
      i. Objects that act like functions
      ii. Overload the function call operator `operator()` in the class definition

   (d) Function pointers

   (e) Inline function [14, Chapter 1, pp. 16–17]:
      i. Has the efficiency of C/C++ macros
      ii. Has predictable behavior
      iii. Has type safety of regular function
      iv. Parameters of the function would not be evaluated multiple times
      v. Parameters inside the function body need not be parenthesized
      vi. It obeys scope and access rules
      vii. *inline void callWithMax(const T& a, const T& b)* { *f(a > b ? a : b);* }

   (f) The `const` keyword can be applied to [14, Chapter 1, pp. 17]:
      i. outside of classes:
         A. constants at global or namespace scope
         B. objects that are declared static at file, function, or block scope
      ii. inside classes:
         A. static data members, and pointers to these members
         B. non-static data members, and pointers to these members
         C. within a function declaration [14, Chapter 1, pp. 18]:
            - A function's return value
            - Individual parameters: pass-by-reference-to-const, or pass-by-pointer-to-const
            - The entire function (for member functions); i.e., const member functions [14, Chapter 1, pp. 19] — For bitwise constness (don't modify any of the object's non-static data members-here, object refers to an instance of the class that the function belongs to); and for logical constness [14, Chapter 1, pp. 21]
      iii. The `const` keyword can be listed before or after the data type; in either case, they have the same meaning [14, Chapter 1, pp. 18]
      iv. iterators (i.e., `const_iterator`) – Iterators that point to things that can't be modified [14, Chapter 1, pp. 18]

(g) Template metaprogramming (TMP) [14, Chapter 7, pp. 233–238, Item 48]

(h) [17]
(i) More Effective C++ [12]
(j) STL [13]

5.10 Notes on VLSI Design

Notes on VLSI design:

1. using VIM for `Verilog`, `C`, `Perl`, `csh/tcsh`, & `Tcl`; see `http://www-vlsi.stanford.edu/~jsolomon/vim/`
2. `Perl` scripts to drive `Synopsys` tools; see `http://www-vlsi.stanford.edu/~jsolomon/SPP/`
3. learn how to use and tweak configuration management tools, such as `CVS`, `Subversion`, and `Git`, for VLSI design management:
   (a) configuration management tools are also known as:
      i. revision control system
      ii. version control system
      iii. source control system
      iv. (source) code management (SCM) system
   (b) design management involves and contains:
      i. change management
      ii. handling of Engineering Change Orders (ECOs), including:
         A. bug reports
         B. feature requests
         C. database for all filed ECOs and processed ECOs
         D. corrected faults/bugs
      iii. a framework that supports all the EDA tools that are used by the IC design, verification, testing, and validation teams
      iv. netlists, source code (e.g., HDL, test benches, & HVL), simulation scripts (e.g., `SPICE` deck, `IRSIM` test bench), system-level models (@ the architectural- or microarchitectural-level)
      v. project documentation:
         A. design document: including architectural and microarchitectural designs, RTL/logic designs (schematic), & floorplan
         B. verification plan
         C. testing plan
         D. post-silicon validation plan
         E. user manual for chipset / chip specification document
         F. project management plan: e.g., timeline and/or Gantt chart
         G. scripts that glue the tools together, and scripts that are used to drive/run the tools
         H. scripts that are used to manipulate source code and netlists: e.g., parsers
         I. versions and setup environment of EDA tools
      vi. bug/fault tracking: track the type, severity, and number of bugs/faults
      vii. different builds, subversions, and versions of design, verification, and simulation files, as well as project documentation
   (c) companies that provide VLSI (or IC) "design management systems" include: IC Manage and Cliosoft

5.11 Note on Using the OpenAccess SDK by Donald Chai

A while ago, I needed to use the OpenAccess SDK to write some code to do some physical design. I tried to get the much touted Python wrappers to work, but they were for OA2.1, while I was using OA2.1.1. The OA API changed just enough to foil any attempts at compilation. Since those wrappers were a big monolith of generated code, I figured I'd probably be better off writing my own.

I chose SWIG over Boost.Python because of the more advanced typemaps. I actually like Boost.Python better, because it avoids shadow classes, but SWIG's typemaps seemed like the right thing for wrapping someone else's code. Not all of the API is wrapped, because I didn't need all of the API. (Finishing it is left as an exercise to the reader.)

– Donald Chai; then a EECS Ph.D. candidate @ UC Berkeley under the supervision of Prof. Andreas Kuehlmann; see `http://www.cs.berkeley.edu/~donald/code/pyoa/`; last viewed in mid 2009

5.12 Publishing Research Papers in EDA

Publishing research papers in EDA:

- use DUDE to check for the amount of new material for publication; see `http://ariane4.eecs.umich.edu/DUDE/`
- join/create reading groups for reviewing and discussing journal and conference papers related to my research area; e.g., reading groups for VLSI formal verification, SMT/SAT, MPSoC, & NoC
- Pat Hanrahan says: "Future Work section must be earned". If you haven't made us care about your contribution thus far, we won't care to read `Future Work` either.
- analog formal verification: Quo Vadis? — Where are we going with analog formal verification? What are the research directions for analog formal verification?
- **When writing journal and conference papers, turn your shortcomings into advantages. For example, "while it cannot do Task A, it is acknowledged that Task B is more important than Task A. So, it doesn't matter if Task A can be done very well, or done at all."**
- Nature Publishing Group (a division of Macmillan Publishers Limited):
  1. Kendall Powell, "Publications: Publish like a pro," in *Nature*, Vol. 467, No. 7317, pp. 873–875, October 14, 2010. Available online at: `http://dx.doi.org/10.1038/nj7317-873a`; last accessed on December 31, 2010.
     - Find out how do prolific researchers get their manuscripts noticed, reviewed, and approved for publication
     - Take rejection of your journal/conference paper submission "at face value."
     - Prolific researchers get more papers published, since they are more persistent in trying to get the paper published.
     - "If you've got the gut feeling that you've got good stuff, you've just got to be persistent." – Matt Rayner
     - "The refereeing process can be haphazard" – Mark Hauber
     - "The moral of the story is that the publishing process requires not only hard work but also resilience – and struggling young authors can learn valuable lessons from those who have already navigated that process."
     - Quote: In taking the all-important route to publication, inexperienced authors have several factors to consider and obstacles to overcome, from finding ways to combat writer's block to gracefully pursuing the journal-submission and review process. Established authors and journal editors suggest thinking early about the right journal and finding an

appropriate editor, the best reviewers and, of course, an appropriate audience of readers. And new authors should be careful to polish their work and respond meticulously and politely to reviewers' comments without getting overwhelmed or frustrated by lengthy, time-consuming queries. Those who follow such advice are more likely to find success. Those who don't could end up on the wrong side of the 'publish or perish' divide.

– Quote: For many publishing veterans, the writing process starts at the earliest stages of designing a research project. "Nothing beats a comprehensive, thought-out experiment. Do that up front and your writing will come so much more easily," says Mark Blumberg, a neuroscientist at the University of Iowa in Iowa City and editor-in-chief of Behavioral Neuroscience. It can be helpful to think of the project as a tentative title for the article it will become, a constant reminder of the scope it should span, recommends Bill Nazaroff, an environmental engineer at the University of California, Berkeley.
– Quote: Eileen White, associate director of the Cancer Institute of New Jersey in New Brunswick and a senior editor at Cancer Prevention Research, says that researchers should have a "neat package" of an interesting question, experiments to test it and a final answer. And a key to winning over editors and reviewers, says White, is having strong data to support conclusions. "Some people don't appreciate the fact that a lot of weak data does not make up for having less, but more powerful, data," she says.
– Quote: Before starting to write the paper, authors should carefully choose a journal audience for their research story – and initially aim for the highest-impact, highest-profile journal possible. "The submission process is fast enough today; it's worth the effort of sending your paper to the highest journals where it belongs," says Hauber.
– It is better to have a smaller number of publications of outstanding quality than many decent publications, since recruitment of researchers and tenure-track faculty focuses on high-impact publications.
– "Junior scientists should be thinking about the one paper that they'll place proudly at the top of their CV, job applications and grant proposals." – Mark Hauber
– Quote: When should the writing begin? Research presented at a meeting or as a poster is almost certainly far enough along to begin writing up. But some researchers say that it is never too early to start – students should be writing a little bit every day. Keeping a folder of pertinent literature and beginning with a simple outline of relevant points gleaned from that literature can provide the essential elements of a paper's introduction. Likewise, taking time during field or bench work to write short chunks of the materials and methods used can help to document these activities while they are fresh.
– Writers' tips: The key to effective writing:
  (a) You are only as good as your last paper – previous success does not guarantee future acceptance.
  (b) You've got to hook the editor with the abstract.
  (c) Don't delete those files. Keep every version. You never know what aspect you can use for some other piece of writing.
  (d) Writing is an amazingly long learning curve. Many authors say that they're still getting better as a writer after several decades.
  (e) The most significant work is improved by subtraction. Keeping the clutter away allows a central message to be communicated with a broader impact.
  (f) Write every day if possible.
  (g) Once you've written what you wanted to convey, end it there. K.P.
– Look at papers from the same journal to determine their length and organization. Gauge the length of the paper and sections by the number of paragraphs. Determine the number of figures and references in the papers. Break up the "daunting sections" "into more

manageable portions".

- To overcome writer's block, start by carrying out the easier tasks of writing first. E.g., contacting a collaborator, "checking a reference," or edit material that was written the previous day.
- "Recognizing that writing is a long process is valuable. Find a mentor in that process, somebody to guide and coach you." – Bill Nazaroff
- Be aware of my working patterns, and write when I am most productive. Get myself into the writing zone, and get stuff done.
- "Be clear and concise and use simple language whenever possible."
- "Important but poorly written papers could end up being sent back unreviewed by busy editors."
- "Editors stress the importance of clarity above all else, to help convey arguments and logic to them and to readers. They say that most writers make the mistake of assuming too much knowledge on the part of their audience. In reality, even at the most specialized journals, only a handful of readers will be such close colleagues that they don't need any contextual set-up."
- Quote: Editors say that one way to identify holes or gaps in logic that would be vulnerable in peer review is to imagine a skeptical audience reading the manuscript. "Think of the most adversarial reader you can imagine, and write to substantiate the veracity of your arguments and to anticipate criticisms and answer them," says Wojtal. Some editors suggest that 'winning over' a skeptical editor, reader or reviewer should be the ultimate goal of any paper's abstract. "Editors read the abstract and start formulating a thumbs-up or thumbs-down, looking for reasons to rip it apart," says White. "You want them to form a positive point of view from the very beginning," she says. Leslie Sage, an astronomy editor at Nature, says authors should avoid an abstract structure that says: we did X, which told us Y, and has implications for Z. Instead, he says, start with why a reader should care about learning more about Z and then explain how this work furthers that goal.
- Quote: Likewise, says Nazaroff, the introduction should persuade readers "that you know what you are talking about and have something new to teach them." Wojtal also advises authors to clearly distinguish between data in the results section and inferences about what they mean in the discussion section. This way, even if an editor or reviewer does not agree with a lab's interpretation of the work, he or she may still see the need to publish such an important data set.
- Quote: Often, less is more for junior scientists crafting manuscripts. The introduction need not cite every background article gathered, the results section should not archive every piece of data ever collected, and the discussion is not a treatise on the paper's subject. The writer must be selective, choosing only the references, data points and arguments that bolster the particular question at hand.
- "Even polished authors go through an average of 10-12 drafts, and sometimes as many as 30."
- Quote: Writers should ask not only the principal investigator to view drafts, but also every co-author, as well as fellow students or postdocs, and colleagues outside the immediate field of research. Lead authors should give co-authors set deadlines of 10 days to two weeks to suggest changes. Experienced authors counsel letting the draft sit for a few days before reading it with fresh eyes to catch mistakes or problems in flow. Blumberg prefers to read drafts aloud with his students to spot errors.
- Quote: When the paper is ready to submit, says Wojtal, the author should devise a cover letter that includes a brief synopsis of the article's argument, and suggestions for a few

potential reviewers, as well as those who should be excluded. Such information, he says, can be very helpful to busy editors, who want to know who is familiar with the work and will be easy to reach. Authors should not suggest reviewers who are personal friends or institutional colleagues; including those people could immediately erode the editor's trust. Authors need to find a balance – it is fine to exclude a couple of reviewers who are direct competitors or known naysayers, but restricting too many qualified reviewers can backfire. "As an author, your job is to make the editor's job as easy as possible," says Blumberg.

- Address each comment of the reviewers step-by-step. Work through the lists of comments by the reviewers, "and explain how the criticisms were addressed, or why they were not, in the resubmission cover letter to the editor."
- "A clear, succinct resubmission letter may result in an editor making their own decision rather than sending the paper back out for another round of review."
- "The worst thing an author can do is to ignore a reviewer's criticism and send it back without an explanation." – Steven Wojtal
- "If you strongly believe the reviewers have erred and that the editor should hear from you, definitely send an e-mail ... Be as polite as possible, stick to the facts, and keep it to the point." – Mark Blumberg
- "Young scientists would be wise to embrace written communication as the foundation of an academic career and the key to earning tenure, winning funding and, ultimately, sustaining a research program ... If your result is not published, you haven't done anything ... You might not set out to be an enthusiastic writer, but you should try to learn to love it." – Mark Hauber

2. David A. Mackey, "Recruiters & Academia: Training peer reviewers," in *Nature*, Vol. 443, No. 7113, pp. 880, October 19, 2006. Available online at: http://dx.doi.org/10.1038/nj7113-880b; last accessed on January 1, 2010.
3. Marwan Azar, "Prospects: The method and madness of publishing," in *Nature*, Vol. 464, No. 7289, pp. 799, April 1, 2010. Available online at: http://dx.doi.org/10.1038/nj7289-799b; last accessed on January 1, 2010.
   (a) Quote: The process of publishing a scientific paper is twofold. Most important, it involves the attainment, integration, assessment and correction of knowledge, a process more commonly known as the 'scientific method'. But it also requires navigating a minefield of administrative and bureaucratic issues.
   (b) "But although conducting the science that goes into writing a paper is rewarding, publishing has, unfortunately, as much to do with exasperating administrative issues."
   (c) "their findings can be distilled into a cogent, cohesive and engaging scientific article" [Distill my research findings into "a cogent, cohesive, and engaging scientific article."]

Notes on writing research papers:

1. Explicitly address the following questions in the introduction of the paper in a succinct manner:
   (a) What is the problem?
   (b) Why is the problem important?
   (c) What has been done so far on the problem?
   (d) What is the contribution of the paper on the problem?
   (e) Is the contribution original? Explain why.
   (f) Is the contribution non-trivial? Explain why.

5.13 Giving a Research Presentation on an EDA Topic

Giving a research presentation on an EDA topic:

- **Introduction phase of the presentation**
- Before telling me what you did, tell me why I should care [about your research].
- **General advice for giving presentations**
- A great speaker could make any topic seem interesting and accessible, and some of the talks I ended up enjoying the most were ones I thought I wouldnt even understand!
- During a presentation on my research, answer the following question, "How is the big picture divided into subproblems, and where do you fit in?" Now that the big picture is clear, what are the specific subproblem challenges? Which part of which subproblem are you working on? Some of this will necessarily get into details that not everyone in the audience can follow. If your efforts succeed, what will you have demonstrated? Another way to ask this is: what is the "research question" (or questions) being addressed here? In other words, ten years from now, when the hardware, software, etc. have all changed and the computers of the day make today's computers look like Tinkertoys, what fundamental nugget of an idea will still be considered relevant and applicable? This is often very hard to identify, and it may be that your own piece of the project contributes only a small part toward forming that Big Idea...
- When giving a presentation on my research, take time to explain the motivation of the problem well, and be sure everyone understands (at least at a high level) why it's useful, interesting, etc. This is time well spent: if people don't understand the ultimate goal, they probably won't pay attention to what you did.
- **Advice for the Q&A part of the presentation**
- "Know how to listen, and you will profit even from those who talk badly." – Plutarch, on Terry Tao's blog; see http://mathgradblog.williams.edu/?p=17#more-17
- Valerie M. Balester, "The Perfect Defense: The Oral Defense of a Dissertation," in *YouTube: Texas A&M University, AggieMedia*, Google Inc., Mountain View, CA, October 26, 2010. Available online at: http://www.youtube.com/watch?v=edQv9OKvfdU; last accessed on August 9, 2013. ["Dr. Valerie Balester of Texas A&M University talks about how to prepare and what to expect when defending your dissertation."]
  1. Ask the question to them rhetorically, to slow things down or when I don't know what are they asking about. This gives me sometime to think about the question, and how to answer it.
  2. Correct myself if I need to
  3. I need a little time to answer that. Can I just have a little moment to gather my thoughts?
  4. Take an attempt to answer; try to answer the question...
  5. Thank you for that question. That's a really good point.
  6. They are trying to see if I can think. Verbalize my thought process.
  7. When they ask you to make a revision to your dissertation, ask them to summarize the major points of the revision so that I can take notes about the major revision now.
  8. When you don't know, [you can say]:
     (a) You asked me (repeat the question), and I'm not sure, but I think...
     (b) If I knew x, I would say y.
     (c) I don't know, but that the question has interesting implications, for example, if I knew that I could...

5.14 Skills that EDA Companies Look for

Skills that EDA companies look for:

- **EDA tools for front-end design and verification, and formal verification (system-level, high-level, or behavioral RTL)**
- Excellent knowledge of software design, algorithms and data structures, Verilog/VHDL, and design verification.
- Experience with compiler, simulator or verification tool development is highly desirable.
- In depth knowledge of and experience in using formal methods and analysis is highly desirable
- Excellent oral and written communication skills.
- We are looking for talented developers with research background in formal verification areas, such as SAT solvers, BDD techniques, model checking, and equivalence checking.

- **formal verification**
- Automate regression analysis, test and deployment.
- Enhance the performance of logic gate-level simulators.
- Knowledge in formal verification techniques, SAT based engines, BDD, Symbolic, ATPG
- Practical knowledge of formal verification, simulation, and synthesis protocols
- Research efficient algorithms and implementation of formal engines, including SAT based bounded model checkers, inductive provers, and iterative proof-based abstraction mechanisms

---

- **formal verification**
- Design, develop and debug EDA software used for timing constraint generation, verification and debugging.
- Independently analyze and create solutions for complex software products and will be responsible for all phases of software development from initial problem identification and solution specification to final testing and debugging.
- Related experience in timing analysis and synthesis.
- Must possess very strong expertise in C/C++, and strong experience with Verilog and VHDL languages.
- Must have very strong knowledge of timing constraint and timing exception generation in SDC format.
- Strong experience in and knowledge of EDA, and the digital design process also required.
- Practical knowledge and experience in the development of timing analysis and synthesis a must.
- Strong spoken and written communication skills a requirement.
- Knowledge of logic simulation and formal verification also desirable.
- Knowledge of FPGA synthesis and FPGA timing constraint formats a plus.

- **formal verification**
- Develop advanced timing constraint generation, timing constraint verification, RTL analysis and functional verification tools.
- Work includes RTL synthesis of Verilog, VHDL and System Verilog; and developing timing constraint generation, verification and static RTL analysis tools.
- Strong knowledge of and experience in C/C++ programming and scripting languages (Tcl,Perl) required.
- Previous experience in developing RTL synthesis, analysis or simulation tools required.

- Knowledge of and experience with Verilog and VHDL a requirement.
- You must understand high level model generation, netlist creation, traversal and analysis techniques.
- Knowledge of System Verilog, strong skills in Lex and Yacc, and knowledge of the .lib format are also desirable.
- Knowledge of PSL or SVA assertion languages is a plus.

- **formal verification**
- Atrenta is seeking world-class EDA software architect and developers within the area of formal verification, power optimization using advanced formal techniques, architectural power planning, physical implementation and 3D IC design automation and architecture exploration.
- In-depth knowledge and experience in C/C++ programming, data structures and algorithm complexity analysis
- Excellent knowledge of the design conception flow
- Knowledge and experience with any of the following technologies: SAT, BDDs, ATPG, Model Checking, Bounded Model Checking, Equivalency Checking, Timing Analysis, Simulation, Optimization, Synthesis
- Excellent software engineering skills, very hands on with development tools such as debuggers, purify, etc.
- Familiarity with script languages such as Perl and Tcl/Tk a plus
- Able to learn new technologies and integrate them in to existing products
- Excellent English communications skills and technical interpretation skills
- Solid interpersonal skills and the ability to work with a remote development team
- Engineering degree from a French "Grandes Ecoles" is a plus
- Ph.D. in EE/CE/CS with no industry experience or with industry experience through CIFFRE convention [it should be "CIFRE convention"; CIFRE – Industrial Agreements for Training through Research; Ministry of Higher Education and Research; ANRT – Association Nationale de la Recherche et de la Technologie]
- Familiar with Verilog and VHDL and design tools
- Experience with EDA tools (synthesis, verification, P&R) is a plus
- Able to deliver projects in a timely manner with high quality and take ownership for all deliverables

---

- **digital VLSI formal verification and high-level ATPG**
- Knowledge in formal verification techniques, SAT based engines, BDD, Symbolic, ATPG

- **high-level synthesis**
- Very strong understanding of C/C++
- Experience in the use and development of tools for the Electronic Design Automation (EDA) industry
- Experience in development of complex software projects on Linux and Solaris operating systems
- Understanding of product regression and release procedures
- Strong background in software engineering, data structures and algorithms
- Experience with scripting languages such as TCL or Perl and configuration management with CVS
- Strong documentation skills
- Experience with behavioral synthesis a plus

- Experience with Verilog, hardware simulation, logic synthesis, ASIC / FPGA development a plus

---

- **high-level synthesis**
- This position will require developing software to integrate FishTail products with third-party software components.
- It will require understanding an existing body of C++ code and taking ownership of the code.
- It will require working on user-interface related code.
- The engineer will need to have strong C++ software implementation skills.
- Prior experience in using the Standard C++ library (STL) is required.
- Experience with software development tools (gdb, purify, etc.) on a Linux platform is required.
- Strong algorithmic skills and knowledge of graph traversal algorithms is required.
- Our products deal with large amounts of data and so developing software that is high-performance and is efficient in its memory utilization is key.
- Prior knowledge of digital design and EDA tools is preferred.
- Prior knowledge of System Verilog is preferred.
- The ideal candidate will have an M.S. in Computer Science and prior EDA software development experience.
- Prior startup experience is strongly preferred.
- This position requires working out of our corporate headquarters at Lake Oswego, Oregon.

- **skills, knowledge, and experience for: computational lithography, RET / OPC, lithography simulation, DFM, TCAD**
- background in electromagnetism, optical lithography and imaging
- prior experience programming in MATLAB, and with software packages for lithography and electromagnetic simulations
- the successful candidate will be involved in the understanding and qualification of image formation in advanced lithography, studying the effect of mask material and topography on imaging and across resolution enhancement techniques, and using customized optimization techniques for the design of the mask shapes

---

- **DFM position**
- Develop new EDA (Electronic Design Automation) tools using C, C++ & Tcl in Unix/X-Window environment
- research and develop lithography process modeling algorithms for layout patterning simulation
- Research, design, and develop algorithms and applicable programs for UDSM IC (Ultra-Deep Sub-Micron Integrated Circuit) layout correction including:
  1. OPC (Optical Proximity Correction)
  2. PSM (Phase Shifter Mask)
  3. SRAF (Sub-Resolution Assistant Feature) generation
- Design and implement computational geometry and computer graphics algorithms for IC layout operation, such as:
  1. layout pattern matching
  2. polygon resizing
  3. layout boolean operation
- Research and develop algorithms and computer applications for high-yield IC design and manufacturing

- Design, implement, and support programs for very large scale layout database handling
- develop and design computer applications for handling GDSII and OASIS layout data files
- Develop, support, and maintain graphical user interface for layout debugging and display
- Set up diagnostic routines for applications
- Maintain and support existing layout tools as needed
- MSEE + 2 yrs. exp. Will accept Ph.D in EE in lieu of 2 yrs. exp. with research exp. in optical proximity correction system & lithography simulation system development
- Must be able to perform job duties

- **DFM position**
- Excellent software development skills in C/C++, knowledge of assembly language is a plus
- Good understanding of object-oriented design, data structures and algorithms
- Strong UNIX background, ability to program in scripting languages (sh, perl)
- Experience in network programming (TCP/IP) and multi-threaded applications
- Experience with embedded system and hardware interface is desirable but not required
- PhD in Computer Science, Electrical Engineering, Physics or any related engineering, mathematics or science discipline
- Strong interest and skill in computer programming and a desire to expand existing programming skills
- Excellent problem solving skills
- Ability to excel in cross functional, team environment

- **DFM position**
- Proficiency in C/C++ programming
- Strong analytical skills and high creativity
- Solid background in mathematics and/or physics
- Detail orientated work habit
- Team player
- Proficiency in code optimization
- Knowledge of:
  1. advanced engineering mathematics
  2. numerical algorithms
  3. computer graphics
  4. computational geometry
  5. image and signal processing
  6. statistics
  7. optimization algorithms
  8. optical imaging theory
  9. lithography modeling
- Familiarity with UNIX environment, scripting languages
- Knowledge in semiconductor lithography and IC design

- **DFM position**
- Software Engineering Genius: algorithms
- Experience: Track record of delivering both optimization and deterministic algorithms with business success

- Attitude: 100.000% completion
- Location: San Jose, California or Shin-Yokohama, Japan
- Description: We need a superstar algorithmist. We need to attain near-optimal solutions in reasonable time while assuring 100.000% correctness and completeness. Because of the speed requirement, wherever possible, we need to exploit the narrowness of the space to derive the solution mathematically. In some subspaces, however, an "inverse" solution is not available and we must deploy optimization techniques to "fit". Aspects of this task are like image processing, while other aspects of it are like interconnect routing.

- **DFM position**
- Software Engineering Genius: database and visualization software and architecture
- Experience: Track record of delivering reliable, extensible, and beautifully architected software systems that people love to use
- Attitude: Quiet confidence
- Location: San Jose, California or Shin-Yokohama, Japan
- Description: We need a superstar software engineer. Someone who works well with others, but is creative, innovative, and with a track record of delivering reliable software. Someone that believes that the fastest way to develop code is to have half the code be testing code. Someone who can traverse the hierarchy from the conceptual architecture down to bit-level optimization. Someone who can think like the compiler-optimizer and write his/her own memory allocators for the critical paths, but can be object-oriented, assertion-filled, and self-debugging for a beautifully architected, designed, and engineered system. If managing a trillion rectangles in one interactive session sounds like fun to you, please give us a ring.

- **Litho simulation: optical engineering, applied math background, and finite-difference time-domain (FDTD) algorithm, modal and frequency-domain analysis of antennas and waveguides**
- A bachelor's degree in a relevant computer-related discipline with an excellent record of academic achievement;
- In-depth knowledge of C++ and object-oriented design and programming;
- Experience with numerical analysis;
- Background in vector calculus, differential equations, linear algebra and complex variables; and
- Good written & verbal communication and documentation skills.
- 1-2 years of industry experience using standard software engineering practices in a professional environment;
- Familiarity with MATLAB or other technical computing languages;
- Experience using debugging, profiling and revision control tools;
- Experience developing applications in the Visual Studio environment; and
- Experience with the QT application framework

- **analog/RF, digital, and mixed-signal circuit simulation: SPICE (traditional SPICE, Analog FastSPICE, digital fastSPICE simulators, and RF fastSPICE simulators, multi-mode SPICE simulation, mixed-mode SPICE simulation, SPICE-HDL co-simulation)**
- Good knowledge on spice simulation technology, including MNA (Modified Nodal Analysis), Euler algorithm [Euler method – I implemented this as a sophomore in my applied math class, Vector Analysis and Complex Analysis], dc, tran analyze method, etc.

- **Analog and Mixed-Signal Circuit Simulation**
- Graphical user interface (GUI) design and implementation using WTL and Qt: [WTL refers to the Windows Template Library]
- Component software development utilizing COM
- Software testing and QA:
  1. Unit, system, and regression testing
  2. DOS batch files
  3. Linux command shell scripting
- Numerical and scientific programming:
  1. Nonlinear circuit analysis (SPICE, Harmonic Balance)
  2. Electromagnetics (Method-of-Moments, Finite Element)
  3. Communication systems/signal processing
  4. General numerical analysis/operations research
- Physical design tool development - familiarity with OpenAccess, solid understanding of back-end flow for Si IC design
- Release engineering - manage automatic build and test systems, author and maintain installation scripts, manage source code control and defect tracking databases
- Successful candidates will posses strong technical and coding skills. Demonstrated ability with C++ is required for most positions, along with a comprehensive understanding of STL
- Knowledge of component software development and object oriented design, as well as familiarity with design patterns and agile software development techniques is strongly preferred

---

- **EDA Databases**
- Experience with EDA Databases, such as OpenAccess

- **TCAD**
- technology CAD (TCAD) software engineer
- development and application of software for the design and characterization of devices for integrated circuits
- meshing algorithms
- emphasis on numerical methods

- **2D/3D Semiconductor Device Simulator**
- Ph.D. in Physics/Engineering with 3 years relevant development experience of device simulators.
- Extensive knowledge of semiconductor physics and nonlinear solution techniques.
- Comprehensive knowledge of C/C++ required (familiarity with additional programming languages an advantage).
- Commercial software development experience.
- Experience with Silvaco TCAD software a plus

- **2D/3D Diffusion Simulation and Process Simulation**
- PhD in Physics, Electrical Engineering or Material Science with 3 years experience of scientific or TCAD software development.
- Deep knowledge of modern impurity and defect diffusion models, numerical methods and nonlinear solution techniques.

- Practical knowledge of modern diffusion related processes used in semiconductor industry.
- Comprehensive knowledge of C/C++ is required
- Experience with TCAD simulation software (ATHENA, SUPREM4) is a plus.

- **Numerical Solvers**
- Develop and maintain a state of the art linear solver library
- Candidate must be experienced with direct and iterative methods for large sparse linear systems
- Ph.D. with 5 years direct development experience of sparse linear solvers
- Strong in C/C++

- **device measurement, extraction and SPICE modeling software**
- BS degree in computing or engineering
- Minimum of 5 years experience developing C++ software on UNIX, Linux or Windows
- Experience in instrument control, GPIB and device measurements
- Hands-on experience in SQL using Oracle/Firebird would be beneficial

---

- **logic simulator**
- Good communication skills and ability to work with remote team
- Solid understanding of cycle accurate and functional simulation techniques. Basic knowledge of levelized code and time wheel based event driven simulation.
- Working knowledge of Verilog/VHDL/SystemC
- Strong coding skills in C/C++, Java, Python, or Ruby
- Strong OS experience in Linux/Unix preferred
- Minimum BSEE/CS with 3-5+ years experience

- **Basic set of EDA skills, knowledge, and experience**
- C++ programming and GUI software development experience.
- Unix/Linux system programming experience (particularly inter-process communication) is required.
- Experience with Qt development will be a big plus.
- Experience with shell script programming (Perl, Tcl, Bourne, Korn, or C) is highly desired
- IC design experience with HDLs (VHDL, Verilog, SystemVerilog, VHDL-AMS, Verilog-AMS, SystemC, & SystemC-AMS)
- Experience with SPICE circuit simulation

- **Basic set of EDA skills, knowledge, and experience**
- We are looking for software engineers who have experience in the development of EDA tools for IC implementation
- They need to have in depth knowledge of at least one of the following areas:
  1. RTL synthesis
  2. logic synthesis
  3. physical synthesis
  4. area/timing/power optimization
  5. static timing analysis
  6. floorplanning
  7. placement

8. routing

- And have at least a Ph.D in EE or an MS with 4+ years of relevant working experience on commercial EDA tools
- Programming skills: C/C++.

- **3D volumetric and surface mesh generators**
- Ph.D. in Maths/Computer Science plus three years of relevant experience
- Experience in developing unstructured volumetric and surface mesh, solid modeling techniques and surface definitions, adaptive mesh refinement and de-refinement
- Good knowledge of data structures and algorithm design and C++ object-oriented design techniques
- Ability to assimilate code quickly and leverage new code off existing libraries
- Familiar with domain decomposition, adaptive smoothing and octree techniques is a plus

---

- **EDA software development using GPGPU computing and graphics programming**
- 1+ years of experience in 3D Graphics with OpenGL
- C++, STL, Boost, **Generic Programming** a plus
- R3Logic is seeking a Software Engineer to develop next generation Electronic Design Automation (EDA) tools
- This position focuses on designing and implementing a rendering engine employing OpenGL.

- **Thermal Analysis, Signal Integrity Analysis, EM Simulation, & Computational Electromagnetics**
- C/C++ development of physics-based simulation software
- Computational geometry and differential geometry
- EDA analog design tools for layout, simulation, etc
- EDA frameworks, esp. the Si2 OpenAccess model and API
- FD or FEM discretization, and related numerical methods, to solve PDEs (esp. Poisson's equation)
- GUI development (e.g., 3-D graphics) and 3-D visualization software with OpenGL, Qt; data visualization; ray tracing
- Numerical methods
- Parallel computing, including concurrent programming, MPI, and CUDA
- Self-directed learning about existing code, and reconstruction of algorithms
- Simulation algorithms
- Working knowledge of Tcl, Perl, doxygen, gdb, gmake

Other skills that are related to EDA jobs:

- **compiler design and construction**
- At least 1 year experience in writing optimizing compilers for higher order and/or lazy functional languages
- At least 1 year experience in Java and C++ including their execution models
- At least 1 year experience in term rewriting, lambda calculus
- At least 2 years experience in static analysis

- At least 1 year experience in program transformation

- **compiler design, construction, and optimization**
- At least 2 years experience in microprocessor architecture and micro-architecture
- At least 2 years experience in optimizing compiler
- At least 2 years experience in polyhedral optimization

- **compiler design, construction, and optimization**
- Looking for self-motivated, team oriented software engineer to lead and sustain development effort for compiler environment to support an advanced programmable hardware engine
- Responsible for design and implementation of compilers for low level and high level languages targeting a proprietary reconfigurable medium grained parallel processor
- Must be enthusiastic self-starter interested in solving unusual problems and inventing syntactic constructs and back-end implementations in support of novel hardware features
- You will work as an integral part of a small team to develop and deliver new compiler capabilities that span both the front-end and back-end of the compiler.
- Participate in the entire development lifecycle of the compiler including specification, architecture, design, implementation, unit test and verification.
- 10+ years experience with compiler front ends and back ends, including parsing, code generation, relocation fix-ups, etc.
- Strong understanding of compiler theory and practice, including tools such as Lex and Yacc.
- Experience implementing in multiple high level and assembly languages, and willingness/ability to rapidly learn and use new implementation languages.
- Ability to read and understand hardware architecture documentation and produce supporting compiler technology.
- Must thoroughly understand computer architecture; parallel architecture background a plus.
- Willingness to implement and maintain unit and regression tests.
- Experience with paradigmatic languages such as Lisp, Ruby, Python, Smalltalk, Haskell, etc. a plus.
- Experience with parallel processors, reconfigurable processing, dataflow processing, and/or digital hardware design a large plus.
- Reproduce and resolve issues reported by internal and external application developers.
- Create benchmarks and other tools used to measure and increase compiler performance and efficiency.
- Help create and maintain automated regression tests for quick validation of compiler functionality.
- Work with other team members to ensure that the compiler is ready for shipment.
- Minimally requires BSEE/CS combined with 5-10+ yrs related

- **concurrent programming, parallel computing (computer architecture), and compiler design & construction (dynamic compilation and runtime profiling techniques)**
- pursue research in programming models and compilation techniques for parallel programs
- develop runtimes for parallel program optimization
- architecture design to support efficient execution of parallel codes

- We are seeking candidates with experience in both compiler optimization techniques, runtime systems and good architecture understanding to participate in research on managed runtimes and dynamic compilation. The ideal candidate will be able to design and implement hardware structures to aid dynamic compilation, runtime profiling techniques to monitor application behavior and trigger optimization, and design and implement dynamic compiler techniques that take advantage of runtime information to optimize the execution of parallel programs.

- **parallel computing, concurrent programming, performance analysis / optimization, and GPU architecture**
- evaluating and optimizing the performance of common concurrency patterns on graphics processing units (GPUs)
- identifying the concurrency characteristics of general-purpose computing on GPUs (GPGPU)
- develop techniques to identify the concurrency characteristics of general-purpose tasks off-loaded to GPUs
- design and adapt synchronization mechanisms that leverage the architectural characteristics of GPUs
- have experience in programming GPUs, concurrent programming, experimental evaluation, and familiarity with processor and memory hierarchy architectures and their impact on application and system performance
- At least 2 years experience in Concurrent Programming
- At least 2 years experience in Computer Architecture
- At least 2 years experience in GPU Architecture

- **performance analysis, & thermal and power management**
- At least 4 years experience in computer science and performance analysis with a focus on system level performance and cache hierarchies in CMP systems
- At least 1 year experience in power and thermal issues in CMP systems

---

- **Digital VLSI design**
- At least 2 years experience in circuit design, architecture and microarchitecture
- At least 2 years experience in computer aided design

- **Digital (and Analog) VLSI design**
- At least 1 year experience in Circuit Design
- At least 1 year experience in Simulation analysis
- At least 1 year experience in CAD tool experience
- develop high-performance, energy-efficient VLSI circuits for application to advanced CMOS technologies
- our projects include arrays, sensors, memory organization, power regulation & distribution, and advanced 3D technology research

---

- **Analog/RF VLSI design**
- At least 2 years experience in High Speed Circuit Design, especially I/Os and/or PLLs
- At least 1 year experience in Custom Digital Design
- At least 1 year experience in Circuit Design Tools/Languages such as Cadence, VHDLnet or comparable),

- Basic knowledge in high-frequency laboratory measurement experience
- Basic knowledge in behavioral modeling, simulation/analysis: Matlab

---

- **embedded software development**
- At least 2 years experience in cell programming (develop, implement, and optimize algorithms on the cell processor)
- Parallel programming skills preferred

- **SDR (Software Defined Radio) Application Engineer**
- Among other requirements: OFDMA or CDMA background required; and experience with SDR algorithms (Rake, FEC, Viterbi, Turbo Codec, Symbol processing)

5.15 Innovations in EDA Research

Innovations in EDA research:

1. engage in breakthrough, disruptive, or orthogonal innovation, rather than incremental innovation
2. be an expert in 1 (or 2) field(s), and be able to communicate across disciplines
3. be a connector who can synthesize ideas from different disciplines
4. do not make too rigid technological assumptions that limit generalizations. It is however important to understand the details ... **The devil is in the details unfortunately**
5. do not make too rigid assumptions to get a broad perspective on the problems that may limit innovation.
6. do not spend undue time in securing industrial uptake of academic research... That is, do not worry about whether if the industry is going to adopt your innovations, algorithms, tools, and methodologies. Pursue good research, and the industry will adapt good technological innovations.
7. be innovative without challenging sound scientific practice or limiting industrial impact
8. determine what the value proposition of your product would be early on – I need to know this early to earn angel or VC funding
9. create a new market segment with my R&D effort/investment, rather than join a market segment that is packed/filled with big market players
10. pursue organic growth and development
11. attempt to come up with radical innovations that will generate a positive cash flow soon after implementation
12. be aware of my environment and the business ecosystem that I am in, such as being aware of market consolidation and production consolidation, and things that are getting commoditized
13. **develop a sound plan for "technology-to-product transition"**:
    (a) In business and engineering, new product development (NPD) is the term used to describe the complete process of bringing a new product or service to market. There are two parallel paths involved in the NPD process: one involves the idea generation, product design, and detail engineering; the other involves market research and marketing analysis.
    (b) **continually improve research-to-product transition; start from doing good research to develop radically innovative technologies, and turn such technologies into products**
    (c) **having a good research-to-product development flow is an important factor in the success of a start-up**

   (d) "EDA 3.0: So you are an EDA startup?" By Tom Kozas and Michael Sanie (03/17/2009 11:37 AM EDT), EDA DesignLine; see http://www.edadesignline.com/howto/215900624;jsessionid=Q3ZN3TWP2APXKQSNDLRSKH0CJUNN2JVN?pgno=1 and http://www.edadesignline.com/215900624;jsessionid=Q3ZN3TWP2APXKQSNDLRSKH0CJUNN2JVN?printableArticle=true
   (e) "EDA 3.0: EDA startup guidelines" By Nicolas Mokhoff (03/17/2009 9:26 AM EDT), *EE Times*; see http://www.eetimes.com/showArticle.jhtml?articleID=215900871

14. HPs Rules of the Garage (Hewlett Packard):
   (a) Believe you can change the world.
   (b) Work quickly, keep the tools unlocked, work whenever.
   (c) Know when to work alone and when to work together.
   (d) Share – tools, ideas. Trust your colleagues.
   (e) No politics. No bureaucracy. (These are ridiculous in a garage.)
   (f) The customer defines a job well done.
   (g) Radical ideas are not bad ideas.
   (h) Invent different ways of working.
   (i) Make a contribution every day. If it doesn't contribute, it doesn't leave the garage.
   (j) Believe that together we can do anything.
   (k) Invent.

15. David Packard's Eleven Simple Rules:
   (a) Think first of the other fellow.
   (b) Build up the other persons sense of importance.
   (c) Respect the other persons personality rights.
   (d) Give sincere appreciation.
   (e) Eliminate the negative.
   (f) Avoid openly trying to reform people.
   (g) Try to understand the other person.
   (h) Check first impressions.
   (i) Take care with the little details.
   (j) Develop genuine interest in people.
   (k) Keep it up.

16. the revised HP way by HP's former CEO Carly Fiorina:
   (a) Trust Your people
   (b) Match the right people to the right job
   (c) Hire the best
   (d) Continually learn
   (e) Motivate by vision
   (f) Manage through processes

17. The HP Way, http://www.hpalumni.org/hp_way.htm:
   (a) We have trust and respect for individuals.
      i. We approach each situation with the belief that people want to do a good job and will do so, given the proper tools and support.
      ii. We attract highly capable, diverse, innovative people and recognize their efforts and contributions to the company.
      iii. HP people contribute enthusiastically and share in the success that they make possible.
   (b) We focus on a high level of achievement and contribution.

  i. Our customers expect HP products and services to be of the highest quality and to provide lasting value.
  ii. To achieve this, all HP people, especially managers, must be leaders who generate enthusiasm and respond with extra effort to meet customer needs.
  iii. Techniques and management practices which are effective today may be outdated in the future.
  iv. For us to remain at the forefront in all our activities, people should always be looking for new and better ways to do their work.

(c) We conduct our business with uncompromising integrity.
  i. Our customers expect HP products and services to be of the highest quality and to provide lasting value.
  ii. To achieve this, all HP people, especially managers, must be leaders who generate enthusiasm and respond with extra effort to meet customer needs.
  iii. Techniques and management practices which are effective today may be outdated in the future.
  iv. For us to remain at the forefront in all our activities, people should always be looking for new and better ways to do their work.

(d) We achieve our common objectives through teamwork.
  i. We recognize that it is only through effective cooperation within and among organizations that we can achieve our goals.
  ii. Our commitment is to work as a worldwide team to fulfill the expectations of our customers, shareholders and others who depend upon us.
  iii. The benefits and obligations of doing business are shared among all HP people.

(e) We encourage flexibility and innovation.
  i. We create an inclusive work environment which supports the diversity of our people and stimulates innovation.
  ii. We strive for overall objectives which are clearly stated and agreed upon, and allow people flexibility in working toward goals in ways that they help determine are best for the organization.
  iii. HP people should personally accept responsibility and be encouraged to upgrade their skills and capabilities through ongoing training and development.
  iv. This is especially important in a technical business where the rate of progress is rapid and where people are expected to adapt to change.

18. Simula Culture:
  (a) Reference: Simula Research Laboratory, "Simula Culture," Simula Research Laboratory, Fornebu, Norway. Available online at: http://simula.no/about/culture and http://ploneint.simula.no/pdf/simula_culture/simula_culture.pdf; last accessed on July 28, 2012.
  (b) Reference: Simula Research Laboratory, "The Simula Code of Ethics," Fornebu, Norway. Available online at: http://simula.no/about/the-simula-code-of-ethics; last accessed on July 28, 2012.

19. evolutionary incremental innovation:
  (a) low uncertainty
  (b) linear and continuous trajectory
  (c) short-term
  (d) exploitation of existing technology
  (e) idea generation (project charter) ⇒ concept feasibility (business plan) ⇒ capability development (launch proposal) ⇒ ramp up (sanity check) ⇒ launch

   (f) weaknesses:
      i. low gatekeeper knowledge (lack of expertise knowledge about product domain/technology) leads to poor judgments
      ii. slow and serial
      iii. concept frozen too early
      iv. focused on milestones and deliverables, not the customer(s)
      v. long review preparation time
      vi. narrow criteria [for success]
      vii. maturity focus versus learning focus ... [employees don't grow personally and professionally; they lose touch with contemporary market trends, and their skills and knowledge become out of date]

20. revolutionary radical innovation:
   (a) high uncertainty in terms of technical challenges, market and organizational risks, and resource utilization/requirement
   (b) sporadic and discontinuous trajectory
   (c) business model and plan evolves through discovery-based learning
   (d) long-term
   (e) sporadic - starts and stops, & dead ends and revivals
   (f) nonlinear - detours, & recycling back through activities in response to discontinuities and setbacks
   (g) stochastic - waxing and waning of interest and funding, key players come and go, & priorities change
   (h) context dependent - corporate culture, history, personalities, informal relations, and experience all create a mix of accelerating and retarding factors
   (i) formality of process can increase gradually over time as uncertainty is reduced with gradual progress (accomplished milestones) made in the projects
   (j) freeze the concept later in the project than with incremental innovation:
      i. longer concept development time
      ii. shorter implementation time
      iii. later part of concept development time overlaps with early phase implementation time; this can help avoid missing newly emerging technologies
   (k) is effective for changing business environment
   (l) need for strong creative chaos and strong guiding structure
   (m) fuzzy front end
   (n) research $\Rightarrow$ technology $\Rightarrow$ design $\Rightarrow$ production $\Rightarrow$ market — the first two phases involve radical innovation, and the last two involve incremental innovation (Kaizen)
   (o) design is at the interface of radical innovation and incremental innovation
   (p) exploration of new technology
   (q) fuzzy front-end elements:
      i. Opportunity Identification
      ii. Opportunity Analysis
      iii. Idea Genesis
      iv. Idea Selection
      v. Concept & Technology Development

21. structured NPPD (New Product & Process Development) strategy
22. Avant-garde means "advance guard" or "vanguard". The adjective form is used in English, to refer to people or works that are experimental or innovative, particularly with respect to art, culture, and politics.

23. the market was the source of new ideas for directing R&D, which had a reactive role in the process. The stages of the "market pull" model are: Market need $\Rightarrow$ Development $\Rightarrow$ Manufacturing $\Rightarrow$ Sales
24. types of research:
    (a) exploratory research: structures and identifies new problems
    (b) constructive research: develops solutions to a problem
    (c) empirical research: tests the feasibility of a solution using empirical evidence
25. diffusion of innovation is a theory of how, why, and at what rate new ideas and technology spread through cultures:
    (a) theory on the "diffusion of innovation"
    (b) the diffusion of innovation theory
    (c) technology adoption lifecycle ... [there is a nice graph in Wikipedia to denote the proportion of innovators, early adopters, early majority, late majority, and laggards in a population]
    (d) technology acceptance model: perceived usefulness and perceived ease-of-use
26. From a layman's perspective, the technological maturity can be broken down into five distinct stages:
    (a) Bleeding edge: any technology that shows high potential but hasn't demonstrated its value or settled down into any kind of consensus ... Early adopters may win big, or may be stuck with a white elephant.
    (b) Leading edge: a technology that has proven itself in the marketplace but is still new enough that it may be difficult to find knowledgeable personnel to implement or support it.
    (c) State of the art: when everyone agrees that a particular technology is the right solution.
    (d) Dated: still useful, still sometimes implemented, but a replacement leading edge technology is readily available.
    (e) Obsolete: has been superseded by state-of-the-art technology, maintained but no longer implemented.
27. take note of uptrends and downtrends in industry
28. emerging technologies is a general term used to denote significant technological developments that in effect, broach new territory in some significant way in their field. Examples of currently emerging technologies include nanotechnology, biotechnology, cognitive science, robotics, and artificial intelligence
29. Emerging technologies are those which represent new and significant developments within a field; converging techologies represent previously distinct fields which are in some way moving towards stronger inter-connection and similar goals.
30. A disruptive technology or disruptive innovation is marketing speak for a technological innovation that improves a product or service in ways that the market does not expect, typically by being lower priced or designed for a different set of consumers.
31. What Does Value Proposition Mean? A business or marketing statement that summarizes why a consumer should buy a product or use a service. This statement should convince a potential consumer that one particular product or service will add more value or better solve a problem than other similar offerings.
32. Unique Selling Proposition (also Unique Selling Point)
33. proprietary technology, sustainable advantage, and follow-on products

34. "I take notice of companies that are going to change the fundamental rules, NOT incrementally improve things. 20x runtime and 100x capacity is a rule breaker. Oasys is as interesting as it is compelling." – Joe Costello, former CEO, Cadence Design Systems; see http://www.oasys-ds.com/ ... last viewed May 27, 2009
35. develop and grow my portfolio of innovative formal verification technology patents
36. Develop EDA tools to empower engineers, not to replace them

5.16 Career Advice for Research in EDA

Career advice for research in EDA:

- Come up with ideas for a new product/service, develop design plans, prove the concept with a prototype/simulation, pitch it to your managers and senior management (corporate executives/C-Suite); that is, "get an opportunity to have an idea floated by someone, and try to get funding. That's quite an experience, very broadening" – Rico Mariani ... knowing your facts, knowing what's important, knowing what's good for the business, what makes a good business case ... Think of Bill Mullen's (VP @ Synopsys in 2007) experience in getting funding for his power and timing analysis/optimization project; he had developed the prototype, and had three/four members in the team then
- Prof. Ian Parberry's philosophy for running the Game Development Research Group (GDRG) at University of North Texas (UNT) is given as follows; see http://www.eng.unt.edu/ian/research/philosophy.html; available May 8, 2010. GDRG values research that is:
  1. Deep, not superficial. GDRG papers will not be hastily-written, shallow, nor destined to obscurity.
  2. Done in small groups, not large anonymous ones. We will be sharing knowledge and understanding with graduate and undergraduate students, not working in large groups where it is hard to assign credit, nor working in isolation.
  3. Tied into the "web of research," not done in a vacuum. We will acknowledge prior related work in full, aiming for a paper that is significant enough to be cited in turn.
  4. Innovative, not repetitive. We will follow new threads, not embroider past achievements.
  5. Well-grounded, not half-baked. We will implement systems, not just describe them. Experimental data generated by these systems will be large enough to achieve statistical significance.
  6. Knowledge, not just data.
- move people's focus/attention from minor problems, which you can solve, to major problems that you are seeking help with or opinions for
- ask penetrating questions about the product/technology
- don't be a "one hit wonder"; the term "One Hit Wonder" describes an artist or band who had great popularity with a single song, and who then never reached such popularity again
- beware of regularly developing one-trick ponies that may cause inconvenience about people; the slang term "one trick pony" is used to refer to something that can only be used for one very specific application
- make a plan for what you do
- be deliberate about what you do
- be autonomous, and drive technology
- **ask, "How bad can it be?"** ... When you make a mistake, always ask this question. At the worst, move on!

- have fun with what you do, experiment with your product to see how well it works and determine its failure modes, keep up with current and emerging technologies (even if it is outside of your field), and be creative (try new things to make your product/innovation better)
- nurture your ideas, and realize them ... ideas are worthless
- suppose it [this job] doesn't work out, you go get another job! – Jim Gray to Catharine van Ingen
- when/if you are not having fun, what's the point? – Catharine van Ingen
- Edsger Dijkstra's tips for PhD students:
  - Never compete with colleagues
  - Try the most difficult thing you can do
  - Choose what is scientifically healthy and relevant. Don't compromise on scientific integrity.
- choose work that is challenging and interesting, because it will keep your attention and help you learn ... also, it saves some time playing with good ideas
- read several sources for a concept (thoroughly), so that I can see different approaches to the material and then synthesize what's most relevant
- if I cannot obtain a permanent full-time position at an organization I would like to work for, I can take a temporary job or paid internship there, or volunteer there as an unpaid intern
- when I take up a job, think about acquiring the skills that I will need for the next position that I am aiming for
- take relevant classes for research, such as research methodology, statistics, technical writing, and management
- get the [inside] scoop, and learn about a prospective department's tenure practices by talking to faculty and students
- become aware of trends that may affect hiring, such as recession, the emergence of new subfields, and rates of faculty retirement
- get teaching experience in graduate school, or as a postdoc, before I apply
- **comparisons of tools on a set of benchmarks cannot yield infinite speedup; all speedups should be indicated as a finite number, say 10x or 50x**
- do my salary homework (e.g., see IEEE-USA salary survey), but be reasonable
- use my job to my advantage... If my first academic job is less than desirable, do the best job that I can, author or co-author a solid piece, send out applications, and get out of there so I can get on the tenure track
- do more of the heavy lifting, or legwork, in research projects
- never let the best (is it really the best?) get in the way of better: while building the best technology, but taking some steps pragmatically forward is a much better thing for everybody... We learn by doing
- 4 Ps in R&D: papers, projects, programs, and people
- "pick a good project, and pick a good problem (that's the most important thing... to invent hard stuff) in many areas and advancing the state of the art in those things" – Jim Gray, interview for Microsoft's "Behind the Code" series on the ResearchChannel
- "the goal is to find something that is simple enough to make progress on it" – Jim Gray, interview for Microsoft's "Behind the Code" series on the ResearchChannel
- "develop a strong mix of breath and depth" – Jim Gray, interview for Microsoft's "Behind the Code" series on the ResearchChannel
- "be able to go back to first principles when solving problems" – Jim Gray, interview for Microsoft's "Behind the Code" series on the ResearchChannel

- "say what you mean, and mean what you say" – Jim Gray, interview for Microsoft's "Behind the Code" series on the ResearchChannel
- "know the reasons for the things you do" – Jim Gray, interview for Microsoft's "Behind the Code" series on the ResearchChannel
- avoid teams that are micromanaged by supervisors, managers, and/or advisors that act like control freaks; **beware of bureaucratic people, "heavy-handed legal department," micromanagement, narcism, and technophobia** ... Similarly, avoid technophobes.
- avoid engaging in one-upmanship, and reduce/eliminate rankism; do not dish out putdowns ... Do not be bent on these, since they affect the unity and productivity of the team
- develop a specialized skill that is not taught at most universities
- become a shockingly good programmer
- figure out how to help chip designers achieve wickedly high performance chips – such as general-purpose processor, graphics processors, or DSP designs at 45 nm, 32 nm, and below
- keep an "open door policy," so that people feel welcomed to come in and talk to you
- keep building my network of research collaborators (from undergraduate through PhD and beyond)
- keep track of my research interests, including its history since my undergraduate days at the University of Adelaide
- **regularly check if my research topic is too broad/narrow**
- research area/field/topic:
  1. area:
     (a) extent, range, or scope
     (b) field of study, or a branch of a field of study
  2. field:
     (a) a sphere of activity, interest, etc., esp. within a particular business or profession
     (b) an area of human activity or interest
     (c) a topic, subject, or area of academic interest or specialization
     (d) a branch of knowledge
  3. topic:
     (a) a subject of conversation or discussion
     (b) the subject or theme of a discourse or of one of its parts
     (c) thesis, subject matter
     (d) the subject of a speech, essay, thesis, or discourse
     (e) a subject of discussion or conversation
     (f) a subdivision of a theme, thesis, or outline
     (g) the subject of any distinct portion of a discourse, or argument, or literary composition
     (h) the general or main subject of the whole
     (i) a subject, as of conversation or of thought
- bring computational thinking to other fields
- **learn how industrial-scale EDA research is carried out, and do that kind of research**
- Thomas Friedman's commencement speech at Grinnell College on May 18, 2009:
  1. Do the synthesis and connect the dots from [my] different interests ... the whole buffet in liberal arts: mastered 2 or more (quite) different fields, use the framework in one to think afresh about the other, and mash them together into something that no one else has thought of... do not spend [my] whole life in one silo
  2. "If it is not happening, it's because I'm not doing it"
- from a guest blog by Prof. Andrew B. Kahng from UCSD:
  1. `http://www.edadesignline.com/guest_blogs/217200218`

2. "Does the EDA industry have a roadmap?", in EDA DesignLine (04/27/2009 2:02 PM EDT)
3. Roadmaps look ahead. For example, the 2001 ITRS told us that power and productivity imperatives would drive the industry to **SOC design methodology convergence** and a multicore-centric future, and that **embedded software** and the **design-manufacturing interface** presented looming crises. **The implications of such predictions cannot be addressed by crash programs instituted only after wakeup calls; they demand proactive and sustained attention.**
4. reduced risk of another type of delivery error, namely, solutions that are plain wrong or mismatched to designer needs
5. I can say that **there are really only around 6000 EDA R&D heads worldwide at any time.** This number is at best stable (cf. industry revenues, research funding levels, etc.).
6. Some of these R&D resources are inevitably spent on fruitless directions, and are also subject to overlapping efforts (competitors build equivalent technologies, and geography (North America, Asia-Pacific, EU) creates triple-redundancy in funding of research programs)
7. As the 6000 number includes professors and graduate students, this is clearly not a large resource; it must be as well-targeted as possible.
8. The wisdom at Intel and many other semiconductor companies is that **around 6-7 years are needed to bring a new EDA technology "from DAC paper to production flow"**. Again, correct directions must be identified as early as possible.
9. [the] industry [needs to focus on its] priorities, [and establish] agreements on non-differentiating ("commodity") infrastructure and standards, so that these crises can be tackled:
   (a) software and concurrency
   (b) analog/RF
   (c) die-package-board
   (d) die stacking
10. with a fixed number of EDA R&D engineers, we need to stop working on some issues
11. In the big picture, who wins a power format or current-source model or process variation model war is less critical to semiconductor industry health than consensus, interoperability, and moving on to fundamental design technology challenges.
12. EDA roadmaps:
   (a) International Technology Roadmap for Semiconductors (ITRS):
      i. the main EDA roadmap is found in the **Design chapter of the ITRS**, which provides quantified technology needs and potential solutions across system-level, logic- and physical-level, verification, DFT and DFM design technology arenas.
      ii. The closely aligned **ITRS System Drivers chapter** roadmaps key product drivers and aligns with the iNEMI (International Electronics Manufacturing Initiative) roadmap
   (b) MEDEA+ EDA roadmap; see http://www.medeaplus.org/index.php
   (c) Japan's STRJ Working Group 1's roadmap for design technology
13. [build up and align my research portfolio with EDA roadmaps]
14. Questions to ask concerning roadmaps:
   (a) EDA engineers [and grad students]: "Do you know your company's [or research lab's] roadmap?"
   (b) EDA managers [and professors]: "How much of your product roadmap and R&D budget is directed to solving long-term – i.e., more than 7 years – design technology needs?"
   (c) EDA customers: "Do you proactively define (at what time horizon), or do you just accept and live with, the roadmaps of your EDA suppliers?"
15. Arguably, design technologists today do not have a roadmap – at least, not in the way that manufacturing technologists do.

16. Many companies are in the throes of reinventing themselves; nearly all are focused on a 1-quarter horizon and not a 20-quarter horizon. Perhaps those MRMs (management review meetings) with key customers and partners (Intel, TSMC, etc.) are a sufficient roadmapping process for EDA.
17. http://www.edadesignline.com/guest_blogs/ or http://www.edadesignline.com/guest_blogs/217500437
18. "Risk-reward structures in EDA," in EDA DesignLine, May 15, 2009
19. **The EDA Innovator's Dilemma** – In the "innovator's dilemma" (Christensen, 1997), a market leader competes by adding incremental features to its market-leading product until eventually displaced by a "disruptive technology" in one of two ways. :
    (a) In low-end disruption, the market leader's stream of added features eventually overshoots the market need, opening the door to simpler and lower-cost solutions. The history of Tangent, Avant!, Magma, Silicon Perspective, (Sierra, Atoptech, etc.) in the SP&R space is a good example of cyclical low-end disruption in EDA... It recalls an old March 2003 quote, "SOC designers ultimately can't use big Swiss Army knives stuffed with too many back-end EDA technology options," and the saying that "those who don't learn from history are doomed to repeat it".
    (b) In new-market disruption, a disruptive technology opens up markets that did not previously exist. From the analyst and investor standpoint, a key challenge for the EDA industry is that it doesn't experience nearly enough new-market disruption.
20. **The EDA Innovator's Solution**:
    (a) The "innovator's solution" (Christensen and Raynor, 2003) is a set of precepts and strategies with respect to market segmentation, leadership, growth, etc., according to which successful companies can identify, develop and unleash the next disruptive technology.
    (b) Two key elements of the "innovator's solution" are:
        i. outsourcing of commoditized, "more than good enough" solutions
        ii. strategic investment in disruption-based growth
    (c) Unfortunately, the EDA industry's outsourcing and investment doesn't match the "innovator's solution" model: companies still compete on commodities (back-end data models, power formats, etc.) while outsourcing long-range R&D
    (d) **Why does this fail as an "innovator's solution"?**:
        i. **First, outsourcing of R&D is neither sufficient nor stable.**
        ii. The "research funding gap" mentioned in my last posting is a world-wide insufficiency that is, essentially, a "tragedy of the commons". This has occurred despite the efforts of many champions across companies, consortia, and funding agencies who have tried to maintain research investment levels and well-considered project portfolios.
        iii. As for stability, the past few years have seen the pools of startup-based "outsourced EDA R&D", as well as EDA research at universities and consortia, dry up as a consequence of macroeconomics and new funding targets (cleantech, biotech, homeland security, etc.).
        iv. **Second, long-term success is difficult when one has outsourced the strategic R&D that gives rise to future disruptive technologies.**
21. Consider the risk-reward structure in an established EDA (or, IC design) company. Technology assets are primarily software (C++ or Verilog), the original developers are long gone, and there is a lot of "legacy" in the IP portfolio. This leads to a highly risk-averse profile, i.e., a tendency toward "don't touch" and "leave well enough alone".
22. Legacy code – despite the benefits of seasoning and reuse – eventually becomes an albatross. Given the incremental product evolution that is inherent in the "innovator's dilemma" (recall the path to low-end disruption!), it is no wonder that market leaders can end up far off target with their "next-generation" products.

23. I would claim that low-end disruption in EDA does not resemble laser mask writers being displaced by e-beam mask writers, or vertically integrated steel mills being replaced by mini-mills. All too often, the periodic event in the EDA world is not the emergence of a new and disruptive technology, but simply a mundane, inertial loss of agility and ability to respond to the market.
24. Next, consider the risk-reward structure for academic researchers. The first point to understand is that "risk" equals "effort" (how many researcher-months of salary and benefits and opportunity costs are sunk into a project), and "reward" equals "likelihood that a respectable and impactful publication is the result".
25. The classic assumption is that research targets exist along a continuum. At one end, there is low-risk-low-reward research: incremental, turn-the-crank, with some "minimum publishable unit" outcome. At the other end, there is high-risk-high-reward research: e.g., trying to prove that P != NP. Depending on personality, talents, career stage, and other factors, academics choose their targets along this continuum. But in EDA research, again we've got it backward.
26. For EDA researchers today, speculative, far-out research often has low risk but high reward. An example from my past: very few comparisons or proofs of superiority were needed to publish that first paper on physical design of quantum-dot cellular automata based chips.
27. On the other hand, highly incremental research can require an enormous amount of effort and industry-grade experimental infrastructure to support any claimed contribution.
28. Again drawing on examples from my past: making the next speed-quality improvement in KL-FM bipartitioning or mixed analytic and partitioning-based global placement are very high-risk research propositions. This anomaly is magnified when reviewer criteria set "not novel" as grounds for rejection.
29. The result: academic researchers constantly pose "new" problem formulations (escaping the duty of apples-to-apples comparison with previous works), blurring the leading edge of knowledge and ultimately slowing progress on real problems.
30. Question: How can EDA researchers regain a risk-reward tradeoff that correctly associates high reward and high impact with high risk?
31. More than 10 years ago, the MARCO Gigascale Silicon Research Center established the Bookshelf of Fundamental CAD Algorithms (see "Toward CAD-IP Reuse: A Web Bookshelf of Fundamental Algorithms", A. E. Caldwell, A. B. Kahng and I. L. Markov, IEEE Design and Test, May-June 2002, and VLSI CAD Bookshelf 2). See http://vlsicad.eecs.umich.edu/BK/
32. The idea of the Bookshelf was to provide a repository for reusable "CAD-IP" – source codes, benchmark data, evaluation scripts, etc. – to enable R&D engineers in both academia and industry to advance the leading edge without having to reinvent the wheel. The food for thought in this posting: Might it be time to revisit some of the ideas that motivated the creation of the Bookshelf?
33. An EDA company that better manages the ball and chain of legacy code will have greater agility and more resources that can be directed toward new markets and new technologies. This implies that companies must make greater investments in foundations and infrastructures for product and technology innovation. In the context of IC design and EDA, this includes software engineering methodology, software infrastructure, and fundamental algorithms (e.g., repurposable optimization engines, machine learning engines, etc.).
34. In the EDA research context, the MARCO GSRC Bookshelf promoted open-source publication, along with standardized benchmarking and comparison methodologies. By reducing the overheads of research at the leading edge, such culture changes can effectively reduce the

"research funding gap" and rebalance risk-reward structures toward research that has impact on core technology challenges.

35. The past decade has also seen an important addition to the original Bookshelf concept: simplification. When researchers release elegant, compact codes that embody the leading edge in astonishingly few lines of code – global routers from Michigan, or synthesis packages from Berkeley – the impact across both academia and industry can be profound. If conference and journal reviewers and funding agencies follow suit, rewarding or incenting simplifications and productivity (speedup) improvements at least on par with 5 percent and "new problem" advances, then the world will have changed for the better.

- from a blog by Peggy Aycinena:
  1. http://www.edadesignline.com/guest_blogs/;jsessionid=AC2OUH2T42JMQQSNDLOSKHSCJUNN2JV
  2. "Open source EDA software defeats Lock-in — Dream on," in EDA DesignLine Engineering Guest Blog, June 10, 2009
  3. "You have to distinguish between open source software and open source EDA software. There's a big difference. For open source software, you just need to create an application, but for open source EDA software you also need to have a scientific background, need to understand physical design, and have to have some mathematical algorithms to code. In the case of a synthesis tool, for instance, you have to be able to minimize the gate count while also improving the timing closure. Cadence or Synopsys spend a lot of money on research developing these types of features, a [level of resource] completely lacking in the open source EDA community."
  4. "The EDA vendor community understands today that interoperability is very important, but this is frequently a weakness in the open source community. Open source EDA developers often try to invent their own file formats. Very few of them use industry standard formats to avoid having to start from scratch. Through FEL [Fedora Electronic Lab], we are trying to change that, however, to convince developers to work so they can interchange their data easily. If we can share data between the different applications developed by the open source EDA community and the vendor tools, it will be a way to encourage tool users and designers to use and contribute to the open source tools."
- Takako Hashimoto, "Seven Practices of Working Mother Researchers," IEEE Women in Engineering meeting/seminar. Available online at: http://womenandmath.files.wordpress.com/2010/09/brochure100811takako.pdf; last accessed on November 22, 2010. [Also, see: Caroline Series, "Seven Practices of Working Mother Researchers," in the *Women in Mathematics* blog, September 30, 2010. Available online at: http://womenandmath.wordpress.com/2010/09/30/seven-practices-of-working-mother-researchers/; last accessed on November 22, 2010.]:
  1. Prof. Takako Hashimoto; IEEE JC WIE Vice Chair; Associate Professor, Chiba University of Commerce
  2. "Consider Woman Life as an Optimization Problem: Take not a local maximum point but a real maximum having a long-term view"
  3. "Community formation provides high added value"
  4. Seven Practices of Working Mother Researchers:
     (a) Never give up your research work.
     (b) Conduct good communication.
     (c) Find and leverage your time.
     (d) Show a visible result.
     (e) Defend your home.
     (f) Education and home discipline are of great value.

(g) Be active.

- "Analog adoption takes a lot of time. Any adoption of an analog tool probably takes 5-7 years, and this is beyond the time horizon of most VCs. To get money from a VC, you'd have to have an adoption time of 3-5 years. You'll never get there. In analog EDA, the slow road is the way to go." – Henry Chang, in an interview with Richard Goering, in Expert's Corner: "Analog complexity demands new verification approaches," By Richard Goering, 12/19/07; see http://www.scdsource.com/experts.php?id=64&page=1
- Is this improvement incremental or substantial?
- Am I doing fundamental research on [insert my research topic]?
- Have I recently attended any "eye-opening talks" related to my research?
- "Users are exceptionally good at what they do, completely overloaded in their daily jobs, reluctant to transition away from a known good solution, and realistic about EDA marketing claims. Users dont do science projects. They very well understand the realities of bringing up a new tool, they have hard budgets and P&Ls to deal with, they always have other fish to fry, and they have very conservative profiles when it comes to benefit-cost propositions." – DFM luminary, Prof. Andrew Kahng, http://www.scdsource.com/experts.php?id=178

5.17 Advice on Choosing an Advisor

Advice on choosing an advisor:

1. Check if the professor/researcher has the following characteristics/attributes:
   (a) **Listen patiently**. Give the student time to get to issues they find sensitive or embarrassing.
   (b) **Build a relationship**. Simple joint activities – walks across campus, informal conversations over coffee, attending a lecture together – will help to develop rapport. Take cues from the student as to how close they wish this relationship to be.
   (c) **Don't abuse your authority**. Don't ask students to do personal work, such as mowing lawns, baby-sitting, and typing.
   (d) **Nurture self-sufficiency**. Your goal is not to "clone" yourself, but to encourage confidence and independent thinking.
   (e) **Establish "protected time" together**. Try to minimize interruptions by telephone calls or visitors.
   (f) **Share yourself**. Invite students to see what you do, both on and off the job. Tell of your own successes and failures. Let the student see your human side and encourage the student to reciprocate.
   (g) **Provide introductions**. Help the student develop a professional network and build a community of mentors.
   (h) **Be constructive**. Critical feedback is essential to spur improvement, but do it kindly and temper criticism with praise when deserved.
   (i) **Don't be overbearing**. Avoid dictating choices or controlling a student's behavior.
   (j) **Find your own mentors**. New advisers, like new students, benefit from guidance by those with more experience.
   (k) Reference: Empowering Leadership Alliance, "Advice for New Mentors," Empowering Leadership Alliance, Rice University, Houston, TX, 2010. Available online at: http://empoweringleaders rice.edu/Content.aspx?id=194; last accessed on June 11, 2012. Also, see American Chemical Society's *Project Seed: Student, Mentor and Coordinator Handbook*

2. A. Lee, C. Dennis, and P. Campbell. Nature's guide for mentors. Nature, 447(7146):791–797, June 14, 2007 [10].

5.18 Advice on Picking a Ph.D. Dissertation Topic

Advice on picking a Ph.D. dissertation topic:

1. Choose a topic you love:
   (a) This may be the most important criteria
   (b) You're going to be spending so much time with this project, and your quality of life will be much better if these hours are spent enjoyably
   (c) What's more, the quality of your research, writing, and arguments will be much better if you feel genuine passion for your work
   (d) Choose a topic you find both fascinating, and socially [intellectually] significant [Or, one that may lead to the commercialization of your research as a spin-off or start-up]
   (e) Never let someone pressure you into writing about a certain topic!
2. Pick something your advisor finds interesting and is knowledgeable about:
   (a) Of course, if this is not possible, you might want to change your advisor instead of changing your topic
3. Pick a topic that will be helpful in your career path:
   (a) If your goal is an academic career, pick a topic that you can easily modify into journal articles or a book, and that will lend itself well to future research
   (b) If you want to work at a teaching oriented institution, consider a topic you can use in the classroom
   (c) If you are going into industry, choose a topic that will make you more marketable.
4. Find a topic that establishes your niche in your field:
   (a) Do your research and find a topic that fits into existing bodies of literature, but that builds upon theory and expands it [It is easier to do this, rather than to create a new field (of your own)]
5. Choose research that is unique:
   (a) Do significant research to make sure this topic has not been done before
   (b) Be creative and choose an idea that stands out from the pack as original and innovative
6. Think carefully before you choose a controversial topic:
   (a) Academics are a sensitive lot, and in every field there are certain topics and positions that will send highly educated people into intellectual temper tantrums
   (b) This doesn't mean you should avoid topics that push people's buttons
   (c) However, if you choose a controversial topic, think carefully about whether it might restrict your employment, tenure, or publishing opportunities
7. Pick a topic that you already have some expertise about:
   (a) This will help preserve your sanity and get you out the door faster
   (b) This isn't the time to explore a brand new area
   (c) Along the way, take coursework and write class papers that will help you write your dissertation or thesis
8. Pick a manageable topic:
   (a) This is a huge project, but it isn't your life's research
   (b) A good advisor will help you narrow down your topic so that you don't remain in graduate school for many long years

9. **"Whatever you do, hopefully in the end you will have found an area, a problem within that area, and a professor who works on that problem that are all a good (enough) match for you"**; see http://mathgradblog.williams.edu/?p=10#more-10
10. **When I am considering a dissertation topic, and am investigating a particular technique/technology, ask the following questions:**
    (a) **What is the value proposition of [this technique/technology that I'm considering]?**
    (b) **Can this technology be leveraged? If so, how?**
    (c) **How much can it improve design and verification productivity?**
    (d) **Have publications about this technique/technology gained a lot of traction in academia and in industry?**
    (e) **While these question should not be used to judge whether it is worth pursuing research in a particular topic, it can help me determine if there are opportunities to develop radical innovations since a considerable amount of research has not been done in this topic. That is, if this research topic is an emerging research topic, whatever "little" that I do can be considered significant.**
    (f) **I should pursue a research topic because I am passionate about it, and want to know more about it. As such, I want to carry out investigations and run experiments/simulations to determine aspects of it that no one else has, or prove that existing theories, models, or solutions are inadequate and develop better theories, models, or solutions.**
    (g) **However, I need not let this research pursuit remain purely an idealistic academic endeavor. Doing good research in my chosen research topic can lead to the creation of start-ups, opportunities of postdoc fellowships with world renowned research scientists and engineers, research positions in industry or nonprofit research institutes, or tenure-track positions in academia!!!**
    (h) **For example, with respect to the Ph.D. position at the Uni Trento, ask the following:**
        i. **What is the value proposition of SMT-based formal verification?**
        ii. **Can this technology be leveraged? If so, how?**
        iii. **How much can it improve design and verification productivity?**
        iv. **Have research publications on SMT-based formal verification gained a lot of traction in academia and in industry?**
11. Factors for determining whether a research project is good (from http://cacm.acm.org/blogs/blog-cacm/91102-choosing-successful-research/fulltext):
    (a) **Are you solving an important problem?** Why can't it be solved using today's technology? How will solving it make the world better?
    (b) **Is the market big enough?** In other words, who will benefit from you solving this problem, and by how much? Who will care?
    (c) **Is it simple?** The more complex an idea, the harder it is to communicate. If you can't explain it to an executive in one slide, they're unlikely champion your idea. If the product team doesn't understand your implementation, they will never take ownership of it.
    (d) **Is it a breakthrough?** What idea or technology do you have that your competitors don't? What prevents the product group from solving this problem on their own using known technology? Does your approach compare favorably to the alternatives, both in product and in the academic literature?
    (e) Does this topic fascinate me?
    (f) If you don't have a burning question to answers, don't pursue a PhD.

12. Finding a Research Topic: http://compscigail.blogspot.com/2010/04/cra-w-grad-cohort-finding-
html

5.19 Advice for Initial Research on Chosen Topic

Advice for initial research on chosen topic:

1. Planning phase: Spend about three months on the literature survey. I should continue to regularly read research publications on this topic for at least the first year, so that I can be ready to take my qualifying examination. For each research publication that I read, enter its BibTeX database, summary, and paraphrase key points into my BibDesk database.
2. Last viewed on June 6, 2010. See http://www.dcsc.tudelft.nl/Education/Graduation%20Guide0910.pdf.
3. Questions that I can ask concerning my approach to solve a research problem (or answer a research question):
   - What is GENI?
   - Why is GENI doing this work?
   - Tell us some background about Buckminster Fuller.
   - How do power grids work?
   - How technically feasible is the global grid? What percentage of it can be done now?
   - What are the environmental implications; good and bad?
   - What about the EMF issue? and the impact on GENI's work?
   - How will this reduce acid rain, the greenhouse effect, and toxic wastes?
   - How much will the project cost?
   - Where will the money come from?
   - How do the costs and benefits weigh out?
   - Tell me more about the Computer Simulation Model.
   - Who will administer this global system?
   - How will this affect trade between countries?
   - How can we expect countries to trust one another enough to become so interdependent?
   - How will this effect developed and developing countries?
   - How will it effect the economies of every nation?
   - What is the criteria to determine the effect for each country?
   - How will it make the leap and take advantage of remote renewable generation away from fossil fuels?
   - Can the United States and the Soviet military and industrial complexes use this project as a means to convert from defense to civilian related interests?
   - Who are the winners and the losers, if any?
   - How do we set this up as a win/win solution?
   - How will this reduce hunger?
   - How will this reduce overpopulation?
   - How do you propose this shift will occur? What's GENI's role in that?
   - How is GENI funded?
   - References:
     (a) GENI, "Medit Kit", in *What's GENI?: Media Kits*, Global Energy Network Institute (GENI), San Diego, CA, July 28, 2009. Available online at: http://www.geni.org/globalenergy/projects/news/media_kit/index.shtml; last accessed on June 16, 2012.

   (b) GENI, "Frequently Asked Questions", in *What's GENI?: Frequently Asked Questions*, Global Energy Network Institute (GENI), San Diego, CA, July 17, 2009. Available online at: http://www.geni.org/globalenergy/faq/index.shtml; last accessed on June 16, 2012.

5.20 Advice for Comps/Prelims and Quals

Advice for Ph.D. comprehensive examination (or comprehensive exam/comps), Ph.D. preliminary examination (prelims), and Ph.D. qualifying exam (quals):

1. it may consist of three written exams and an oral examination, only written examinations, or only oral examinations — Check with the department
2. **Are comps the same as the general examination (or generals)?**
3. comps/prelims are used to establish that the student is qualified to embark on Ph.D.-level research in the chosen field
4. quals are used to make sure that the student has a good plan for finishing the Ph.D., and that the committee (especially the advisor) and the student agree on the scope of the future research
5. study strategically and organize citations
6. keep up with class readings for seminar classes, which are classes that typically involve reading lots of papers and writing literature reviews (survey papers)
7. review my study notes for my classes, and formulate my ideas
8. read strategically:
   (a) while some departments disseminate a suggested reading list to graduate students, many require students to create their own
   (b) students who write their own reading list, which typically includes journal articles and review papers, should make sure to organize it by topic and have a professor review it
   (c) students can more readily synthesize and recall information if they read journal articles in a logical order, organized chronologically and by topic
   (d) use a good reference management software, such as `BibDesk`, to facilitate the organization of research publications that I have read
9. keep abreast of my field:
   (a) professors often get ideas for questions from reading the latest research and controversies
   (b) when studying for the exams, graduate students should take time to peruse the latest journals in their field
10. take detailed notes:
   (a) after reading an article, write down the main points of the article
   (b) summarizing the research publication forces me to think critically as I read and the summaries provide material to study later
   (c) **Aim to read >100 papers, while writing my qualifying exam research proposal**
   (d) make sure that I try to find the importance of every paper that I read
11. get organized:
   (a) spend time before quals using a good reference management software (e.g., `BibDesk`) to get my research publications in order; **I should continue to keep my research publications in order, as long as I am doing research**

(b) use `BibDesk`, or other good reference management software (also, citation management software or citation software), to keep track of where I store digital copies of research publications on my computer ... By using keywords and/or the name of the author(s), I should be able to search for a research publication, and access its digital copy (if any) ... While I can tag each BibTeX entry in a BibTeX file, where I store all my references, I would still have to use Finder™, and perhaps Spotlight™, to find the digital copy (if any) of the research publication ... With a good reference management software, such as `BibDesk`, I can use the keywords to access the BibTeX entry and the digital copy (if any) of the research publication

(c) use `BibDesk`, or other good reference management software, to keep summaries and notes of research publications that I have read ... Again, I can enter such summaries and notes of research publications as comments after each BibTeX entry in a BibTeX file; however, I would have to take extra steps to locate the BibTeX entry and the digital copy (if any) of the research publication, and I may have trouble locating the BibTeX entry and the file ... With a good reference management software, such as `BibDesk`, I can use the keywords to access the BibTeX entry, and my notes and summary for the research publication, as well as the digital copy (if any) of the research publication

(d) **I should only use ONE reference management software for any significant period of time; else, I would lose some of the benefits of managing the references and their storage (digital copies of references)** — I would otherwise need to keep porting my repository of research publications between reference management software

(e) create an index of research publications, which is organized by topic and author

(f) know where all my research publications are (don't scatter them around); use BibDesk to search for and keep track of research publications, and my corresponding notes and summaries for them

(g) make an itemized report of what information (my summaries and notes of publications) I have

12. seek advice from older students:
    (a) students who have already passed their comprehensive exams make great resources
    (b) don't shy away from asking them (advanced Ph.D. students/candidates) for advice, or even for copies of their graded research proposals
    (c) some colleges keep copies of past qualifying exams; **if I need to take written and/or oral exam(s), try and see if I can get a copy of such past qualifying exams from the department's administrators for revision**

13. ask questions:
    (a) when you receive your exam questions, and if you are not sure what any question is asking, seek the faculty for clarification; else, I may end up taking a wrong guess about what the question is asking for, and give the wrong answer
    (b) do not feel that I am admitting ignorance if I ask a question; the question can be expressed better
    (c) it is better to risk looking uninformed than to give the wrong answer to a question (especially an answer that is off-topic)

14. read the directions:
    (a) under pressure, even normally meticulous students might fail to notice formatting details
    (b) test-takers should note the number of pages (or marks, as a percentage) allotted to a question, and the formatting requirements of the comprehensive exam
    (c) double check that I have answered every part of each (essay) question

15. be thorough, not encyclopedic:
    (a) **packing in too many citations is an all too common mistake**
    (b) I can spend a lot of [**unnecessary**] time in the library or on the Internet, finding every article remotely relevant to the question ... **Don't do that and waste my time; remember what Alicia Lowery told me about "researching everything under the sun before I finish my thesis"**
    (c) I need to be aware of the major trends, but I don't have to write an anthology
    (d) while the intensive preparation can be exhausting, it pays
    (e) I may not get any of those curveball questions; don't worry about such questions
    (f) at the end of the [oral exam of the] quals, elicit feedback from the professors [and research scientists] on my thesis committee about my professional development; this can be validating in terms of the hard work that I have put in, and my research direction

5.21 Resources for Prelims

Resources for Prelims:

1. Past examination questions (and relevant resources for prelims):
    (a) Missouri University of Science and Technology (formerly University of Missouri-Rolla):
        i. Electrical and Computer Engineering (MS&T calls its ECE prelims, "Quals"): http://ece.mst.edu/usefullinks/qualifyingexams.html
    (b) Stanford University:
        i. CS Comprehensive Exams: http://www-cs-students.stanford.edu/phd/comps/ and http://www-cs-students.stanford.edu/phd/comps.php
        ii. EE Qualifying Exams: http://ee.stanford.edu/students/eequals.php; http://ee.stanford.edu/phd/quals/courses.php?node=hardware&id=a1; and http://ee.stanford.edu/phd/qualsguide/. Stanford has no prelims for EE.
        iii. Robert Chen, "A Guide to the Electrical Engineering Qualification Exam at Stanford University,"
        iv. Steven Soneff, *Past EE Quals*. Available at: http://www.stanford.edu/~ssoneff/quals/; last accessed on September 10, 2010.
    (c) UC Berkeley:
        i. CS Prelim Exam Information: http://www.eecs.berkeley.edu/GradAffairs/CS/Prelims/
        ii. EE Prelims: http://www.eecs.berkeley.edu/GradAffairs/EE/Prelims/ and http://www.eecs.berkeley.edu/GradAffairs/EE/Prelims/Syllabi/cad-rev-10-2004.doc
    (d) University of Michigan:
        i. Department of Electrical Engineering and Computer Science, College of Engineering:
            A. [19]

5.22 Resources for Quals

Resources for Quals:

1. Past examination questions (and relevant resources for Quals):
    (a) Stanford University:
        i. CS Qualifying Exams: http://www-cs-students.stanford.edu/phd/quals.php, and https://cs.stanford.edu/degrees/phd/QualifyingExams/QualifyingExams or https://cs.stanford.edu/degrees/phd/Main/QualifyingExams
        ii. Stanford's EE department has no information on the depth examination (Quals).
    (b) University of Wisconsin-Madison:

i. Department of Computer Sciences (Computer Sciences Department):
   A. *Ph.D. Qualifying Exams Archive*:
      - http://www.cs.wisc.edu/phdquals/archive.html
      - UW-Madison has "two" rounds of "qualifying exams". The first round examines technical depth in a research area, while the second round examines the student's expertise in her/his chosen research topic. See http://pages.cs.wisc.edu/~pubs/grad-guidebook/.
      - UW Operating Systems Qualifying Exam: http://www.cs.wisc.edu/areas/os/Qual/
      - UW-Madison Programming Languages Qualifying Exam: http://www.cs.wisc.edu/areas/pl/quals.php
      - Performance Modeling and Analysis – MA Depth Qualifying Exam: http://www.cs.wisc.edu/areas/ma/ma-reading.html
      - Numerical Analysis Screening Exams: http://www.cs.wisc.edu/areas/na/screening.html
      - mathematical optimization/programming: http://www.cs.wisc.edu/includes/quals/mp.html and http://www.cs.wisc.edu/math-prog/optimization-reading-list pdf
      - Artificial Intelligence PhD Qualifying Exam (AI Depth Exam): http://www.cs.wisc.edu/areas/ai/qual.html and http://www.cs.wisc.edu/includes/quals/ai.html
      - UW-Madison Computer Architecture Qualifying Exam (or Qualifier Exam): http://pages.cs.wisc.edu/~arch/uwarch/?q=node/82 or http://www.cs.wisc.edu/arch/uwarch/?q=node/82
      - :
      - :

2. Faculty advice:
   (a) Carnegie Mellon University, Department of Electrical and Computer Engineering:
      i. Tom Martin, "The Ph.D. Qualifying Exam in ECE at Carnegie Mellon," Department of Electrical and Computer Engineering, Carnegie Mellon University, October 16, 1997. Available at: http://www.cs.cmu.edu/~tlm/qual.html; last accessed on August 27, 2010. See http://www.cs.cmu.edu/~tlm/qualsched.html for Tom's Quals study schedule. A copy of this is available at: http://www.ece.cmu.edu/~koopman/student_info/tom_martin_qual.html.
      ii. **Philip Koopman, "Thoughts on Ph.D. Qualifiers," Department of Electrical and Co**
      iii. Official Quals web page: http://www.ece.cmu.edu/graduate/phd/qualifying.html

5.23 Notes about the Depth Exam Style

Notes about the depth exam style:
This would have more emphasis on the broad context of the assigned papers than solely on the specific details of the individual papers, although demonstrated understanding of the assigned papers would still be critical. It would be expected to include much more about papers relevant to the area but not necessarily directly related to the assigned papers. There would tend to be somewhat less emphasis on recapitulating the specific techniques of the assigned papers but more on their context and on the general techniques used within the area as a whole. It would be expected to include discussion of avenues and methods that might be fruitful for future research in the context of the whole range of research in the area.

5.24 Writing a Ph.D. Thesis

Notes on writing a Ph.D. Thesis:

1. *The contribution to knowledge of a Master's thesis can be in the nature of an incremental improvement in an area of knowledge, or the application of known techniques in a new area. The Ph.D. must be a substantial and innovative contribution to knowledge.*
2. Take at least one complete term for writing the thesis, so that I have sufficient time to organize my arguments and results
3. As I formalize my results into a well-organized thesis document, check if my thesis is capable of withstanding the scrutiny of expert examiners. Have I discovered any weaknesses in my thesis? How do I fix these weaknesses? Have I fixed these weaknesses?
4. Spell difficult new concepts out clearly
5. Choose section titles and wordings to give the examiners clear directions to information they are seeking for in my thesis. Make it easy for my examiners to easily determine my problem statement, my defence of the problem, my answer to the problem, and my conclusions and contributions. Doing these can reduce the need for major revisions, and the number of minor revisions I have to make for my Ph.D. thesis.
6. Remember that a thesis is not a story: it usually doesn't follow the chronology of things that you tried. It's a formal document designed to answer only a few major questions such as: `What is this students research question? Is it a good question? (Has it been answered before? Is it a useful question to work on?) Did the student convince me that the question was adequately answered? Has the student made an adequate contribution to knowledge?`
7. Avoid using phrases like "Clearly, this is the case..." or "Obviously, it follows that ..."; these imply that, if the readers don't understand, then they must be stupid. They might not have understood because you explained it poorly.
8. Avoid red flags, claims (like "software is the most important part of a computer system") that are really only your personal opinion and not substantiated by the literature or the solution you have presented. Examiners like to pick on sentences like that and ask questions like, "Can you demonstrate that software is the most important part of a computer system?"
9. The purpose of your thesis is to clearly document an original contribution to knowledge. You may develop computer programs, prototypes, or other tools as a means of proving your points, but remember, the thesis is not about the tool, it is about the contribution to knowledge. Tools such as computer programs are fine and useful products, but you can't get an advanced degree just for the tool. You must use the tool to demonstrate that you have made an original contribution to knowledge; e.g., through its use, or ideas it embodies.
10. the dissertation can be viewed as a teaching exercise, in which I learn how to conduct, design, and analyze independent research.
11. commit 10 to 20 hours per week for 12 to 18 months to avoid becoming a casualty to the "All But Dissertation (ABD)" label
12. set specific work hours and choose a specific place to work
13. communicates my expectations and needs on topics such as deadlines, meeting frequency, authorship credit, and work hours ... For example, if I crave more praise or regular feedback than my advisor dishes out, I can confront the problem directly, such as by saying, "I am getting discouraged: Tell me three things I am doing right here"
14. beware of these: procrastination, a tendency toward perfectionism, conflicts with advisers, lack of communication with dissertation committees, and failure to finish the dissertation

15. do not push aside unstructured tasks, such as the dissertation, in favor of more immediate tasks ... manage time well for these unstructured tasks
16. advice on how to reach the finish line:
    (a) set priorities and goals: set daily, monthly, and yearly goals
    (b) pick a timeline and stick to it
    (c) know what you're getting into
    (d) choose a manageable dissertation topic
    (e) establish a network within the program
    (f) be assertive and set boundaries when needed
    (g) relax and have the right attitude
17. find a "study-buddy" to work with ... that way, I would be committed to working on my dissertation for myself and my "study-buddy"
18. Introduction of a thesis: "the first part of my thesis lays the groundwork necessary to carry out similar investigations in directions not considered by Cremona" ... "In part two of my thesis, I use the well-developed theory of modular forms, combined with the algorithms of part one, to investigate two outstanding conjectures."
19. Samples of how I can display my Ph.D. thesis:
    (a) Gurmeet Singh Manku, "Dipsea: A Modular Distributed Hash Table," Ph.D. Thesis, Department of Computer Science, Stanford University, August, 2004. Available online at: http://www-cs-students.stanford.edu/~manku/phd/index.html; last accessed on December 22, 2010. [This web page includes a table of contents (for the Ph.D. thesis), a list of abstracts for each chapter in the Ph.D. thesis, a list of publications that this thesis was based, and its BibTeX entry.]
    (b) Andrei Alexandrescu, "Scalable Graph-Based Learning Applied to Human Language Technology," Ph.D. Dissertation, Department of Computer Science & Engineering, University of Washington, 2009. Available online at: http://erdani.com/research/; last accessed on December 22, 2010.
    (c) Pau Arumí, "Real-time Multimedia Computing on Off-the-Shelf Operating Systems: From Timeliness Dataflow Models to Pattern Languages," Departament de Tecnologies de la Informació i les Comunicacions, Universitat Pompeu Fabra, June 30, 2009. Available online at: http://parumi.org/thesis/; last accessed on March 7, 2011.

---

20. resources:
    (a) University of California, Berkeley:
        i. Department of Electrical Engineering and Computer Sciences, College of Engineering:
            A. Bernhard E. Boser, "Tips for Writing an Engineering Thesis," Department of Electrical Engineering and Computer Sciences, College of Engineering, University of California, Berkeley. Available online at: http://www.eecs.berkeley.edu/~boser/ThesisTips.html; last accessed on December 28, 2010.
                - Reader: Filing a thesis is a degree requirement, but really it is written for a reader. Be considerate to this reader, **giving him all the information he needs to appreciate and build on your work**. At the same time be concise as the reader probably is in a hurry - just like you.
                - Scope: Focus on your contributions, not the field at large. Many theses contain a very large tutorial component, while the actual work is only touched on. There is no set length for a good thesis, but 5 pages are usually not sufficient to clearly describe your work, and 50 or more pages of introductory material will deter most readers.

- Completeness: **Give all information needed to recreate your work**. Often only the general design procedure and perhaps a textbook equation are included, but actual data is missing. **Your contribution is not only finding the equation, but also applying it, so document the entire process.**
- Conclusion: **Draw conclusions from your results, explain why something is important (or not), and rank considerations in order of importance. This applies to every topic, chapter, and the complete thesis. Be careful, though, not to extend your conclusions too far and state limitations. Your solution may be the best in a particular situation but not necessarily always.**
- Language: Incorrect language makes the most interesting thesis unreadable. Run the spell checker and proofread everything.
- Format: Neatness counts again since this improves readability. But do not waste your time with unnecessary frills of modern word-processors.

5.25 Advice for Ph.D. Oral Defense

Advice for the Ph.D. oral exam, oral defense (German: verteidigung/kolloquium), viva voce (/viva), or dissertation defense (/thesis defense) – how to present a successful dissertation defense:

1. Note that a shorter version of the oral defense can be included in a "job talk" (usually an hour-long lecture) that I would give during an interview for an academic position, except that the "job talk" would involve a presentation of my work to date as well as my future research plans ... The job talk shall not focus on the minutia of my past research, while leaving out the larger context of my work. Show attendees of the talk, professors and students (grad students and undergrads) that I have a vision, and I don't have to just stick to one study, or even my own work... Allow my personality to come through
2. don't worry about hardball questions
3. learn what's expected, anticipate the hard-hitting questions, open myself to feedback and, most importantly, remember to relax
4. enjoy myself in the Ph.D. oral defense ... Because it's not every day that I have a roomful of scholars completely interested in what I have to say – it's something special I should enjoy
5. learn the rules and norms for a defense delivery for the department (or, specifically the Ph.D. program) that I am in ... determine my department's expectations by talking with my dissertation chair or fellow students:
   (a) Am I expected to bring refreshments, or is that practice discouraged?
   (b) Are I allowed to invite friends and family members, or is the defense open only to other graduate students or faculty?
   (c) Should my presentation be 10 minutes or 30?
   (d) Should I hand a final copy of my dissertation to my committee a month in advance, or is two weeks the norm?
   (e) Usually, refreshments are not a requirement and defenses are open, but don't assume that's the norm for my department
   (f) students are usually expected to book the room and date for their defense, which can take time ... Give myself a month to do that. It can be challenging to find a time when five busy faculty can meet
6. communicate with my committee members:
   (a) talk regularly with my thesis committee about how my research is going

   (b) my thesis committee can range from four to five members, depending on the program, who I hand-picked when I began the dissertation process
   (c) consult them if I need to alter my methodology or circulate drafts, and to gather advance input if possible
   (d) be open to the perspectives of the committee members ... but don't be shy about sharing my expertise or defending a point of view
   (e) get as much feedback as time permits in both written commentary and in-person meetings with committee members ... It's a chance for them to ask the tough questions ahead of time
   (f) document my progress by having committee members sign off on any major revisions they request at the proposal stage ... Taking that step can prevent confusion among faculty at the defense meeting about why dramatic changes were made
   (g) don't go off on my own; if I do, I may fail my Ph.D. oral defense
   (h) do what I agreed to do in my proposal, and communicate with my committee about changes
   (i) contact with my committee can provide some valuable insight into the types of questions they might ask during the defense – as can doing a little advance detective work
   (j) know my committee members' likes, dislikes, and pet peeves
   (k) ask people who have been through a defense with them, read their articles, and surf the Web for more information on their research expertise and specialty areas

7. practice and prepare:
   (a) be prepared to present a clear explanation of why I did the study, a brief overview of my methodology and results, and a discussion of the implications of my research, but don't recite the manuscript
   (b) the assumption is that my committee has already read my dissertation in detail ... I don't want to bore them by going through it again; I just want to refresh them
   (c) at the same time, don't assume that my committee members have memorized my manuscript
   (d) if they ask a question that I think I addressed, don't assume they remember that I addressed it; repeat myself patiently
   (e) for the question-and-answer portion that follows the presentation, students should be primed to answer questions about their methodology, to defend and explain their choice of analysis, and discuss how their study contributes to the literature, informs theory and where the research might go next
   (f) staging a mock defense with fellow graduate students is a great way to practice answering the types of questions I may be asked
   (g) join a student-research group that regularly organizes practice defenses
   (h) in some cases, the dry run will be more challenging than the defense; sometimes, students ask harder questions than faculty
   (i) many students say attending another student's defense helps them prepare and know what to expect
   (j) pick a well-prepared peer, since attending a defense that doesn't go well can be anxiety-provoking instead of helpful
   (k) practice my talk in the room where I'll eventually defend
   (l) know where I will move, look, sit, and take notes
   (m) the less I have to react to in the moment, the more focused I can be on the task at hand, which is to demonstrate I have strong knowledge of my project

8. develop the right attitude:
   (a) approaching the defense as a critically constructive experience is key

   (b) avoid coming off as too protective about my work during the meeting, but to also not be overly compliant about committee members' feedback
   (c) students should be open to the perspectives of the committee members, who are committed to helping improve their dissertation, but they shouldn't be shy about sharing their expertise or defending a point of view if they feel their committee may be misinformed
   (d) what's more, students shouldn't feel discouraged if their committee asks for minor revisions to their manuscript after the defense
   (e) it's not at all uncommon for committee members to suggest a different analysis, some changes in a table, or to rework the discussion section to clarify a certain point ... students sometimes have a couple days work ahead of them to put it in final shape

9. breathe, then answer:
   (a) stumped by a question? ... Don't be afraid to take a moment to consider it, paraphrase it back for clarification or ask that it be restated
   (b) similarly, if I don't know the answer, it's better to say so and give the best answer I can, rather than digressing for a few minutes
   (c) keep in mind that for the most part faculty are just asking questions to see if I can think critically – they are not trying to be difficult or stump me
   (d) staying calm can be one of your greatest assets during the defense
   (e) it's normal to be anxious and scared about my defense, but many people before me have passed, and I can too
10. stage a mock dissertation defense with fellow students to practice answering questions.

5.26 Advice for Applying for Postdoc Positions

Advice for securing postdoc positions, and achieving postdoc success:

1. start my applications early ($\geq$1 year before expected date of Ph.D. oral defense)
2. Set goals:
   (a) Before I begin my search, decide what I want to get out of the experience
   (b) Do I want to learn new skills?
   (c) Acquire in-depth training in an area of expertise?
   (d) Get great mentoring?
   (e) Look ahead to what I think I want to be doing, and work backwards from there
3. Do your homework:
   (a) Thoroughly investigate potential positions
   (b) Will the position fit with the goals I've set?
   (c) Will I get the mentoring and training I need?
   (d) Are there health benefits?
   (e) Have others been happy there?
   (f) Talk with former postdocs and other psychologists to find out
4. Don't give up:
   (a) even if it seems too late in the year, keep applying and networking
   (b) new funding or postdocs who backed out of a position could open up opportunities
5. Think ahead:
   (a) Once I'm in a position, keep my next steps in mind

   (b) Where is my first "regular" job going to come from?
   (c) If I don't get a job right away, what will I do?:
      i. engage in open research independently, and with others
      ii. publish lots of papers ... publish, publish, publish!!!
      iii. engage in open source software development
      iv. engage in open source IC design
      v. advice students on their honors (senior capstone) and Masters projects
      vi. Where do I want to go after my postdoc?
      vii. What happens if that excellent but elusive job isn't available?

6. Find a career mentor:
   (a) Link myself with a mentor who can help me through the job search
   (b) a mentor can be an early-career/seasoned professor or research scientist
   (c) Someone who can honestly discuss what job opportunities look like, what the appropriate things to ask for, and say, and do are
7. Spell out my expectations:
   (a) Discuss my duties and goals with my supervisors before I take the position
   (b) Some even advise writing up a contract that includes what the supervision will entail, what skills or training I'm going to get from the fellowship, and how much autonomy will I get in my research projects ... This is helpful for organizations without an established training program for postdocs; get our goals and expectations articulated ... Beware of make-it-up as I go positions
8. Check my progress:
   (a) once I start my supervision, regularly discuss my progress toward my goals with my supervisor
   (b) write down my fellowship goals at the beginning of the year, and constantly referred back to them to stay on track during my postdoc program(s) ... Determine if I have the flexibility to alter them
   (c) ask the right questions, and be proactive about it
   (d) seek and earn more degrees of freedom to try to get some additional training
9. regard pursuing a postdoc research fellowship as taking that step toward what I really want to be doing as an independent investigator; it may take more than one postdoc to make that transition
10. Mentor fit:
   (a) think about the qualities of my Ph.D. advisor that are good for me and that help me work well, and look for those qualities in a postdoctoral mentor
   (b) research gets done better and more quickly when I really click with the person I'm working with
   (c) Has the researcher provided good mentoring to other fellows?
   (d) Have fellows published papers with the mentor?
   (e) What was the authorship order of those papers?
   (f) Talk with current and former postdocs, interns, and staff to get a feel for how things work
   (g) think about it as I am hiring this mentor to work for me ... do the vetting of the prospective mentor before I start talking to him/her about writing a grant together
   (h) consider whether the mentor will allow me the level of independence I need ... if I am transitioning to a new research area, I'm not ready for complete independence; however, when I'm ready to work autonomously, I need to work in a lab that gives me the freedom that I need ... In this case, a different lab may be a better fit
   (i) Get a postdoctoral advisor who is willing to bat for me to overcome bureaucracy.

11. Funding:
    (a) How will my research be funded?
    (b) If I'll be on a grant, how long will the grant last? Is that enough time to complete my work?
    (c) I may need to begin applying for the next year's funding shortly after I begin my fellowship.
    (d) Two common funding approaches are to get written into an investigator's research grant or, to show I am competitive with others, to write my own research grant [such as those for NSF/SRC in the US]
12. Goals and expectations:
    (a) Talk over my fellowship objectives with my mentor and seek agreement, perhaps writing a contract that seals it.
    (b) Once in my fellowship, continually evaluate my progress based on those goals
    (c) However, that doesn't mean my objectives can't shift
    (d) fellowships often represent the last time I have such freedom to explore my various research interests as a researcher
    (e) In terms of my intellectual development, nothing beats it
    (f) I don't have faculty meetings, don't have to teach. I can just focus on my research.

5.27 Resources for Postdoc Positions

Resources for postdoc positions:

1. The Chronicle of Higher Education:
    (a) Zoe Smith and Ariana Sutton-Grier, "Making the Most of Your Postdoc," The Chronicle of Higher Education: Advice: Do Your Job Better, July 15, 2010. Available at: http://chronicle.com/article/Making-the-Most-of-Your/66265/; last accessed on September 6, 2010.
2. Inside Higher Ed:
    (a) http://www.insidehighered.com/
    (b) Has information concerning the application for faculty positions, and advice for career paths in higher education institutions.
    (c) Kerry Ann Rockquemore, "Winning Tenure Without Losing Your Soul: Stop Talking, Start Walking," Inside Higher Ed: Career Advice, Inside Higher Ed, January 25, 2010. Available at: http://www.insidehighered.com/advice/winning/winning2; last accessed on September 7, 2010.
3. Nature Publishing Group (a division of Macmillan Publishers Limited):
    (a) Kendall Powell, "Careers and Recruitment: A foot in the door," in *Nature*, Vol. 463, No. 7281, pp. 696-697, February 4, 2010. Available online at: http://dx.doi.org/10.1038/nj7281-696a; last accessed on December 31, 2010.
        i. I should commence looking for a postdoctoral research position early in my penultimate year.
        ii. When looking for a postdoctoral research position, I shall take note of the name of research labs which journal/conference papers that I find are interesting. That is, each time I read an interesting journal/conference paper, take note of the name of the research labs and its affiliation (name of department, and organization/university).
        iii. Compile a list of 20-30 research labs that I find are interesting. Pare down this list to about 5-6 labs, which I will apply to. Use the same heuristics for applying to graduate school. These 5-6 research labs shall include:

   A. A reach lab, which I want to be part of but may have difficulty securing a postdoctoral research position in
   B. 3-4 match/target labs, which postdocs and Ph.D. students are comparable to me in terms of research excellence, and which I shall have a good chance of securing a postdoctoral research position in
   C. 1-2 reliable/safety labs, which I should be able to secure a postdoctoral research position in

iv. Network with students at those labs in conferences, workshops, symposiums, and other events (such as "summer schools"). Network with the principal/primary investigators (PIs) too!

v. Make notes/summaries of poster presentations and talks from researchers in these labs.

vi. Make notes/summaries of their conference and journal papers. Try to get access to Ph.D. theses from their labs, and read them too. Pay special attention to the "Future Work" sections of the Ph.D. theses and conference papers.

vii. To secure a good postdoctoral research position, I must carry out "diligent background research aimed to answer the question, '**What do I want to get out of my postdoc?**'."

viii. The question, "What do I want to get out of my postdoc?," is analogous to the question for prospective Ph.D. students, "What do I want to get out of my Ph.D. program?"

ix. "Although they are short-term assignments, postdoc positions should be viewed as stepping stones to a longer-term independent career – whether in academia, industry or another science-related post."

x. "For that reason, it is hard to overstate the importance of the postdoc application. It is the fledgling scientist's bid to get noticed – to gain a phone or in-person interview with labs. Background research, a carefully crafted curriculum vitae (CV) and cover letter, and personalization of each application will open doors. Form letters and typos will get applicants nowhere."

xi. "I wanted to apply to proven, top-notch labs where I was going to have the success and track record of the people coming out of these labs" – Toby Franks

xii. **WARNING: Note that some postdoc positions require applications 12-18 months in advance.**

xiii. Treat the process of applying for postdoctoral research positions like applying for R&D jobs in the EDA/semiconductor industry. Apply to several positions simultaneously, so that I can interview in person with potential employers and lab colleagues prior to making my decision (about which offer of a postdoctoral research position to accept).

xiv. "An application typically consists of a cover letter introducing the applicant and his or her reasons for joining this particular lab; a CV outlining education, publication record, honors and accomplishments; and three referees who will provide supportive letters of recommendation on request. Some students also include a research summary of their graduate work; others incorporate this into their cover letter."

xv. "A little preparation goes a long way at this stage. Consider taking a workshop on writing cover letters and CVs, have senior colleagues review them, and proofread them carefully."

xvi. The cover letter and CV shall be free of mistakes.

xvii. Things that PIs look for when hiring postdocs:
   A. First-author conference and journal publications, which prove that I "can complete a project from start to finish" as a junior scientist.
   B. Given the size of my home institution and resources available to my Ph.D. research lab, what have I accomplished? PIs want to see if Ph.D. students can accomplish a significant amount of things, despite the constraints in resources.
   C. List experiences that illustrate non-research responsibilities, including sitting on grad

school or department committees, or hosting seminar speakers.

xviii. "Applicants should highlight what they hope to accomplish in general in a postdoc position. Specific details of projects should be left for the interview."

xix. Quoted: Agneta Nordenskjöld, a genetics researcher at the Karolinska Institute in Stockholm, advises spelling out your contributions to a graduate research project. "Write it in a way that says, 'I did this' or 'My part of the project was', especially if you did something outstanding," she says.

xx. "Those applying after taking a break from science must work harder to convince a lab head. Kristofor Langlais had been teaching high-school science at a ski academy in Vermont when he applied for postdoc positions in the Washington DC area."

xxi. "After extensive research into each lab's publications, websites and even annual reports, he wrote his cover letters from the angle of someone already in the lab. He mentioned specific results he found interesting and the next natural steps the lab might take. "I tried to make it sound like I could walk in that day and be self-sufficient immediately." He spent 20 hours or more on each application and his strategy paid off – he had four phone interviews, and ended up in a molecular-genetics fellowship at the US National Institute of Child Health and Human Development in Bethesda, Maryland."

xxii. Quoted: Likewise, when Xiaoli Du was finishing up her doctorate at Peking Union Medical College in Beijing, she knew she would need to send applications to 30-40 labs if she wanted to obtain a postdoc in the United States. But she avoided the form-letter strategy. " 'Dear Professor' does not show respect or that you are really interested in their lab," she says. Instead, she personalized each application and stated how her training and experience would distinguish her from other applicants. Her hard work led to a postdoc at the US National Cancer Institute in Bethesda, Maryland. Du suggests attending international meetings to make first contact with potential advisers.

xxiii. Quoted: Few things, though, confer more of an advantage than secured funding. "If a postdoc has their own fellowship, they can write their application to me in crayon and I'll take them," says Phil Baran, an organic chemist at the Scripps Research Institute in La Jolla, California. Unfunded applicants should assure the lab head that they have checked on specific fellowship possibilities and outline a plan to apply for them.

xxiv. Quoted: There are some definite 'wrong ways' to apply. Goldstein, whose e-mail inbox is so overloaded that his system sends an automated response to direct queries to assistants and lab managers, says there is no room for red flags in the competitive arena. Avoid telling personal-life woes, bad-mouthing previous labs or advisers or expressing a desire to work at night so that you can surf during the day. Explain gaps in a CV or publication record.

xxv. Quoted: "Anything that signals the person is a prima donna, no matter how great they are, I don't go for," says Ken Yamada, laboratory chief at the National Institute of Dental and Craniofacial Research in Bethesda, Maryland. "Research requires teamwork."

xxvi. Quoted: Lab heads want a clear indication that applicants have carefully thought through their career goals and chosen this lab as the appropriate stepping stone. "Does a genuine passion, drive, and hunger for research come through in their letters or on the phone?" asks Yamada. "Would they be doing the same thing if they were suddenly independently wealthy?"

xxvii. Postdoc application to-do list:

A. Send your application by e-mail or overnight delivery. Consider a paper packet if you have unpublished manuscripts you want to include.

B. Make it easy for lab heads to contact you by e-mail or phone.

C. Follow-up by e-mail in 1-2 weeks to make sure they received your application. Don't phone.

D. Choose referees who really know you, such as collaborators, unofficial advisers or others beyond the standard committee members.
E. Meet with your referees to explain your career goals to them.
F. Encourage referees to send their letters promptly (Salk Institute cell biologist Martin Hetzer says that the speed with which a letter lands in his inbox is usually much more telling than the letter's content).
G. Prepare for the possibility of phone interviews, which may be scheduled or spontaneous. Make sure the conversation is two-way and ask your own questions, too. Have a list of bullet points handy in case you get nervous.

(b) Rania Sanford, "How to navigate the road ahead," in *Nature*, Vol. 467, No. 7315, pp. 624, September 30, 2010. Available online at: http://dx.doi.org/10.1038/nj7315-624a; last accessed on December 31, 2010.

i. Check out how much mentoring is being provided to postdoctoral researchers (postdocs) by principal investigators (PIs).
ii. Check if the postdoc position allow me time to learn and seek guidance.
iii. Choose a research lab to do my postdoc, so that I can have a "meaningful experience," obtain "strong [research] results," and would be able to get "a desirable job" based on my postdoc research. Avoid research labs that don't allow me to satisfy these preferences.
iv. "Good mentoring is an acquired skill."
v. Find out if the PI:
A. Is aware of "the stages in developing a mentoring relationship"
B. Coaches students and postdocs, rather than supervises them
C. "Build[s] trust with protégés and postdocs' expectations"
D. Replicates the style of mentoring to all her/his mentees. Mentoring should be individualized/customized, since each person is a unique individual and differences exist between cultures, genders, and generations.
vi. Find out if the institution/university penalizes postdocs who have their own funding or fellowships (including competitive awards). The institution/university may provide medical insurance and other benefits to postdocs who are funded by research grants; hence, they are treated as staff members. If postdocs have their own funding or fellowships, they may not be eligible for medical insurance and other benefits that are provided by the university.
vii. Find out if support is given to non-US residents with regards to applying for non-immigrant J-1 visas or equivalent, relocation to the university's community, and assimilation into the university's community.

(c) Katharine Sanderson, "Training: The career doctor," in *Nature*, Vol. 467, No. 7315, pp. 623, September 30, 2010. Available online at: http://dx.doi.org/10.1038/nj7315-623a; last accessed on December 31, 2010.

(d) Karen Kaplan, "Careers and Recruitment: Industrial endeavours," in *Nature*, Vol. 461, No. 7263, pp. 554–555 September 24, 2009. Available online at: http://dx.doi.org/10.1038/nj7263-554a; last accessed on December 31, 2010. [Has information on industrial postdoctoral research positions in the biotech industry.]

(e) *naturejobs.com*:

i. Career toolkit:
A. Podcasts (*Naturejobs* podcasts):
- Available online at: http://www.nature.com/naturejobs/career-toolkit/podcasts/index.html; last accessed on December 31, 2010.

4. National Postdoctoral Association:

   (a) Resources on Becoming a Postdoc: http://www.nationalpostdoc.org/graduate-students
   (b) Faculty & Administrators: http://www.nationalpostdoc.org/faculty-administrators
   (c) Diversity Programs & Resources: http://www.nationalpostdoc.org/diversity-issues
   (d) International Postdocs: http://www.nationalpostdoc.org/international-issues
   (e) The NPA Postdoctoral Core Competencies Toolkit (NPA Core Competencies): http://www.nationalpostdoc.org/competencies

5. Research Foundation – Flanders (FWO), or "Fonds voor Wetenschappelijk Onderzoek – Vlaanderen" (FWO):
   (a) See http://www.fwo.be/en/index.aspx
   (b) This is sponsored by the National Fund for Scientific Research (Belgium)
   (c) Research Foundation - Flanders (FWO): five-yearly prizes:
      i. Two Dr. A. De Leeuw-Damry-Bourlart prizes (€100,000): For exact and applied sciences.
      ii. Two Dr. Joseph Maisin prizes: For fundamental biomedical sciences and clinical biomedical sciences.
      iii. The Ernest John Solvay prize: For humanities and social sciences.
   (d) The Research Foundation Flanders (FWO) pays a researchers income for the following categories:
      i. Ph.D. student
      ii. Postdoctoral researcher
      iii. Clinical fellowship
6. *PhDjobs.com*: http://www.phdjobs.com/
7. *innovation report*:
   (a) Jobs (in industry and academia): http://www.innovations-report.com/jobs/jobs.php
8. Princeton University:
   (a) Department of Computer Science, School of Engineering and Applied Science (SEAS):
      i. Jeff Erickson and Boaz Barak, *TCS opportunities: Postdocs and other positions in theoretical computer science*, Center for Computational Intractability, Department of Computer Science, School of Engineering and Applied Science (SEAS), Princeton University. Available at: http://intractability.princeton.edu/jobs/; last accessed on September 15, 2010.
9. Stony Brook University (State University of New York at Stony Brook):
   (a) Department of Computer Science:
      i. Prof. Radu Grosu. Available online at: http://www.cs.sunysb.edu/~grosu/; last accessed on February 17, 2010.
10. Computing Research Association:
   (a) PostDocs:
      i. http://cra.org/postdocs/
      ii. http://cra.org/postdocs/pdoc/thedoc.htm
      iii. http://cra.org/postdocs/Issues-PostDoc-1-28-2011.pdf
      iv. http://cra.org/postdocs/reading-list.php
      v. http://cra.org/postdocs/related-articles.php
      vi. Taulbee Survey: http://cra.org/resources/taulbee/
      vii. Computing Innovation Fellows (CIFellows) Project: http://www.cra.org/ccc/cifellows
11. Simons Foundation: Simons Postdoctoral Fellows Program, https://simonsfoundation.org/funding-guidelines/simons-postdoctoral-fellows-program
12. IBM Postdoctoral Fellowships:

   (a) IBM Herman Goldstine Postdoctoral Fellowship in Mathematical and Computer Sciences; see http://domino.research.ibm.com/comm/research_projects.nsf/pages/goldstine.index.html
   (b) Josef Raviv Memorial Postdoctoral Fellowship; see http://domino.research.ibm.com/comm/research.nsf/pages/d.compsci.josef.raviv.general.info.html, http://domino.research.ibm.com/comm/research.nsf/pages/d.compsci.raviv.winner.html, and http://domino.research.ibm.com/comm/research.nsf/pages/d.compsci.raviv.winner2008.html

13. Computing Innovation Fellows (CIFellows); post my profile on http://cifellows.org/profiles/; also see http://www.cifellows.org/
14. Society for Industrial and Applied Mathematics:
    (a) Job Search Resources for Students: http://www.siam.org/careers/resources.php
    (b) SIAM Job Board: http://jobs.siam.org/
    (c) Careers in the Math Sciences:
        i. http://www.siam.org/careers/sinews.php
        ii. Has advice about:
            A. planning for my career (life after grad school)
            B. research careers in research institutes (and/or corporate research labs)
            C. seeking a junior academic position in the US
            D. obtaining/procuring reference letters
15. The European Mathematical Society:
    (a) Open Positions: http://www.euro-math-soc.eu/jobs.html
16. Mathematical Optimization Society (MOS; formerly known as Mathematical Programming Society, MPS):
    (a) Job Postings: http://www.mathprog.org/?nav=links
17. NSF Mathematical Sciences Institutes: http://mathinstitutes.org/

5.28 Resources Concerning the Application for Junior Faculty Positions

Resources concerning the application for junior faculty positions:

1. Computing Research Association (CRA):
   (a) http://www.cra.org/for-students/
   (b) CRA's Job Announcements: http://www.cra.org/ads/
   (c) Computing Postdoc Job Opportunities: http://cifellows.org/opportunities
   (d) Computing Postdoc Profiles: http://cifellows.org/profiles
   (e) Mailing lists of the Computer Research Association's Committee on the Status of Women in Computing Research (CRA-W):
       i. Subscribe to these and receive announcements concerning openings for junior faculty positions.
       ii. http://www.cra-w.org/mailinglists
       iii. http://www.cra-w.org/PhdjobhuntHers
2. American Association of University Professors:
   (a) Career Center: http://careercenter.aaup.org/search.cfm
   (b) Issues in Higher Education: http://www.aaup.org/AAUP/issues/
3. The Chronicle of Higher Education:
   (a) Global Jobs: http://chronicle.com/section/Global-Jobs/434/
   (b) Great Colleges to Work For:

      i. http://chronicle.com/section/Great-Colleges-to-Work-For/156/
      ii. http://chronicle.com/section/The-Academic-Workplace/156
      iii. http://chronicle.com/article/Great-Colleges-to-Work-For/65724/
   (c) CV Doctor: http://chronicle.com/article/The-CV-Doctor-Is-Back/49086/

4. Nature Publishing Group (a division of Macmillan Publishers Limited):
   (a) *naturejobs.com*:
      i. Jobs and careers: http://www.nature.com/naturejobs/index.html
      ii. Career toolkit:
         A. http://www.nature.com/naturejobs/career-toolkit/index.html
         B. Salaries:
            - http://www.nature.com/naturejobs/career-toolkit/salaries/index.html
            - Nature Publishing Group, "Naturejobs international salary survey, 2010," in *naturejobs.com*: Career toolkit: Salaries: Salary survey, 2010. Available online at: http://www.nature.com/naturejobs/salary/survey/2010/index.html; last accessed on December 31, 2010.:
               – There are significant differences in the incomes of researchers between academia and industry in the U.S. and in Europe.
               – Academics in Northern and Central Europe tend to make more than those in Southern Europe.
               – Academics in the U.S. still dominate in terms of wages and benefits.
               – There are less differences between the salaries of men and women in Southern Europe than the rest of Europe and the U.S..
         C. Podcasts:
            - Nature Publishing Group, "Lecturer jobs," *naturejobs.com*: Career toolkit: Podcasts. Available online at: http://media.nature.com/download/nature/podcast/naturejobs/naturejobs-2010-06-25.mp3; last accessed on December 31, 2010.
               – Being a tenured/tenure-track professor involves spending about 20% of my time on teaching, 10% of my time on administrative activities, and the rest of my time (70%) on research.
               – Being a tenured/tenure-track or teaching professor involves: curricular design; and dealing with students from a broad spectrum of cultures and abilities (including students with disabilities and behavioral problems) – undergraduates tend to start college in their late teens, and many of them are not sufficiently matured.
               – The time spend on research would involve: managing a research lab; hiring students (undergraduates and grad students), postdocs, and technicians; and applying for funding.
               – Two stages of getting tenure: 1) find a research topic/problem to work on, plan my research projects, broaden my social network of researchers so that I can collaborate with them or allow some members of the review committee for funding applications to get to know me, and be a co-instructor for graduate classes (& possibly, lead the discussions/tutorials) and teach undergraduate classes; 2) write and submit research grant applications, & start teaching classes as the primary instructor.
               – Try to finish writing research papers based on my postdoctoral research projects prior to becoming a professor, so that I can start writing research grants based on my publications.

- Take note that my start-up fund as a junior professor may be low (say tens of thousands of dollars/Euro). This may prevent me from doing much research, unless only a small capital is needed to get my research started.
- When writing my grant proposal, specify and explain what would make my proposed technique stand out from the others.
- Give talks, including those in department seminars.

(b) Quirin Schiermeier, "Graphic Detail: The real value of a scientist's wage," in *Nature*, Vol. 450, No. 7170, pp. 597, November 29, 2007. Available online at: http://dx.doi.org/10.1038/450597a; last accessed on December 31, 2010. ["Graphic Detail: The real value of a scientist's wage – Researchers' spending power is not what it seems."]
   i. Salaries of academics in Switzerland are the highest, but Switzerland has the highest cost of living.
   ii. Salaries in Northern and Central Europe are better than those in Southern Europe.
   iii. Also, salaries in the US and Japan are comparable to those in Northern and Central Europe.
   iv. Salaries in Israel are comparable to those in Southern Europe. E.g., they are comparable to those in France and Italy.

5. Times Higher Education (THE):
   (a) Advanced Job Search: http://www.timeshighereducation.co.uk/jobs_home.asp?navCode=84
   (b) Career: http://www.timeshighereducation.co.uk/section.asp?navcode=96
6. *Mathematical Association of America* (MAA):
   (a) Information for new Ph.D.s seeking academic careers
   (b) http://www.maa.org/careers/
   (c) MAA Math Classifieds: http://www.mathclassifieds.org/home/index.cfm?site_id=1925
7. American Mathematical Society:
   (a) Information on applying for positions in academia: http://www.ams.org/programs/students/programs/students/gradinfo/gradinfo
   (b) http://www.ams.org/profession/career-info/new-phds/new-phds
8. Association for Women in Science (AWIS) career library: http://www.awis.affiniscape.com/displaycommon.cfm?an=1&subarticlenbr=249
9. American Psychological Association's resources for grad students and postdocs: http://www.apa.org/education/grad/index.aspx
10. *HigherEdJobs.com*: http://www.higheredjobs.com/
11. IEEE:
   (a) IEEE Real World Engineering Projects (RWEP):
      i. resources to help plan and run projects in EECS and related fields that will benefit humanity
      ii. http://www.realworldengineering.org/
   (b) IEEE Circuits and Systems Society, "How to Have a Successful Academic Career?," in the *PhD/GOLD Special Session* at the IEEE International Symposium on Circuits and Systems, Paris, France, May 31, 2010. Available online at: http://cassnewsletter.org/Volume4-Issue3/GOLD_News.html; last accessed on February 28, 2010.:
      i. Topics of the talks include:
         A. "Time management - Balancing Research and Teaching and Volunteer Work with Life," by Maciej Ogorzalek, IEEE Fellow, CASS Past President (2008), Jagiellonian University. Kraków, Poland

B. "Your First Job: Learning the Ropes and Riding the Waves," by Ljiljana Trajkovic, IEEE Fellow, CASS Past President (2007), Simon Fraser University, Vancouver, Canada
C. "How to Turn Your Profession as a University Professor into the Best Job in the World," by Wouter Serdijn, Editor-in-Chief, TCAS-I, Delft University of Technology, The Netherlands
D. "Mobility and Multi-Cultural Working Experiences, Key Elements to Reach the Desired Academic Job," David Atienza, CASS BoG Member, EPFL, Switzerland

12. Howard Hughes Medical Institute:
    (a) More Resources for Scientific Management Training:
        i. http://www.hhmi.org/resources/labmanagement/resources.html
        ii. Budget planning
        iii. Project planning
        iv. Communication skills
        v. Managing logistics
        vi. How to set up your research lab/group?
13. iBerry (collection of job lists): http://iberry.com/cms/jobs.htm
14. Common CV Network (for research and academic positions in Canada): http://www.cvcommun.net/index_e.html
15. University of Pennsylvania:
    (a) Stephanie Weirich, *Computer Science Faculty Job Search Resources*, Department of Computer and Information Science, School of Engineering and Applied Science, University of Pennsylvania. Available at: http://www.seas.upenn.edu/~sweirich/resources.htm; last accessed on September 5, 2010.
16. Blue Lab Coats, *Academic Job Applications*. Available at: http://bluelabcoats.wordpress.com/application-pkgs/; last accessed on September 10, 2010.
17. Inside Higher Ed: http://www.insidehighered.com/career/seekers
18. Advice on giving the job talk:
    (a) John Farrell, "What to Say in a Good Research Talk," Department of Computer Science, James Cook University, May 1994. Available at: http://www.computersciencestudent.com/SS/HowTo/ResearchTalkJohnFarrell.html; last accessed on August 25, 2010.
19. —
20. **Residential Education**:
    (a) Telluride Association:
        i. Information about how to become a Faculty Fellow at the Cornell Branch and the Michigan Branch of Telluride Association, which are "residential colleges": http://www.tellurideassociation.org/programs/college_faculty.html
21. —
22. **Awards, grants, and fellowships to look out for**:
    (a) National Science Foundation:
        i. Research Experiences for Undergraduates (REU): http://www.nsf.gov/crssprgm/reu/reu_search.cfm and http://www.nsf.gov/funding/pgm_summ.jsp?pims_id=5517&org=NSF
        ii. NSF Career Grant / NSF Career Award
    (b) Heinz Family Philanthropies, The Heinz Awards: http://www.heinzawards.net/awards
    (c) Sloan Research Fellowships; Alfred P. Sloan Foundation:

i. Applicants must "hold a Ph.D. (or equivalent) in chemistry, physics, mathematics, computer science, economics, neuroscience or computational and evolutionary molecular biology, or in a related interdisciplinary field"
ii. Applicants must be tenure-track junior faculty in a North American college or university
iii. http://www.sloan.org/fellowships
iv. Eligibility Requirements: http://www.sloan.org/pages/28/eligibility-requirements

(d) Santa Fe Institute's Miller Distinguished Scholarship (for excellent interdisciplinary researchers): http://www.santafe.edu/research/miller-scholars/
(e) Lemelson-MIT Prize (for mid-career innovators): http://web.mit.edu/invent/a-prize.html
(f) Research Corporation for Science Advancement:
i. Cottrell College Science Awards (for science researchers at predominantly undergraduate US colleges and universities): http://www.rescorp.org/cottrell-college-science-awards
ii. Cottrell Scholar Awards (for science researchers at US colleges and universities): http://www.rescorp.org/cottrell-scholar-awards/

(g) *Semiconductor Industry Association* University Researcher Awards: http://www.sia-online.org/cs/papers_publications/press_release_detail?pressrelease.id=1722
(h) Packard Fellowships for Science and Engineering (for professors at selected US universities): http://www.packard.org/genericDetails.aspx?RootCatID=3&CategoryID=152
(i) *American Society for Engineering Education* awards: http://www.asee.org/activities/awards/division.cfm
(j) Hellman Fellowship in Science and Technology Policy (for early-career professionals with training in science or engineering who are interested in transitioning to a career in public policy and administration): http://www.amacad.org/hellman.aspx
(k) Simons Foundation's *Collaboration Grants for Mathematicians*: https://simonsfoundation.org/funding-guidelines/current-funding-opportunities/collaboration-grants-for-mathem
(l) The National Science Foundation:
i. Alan T. Waterman Award (for junior faculty in the US): http://www.nsf.gov/od/waterman/waterman.jsp

(m) Computing Research Association (CRA): A. Nico Habermann Award, http://www.cra.org/awards/habermann-current/
(n) Microsoft New Faculty Fellowship Award:
i. http://research.microsoft.com/en-us/collaboration/awards/msrff_all.aspx#2005
ii. Microsoft Corporation, "Faculty Fellowship Recipients," from *Microsoft Research: Microsoft Research Connections: Faculty Fellowship Program*, Microsoft Corporation, Redmond, WA, 2014. Available online at: http://research.microsoft.com/en-us/collaboration awards/msrff_all.aspx; last accessed on August 5, 2014.
iii. Microsoft Corporation, "Faculty Fellowship Program," from *Microsoft Research: Microsoft Research Connections*, Microsoft Corporation, Redmond, WA, 2014. Available online at: http://research.microsoft.com/en-us/collaboration/awards/msrff.aspx; last accessed on August 5, 2014.

(o) Global Semiconductor Alliance (GSA): Dr. Morris Chang Exemplary Leadership Award

23. —
24. **Underrepresented minority outreach**:
(a) The Mathematical Association of America:
i. Pre-College Programs: http://www.maa.org/funding/pre_college.html
ii. Programs for Undergraduate Students: http://www.maa.org/funding/undergraduate.html

iii. Special Interest Group of the MAA on Research in Undergraduate Mathematics Education (SIGMAA RUME): http://sigmaa.maa.org/rume/. Also, see http://sigmaa.maa.org/rume/highlights.html.

25. —

26. **Resources for preparing the teaching statement, and developing a teaching philosophy and methodology**:
   (a) Missouri University of Science and Technology (formerly University of Missouri-Rolla):
      i. Department of Electrical and Computer Engineering:
         A. Cp Eng 382 - Teaching Engineering (Summer 20XY):
            - http://ece.mst.edu/documents/CpE_382.doc
            - Learn about:
               – teaching objectives and techniques
               – knowledge taxonomies
               – cognitive development
            - Learn how to:
               – use course objectives to design a course
               – communicate using traditional and cutting-edge media, which includes making effective presentations via various communication modes, including traditional lecture, transparencies, and PowerPoint slides
               – select textbook(s) and recommended books
               – assess student learning
               – grade/evaluate assignments, presentations, and projects in a way that addresses the psychological aspects of learning
               – recognize and adapt to different student learning styles,
               – facilitate cooperative/active learning
               – improve student discipline
            - 3. Develop course content and student evaluation techniques to meet the elements of ABET criteria 4. Critique evaluation methods including homework, exams, quizzes, reports/papers, and portfolios 5. Learn to speak and/or write competently on educational hot topics, including collaborative learning, distance education, self-paced learning, web tools/ educational software, and learning communities 6. Develop course objectives 7. Describe the parts of a syllabus, and prepare a syllabus 8. Describe methods to prevent cheating 9. Describe the elements of effective student advising at both the undergraduate and graduate levels 10. List what makes one a good listener 11. Prepare an academic application and identify the issues involved in the academic hire 12. Describe appropriate responses to student situations such as death, depression, cheating, pregnancy, hugs 13. Name sources of additional teaching training
   (b) Sergio Toral Marin, María del Rocío Martínez-Torres, Federico Barrero, "Reforming ICT Graduate Programs to Meet Professional Needs," *Computer*, vol. 43, no. 10, IEEE Computer Society Press, pp. 22-29, June 2010, DOI:10.1109/MC.2010.186.:
      i. To access this article, use its DOI Bookmark: http://doi.ieeecomputersociety.org/10.1109/MC.2010.186 or http://dx.doi.org/10.1109/MC.2010.186.
      ii. It discusses how ICT Masters degree programs in Europe can meet the needs and demands of the ICT industry and society. That is, it discusses how well European universities empower Masters students for jobs and careers in the ICT industry.
      iii. Note that the definition and usage of ICT is non-standard among researchers of computer science and engineering higher education in the United States. Here, I am referring to researchers (mostly professors and Ph.D. students) who do research about::

A. the quality of computer science and engineering education;
B. how well do these education programs empower computer science and engineering students to pursue the careers that they desire; and
C. issues concerning the diversity of the student body and faculty in computer science and engineering.

iv. ICT is reasonably well-defined and used in Europe, and poorly defined and used elsewhere (especially in Asia).
v. In American, programs related to the computing professions are: computer science; computer engineering; electrical engineering; and information systems. There may be more esoteric "majors" (as as "gaming and animation"), but these majors don't usually have recommended frameworks and standards for their curriculum.

(c) ACM and IEEE Computer Society are professional organizations for the IT industry (and related industries, such as the semiconductor and electronic industries) that are leaders in helping educators plan and design their curriculum in computer science and engineering.:
i. A gist of what computer science and computer engineering degree programs should empower students to have are listed in the short article on, "Skills You'll Learn if You Study Computing". Available online at: `http://computingcareers.acm.org/?page_id=15`; last accessed on November 22, 2010.
ii. Recommended computer science and computer engineering curricular can be found online at: `http://www.acm.org/education/curricula-recommendations` or `http://www.computer.org/portal/web/education/Curricula`; last accessed on November 22, 2010:
A. As a Ph.D. student in computer science (or computer engineering), you should have most of the skills and knowledge specified in the recommended curricular for students in computer science (or computer engineering).
B. You mostly probably need these skills and knowledge to pass your Ph.D. preliminary exam in a decent US CS Ph.D. program.
iii. ACM Special Interest Group on Computer Science Education (SIGCSE): `http://www.sigcse.org/`
iv. Association for Computing Machinery's Special Interest Group for Information Technology Education (SIGITE): `http://www.sigite.org/`
v. "Future of Computing Education Summit": `http://www.acm.org/education/future-of-compu`

(d) ABET, Inc. (formerly the Accreditation Board for Engineering and Technology):
i. ABET, *Download Accreditation Criteria and Forms*, 2010:
A. `http://www.abet.org/forms.shtml`
B. Look at the sections on "student outcomes," and "program criteria (including curriculum)."
C. ABET Engineering Accreditation Commission, "Criteria for Accrediting Engineering Programs: Effective for Evaluations During the 2011-2012 Accreditation Cycle," ABET, Baltimore, MD, October 30, 2010. Available online at: `http://www.abet.org/Linked%2520Documents-UPDATE/Program%2520Docs/abet-eac-criteria-2011-2012.pdf`; last accessed on November 22, 2010.
D. ABET Engineering Accreditation Commission, "Criteria for Accrediting Computing Programs: Effective for Evaluations During the 2011-2012 Accreditation Cycle," ABET, Baltimore, MD, October 30, 2010. Available online at: `http://www.abet.org/Linked%2520Documents-UPDATE/Program%2520Docs/abet-cac-criteria-2011-2012.pdf`; last accessed on November 22, 2010.

(e) American Institute of Mathematics:
i. Resources for the Math Community:
A. `http://www.aimath.org/mathcommunity/`

B. David W. Farmer, “The AIM REU: individual projects with a common theme,” in the *Proceedings of the Conference on Promoting Undergraduate Research in Mathematics*, American Mathematical Society, 2006. Available online at: http://www.aimath.org/mathcommunity/farmerREU.pdf; last accessed on January 9, 2010. [“AIM Research Experience for Undergraduates (REU)”]

C. Sally Koutsoliotas and David W. Farmer, “Preparing students to give talks,” American Institute of Mathematics. Available online at: http://www.aimath.org/mathcommunity/studenttalks.pdf; last accessed on January 9, 2010. [“Preparing students to give talks”]

5.29 Notes about Higher Education in Europe

Notes about Higher Education in Europe:

1. When I look at higher education programs in Europe, note the amount of ECTS credits for each class that I can, or have to, take. Determine if I am chalking up enough ECTS credits to fulfill the requirements of the educational program, and of the Bologna process.
2. The purpose of the **Bologna process** (or Bologna accords) is to create the European higher education area by making academic degree standards and quality assurance standards more comparable and compatible throughout Europe, in particular under the Lisbon Recognition Convention. [This means that if I obtain a degree in a European university, which abides by the Bologna process, that degree can be recognized anywhere in Europe that also abides by the Bologna process. And, if I were to study abroad, the Bologna process makes it easier to get my credits/classes recognized in my home institution, and get them counted towards graduation for my academic program]
3. **European Credit Transfer and Accumulation System (ECTS)** is a standard for comparing the study attainment and performance of students of higher education across the European Union and other collaborating European countries.
4. The **ECTS grading scale** is a grading system defined in the ECTS framework by the European Commission.
5. See http://www.eua.be/ for the European University Association (EUA)
6. Networks for Ph.D. students:
   (a) Eurodoc (= European Council of doctoral candidates and young researchers) is the European lobby organisation for young researchers and scientists. See http://www.eurodoc.net/
   (b) International PhD forum, a project of the Technical University of Kaiserslautern and the Mercator foundation. See http://www.docfor.uni-kl.de/
   (c) This initiative for doctoral students is a network of graduates from various science foundations and grants programs. See http://www.promovierenden-initiative.de/ ... FIX THIS
7. a note on how to apply for Ph.D. programs in Germany/Europe; see http://www.uni-jena.de/Doctoral_Studies-lang-en.html ... FIX THIS

5.30 Professional Organizations, Research Networks, and Consortiums

Professional Organizations, Research Networks, and Consortiums:

1. Institute of Electrical and Electronics Engineers (IEEE):
   (a) IEEE Computer Society
   (b) IEEE Circuits and Systems Society
   (c) IEEE Solid-State Circuits Society
2. Association for Computing Machinery (ACM):

   (a) **ACM Special Interest Group on Design Automation (SIGDA)**
   (b) ACM Special Interest Group on Embedded Systems (SIGBED)
   (c) ACM Special Interest Group on Algorithms and Computation Theory (SIGACT)
   (d) ACM Special Interest Group on Artificial Intelligence (SIGART)
   (e) ACM Special Interest Group on Computer Architecture (SIGARCH)
   (f) ACM Special Interest Group on Microarchitecture (SIGMICRO)
   (g) ACM Special Interest Group on Programming Languages (SIGPLAN)
   (h) ACM Special Interest Group on Symbolic and Algebraic Manipulation (SIGSAM)
   (i) ACM Special Interest Group on Simulation and Modeling (SIGSIM)
   (j) ACM Special Interest Group on Software Engineering (SIGSOFT)
   (k) ACM Special Interest Group on Measurement and Evaluation (SIGMETRICS)

3. Institute for Operations Research and the Management Sciences (INFORMS); see http://www.informs.org/
   (a) INFORMS Artificial Intelligence Section
   (b) INFORMS Computing Society
   (c) INFORMS Applied Probability Society
   (d) INFORMS Optimization Society
   (e) INFORMS Simulation Society
4. Society for Industrial and Applied Mathematics (SIAM); see http://www.siam.org/
   (a) SIAM Activity Groups (SIAGs)
   (b) SIAM Activity Group on Computational Science and Engineering (SIAG/CS&E)
   (c) SIAM Activity Group on Optimization (SIAG/OPT)
   (d) SIAG on Discrete Mathematics (SIAG/DM)
   (e) SIAM Activity Group on Analysis of Partial Differential Equations (SIAG/APDE)
   (f) SIAM Activity Group on Linear Algebra (SIAG/LA)
   (g) SIAM Activity Group on Supercomputing (SIAG/SC)
   (h) SIAM Activity Group on Geometric Design (SIAG/GD)
   (i) SIAM Activity Group on Control and Systems Theory (SIAG/CST)
   (j) SIAM Activity Group on Dynamical Systems (SIAG/DS)
5. Formal Methods Europe (FME); see http://www.fmeurope.org/
6. Association for the Advancement of Artificial Intelligence (AAAI); see http://www.aaai.org/home.html
7. Mathematical Programming Society (MPS); http://www.mathprog.org/
8. American Mathematical Society (AMS); see http://www.ams.org/

---

9. Italian Association for Artificial Intelligence / Associazione Italiana per l'Intelligenza Artificiale (AI*IA); see http://www.aixia.it/
10. Italian Operational Research Society / Associazione Italiana di Ricerca Operativa: http://www.airo.org/

---

11. CATRENE (Cluster for Application and Technology Research in Europe on NanoElectronics):
    (a) http://www.catrene.org/index.php
    (b) a recent EUREKA programme
    (c) **Funding source**; see http://www.catrene.org/web/calls/prepare_pp.php

12. ArtistDesign European Network of Excellence on Embedded Systems Design (ArtistDesign); see http://www.artist-embedded.org/artist/
13. European Network of Excellence on High Performance and Embedded Architecture and Compilation (HiPEAC); see http://www.hipeac.net/
14. CORDIS: An information space for European Research and Development (R&D) and exploitation of European R&D results; http://cordis.europa.eu/home_en.html
15. European Heads of Research Councils (EUROHORCs):
    (a) http://www.eurohorcs.org/E/Pages/home.aspx
    (b) "EUROHORCs is a European association of the heads of research funding organisations (RFO) and research performing organisations (RPO)."
    (c) Initiatives: Money Follows Researcher: Participating granting organizations: http://www.eurohorcs.org/E/initiatives/mfr/Pages/participating.aspx
    (d) Members: http://www.eurohorcs.org/E/members/Pages/default.aspx
16. European Science Foundation (ESF):
    (a) http://www.esf.org/
    (b) Member Organizations: http://www.esf.org/about-us/79-member-organisations.html and http://www.esf.org/about-us/79-member-organisations/full-list-of-esf-member-orga
    html
17. European Commission:
    (a) EU Research: http://ec.europa.eu/research/index.cfm?lg=en
    (b) EU 7^{th} Research Framework Programme:
        i. European Research Council (ERC):
            A. http://erc.europa.eu/
    (c) European Research Area:
        i. http://ec.europa.eu/research/era/index_en.htm
        ii. http://ec.europa.eu/research/era/understanding/what/what_is_era_en.htm
        iii. "The European Research Area is composed of all research and development activities, programs and policies in Europe which involve a transnational perspective."
        iv. "Together, they enable researchers, research institutions and businesses to increasingly circulate, compete and co-operate across borders."
        v. "The aim is to give them access to a Europe-wide open space for knowledge and technologies in which transnational synergies and complementarities are fully exploited."
        vi. "ERA consists of activities, programs and policies which are designed and operated at all levels: regional, national and European."

18. edacentrum: German R&D Network for Electronic Design Automation; see http://www.edacentrum.de/
19. Gigascale Systems Research Center (GSRC):
    (a) established to perform long-range, collaborative research that address the challenges of chip design and test that we expect to be facing in the 8-12 year timeframe
    (b) see http://www.eecs.berkeley.edu/Research/Areas/Centers/
20. European Electronic Chips & Systems design Initiative; see http://www.ecsi.org/ecsi/
21. Project on Advanced Research of Architectures and Design of Electronic Systems (**PARADES**), a European Group of Economic Interest among Cadence, Magneti-Marelli and SGS-Thomson

22. National Centre for Scientific Research (Centre national de la recherche scientifique or CNRS)

23. (Italian) National Research Council, or Consiglio Nazionale delle Ricerche (CNR); see http://www.cnr.it/sitocnr/Englishversion/Englishversion.html

---

24. Spanish Association for Artificial Intelligence; see http://aepia.dsic.upv.es/indexenglish.html
25. Catalan Association for Artificial Intelligence (ACIA); see http://www.acia.org/ang/index.php
26. European Coordinating Committee on Artificial Intelligence (ECCAI); http://www.eccai.org/
27. Complexity-NET: http://www.complexitynet.eu
28. Complex Systems Society (CSS): http://cssociety.org/tiki-index.php
29. Institute of Neuromorphic Engineering (INE): http://www.ine-web.org/
30. Colognet (Network of Excellence in Computational Logic):
    (a) http://www.colognet.org/ (**FAILED HYPERLINK**) or http://www.ist-world.org/ProjectDetails.aspx?ProjectId=4c6789a77c34458ba79cd404f3be47f0&SourceDatabaseId=951ef712ecba4f7ca229527509db4a81
    (b) Constraint Logic Programming; see http://slash.math.unipd.it/
    (c) Logic for Natural Language Processing; see http://nlp.colognet.org/
    (d) COMPULOG Americas, or Logic Programming in the Americas; see http://www.cs.nmsu.edu/~complog/

Other organizations:

1. Mathematical Association of America (MAA); see http://www.maa.org/
2. National Postdoctoral Association (NPA); see http://www.nationalpostdoc.org/

5.31 SIGDA Primary Areas of Interests

SIGDA primary areas of interest:

1. Electrical-level circuit and timing simulation
2. Discrete simulation
3. Critical path analysis and timing verification
4. Power estimation
5. Testing, fault modeling and simulation, TPG, test validation and DFT
6. Design and implementation verification (excluding layout verification)
7. Floorplanning and placement
8. Global and detailed routing
9. Module generation and compaction, transistor sizing and cell library optimization, layout verification
10. Technology independent, combinational logic synthesis
11. Technology dependent logic synthesis, library mapping, interactions between logic design and layout
12. Sequential logic synthesis and optimization
13. High-level synthesis
14. Asynchronous logic synthesis
15. Hardware Description Languages
16. Hardware/Software co-design, partitioning, system-level specification and design aids

17. Software synthesis and retargetable compilation
18. Hardware/Software co-simulation
19. Interconnect and packaging modeling and extraction
20. Signal integrity and reliability analysis
21. Analog, mixed-signal, and RF design tools
22. Microsensor and microactuator design tools
23. Statistical design and yield maximization

5.32 Information About Research Laboratories in Industry

Information about research laboratories in industry:

1. IBM:
    (a) IBM Research:
        i. IBM Austin Research Laboratory: Computer-Aided Design and Analysis group
        ii. IBM Haifa Research Lab
        iii. IBM Thomas J. Watson Research Center:
            A. Design Automation Department, in Yorktown Heights, NY
            B. logic synthesis
            C. physical synthesis
            D. statistical timing analysis
            E. static analysis
            F. power analysis
            G. design for manufacturing
            H. system-level design automation
    (b) IBM Dublin Software Laboratory
    (c) IBM Centers for Advanced Studies (CAS); see https://www-927.ibm.com/ibm/cas/
2. Intel:
    (a) Intel Capital
    (b) Israel Strategic Investments in Intel Capital
    (c) Intel Labs:
        i. (previously Corporate Technology Group)
        ii. primary new focus areas for the Labs group:
            A. mobility: includes mobile internet devices
            B. visual computing
            C. system-on-chip design
            D. enterprise
            E. power efficiency and "eco-innovation":
                - includes new sub-group geared to energy systems, such as smart grids and smart homes
        iii. existing activities:
            A. microprocessing/programming
            B. circuits and systems
        iv. new activities:
            A. integrated platforms
            B. future technologies
        v. Intel China Labs has been promoted to be a standalone unit within the broader Labs activity
        vi. All the units report to CTO Justin Rattner, also an Intel senior fellow and director of Intel Labs, which employs 1,000 researchers in 10 locations
        vii. like IBM's new programs, most of the projects are not bluesky, but geared to near term or midterm commercialization [*say 3-7 years*]

viii. "A result that doesn't find its way into a product doesn't have much value... We actually welcome failure in the labs. If we don't have enough failures, something is wrong; we're not taking enough risk and chasing the larger goals." – Intel CTO Justin Rattner, at an open day for Intel Labs; see http://www.rethink-wireless.com/?article_id=1587

ix. Mike Mayberry, vice president of the Technology and Manufacturing Group, describes the distinction between his organization and what's now called Intel Labs as follows: "Technology and Manufacturing does the process R/D and runs all the Intel manufacturing facilities. Intel Labs' job is to figure out what we should build, and how users would live with it if we did."; see http://www.edn.com/blog/1690000169/post/1970045797.html?nid=3351&rid=8705909

x. `Intel Developer Forum (IDF)`

xi. see http://blogs.intel.com/research/

(d) Design & Test Technology group

(e) Logic Verification Research Group

(f) Validation Research Lab

(g) Intel's SoC Enabling Group

(h) Low Power Design Technology, Enterprise Platforms Group, Intel Corp; Address: 3600 Juliette Lane, MS: SC12-603, Santa Clara, CA 95051

(i) Intel-UPC Barcelona Research Center, in Barcelona, Spain:
   i. (or Intel Barcelona Research Center)
   ii. Intel Labs - UPC

(j) Intel Circuits Research Labs, Hillsboro, OR

(k) Intel Design Center, Haifa, Israel

(l) Logic and Validation Technology, Design Technology Division, Intel Development Center (IDC), Haifa, Israel

(m) Logic and Validation Technologies, Intel Architecture Group in Haifa, Israel

(n) Formal Verification Group @ Intel, Haifa

(o) Intel CAD Department in Haifa

(p) Future CAD Technologies, Formal Verification Group, MATAM - Advanced Technology Center, P.O.B. 1659, Haifa 31015, Israel

(q) Intel Corporation, RA2-451, 2501 NW 229th Ave, Hillsboro, OR, 97124

(r) Principal Engineer, Strategic CAD Labs, Intel Corporation, Hillsboro, Oregon, USA

(s) Logic Verification, Strategic CAD Labs, Intel Corporation

(t) Intel Strategic CAD Labs, Hillsboro, Oregon, US

(u) Steven Burns, technical leader of the convergent design flow group in Intel's Strategic CAD Labs (SCL)

(v) Positions: Intel Fellow

(w) Greg Taylor: Intel Fellow and Director of Advanced Circuit Design Research

3. Fondazione Bruno Kessler, Trento:
   (a) Fondazione Bruno Kessler (FBK), formerly known as IRST
   (b) Istituto Trentino di Cultura – Istituto per la Ricerca Scientifica e Tecnologica (ITC-IRST)
   (c) see http://sra.itc.it/

4. Cadence Design Systems:
   (a) Cadence European Laboratories

   (b) Cadence Research Laboratories (CRL); see http://www.cadence.com/cadence/cadence_labs/Pages/default.aspx

5. Synopsys:
   (a) Advanced Technology Group: http://research.synopsys.com/ [According to Himanshu Jain, Synopsys no longer runs this research group. Synopsys has broken it up into different groups.]
   (b) Synopsys Fellow Mike Keating

6. Xilinx Research Labs:
   (a) Head of the DSP Systems and Applications research group, Xilinx Labs. USA
   (b) Xilinx Research Labs will initially recruit PhD qualified engineers as Principal Researchers, and PhD or experienced Masters degree holders as Staff Research Engineers
   (c) Xilinx folks mostly focus on higher level problems (system level design, hardware compilation, runtime reconfiguration, etc.)
   (d) Xilinx Research Labs has an internal webpage (visible only to Xilinx employees on our network)
   (e) Xilinx Boulder: open applications for Ph.D. positions

7. Altera Toronto Technology Centre:
   (a) Altera R&D center in Toronto

8. Sun Microsystems Laboratories, or Sun Labs; see http://research.sun.com/
9. NVIDIA:
   (a) NVIDIA Research (or NVIDIA Research Group)
   (b) http://research.nvidia.com/, or http://www.nvidia.com/object/nvidia_research.html
   (c) "*CUDA Research Centers* are recognized institutions that embrace and utilize GPU computing across multiple research fields":
      i. Barcelona Supercomputing Center, UPC (Spain)
      ii. Clemson University
      iii. HP Labs
      iv. Massachusetts General Hospital - Northeastern University
      v. Swinburne University of Technology (Australia)
      vi. University of California at Los Angeles, UCLA
      vii. University of Warsaw (Poland)
      viii. John Hopkins University
      ix. Nanyang Technological University
      x. Technical University of Ostrava (Czech Republic)
      xi. CSIRO (Australia)
      xii. ICHEC (Ireland)
      xiii. SINTEF (Norway)
   (d) "*CUDA Teaching Centers* are institutions that have integrated GPU computing techniques into their mainstream computer programming curriculum":
      i. American University of Beirut (Lebanon)
      ii. Florida A&M University
      iii. Hood College
      iv. McMaster University (Canada)
      v. University of California at Los Angeles - UCLA
      vi. University of Minnesota
      vii. University of North Carolina at Charlotte
      viii. State University of New York, Potsdam
      ix. California Polytechnic State University, San Luis Obispo

x. ITESM (Mexico)
xi. Czech Technical University (Czech Republic)
xii. Qingdao University (China)

(e) "*CUDA Center of Excellence program*, [which is an] elite network of 11 institutes focused on advancing parallel computing on the GPU":
i. Cambridge University
ii. Chinese Academy of Sciences
iii. Georgia Institute of Technology
iv. Harvard University
v. University of Maryland
vi. National Taiwan University
vii. Tokyo Tech
viii. Tsinghua University
ix. University of Illinois at Urbana-Champaign
x. University of Tennessee
xi. University of Utah

10. Qualcomm: Qualcomm Research Center:
    (a) `http://www.qualcomm.com/innovation/research/index.html`
    (b) Qualcomm Research Silicon Valley: `http://www.qualcomm.com/about/research/locations/bay-area`
11. Fujitsu Laboratories of America, Inc. (FLA) [Fujitsu American Research Lab]; see `http://www.fujitsu.com/us/about/other/fla/`
12. Infineon Technologies:
    (a) Corporate Research; Munich, Germany ... *Corporate Research Group??? Or Corporate Research Labs???*
13. Texas Instruments' Kilby Research Labs (or Kilby Labs)
14. Mitsubishi Electric Research Labs (MERL); see `http://www.merl.com/`
15. INESC; see `http://www.inesc-id.pt/`:
    (a) INESC is a leading research Institute in Lisbon, associated with IST, the Engineering School of the Technical University of Lisbon, Portugal.
    (b) Within INESC, the Algorithms for Optimization and Simulation and the Software Algorithms and Tools for Constraint Solving groups have been developing into a center of excellence for applied research in CAD.
    (c) One of the main focuses of research has been in the area of Modeling and Simulation of Large-Scale Parasitics.
    (d) The Electronic System Design and Automation (ESDA) group works on (co)synthesis algorithms for the acceleration of generic applications by migrating its more computational intensive parts to dynamically configurable hardware
    (e) `http://algos.inesc-id.pt/algos/projects.php?proj=IE02001`
16. ST Microelectronics: Functional Verification Group (Grenoble, France); STMicroelectronics Berkeley Labs
17. Samsung: Samsung Austin Semiconductor Research Center (SARC), or Samsung Semiconductor Austin R&D Center
18. Renesas:
    (a) Renesas Design Corporation (RDC):
        i. EDA Technology Group: Development of EDA systems, and provision of user support
        ii. Software Development Group: Development of EDA software

iii. Renesas EDA advanced technology development

19. Google:
    (a) Google Labs; see http://www.googlelabs.com/
    (b) Google Research; see http://research.google.com/
20. Bell Laboratories (Bell Labs); see https://networks.nokia.com/bell-labs and https://www.bell-labs.com/connect/internships/. Old link: http://www.alcatel-lucent.com/wps/portal/BellLabs.
21. HP Labs: http://www.hpl.hp.com/
22. Xerox:
    (a) Xerox Research Centre Europe (XRCE); Meylan, France: http://www.xrce.xerox.com/About-XRCE and http://www.xrce.xerox.com/About-XRCE/Internships
23. PARC (Palo Alto Research Center, Inc.); see http://www.parc.com/
24. General Electric:
    (a) Advanced Manufacturing and Software Technology Center
    (b) GE Global Research; see http://ge.geglobalresearch.com/ or http://www.ge.com/research/
25. Philips:
    (a) Philips Research; Eindhoven, The Netherlands
26. Yahoo! Research; see http://research.yahoo.com/; Also, see Yahoo! Labs: http://labs.yahoo.com/
27. Adobe's Advanced Technology Labs; see https://research.adobe.com/. Formerly, at: http://www.adobe.com/technology/
28. Apple Inc.:
    (a) Apple's Advanced Computation Group (ACG); see http://www.apple.com/acg/
    (b) Speech and Language Technologies Group
29. Avaya Labs Research; http://www.research.avayalabs.com/
30. Nokia Research Center: http://research.nokia.com/
31. Disney Research: http://disneyresearch.com/index.html
32. Ford Research & Advanced Engineering (Dearborn, MI)
33. NEC Labs America (Cupertino, CA): http://www.nec-labs.com/
34. Coverity, Inc.
    (a) Coverity Security Research Laboratory (SRL). Reference: Coverity, "Coverity Announces Formation of Security Research Laboratory," in *Press Releases*, Coverity, January 24, 2012. Available online at: http://www.coverity.com/html/press/coverity-announces-formation-of-html; last accessed on April 12, 2012.
35. Tata Consultancy Services Limited:
    (a) TCS Innovation Labs: http://www.tcs.com/about/tcs_difference/innovation/tcs_labs/Pages/default.aspx
36. Simula Research Laboratory: http://simula.no/
37. CA Technologies: CA Labs, http://www.ca.com/us/about-us/Innovation/ca-labs.aspx
38. OpenDNS: OpenDNS Security Labs
39. Accenture Technology Labs: Dublin, Ireland.
40. Toyota Research Institute

Note that Intel and IBM are probably the only IDMs that would make significant, if not considerable, investment into developing their own EDA tools. Also, most IDMs would shed their internal CAD R&D way before they drop their fabs to become fabless; see http://www.dac.com/newsletter/shownewsletter.aspx?newsid=53.

5.33 Information About Research Institutes

Information About Research Institutes:

1. Alan Turing Institute Almere (ATIA): http://www.alanturinginstitutealmere.nl/
2. Artificial Intelligence Research Institute (IIIA):
   (a) see http://www.iiia.csic.es/
   (b) part of the "Spanish Scientific Research Council" (CSIC); see http://www.csic.es/index.do
   (c) Reasoning and Logic:
      i. Approximate Reasoning and Soft computing:
         A. Many real problems cannot be modelled in the frame of Classical Logic.
         B. Different methods and their formalization to deal with uncertainty and imprecision are studied and applied to build intelligent systems.
         C. Approximate Reasoning based on:
            - Fuzzy, Multiple-valued and Possibilistic logics
            - Similarity Logic
            - Probabilities
         D. Fuzzy measures
         E. Aggregation operators
         F. Data fusion
         G. Decision theory
         H. Possibilistic logic programming
      ii. Automated Reasoning:
         A. This field deals with deduction problems expressed in some kind of Logic.
         B. Currently, we work actively with classical, multivalued and higher order Logics.
         C. The goal is to chain adequately the inferences to solve automatically the deduction problems at hand.
         D. As the deduction problems we treat are difficult to solve, the analysis and optimization of the needed resources, mainly the time, is mandatory.
         E. Tractability in Non Clausal forms
         F. Satisfiability problem
         G. Automated Deduction in Many-valued Logics
         H. Higher-Order Logics (Unification problem)
      iii. Constraint Satisfaction:
         A. It involves finding values to variables satisfying some constraints.
         B. If all constraints cannot be satisfied simultaneously, a solution is the assignment that best respects the constraints.
         C. While this problem is computationally intractable, many real problems can be modelled as constraint satisfaction.
         D. Algorithms for Max-CSP and non-binary constraints
         E. Generic and symmetry-based heuristics
         F. Interleaved depth-first search
         G. Distributed constraint satisfaction
      iv. Incremental design of formal specification
   (d) Learning Systems:

        i. Case-Based Reasoning
        ii. Data Privacy
        iii. Integration of Problem Solving and Learning
        iv. Machine Learning for Music
    (e) Multiagent Systems:
        i. Autonomous Robots
        ii. Electronic Institutions
        iii. Expert Systems
        iv. Negotiation
        v. Personal Information Agents
        vi. Trust and Reputation

3. Broad Institute:
    (a) http://www.broadinstitute.org/
    (b) Affiliated with Harvard University and MIT
    (c) Carries out research in biology and related fields
4. Howard Hughes Medical Institute (HHMI): http://www.hhmi.org
5. Industrial Technology Research Institute:
    (a) Internship program: http://www.itri.org.tw/eng/econtent/careers/careers05.aspx
    (b)
6. Institute of Cognitive Sciences and Technologies:
    (a) http://www.istc.cnr.it/
7. Institute for Scientific Interchange (ISI Foundation) (Turin/Torino):
    (a) http://www.isi.it/
8. IDSIA (Istituto Dalle Molle di Studi sull'Intelligenza Artificiale) / Dalle Molle Institute for Artificial Intelligence
9. Leibniz Institute in Forschungsverbund Berlin e.V.:
    (a) http://www.fv-berlin.de/
    (b) Weierstrass Institute for Applied Analysis and Stochastics (WIAS):
        i. http://www.wias-berlin.de/
        ii. Research at WIAS focuses on the following application areas:
            A. Nano- and optoelectronics
            B. Optimization and control of technological processes
            C. Phase transitions and multifunctional materials
            D. Flow and transport processes in continua
            E. Random phenomena in nature and economy
10. **New England Complex Systems Institute (NECSI)**; see http://necsi.org/index.html
11. RENCI (Renaissance Computing Institute): http://www.renci.org/
12. **Santa Fe Institute**; see http://www.santafe.edu/
13. SINTEF Group (Scandinavia): http://www.sintef.no/Home/
14. **SRI International**, founded as Stanford Research Institute; see http://www.sri.com/
15. The Allen Institute for Artificial Intelligence: http://allenai.org/index.html.
16. Tyndall National Institute (Cork, Ireland; focus on semiconductor manufacturing and solid-state devices): http://www.tyndall.ie/

5.34 Information About Ph.D. and Postdoc Fellowships

Information about Ph.D. and postdoc fellowships:

1. IEEE Circuits and Systems Society Pre-Doctoral Scholarships:
   (a) IEEE Circuits and Systems Society Pre-Doctoral Scholarship #1:
      i. CAS will support a scholarship for one pre-doctoral student to visit a research institution or another university abroad for up to one year: US$15k
      ii. Selection criteria: A CAS Member for at least two years and is enrolled in a Ph.D. program related to the fields of interest of CAS may submit a proposal (up to 3 pages) including
         A. CV (with academic records attached)
         B. Certificate/evidence of the enrollment in a Ph.D. program (student ID card)
         C. Description of the Ph.D. study plan
         D. The expected outcome of the research program and how it advances the state-of-the-art
         E. Endorsements from both institutions
         F. Brief description of the importance of receiving the CAS scholarship
   (b) IEEE Circuits and Systems Society Pre-Doctoral Scholarship #2:
      i. CAS will offer a pre-doctoral fellowship to recognize a young member enrolled in a Ph.D. program in a field related to the Circuits and Systems Society field of interest. The winner of the award will receive paid travel to ISCAS 2010 to receive the award: US$25k (stipend of US$15k, tuition and fees up to a maximum of US$8k, and a grant of US$2k to the department in which the recipient is registered)
      ii. Selection criteria: A CASS Member for at least two years and is enrolled in a Ph.D. program related to the field of interest of CAS may submit a proposal (up to 3 pages) including
         A. CV (with academic records attached)
         B. Certificate/evidence of the enrollment in a Ph.D. program (student ID card)
         C. Description of the Ph.D. study plan
         D. The expected outcome of the research program and how it advances the state-of-the-art
         E. Evidences showing the significance of the research (papers, patents)
         F. Endorsement from his/her supervisor
         G. Funding rules
2. Intel:
   (a) Intel Foundation Fellowship:
      i. Intel Foundation Ph.D. Fellowship
      ii. see http://www.intel.com/education/highered/studentprograms/fellowship.htm
      iii. This awards two-year fellowships to Ph.D. candidates pursuing leading-edge work in fields related to Intel's business and research interests.
      iv. Fellowships are available at select U.S. universities, by invitation only, and focus on Ph.D. students who have completed at least one year of study.
      v. The fellowship includes a cash award (tuition/fees/stipend), an Intel mentor, and the opportunity to participate in an internship at Intel.
3. Google:
   (a) Google Fellowship Program; see http://googleblog.blogspot.com/2009/05/best-and-brightest html
4. NVIDIA:

   (a) NVIDIA Fellowship Program; see http://www.nvidia.com/content/research/graduate-fellowsh html

5. Semiconductor Research Corporation:
   (a) member companies; see http://grc.src.org/member/about/membercompanies.asp:
      i. Intel
      ii. IBM
      iii. Cadence Design Systems
      iv. Mentor Graphics Corporation
   (b) http://grc.src.org/member/students/fellowships.asp
   (c) International Graduate Fellowship Program (IGFP); see http://grc.src.org/member/about/about_igfp.asp
   (d) unis; see http://grc.src.org/member/about/participatinguniversities.asp
   (e) From http://www.src.org/compete/ about anticipated Calls for GRC White Papers:
      i. **Computer-Aided Design and Test Sciences (CADTS) - Logic and Physical Design - Open January 2011**
      ii. **Computer-Aided Design and Test Sciences (CADTS) - Verification - Open July 2011**
   (f) Computer-Aided Design and Test Sciences (CADTS) researchers work to (see http://www.src.org/program/grc/cadts/):
      i. support the design of systems of greatly increased complexity, which include reused IP, significant analog and mixed signal function, and innovative circuit designs
      ii. address the increasing interaction between different levels of design, the greater influence of interconnect and power considerations, and the increased difficulty of test at higher speed
      iii. maintain and improve levels of design productivity, correctness, testability, signal integrity, manufacturability, and reliability through innovative tools and methods
   (g) Research in **Design Verification** includes basic model checking, processor verification, hybrid methods, asynchronous system verification, and coverage analysis. Design verification needs of GRC members also include tools for system level and architectural verification, hybrid techniques, and improved model checking.
   (h) Researchers in **Test and Testability** work to develop crosstalk fault models, low-cost self test, analog/mixed signal test, defect diagnosis, and SOC test time reduction. Test and testability needs of GRC members include the development of test for mixed signal, BIST, embedded cores, and signal integrity.
   (i) Research work in **Logic Design** includes logic synthesis of system-on-chip design, embedded software, library optimization, logic design with layout, and tools for power estimation. Logic design needs of GRC members include synthesis for increased complexity, physical and electrical issues, and analog/mixed signal.
   (j) Researchers in **Physical Design** work to develop optimization algorithms, interconnect modeling, wire-size optimization, power/ground design and analog layout synthesis. Physical design needs of GRC members also include synthesis and interconnect planning and estimation; and physical design for analog, RF, and mixed-signal.

6. IBM:
   (a) http://www-304.ibm.com/jct01005c/university/scholars/phdfellowship
   (b) IBM Ph.D. Fellowship Award
   (c) The IBM Ph.D. Fellowship Awards is an intensely competitive program which honors exceptional Ph.D. students in many academic disciplines and areas of study, for example: computer

science and engineering, electrical and mechanical engineering , physical sciences (including chemistry, material sciences, and physics), mathematical sciences (including optimization), business sciences (including financial services, communication, and learning/knowledge), and service sciences, management, and engineering.

   (d) IBM Herman Goldstine Postdoctoral Fellowship in Mathematical Sciences: http://domino.research.ibm.com/comm/research_projects.nsf/pages/goldstine.index.html

7. AMD: http://developer.amd.com/programs/fellowship/Pages/default.aspx
8. Gottlieb Daimler and Karl Benz Foundation:
   (a) Ph.D. fellowship for international students to study in Germany; see http://www.daimler-benz-stif de/home/fellowship/en/start.html
9. American Society for Engineering Education: http://blogs.asee.org/fellowships/
10. IET: Hudswell International Research Scholarship, http://www.theiet.org/about/scholarships-award ambition/postgraduate1/hudswell-what.cfm; and IET Postgraduate Scholarship, http://www.theiet.org/about/scholarships-awards/ambition/postgraduate1/postgrad-what.cfm.
11. Yahoo! Labs: http://labs.yahoo.com/ksc
12. Microsoft: Microsoft Research PhD Fellowship, http://research.microsoft.com/en-us/collaboration awards/fellowships.aspx
13. Facebook Fellowship Program: http://www.facebook.com/careers/fellowship.php
14. Kurt Gödel Research Prize Fellowship:
   (a) 2 Ph.D. (pre-doctoral) fellowships
   (b) 2 post-doctoral fellowships
   (c) 1 unrestricted fellowship
   (d) [Scope of the] original fellowship proposals [includes] the areas of:
      i. set theory
      ii. recursion theory
      iii. proof theory/intuitionism
      iv. model theory
      v. computer assisted reasoning
      vi. philosophy of mathematics
   (e) All fellowship proposals, regardless of subject area, will be judged according to:
      i. the relevance and resemblance of the research (finished and proposed) to the great insights and originality of Kurt Gödel
      ii. its general interest and clarity of motivation
      iii. its rigorous scientific quality and depth.
   (f) http://fellowship.logic.at/
15. Alexander von Humboldt-Stiftung/Foundation:
   (a) Feodor Lynen Research Fellowship for Postdoctoral Researchers (junior postdocs): http://www.humboldt-foundation.de/web/feodor-lynen-fellowship-postdoc.html
   (b) Friedrich Wilhelm Bessel Research Award (mid-career researchers): http://www.humboldt-foundati de/web/bessel-award.html
   (c) Georg Forster Research Fellowship for Postdoctoral Researchers (for non-German junior postdocs "with above average qualifications"): http://www.humboldt-foundation.de/web/georg-forster-fellowship-postdoc.html
   (d) Humboldt Research Fellowship for Postdoctoral Researchers (junior postdocs): http://www.humboldt-foundation.de/web/771.html
   (e) Sofja Kovalevskaja Award (junior postdocs): http://www.humboldt-foundation.de/web/kovalevskaja-award.html

   (f) Fraunhofer-Bessel Research Award: http://www.humboldt-foundation.de/web/fraunhofer-besse
       html
   (g) http://www.humboldt-foundation.de/web/home.html

16. Santa Fe Institute: Omidyar Postdoctoral Fellowship; see http://www.santafe.edu/education/fellowships/omidyar-postdoctoral/
17. Applied Materials: Applied Materials Graduate Fellowship
18. Industry Fellowships for Graduate Students (administered by EECS Berkeley): http://www.eecs.berkeley.edu/Programs/scholarship/Industry.html
19.

Germund Dahlquist graduate fellowship???

5.35 Information About Ph.D. Dissertation Awards

Information about Ph.D. dissertation awards:

1. ACM Outstanding Ph.D. Dissertation Award in Electronic Design Automation: http://www.sigda.org/opda.html
2. ACM Doctoral Dissertation Award: http://awards.acm.org/doctoral_dissertation/
3. EDAA Outstanding Dissertation Award (European Design and Automation Association, EDAA): http://www.edaa.com/dissertation_award.html and http://www.esat.kuleuven.be/micas/EDAA-Award/index.php
4. EuroSys Roger Needham PhD Award (in the systems area):
   (a) Areas in systems include:
      i. operating systems
      ii. distributed systems
      iii. real-time systems
      iv. systems aspects of databases
      v. language runtimes
      vi. **embedded systems**
      vii. computer networks
   (b) http://www.eurosys.org/phdprize/index.php
5. EURO Doctoral Dissertation Award (EDDA) (in operations research): http://www.euro-online.org/display.php?page=edda1
6. INFORMS George B. Dantzig Dissertation Award: http://www.informs.org/Recognize-Excellence/INFORMS-Prizes-Awards/George-B.-Dantzig-Dissertation-Award
7. Association for Symbolic Logic:
   (a) "The Sacks Prize is awarded for the most outstanding doctoral dissertation in mathematical logic".
   (b) http://www.aslonline.org/Sacks_nominations.html and http://www.aslonline.org/info-prizes.html
8. SIAM Richard C. DiPrima Prize:
   (a) The Richard C. DiPrima Prize is awarded every two years to a junior scientist, based on an outstanding doctoral dissertation in applied mathematics.
   (b) http://www.siam.org/prizes/nominations/nom_diprima.php
   (c) http://www.siam.org/prizes/sponsored/diprima.php
9. MOS A.W. Tucker Prize:
   (a) It is awarded for an outstanding doctoral thesis in any aspect of mathematical optimization.

   (b) http://www.mathprog.org/?nav=tucker

10. ACM SIGPLAN Outstanding Doctoral Dissertation Award: http://www.sigplan.org/award-dissertat htm
11. European Association for Computer Science Logic (EACSL):
    (a) Ackermann Award (for outstanding dissertations in Logic in Computer Science): http://www.eacsl.org/ and http://www.eacsl.org/award.html
12. European Coordinating Committee for Artificial Intelligence (ECCAI):
    (a) 201X Artificial Intelligence Dissertation Award: http://www.eccai.org/diss-award/current.shtml

Germund Dahlquist graduate fellowship??? Or, Germund Dahlquist prize???

5.36 Landmark Contributions by Students

Landmark contributions by students:

1. Bill Bonvillian, Susan Graham, Anita Jones, Ed Lazowska, Pat Lincoln, Fred Schneider, and Victor Zue, "Landmark Contributions by Students in Computer Science," Version 11, Computing Community Consortium blog, Computing Community Consortium, Computing Research Association, Washington, D.C., September 15, 2009. Available online at: http://www.cra.org/ccc/docs/Student_Achievements.pdf; last accessed on August 18, 2012.
   - Use of Boolean logic to model digital circuits: Claude Shannon at MIT. Shannon is better known for inventing information theory, but this work, his MIT Masters thesis, was a landmark. (1937)
   - Huffman coding: David Huffman at MIT. Huffman was a Ph.D. student when he invented Huffman coding as a term paper project in a course taught by Robert M. Fano. (1951)
   - Mathematical foundation of packet communication: Len Kleinrock at MIT. Kleinrock's Ph.D. dissertation, Message Delay in Communication Nets with Storage, established a crucial foundation for the ARPANET revolution, in which he played a central role. (1962)
   - Interactive computer graphics: Ivan Sutherland at MIT. Sutherland's Sketchpad system – his Ph.D. work – laid the groundwork for several decades of advances in computer graphics. Ed Catmull's 1974 Ph.D. dissertation, supervised by Sutherland at the University of Utah, was another landmark contribution to the field, as was John Warnock's 16-page 1969 University of Utah Ph.D. attacking the hidden surface problem. (1963)
   - Computer vision: Larry Roberts at MIT. Larry Roberts is best known for his role in the ARPANET effort, but his 1963 MIT Ph.D. dissertation, Machine Perception of Three-Dimensional Solids, laid out the entire process for computer vision as we know it today. (1963)
   - Symbolic mathematics: William A. Martin and Joel Moses at MIT. Martin's 1967 Ph.D. dissertation on symbolic algebra and Moses's 1967 Ph.D. dissertation on symbolic integration launched the MACSYMA project at MIT, the predecessor of todays ubiquitous Mathematica. (1967)
   - The FLEX language and machine: Alan Kay at the University of Utah. Kay's Ph.D. dissertation, The Reactive Engine, defined the FLEX language, the first extensible dynamic graphical interactive object-oriented language, and the direct antecedent of Kay's Smalltalk work at Xerox PARC. It also defined the FLEX machine, a desktop computer with a fully general display and a multiple clipping window user interface, the inspiration for Kay's Dynabook. Kay says "Most of us regarded what we were doing at PARC as 'phase two' of what we'd already started in graduate school." (1969)

- The Boyer-Moore theorem prover: Robert S. Boyer and J Strother Moore at the University of Edinburgh. Moore was a Ph.D. student and Boyer was a postdoctoral research associate. (1971)
- Efficient graph planarity testing using depth-first search: Bob Tarjan at Stanford. Tarjan's thesis marked a crucial advance in the depth and elegance of the analysis of data structures for basic computational problems. (1972)
- Ethernet: Bob Metcalfe at Harvard. The engineering was done at Xerox PARC, but the invention and analysis of binary exponential backoff as an alternative to the fixed-backoff (and thus unstable) Aloha Network scheme was Metcalfe's Ph.D. dissertation. (1973)
- BSD Unix: Bill Joy at Berkeley, working with Bob Fabry and Domenico Ferrari. (1977)
- VisiCalc: Bob Frankston and Dan Bricklin at MIT. The spreadsheet – the "killer app" for personal computing – was invented by Frankston and Bricklin while students at MIT LCS, and sold through the company they formed, Software Arts. (1979)
- Public key cryptography: Ralph Merkle at Berkeley and Stanford, along with Martin Hellman and Whitfield Diffie. Merkle's 1977 Berkeley Masters thesis and 1979 Stanford Ph.D. dissertation made fundamental contributions. Hellman maintains that the "Diffie-Hellman" key exchange protocol should be called "Diffie-Hellman-Merkle." Public key certificates, as used ubiquitously in HTTPS-based web sites, were introduced in the 1978 MIT Bachelors thesis of Loren Kohnfelder, advised by Len Adleman of RSA fame. (1979)
- The Sun workstation: Andy Bechtolsheim at Stanford, working with Forest Baskett. (1982)
- The Connection Machine: Danny Hillis at MIT. Hillis co-founded the Thinking Machines Corporation while an MIT AI Lab Ph.D. student. (1983)
- Sphinx (large-vocabulary, speaker-independent, continuous speech recognition): Kai-Fu Lee at Carnegie Mellon. While speech recognition has a long history both before and after Lee's work, his Sphinx system, the subject of his Ph.D. dissertation, was a landmark. (1988)
- Linux: Linus Torvalds at the University of Helsinki. Torvalds was a second-year computer science student at the University of Helsinki when he launched the Linux open source operating system project, inspired by Andy Tanenbaum's MINIX operating system and Richard Stallman's GNU project. (1991)
- BDD-based symbolic model checking: Ken McMillan at Carnegie Mellon, working with Ed Clarke. Symbolic Model Checking (the title of McMillan's Ph.D. dissertation) has transformed hardware and software verification. (1992)
- Mosaic: Mark Andreessen at the University of Illinois. Andreessen invented the revolutionary graphical Web browser while an undergraduate at the University of Illinois working at the National Center for Supercomputer Applications. He subsequently founded Netscape Communications. Microsoft licensed Mosaic from UIUC as the foundation for Internet Explorer. (1994)
- The PCP theorem: Sanjeev Arora at UC Berkeley. The PCP theorem (Probabilistically Checkable Proofs) is the cornerstone of the theory of computational hardness of approximation. (1994)
- Google: Larry Page and Sergey Brin at Stanford. The Page Rank algorithm is central to Google's success. (Note that many earlier web search innovations were also due to students; for example, Brian Pinkerton at the University of Washington invented WebCrawler in 1994, the first successful full-text web search engine.) (1998)
- Akamai (content delivery networks): Danny Lewin at MIT, working with Tom Leighton. (1999)
- Peer-to-peer file sharing: Shawn Fanning at Northeastern University. Fanning was an undergraduate at Northeastern University when he wrote the code for Napster, inventing the

concept of peer-to-peer file sharing, which has many important legal uses in addition to the questionable ones enabled by Napster. (1999)
- The foundations of rocketry and astronautics: Hermann Oberth at Göttingen. Oberth's Ph.D. dissertation, Die Rakete zu den Planetenräumen ("By Rocket into Planetary Space"), was rejected by the faculty of Göttingen as "utopian." Along with Tsiolkovsky and Goddard, he became one of the three (independent) founding fathers of rocketry and astronautics, the foundation of which was described in his dissertation. (1922)
- The discovery of the double helix: James Watson and Francis Crick at Cambridge University. Crick was a Ph.D. student and Watson a postdoctoral fellow at the time of their landmark discovery. (1953)
- The Josephson effect: Brian Josephson at Cambridge University. Josephson was a Ph.D. student in Physics when he discovered the Josephson effect (superconductivity), for which he received the Nobel Prize in 1973, at the age of 33. (1962)
- Fullerenes: Jim Heath, working with Robert Curl, Harold Kroto, and Richard Smalley at Rice University. Heath was the Chemistry Ph.D. student who conducted the experiments that generated the first C60 molecules and, ultimately, won the Nobel Prize in Chemistry for the three senior members of the collaboration. (1985)
- Alternate reference: Ed Lazowska, "Landmark Contributions by Students in Computer Science," Computing Community Consortium blog, Computing Community Consortium, Computing Research Association, Washington, D.C., August 28, 2009. Available online at: http://www.cccblog.org/2009/08/28/landmark-contributions-by-students-in-computer-science; last accessed on August 18, 2012.

2. Reference: "Tire Tracks" diagram (the role of research in creating billion dollar industry segments), on the web page for *Designing a Digital Future: Federally Funded Research and Development in Networking and Information Technology. Report to the President and Congress – President's Council of Advisors on Science and Technology, December 2010*, Department of Computer Science & Engineering, University of Washington, Seattle, WA, December 16, 2010. Available online at: http://www.cs.washington.edu/homes/lazowska/nitrd/TT.pdf; last accessed on August 18, 2012. See Figure 1 and 2.

6 CS Ph.D. Program Coursework at the University of Trento

6.1 Coursework Requirements

Number of required course credits for:
- ICT doctoral courses: $\geq$12
- ICT doctoral courses in the 1^{st} year: $\geq$15
- ICT doctoral courses in the 2^{nd} year: $\geq$9
- ICT advanced courses: $\leq$6

Italian Culture and Language course is compulsory for foreign students.

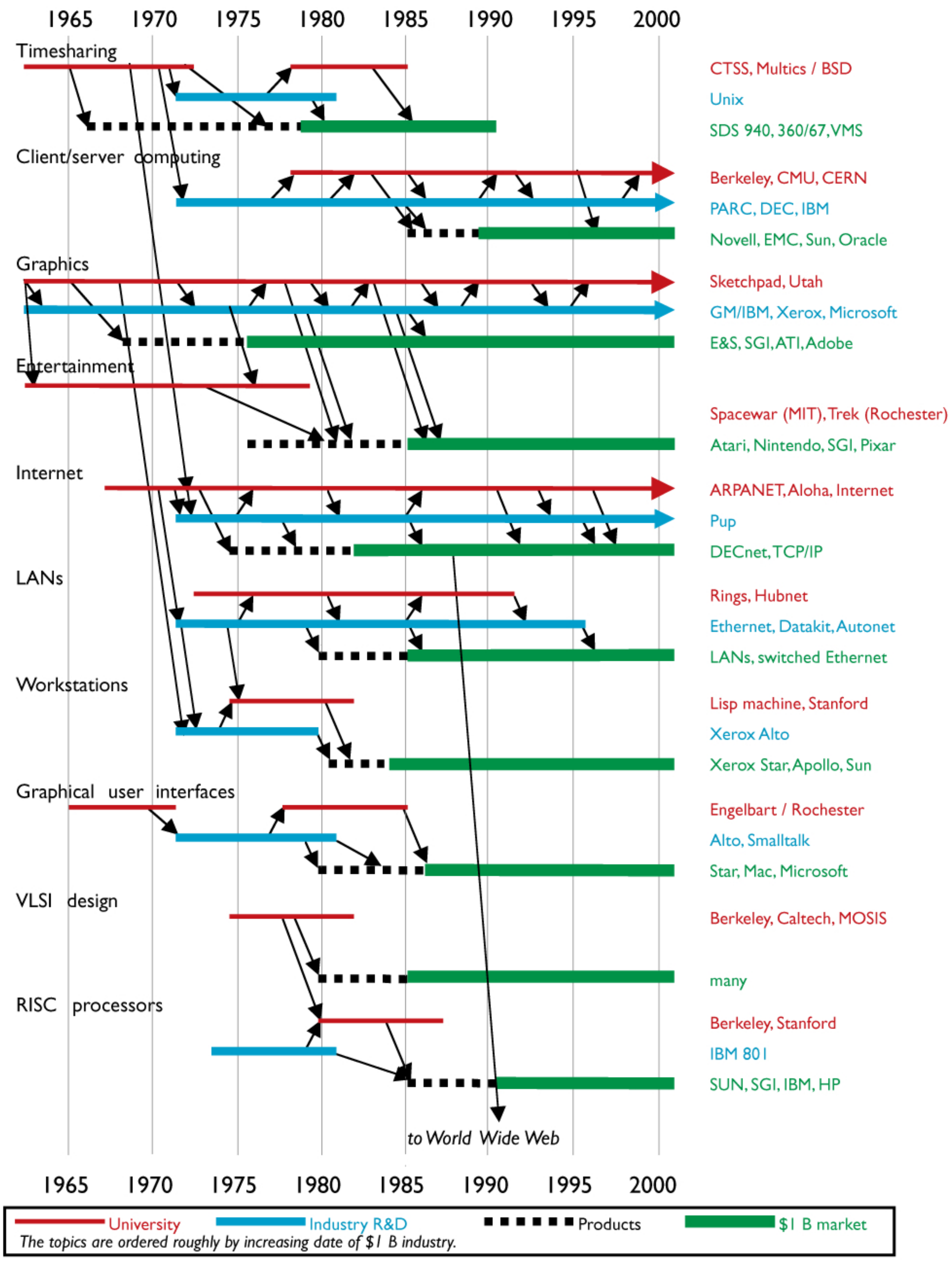

Figure 1: “Tire Tracks” diagram–The role of research in creating billion dollar industry segments. Part 1.

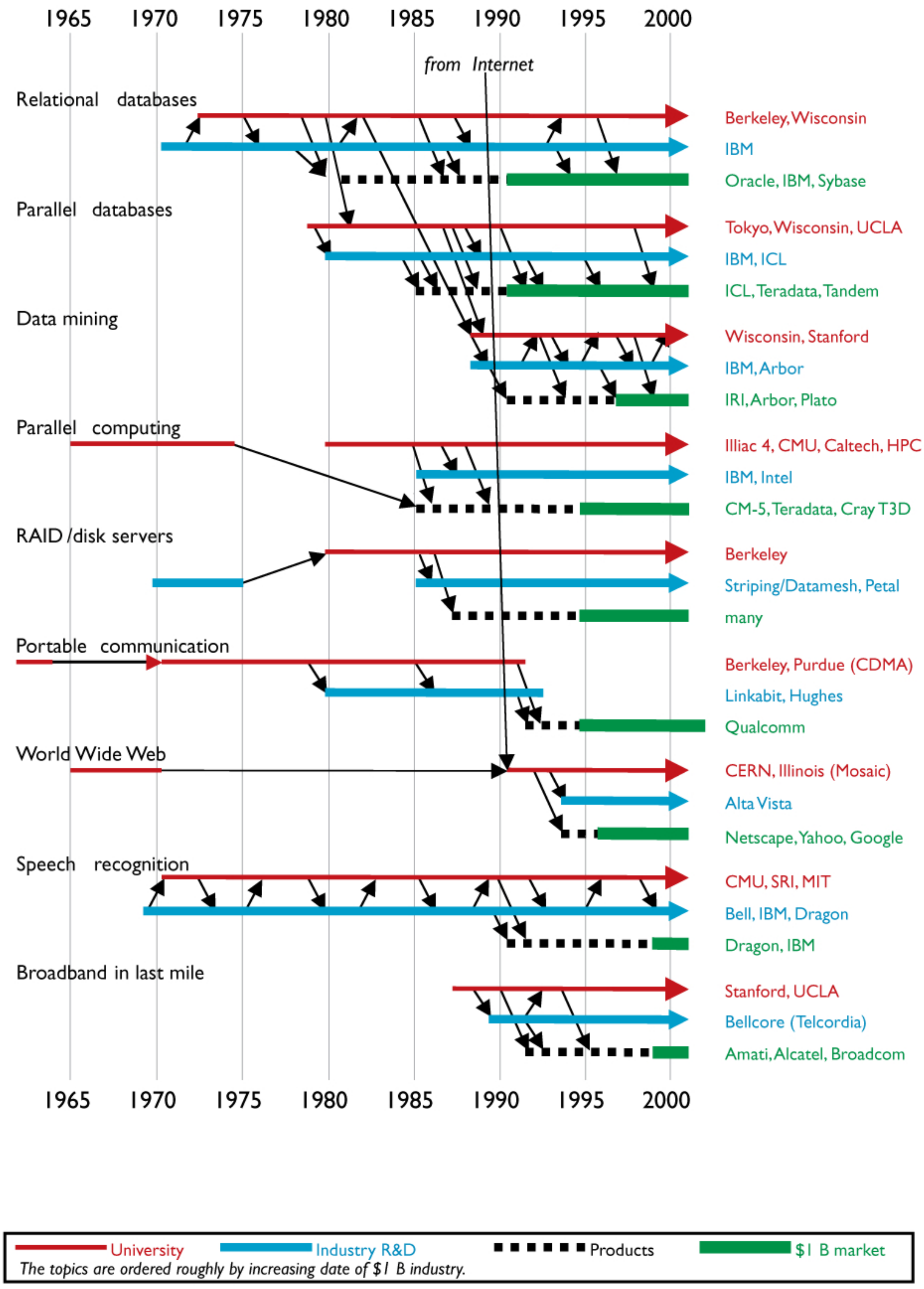

Figure 2: “Tire Tracks” diagram–The role of research in creating billion dollar industry segments. Part 2.

6.2 Doctoral Classes That I Plan To Take

ICT doctoral courses that I plan to take:

- See http://ict.unitn.it/edu/ict/courselist.xml?year=2009 for the list of classes
- First year:
  - Formal Verification of Programs
  - Languages for embedded systems design and programming
  - Optimization Methods
  - Solving Combinatorial Problems Using Stochastic Local Search
  - Laboratory of Embedded Systems
- Second year:
  - Advanced Topics in Formal Verification — Automata-based Decision Procedures
  - Advanced Embedded System Design on FPGA
  - Complex Systems
  - VHDL-based digital design (Maybe)
  - Advanced Topics in Software Engineering — Conceptual Modeling and Ontological Analysis (Maybe)

6.3 Doctoral Classes That I May Take

ICT doctoral courses that I may take, or classes under consideration:

- Research Methodology - I
- Research Methodology - II
- Machine Learning
- Technical English (First Session)
- Technical English (Second Session)
- Computational Systems Biology (Maybe)
- Entrepreneurship in High Tech Start Up (Maybe)
- Ethics of Computing (Maybe)

6.4 Advanced Classes That I Can Take

ICT advanced courses:

- For list of advanced classes that I can take up to 6 units, see http://mcs.dit.unitn.it/edu/compsciences/courses-mcs-esai.xml?cds_id=10117&aa_ord_id=2008&aa_off_id=2008&pds_id=10004
- Concurrency theory (6 credits/Fall)
- Mathematical Logic (6 credits/Fall/Maybe)
- Computational complexity (6 credits/Fall/Maybe)
- Computability (6 credits/Fall/Maybe)

6.5 High-Performance Computing (HPC) Resources

High-performance computing (HPC) resources:

1. `MacResearch.org`:
   (a) community for scientists and engineers using `Mac OS®X` for research
   (b) see http://www.macresearch.org/ and http://www.macresearch.org/openmacgrid
   (c) use the `OpenMacGrid` computing grid to facilitate my research
   (d) use the `MacResearch.org Script Repository` to get scripts to carry out various tasks for research; see http://www.macresearch.org/script_repository
2. NVIDIA:
   (a) Student Organization Sponsorship — By sponsoring engineering organizations for students, and assisting these organizations through donations, organizational support and guest lecturers, NVIDIA hopes to foster a stimulating environment for tomorrow's leading engineers.
   (b) To learn more about any of these opportunities, please email: hr-universityteam@nvidia.com
   (c) see http://www.nvidia.com/page/other_resources.html
   (d) Resources — Our award winning books are now freely available online in HTML
   (e) NVIDIA Developer Web Site; see http://developer.nvidia.com/page/home.html

7 Sources of Information

7.1 Google These Items

Google these search items:

- **"word-level algebraic SAT solver utilizes parallel processing on multi-core platforms to break a problem into several sub-problems to be solved in parallel to support larger designs and faster solution generation"**
- word-level formal verification (Does word-level formal verification exists? — google "word-length formal verification") — formally verify embedded systems, ESL verification, processors (including control logic, memory systems, and arithmetic circuits — Does the data path matter here? Check out on-chip memories)
- word-level verification ieeexplore.ieee.org
- Word-level decision diagrams like binary moment diagrams (BMDs)
- "Verification of Arithmetic Functions with Binary Moment Diagrams"
- binary moment diagrams (BMDs)
- "binary moment diagram" BMD ieeexplore.ieee.org
- SMT-based SMV???
- Satisfiability-Modulo Theory (SMT)-based Bounded Model Checking (BMC)
- SMT-based Sequential Equivalence Checking (SEC)
- SMT Satisfiability Modulo Theories verification VLSI processor
- "Parallel SAT solving in bounded model checking"
- Formal Verification of Pipelined Microprocessors by Correspondence Checking
- SAT-based bounded model checking ... SMT-based???
- "formal verification of communications in network-on-chips"
- SMT ($\equiv$ SMT Modulo Theories???)
- floating-point verification
- analog verification

- Coverage Metrics for Formal Verification
- Model checking... SAT-based model checking
- equivalence checking
- theorem proving
- model logic – first-order logic, second-order logic, and higher-order logic
- "propositional satisfiability" ieeexplore.ieee.org
- "Boolean satisfiability problem" ieeexplore.ieee.org
- "SAT solver" ieeexplore.ieee.org
- "Constraint Solving and Optimization"
- "Propositional approximations for bounded model checking of partial circuit designs"
- "Tool-support for the analysis of hybrid systems and models"
- "A Computing Procedure for Quantification Theory"
- "A machine program for theorem-proving"

- "Simple symbolic and numerical computations based on equations and inequalities"
- "automated reasoning"
- "constraint solving"

- NoC support for CMP/MPSoCs

- System-level / ESL / Behavioral / High-Level / Algorithmic Synthesis
- Behavioral simulation
- heterogeneous systems – electronic / mechanical / biological / optical

---

- lithography simulation check
- Lithography Simulation
- OPC verify, OPCverify, and OPC verification
- image-verification tool
- OPC – Optical Proximity Correction: rule-based & model-based
- RET – Reticle enhancement technology / Resolution Enhancement Technique
- SRAF – Sub Resolution Assist Features `a kind of OPC, mainly for RET`
- Computational lithography / computational scaling
- google "OPC DFM"

- CMP – Chemical-mechanical planarization or Chemical-mechanical polishing $\Longrightarrow$ `rule-based and optimization-driven MB tiling`
- chemical mechanical planarization polishing CMP process variation design for manufacturing manufacturability dummy metal fill insertion synthesis
- CMP fill synthesis
- dummy fill insertion
- metal fill density
- metal fill insertion $\Longrightarrow$ `metal density smoothness + metal thickness smoothness`
- rule based tiling tile

- DFM – design-for-manufacturing / Design for manufacturability
- PSM – Phase-shift mask

- OAI – Off-axis illumination
- MDP – Mask Data Preparation, prior to RET/OPC
- **Lithography $\Longrightarrow$ RET $\equiv$ OPC + PSM (+ OAI???)**
- Photolithography
- Nanolithography

- EM field solver / Electromagnetic field solver / field solver
- "architectural/organizing principles of multicellular computing"

Use http://www.google.com/help/features.html to help me search for information, if and when I have difficulties getting `Google search` to return "good" results.

7.2 Journals and Conference Proceedings to Check Out

Journals and conferences to check out:

1. *now*:
   (a) ACM Transactions on Design Automation of Electronic Systems (TODAES)
   (b) ACM Transactions on Embedded Computing Systems (TECS)
   (c) ACM Journal on Emerging Technologies in Computing Systems (JETC)
2. *later*:
   (a) ACM Transactions on Computer Systems TOCS
   (b) ACM Transactions on Programming Languages and Systems (TOPLAS)
   (c) ACM Transactions on Architecture and Code Optimization
   (d) ACM Transactions on Algorithms
   (e) ACM Transactions on Computational Logic (TOCL)
   (f) Journal of the ACM (JACM)
   (g) ACM Journal of Experimental Algorithmics
3. *way later*:
   (a) ACM Transactions on Modeling and Computer Simulation (TOMACS)
   (b) ACM Transactions on Autonomous and Adaptive Systems (TAAS)
   (c) ACM Transactions on Software Engineering and Methodology (ACM TOSEM)
   (d) ACM Transactions on Mathematical Software (TOMS)
   (e) ACM Transactions on Computation Theory (ToCT)

7.3 Web Pages To Visit

Web pages to visit:

- http://www.nascentric.com/moreinfo.html
- http://www.kenmcmil.com/
- http://domino.research.ibm.com/comm/research.nsf/pages/r.da.html
- http://www.research.ibm.com/da/DesignAuto.html
- http://tima.imag.fr/sls/publications_sls.html
- http://async.org.uk/tr.html
- http://www.ece.umd.edu/DSPCAD/papers/contents.html
- http://www.phil.cmu.edu/~avigad/formal/
- http://web.cecs.pdx.edu/~mperkows/CLASS_574/index.3.html

- http://eigold.tripod.com/
- http://w2.cadence.com/webforms/cbl_software/index.aspx — software tools from Cadence Design Systems
- web pages of Ed Clarke, and EDA labs at Berkeley & UT Austin
- EECS Ph.D. and MS dissertations, and technical reports at Berkeley; see http://www.eecs.berkeley.edu/Pubs/Dissertations/Years/1994.html
- ECE Ph.D. and MS dissertations, and technical reports at CMU; see http://www.ece.cmu.edu/research/publications/2007/?y=2007

- http://www.eecs.berkeley.edu/~newton/Presentations/ICCADVerif11_89/sld004.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/Arpa10_89/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/MCM9_92/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/DACKeynote6_95/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/DACKeyText.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/DACWomen6_98/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/McClymonds5_98/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/Haas9_96/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/EDARev8_96/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/EE290Seminar9_96/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/EECSColloq11_96/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/ee290a2_97/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/collabNeed/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/panelSlides/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/Si2010_9_97/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/ASEKeynote11_97pres/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/WebTutorial/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/SRC3_98/index.htm
- http://www.eecs.berkeley.edu/~newton/Presentations/WebArchTutorialPrint/index.htm

---

- http://ce055.cm.utexas.edu/Research/Sub_Files/Photomask_Lithography/
- http://willson.cm.utexas.edu/Research/Sub_Files/Photomask_Lithography/

Good sources of information:

1. UCSD EDA group – ABK; see http://vlsicad.ucsd.edu/Publications/Conferences/ and http://vlsicad.ucsd.edu/Publications/Journals/
2. http://www.eecs.berkeley.edu/~kuh/dissertations.html — Ph.D. dissertations of Prof. Ernest S. Kuh's students at UC Berkeley
3. collection of CS Ph.D theses from the Technion; see http://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-same-author.cgi?
4. theses and technical reports from Stanford; see http://infolab.stanford.edu/TR/csltr9x.html
5. theses from Caltech; see http://thesis.library.caltech.edu/information.html
6. EDA publications from UT Austin; see http://www.cerc.utexas.edu/utda/publications/publication
html

7. EDA publications from a professor at NTU; see http://cc.ee.ntu.edu.tw/~cchen/Publications.htm
8. papers on NoCs and SoCs, and theorem proving; see http://homepages.inf.ed.ac.uk/kgoossen/www/research.html
9. ACM SIGDA Super Compendium; see http://www.cs.york.ac.uk/rts/docs/SIGDA-Compendium-1994-papers/2004/year.htm (collection of papers from EDA), and http://www.cs.york.ac.uk/rts/docs/ (include CASES, CODES-ISSS, EMSOFT papers)
10. random collection of information for computer science; see http://www.ocf.berkeley.edu/~horie/project.html; see http://www.ranlog.com/ramsey/bookmarks.html; see http://pyre.third-bit.com/blog/archives/885.html/comment-page-1; see http://www2.iiia.csic.es/~vtorra/altres.php?llengua=en; see http://www.lsi.upc.edu/~mengjt/resource.html
11. "How To Become A Hacker," By Eric Steven Raymond; see http://catb.org/esr/faqs/hacker-howto.html
12. "The Cathedral and the Bazaar," By Eric Steven Raymond; see http://catb.org/esr/writings/cathedral-bazaar/
13. HiPEAC Roadmap; see http://www.hipeac.net/roadmap
14. EE Times 60 Emerging Startups; google this list of startups, since this list gets updated regularly

7.4 Random Information about Uni Rankings & Research Ecosystem

Random/Other information about grad school rankings, and the research ecosystem between academia, industry, and the government:

- US grad school rankings for CS programs; see http://www.cra.org/statistics/nrcstudy2/home.html
- Info on R&D in science and engineering in the US (Industry and academia); see http://sites.nationalacademies.org/cstb/index.htm
- `optimization`: http://www.sce.carleton.ca/faculty/chinneck/po.html, http://www.sce.carleton.ca/courses/sysc-5004/, http://www.sce.carleton.ca/courses/sysc-3200/
- http://openresearch.org/wiki/Main_Page
- ranking of European graduate schools; see http://www.excellenceranking.org/eusid/EUSID
- ranking of European graduate schools; see http://ranking.zeit.de/che9/CHE_en

7.5 Grad School Information

Things for me to look into when considering graduate and professional programs:

1. Is the graduate or professional program accredited?
2. Would attending the graduate/professional program help me obtain my career goals?
3. Find out what are the statistics regarding completion rates and time-to-degree duration for the academic program. E.g., on average, how long does it take for a student to graduate? What is the proportion of students who graduate within the expected time-to-degree period (e.g., 4 years for a Ph.D. program)? See http://www.princeton.edu/gradschool/facts/time_to_degree/engineering/ and http://gradschool.princeton.edu/facts/time_to_degree/naturalsciences/ for statistics of graduates from various programs at Princeton University.
4. Some statistics that you may wanna look into include for assessing the department (or at least, the academic program):
   (a) Average number of Ph.D. students per Ph.D. advisor (tells you about advisor-to-advisee ratios)

   (b) Number of Ph.D. graduates for the Ph.D. program per year: tells you about the size of the department, in terms of number of faculty members and graduate students (number of admitted students each year)
   (c) Graduate outcomes:
      i. What did alumni do after their Ph.D. programs?
      ii. How long did they take to find a job?
      iii. If they switched fields, or even professions,
   (d) Drop-out rate... What is the proportion of students who graduate within the expected time-to-degree period (e.g., 4 years for a Ph.D. program)?
   (e) Average duration for time-to-degree... On average, how long does it take for a student to graduate?
   (f) Failure rate for Ph.D. preliminary and qualifying examinations
   (g) Median time taken to pass Ph.D. preliminary examinations
   (h) Median time taken to pass Ph.D. qualifying examinations
   (i) Reasons for dropping out of the Ph.D. program

5. Some research labs do list the names of alumni members (Ph.D. and Masters students, undergraduates, and postdocs). They may also include their first vocation after their Ph.D. program, and what are they currently doing. You can use this to help you gauge the strengths/weaknesses of the lab/advisor... Determine if a high proportion of Ph.D. alumni members have switched fields.
6. Assessing the research lab:
   (a) Determine if there are publications from this research lab that are considered seminal papers.
   (b) Determine if the research lab is doing, or has recently been involved in, transformative research:
      i. Publications that led to the creation of a new research subtopic, and possibly a new research topic (or even a new research area).
      ii. Publications describing experimental results on publicly accessible benchmarks that indicate its superiority (e.g., a SAT solver that is 10-100X better than other existing SAT solvers)
      iii. Publications
   (c) Determine if the research is interdisciplinary
   (d) Determine if the research goals matches with mine
   (e) Determine the diversity of the research lab
   (f) Interact with lab members to determine if they are inclusive
   (g) Determine the quality of the lab members: undergrads, grad students, postdocs, and other research staff. Examine their academic background, publication record, and work/research experience (including internships).
   (h) Assess the economic impact of their research. Look at their technology transfer output: examine their patent portfolio, and determine the number of industrial collaborators and number of start-ups or university spin-offs that arise from its research.
   (i) Determine if there is a collaborative culture of excellence in the research lab. Do student lab members have a strong desire and will to work really, really hard? Do they take risks: to create new solutions for their class assignments/projects so that they can earn extra credit; to try new things – such as a new VLSI design flow, or learn a new EDA tool or HDL/programming language; or to create "commercializable" IC designs and EDA software)? Basically, are they really, really smart and ridiculously hardworking engineers, who subscribe to a compelling (for some of my engineering homies) and repulsive/odd (for most people, including engineers) subculture that is the stuff of legends?:
      i. References:

A. Benny Evangelista, "Facebook's Hacker Way – 'Move fast and break things'," in *The San Francisco Chronicle: The SF Gate: Blogs at SFGate: The Tech Chronicles*, Hearst Communications Inc., San Francisco, CA, February 1, 2012. Available online at: http://blog.sfgate.com/techchron/2012/02/01/facebooks-hacker-way-move-fast last accessed on April 18, 2013.
B. David Kushner, "Facebook Philosophy: Move Fast and Break Things – Hacker culture is alive and well at Facebook," in *IEEE Spectrum: At Work News & Articles: Innovation News & Articles*, IEEE Press, Piscataway, NJ, June, 2011. Available online at: http://spectrum.ieee.org/at-work/innovation/facebook-philosophy-move-fast-0; last accessed on April 18, 2013.
C. Rachel F. [/Fong???], "Move Fast And Break Things," in *MIT Admissions*, Admissions Office, Massachusetts Institute of Technology, Cambridge, MA, June 25, 2011. Available online at: http://mitadmissions.org/blogs/entry/move_fast_and_break_things; last accessed on April 18, 2013.
D. The Huffington Post, "Facebook IPO: Mark Zuckerberg On 'The Hacker Way'," in *The Huffington Post: Canadian Business and Financial news: Small Business News and Opinion: Money–Pictures, Videos, Breaking News*, The Huffington Post, New York, NY, February 1, 2012. Available online at: http://www.huffingtonpost.ca/2012/02/01/facebook-ipo-mark-zuckerberg-hacker-way_n_1248662.html; last accessed on April 18, 2013.

7. Determine if the professor encourages students to turn class projects into mini-research projects, which lead to conference papers or participation in research-based programming contests.
8. Does the department offer classes in research areas/topics that I can and want to take? Remember that some advance classes for grad students are offered only in a few universities, where students will be challenged with class projects, class assignments, and examinations (midterms and finals). Preferably, consider programs that have 3-8 classes in my research area. Else, I would have to learn that stuff (or at least the background knowledge) on your own. It can be hard to learn things on my own, especially if I do not know how to learn independently to a sufficient extent (especially with respect to knowledge relevant to that area/topic).
9. Does the academic program require/allow me to take a set of graduate classes spanning the breadth of the departments's academic field? If so, determine if I can do well in such classes. Do I have several options regarding these requirements?
10. Find out about the culture and subcultures of the on-campus community, student body (student culture is prominent in the U.S.). Your lifestyle choices and preferences would probably be a major factor in helping you decide where to go. You wanna ski or wind surf regularly? Find a campus near the mountains or the sea. Do you wanna golf or hike regularly? Do you need to hit the clubs/bars and party regularly? Do you need to shop regularly to destress? These will help you determine if you wanna stay in a big city, a small city, or a college town... Underrepresented minorities and GLBT members find it hard in many campuses around the world. Here, try to find information regarding the diversity and inclusiveness of the student body (i.e., campus diversity). Note that a diverse body does not guarantee that the campus is inclusive. It may have students from a wide range of economic and cultural backgrounds, but students may (strongly) prefer to socialize in self-segregated enclaves; see student self-segregation (or auto-segregation)... Facebook groups/pages (and those on comparable social media outlets) reflect aspects of the student culture, beyond the universities' web pages (including those for student organizations). "College Confidential" or equivalent can offer your some interesting tidbits or gossip that may expose things about the campus, including systemic failures of the universities (e.g., to address racism or sexual assault on campus), poor teaching, or poor administrative support.

11. Does the university have a good student support service staff? Does it provide good student support for international students? Are adequate support services provided to help me apply for the visa, relocate and settle down (e.g., help you find an accommodation or at least temporary lodging, create a bank account, and what not)?
12. Accommodation options. Is it hard to find decent and affordable housing in the surrounding area? Is housing in the surrounding area cheap? What are some of the statistics of the local population? How diverse is the community? LA Times has some statistics of local populations in LA. You can determine the average income, highest level of education, political views, and so on... Try to determine if there is an equivalent of that for the local area... In addition, noisy neighbors and terrible roommates can ruin your semester. Unfortunately, many universities do not provide enough decent and affordable housing options.
13. Financial cost of enrolling in the academic program
14. Preferences of dependent family members and/or significant other
15. Weather and physical environment... Some people with asthma may wanna avoid certain places with hay fever or very cold weather. People with disabilities may find it hard to navigate certain terrains or campuses/cities/towns.
16. Availability of food sources that are needed to meet strict dietary restrictions, if any... E.g., Kosher or Halah food requirements, food allergies, or vegan diets and their variants) or disabilities.
17. Availability of certain long-term support services. If you have a need for long-term counseling support or medical consultation (e.g., medical checkups for a chronic/recurrent medical problem), you need to determine if such services can be accessed at affordable costs.
18. Ease of access... Some cities/towns are hard to reach... It may take you hours to get to the nearest airport, which may be very small. Traveling to research conferences, job interviews, or getaway places (for vacations) could be painful experiences.
19. Is the university's administration associated with any systemic failure or controversial decisions? For example, some senior university administrators would raise tuition fees so that they can dramatically increase their salaries; the growth rate of their salaries (and hence, the tuition fees) would rise (much) faster than the nation's economic inflation rate. In addition, the university's administration make expand the university's campus or create new satellite campuses at a rate greater/faster than the university's sustainable rate. This can cause financial problems for the university, which may lead to the destruction of research programs and schools/departments (e.g., engineering school at Tulane University and the computer science department at the University of Florida)
    (a) E.g., Moody's Global Credit Research, which is part of Moody's Investors Service, Inc. (New York, NY), published a forward-looking report (in April 2013) about the U.S. higher education industry's financial problems: "Industry Outlook: US Higher Education Outlook Negative in 2013." This report indicates problems with the "business models" of "market-leading, research-driven colleges and universities." Their "diversified revenue streams" are not generating adequate "revenue growth." For "long-term financial sustainability" and to maintain/improve their abilities to "fund future initiatives," they need to "lower their cost structures." They face "prolonged muted revenue growth." With the exception of world-class/elite universities, colleges and "universities face diminished student demand and increased price sensitivity." Hence, they cannot keep raising tuition fees to raise revenue. Also, "all non-tuition revenue sources are also strained." That is, a number of colleges and universities are "on a borrowing spree" and are "struggling to stay afloat" financially. Most people do not care about the credit ratings (or bond ratings) of colleges and universities.

   (b) Reference: Jeffrey J. Selingo, "Colleges Struggling to Stay Afloat," in *The New York Times: [United States] National News: Education News: Education Life: Viewpoint | Balance Sheet*, The New York Times Company, New York, NY, April 12, 2013. Available online at: http://www.nytimes.com/2013/04/14/education/edlife/many-colleges-and-universities-face-fina html; last accessed on April 18, 2013.
      i. Will outdated buildings be renovated?
      ii. Will more part-time instructors replace retiring professors?
      iii. Will classes get bigger or will students end up in partly online courses?
      iv. Will the school even be in business by the time your child graduates in four years?
      v. Many colleges and universities face multi-faceted challenges due to "technological, demographic and economic" trends. Is this university one of those universities? Try to obtain a publicly accessible financial statement of the research university, and use it to find out about the financial health of the university.

20. Macro trends... Thing of local, regional, and global trends in economics, technology, and what not. Also, pay attention to local (e.g., city-/town-level), regional (at the state/provincial/territorial level or the continental level, such as the EU), and national policies (i.e., public policies). For example, many public universities in California (i.e., the University of California and California State University systems) had to force faculty and staff to take leave, and cut many classes in various campuses due to budget restrictions in recent years due to California's financial problems and the global financial crisis. Nobody wants to get messed up by politicians and policymakers.
21. Can I transfer credit from prior graduate-level classes?
22. Can I place out of intermediate classes, typically taken by underclassmen and junior grad students.
23. Reference: "A Need-to-Know List," New York Times, November 4, 2011. Available online at: http://www.nytimes.com/2011/11/06/education/edlife/a-need-to-know-list.html?ref=edlife; last accessed on November 7, 2011.:
   (a) On campus tours, science and engineering schools love to show off their high-tech equipment and promote how many graduates work for Google. But dont expect them to mention how tough they grade or how many students wash out. To cut through the hype, here are questions to ask each school to gauge where you have the best chance of succeeding:
      i. What percentage of science and engineering students stick with those majors through graduation?
      ii. What is the average grade for students in my major, and are there limits on how many can receive As and Bs?
      iii. How large are the freshman lectures? How much tutoring do you provide for students who need help to keep up?
      iv. And besides the superstars, how many undergraduates really get to help professors with research?

Grad school info:

- general information and advice about grad school:
   1. `Resources from Professional Organizations/Societies.`
   2. IEEE:
      (a) Susan Karlin, "How to Choose A Grad School: Figure out what you want and who can give it to you," *IEEE Spectrum*, September 2005. Available at: http://spectrum.ieee.org/at-work/education/how-to-choose-a-grad-school; last accessed on August 28, 2010.
      (b) IEEE Potentials, Volume 21, Issue 3, Aug/Sep 2002.
         i. A masters degree helps you gain advanced technical skills, while a Ph.D. is a lifestyle choice. (page 2/3)

   (c) IEEE Potentials, Volume 24, Issue 3, Aug/Sep 2005.

3. ACM:
   (a) ACM, *ACM Crossroads Student Resources*, ACM, New York, NY, Aug 17, 2005. Available at: http://oldwww.acm.org/crossroads/resources/; last accessed on August 29, 2010.
   (b) ACM, *Graduate Educational Resources from ACM Crossroads*, ACM, New York, NY, Aug 7, 2005. Available at: http://oldwww.acm.org/crossroads/resources/graduate.html; last accessed on August 29, 2010.
   (c) *ACM Crossroads* articles on grad school application and life in grad school: [1,3–6,15,21].

4. American Society for Engineering Education:
   (a) http://www.asee.org/
   (b) Prism: http://www.prism-magazine.org

5. American Psychological Association:
   (a) American Psychological Association, *gradPSYCH* [magazine], American Psychological Association, Washington, DC. Available at: http://www.apa.org/gradpsych/; last accessed on September 1, 2010. [Read issues from May 2003 till January 2005; **This is an excellence source of information about doing well in graduate school and internships, and seeking research careers in academia and the industry.**]

6. Computing Research Association (CRA):
   (a) Computing Community Consortium:
      i. Computer Science Research Opportunities and Graduate School: http://cra.org/ccc/csgs
      ii. CS URGE–Computer Science Undergraduate Research and Graduate Education: http://cra.org/ccc/csurge
   (b) Information for Undergraduate and Graduate Students: http://www.cra.org/for-students/
   (c) CRA Taulbee Survey:
      i. The Taulbee Survey is the principal source of information on the enrollment, production, and employment of Ph.D.s in computer science and computer engineering (CS & CE) and in providing salary and demographic data for faculty in CS & CE in North America.
      ii. Statistics given include gender and ethnicity breakdowns.
      iii. http://www.cra.org/statistics/
   (d) Computer Research Association's Committee on the Status of Women in Computing Research (CRA-W), "Graduate Student Information Guide". Available at: http://www.cra-w.org/sites/default/files/grad-guide.pdf; CRA-W ⇒ Resources ⇒ Publications ⇒ link and description to the guide, "Graduate Student Information Guide"; last accessed on September 3, 2010.
   (e) Computer Science Post-docs:
      i. http://cra.org/postdocs/
      ii. The Reading List: http://cra.org/postdocs/reading-list.php
   (f) The CRA Center for Evaluating the Research Pipeline (CERP): http://cra.org/cerp/

7. American Mathematical Society:
   (a) *Applying to Graduate School*. Available at: http://www.ams.org/profession/career-info/grad-school/grad-school; last accessed on September 2, 2010.

8. The Mathematical Association of America:
   (a) *MAA Students*. Available at: http://www.maa.org/students/; last accessed on September 2, 2010.

9. American Institute of Mathematics:

(a) Resources for the Math Community:
  i. http://www.aimath.org/mathcommunity/
  ii. David W. Farmer, "The AIM REU: individual projects with a common theme," in the *Proceedings of the Conference on Promoting Undergraduate Research in Mathematics*, American Mathematical Society, 2006. Available online at: http://www.aimath.org/mathcommunity/farmerREU.pdf; last accessed on January 9, 2010. ["AIM Research Experience for Undergraduates (REU)"]
  iii. Sally Koutsoliotas and David W. Farmer, "Preparing students to give talks," American Institute of Mathematics. Available online at: http://www.aimath.org/mathcommunity/studenttalks.pdf; last accessed on January 9, 2010. ["Preparing students to give talks"]

10. Institute for Operations Research and the Management Sciences (INFORMS):
  (a) *Career Center* (has some information on funding/fellowships, and academic careers): http://www.informs.org/Build-Your-Career/INFORMS-Student-Union/Career-Center
11. `Resources from Multi-National Organizations.`
12. European Commission:
  (a) Marie Curie Actions: http://ec.europa.eu/research/mariecurieactions/index.htm... **SCHOLARSHIPS!!!**
  (b) *EURAXESS* Research Job Vacancies: http://ec.europa.eu/euraxess/index.cfm/jobs/jvSearch or http://ec.europa.eu/euraxess/index_en.cfm?l1=13&l2=3&initSearch=1#... **SCHOLARSHIPS!!!**
  (c) European Commission, "Third European Report on Science & Technology Indicators 2003," Directorate-General for Research, European Commission. Available at: http://cordis.europa.eu/indicators/contacts.htm; last accessed on September 1, 2010. Report on European universities and research initiatives.
  (d) Publications by the Directorate-General for Research, European Commission, about research initiatives and output in Europe: http://cordis.europa.eu/indicators/publications.htm
13. `Resources from governmental organizations/agencies.`
14. U.S. Department of Education:
  (a) National Center for Education Statistics (NCES); Institute of Education Sciences:
    i. National Center for Education Statistics, *Projections of Education Statistics to 2018* [report], National Center for Education Statistics, Institute of Education Sciences, U.S. Department of Education, September 2009. Available online at: http://nces.ed.gov/programs/projections/projections2018/index.asp; last accessed on January 7, 2010.
    ii. Look at its report on, "The Condition of Education 20XY," where X and Y are integers.
  (b) College Affordability and Transparency Center: http://collegecost.ed.gov/catc/
15. `Resources from research universities.`
16. *Highly-ranked research universities.*
17. University of California, Berkeley:
  (a) Matthew Moskewicz, Parallel Computing Laboratory (Par Lab), Department of Electrical Engineering and Computer Sciences:
    i. Developed the *Chaff* SAT solver with Conor Madigan as undergrads that is 10-100X faster than then existing SAT solvers.
    ii. He is named a co-winner of the 2009 CAV Award, along with his co-developers of *Chaff* and the developers of the *GRASP* SAT solver, for his fundamental contribution to the field of Computer Aided Verification.

iii. http://www.princeton.edu/engineering/eqnews/spring01/feature5.html
iv. http://parlab.eecs.berkeley.edu/people/matthew-moskewicz

(b) Mark Borgschulte, "Economics Grad School Application Advice," Department of Economics, University of California, Berkeley. Available online at: http://sites.google.com/site/markborgschulte/economicsgradschoolapplicationadvice; last accessed on January 9, 2010.

(c) Mark Borgschulte, "Info for Berkeley Admits," Department of Economics, University of California, Berkeley. Available online at: http://sites.google.com/site/markborgschulte/infoforberkeleyadmits; last accessed on January 9, 2010.

(d) Mark Borgschulte, "Reading List," Department of Economics, University of California, Berkeley. Available online at: http://sites.google.com/site/markborgschulte/readinglis
last accessed on January 9, 2010.

(e) *Secret Blogging Seminar* is a blog written by recent Ph.D. graduates from Berkeley's Department of Mathematics. Noah Snyder, "Thoughts on graduate school," May 13, 2009. Available at: http://sbseminar.wordpress.com/2009/05/13/thoughts-on-graduate-school
last accessed on September 1, 2010.

(f) *UC Berkeley Career Center* [Online], "Graduate School - Letters of Recommendation," UC Berkeley. Available at: https://career.berkeley.edu/grad/gradletter.stm; last accessed on September 5, 2010.

(g) UC Berkeley Career Center [Online], *Graduate School*, in *Graduate & Professional School*, Division of Student Affairs, Office of the Vice Chancellor, University of California, Berkeley, October 28, 2010. Available online at: https://career.berkeley.edu/Grad/Grad.stm; last accessed on March 5, 2011.

(h) David Wagner, "EECS Undergraduate Notes 2011-2012," Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Berkeley, CA, 2011. Available online at: http://www.eecs.berkeley.edu/Programs/Notes/; last accessed on April 30, 2012:

i. "Although degree plans and goals tend to evolve as you proceed in your studies, it is very important that you start the process already in your first semester at Berkeley. For your degree you have a choice of hundreds of courses offered by the department, the college, and the university. You need to start looking at your options now to ensure that you follow the program that best fits you and your goals. Some of your choices may have profound impact on your career opportunities for years to come."

ii. "Learning happens not only in the classroom. The department offers a wide range of options to learn about the field, including undergraduate research opportunities and internships. Many of these are in high demand and often require appropriate preparation (e.g. taking specific courses ahead of time). Becoming an undergraduate teaching assistant is an excellent opportunity to deepen your understanding in core areas of engineering. The EECS Honors degree program gives additional flexibility in your program and the opportunity to select an academic concentration outside EECS."

iii. "Learning is not a passive activity. I invite you to challenge your creativity to put together a degree program that engages your talents and starts a fruitful career."

18. Carnegie Mellon University:

(a) James C. Hoe, "Advice Column for ECE Undergrads," Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, January 14, 2012. Available online at: http://www.ece.cmu.edu/~jhoe/doku/doku.php?id=advice_column; last accessed on March 1, 2012.

(b) Carnegie Mellon University, *Ph.D. in Computer Science*, Computer Science Department,

Carnegie Mellon University, Pittsburgh, PA. Available at: http://www.csd.cs.cmu.edu/education/phd/index.html; last accessed on August 28, 2010.

(c) Mark Leone, *Advice on Research and Writing*, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA. Available at: http://www-2.cs.cmu.edu/afs/cs.cmu.edu/user/mleone/web/how-to.html; last accessed on August 28, 2010. Also, see http://www.cs.cmu.edu/~mleone/how-to.html for another copy. [Mark Leone has graduated with a MS CS from CMU.]

(d) Jason I. Hong, *Grad School Advice*, Human Computer Interaction Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, Sept 20, 2006. Available online at: http://www.cs.cmu.edu/~jasonh/advice.html; last accessed on December 17, 2010.

(e) women@SCS School of Computer Science:

  i. Career Advice: http://women.cs.cmu.edu/Resources/JobsResearch/careeradvice.php

(f) Office of the Dean of Student Affairs, "Steps Students Might Take to Promote Academic Integrity," in *Academic Integrity: Promoting Academic Integrity*, Division of Student Affairs, Carnegie Mellon University, Pittsburgh, PA. Available online at: http://www.studentaffairs.cmu.edu/acad_int/promotestudent.html; last accessed on March 1, 2012.

19. University of California, San Diego:

(a) UCSD VLSI CAD Laboratory, *Useful tips on how to succeed in graduate school and your subsequent research career*, Department of Computer Science and Engineering & Department of Electrical and Computer Engineering, University of California, San Diego. Available at: http://vlsicad.ucsd.edu/Research/Advice/index.html; last accessed on August 28, 2010. **EXCELLENT!!!**

(b) Mihir Bellare, "The Ph.D Experience," Department of Computer Science & Engineering, University of California at San Diego. Available at: http://cseweb.ucsd.edu/~mihir/phd.html; last accessed on September 13, 2010. [See *Educational* material about graduate school, research, and technical writing at: http://cseweb.ucsd.edu/users/mihir/education.html.]

(c) Fan Chung Graham, "A few words on research for graduate students (especially for those potential combinatorialists)," in *Teaching*, Department of Mathematics, University of California, San Diego. Available online at: http://math.ucsd.edu/~fan/teach/gradpol.html; last accessed on January 9, 2010.

20. University of Michigan, Ann Arbor:

(a) Igor Markov, Department of Electrical Engineering and Computer Science: http://www.eecs.umich.edu/~imarkov/i_students.html. In particular, see his "Advice for graduate students".

(b) **Me: Does performing well in research-type programming contests help? For example, would doing well in the ISPD programming contest, SAT competition, SMT competition, and/or TAU Workshop programming contest help to indicate that there are good technical innovations in developing the SAT/SMT solver or EDA tool that enabled this team/individual to win? Prof. Igor Markov: Not necessarily. These contests are very complex technically, the number of competent teams is small (around 5-6 at ISPD, maybe double that at SAT). Organization quality varies widely. Some past winners and runners up mostly reimplemented known algorithms and were not able to convincingly explain (in their follow-up publications) what innovations they came**

up with. In some cases, the claimed innovations were downplayed in later publications from the same group. At ISPD, just a few weeks after the contest, the best results are often improved. In some years, several teams were caught cheating. At the SAT contest, there are many complaints about scoring. So, I would look at the combination of contest entries and subsequent publications. Reference: Igor Markov, Response to my comment on his answer to "Does a good profile in competition programming help in getting admission in top MS/Phd program at top US universities for Algorithms/Theoretical CS/Systems?", in *Does a good profile in competition programming help in getting admission in top MS/Phd program at top US universities for Algorithms/Theoretical CS/Systems?*, Quora, Inc., Palo Alto, CA, August 4, 2012. Available online at: http://www.quora.com/Competitive-Programming/Does-a-good-profile-in-competition-programming-help-in-getting-admission-in-top answer/Igor-Markov; last accessed on September 14, 2012.

21. University of California, Los Angeles:
    (a) Philip E. Agre, "Advice for Undergraduates Considering Graduate School," UCLA Department of Information Studies, University of California, Los Angeles, October 1996 (Modified: May 2001). Available at: http://polaris.gseis.ucla.edu/pagre/grad-school.html; last accessed on August 28, 2010. See http://polaris.gseis.ucla.edu/pagre/grad-school.pdf for a PDF copy of this article. **CLASSIC!!!**. See http://polaris.gseis.ucla.edu/pagre/index.html for more articles.
    (b) Terence Tao, *Career advice*, Department of Mathematics, University of California, Los Angeles. Available at: http://terrytao.wordpress.com/career-advice/; last accessed on September 1, 2010. Additional information can be found at: http://www.math.ucla.edu/~tao/.
    (c) Yu Hu:
        i. Yu Hu, "Links: Programming Tools and Tips; and Documentation and Presentation," Electrical Engineering Department, University of California, Los Angeles, Aug 27, 2007. Available online at: http://www.ee.ucla.edu/~hu/links.htm; last accessed on September 18, 2010. [This web page has resources for computer programming, and creating presentation slides and documentations.]
        ii. Courses @ UCLA, April 22, 2006: http://www.ee.ucla.edu/~hu/course.htm
        iii. Research: http://www.ee.ucla.edu/~hu/project.htm
22. Stanford University:
    (a) Eran Magen, *How I Got Into the Stanford Psychology Ph.D. Program*, Department of Psychology, Stanford University. Available at: http://www.howigotintostanford.com/; last accessed on September 1, 2010.
    (b) Jeffrey Michael Heer, Department of Computer Science:
        i. The only Ph.D. student to have ever won the Microsoft Graduate Fellowship and the IBM Ph.D. Fellowship concurrently. After he won these fellowships as a Ph.D. student at Berkeley, IBM changed the rules for its Ph.D. fellowship so that nobody else can do this anymore. This prevents other companies from competing with IBM for hiring these fellows as research interns.
        ii. http://hci.stanford.edu/jheer/cv/
    (c) John Ousterhout, "My Favorite Sayings," Department of Computer Science, Stanford University, September 09, 2009. Available at: http://www.stanford.edu/~ouster/cgi-bin/sayings.php; last accessed on September 4, 2010. [Also, see *Odds & Ends*: http://www.stanford.edu/~ouster/cgi-bin/misc.php]
    (d) Philip Guo, *Academic Home Page*, Department of Computer Science, Stanford University. Available at: http://www.stanford.edu/~pgbovine/academic.htm; last accessed

on September 1, 2010. [See resources at the bottom of the page. Also, see http://www.stanford.edu/~pgbovine/writings.htm for his non-academic/research articles.]

(e) Ravi Vakil, "For potential students," Department of Mathematics, Stanford University. Available at: http://math.stanford.edu/~vakil/potentialstudents.html; last accessed on September 1, 2010. ["Great articles and books" in mathematics: http://math.stanford.edu/~vakil/greatwriting.html. Information about getting/writing letters of recommendation: http://math.stanford.edu/~vakil/recommendations.html.]

(f) Stanford Computer Science Department, *Load Balancing*, Stanford Computer Science Department, Stanford School of Engineering, Stanford, CA, 2012. Available online at: https://cs.stanford.edu/orientation; last accessed on July 17, 2012.

  i. "As you probably already know, computer science classes are often much more rigorous and time-consuming than the average class at Stanford. With this in mind it is important to properly balance your quarterly class load so that you do not overextend yourself. As a general guideline, having two computer science classes in a given quarter will keep you pretty busy but won't crush you with an inordinate amount of work. Taking three computer science classes in one quarter is generally considered difficult and will probably not leave much room for extra-curricular activities. To even consider taking four computer science classes in one quarter is strongly discouraged. Let's face it, you need some time for eating, showering, and sleeping. To sum things up, see which of these attitudes best fits your lifestyle."
  ii. See Table 1.
  iii. "In all seriousness, overextending oneself can have far-reaching consequences as the quarter progresses. Typically, students who take more classes than they can handle end up doing poorly in all of their classes and may be forced to drop a class anyway. This is not meant to discourage anyone from being ambitious when planning a class schedule (there are students who have successfully taken four or more computer science classes in a given quarter), but be prepared for the amount of work required."

(g) Stanford Computer Science Department, *New Student Orientation*, Stanford Computer Science Department, Stanford School of Engineering, Stanford, CA, 2012. Available online at: https://cs.stanford.edu/orientation; last accessed on July 17, 2012.

(h) Stanford University, "Guidelines for Advising Relationships between Faculty Advisors and Graduate Students," Office of the Vice Provost for Graduate Education (VPGE), Stanford University, 2009. Available online at: http://vpge.stanford.edu/docs/Advisor_Guidelines.pdf; last accessed on December 22, 2010.

(i) Stanford University, *Tomorrow's Professor*^{SM} *Mailing List Links*, Center for Teaching and Learning, Stanford University. Available at: http://www.stanford.edu/dept/CTL/Tomprof/links.html; last accessed on September 1, 2010.

(j) Stanford University, *Tutoring and Academic Support*, [Office of the] Vice Provost for Undergraduate Education, Stanford University. Available at: http://ual.stanford.edu./ARS/index.html; last accessed on September 1, 2010.

23. University of Washington:

(a) University of Washington, *10-Year Review Self-Study*, Department of Computer Science & Engineering, University of Washington, January 2000. Available at: http://www.cs.washington.edu/homes/lazowska/selfstudy/; last accessed on September 2, 2010. See other information on Prof. Ed Lazowska's web page: http://www.cs.washington.edu/homes/lazowska/.

(b) Yuriy Brun, *Yuriy Brun's Advice*, Department of Computer Science & Engineering, University of Washington. Available at: http://www.cs.washington.edu/homes/brun/advice/; last accessed on August 28, 2010. See http://www.cs.washington.edu/homes/brun/advice/PhDAdvice.pdf for: Yuriy Brun, "Getting a Ph.D. at the University of Southern California," May 20, 2010.

(c) Michael Ernst, *Advice for researchers and students*, Department of Computer Science & Engineering, University of Washington. Available at: http://www.cs.washington.edu/homes/mernst/advice/; last accessed on August 28, 2010.

(d) Karin Strauss, *For graduate students* [see the links on the left side of her home page]. Available at: http://www.cs.washington.edu/homes/kstrauss/; last accessed on September 3, 2010.

(e) Wanda Pratt, *Advice*, Information School & Division of Biomedical & Health Informatics / Department of Medical Education and Biomedical Informatics / School of Medicine, University of Washington. Available at: http://faculty.washington.edu/wpratt/advice.htm; last accessed on September 3, 2010.

(f) William A. Stein, *Home Page*, Department of Mathematics, University of Washington. Available at: http://wstein.org/; last accessed on September 5, 2010. [**Has GREAT resources for junior faculty application and research grant proposals. He has provided tar balls (or zip files) and PDF files of these material.**]

(g) University of Washington Graduate School, *Re-envisioning the Ph.D. project*, University of Washington Graduate School, University of Washington. Available at: http://www.grad.washington.edu/envision/index.html; last accessed on August 28, 2010. [This research is about issues concerning Ph.D. programs, such as: how to improve the quality of Ph.D. programs and student outcomes, and the lifestyle (including social life) of Ph.D. students; and funding issues.]

(h) University of Washington, "Erroneous NRC Ranking Data for UW CSE," Department of Computer Science & Engineering, University of Washington, Seattle, WA, Fall 2010. Available online at: http://www.cs.washington.edu/nrc/; last accessed on August 22, 2012.

24. Duke University:
    (a) Xiaowei Yang, *Advice Collection*, Department of Computer Science, Duke University. Available at: http://www.cs.duke.edu/~xwy/advices.html; last accessed on August 28, 2010.

25. Columbia University:
    (a) Department of Economics:
        i. Donald R. Davis, "Ph.D. Thesis Research: Where do I Start?", Department of Economics, Columbia University, February 2001. Available online at: http://www.columbia.edu/~drd28/Thesis%20Research.pdf; last accessed on January 9, 2010.

26. Princeton University:
    (a) Boaz Barak, Department of Computer Science:
        i. He won the ACM Doctoral Dissertation Award.
        ii. http://awards.acm.org/doctoral_dissertation/
        iii. http://www.cs.princeton.edu/~boaz/

27. The University of Texas at Austin:
    (a) Department of Computer Science:
        i. Mike Dahlin, "Advice to systems researchers," Department of Computer Science, The University of Texas at Austin. Available online at: http://www.cs.utexas.edu/users/dahlin/advice.html; last accessed on January 9, 2010.
    (b) Department of Electrical & Computer Engineering:
        i. Adnan Aziz:

- **Lots of helpful resources**

ii. Brian L. Evans, "Suggestions for Current and Prospective Graduate Students," Department of Electrical and Computer Engineering, The University of Texas at Austin, January 20, 2011. Available online at: http://www.ece.utexas.edu/~bevans/suggested_courses.html; last accessed on March 7, 2011. **[AWESOME!!!]**

28. Massachusetts Institute of Technology:
    (a) MIT Computer Science and Artificial Intelligence Laboratory; :
        i. Jean Yang, *Resources*, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 2008. Available online at: http://people.csail.mit.edu/jeanyang/links.htm; last accessed on March 8, 2011.
    (b) MIT TechTV series on MIT MSRP 2012: https://techtv.mit.edu/collections/odge:2802
29. University of Pennsylvania:
    (a) Stephanie Weirich, *Advice for Graduate Studies*, Department of Computer and Information Science, School of Engineering and Applied Science, University of Pennsylvania. Available at: http://www.seas.upenn.edu/~sweirich/phdadvice.htm; last accessed on September 5, 2010.
30. University of Southern California:
    (a) Office of College Advising; USC Dana and David Dornsife College of Letters, Arts and Sciences:
        i. Pre-Graduate School Advising: http://dornsife.usc.edu/pre-graduate-school/
        ii. Shayna Kessel (former Pre-Graduate School Advisor)
        iii. Interfolio: http://www.interfolio.com/
    (b) Department of Computer Science; USC Andrew and Erna Viterbi School of Engineering:
        i. Jernej Barbič:
            - http://www-bcf.usc.edu/~jbarbic/
            - My name is pronounced like "Yerney Barbeach."
            - Guide on applying to grad school in US (in Slovenian): http://www-bcf.usc.edu/~jbarbic/vodic.html
            - Information about U.S. higher education (in Slovenian): http://www-bcf.usc.edu/~jbarbic/barbic-visoko-solstvo-2010.pdf
31. Northwestern University:
    (a) Department of Electrical Engineering and Computer Science; Robert R. McCormick School of Engineering and Applied Science:
        i. Lance Fortnow and William Gasarch, "Graduate Student Guide," in their blog *Computational Complexity*, Department of Electrical Engineering and Computer Science, Robert R. McCormick School of Engineering and Applied Science, Northwestern University, February 21, 2007. Available at: http://blog.computationalcomplexity.org/2007/02/graduate-student-guide.html; last accessed on September 14, 2010. [William Gasarch is from the Department of Computer Science at the University of Maryland, College Park. THIS IS EXCELLENT!!!]
32. Purdue University:
    (a) Douglas E. Comer, *A few essays about Computer Science*, Department of Computer Science, Purdue University:
        i. http://www.cs.purdue.edu/homes/dec/
        ii. Look for the section, "A few essays about Computer Science". The essay, "How to generate a CS research topic," is funny: http://www.cs.purdue.edu/homes/dec/essay.topic.generator.html.

iii. Douglas E. Comer, "Notes On The PhD Degree," Department of Computer Science, Purdue University. Available at: http://www.cs.purdue.edu/homes/dec/essay.phd.html; last accessed on September 12, 2010. [Also, see http://www.cs.virginia.edu/~jwd/comer-phd.htm.]
iv. Douglas E. Comer, "How To Write A Dissertation or Bedtime Reading For People Who Do Not Have Time To Sleep," Department of Computer Science, Purdue University. Available online at: http://www.cs.purdue.edu/homes/dec/essay.dissertation.html; last accessed on March 5, 2011.

(b) Jan Vitek, *Home Page: Miscellaneous*, Department of Computer Science, Purdue University. Available online at: http://www.cs.purdue.edu/homes/jv/; last accessed on September 28, 2010.

33. Cornell University:
(a) CU-ADVANCE Center (research and resource center concerning diversity and gender equity): http://advance.cornell.edu/
(b) Department of Computer Science, Faculty of Computing and Information Science (CIS):
i. Charles F. Van Loan: http://www.cs.cornell.edu/cv/default.htm

34. University of Minnesota Twin Cities:
(a) Department of Electrical and Computer Engineering; College of Science and Engineering (CSE):
i. Sachin Sapatnekar, "A simple (incomplete) checklist for writing papers," Department of Electrical and Computer Engineering, College of Science and Engineering, University of Minnesota Twin Cities. Available online at: http://www.ece.umn.edu/~sachin/misc/writing.html; last accessed on March 5, 2011.
ii. Sachin Sapatnekar, "Thinking of research in CAD?", Department of Electrical and Computer Engineering, College of Science and Engineering, University of Minnesota Twin Cities. Available online at: http://www.ece.umn.edu/users/sachin/research.html; last accessed on March 5, 2011.

35. Rice University:
(a) Richard G. Baraniuk, "Seven Steps to Success in Graduate School (and Beyond)," Department of Electrical and Computer Engineering, George R. Brown School of Engineering, Rice University. Available online at: http://www.ece.rice.edu/~richb/success.html; last accessed on January 9, 2010.

36. Yale University:
(a) Stephen C. Stearns, "Some Modest Advice for Graduate Students," Department of Ecology and Evolutionary Biology, Yale University. Available at: http://www.yale.edu/eeb/stearns/advice.htm; last accessed on August 28, 2010.
(b) Stephen C. Stearns, "Designs for Learning," Department of Ecology and Evolutionary Biology, Yale University. Available at: http://www.yale.edu/eeb/stearns/designs.htm; last accessed on August 28, 2010.

37. Harvard University:
(a) H. T. Kung, "Useful Things to Know About Ph. D. Thesis Research," Harvard School of Engineering and Applied Sciences, Harvard University. (Prepared for "What is Research" Immigration Course, Computer Science Department, Carnegie Mellon University, 14 October 1987). Available online at: http://www.eecs.harvard.edu/~htk/thesis.htm; last accessed on August 26, 2014.
(b) Susan Athey, "Advice for Applying to Grad School in Economics," Department of Economics, Harvard University. Available online at: http://kuznets.fas.harvard.edu/

~athey/gradadv.html; last accessed on January 9, 2010. Prof. Athey has also provided an article, "Negotiating Senior Job Offers," on her web page; this would concern senior faculty job offers.
   (c) The Collaborative on Academic Careers in Higher Education, Harvard University Graduate School of Education: http://isites.harvard.edu/icb/icb.do?keyword=coache&tabgroupid=icb.tabgroup104863

38. University of Wisconsin-Madison:
   (a) Department of Electrical and Computer Engineering; College of Engineering:
      i. Azadeh Davoodi: http://homepages.cae.wisc.edu/~adavoodi/Links.htm
   (b) Office of Graduate Studies and Professional Development:
      i. Professional Development: http://www.cals.wisc.edu/gradstudies/profdev/
   (c) Dorothea Salo, *A Tale of Graduate School Burnout*. Available at: http://members.terracom.net/~dorothea/gradsch/index.html; last accessed on August 28, 2010.
   (d) Dorothea Salo, *Straight Talk about Graduate School*. Available at: http://members.terracom.net/~dorothea/gradsch/straighttalk.html; last accessed on August 28, 2010.
   (e) Dorothea Salo, *What to do before applying to graduate school*. Available at: http://members.terracom.net/~dorothea/gradsch/success.html; last accessed on August 28, 2010.

39. *Not-so-highly-ranked research universities.*

40. Pennsylvania State University:
   (a) Tao Xie and Yuan Xie, *Advice Collection*, Department of Computer Science at North Carolina State University, and Department of Computer Science and Engineering at Pennsylvania State University. Available at: http://www.cse.psu.edu/~yuanxie/advice.htm; last accessed on August 25, 2010. Also, see http://people.engr.ncsu.edu/txie/advice/index.html and http://people.engr.ncsu.edu/txie/advice.htm.
   (b) Office of Engineering Diversity; Penn State College of Engineering:
      i. Office of Engineering Diversity, "Tips for Graduate Students," Penn State College of Engineering, Pennsylvania State University, 2009. Available online at: http://www.engr.psu.edu/mep/tips.html; last accessed on December 9, 2010.:
         – Getting the most out of the relationship with your research advisor or boss:
            ∗ **Meet regularly** - you should insist on meeting once a week or at least every other week because it gives you motivation to make regular progress and it keeps your advisor aware of your work.
            ∗ **Prepare for your meetings** - come to each meeting with: List of topics to discuss; Plan for what you hope to get out of the meeting; Summary of you have done since your last meeting; List of any upcoming deadlines; & Notes from your previous meeting
            ∗ **Email him/her a brief summary of EVERY meeting** - this helps avoid misunderstandings and provides a great record of your research progress. Include (where applicable): Time and plan for next meeting; New summary of what you think you are doing; To do list for yourself; To do list for your advisor; List of related work to read; List of major topics discussed; List of what you agreed on; & List of advice that you may not follow
            ∗ **Show your advisor the results of your work as soon as possible** - this will help your advisor understand your research and identify potential points of conflict early in the process. Summaries of related work. Anything you write about your research. Experimental results.

    * **Communicate clearly** - if you disagree with your advisor, state your objections or concerns clearly and calmly. If you feel something about your relationship is not working well, discuss it with him or her. Whenever possible, suggest steps they could take to address your concerns.
    * **Take the initiative** - you do not need to clear every activity with your advisor. He/she has a lot of work to do too. You must be responsible for your own research ideas and progress.
  - Getting the most out of what you read:
    * **Be organized**. Keep an electronic bibliography with notes & pointers to the paper files. Keep and file all the papers you have read or skimmed.
    * **Be efficient** - only read what you need to. Start by reading only the conclusion, scanning figures & tables, and looking at their references. Read the other sections only if the paper seems relevant or you think it may help you get a different perspective. Skip the sections that you already understand (often the background and motivation sections).
    * **Take notes on every paper you find worth reading** - What problem are they trying to solve? What is their approach? How is it different from other approaches?
    * **Summarize what you have read on each topic** – after you have read several papers covering some topic, note the: key problems; various formulations of the problem they are addressing; relationship among the various approaches; and alternative approaches
    * **Read Ph.D. theses** – even though they are long they can be very helpful in quickly learning about what has been done is some field. Especially focus on: Background sections; Method sections; and Your advisor's thesis (this will give you an idea for what he/she expects from you).
  - Making continual progress on your research:
    * **Keep a journal of your ideas** - write down everything you are thinking about even if you think it is stupid. It will help you keep track of your progress and keep you from going in circles. Do not plan to share it with anyone, so you can write freely.
    * **Set some reasonable goals with deadlines**: Identify key tasks that need to be completed; Set a reasonable date for completing them (on the order of weeks or months); Share this with your advisor or enlist your advisors help in creating the goals and deadlines; and Set some deadlines that you must keep (e.g., volunteer to give a student seminar on your research, work toward a conference paper submission deadline, etc.)
    * **Keep a to do list** - Checking off things on a to do list can feel very rewarding when you are working on a long-term project. List the small tasks that can be done in about an hour. Pick at least one that has to be completed each day.
    * **Continually update your: Problem statement**, Goals, Approach (or a list of possible approaches); One-minute version of your research (aka the elevator ride summary); and Five-minute version of your research
    * **Discuss your research with anyone who will listen** - use your fellow students, friends, family, etc. to practice discussing your research on various levels. They may have useful insights or you may find that verbalizing your ideas clarifies them for yourself.
    * **Write about your work**. Early stage: Write short idea papers and share

    them with your advisor and colleagues. Intermediate stage: Find workshops and conferences for submitting preliminary results. This can also help you set deadlines. Advanced stage: Target relevant journals.
  * **Avoid distractions** - it is easy to ignore your research in favor of more structured tasks such as taking classes, teaching classes, organizing student activities, creating web pages like this, etc. Minimize these kinds of activities or commitments.
  * **Confront your fears and weaknesses**: If you are afraid of public speaking, volunteer to give lots of talks. If you are afraid your ideas are stupid, discuss them with someone. If you are afraid of writing, write something about your research every day. One-minute version of your research (aka the elevator ride summary). Five-minute version of your research. List of advice that you may not follow.
  * **Balance reading, thinking, writing and hacking** - often research needs to be an iterative process across all of those tasks.
  * **Finding a thesis topic or formulating a research plan**. Pick something you find interesting - if you work on something solely because your advisor wants you to, it will be difficult to stay motivated ... Pick something your advisor finds interesting - if your advisor doesn't find it interesting he/she is unlikely to devote much time to your research. He/she will be even more motivated to help you if your project is on their critical path (although this has down sides too!)... Pick something the research community will find interesting - if you want to make yourself marketable... Make sure it addresses a real problem. Remember that your topic will evolve as you work on it. Pick something that is narrow enough that it can be done in a reasonable time frame. Have realistic expectations (i.e. Don't expect the Nobel Prize). Don't worry that you will be stuck in this area for the rest of your career. It is very likely that you will be doing very different research after you graduate.
- Characteristics to look for in a good advisor, mentor, boss, or committee member:
  * It is unreasonable to expect one person to have all of the qualities you desire. You should choose thesis committee members who are strong in the areas where your advisor is weak.
  * **Willing to meet with you regularly (about 1 hour every week or every other week). You can trust him/her to**: Give you credit for the work you do; Defend your work when you are not around; Tell you when your work is or is not good enough; Help you graduate in a reasonable time frame; and Look out for you professionally and personally
  * **Is interested in your topic. Has good personal and communication skills**: You can talk freely and easily about research ideas; Tells you when you are doing something stupid; Patient, Never feels threatened by your capabilities; Helps motivate you and keep you unstuck
- **Has good technical skills**. Can provide constructive criticism of papers you write or talks you give. Knows if what you are doing is good enough for a good thesis. Can help you figure out what you are not doing well; Can help you improve your skills; Can suggest related articles to read or people to talk to; Can tell you or help you discover if what you are doing has already been done; Can help you set and obtain reasonable goals. Will be around until you finish. Is well respected

in his/her field. Has good connections for the type of job you would want when you graduate; Willing and able to provide financial and computing support.
- **Avoiding the research blues**:
  - When you meet your goals, reward yourself.
  - Don't compare yourself to senior researchers who have many more years of work and publications.
  - Don't be afraid to leave part of your research problem for future work.
  - Exercise.
  - Use the student counseling services
  - Occasionally, do something fun without feeling guilty!

41. Technion - Israel Institute of Technology:
    (a) Department of Computer Science:
        i.
42. Ecole Polytechnique Fédérale de Lausanne (Swiss Federal Institute of Technology in Lausanne):
    (a) School of Computer and Communication Sciences:
        i. Philipp Haller:
           - http://lamp.epfl.ch/~phaller/
           - He seems to be pursuing a postdoc position simultaneously at EPFL and Stanford. Ask him how to do this.
           - As a postdoc, he was also a lecturer at a European-style summer school.
43. Johns Hopkins University:
    (a) Adam Ruben, Department of Biological Chemistry, School of Medicine, Johns Hopkins University. Adam Ruben, "Surviving Your Stupid, Stupid Decision to Go to Grad School," Broadway Books, New York, NY, 2010.
44. *Average research universities.*
45. University of California, Santa Cruz:
    (a) Department of Computer Science; Jack Baskin School of Engineering:
        i. CMPS/CMPE 200 - Research and Teaching in Computer Science and Engineering (and Applied Math and Statistics and Technology and Information Management) (Fall 2010) by Prof. Cormac Flanagan and Prof. Jose Renau (& Prof. Alexandre Brandwajn???). Available online at: http://www.soe.ucsc.edu/classes/cmps200/Fall10/; last accessed on December 17, 2010. [Also, see *CMPS 200 - Research and Teaching in Computer Science and Engineering*: http://www.cs.ucsc.edu/courses/course?cmps200]
46. The University of North Carolina at Chapel Hill:
    (a) Ronald T. Azuma, "So long, and thanks for the Ph.D.!" a.k.a. "Everything I wanted to know about C.S. graduate school at the beginning but didn't learn until later." The 4th guide in the Hitchhiker's guide trilogy (and if that doesn't make sense, you obviously have not read Douglas Adams, v. 1.08, Department of Computer Science, The University of North Carolina at Chapel Hill, January 2003. Available at: http://www.cs.unc.edu/~azuma/hitch4.html; last accessed on September 3, 2010. [Also, see *Guides to surviving Computer Science graduate school* at: http://www.cs.unc.edu/~azuma/azuma_guides.html.]
47. University of British Columbia:
    (a) UBC Faculty of Graduate Studies, *The Graduate Game Plan*, UBC Faculty of Graduate Studies, University of British Columbia. Available at: http://www.grad.ubc.ca/current-students/gps-graduate-pathways-success/graduate-game-plan; last accessed on August 28, 2010.

(b) UBC Faculty of Graduate Studies, *Resources for Achieving Success*, UBC Faculty of Graduate Studies, University of British Columbia. Available at: http://www.grad.ubc.ca/current-students/gps-graduate-pathways-success/resources-achieving-success; last accessed on August 28, 2010.
(c) UBC Faculty of Graduate Studies, *Research on the Lived Experience of Graduate Students*, UBC Faculty of Graduate Studies, University of British Columbia. Available at: http://www.grad.ubc.ca/current-students/gps-graduate-pathways-success/research-l
last accessed on August 28, 2010.
(d) UBC Faculty of Graduate Studies, *Resources for Graduate Student Career Development*, UBC Faculty of Graduate Studies, University of British Columbia. Available at: http://www.grad.ubc.ca/current-students/gps-graduate-pathways-success/resources-graduat
last accessed on August 28, 2010.
(e) UBC Faculty of Graduate Studies, *Graduate Guides*, UBC Faculty of Graduate Studies, University of British Columbia. Available at: http://www.grad.ubc.ca/current-students/gps-graduate-pathways-success/graduate-guides; last accessed on August 28, 2010.
(f) UBC Faculty of Graduate Studies, *Present and Publish Your Research*, UBC Faculty of Graduate Studies, University of British Columbia. Available at: http://www.grad.ubc.ca/current-students/gps-graduate-pathways-success/present-publish-your-research; last accessed on August 28, 2010.

48. Oxford University:
    (a) Computing Laboratory (Computer Science department):
        i. Computing Laboratory, "D.Phil. in Computer Science," Oxford University. Available online at: http://www.comlab.ox.ac.uk/admissions/dphil/transfer.html; last accessed on October 29, 2010.
        ii. Marta Kwiatkowska, "How to apply for a Doctorate in the Computing Laboratory," Computing Laboratory, MT 2009. Available online at: http://www.comlab.ox.ac.uk/admissions/dphil/howtoapply2009.pdf; last accessed on December 17, 2010.
            – A B.S. or B.A. gives you general education; a Masters is your license to practice; and a Ph.D. is a license to teach, do research, and examine people for their Ph.D. defense.
            – A Ph.D. "is a research degree," and an "apprentice in research". It is "awarded for a significant and substantial piece of research," and "examined by experts and defended in a viva."
            – Research is about finding out about something, and discovering how to do that. It involves taking responsibility for organizing my time and taking charge of the investigation.
            – Doing a Ph.D. is exciting as I "carry out investigations into [the] unknown," and it is "enriching to learn and master new techniques" while doing so.

49. University of California, Irvine; Donald Bren School of Information and Computer Sciences:
    (a) *UCI on iTunes U*, "Improving your Grad School Application," University of California, Irvine: Donald Bren School of Information and Computer Sciences: Bren School Honors Seminar on November 12, 2008. Available at: http://deimos3.apple.com/WebObjects/Core.woa/Browse/uci.edu.1983660442.01983660444.1975757832?i=204636387
last accessed on August 28, 2010. Also, see http://www.oit.uci.edu/itunesu/ for *UCI on iTunes U*, and http://www.ics.uci.edu/about/videos/index.php for *Bren School iTunes U* content. [I can access this from the main iTunes site as follows: Look at the "Find Educational Provider" tab on the left panel (it's in the middle), and select "Universities andColleges" $\Longrightarrow$ Select "UC Irvine" $\Longrightarrow$ Under the Courses panel, select

"Donald Bren School of Information and Computer Sciences" $\Longrightarrow$ Under the "Community Outreach" panel, select "Improving your grad school application" $\Longrightarrow$ watch this video. The video clip is about a panel discussion of professors about how to get into a top-tier graduate program in CS. It talks about things that admission officers look for, how to get strong letters of recommendation.]

(b) Michael Franz, "[Information and Advice for] Prospective Graduate Student[s]," Department of Computer Science, Donald Bren School of Information and Computer Sciences + Department of Electrical Engineering and Computer Science, Henry Samueli School of Engineering; October 23, 2003. Available online at: http://www.ics.uci.edu/~franz/Site/prospectivestudents.html; last accessed on February 11, 2010. **Good article. It also addresses a primary reason of why many non-US residents choose to go to the United States to pursue an advanced degree in science and engineering–immigration to the United States via the H-1B visa (and the green card) as high-skilled employees. In addition, it deals with non-US residents who lie about their credentials and how good their alma maters are.**

50. University of California, Davis:
    (a) Biswanath Mukherjee, "How to be a good graduate student," in *iTunes U: Universities & Colleges: University of California, Davis: Engineering*, Department of Computer Science, University of California, Davis. Available online at: http://itunes.apple.com/us/podcast/how-to-be-good-graduate-student/id389260866?i=86454453; last accessed on March 7, 2011.
    (b) Galois Group, Department of Mathematics:
        i. University of California, Davis, *Useful things to know when starting graduate school... ...as contributed by experienced grad students!*, Department of Mathematics, University of California, Davis. Available at: http://galois.math.ucdavis.edu/UsefulGradInfo/HelpfulAdvice/WishIdKnown; last accessed on September 1, 2010.
        ii. University of California, Davis, *LaTeX Tutorial*, Department of Mathematics, University of California, Davis. Available at: http://galois.math.ucdavis.edu/UsefulGradInfo/GettingStarted/LaTeXTutorial; last accessed on September 1, 2010. [Has LaTeX template for research proposal that is required for the Ph.D. qualifying exam.]
        iii. University of California, Davis, *Writing Your Doctoral Thesis*, Department of Mathematics, University of California, Davis. Available at: http://galois.math.ucdavis.edu/UsefulGradInfo/HelpfulAdvice/WritingYourThesis; last accessed on September 1, 2010. [Has LaTeX template for Ph.D. dissertations.]
        iv. "If you are a UC Davis Math Grad Student, then you are a member of the Galois Group."

51. University of Chicago:
    (a) Pedro F. Felzenszwalb, Department of Computer Science:
        i. http://people.cs.uchicago.edu/~pff/
        ii. His paper, "Digipaper: A Versatile Color Document Image Representation," has been cited 24 times in about 11 years since publication (as of September 1, 2010). He was an undergraduate then, and probably did this work as a junior or early in his senior year.
        iii. His paper, "Efficient Matching of Pictorial Structures," is probably based on his work done as a senior. As of September 1, 2010, this paper as been cited 222 times in about 10 years. From http://cs.uchicago.edu/, it states the following in its news section on September 1, 2010. "Pedro Felzenszwalb receives Longuet-Higgins prize. The 2010 Longuet-Higgins award has been given to Pedro Felzenszwalb and Daniel Huttenlocher, for their paper "Efficient Matching of Pictorial Structures",

Conference on Computer Vision and Pattern Recognition 2000. This award goes to a paper from 10 years ago that has made a fundamental impact on computer vision. Congratulations, Pedro!"

52. University of Virginia:
    (a) David Evans, *Advice*, Department of Computer Science, University of Virginia. Available at: http://www.cs.virginia.edu/~evans/advice/; last accessed on September 2, 2010. Also, see "advice for prospective research students": http://www.cs.virginia.edu/~evans/advice/prospective.html.
53. University of Maryland, Baltimore County:
    (a) Marie desJardins, Department of Computer Science and Electrical Engineering:
        i. http://www.cs.umbc.edu/~mariedj/
        ii. Has information on "How to Succeed in Graduate School," "how to organize a workshop," and "Presenting your research: Papers, talks and chats".
        iii. E.g., Marie desJardins, "How to Succeed in Graduate School," Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County: http://www.cs.umbc.edu/~mariedj/papers/advice-summary.html
54. University of California, Riverside:
    (a) Department of Computer Science and Engineering; Bourns College of Engineering:
        i. Prof. Philip Brisk:
            – Philip Brisk, "Prospective Students," in his personal web page *Philip Brisk: Prospective Students*. Available online at: http://www1.cs.ucr.edu/faculty/philip/prospective_students.html; last accessed on July 30, 2011. [This is excellent!!!]
55. University of Twente:
    (a) Computer Architecture for Embedded Systems (CAES) group; Faculty of Electrical Engineering, Mathematics and Computer Science:
        i. Sabih Gerez, *Hints for Literature Research*, Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, August 24, 2008. Available online at: http://wwwhome.ewi.utwente.nl/~gerezsh/hints/literature.html; last accessed on March 7, 2011.
56. University of Maryland, College Park:
    (a) Dianne Prost O'Leary, *Graduate Study in the Computer and Mathematical Sciences: A Survival Guide*, Department of Computer Science, University of Maryland, College Park. Available at: http://www.cs.umd.edu/~oleary/gradstudy/gradstudy.html; last accessed on August 28, 2010. It is also available at: http://www.cs.umd.edu/~oleary/gradstudy/. See http://www.cs.umd.edu/~oleary/ for more articles about "the accessibility of computer science," "8 rules for career success," and the disparity in gender ratios in STEM fields.
    (b) Arunchandar Vasan, "Tips on Writing a Statement of Purpose," Department of Computer Science, College of Computer, Mathematical and Natural Sciences, University of Maryland, College Park, College Park, MD. Available online at: http://www.cs.umd.edu/Grad/sop.shtml; last accessed on August 31, 2012. [**AWESOME resource**]
57. University of Utah:
    (a) Matt Might, *The illustrated guide to a Ph.D.*, School of Computing, University of Utah. Available at: http://matt.might.net/articles/phd-school-in-pictures/; last accessed on September 13, 2010. [See *blog.might.net* for more articles about graduate school: http://matt.might.net/articles/; e.g., read "10 easy ways to fail a Ph.D." at: http://matt.might.net/articles/ways-to-fail-a-phd/. His web page, http://matt.might.net/, has a sample of these articles. THIS IS EXCELLENT!!!]

(b) Department of Electrical and Computer Engineering:
   i. Prof. Cynthia Furse:
      – http://www.ece.utah.edu/~cfurse/
      – Cynthia Furse, *Graduate Student Survival 101*, February 2009. Available online at: http://www.ece.utah.edu/~cfurse/Tutorials/UU%20Thesis/How%20to%20Write%20a%20Thesis.htm; last accessed on December 10, 2010.
      – Cynthia Furse, *Dr. Furse's OnLine Tutorials*, August 2007. Available online at: http://www.ece.utah.edu/~cfurse/Tutorials/tutorialsUofU.htm; last accessed on December 10, 2010. [Has a lot of good resources for teaching/lecturing, academic/technical writing (including thesis writing), advice for grad/Ph.D. students, making presentations and giving talks, writing grants and proposals, entrepreneurship, and resources for job hunting.]

58. Indiana University:
   (a) Indiana University, *What Every New Grad Student Should Know*, School of Informatics and Computing, Indiana University. Available at: http://www.cs.indiana.edu/docproject/grad.stuff.html; last accessed on September 1, 2010.
   (b) David Chapman (Editor), *How to do Research At the MIT AI Lab*, AI Working Paper 316, MIT AI Lab, Massachusetts Institute of Technology, October, 1988. Available at: http://www.cs.indiana.edu/docproject/grad.stuff.html; last accessed on September 1, 2010.
   (c) Marie desJardins, *How to Be a Good Graduate Student*, SRI International (formerly Stanford Research Institute), March 1994. Available at: http://www.cs.indiana.edu/docproject/grad.stuff.html; last accessed on September 1, 2010.

59. The University of Arizona:
   (a) Jonathan Sprinkle, *Students: So, you want to be my student*, Department of Electrical and Computer Engineering, The University of Arizona. Available at: http://www2.engr.arizona.edu/~sprinkjm/Main/Students; last accessed on September 5, 2010. "Choose 2-3 IEEE or AIAA journal or conference publications from my website that interest you. Do not choose technical reports, or student papers. Write a critical review of the papers, including why the work is interesting, but most importantly where you think the work should go next. In this review, you are proving to me that you understand the purpose of research, and most importantly that you understand the technical details of the paper and how they relate to research."

60. Dartmouth College:
   (a) Mark L. Tomforde, "I've passed my quals, now what? - A guide for Ph.D. candidates in mathematics at Dartmouth College," Department of Mathematics, Dartmouth College, August 5, 2002. Available online at: http://www.math.dartmouth.edu/graduate-students/current/guide/GradGuide.pdf; last accessed on December 22, 2010. [Also, available at: http://www.math.dartmouth.edu/graduate-students/current/guide/]

61. Vienna University of Technology (TU Vienna):
   (a) Silvia Miksch, *Tips: How to Do Research*, Faculty of Informatics, Vienna University of Technology. Available at: http://www.ifs.tuwien.ac.at/~silvia/research-tips/; last accessed on September 1, 2010. It has plenty of resources about:
      i. "How to Do Research"
      ii. "How to Write a Scientific Paper"
      iii. "How to Design a Poster"
      iv. "Tips on Organizing Conferences, Workshops, and Symposia"
      v. "How to Review"

vi. "Digitial Libaries"
vii. "Tips for Writing Correct English"

62. Tufts University:
    (a) Norman Ramsey, *Resources for Students*, Department of Computer Science, Tufts University. Available at: http://www.cs.tufts.edu/~nr/students/; last accessed on September 2, 2010.
    (b) Norman Ramsey, *How to get admitted to a PhD program*, Department of Computer Science, Tufts University. Available at: http://www.cs.tufts.edu/~nr/students/admit.html; last accessed on September 2, 2010.
63. Portland State University:
    (a) Department of Computer Science; Maseeh College of Engineering and Computer Science:
        i. PSU CS 569 (MS Students) and CS 669 (PhD students) - Scholarship Skills (Fall 2010) by Prof. Andrew Black and Tim Sheard. Available online at: http://web.cecs.pdx.edu/~black/ScholarshipSkills/; last accessed on September 29, 2010.
64. State University of New York at Buffalo:
    (a) William J. Rapaport, *Information for Grad Students in Computer Science & Engineering at UB*, Department of Computer Science and Engineering, Department of Philosophy, and Center for Cognitive Science, State University of New York at Buffalo, Buffalo, NY. Available at: http://www.cse.buffalo.edu/~rapaport/GRAD/; last accessed on September 2, 2010.
65. Arizona State University:
    (a) **Center for Measuring University Performance** [has an awesome ranking of the top 50 U.S. research universities]: http://mup.asu.edu/index.html and http://mup.asu.edu/research.html
66. The Ohio State University:
    (a) The Ohio Science and Engineering Alliance:
        i. Graduate School: http://www.ohiosea.org/academic/academic_grad.html
67. University of New South Wales:
    (a) School of Physics:
        i. Joe Wolfe, *Educational Pages*, School of Physics, University of New South Wales. Available online at: http://www.phys.unsw.edu.au/~jw/education.html; last accessed on March 5, 2011.
        ii. Joe Wolfe, "How to Write a PhD Thesis," School of Physics, University of New South Wales, June 2, 2006. Available online at: http://www.phys.unsw.edu.au/~jw/thesis.html; last accessed on March 5, 2011.
68. Case Western Reserve University:
    (a) VLSI System Architecture Laboratory; Department of Electrical Engineering and Computer Science; Case School of Engineering:
        i. Xinmiao Zhang, *Home page: For students*, VLSI System Architecture Laboratory, Department of Electrical Engineering and Computer Science, Case School of Engineering, Case Western Reserve University, Cleveland, OH. Available online at: http://engr.case.edu/zhang_xinmiao/Resource.html; last accessed on July 4, 2012. [Resources on academic writing, and preparing for and making presentations.]
69. Swarthmore College:
    (a) Department of History:

i. Timothy Burke, "Should You Go to Graduate School?," in his blog *Easily Distracted: Culture, Politics, Academia and Other Shiny Objects*, Department of History, Swarthmore College. Available at: http://weblogs.swarthmore.edu/burke/permanent-features-advice-on-academia/features/; last accessed on September 14, 2010. [Also, see http://www.swarthmore.edu/SocSci/tburke1/gradschool.html.]

ii. Timothy Burke, "From ABD to the Job Market: Advice for the Grad School Endgame," in his blog *Easily Distracted: Culture, Politics, Academia and Other Shiny Objects*, Department of History, Swarthmore College. Available at: http://weblogs.swarthmore.edu/burke/permanent-features-advice-on-academia/abd/; last accessed on September 14, 2010.

70. Georgetown University:
   (a) Department of Economics:
      i. Garance Genicot, "Applying to Grad School in Economics," Department of Economics, Georgetown University. Available online at: http://www9.georgetown.edu/faculty/gg58/GradSchool.html; last accessed on January 9, 2010.

71. Norwegian University of Science and Technology:
   (a) "PhD Studies," Department of Computer and Information Science; Faculty of Information Technology, Mathematics and Electrical Engineering. Available online at: http://www.idi.ntnu.no/research/phd.php; last accessed on September 29, 2010. [Includes an outline of a Ph.D. research proposal that is required for applications to its Ph.D. program in computer science.]

72. North Carolina State University:
   (a) Richard M. Felder, "An Engineering Student Survival Guide," Department of Chemical and Biomolecular Engineering, North Carolina State University, 1993. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Papers/survivalguide.htm; last accessed on August 27, 2010.
   (b) Matthias F. (Matt) Stallmann, "What CSC Graduates Should Know," Department of Computer Science, North Carolina State University, February 9, 1996. Available online at: http://people.engr.ncsu.edu/mfms/Teaching/what-grads-should-know.html; last accessed on October 6, 2010.
   (c) Paul D. Franzon, *Advice*, Department of Electrical and Computer Engineering, North Carolina State University, September, 2009. Available online at: http://www.ece.ncsu.edu/erl/faculty/paulfwww/advice.html; last accessed on February 10, 2010. [Has advice for undergraduates, prospective graduate students, current graduate students, recent graduates, and information/advice on Prelims and Quals.]

73. Michigan State University:
   (a) The Graduate School:
      i. The Graduate School, *"Setting Expectations and Resolving Conflict" Program: Developing Communication and Conflict Management Skills to Save Time and Enhance Productivity*, The Graduate School, Michigan State University, July 12, 2010. Available online at: http://grad.msu.edu/conflictresolution/; last accessed on December 22, 2010.
   (b) Collegiate Employment Research Institute:
      i. http://www.ceri.msu.edu/
      ii. Recruiting Trends 2010-2011: http://www.ceri.msu.edu/recruiting-trends-2009-2010/

74. *Other colleges and universities.*

75. San Francisco State University:

   (a) Eric Hsu, Department of Mathematics:
      i. Eric Hsu, *Math Education Job Search Resources*, Department of Mathematics, San Francisco State University. Available at: http://bfc.sfsu.edu/cgi-bin/hsu.pl?Math_Education_Job_Search_Resources; last accessed on September 1, 2010. Also, accessible at: http://math.sfsu.edu/hsu/jobs.html.
      ii. Has lots of information on applying for positions in academia.
      iii. Has lists of postdoc positions and (junior) faculty openings.

76. Cardiff University:
   (a) School of Psychology, College of Biomedical and Life Sciences:
      i. Chris Chambers:
         – http://psych.cf.ac.uk/contactsandpeople/researchfellows/chambers.html
         – Chris Chambers, "Tough love: An insensitive guide to thriving in your PhD," in his blog *NeuroChambers*, May 9, 2012. Available online at: http://neurochambers.blogspot.co.uk/2012/05/tough-love-insensitive-guide-to.html; last accessed on November 3, 2012. [Excellent!]

77. Colorado School of Mines:
   (a) Department of Physics:
      i. Robert L. Read, "How to be a Programmer," APPLICATI, 2003. Available online at: http://samizdat.mines.edu/howto/; last accessed on September 28, 2010. [Another publisher is FEINHOCHBURG]

78. New Mexico Institute of Mining and Technology (New Mexico Tech):
   (a) Brian Borchers, "Recommendation Letters," Department of Mathematics, New Mexico Institute of Mining and Technology. Available at: http://infohost.nmt.edu/~borchers/recletters.html; last accessed on September 2, 2010.

79. University of Minnesota Duluth:
   (a) University of Minnesota Duluth, *Is Graduate School Right For You?*, Career Services, University of Minnesota Duluth. Available at: http://www.d.umn.edu/careers/grad_school/right_for_you.html; last accessed on September 2, 2010.

80. New Jersey Institute of Technology:
   (a) Department of Information Systems; College of Computing Sciences:
      i. Prof. Michael Bieber:
         – http://www-ec.njit.edu/~bieber/
         – Research Interests, Ph.D. Projects, Selected Publications, Presentations and Other Reference: http://www-ec.njit.edu/~bieber/pubs.html

81. University of Evansville:
   (a) Department of Electrical Engineering & Computer Science; College of Engineering and Computer Science:
      i. Prof. Deborah J. Hwang:
         – Interesting Graduate School Links: http://csserver.evansville.edu/~hwang/gradlinks.html

82. Franklin University:
   (a) Department of Computing Sciences and Mathematics:
      i. Esmail Bonakdarian, "Student Page," in his personal web page. Available online at: http://cs.franklin.edu/~esmail/students.html; last accessed on August 11, 2011. [Has good information about: NSF's Research Experience for Undergraduates (REU) program; professional organizations; resources for Ruby, Python, UNIX, and Make; how to email professors; how to ask questions; study tips; and debugging tips.]

83. The University of Waikato:
   (a) Sean Oughton, *Graduate School Survival Guide*, Department of Mathematics, The University of Waikato. Available at: http://www.math.waikato.ac.nz/~seano/grad-school-advi html; last accessed on September 3, 2010. [Also, see http://www.math.waikato.ac.nz/~seano/ for Sean's web page.]
84. `Resources from non-governmental organizations & nonprofit organizations.`
85. The PhD Project:
   (a) Resources for Potential/Current Doctoral Students:
      i. http://www.phdproject.org/resources.html
      ii. Information about good business schools that offer Ph.D. programs, preparation for the GMAT, and the life in graduate school as a Ph.D. student.
      iii. Suggested Reading:
         – http://www.phdproject.org/reading.html
         – Has information life in graduate school as a Ph.D. student, racial diversity/issues in higher education, job searching in academia, and work-life balance for female Ph.D. students.
86. GradShare: http://www.gradshare.com/aboutus.html
87. PhDNet (online community for graduate students and PhDs): http://phdnet.org/
88. `Resources from companies.`
89. ABD Solution Company:
   (a) Dr. Carters Educational Group, L.L.C.:
      i. Wendy Y. Carter, *TA-DA!™ Thesis and Dissertation Accomplished*, 2011. Available online at: http://www.tadafinallyfinished.com/index.html; last accessed on January 8, 2010.
         – *TA-DA!™* Links and Resources: http://www.tadafinallyfinished.com/links/index.html
      ii. Educational Research Institute:
         – http://www.educationalresearchinstitute.org/
90. About.com:
   (a) About.com, *Graduate School*. Available at: http://gradschool.about.com/; last accessed on August 25, 2010.
   (b) Timothy Dzurilla, *Writing Graduate Application Essay: Tips to Writing a Successful Personal Statement*, About.com, Nov 1, 2007. Available at: http://www.suite101.com/content/writing-graduate-application-essay-a34598; last accessed on September 1, 2010. [Help with writing a statement of purpose]
   (c) Naomi Rockler-Gladen, *How to Choose a Graduate School: Faculty, Fit, Student Culture, and Other Grad Program Considerations*, About.com, Nov 12, 2007. Available at: http://www.suite101.com/content/how-to-choose-a-graduate-school-a35387; last accessed on September 1, 2010. [How to select a graduate program ... EXCELLENT]
   (d) Naomi Rockler-Gladen, *How to Choose a Graduate Advisor: Finding a Faculty Member to Direct an MA Thesis or PhD Dissertation*, About.com, Oct 30, 2007. Available at: http://www.suite101.com/content/how-to-choose-a-graduate-advisor-a34468; last accessed on September 1, 2010. [How to pick an advisor ... EXCELLENT]
91. UndergradEcon.com, *Graduate School Economics*. Available online at: http://www.undergradecon.com/grad_school.html; last accessed on January 9, 2010.
92. `Resources from individuals.`
93. Anonymous:

   (a) *Lightning Strikes Everyday* blog:
      i. http://hawkeyeview.blogspot.sg/
      ii. "Statement of Purpose (This applies to MBA folks also)," in the blog *Lightning Strikes Everyday*, September 6, 2005. Available online at: http://hawkeyeview.blogspot.sg/2005/09/statement-of-purpose-this-applies-to.html; last accessed on May 15, 2013.

94. Dario Toncich, *Key Factors in Postgraduate Research - A Guide for Students*. Available at: http://www.doctortee.net/KeyFactors.html; last accessed on September 1, 2010.
95. Karen Kelsky: http://theprofessorisin.com/
96. Sumit Gupta, *Articles and Information about Graduate School*. Available at: http://www.4bearsonline.com/collections/grad/index.shtml; last accessed on August 25, 2010.
97. William Stallings:
   (a) *Computer Science Student Resource Site: How-To*: http://www.computersciencestudent.com/SS/SS-howto.html
   (b) *Computer Science Student Resource Site: Computer Science Careers*: http://www.computersciencestudent.com/SS/SS-career.html
   (c) *Computer Science Student Resource Site*: http://www.computersciencestudent.com/

- Grad school admission advice:
  1. University of California, Berkeley:
     (a) Department of Economics, "Criteria," Department of Economics, University of California, Berkeley. Available at: http://www.econ.berkeley.edu/econ/grad/admit-criteria.shtml; last accessed on August 28, 2010.
  2. Harvard University:
     (a) Susan Athey, "Advice for Applying to Grad School in Economics," Department of Economics, Harvard University. Available at: http://kuznets.fas.harvard.edu/~athey/gradadv.html; last accessed on August 28, 2010.
  3. Statementofpurpose.com: http://www.statementofpurpose.com/
  4. Brian Rybarczyk, "Sell Yourself: Guidance for Developing Your Personal Statement for Graduate School Applications," in *Science*: Science Careers: Career Magazine: Previous Issues: 2006: 2006-01-06: Diversity Issues, January 06, 2006. Available online at: http://sciencecareers.sciencemag.org/career_development/previous_issues/articles/2006_01_06/sell_yourself_guidance_for_developing_your_personal_statement_for_graduate_school_applications; last accessed on September 6, 2011.

- Advice concerning research:
  1. University of California, Riverside:
     (a) John Baez, "Advice for the Young Scientist," Department of Mathematics, University of California, Riverside, March 25, 2007. Available at: http://math.ucr.edu/home/baez/advice.html; last accessed on August 28, 2010.
  2. Research Information Network, RIN:
     (a) Researchers resources: http://www.rin.ac.uk/resources/researcher-resources
     (b) Researcher development and skills: http://www.rin.ac.uk/resources/researcher-developme
     (c) Publishing: http://www.rin.ac.uk/resources/publishing
     (d) Learned and professional society: http://www.rin.ac.uk/resources/learned-and-professio

- advice about giving presentations:
  1. Cornell University, Department of Computer Science, Faculty of Computing and Information Science (CIS):

   (a) Charles F. Van Loan, "The Short Talk," Department of Computer Science, Cornell University. Available at: http://www.cs.cornell.edu/cv/ShortTalk.htm; last accessed on August 25, 2010.

2. University of Wisconsin-Madison, Computer Sciences Department:
   (a) Mark D. Hill, "Oral Presentation Advice," Computer Sciences Department, University of Wisconsin-Madison, April 1992, Revised January 1997. Available at: http://pages.cs.wisc.edu/~markhill/conference-talk.html; last accessed on August 25, 2010. It includes a short summary of a presentation on this topic by Prof. David A. Patterson. David A. Patterson, "How to Give a Bad Talk," Computer Science Division, Department of Electrical Engineering and Computer Sciences, University of California-Berkeley, 1983.
3. University of California, Los Angeles:
   (a) Terence Tao, "Talks are not the same as papers," Department of Mathematics, University of California, Los Angeles. Available at: http://terrytao.wordpress.com/career-advice/talks-are-not-the-same-as-papers/; last accessed on September 1, 2010.
4. North Carolina State University, Department of Chemical and Biomolecular Engineering:
   (a) Richard M. Felder, "Tips on Talks," Department of Chemical and Biomolecular Engineering, North Carolina State University. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Papers/speakingtips.htm; last accessed on August 28, 2010.
5. Random information:
   (a) The number of presentation slides is approximately the same as the number of minutes allocated for the presentation. Therefore, for a 15 minutes presentation, the speaker shall use about 15 slides for her/his presentation.
   (b)

- Advice on studying:
  1. State University of New York at Buffalo, Department of Computer Science and Engineering:
     (a) William J. Rapaport, "How to Study: A Brief Guide," Department of Computer Science and Engineering, Department of Philosophy, and Center for Cognitive Science, State University of New York at Buffalo, Buffalo, NY. Available at: http://www.cse.buffalo.edu/~rapaport/howtostudy.html; last accessed on August 25, 2010.
  2. University of Oregon, Teaching and Learning Center:
     (a) Ronald C. Blue, "How to Study," Teaching and Learning Center, University of Oregon. Available at: http://tep.uoregon.edu/resources/faqs/outsidehelp/study.html; last accessed on August 25, 2010.
  3. North Carolina State University, Department of Chemical and Biomolecular Engineering:
     (a) Richard M. Felder, "Handouts for Students," Department of Chemical and Biomolecular Engineering, North Carolina State University. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Student_handouts.html; last accessed on August 28, 2010.
  4. Middle Tennessee State University:
     (a) Carolyn Hopper, "The Study Skills Help Page: Learning Strategies for Success," Middle Tennessee State University. Available at: http://frank.mtsu.edu/~studskl/; last accessed on August 25, 2010.
  5. Joseph Frank Landsberger:

   (a) Joseph Frank Landsberger, *Study Guides and Strategies*. Available at: http://www.studygs.net/; last accessed on August 25, 2010.

- Advice on test preparation and test taking:
  1. North Carolina State University:
     (a) Richard M. Felder, "Random Thoughts: Memo," *Chemical Engineering Education*, Vol. 33, No. 2, pp. 136–137, 1999. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Columns/memo.html; last accessed on August 28, 2010.
     (b) Richard M. Felder and James E. Stice, "Tips on Test Taking," Department of Chemical and Biomolecular Engineering, North Carolina State University, and Deptartment of Chemical Engineering, The University of Texas at Austin. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Papers/testtaking.htm; last accessed on August 28, 2010.
     (c) Richard M. Felder and Matthias F. (Matt) Stallmann, "Tips for Test Takers," Department of Computer Science, North Carolina State University, February 17, 2005. Available online at: http://people.engr.ncsu.edu/mfms/Teaching/tips-for-test-takers.html; last accessed on October 6, 2010.

- Advice for engineering students:
  1. University of Maryland, Baltimore County; Department of Computer Science and Electrical Engineering:
     (a) Alan T. Sherman (Alan Theodore Sherman), *How To's and Other Generic Course Documents*, Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, September 12, 1995. Available at: http://www.csee.umbc.edu/~sherman/Courses/documents/; last accessed on August 28, 2010. Also, see http://www.csee.umbc.edu/~sherman/Courses/.
     (b) Alan T. Sherman (Alan Theodore Sherman), *Teaching Activites*, Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County. Available at: http://www.csee.umbc.edu/~sherman/mycourses.html; last accessed on August 28, 2010.
  2. North Carolina State University, Department of Chemical and Biomolecular Engineering:
     (a) Richard M. Felder's column, "Random Thoughts," in the journal, *Chemical Engineering Education*. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Columns.html; last accessed on August 28, 2010.
     (b) Richard M. Felder, "An Engineering Student Survival Guide," Department of Chemical and Biomolecular Engineering, North Carolina State University, 1993. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Papers/survivalguide.htm; last accessed on August 27, 2010.
        - Do not expect people to tell me how to solve certain problems, especially implementation details as a researcher.
        - Learn to find out for myself what I need to know. That is, determine the scope of things that I need to know, the time frame and deadline(s) in which I should acquire knowledge of those skills and knowledge, and create a plan to acquire those skills and knowledge.
        - I should learn to be more resourceful, and determine where can I get help. Particularly, resources (e.g., publications and online material), individuals, and networks of people that can provide a significant amount of help to people.
        - Make a serious effort to solve a problem before I approach others for help. Else, they may get annoyed when I did not bother to learn how to solve problems that are actually very simple. Also, bring my (attempted/considered) solutions to the

people who I seek help from. Show them my flow charts, schematics, calculations, algorithms, and heuristics. This would convince them that I have done my homework, and am not asking bane questions.

- To help me understand how to apply the skills and knowledge that I am acquiring in "practical, real-world applications," I should look at textbooks (including alternative textbooks/books, such as handbooks and encyclopedias) for such examples. I can also look ahead further in the chapter, book, or books of subsequent classes to see how these skills and knowledge will be applied. Note that information that I skip while reading my textbook, manuals, and guides may actually contain the solution that I am looking for. Think of the questions that I asked Anders Franzen about the SMT-LIB manual during my internship at FBK while working on the *MathSAT* project. I did not understand the material in the manual, since I lacked a background in compiler design and formal grammar/languages. So, I had to get he to help me interpret what I was reading. This was like when I was learning about UNIX as a freshman/sophomore. I did not understand what I was reading when I looked at the UNIX manual ("man pages"). Thus, I shall learn how to read technical literature better.
- By reading technical and semi-technical magazines and newspapers/newsletters, I can learn about "practical, real-world applications" of the things that I am learning about. In addition, by talking to others (students further ahead of me in engineering education and professional engineers), I learn to see how can I apply the things that I am learning about in "practical, real-world applications". Furthermore, I can tap into the newsletters and technical magazines of professional organizations, such as ACM and IEEE, to find out about research opportunities/projects where I can apply what I am learning about.
- If my lecture slides/notes and textbooks(s) do not have adequate worked-out examples (e.g., only trivial examples) to help me understand mathematical theories and formulas, and engineering concepts, I shall seek other resources. E.g., I can look at lecture slides/notes from equivalent/similar classes that are taught at other universities. In addition, I can look at other textbooks/books on this subject. Note that for advanced topics, such as those covered by advanced graduate classes, I may only be able to find 1 or 2 books on this topic. So, I may not always have the luxury of looking at worked examples from other textbooks; e.g., I could find many textbooks for differential equations and vector analysis, but not for antenna analysis or satisfiability modulo theories.
- I shall improve my ability to work out problems on my own if I cannot find adequate examples for that problem or similar problems. In addition, I shall document worked solutions digitally, so that I can refer to them during revision for an exam, my prelims/quals, or when I encounter a similar problem during research and development.
- I shall also improve the way I revise/relearn concepts, technologies, skills, and knowledge. Documenting resources and prior solutions to problems would help me relearn things. Such documentation requires proper information management, so that I can reuse the previously acquired knowledge and skills. Remember that using LaTeX on a UNIX-like operating system helps me with information management.
- If I do not understand how and why things work, determine if knowledge of that is required for solving problems in my research/class project, or assignment. If not, I can move on and address this when I have more free time (e.g., "slack" periods during the calendar year). To find out more about how certain things work, I can look

into the references in my lecture slides/notes and textbooks(s), or search/google for references online. Note that learning how and why certain things work may require (advanced) knowledge that is outside the scope of my discipline or research area. Hence, it is important to know when to stop delving into (/investigating/probing) a concept/technique. Remember my problems with understanding Prof. Sanjit A. Seshias publications on adaptive eager encodings, in which he used "polyhedral theory" (I believe in the context of integer programming and combinatorial optimization) to prove a theorem regarding the satisfiability of UTVPI formulas? Ditto for lambda expressions (and lambda calculus) so that I can understand how syntax is represented for a given signature in first-order logic.]

- I shall improve my ability to convert descriptions of architectures and techniques into hardware/software implementations. It is easier to grasp the concepts in pseudocode, flowcharts, schematics, figures, and demonstrations than to learn the concepts from text and create abstractions of those concepts on my own. I shall improve my ability to create pseudocode, flowcharts, schematics, figures, and demonstrations from what I have read, especially in journal and conference papers.
- I shall improve my ability to perform statistical analysis on my experimental data, and analyze the figures/graphs that I have plotted.
- Use a book from the *Schaum's Outline* series from *McGraw-Hill* to help me learn material from introductory and intermediate classes. "Even if you can't find a reference with exactly the type of coverage that works best for you, just reading about the same topic in two different places usually clarifies the ideas." [Remember how I read about the same topic in different advance engineering math textbooks to learn concepts for my classes in differential equations and vector analysis?]
- Working with others allows me to overcome obstacles that I may not be able to overcome on my own. While I may give up on learning certain things or overcome specific problems in individual projects, my teammates may be able to come up with solutions to problems in group projects. In addition, working in a diverse group exposes me to solutions that can be more effective and/or efficient... *students routinely teach one another in group work – and as any professor will tell you, teaching something is probably the most effective way to learn it.*
- Try to find groups of three to four people to work on a problem. When I work in pairs, I may not expose myself to a sufficient variety of approaches. Similarly, when I work in larger groups (i.e., > 4), some individuals may be left out of the "active problem-solving process".
- I shall endeavor to outline solution on my own first, without being boggled or encumbered by the implementation details. Subsequently, I can work out the complete solutions with my group. If each individual does this, each group member can learn how to get started in solving problems in the project. That is, let's outline solutions to the problem, before we meet to discuss our considered solutions and develop the complete solution together.
- "For group work to be fully effective, every group member should be able to explain in detail every solution obtained in a work session. Having the group members (particularly the weaker ones) go through these explanations before ending the session is a good way to make sure that the session has achieved its objectives." This will mitigate the tendency for the more technically challenged and reserved individuals to accept proposed solutions without understanding those solutions.

(c) Richard M. Felder, "How to Survive Engineering School," Department of Chemical and Biomolecular Engineering, North Carolina State University. Available at: http://www4.

ncsu.edu/unity/lockers/users/f/felder/public/Columns/Surviving-School.html; last accessed on August 28, 2010.

- Advice on teaching:
  1. For offline advice, see §5.7.
  2. Stanford University:
     (a) Stanford University, *Teaching at Stanford*, Center for Teaching and Learning, Stanford University. Available at: http://ctl.stanford.edu/teaching-at-stanford.html; last accessed on September 1, 2010.
     (b) Stanford University, *Handouts and Teaching Tips*, Center for Teaching and Learning, Stanford University. Available at: http://ctl.stanford.edu/teachingta/handouts-and-teac html; last accessed on September 1, 2010.
     (c) Stanford University, *Speaking of Teaching Newsletters*, Center for Teaching and Learning, Stanford University. Available at: http://ctl.stanford.edu/speaking-of-teaching-newslet html; last accessed on September 1, 2010.
  3. University of California, Riverside; Department of Mathematics:
     (a) John Baez, "How to Teach Stuff," Department of Mathematics, University of California, Riverside, January 23, 2006. Available at: http://math.ucr.edu/home/baez/teaching.html; last accessed on August 28, 2010.
  4. University of Oregon, Teaching and Learning Center:
     (a) Teaching Effectiveness Program, *Teaching Resources*, Teaching and Learning Center, University of Oregon. Available at: http://tep.uoregon.edu/resources/index.html; last accessed on August 25, 2010. Also, look the "Teaching FAQ's": http://tep.uoregon.edu/resources/faqs/
     (b) Teaching Effectiveness Program, Resources for *Teaching with Technology*, Teaching and Learning Center, University of Oregon. Available at: http://tep.uoregon.edu/technology/index.html; last accessed on August 25, 2010.
  5. North Carolina State University, Department of Chemical and Biomolecular Engineering:
     (a) Richard M. Felder, *Student-centered Teaching and Learning*, Department of Chemical and Biomolecular Engineering, North Carolina State University. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Student-Centered.html; last accessed on August 28, 2010.
     (b) Richard M. Felder, *Index of Learning Styles*, Department of Chemical and Biomolecular Engineering, North Carolina State University. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/ILSpage.html; last accessed on August 28, 2010.
     (c) Richard M. Felder, *Learning Styles*, Department of Chemical and Biomolecular Engineering, North Carolina State University. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Learning_Styles.html; last accessed on August 28, 2010.
     (d) Richard M. Felder, *Richard Felder's Education-related Publications*, Department of Chemical and Biomolecular Engineering, North Carolina State University. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Papers/Education_Papers.html; last accessed on August 28, 2010.
  6. Joseph Frank Landsberger:
     (a) Joseph Frank Landsberger, *Teaching Guides and Strategies*. Available at: http://www.studygs.net/teaching/; last accessed on August 25, 2010.
- Resources to improve my English skills:
  1. *Guide to Online Schools*:

   (a) *Guide to Online Schools* [or *GuideToOnlineSchools.com*], *Resources to Help Improve Your English Pronunciation.* Available at: http://www.guidetoonlineschools.com/tips-and-tools/english-pronunciation; last accessed on August 25, 2010.

- Resources concerning time management:
  1. *Guide to Online Schools*:
     (a) *Guide to Online Schools* [or *GuideToOnlineSchools.com*], *The Best Compilation of Time Management Resources on the Web.* Available at: http://www.guidetoonlineschools.com/tips-and-tools/time-management; last accessed on August 25, 2010.

- Resources for Math and Science Revision:
  1. Basic high school math:
     (a) North Carolina State University, Department of Chemical and Biomolecular Engineering:
        - Kenny Felder and Gary Felder, "Kenny's Math and Physics Help," 2009. Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/kenny/home.html; last accessed on August 28, 2010.
        - Kenny Felder, "Selected Other Educational Sites on the Web". Available at: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/kenny/edulinks.html; last accessed on August 28, 2010.

- Good blogs about graduate school:
  1. Marc Eaddy, *Marc Eaddy: Confessions of an Ex-PhD Student.* Available at: http://marceaddy.blogspot.com/; last accessed on August 28, 2010.
  2. The Academic Blog Portal: http://academicblogs.org/wiki/index.php/Main_Page

- Fun stuff about grad school:
  1. Jorge Cham, *Piled Higher and Deeper.* Available at: http://www.phdcomics.com/; last accessed on August 28, 2010. See the latest "PHD Comics" at: http://www.phdcomics.com/comics.php. This comic strip pokes fun at the [fun, harsh, interesting, and absurd] realities of life in grad school.
  2. Jorge Cham, *Academica.* Available online at: http://academia.edu/academica; last accessed on October 3, 2010.
  3. Jorge Cham, *The PhD Forums.* Available online at: http://www.phdcomics.com/proceedings/index.php; last accessed on October 3, 2010. [Has useful guidelines for surviving graduate school and is a decent resource for some technical support (e.g., with LaTeX or C++).]

- Other information about or related to grad school (and higher education):
  1. `Web 2.0 services (including review sites for professors and universities).`
  2. Chegg (homework help + used textbooks + textbook rental): http://www.chegg.com/
  3. CourseRank: https://www.courserank.com/w/home
  4. MyEdu:
     (a) http://www.myedu.com/
     (b) I can use it to select my classes, plan my class timetable, check my progress towards graduation, and organize my calendar.
  5. PhinisheD: http://www.phinished.org/index.php
  6. RateMyProfessors.com: http://www.ratemyprofessors.com/
  7. `Networking resources.`
  8. *Eurodoc*: union of grad student associations of each European country; see http://en.wikipedia.org/wiki/EURODOC and http://www.eurodoc.net/

9. Think tanks or policy institutes concerning (higher) education.
10. Alliance for Excellent Education: http://www.all4ed.org/
11. American Educational Research Association (AERA): http://www.aera.net/
12. Association of American Colleges and Universities (AAC&U):
    (a) http://www.aacu.org/
    (b) DiversityWeb:
        i. The DiversityWeb project is housed within the Office of Diversity, Equity and Global Initiatives at the Association of American Colleges and Universities (AAC&U)
        ii. http://www.diversityweb.org/index.cfm
13. American Council of Trustees and Alumni (ACTA):
    (a) The American Council of Trustees and Alumni (ACTA) is an independent, non-profit organization committed to academic freedom, excellence, and accountability at Americas colleges and universities.
    (b) Launched in 1995, we are the only organization that works with alumni, donors, trustees, and education leaders across the United States to support liberal arts education, uphold high academic standards, safeguard the free exchange of ideas on campus, and ensure that the next generation receives a philosophically rich, high-quality college education at an affordable price.
    (c) ACTA Publications: https://www.goacta.org/publications/. [ACTA publications cover many aspects of issues concerning higher education institutions, and serve to provide standards of academic excellence and strategies for achieving these standards.]
14. GOOD, *GOOD Education*: http://www.good.is/series/good-education/
15. American Council on Education: http://www.acenet.edu
16. College Measures (a joint endeavor by American Institutes for Research and Matrix Knowledge Group): http://collegemeasures.org/
17. CollegeXpress: http://www.collegexpress.com/
18. Council for Aid to Education (CAE):
    (a) http://www.cae.org/
    (b) Collegiate Learning Assessment (CLA): http://www.collegiatelearningassessment.org/
19. Council for Higher Education Accreditation: http://www.chea.org/
20. Council of Graduate Schools (CGS):
    (a) http://www.cgsnet.org/
    (b) Ph.D. Completion Project: http://www.phdcompletion.org/
21. Education Commission of the States: http://www.ecs.org/
22. Education Sector: http://www.educationsector.org/
23. EDUCAUSE Review Online: http://www.educause.edu/ero
24. *European Association for Quality Assurance in Higher Education* (ENQA): union of accreditation board(s) of each European country; see http://en.wikipedia.org/wiki/ENQA and http://www.enqa.eu/
25. European University Association (EUA): http://www.eua.be/Home.aspx
26. Ewing Marion Kauffman Foundation:
    (a) http://www.kauffman.org/
    (b) Research and Policy:
        i. http://www.kauffman.org/Section.aspx?id=Research_And_Policy
        ii. Research Reports: http://www.kauffman.org/Research.aspx
    (c) Education: http://www.kauffman.org/Section.aspx?id=Education
    (d) Advancing Innovation: http://www.kauffman.org/Section.aspx?id=Advancing_Innovation

    (e) Entrepreneurship: http://www.kauffman.org/Section.aspx?id=Entrepreneurship

27. Excelencia in Education: http://www.edexcelencia.org/
28. Glion Colloquium (Geneva, Switzerland):
    (a) http://www.glion.org/
    (b) "An independent think tank committed to the future and responsibilities of research universities."
    (c) Publications: http://www.glion.org/publications.aspx
29. Graduate Software Engineering 2009 (GSwE2009): http://www.gswe2009.org/
30. Higher Education Academy: http://www.heacademy.ac.uk/
31. *Innolyst*:
    (a) Innolyst, *ResearchCrossroads*, Innolyst:
        i. Ernest Kuh, UC Berkeley: http://www.researchcrossroads.org/Researchers/830167 or http://www.researchcrossroads.org/index.php?option=com_content&view=article&id=49&Itemid=55&user_id=830167
        ii. US-based researchers who receive US government funding for their research have profiles in *ResearchCrossroads*. E.g., I can find the amount of public funding that my professors at USC received, and the organization that funds them. I can also read an abstract of the project that they got funded for.
        iii. http://www.researchcrossroads.org/
    (b) http://www.innolyst.com/
32. Institute for Educational Leadership: http://www.iel.org/
33. Institute for Higher Education Policy: http://www.ihep.org/
34. Institute of International Education:
    (a) http://www.iienetwork.org/
    (b) Open Doors Data:
        i. http://www.iie.org/Research-and-Publications/Open-Doors/Data
        ii. Contains statistical data concerning international students in the United States and U.S. students studying abroad.
35. Lumina Foundation for Education, *Publications*. Available at: http://www.luminafoundation.org/publications/; last accessed on September 4, 2010.
36. National Association for Equal Opportunity in Higher Education (NAFEO): http://www.nafeo.org/community/index.php
37. National Association of Scholars: http://nas.org/
38. National Center for Academic Transformation: http://thencat.org/
39. National Center for Faculty Development & Diversity: http://www.facultydiversity.org/
40. National Center for Public Policy and Higher Education:
    (a) http://www.highereducation.org/
    (b) *Measuring Up 2006*: http://measuringup.highereducation.org/
    (c) Reports and Publications: http://www.highereducation.org/reports/reports.shtml
41. National Student Clearinghouse (Herndon, VA):
    (a) http://www.studentclearinghouse.org
    (b) National Student Clearinghouse® Research Center: http://nscresearchcenter.org
42. Pell Institute for the Study of Opportunity in Higher Education: http://www.pellinstitute.org/
43. Task Force on American Innovation: http://www.innovationtaskforce.org/
44. The Center for College Affordability and Productivity (CCAP): http://centerforcollegeaffordabi
    org
45. The Education Trust: http://www.edtrust.org/

46. Thurgood Marshall College Fund: http://www.thurgoodmarshallfund.net/
47. TIAA-CREF:
    (a) http://www.tiaa-crefinstitute.org/institute/index.html
    (b) Research & Publications: http://www.tiaa-crefinstitute.org/institute/research/index.html
48. Resources concerning business degree programs.
49. Bloomberg L.P.:
    (a) Bloomberg Businessweek:
        i. http://www.businessweek.com/
        ii. Business Schools: http://www.businessweek.com/business-schools/
50. Business-Higher Education Forum: http://www.bhef.com/
51. The Association to Advance Collegiate Schools of Business (AACSB International):
    (a) http://www.aacsb.edu/
    (b) Best Biz Schools: http://www.bestbizschools.com/
52. Resources concerning Student-Atheletes..
53. University of Central Florida:
    (a) DeVos Sport Business Management Graduate Program, College of Business Administration:
        i. Institute for Diversity and Ethics in Sport: http://tidesport.org/
54. Knight Commission on Intercollegiate Athletics: http://knightcommission.org/
55. Resources concerning minorities/underrepresented minorities.
56. Hispanic Association of Colleges and Universities (HACU): http://www.hacu.net/hacu/Default_EN.asp
57. Black Alliance for Educational Options: http://www.baeo.org/
58. American Indian Higher Education Consortium (AIHEC): http://www.aihec.org/
59. Resources for entrance exams: e.g., GRE, GMAT, LSAT, MCAT.
60. GRE: [2, 7, 8, 11, 16, 18, 20, 22]
61. Quora, Inc., "If you had just one day to prepare for the GRE, how would you go about preparing?", in *Quora: Graduate School*, Quora, Inc., Palo Alto, CA, 2012. Available online at: http://www.quora.com/Graduate-School/If-you-had-just-one-day-to-prepare-for-the-GRE last accessed on September 10, 2012.
62. Resources from universities.
63. Columbia University:
    (a) :
        i.
        ii. The Hechinger Report: http://hechingerreport.org
64. Indiana University Bloomington:
    (a) National Survey of Student Engagement; Indiana University Center for Postsecondary Research:
        i. http://nsse.iub.edu/
        ii. NSSE survey: http://nsse.iub.edu/nsse2013/
        iii. NSSE Annual Results 2011–Fostering Student Engagement Campuswide: http://nsse.iub.edu/html/annual_results.cfm
        iv. Institutional Report: http://nsse.iub.edu/_/?cid=402
        v. NSSE Findings: http://nsse.iub.edu/html/reports.cfm
        vi. Survey Instrument: http://nsse.iub.edu/html/survey_instruments.cfm
65. National University of Singapore:

   (a) Centre for Development of Teaching and Learning (CDTL): http://www.cdtl.nus.edu.sg

66. Stanford University:
   (a) Center for Teaching and Learning:
      i. http://ctl.stanford.edu/
      ii. Tomorrow's Professor mailing list: http://cgi.stanford.edu/~dept-ctl/cgi-bin/tomprof/postings.php and http://www.stanford.edu/dept/CTL/Tomprof/index.shtml
   (b) Stanford School of Engineering:
      i. "ENGR 245: The Lean LaunchPad" (Spring 2012) by Steve Blank, Ann Miura-Ko, and Jon Feiber. Available online at: http://e245.stanford.edu/; last accessed on July 30, 2012.
         – Steve Blank's blog posts about ENGR 245 or lean launch pads: http://steveblank.com/category/lean-launchpad/
67. University of California:
   (a) UC Accountability Report: http://accountability.universityofcalifornia.edu/
68. University of Southern California:
   (a) USC Center for Excellence in Teaching: http://cet.usc.edu/ and http://uscta.wikidot.com/ (for TAs)
69. Vanderbilt University:
   (a) Center for Teaching: http://cft.vanderbilt.edu/
70. `Resources from national organizations.`
71. NAFSA / Association of International Educators (formerly, National Association of Foreign Student Advisers):
   (a) http://www.nafsa.org/
   (b) For Students:
      i. http://www.nafsa.org/students.sec/
      ii. Resources about seeking financial aid for study abroad programs, absentee ballot procedure
   (c) *Connecting Our World*: http://www.connectingourworld.org/
72. American Association of State Colleges and Universities: http://www.aascu.org
73. American Association of University Professors: http://www.aaup.org/aaup
74. Association for the European Rural Universities: http://www.ure-apure.org/ and http://enrd.ec.europa.eu/networks-and-networking/eu-organisations/en/apure_en.cfm
75. Association of American Universities (AAU): http://www.aau.edu/
76. Council on Undergraduate Research (Washington, DC): http://www.cur.org/
77. Quality Matters Program (QM): https://www.qualitymatters.org/
78. The Center for Public Education: http://www.centerforpubliceducation.org/
79. The Coalition of Urban Serving Universities: http://www.usucoalition.org/
80. Universities Research Association: http://www.ura-hq.org/
81. `Resources form governmental organizations/agencies.`
82. Department for Business, Innovation and Skills (BIS) – UK:
   (a) http://www.bis.gov.uk/
   (b) Foresight Programme: http://www.bis.gov.uk/foresight
   (c) Council for Science and Technology: http://www.bis.gov.uk/cst
   (d) Government Office for Science (GO-Science): http://www.bis.gov.uk/go-science
   (e) Research Councils UK (RCUK):
      i. http://www.rcuk.ac.uk/

ii. Arts and Humanities Research Council (AHRC): http://www.ahrc.ac.uk/
iii. Biotechnology and Biological Sciences Research Council (BBSRC): http://www.bbsrc.ac.uk
iv. Engineering and Physical Sciences Research Council (EPSRC): http://www.epsrc.ac.uk
v. Economic and Social Research Council (ESRC): http://www.esrc.ac.uk/
vi. Medical Research Council (MRC): http://www.mrc.ac.uk
vii. Natural Environment Research Council (NERC): http://www.nerc.ac.uk
viii. Science and Technology Facilities Council (STFC): http://www.scitech.ac.uk/

83. National Academies:
   (a) National Academy of Sciences (NAS)
   (b) National Academy of Engineering (NAE)
   (c) Institute of Medicine (IOM)
   (d) National Research Council (NRC):
      i. United States National Research Council rankings: http://en.wikipedia.org/wiki/United_States_National_Research_Council_Rankings
   (e) National Academies Summer Institute on Undergraduate Education; Howard Hughes Medical Institute and the National Academies: http://www.AcademiesSummerInstitute.org/
   (f) Policy and Global Affairs (PGA) Committee:
      i. http://sites.nationalacademies.org/PGA/index.htm
      ii. Look at the various Program Units of PGA to find reports of interest to me.
      iii. Data-Based Assessment of Research-Doctorate Programs; Board on Higher Education and Workforce:
         – http://sites.nationalacademies.org/PGA/Resdoc/index.htm
         – **A Data-Based Assessment of Research-Doctorate Programs in the United States**: http://www.nap.edu/rdp/
      iv.
   (g) Division on Engineering and Physical Sciences:
      i. http://sites.nationalacademies.org/DEPS/index.htm

84. The National Science Foundation:
   (a) NSF Innovation Corps (I-Corps):
      i. http://www.nsf.gov/news/special_reports/i-corps/
      ii. "The NSF Innovation Corps (I-Corps) is a set of activities and programs that prepare scientists and engineers to extend their focus beyond the laboratory and broadens the impact of select, NSF-funded, basic-research projects."
      iii. I-Corps Teams:
         – "I-Corps Teams are composed of three main members: the principal investigator, the entrepreneurial lead and the mentor. The principal investigator (PI) serves as the technical lead and project manager."
         – "The entrepreneurial lead (EL), typically a postdoctoral researcher, graduate student, or other student, possesses relevant technical knowledge and a deep commitment to investigate the commercial landscape surrounding the innovation. The entrepreneurial lead should also be prepared to support the transition of the technology, should the I-Corps project demonstrate a level of readiness appropriate to leave the academic institution."
         – "The mentor brings entrepreneurial experience and serves as the principal guide in determining the technology disposition."
      iv. I-Corps Curriculum:

- "The I-Corps curriculum provides real-world, hands-on, immersive learning about what it takes to successfully transfer knowledge into products and processes that benefit society. It's not about how to write a research paper, business plan, or NSF proposal. The end result is not a publication or a deck of slides or even a scientific discovery."
- "Instead the entire I-Corps Team will be engaged with industry; talking to customers, partners, and competitors; and encountering the chaos and uncertainty of creating successful innovations. Getting out of the laboratory/university is what the effort is about."

v. The Innovation Ecosystem:
- "The people, institutions, policies and resources that promote the translation of new ideas into products, processes and services are generally recognized to comprise the innovation ecosystem."

(b) National Center for Science and Engineering Statistics (NCSES):
  i. http://www.nsf.gov/statistics/
  ii. Surveys: http://www.nsf.gov/statistics/survey.cfm
  iii. The web page of NCSES contains statistics about education and research in the United States, as well as the presence and impact of international students and high-skilled immigrant professionals.
  iv. *Women, Minorities, and Persons with Disabilities in Science and Engineering* provides statistical information about the participation of women, minorities, and persons with disabilities in science and engineering education and employment: http://www.nsf.gov/statistics/wmpd/
  v. Science and Engineering Indicators (201X): http://www.nsf.gov/statistics/indicators/ http://www.nsf.gov/statistics/seind10/, or http://www.nsf.gov/statistics/seind12/
  vi. Survey of Earned Doctorates: http://www.nsf.gov/statistics/srvydoctorates/
  vii. Survey of Graduate Students and Postdoctorates in Science and Engineering: http://www.nsf.gov/statistics/srvygradpostdoc/

85. U.S. Department of Education:
  (a) College Affordability and Transparency Center:
    i. College Scorecards: http://www.whitehouse.gov/issues/education/higher-education/college-score-card

86. `Resources from newspapers and media companies.`
87. Inside Higher Ed: http://www.insidehighered.com/
88. New York Times:
  (a) The Choice on India Ink: http://india.blogs.nytimes.com/category/uncategorized/the-choice-on-india-ink/
  (b) The Choice–Demystifying College Admissions and Aid: http://thechoice.blogs.nytimes.com/
89. Newsweek Education: http://education.newsweek.com/index.html
90. The Chronicle of Higher Education: http://chronicle.com/
91. Washington Monthly; W. M. Corporation, Washington, D.C.:
  (a) http://www.washingtonmonthly.com/index.php
  (b) College Guide:
    i. http://www.washingtonmonthly.com/college_guide/ and http://www.washingtonmonth com/college_guide/index.php
    ii. *Washington Monthly* College Guide and Rankings: http://www.washingtonmonthly.com/college_guide/toc_2012.php

   (c) College Reality Check: http://collegerealitycheck.com

92. `Other Resources.`
93. AmeriCareers:
    (a) http://www.americareers.com/
    (b) HigherEdSpace.com: http://www.HigherEdSpace.com/
94. Blackboard: http://www.blackboard.com
95. Books:
    (a) Books from my BibTEX database:
        i.
    (b) The "Higher Education: Handbook of Theory and Research" series from Springer Science+Busines
    Media, B.V.. Its current series editor (as of May 3, 2013) is Michael B. Paulsen, and has been published since 1985.
    (c) The "To Improve the Academy: Resources for Faculty, Instructional, and Organizational Development" series
96. College Confidential: http://www.collegeconfidential.com/
97. College Measures (based in Rockville, MD): http://collegemeasures.org
98. College Portrait:
    (a) http://www.collegeportraits.org/
    (b) Also, see Voluntary System of Accountability (VSA)
99. Evidence: http://www.evidence.co.uk/
100. GradShare: http://www.gradshare.com/aboutus.html
101. GradSchools.com: http://www.gradschools.com/
102. National Institute for Learning Outcomes Assessment (NILOA): http://www.learningoutcomeasses
    org/
103. NerdWallet, Inc.:
    (a) http://www.nerdwallet.com/
    (b) NerdWallet — Education:
        i. http://www.nerdwallet.com/blog/education/
        ii. Compare Colleges: http://www.nerdwallet.com/education/grad_surveys/
104. The Edupunks' Guide:
    (a) http://edupunksguide.org/
    (b) Resources: http://edupunksguide.org/resources
    (c) Tutorials: http://edupunksguide.org/tutorials
105. Unigo:
    (a) http://www.unigo.com/
    (b) Offers students' perspectives on various aspects of college life, from admissions and dorm/college life to studying abroad and academics (studying skills and selecting a major)
106. Voluntary System of Accountability (VSA): http://www.voluntarysystem.org/
107. Zinch: http://www.zinch.com/

7.6 Zhiyang's Suggestions for Graduate School Applications

Notes about applying to graduate school:

1. Unless you go to a good research university in the US, you have to make trade-offs. So, it's a matter of picking a more appropriate trade-off for you, rather than for anybody else. Each person would naturally prefer a different trade-off and perform better in certain types of trade-offs. Hence, your task is to find out what trade-offs do you have to make, and make them so that you can be part of

Table 1: Attitudes of CS students with respect to their CS class/course load.

#CS Classes	Attitude
1–2	I really like computer science, but I think it is important to leave time for other things I like to do as well.
3	I really can't think of anything I'd rather do than sleep, eat, and program.
4	I don't know why I pay for housing because I end up staying in Gates every night.
≥5	I am too busy to have an attitude.

an academic environment that poorly resembles those at good U.S. research universities... If you have not published good research papers, like some undergrads in the US, you don't have much of a choice. And, you can't effectively formulate this as a multi-objective optimization problem and find the most suitable Pareto-optimal trade-off because of your ignorance, bias, and prejudices.

2. As a general guide, a lot of the NCAA Division I schools (especially research universities that are good at sports) are also good or average research universities. Think about Michigan, UT Austin, University of Florida, University of Illinois at Urbana-Champaign, UCLA, USC, University of North Carolina at Chapel Hill, University of Virginia, University of Massachusetts at Amherst.
3. Personally, I would say pick a program based on its technical excellence. Join the best lab in your preferred research area/topic. If it can provide you with a good broad and deep base via interdisciplinary classes and rare advanced classes in your research topic, go for it. As far as the culture and environment of the campus goes, you can adapt to it. If you are smart enough to do a Ph.D., you can find workarounds or poor substitutes for different problems and challenges that you would face. You may not always or regularly find workarounds or poor substitutes, but you can.
4. Things that you may want to look into:
   (a) Availability of research lab and professors in area of research interest
   (b) Availability of very advanced classes for grad students that are hardly offered in the world. For example, a class on satisfiability modulo theories (SMT), neural implant engineering (alternatively known as neural prostheses design, neuroprosthetics design, or neural prosthetics design), neural image processing, or sequential equivalence checking may only be offered by 1-5 universities in the world. These classes should have 2-4 big projects, a weekly/fortnightly assignment, ≥1 midterm (mid-semester exam), and (perhaps) a final (exam). You should look to see if programs have several graduate classes in your research area of interest; good Ph.D. programs have several graduate classes for their Ph.D. students to help them gain technical knowledge in their research interests/topics (via the big projects using industrial tools and tool/work flows for EE/CS grad students). These classes have small numbers of students (e.g., 5-30), and have high class numbers (e.g., EE 290A at Berkeley or EE 681 at USC)
   (c) Are there enough advanced classes for grad students to help you with learning additional information and skills (this may not be that helpful in outside science and engineering), especially for subject material that is hard to pick up on your own? What is the quality of the coursework involved like? In terms of classes to be taken, how much is the academic workload for Ph.D. students?

(d) For the research labs that you want to join, how many of the grad students get funded? Does the professor help her/his students get scholarships, fellowships, and internships? Have any of the professor's students or alumni win best paper awards, best dissertation awards from organizations such as ACM, and prestigious Ph.D. and postdoc fellowships from Microsoft, Intel, IBM, NVIDIA, and Google? How accomplished are the professors that you wanna work with? If you want to be a professor, how many of the professors's former students (including alumni from her/his current research lab) end up as tenured faculty in world-class universities? If you wanna become a research scientist, how many of them are working as research scientists at IBM, Intel, Sun Microsystems, HP, Bell Labs, Google, Yahoo, Microsoft, NVIDIA, or Adobe?

(e) Availability of funding (for the project and your grad student stipend)

(f) Number of professors in research area (preferably 2-6), which is more general than a research topic

(g) Start-up/entrepreneurship and industrial involvement (created, co-founded, or funded start-ups and collaborate with industry on research projects). Pay attention to professors who get funding from big companies or consortiums in your research area. E.g., Google or Intel

(h) Support and encouragement for interdisciplinary research

(i) Does the culture of the research lab suit you? You can guess the culture of the lab by looking at the web pages of the professor(s) and students in the lab. Do they use programming languages that most people have difficulty learning or are too lazy to learn (e.g., Clojure) or software development tools that average students do not use (e.g., GNU Autotools)? Do they use a UNIX-like OS? Is the lab diverse? As in, are they mostly Indian, Persian, or Korean?

(j) Weather of the city/town

(k) Campus safety (Are there mad people running around with guns?) as well as safety in the surrounding communities

(l) Support for student groups, such as international students, women, and minorities. This also includes support groups and staff teams on campus, such as the departments dealing with residential education, support for sexual assault survivors, support for students whom have to deal with deaths of loved ones, and support for the significant others of Ph.D. students.

(m) Amount of stipend for grad students

(n) Student services, including medical, dental, and counseling services

(o) Dining options on and around campus. Trust me, you don't want to eat goop and rice everyday. Also, if you have to stay up till 4 or 6 am to work on a class project or conference paper, can you get food (e.g., a sandwich, burger, or burrito) and coffee/tea or energy drinks (e.g., Red Bull or Rockstar) at 3 am?

(p) Environment of the campus and neighborhood. Does it have a nice arboretum that you can walk in, just like those at Berkeley, Stanford, or UCLA? Psst, it's a nice place to walk around to chill out with a date, to run in, or just to take your mind off research.

(q) Social life on campus and in the town/city. Do you want to be single for several more years?

(r) Student housing. Can you stay in a dorm or residential college, which have various activities to help you develop academically, personally, and professionally? Events at the residential colleges that I have been in include career talks by engineers from Motorola, a visit to the university's supercomputer center and research lab in computer vision, tutorials to provide academic support for students, seminars on writing your resume/CV (there is a difference between the CV and resume in the US), seminars on applying for internships as international students, and many cultural and social events. If you prefer to live in an apartment, can you find affordable apartments on campus or close to campus? Does the campus have excellent social events for students, including those in the arts and music?

(s) The dominant language and culture on campus (a Taiwanese university will issue Chinese names to students who do not have one), and the surrounding neighborhood around campus and in the town/city.
(t) Social, economic, and political stability of the country.
(u) Immigration issues for international students.
(v) Some advisors are crazy, and treat you like their property/slave. Are there any professors in the Ph.D. program who are like that? Can you find out if the professor who you want to work with is supportive of students who desire to get married during their Ph.D. programs, or female students whom get pregnant then, and students whom have to deal with deaths of loved ones? With regards to the first two cases, I am not suggesting that you make an absolute decision about whether you will get married or have a child during the Ph.D. program and stick to it. What I would want to know is that if I decide to get married during my Ph.D. candidature, regardless of my current decision about getting married as a grad student, I would like to know if my advisor would disapprove of it. For female grad students, it would be nice to have an advisor who cares about your personal needs and desires as well as your academic and research progress.

5. Good research universities will satisfy most of your preferences for the above considerations (in the sub-list, "Things that you may want to look into"). As you can see, you will miss out on a lot when you are not studying in a world-class research university in the US. Hence, you should find a Ph.D. program that allows you to have as many of these benefits/advantages as possible. I would encourage you to focus on satisfying your considerations for the first few points that are relevant to the classes that you will take and research that you will carry out.
6. In summary, examine your personal and professional goals and desires, skills and knowledge, as well as strengths and weaknesses. Subsequently, determine which Ph.D. program and higher education system (e.g., US or European system) is more suitable for you to become a good researcher in your chosen field (e.g., computer science) and research area/topic (e.g., electronic design automation / logic synthesis). Next, apply to programs that are more suitable for you. When you get your Ph.D., if you are considering a career in academia, you can apply the same method to select research labs that you may want to join as a postdoc.
7. In Apply-to-Grad-Schools, when no research area is preferred [lines 1-2], you may want to apply to the best Ph.D./graduate program that you can get into. This requires you to estimate the rank of the Ph.D. programs under consideration. This means that the ranking that you use would be based on the (biased) opinions of others. You can attempt to rank the universities on your own by obtaining your own data. However, this involves a lot of work, and would be an overkill... [lines 1-17] A lot of US Ph.D. programs don't require you to work only in the research labs that you mention in your statement of purpose. In Europe, you must convince a professor (who is leading the research group/lab that you want to join) or the admissions committee that a particular research topic interests you by explaining why it interests you... [line 13] Use reference management software to manage the database of references. Examples of reference management include *Mendeley*, *JabRef*, and *BibDesk*... [line 20] Note that US Ph.D. programs tend to have deadlines in December/January; European Ph.D. programs have different deadlines, depending on the country and the university. Canadian Ph.D. programs have deadlines from December till March, Taiwanese Ph.D. programs have deadlines in March and April, and Ph.D. programs at the Technion in Israel have deadlines on April 15 and Sep 15... [line 24] By applying to reach schools, match schools, followed by safety schools, I would have some room for error with respect to match schools. If I mess up an earlier application, I can learn from my mistakes... [line 25] Use comments in your LaTeX documents to comment/uncomment previously written sections that

can be reused later... [line 26] Examples of revision control systems include *CVS*, *Subversion*, *Git*, and *Mercurial*... [line 26] When you apply to labs in a given research area, some of the projects in these labs may be similar. Hence, you can reuse information from previous SOPs. However, always customize your SOP...

```
Apply-to-Grad-Schools()
    // Input : None
    // Output : None

 1  if no-research-area-preferred
 2      Apply to the best Ph.D. program that you can get into
 3      exit // End of procedure
 4  elseif you want to work in a research area that you like
        // x := number of weeks for literature review
 5      int x = 4;

 6      while $x > 0$
 7          Read technical magazines to get an idea of research trends
            // E.g., IEEE Computer magazine or Communications of the ACM

 8          if (topic is too difficult/challenging for me)
 9              ignore that topic
10          elseif (topic is interesting and doable)
11              take note of that topic
12              read journal and conference papers of that topic
13              for each paper, enter it into your database of references

14              if (a professor's papers are very interesting)
15                  apply to that lab

16          narrow down list of research topics to 2-3 related topics
            // use the 2/3 topics for grad school applications
17          x–;

18  if (research area has been selected)
19      Decide which higher education system do I prefer (e.g., US/Europe)
20      Determine the academic requirements and application deadlines
21      Group research labs or Ph.D. programs into match schools, safety schools, & reach schools
        // Most of your programs that you want to apply to should be match schools
        // Apply to some reach schools and safety schools
        // Or group them into reach, target, and reliable schools.

22      Put the deadlines on a digital calendar // E.g., use Google Calendar or iCal
23      Create a prioritized timeline of the deadlines // A Gantt chart will help
24      Apply to reach schools, match schools, followed by safety schools
25      Use LaTeX to write your resume/CV, and statement of purpose (SOP)
26      Use a revision control system to keep track of the changes in your LaTeX documents
```

7.7 Information on Academic Careers

Information on academic careers:

1. University of Michigan, Ann Arbor:
   (a) James and Anne Duderstadt Center:
      i. The Millennium Project:
         A. http://milproj.dc.umich.edu/home/
         B. Publications: http://milproj.dc.umich.edu/publications/
         C. Classes:
            - http://milproj.dc.umich.edu/classes/
            - Rackham 570 – Preparing for Academic Careers, Rackham School of Graduate Studies (Fall 2000) by Prof. James J. Duderstadt: http://milproj.dc.umich.edu/classes/rack570/index.html
            - Engineering 390 – Topics in Engineering: An Introduction to the Engineering Profession (Winter Term, 1998) by Prof. James J. Duderstadt: http://milproj.dc.umich.edu/classes/eng390/index.html and http://milproj.dc.umich.edu/classes/eng390/index2.html
            - SP733 – Seminar on Issues in Higher Education: A Graduate Seminar with Prof. Edie Goldenberg and Prof. James J. Duderstadt (Fall 1999): http://milproj.dc.umich.edu/classes/spp733/
2. College and University Professional Association for Human Resources (CUPA-HR); Knoxville, TN: http://www.cupahr.org/index.aspx
3. http://gradpsych.apags.org/sep03/cv.cfm
4. http://vpge.stanford.edu/students/acad_chats.html
5. *AdjunctNation.com* (The Adjunct Advocate Magazine): http://www.adjunctnation.com/jobs/

7.8 Research and Writing Skills

Research and writing skills:

- http://www.cs.berkeley.edu/~fox/paper_writing.html
- http://www.sigsoft.org/resources/
- http://www.sigplan.org/authorInformation.htm
- http://www.ece.umn.edu/users/sachin/misc/writing.html
- http://vlsi1.engr.utk.edu/~bouldin/MUGSTUFF/NEWSLETTERS/DATA/1205.html
- http://www.cs.berkeley.edu/~pattrsn/talks/writingtips.html
- http://www.alg.ewi.tudelft.nl/masters.php
- http://www.st.ewi.tudelft.nl/~mathijs/writinglinks.html

7.9 Research Impact

Research impact:

1. Publish or perish: indicators of my research impact based on different metrics that measure my number of citations and publications; see http://www.harzing.com/pop.htm
2. scHolar index: http://insitu.lri.fr/~roussel/projects/scholarindex/index.cgi

7.10 Publications to Check Out

Publications that I am not able to get a PDF copy of, and **MUST** get hold of:

1. Miroslav N. Velev, “Formal Verification of Pipelined Microprocessors by Correspondence Checking”, Ph.D. thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, May 2004.

Publications that I am not able to get a PDF copy of, and **SHOULD** get hold of:

1. C.-J. Richard Shi, “Optimum Logic Encoding and Layout Wiring for VLSI Design: A Graph-Theoretic Approach,” Ph.D. Disseration, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, December 1993 – Thesis supervisor: J. A. Brzozowski; External Examiner: E. S. Kuh, UC Berkeley

7.11 Key Figures in Research

Professors and industrial research scientists on the design automation of heterogeneous systems:

- Giovanni De Micheli

7.12 Poor Journals and Conferences

Poor Conferences:

1. International Conference on Artificial Intelligence and Symbolic Computation (AISC)
2. DesignCon, for VLSI designers
3. International Association of Engineers (IAENG) Conferences: `http://www.iaeng.org/conferences.html`
4. Indian International Conference on Artificial Intelligence: `http://www.iiconference.org/`

Poor journals:

1. International Journal of Design, Analysis and Tools for Integrated Circuits and Systems:
   (a) digital, analog, mixed-signal and asynchronous design
   (b) processor, memory and RF design
   (c) DSP and FPGA/ASIC-based design
   (d) synthesis and physical design
   (e) ATPG, design-for-testability and built-in self test methodologies
   (f) embedded system hardware/software co-design and co-verification
   (g) CAD/EDA methodologies and tools
   (h) statistical timing analysis and low power design methodologies
   (i) network and system on-a-chip and applications
   (j) communication and wireless sensor networks (WSNs)
   (k) specification languages: SystemC, SystemVerilog and UML
   (l) theory and foundations: model checking, SAT-based methods, use of PSL, compositional methods and probabilistic methods
   (m) applications of formal methods: equivalence checking, CSP applications and transaction-level verification
   (n) verification methods based on hardware description/system-level languages (e.g. VHDL, SystemVerilog and SystemC)

(o) industrial experience reports and case studies
(p) real-time, hybrid and embedded systems
(q) all areas of modelling, simulation and verification of systems
(r) formalisms: process algebras, petri-nets, automaton theory and BDDs
(s) software engineering (including real-time Java, real-time UML and performance metrics)
(t) reversible computing
(u) http://datics.nesea-conference.org/IJDATICS/call_for_papers.html

7.13 Help with Translation

Help with translation:

1. Google Translate: www.google.com/translate
2. WordReference Dictionaries (WordReference.com):
   (a) http://www.wordreference.com/
   (b) Has bilingual dictionaries, including Italian-English and English-Italian dictionaries.
3. WorldLingo Translations LLC: http://www.worldlingo.com/

7.14 Engineering Resources

Engineering resources:

1. tutorials on different technologies in telecommunications, computer networking, consumer electronics, and computing; see http://www.iec.org/online/tutorials/
2. Information about VLSI design, EDA tools, open source CAD tool flow, general information about electrical engineering (EE), EE jobs in VLSI design, ... See http://jonahprobell.com/bookmarks.html

7.15 Try Block

Try block:

1. I will say of the Lord, “He is my refuge and my fortress, my God, in whom I trust.” Psalm 91:2 $\Longrightarrow$ CS Prof. Wai-Kei Mak @ NTHU, physical design and metal fill synthesis; see http://www.cs.nthu.edu.tw/~wkmak/ ... http://140.114.75.33/nthu_cs_lab223.pdf ... http://140.114.75.33/
2. functional verification of SoC, noise and thermal analysis, power gating, and “Delay variation tolerance design and design automation”; http://www.cs.nthu.edu.tw/~scchang/

8 No Exponential is Forever: But “Forever” can be Delayed!

Gordon Moore, “No exponential is forever: but “Forever” can be delayed!,” Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC), vol.1, pp. 20-23, 2003

no exponential is forever... but we can delay “forever”

No exponential is forever: but “Forever” can be delayed!

“If we knew what it was we were doing, it would not be called research, would it?”
– Albert Einstein

Things to do (remove this list when tasks are completed):

1. check if I have viewed:
   (a) http://www.entasys.com/
   (b) http://www.coware.com/
   (c) http://www.ligasystems.com/
   (d) http://www.eve-team.com/index.php
   (e) http://www.achronix.com/
   (f) http://www.nusym.com/
   (g) http://www.pulsic.com/
   (h) http://www.azuro.com/
   (i) http://www.teklatech.com/
   (j) http://physware.com/
   (k) sisoft.com
   (l) http://www.cst.com/
   (m) http://www.spatial.com/
   (n) http://www.gradient-da.com/home/index.htm
   (o) Silicon Frontline; see http://www.siliconfrontline.com/
   (p) http://www.pextracorp.com/
   (q) http://www.legenddesign.com/
   (r) http://www.accit-newsystemsresearch.com/
   (s) http://www.accelicon.com/index.shtml
   (t) http://www.apache-da.com/
   (u) http://www.prolificinc.com/
   (v) http://www.atrenta.com/
   (w) http://www.sagantec.com/
   (x) http://www.takumi-tech.com/
   (y) http://www.sequencedesign.com/
   (z) http://www.shearwater.com/about.html
2. see SMT related conferences and workshops to determine who is working on SMT-based VLSI formal verification in Europe
3. add all BibTeX entries from this document into my BibTeX database
4. check out opportunities (Ph.D. positions) at **SAT Live!** and **ElectronicSystemLevel.com forum ⇒ ESL Jobs available**
5. Game Plan:
   (a) Close all tabs in all web browsers
   (b) Sort papers from “dump folders”
   (c) Conferences to check out:
      - ACM
      - SIGDA conferences DAC ASP-DAC ICCAD DATE

- IEEE D&T, TCAD (T. CAD),TCASS I & II TC (T. Computing)
- IEEE CS
- ISPD
- IEEE International High Level Design Validation and Test Workshop / HLDVT
- FM200X (not in IEEE/ACM; it's in Springer-Verlag, I reckon)
- MEMOCODE
- CODES-ISSS ... CODES/ISSS: IEEE/ACM/IFIP International CODES/ ISSS Merged Conference
- CAV
- FMCAD
- GLSVLSI
- IEEE/ACM/IFIP international conference on Hardware/software codesign and system
- kroening.com (add to VLSI formal verification section)
- ICCD
- IEEE Circuits and Systems Magazine
- Cadence Research Labs
- Synopsis Advanced Technology Group
- IBM Research in the US
- SLIP: International Workshop on System Level Interconnect Prediction
- TAU: ACM/IEEE International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems
- FPGA: ACM/SIGDA International Symposium on Field Programmable Gate Arrays
- ISLPED: International Symposium on Low Power Electronics and Design
- MEMOCODE: ACM/IEEE Conference on Formal Methods and Programming Models for Codesign
- ISQED
- ACSD: International Conference on Application of Concurrency to System Design
- IWLS: International Workshop on Logic and Synthesis
- PATMOS power verification
- ISCAS
- ISSSC
- Hot Interconnects
- Hot Chips
- IEEE/ACM International Conference on Compilers, Architectures, and Synthesis for Embedded Systems (CASES)
- IEEE International SOC Conference (SOCC)
- Conference on Correct Hardware Design and Verification Methods - CHARME
- IEEE Computer Society Annual Symposium on VLSI
- BMAS / IEEE Behavioral Modeling and Simulation Conference
- IEEE Symposium on Computer Arithmetic
- Kuen-Yu Tsai — NTU DFM
- SIGDA / ACM / EDAA Ph.D. Dissertation Award winners — DONE

(d) see §**??** for more information on DFM

(e) retrieve information about how to write the abstract, introduction section, and other notes on technical writing from my "Masters" thesis project ... And, collate the information in the comments of my LaTeXdocuments that were meant to be research proposals for internships in Taiwan ... And, also collate the information from the **Research** chapter in this document, including the information in the sections **Random Stuff** and **Academic Stuff** ... Update the **Computing** chapter with information in my research proposal section; keep the research proposal section as a summary of things I ought to be doing in research

(f) Write paper, and google additional refs

6. write my "Research Blurb" for my FB profile

7. use GNUMail for PGP/GPG encryption; see http://www.catrene.org/web/calls/send_pp.php
8. buy the following ebooks:
   (a) http://pragprog.com/titles/cfcar2/the-passionate-programmer — should be available on ACM's 24×7 bookshelf or Safari bookshelf: CHECK!!!
   (b) http://pragprog.com/titles/tpp/the-pragmatic-programmer
9. check out `Grand Central Dispatch`, which is a new feature of `Mac OS® X 10.6 Snow Leopard`
10. check out:
    (a) http://en.wikipedia.org/wiki/Disk_encryption_software
    (b) http://www.dmoz.org/Computers/Security/Products_and_Tools/Cryptography/File_Encrypti
    (c) http://en.wikipedia.org/wiki/Comparison_of_disk_encryption_software
    (d) http://en.wikipedia.org/wiki/Encryption_software
11. register a new domain name:
    (a) domain name registration
12. update web browsers, text editors (XCode and TeXShop), BibDesk, OS
13. add chapter on globalization; put the ACM report on globalization in my BibTeX database; see http://www.acm.org/globalizationreport/
14. Apply for the DAC Engineering scholarship
15. learn these:
    (a) Clojure
    (b) Cilk
16. correct the referencing/labeling of figures and tables for the report format in LaTeX; the figure/table number in the caption does not match with that of the in-text reference; Debug this
17. join USC / uni Adelaide on LinkedIn
18. See http://anna.fi.muni.cz/yahoda/ for a list of formal verification (mostly model checking) tools.

Shift-Command-R brings up the console in the GDB debugger Control-Option-Command-R clears the console in the GDB debugger

cole centrale de Lille Ecole Nationale Suprieure des Tlcommunications de Bretagne Ecole Nationale Suprieure des Tlcommunications de Paris Ecole Natl Vet Toulouse cole Normale Suprieure Ecole Polytechnique Ecole Polytechnique de Montreal or Polytechnique Montral I would be working on VLSI formal verification on an industry-sponsored project. ParisTech the Paris Institute of Technology France (Ecole Poly + INRIA + Verimag) + cole Normale Superieure (France) + loria.fr/ ... Grandes coles

Institut de Logique, CdRS http://www.unine.ch/cdrs/ LORIA-INRIA-Lorraine / http://www.loria.fr/~ranise/ OFFIS, Oldenburg, Germany

Paris-Sud University ParisTech TIMA http://tima.imag.fr/tima/fr/index.html – France tu-chemnitz.de / TU Chemnitz Technische Universitt Dresden / TU Dresden / Technical University of Dresdens

Technische Universitt Hamburg-Harburg / TU Hamburg-Harburg TU Ilmenan TU Ilmenau Poznan University of Technology / TU Poznan??? Univ. of Glasgow University of Jyvaskyla University of LaPlata Universidad de La Rioja, Spain Universit Lille Nord de France - University of Science and Technology of Lille (USTL) U. of Littoral U. of Ljubljana U. of Luebeck University of Malaga / University of Málaga U. of Montpellier II University of Oulu University of Paris-Sud University of Paris VI: Pierre et Marie Curie University of Peloponnisos Univ. Perpigan

Universit Pierre et Marie Curie, Prof. Greiner, SPIN U. Pisa / Universita di Pisa / University of Pisa Universitat Politcnica de Catalunya / Universidad Politecnica de Catalunya / UPC http://www.lsi.upc.es/~roberto/ ... http://petrus.upc.es/US/sistemas.html – UPC, Spain University Politecnica de Valencia / UPV Valencia, Spain / Universidad Politecnica de Valencia DISCA U Poznan University of Porto University of Potsdam Universit di Reggio Calabria, Reggio Calabria, Italy U. Rome / University of Rome, Tor Vergata / University of Rome, La Sapienza / u roma / rome / Universit di Roma La Sapienza, Italy / w3.uniroma1.it/ UPS Toulouse University of Saarbrcken University of Salamanca University of Salerno University of Salzburg Universita di SantAnna Univ. Santiago de Compostela University of Seville / U. Sevilla, Spain??? University of Sibiu http://www.uni-siegen.de/ / Univ. Siegen University of Siena Louis Pasteur University of Strasbourg Universitaet Stuttgart / uni-stuttgart.de / University of Stuggart GOOD Universit della Svizzera Italiana Universit de Technologie de Compigne, Universit Telematica Internazionale UNINETTUNO / uninettunouniversity University of Turin / Uni Torino / University of Torino University of Toulouse, France UTRC Universitt Trier University of Trieste University of Tuebingen / University of Tubingen??? University of Twente (or Twente University) U. Udine / Universita di Udine University of Ulm / http://www.uni-ulm.de/in/theo/mitarbeiter/toran.html Uni Valencia / Universidad de Valencia (Spain) U. of Valenciennes Universit degli Studi di Verona / University of Verona Univ. Vienna / University of Vienna mescalin.tbi.univie.ac.at University of Wroclaw Bergische Universitt Wuppertal, Germany / University of Wuppertal, Germany http://maveric0.uwaterloo.ca/ brzozo/ Department of Informatics and Telecommunications Engineering, University of Western Macedonia, Karamanli & Ligeris, Kozani 50100, Greece http://www.wu-wien.ac.at University of Wrzburg / University of Wurzburg University of Zaragoza / Saragossa University Utrecht University Wilfrid Laurier Univ Wageningen Univ Warsaw University http://www.uc3m.es/portal/page/portal/inicio http://www.uclm.es/ http://www.uib.es/ca/ http://www.vtt.fi/vtt/index.jsp

http://www.mit.edu/people/cdemello/it.html ... list of colleges

http://www-syscom.univ-mlv.fr/ vignat/Signal/ http://www.esiee.fr/ http://www.insa-rennes.fr http://www.ece.fr http://efrei.fr http://www.eigsi.fr http://www.eisti.fr/ http://www.ensea.fr http://www.enseeiht.fr http://www.univ-valenciennes.fr/ensiame/ http://www.ensi-bourg fr http://www.ensiie.fr/ http://www.ensil.unilim.fr http://www.ensisa.uha.fr/ http://www.ensm-douai.fr http://www.ensma.fr http://www.ens2m.fr http://www.mines.inpl-nancy.fr/ http://www.epf.fr http://www.cpe.fr http://www.esiea.fr http://www.esiee.fr/ http://www.esiee-amiens fr/ http://www.esigetel.fr http://www.esme.fr http://www.hei.fr http://www.ifips.u-psud.fr http://www.insa-rennes.fr http://www.insa-rouen.fr http://www.insa-strasbourg.fr/ http://www.insa-toulouse.fr http://www.isep.fr/ http://www.isima.fr http://www.loria.fr/~ranise/ http://www.lri.fr/equipes/asspro/ http://phelma.grenoble-inp.fr/ http://www.polytech.ujf-greno fr http://www.polytech-lille.fr/ http://www.polytech.univ-mrs.fr http://www.polytech.univ-mont fr http://www.polytech.univ-nantes.fr http://www.sciences.univ-nantes.fr/info/recherche/mgl/membres.html http://www.polytech.unice.fr/ http://www.univ-orleans.fr/polytech http://www.polytech.upmc.fr/ http://www.polytech.univ-savoie.fr http://www.polytech.univ-tours.fr http://www.supelec.fr http://www.supmeca.fr/ http://www.enst-bretagne.fr http://www.telecom-st-etienne.fr/ Universit della Calabria / University of Calabria Martin-Luther-Universitt Halle Bergische Universitt Wuppertal Universidad Carlos III, Madrid, Spain University of Catania University of Toulouse, France University of Florence, Italy University of Wuppertal University of Modena Reggio Emilia, Italy

workshop Formal Verification of Analog Circuits (FAC) '09 : University of Frankfurt; CNRS-Verimag, Grenoble; UJF-Verimag, Grenoble; University of Freiburg; Concordia University; Queen Mary University; University of Oldenburg; CNRS-Verimag, Grenoble, France; University of British Columbia

use "label:muted" to search for muted emails in Gmail check alicia's address, so that I can mail her chocolates ask Prof. Jordi Cortadella from UPC: how can I join your lab as a Ph.D. student? addendum

Recession Compiler Recession Compiler Ultra In God we trust, all others bring data the law of conservation of talent / the law of talent conservation for the EDA industry

components of SAT solvers:

1. parser
2. CNF translator
3. DPLL(X) engine
4. solver for EUF
5. solver for difference logic

Model Order Reduction

For conference papers, when indicating their booktitles in BibTeX, try: booktitle (acronym of conference) Do not place a comma in between the booktitle and the acronym of the conference

use $$ to enclose math, science, and engineering symbols, including variables within the text use italics only for text, which do not denote symbols

IEEE Press IEEE Computer Society Press ACM Press

Is this an undecidable problem? If not, it is decidable. Then, is it intractable (or NP-complete)? If so, reduce it to 3-SAT. Else, this problem should be reasonably doable.

http://www.fi.muni.cz/usr/brim http://www.fi.muni.cz/paradise ParaDiSe Parallel and Distributed Systems Laboratory I suppose you mean parallel formal verification. This is our long-term topic and I expect to continue working in this area. 3 topics in parallel formal verification The first is about using graphical cards to speed-up formal verification, the other one is on formal verification of specific micro-controllers ... Our colleagues here develop software for specific processors. We hope the size of the state space is much smaller than in general software, hence amenable to brute force verification. implementing (parts of) verification tool on programmable hardware — implement part of a formal verification algorithm/tool in an FPGA

Does your research lab only seek software or hardware solutions to formal verification using parallel computing? Typically yes, we also explore using hard-disks. This is where is our expertise

tell Prof. abbott,

breakdown of my VLSI formal verification research in SMT solvers, and categories of case studies (datapath + control module + memory module + interconnect network) datapath + control module + memory module + interconnect network (on-chip busses, mesh interconnect structures, NoCs) + I/O circuitry Rabaey03, pp. 379-380 Rabaey03, pp. 439 high-level synthesis = behavioral synthesis = architecture synthesis

The correctness of a system can be explored by scanning all possible executions using symbolic representations of the state space. However this method will not scale up with the size and complexity of the system. Furthermore, though model checking can find potential bugs, it can not provide proofs of correctness.

Godiva chocolates

us travel advisory Current Travel Warnings

bellwether It is considered a bellwether for the American economy

sep 22, NUH ENT at 11 am – get note from them before I leave singapore First Place: Universitat Politecnica de Catalunya, Marc Galceran and Dmitry Bufistov Second Place: National Taiwan University, Kai-Fu Tang and Chi-An Wu

Referee ID number: 1 Salutation: Name: Affiliation: Address: Postal Code: Country: Email Address: Phone Number: Home Page:

email to Blake

dottorato di ricerca (doctorate of research) in Information and Communication Technology (ICT) Dottorato di ricerca

References

[1] Phil Agre. Advice for undergraduates considering graduate school. ACM Crossroads, 3(4):24–27, May 1997.

[2] Anne Curtis. Word Smart for the GRE: A Guide to Perfect Usage. The Princeton Review, New York, NY, second edition, 2007. Updated by Judy Katz Unrein.

[3] Marie Ellen desJardins. How to succeed in graduate school: a guide for students and advisors: part I of II. ACM Crossroads, 1(2):3–9, December 1994.

[4] Marie Ellen desJardins. How to succeed in graduate school: a guide for students and advisors: part II of II. ACM Crossroads, 1(3):1–6, March 1995.

[5] Marie Ellen desJardins. A day in the life of ... Marie DesJardins. ACM Crossroads, 8(2):8–9, December 2001.

[6] Marie Ellen desJardins. How to succeed in graduate school: a guide for students and advisors. ACM Crossroads, 14(4):5–9, June 2008.

[7] Educational Testing Service. Practicing to Take the GRE General Test. Educational Testing Service, Princeton, NJ, tenth edition, 2007.

[8] Sharon Weiner Green and Ira K. Wolf. Barron's How to Prepare for the GRE Test: Graduate Record Examination. Barron's Educational Series. Barron's, Hauppauge, NY, fourteenth edition, 2000.

[9] Brian W. Kernighan and P. J. Plauger. The Elements of Programming Style. McGraw-Hill, New York, NY, second edition, 1982.

[10] Adrian Lee, Carina Dennis, and Philip Campbell. Nature's guide for mentors. Nature, 447(7146):791–797, June 14 2007.

[11] Karen Lurie. Cracking the GRE. Princeton Review Publishing, New York, NY, 2002.

[12] Scott Meyers. More Effective C++: 35 New Ways to Improve Your Programs and Designs. Addison-Wesley Professional Computing Series. Pearson Education, Boston, MA, 1996.

[13] Scott Meyers. Effective STL: 50 Specific Ways to Improve Your Use of the Standard Template Library. Addison-Wesley Professional Computing Series. Addison-Wesley, UNKNOWN, 2001.

[14] Scott Meyers. Effective C++: 55 Specific Ways to Improve Your Programs and Designs. Addison-Wesley Professional Computing Series. Pearson Education, Upper Saddle River, NJ, third edition, 2005.

[15] Rachel Pottinger. Choosing a Ph.D. program in computer science. ACM Crossroads, 6(1):6–13, September 1999.

[16] Mark Alan Stewart. GRE CAT: Answers to the Real Essay Questions. Peterson's, Lawrenceville, NJ, second edition, 2003.

[17] Bjarne Stroustrup. The C++ Programming Language. Addison-Wesley, Boston, MA, special edition, 2000.

[18] The Staff of Kaplan Test Prep and Admissions. GRE and GMAT Exams Writing Workbook. Kaplan, New York, NY, third edition, 2008.

[19] University of Michigan EECS Department. Electrical engineering program: Guidelines for the qualifying oral examination. Available online in *EECS Graduate Programs* at Department of Electrical Engineering and Computer Science, College of Engineering, University of Michigan: `http://eecs.umich.edu/eecs/graduate/ee/qual.pdf`; July 18, 2013 was the last accessed date, November 1999.

[20] Benjamin Wells. The Best Test Preparation for the GRE Computer Science Test. Research & Educational Association, Piscataway, NJ, fifth edition, 2005.

[21] Cathy Wendler, Brent Bridgeman, Fred Cline, Catherine Millett, JoAnn Rock, Nathan Bell, and Patricia McAllister. The path forward: The future of graduate education in the united states. Technical report, Educational Testing Service, Princeton, NJ, 2010.

[22] Yung-Yee Wu. Verbal Workout for the GRE. The Princeton Review, New York, NY, third edition, 2007.