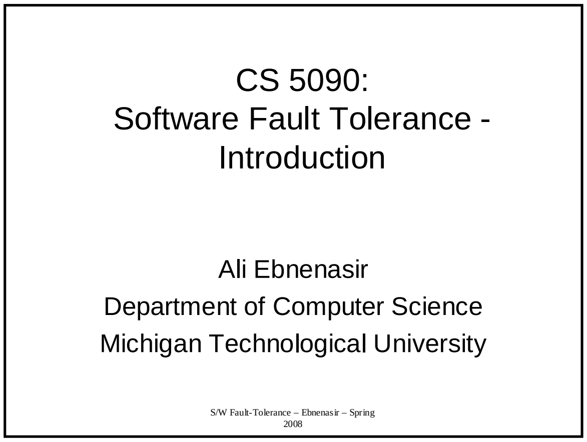
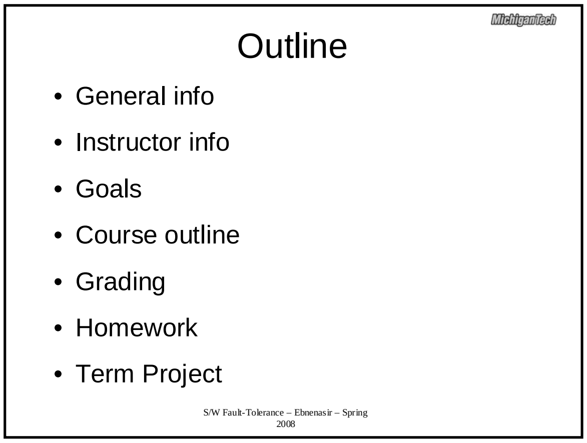


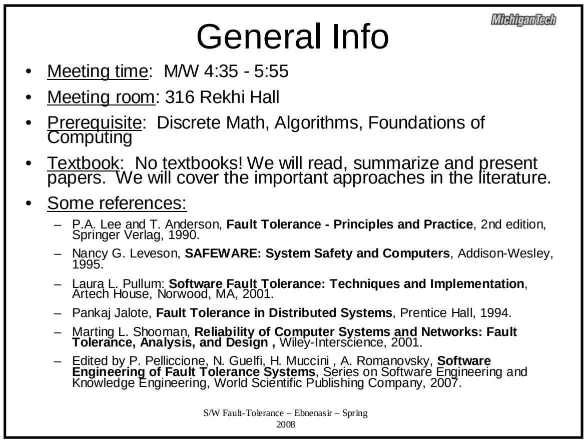
CS 5090: Software Fault Tolerance - Introduction

Ali Ebnenasir
Department of Computer Science
Michigan Technological University



Outline

- General info
- Instructor info
- Goals
- Course outline
- Grading
- Homework
- Term Project



General Info

- Meeting time: M/W 4:35 - 5:55
- Meeting room: 316 Rekhi Hall
- Prerequisite: Discrete Math, Algorithms, Foundations of Computing
- Textbook: No textbooks! We will read, summarize and present papers. We will cover the important approaches in the literature.
- Some references:
  - P.A. Lee and T. Anderson, **Fault Tolerance - Principles and Practice**, 2nd edition, Springer Verlag, 1990.
  - Nancy G. Leveson, **SAFEWARE: System Safety and Computers**, Addison-Wesley, 1995.
  - Laura L. Pullum: **Software Fault Tolerance: Techniques and Implementation**, Artech House, Norwood, MA, 2001.
  - Pankaj Jalote, **Fault Tolerance in Distributed Systems**, Prentice Hall, 1994.
  - Marting L. Shooman, **Reliability of Computer Systems and Networks: Fault Tolerance, Analysis, and Design ,** Wiley-Interscience, 2001.
  - Edited by P. Pelliccione, N. Guelfi, H. Muccini , A. Romanovsky, **Software Engineering of Fault Tolerance Systems**, Series on Software Engineering and Knowledge Engineering, World Scientific Publishing Company, 2007.

Instructor Info

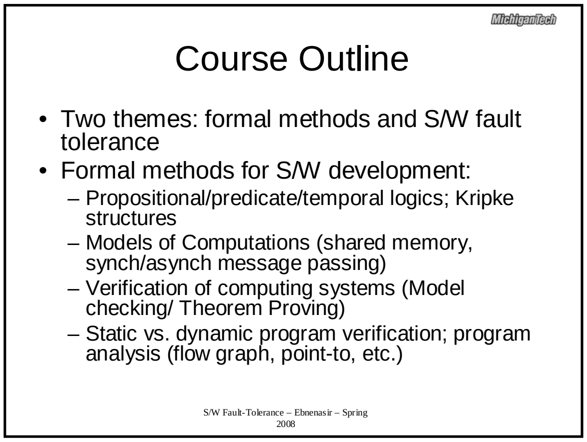
- Name: Dr. Ali Ebnenasir
- Office: 206 Rekhi Hall
- Phone: 487-4372
- E-mail: aebnenas@mtu.edu
- Office Hours: by appointment

Motivations

- Software plays a role in almost every aspect of our lives
- Software failure could lead to loss of life, property, and could damage critical infrastructures (e.g., August 2003 black out)
- Need to educate S/W developers who systematically consider S/W failures
- Introduce some open problem in the field

Goals

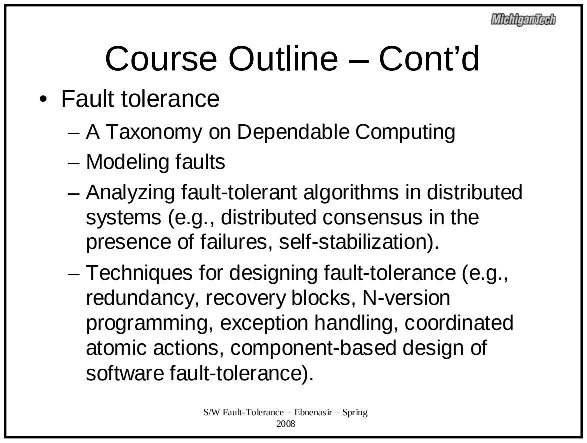
- Learn basic concepts of faults, fault tolerance, methods for developing robust S/W in general
- Learn formal methods for modeling, analysis and design of S/W fault tolerance
- Gain hands-on-experience with
  - automated analysis tools, and
  - implementation techniques
- Experience why it is difficult to develop robust programs



Course Outline

- Two themes: formal methods and S/W fault tolerance
- Formal methods for S/W development:
  - Propositional/predicate/temporal logics; Kripke structures
  - Models of Computations (shared memory, synch/asynch message passing)
  - Verification of computing systems (Model checking/ Theorem Proving)
  - Static vs. dynamic program verification; program analysis (flow graph, point-to, etc.)

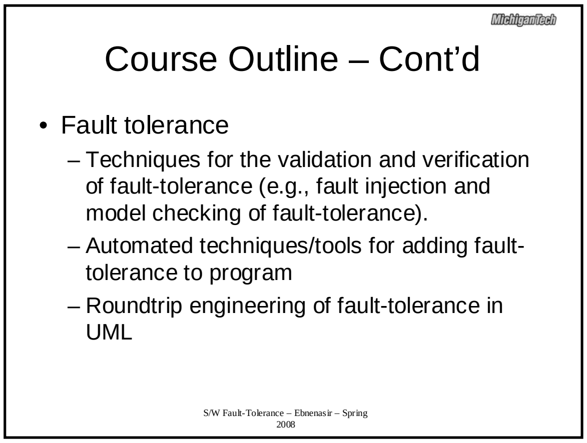
S/W Fault-Tolerance – Ebnenasir – Spring 2008



Course Outline – Cont'd

- Fault tolerance
  - A Taxonomy on Dependable Computing
  - Modeling faults
  - Analyzing fault-tolerant algorithms in distributed systems (e.g., distributed consensus in the presence of failures, self-stabilization).
  - Techniques for designing fault-tolerance (e.g., redundancy, recovery blocks, N-version programming, exception handling, coordinated atomic actions, component-based design of software fault-tolerance).

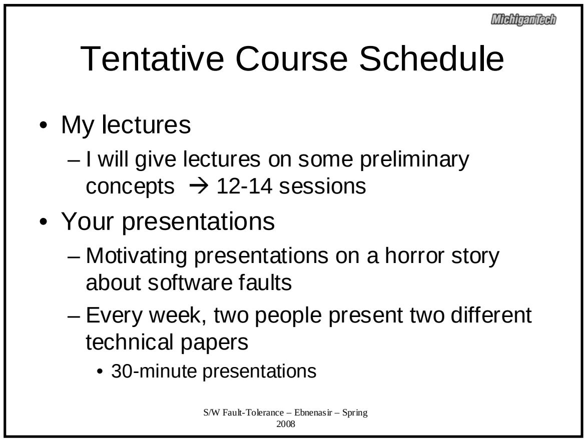
S/W Fault-Tolerance – Ebnenasir – Spring 2008



Course Outline – Cont'd

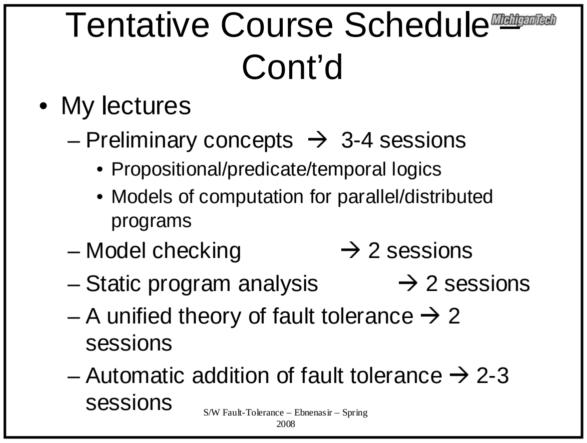
- Fault tolerance
  - Techniques for the validation and verification of fault-tolerance (e.g., fault injection and model checking of fault-tolerance).
  - Automated techniques/tools for adding fault-tolerance to program
  - Roundtrip engineering of fault-tolerance in UML

S/W Fault-Tolerance – Ebnenasir – Spring 2008



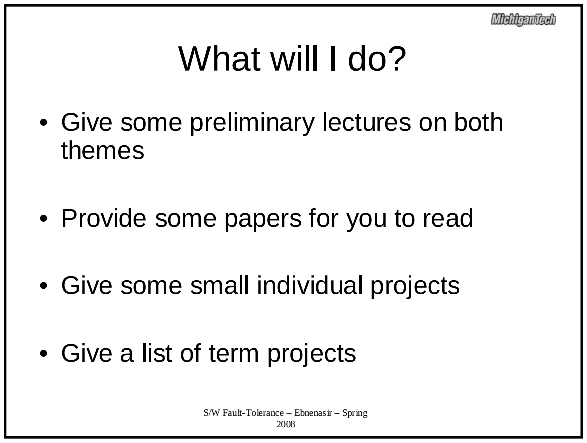
Tentative Course Schedule

- My lectures
  - – I will give lectures on some preliminary concepts → 12-14 sessions
- Your presentations
  - – Motivating presentations on a horror story about software faults
  - – Every week, two people present two different technical papers
    - 30-minute presentations



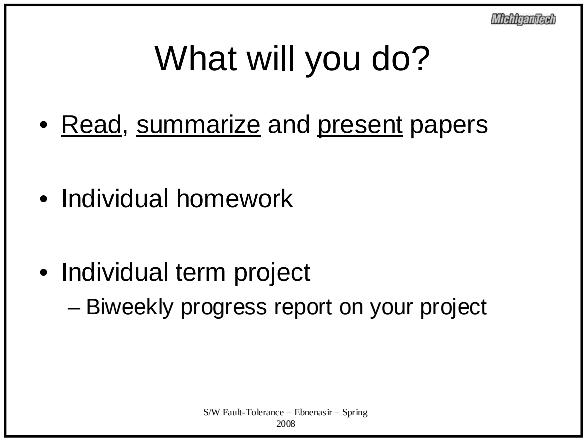
Tentative Course Schedule – Cont'd

- My lectures
  - – Preliminary concepts → 3-4 sessions
    - Propositional/predicate/temporal logics
    - Models of computation for parallel/distributed programs
  - – Model checking → 2 sessions
  - – Static program analysis → 2 sessions
  - – A unified theory of fault tolerance → 2 sessions
  - – Automatic addition of fault tolerance → 2-3 sessions



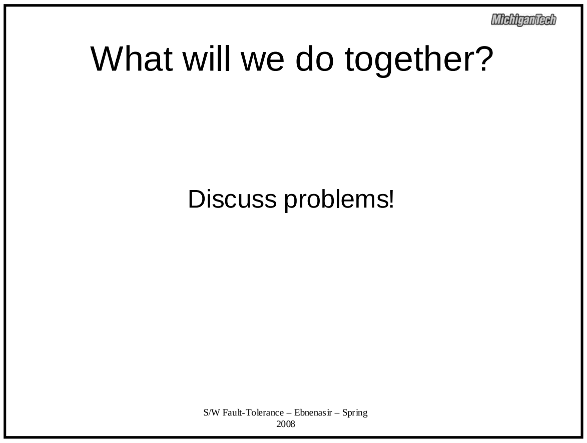
What will I do?

- Give some preliminary lectures on both themes
- Provide some papers for you to read
- Give some small individual projects
- Give a list of term projects



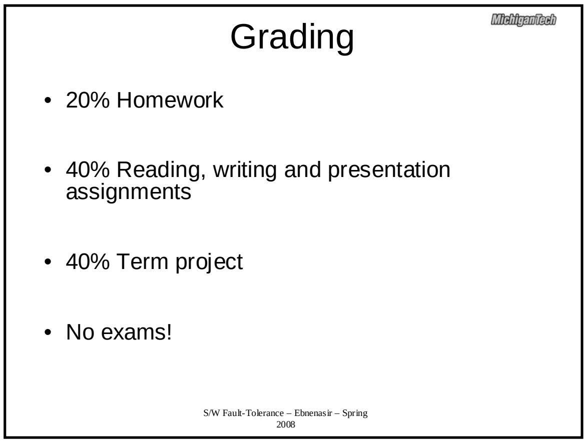
What will you do?

- Read, summarize and present papers
- Individual homework
- Individual term project
  - Biweekly progress report on your project



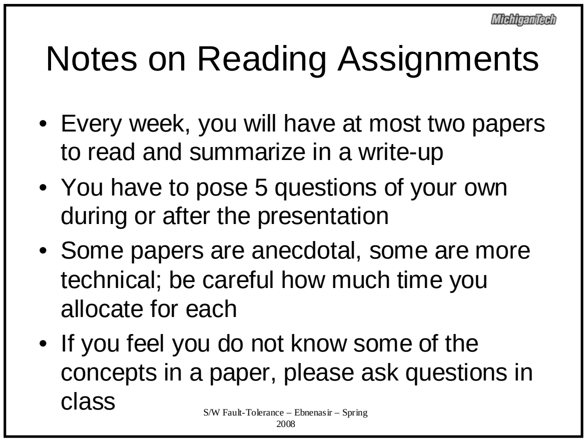
What will we do together?

Discuss problems!



Grading

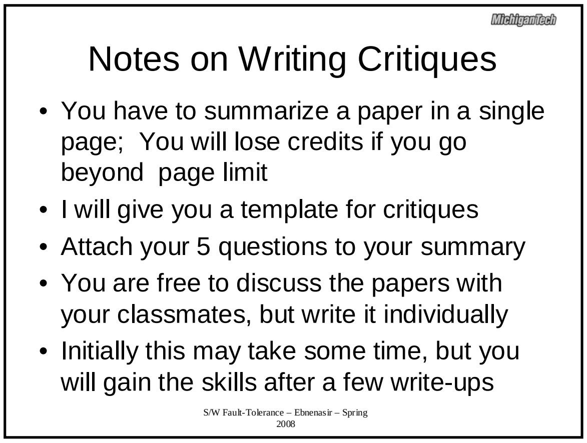
- 20% Homework
- 40% Reading, writing and presentation assignments
- 40% Term project
- No exams!



Notes on Reading Assignments

- Every week, you will have at most two papers to read and summarize in a write-up
- You have to pose 5 questions of your own during or after the presentation
- Some papers are anecdotal, some are more technical; be careful how much time you allocate for each
- If you feel you do not know some of the concepts in a paper, please ask questions in class

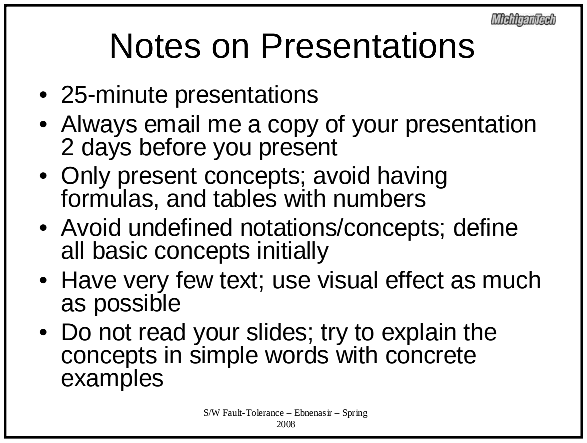
S/W Fault-Tolerance – Ebnenasir – Spring 2008



Notes on Writing Critiques

- You have to summarize a paper in a single page; You will lose credits if you go beyond page limit
- I will give you a template for critiques
- Attach your 5 questions to your summary
- You are free to discuss the papers with your classmates, but write it individually
- Initially this may take some time, but you will gain the skills after a few write-ups

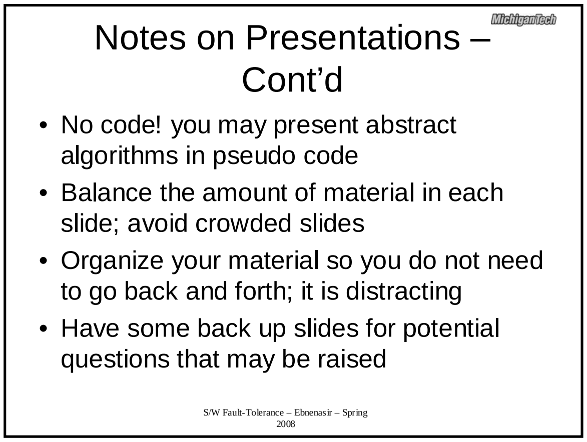
S/W Fault-Tolerance – Ebnenasir – Spring 2008



Notes on Presentations

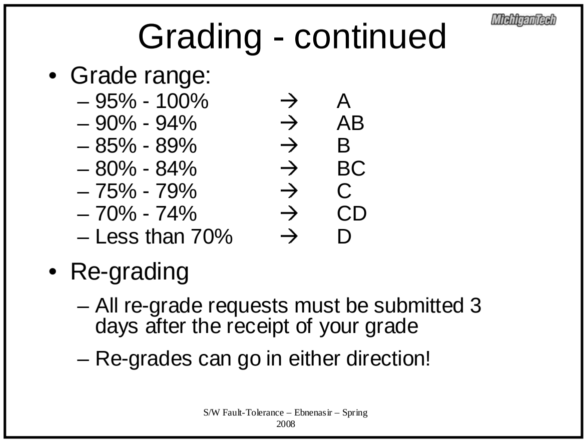
- 25-minute presentations
- Always email me a copy of your presentation 2 days before you present
- Only present concepts; avoid having formulas, and tables with numbers
- Avoid undefined notations/concepts; define all basic concepts initially
- Have very few text; use visual effect as much as possible
- Do not read your slides; try to explain the concepts in simple words with concrete examples

S/W Fault-Tolerance – Ebnenasir – Spring 2008



Notes on Presentations – Cont'd

- No code! you may present abstract algorithms in pseudo code
- Balance the amount of material in each slide; avoid crowded slides
- Organize your material so you do not need to go back and forth; it is distracting
- Have some back up slides for potential questions that may be raised

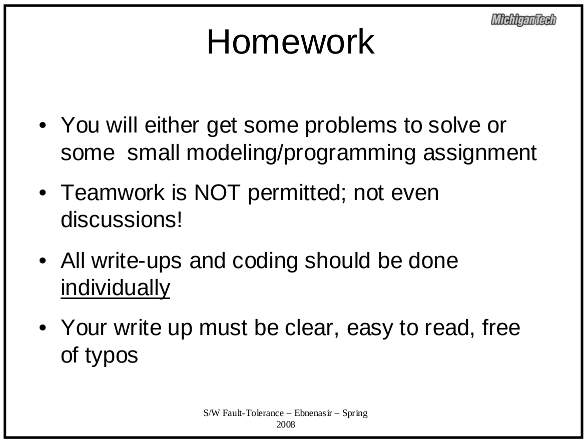


Grading - continued

- Grade range:

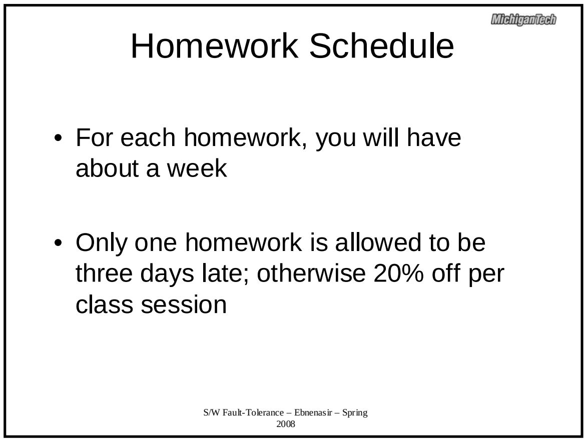
– 95% - 100%	→	A
– 90% - 94%	→	AB
– 85% - 89%	→	B
– 80% - 84%	→	BC
– 75% - 79%	→	C
– 70% - 74%	→	CD
– Less than 70%	→	D

- Re-grading
  - All re-grade requests must be submitted 3 days after the receipt of your grade
  - Re-grades can go in either direction!



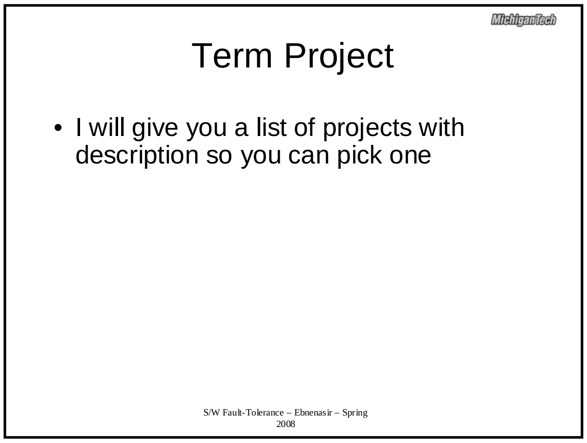
Homework

- You will either get some problems to solve or some small modeling/programming assignment
- Teamwork is NOT permitted; not even discussions!
- All write-ups and coding should be done individually
- Your write up must be clear, easy to read, free of typos



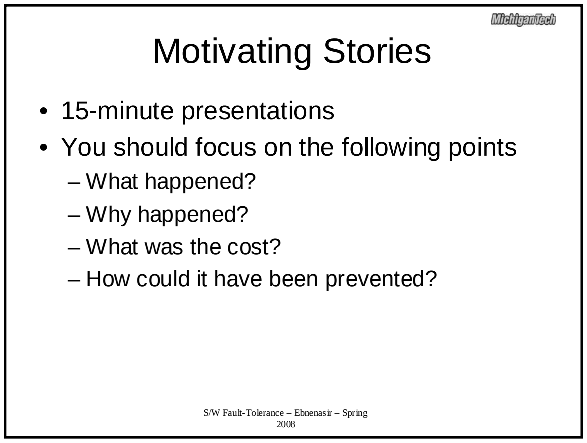
Homework Schedule

- For each homework, you will have about a week
- Only one homework is allowed to be three days late; otherwise 20% off per class session



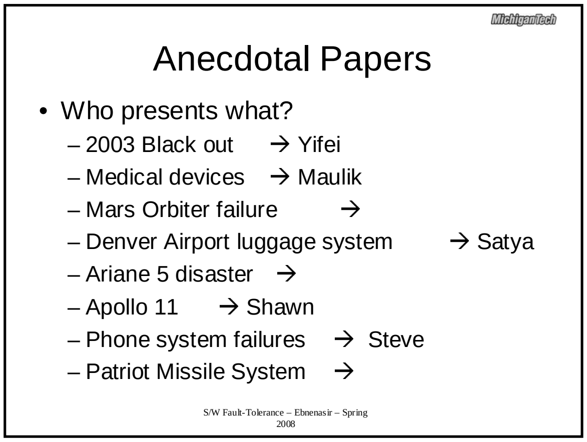
Term Project

- I will give you a list of projects with description so you can pick one



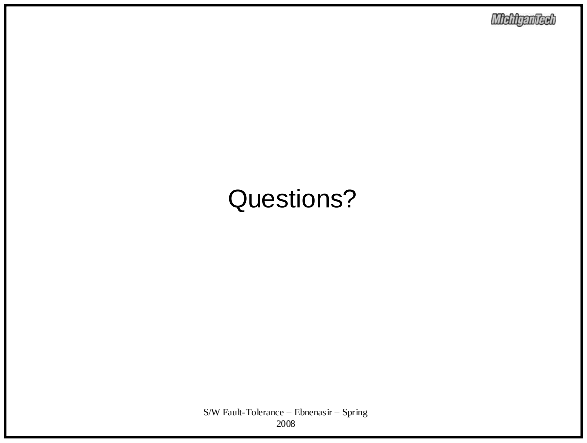
Motivating Stories

- 15-minute presentations
- You should focus on the following points
  - What happened?
  - Why happened?
  - What was the cost?
  - How could it have been prevented?



Anecdotal Papers

- Who presents what?
  - – 2003 Black out → Yifei
  - – Medical devices → Maulik
  - – Mars Orbiter failure →
  - – Denver Airport luggage system → Satya
  - – Ariane 5 disaster →
  - – Apollo 11 → Shawn
  - – Phone system failures → Steve
  - – Patriot Missile System →



Questions?