Projects for 6.975
Basics

• What are the projects?
  ***ONE OF THE following***
  – Programming the IBM/Sony/Toshiba Cell Processor
  – Programming the Tilera TILE64 Processor
  – Paper Design

• Who can do the project
  – If you are signed-up for 6.UAP → Paper Design project by yourself
  – Signed-up for letter grade
    – Paper design project (individual or two people)
    – Programming project (individual)
  – Listener → Talk to me
Programming Projects

• Build the fastest matrix multiply

• Scope
  – Rectangular matrices (n by 8n)
  – Data type
    – 32 bit float in Cell
    – 32 bit integer in Tilera
    – No overflow/underflow!
  – Assume input matrices are in memory
    – No duplication allowed
    – Can choose the distribution
  – But provide a function to stream-in the inputs
  – Results in memory
    – But, provide a function to stream-out the results
  – Find the biggest in-core matrix size (power of 2)
Programming Projects

• Build the fastest matrix multiply

• Timing
  – Multiple invoked, time measured within a single tile/PPU/SPU
    – Need to synchronize with that
  – Only multiplication operations
    – Start timer after data is streamed in
    – Finish timer before streaming it out
  – Performance report
    – Biggest in-core to 1/64th of that (1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64)
    – Gather
      – Run-time
      – FLOPs/OPs
      – Utilization per SPU/PPU/tile
      – Breakdown (computation, communication, overhead, blocked)
      – I/O time (streaming in/out)
Programming Projects

• Build the fastest matrix multiply

• Deliverables
  – Performance numbers
  – Speedup graphs
  – Initial Report (1 page)
    – Biggest chosen matrix size with justification
    – Brief description of what was done so far (with results)
    – Plan of action
  – Final Report (5 pages)
    – What was your approach?
    – What experiments did you do?
    – What worked and what did not?
    – Breakdown of the impact of different optimizations.
    – Qualitative analysis of your effort
    – Final code
Programming Projects

• Build the fastest matrix multiply

• Competition
  – We will choose a matrix size
    – If you did not get that size, you are out.
  – Fastest matrix multiply for that size
Programming Projects

• Build the fastest matrix multiply

• Timeline
  – Tilera tutorial: Wednesday, Oct 10th 7:00PM
  – Cell tutorial: Thursday, Oct 11th 7:00PM
  – Initial report: Nov 5th
  – Final report: Dec 10th
Paper Design Project

• What can be done to make parallel programming easier?

• A full implementation and evaluation of any of these can be a PhD thesis (or multiple theses)
  – So we don’t ask you to do that!
  – We want you to do the initial paper design
You need to

• Define a problem area
• Understand the landscape of related works in the area
• Come-up with a specific idea(s)
• Motivate/justify your solution
• Present your solution
Define a Problem Area

• **Scope:** What is the right size?
  – The parallel programming problem → too big
  – Parallelizing LU decomposition → too narrow

• **Domain**
  – A new programming model
  – A new language or language extension
  – A novel compiler optimization
  – A new tool
  – A novel architectural innovation
  – A combination of these
  – What else??
Related Works

• A LOT of work in this area over more than 4 decades!
  – Some papers are not electronically available
• Don’t reinvent the wheel
• Learn what has done
  – May figure-out it is already proposed
  – May inspire you to do something new
  – May find a better, novel use of an existing technology

• How to start?
  – Google (Google scholar)
  – Look-up premier conferences
  – Ask the Instructors, TA, friends, etc
  – Look at citations of papers and follow the paper trail

• Separating junk from gems
  – Read only the abstract
  – Fast scan
  – Look the citation index for popularity
Come-up with a Specific Idea

• Dig into your own experience
  – As a frustrated programmer
  – As an expert in an area (applications, compilers, languages, tools)

• Define a problem
  – Narrow the scope

• Come-up with a broad solution

• Break the solution into components, Expand on them
Motivation/Justification

• Importance
  – State the problem, give examples
  – Clearly describe what the world will be like with your solution

• Feasibility
  – What are all the components to the solution?
  – Show that they are doable
    – Implement a quick prototype if needed
  – Or describe any hard problems that need to be solved

• Novelty
  – Compare your solution to the closest related works
    – What are the differences?
Presentation

• Paper
  – ACM Conference style 10 page (two column) paper.
  – Lookup your favorite conference proceedings for an example
  – Clearly describe what is already done and what needs to be invented.
  – Try to use less hype and be scholarly

• Presentation
  – 15 minute conference style presentation with 5 minutes of questions
  – Use slides

• Talk to Instructor/TA if you haven’t given a conference presentation or written a conference paper before
Deadlines

• Need to have general area by Oct 12\textsuperscript{th}
  – Send mail to instructors/TA if you need guidance

• Initial meeting (15 minutes Oct 15\textsuperscript{th} & 18\textsuperscript{th} AM, sign-up)
  – Bring a one-page bullet-form outline with a
    – Problem statement
    – Summary of related work
    – Initial ideas
  – A two page Proposal (due Nov 2\textsuperscript{nd})
    – Problem Statement
    – Your Idea (expanded into sub ideas)
    – Detailed related works comparison
    – Action plan to expand the idea, motivation and justification

• Proposal meeting (Nov 5\textsuperscript{th} to 16\textsuperscript{th})
• Final Paper (Dec 3\textsuperscript{rd})
• Presentation (Dec 10\textsuperscript{th})