# Many Cores Reading Group

Organization Meeting
January 26, 2009

# Explore programming languages with high-level concurrency

- The multicore revolution
  - Will happen (a new realization of Moore's Law)
  - Will require extraordinarily-skilled programmers
- To avoid a massive skill shortage, try hiding some of the complexity in the language
  - Give up some potential
  - Enable wider access to what is provided

# Group facilitators

- Roy Lowrance (lowrance@cs.nyu.edu)
- Alexander Rubinsteyn (ar1738@nyu.edu)
- Dennis Shasha (shasha@courant.nyu.edu)
- Denis Zorin (dzorin@mrl.nyu.edu)

# Hardware trends driving parallelism

| From: | To: |
|---|---|
| 1. Power free, transistors expensive | 1. Transistors free, power expensive |
| ٢. Mostly concerned about dynamic power | 2. Concerned about all power |
| ٣. Silicon reliable, pins unreliable | 3. Silicon also unreliable |
| ٤. Size of hardware design can be increased | 4. Large designers harder to realize if <= 65 nm |
| ٥. Prototype chips easy to build | 5. Harder to build |
| ٦. Latency and bandwidth improve in step | 6. Bandwidth improves at least as the square of latency (in many technologies) |
| ٧. Multiply slow, load/store fast | 7. Multiply fast, load/store slow |
| ٨. Easily discovered instruction-level parallelism | 8. Diminishing returns in search for more |
| ٩. Uniprocessor performance doubles every 18 months | 9. Doubling may now take 5 years [IBM: slower] |
| ١٠. Don't parallelize app, just wait for faster sequential performance | 10. The wait may be very long |
| ١١. Improve processor performance by increasing clock frequency | 11. Improve by increasing parallelism |
| ١٢. Less than linear scaling when adding processors is a failure | 12. Any speedup is a success |

Source: [Asanovic-06: A View from Berkeley]

# “manycore” defined

- “thousands of processors on a chip”
  - “A View from Berkeley”, 2006

# manycore existence proof

- In 2006, Cisco shipped a network processor
  - 188 cores
  - 130 nm technology
  - 18 mm x 18 mm, 35W, 250 MHz clock
- Calculating assuming better technology
  - At 45 nm: 725 cores
  - At 30 nm: 1504 cores

Stop Press: “Cisco plans to release a [generalized server] product that threatens to shake up the technology industry” - NY Times, January 20, 2009

Source: [Asanovic-06: A View from Berkeley]

# Topics to prime your thinking

- Transactional memory
- Full-empty bits on memory words
- Shared memory schemes
- Message passing schemes
- Array languages
- X10, Fortress, … (recent designs)
- Non-array languages
- Courant language-use satisfaction survey

# Potential process

- Meet every Monday
  - 12:45
  - Rm 1221
  - 719 Broadway
- One person to present
  - Discuss and volunteer to Roy or Alexander
  - Present 30 minutes (<= 15 slides)
  - Discuss 30 minutes
  - Post presentations on web site

# Agenda

| Date | Topic/Presenter |
|---|---|
| 1/26 | Intro: Alexander, Roy |
| 2/2 | Parallel Haskell: Alexander |
| 2/9 | |
| 2/16 | |
| 2/23 | |
| 3/2 | |
| 3/9 | |
| 3/16 | |
| 3/23 | |
| 3/30 | |
| 4/6 | |
| 4/13 | |
| 4/20 | |
| 4/27 | |
| 5/4 | Wrap up, next steps:Alexander, Roy |

# Remaining agenda for today

- Sign the interest sheet:
  - Name
  - Email
  - Possible topic
- Volunteers?
- Spread the word
- Use the list server to sign up for announcements: www.cs.nyu.edu/mailman/listinfo/manycores
- Other

# EXTRA SLIDES

# Positive trends

- Economical to put thousands of simple processors on a single chip
- Communications among chips to have “very low latency and very high bandwidth”
- Software stack can evolve rapidly via open source

Source: [Asanovic-06: A View from Berkeley]

# Why smaller processors are better

- Parallelism is energy-efficient in achieving performance
- Many small cores give highest performance per unit area for parallel codes
- More smaller cores afford more opportunity to dynamically alter voltage
- Small components are easier to shut down and bypass when they fail
- Small elements are easier to design and verify and model

Source: [Asanovic-06: A View from Berkeley]

# Alternative existence proof

file:///Users/roy/Documents/00-inbox/IDFSp06-tera-scale%20headline.jpg

Source: http://techresearch.intel.com/articles/Tera-Scale/1421.htm

# “and” not “or”

- Programmer productivity

- Execution efficiency