EECS 1, Lecture 5

Computer Programming

Rainer Dömer

doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

Computers

Zuse Z3, 1941 (Wikipedia)

- Over the last few decades, nothing has transformed our lifes as much as the invention of the computer!
- What is a Computer?
  - *"A computer is a general purpose device that can be programmed to carry out a finite set of arithmetic or logical operations."* [Definition by Wikipedia]
- Computer Components
  - Micro-Processor, or Central Processing Unit (CPU)
    - Billions of operations per second, 100% accurate!
  - Memory
    - Practically unlimited amount of (digital) information!
  - I/O Devices
    - Practically unlimited applications!

Computers

- General Purpose


Source: 2jdata.com


Source: energy.gov


Source: wikipedia.org

- Special Purpose: Embedded Systems

Source: airlinersillustrated.com


Source: miele.com


Source: sony.com


Source: bmw.com


Source: hitachi-rail.com


Source: www.trouper.com

- *Only 2% of all processors are in workstations, 98% are found in embedded systems! [Netrino 2005]*
- *Computers are ubiquitous, omnipresent, essential!*

Embedded Systems: Hardware and Software

- Importance of Embedded Processors and Software
  - By Peter Marwedel, University of Dortmund, Germany

"... the New York Times has estimated that the average American comes into contact with about 60 micro-processors every day...." [Camposano, 1996]

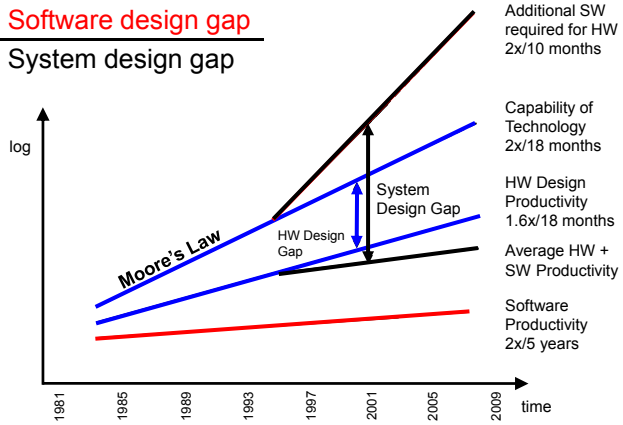
Latest top-level BMWs contain over 100 micro-processors [Personal communication]

Most of the functionality will be implemented in software!

Embedded System Design Challenge

- Productivity Gap

  Hardware design gap
  + Software design gap
  = System design gap



(source: *"Hardware-dependent Software"*, Ecker et al., 2009)

Computer Programming

- What is a Computer?
  - *"A computer is a general purpose device that can be programmed to carry out a finite set of arithmetic or logical operations."*
  - *"Since a sequence of operations can be readily changed, the computer can solve more than one kind of problem."* [Definition by Wikipedia]

➢ It is the **programming** that makes a computer so powerful!

➢ Advise: *Carpe Diem!* ("Seize the day!")
  - ➢ Utilize the opportunity! *Learn programming!*
    - ➢ For your work!
    - ➢ For your life!
    - ➢ For your future!

Learn Pogramming!

- Every Engineer needs Programming
  - Computer Engineers
  - Electrical Engineers
  - Mechanical and Aerospace Engineering
  - Chemical Engineering
  - Materials Science Engineering
  - Civil and Environmental Engineering
  - …
- *Everybody* should learn programming!
  - Don't take my word for it?
  - ➢ Watch this:
    http://www.youtube.com/watch?v=nKlu9yen5nc

Computer Engineering Courses

- Lower Division: **Programming**
  - EECS 10 Computational Methods in ECE
  - EECS 12 Introduction to Programming
  - EECS 20 Computer Systems and Programming in C
  - EECS 22 Advanced C Programming
  - EECS 22L Software Engineering Project in C Language
  - EECS 40 Object Oriented Systems and Programming
- Upper Division: **Software Systems**
  - EECS 111 System Software
  - EECS 112/112L Organization of Digital Computers
  - EECS 113 Processor Hardware/Software Interfaces
  - EECS 114 Engineering Data Structures and Algorithms
  - EECS 159A,B,C Senior Design Project

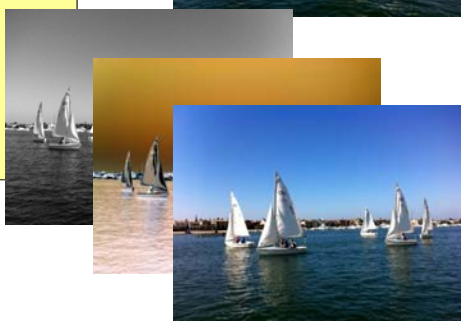
Computer Programming Courses

- Introductory Programming
  - EECS 10: uses C Programming Language (for EE)
  - EECS 12: uses Python Programming Language (for CpE)
- Programming from the Ground Up
  - EECS 20: starts with Assembly Language (on bare CPU), then introduces C Programming Language
- Core Programming Courses
  - EECS 22: *"Advanced C Programming"* (in ANSI C)
  - EECS 22L: *"Software Engineering Project in C"* (ANSI C/C++)
- Object-Oriented Programming
  - EECS 40: introduces objects and classes, hierarchy, and other high-level programming concepts (using Java on Android platform)

Programming Assignment Example 1

- PhotoLab: Digital Image Processing
  - Final Assignment in EECS 10
  - Initial Assignment in EECS 22

```
1: Load a PPM image
2: Save an image in PPM and JPEG format
3: Change a color image to black and white
4: Make a negative of an image
5: Flip an image horizontally
6: Mirror an image horizontally
7: Add border to an image
8: Flip an image vertically
9: Mirror an image vertically
10:
11:
Plea
```



EECS 22: Advanced C Programming

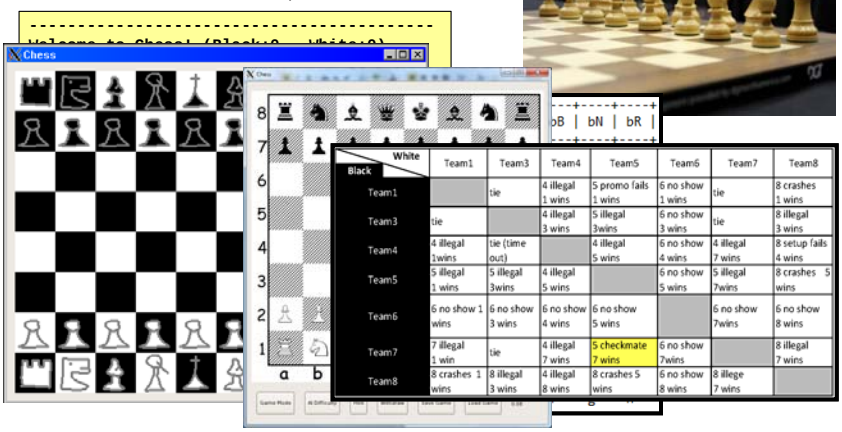
- *"All you want to know about C Programming"*
  - Review and reinforce basic C programming concepts
  - Study advanced features in detail
  - Put concepts and tools to their best use
- Features
  - Dynamic data structures using **`malloc()`**, **`free()`**
  - Keywords **`static`**, **`register`**, **`auto`**, **`extern`**, **`volatile`**, …
  - Advanced data types, variable-length arguments, …
  - Libraries, Makefile, …
- Tools
  - C preprocessor, compiler, and linker
  - Debugger 'gdb' and 'ddd'
  - Dynamic memory allocation checker 'valgrind'

EECS 22L: Software Eng. Project in C

- *"Developing real C Programs in a Team"*
  - Hands-on experience with larger software projects
  - Introduction to software engineering
    - Specification, documentation, implementation, testing
  - Team work
- Features
  - Design efficient data structures, APIs
  - Utilize programming modules, build libraries
  - Develop and optimize contemporary software applications
- Tools
  - Scripting 'make'
  - Version control 'cvs'
  - Testing and debugging with 'gdb', 'gprof', 'valgrind', …

Programming Assignment Example 2

- The Game of Chess: Computer AI
  - First Assignment in EECS 22L
  - Team of 5 students, over 5 weeks

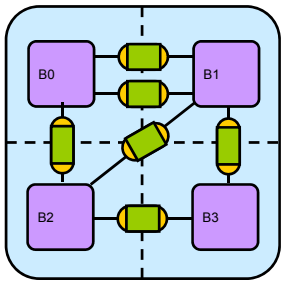


White / Black	Team1	Team3	Team4	Team5	Team6	Team7	Team8
Team1		tie	4 illegal 1 wins	5 promo fails 1 wins	6 no show 1 wins	tie	8 crashes 1 wins
Team3	tie		4 illegal 3 wins	5 illegal 3wins	6 no show 3 wins	tie	8 illegal 3 wins
Team4	4 illegal 1wins	tie (time out)		4 illegal 5 wins	6 no show 4 wins	4 illegal 7 wins	8 setup fails 4 wins
Team5	5 illegal 1 wins	5 illegal 3wins	4 illegal 5 wins		6 no show 5 wins	5 illegal 7wins	8 crashes 5 wins
Team6	6 no show 1 wins	6 no show 3 wins	6 no show 4 wins	6 no show 5 wins		6 no show 7wins	6 no show 8 wins
Team7	7 illegal 1 win	tie	4 illegal 7 wins	5 checkmate 7 wins	6 no show 7wins		8 illegal 7 wins
Team8	8 crashes 1 wins	8 illegal 3 wins	4 illegal 8 wins	8 crashes 5 wins	6 no show 8 wins	8 illege 7 wins	

Research: Embedded System Design

- Electronic System Level Models
  - Abstract description of a complete embedded system
  - Hardware + Software
- Key Concepts in System Modeling
  - Explicit Structure
    - Block diagram structure
    - Connectivity through ports
  - Explicit Hierarchy
    - System composed of components
  - Explicit Concurrency
    - Potential for parallel execution
    - Potential for pipelined execution
  - Explicit Communication and Computation
    - Modules
    - Channels and Interfaces

SystemC Model

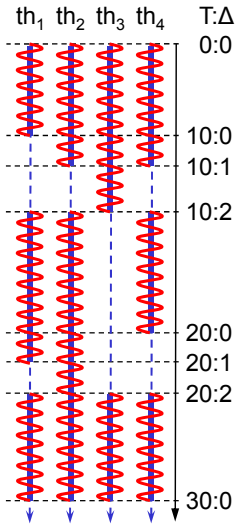


Research: Embedded System Design

- Research Project Example:
  Embedded System Validation through Simulation
  - Efficient system-level simulation is critical
    - Fast, and
    - Accurate!
  - Complexity of system models grows constantly
    - Need for speed!
- Parallel Simulation!
  - Parallelism explicitly specified in model
    - System-level Description Language (SLDL)
      - SystemC [Groetker et. al, 2002]: `SC_THREAD, SC_METHOD`
      - SpecC [Gajski et. al, 2000]: `par { }, pipe { }`
  - Parallel processing available in standard PCs
    - Multi-core host PCs readily available (tens of processor cores)
    - Many-core technology is arriving (thousands of processor cores)

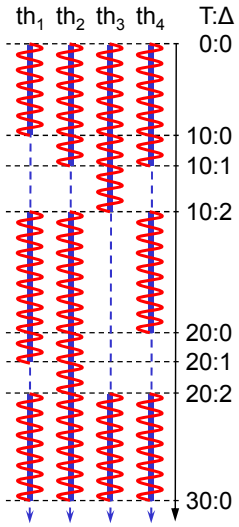
Discrete Event Simulation (DES)

- Traditional DES
  - Concurrent threads of execution
  - Managed by a central scheduler
  - Driven by events and time advances
    - Delta-cycle
    - Time-cycle
  - ➢ Partial temporal order with barriers
- IEEE Standard Simulator
  - SystemC reference simulator
    uses cooperative multi-threading
  - ➢ A single thread is active at any time!
  - ➢ Cannot exploit parallelism
  - ➢ Cannot utilize multiple cores



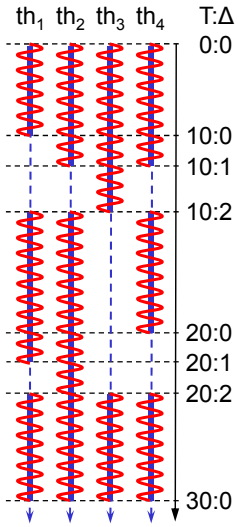
Parallel Discrete Event Simulation

- Parallel DES
  - Threads execute in parallel *iff*
    - in the same delta cycle, *and*
    - in the same time cycle
  - ➢ Significant speed up!
  - *Synchronous* PDES: Cycle boundaries are *absolute barriers*!
- Aggressive Parallel DES
  - Conservative Approaches
    - Careful static analysis prevents conflicts
  - Optimistic Approaches
    - Conflicts are detected and addressed (*roll back*)



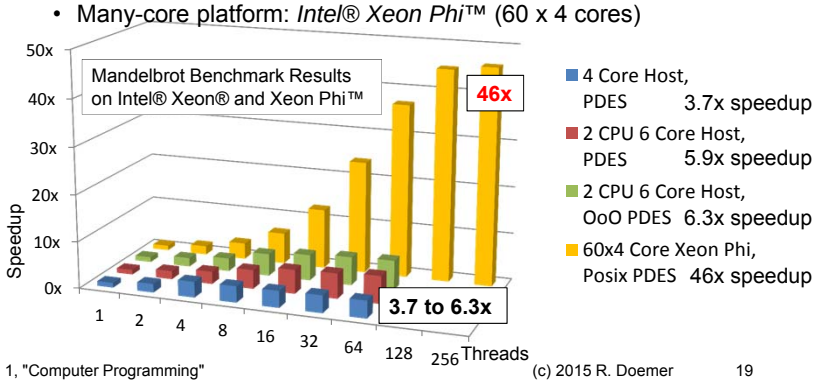
Out-of-Order Parallel DES

- Out-of-Order PDES
  - Threads execute in parallel *iff*
    - in the same delta cycle, *and*
    - in the same time cycle,
    - *OR* if there are no conflicts!
  - Can utilize advanced compiler for static data conflict analysis
  - ➢ Allows as many threads in parallel as possible
  - ➢ Significantly higher speedup!
    - Results at [DATE'12], [IEEE TCAD14]
  - ➢ Fully preserves…
    - ➢ DES execution semantics
    - ➢ Accuracy in results and timing



Research: Embedded System Design

- Research Project Example: Advanced Parallel Simulation
  - Experimental Results
    - Rendering of 100 Mandelbrot images (640x448, depth 4096)
    - Manually created PDES model (Posix-threads based)
    - Multi-core platforms: *Intel® Xeon® CPUs* (4 cores, 2x6 cores)
    - Many-core platform: *Intel® Xeon Phi™* (60 x 4 cores)



Concluding Remarks

- Learn programming!
  - It's where the jobs are…

- Do research!
  - It's where the fun is...