Running Program on Processor

Processor Performance = \frac{\text{Time}}{\text{Program}}

= \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Time}}{\text{Cycle}}

\text{(code size)} \quad \text{(CPI)} \quad \text{(cycle time)}

Architecture --> Implementation --> Realization

Compiler Designer \quad Processor Designer \quad Chip Designer
Abstraction and Complexity

- Abstraction helps us manage complexity
- Complex interfaces
  - Specify what to do
  - Hide details of how
- Goal: remove magic

Scope of this course:

- Application Program
- Operating System
- Compiler
- Machine Language (ISA)
- Digital Logic
- Electronic circuits
- Semiconductor devices
Computer Architecture

• Exercise in engineering tradeoffs analysis
  – Find the fastest/cheapest/power-efficient/etc. solution
  – Optimization problem with 100s of variables

• All the variables are changing
  – At non-uniform rates
  – With inflection points
  – Only one guarantee: Today’s right answer will be wrong tomorrow

• Two high-level effects:
  – Technology push
  – Application Pull
Technology Push

• What do these two intervals have in common?
  – 1776-1999 (224 years)
  – 2000-2001 (2 years)

• Answer: Equal progress in processor speed!

• The power of exponential growth!

• Driven by Moore’s Law
  – Device per chips doubles every 18-24 months

• Computer architects work to turn the additional resources into speed/power savings/functionality!
<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1939</td>
<td>First digital computer</td>
<td>John Atanasoff (UW PhD ’30)</td>
</tr>
<tr>
<td>1947</td>
<td>1\textsuperscript{st} transistor</td>
<td>Bell Labs</td>
</tr>
<tr>
<td>1958</td>
<td>1\textsuperscript{st} IC</td>
<td>Jack Kilby (MSEE ’50) @TI Winner of 2000 Nobel prize</td>
</tr>
<tr>
<td>1971</td>
<td>1\textsuperscript{st} microprocessor</td>
<td>Intel</td>
</tr>
<tr>
<td>1974</td>
<td>Intel 4004</td>
<td>2300 transistors</td>
</tr>
<tr>
<td>1978</td>
<td>Intel 8086</td>
<td>29K transistors</td>
</tr>
<tr>
<td>1989</td>
<td>Intel 80486</td>
<td>1.M transistors, pipelined</td>
</tr>
<tr>
<td>1995</td>
<td>Intel Pentium Pro</td>
<td>5.5M transistors</td>
</tr>
<tr>
<td>2005</td>
<td>Intel Montecito</td>
<td>1B transistors</td>
</tr>
</tbody>
</table>
Performance Growth

Unmatched by any other industry!
[John Crawford, Intel]

- **Doubling every 18 months (1982-1996):** 800x
  - Cars travel at 44,000 mph and get 16,000 mpg
  - Air travel: LA to NY in 22 seconds (MACH 800)
  - Wheat yield: 80,000 bushels per acre

- **Doubling every 24 months (1971-1996):** 9,000x
  - Cars travel at 600,000 mph, get 150,000 mpg
  - Air travel: LA to NY in 2 seconds (MACH 9,000)
  - Wheat yield: 900,000 bushels per acre
Technology Push

- Technology advances at varying rates
  - E.g. DRAM capacity increases at 60%/year
  - But DRAM speed only improves 10%/year
  - Creates gap with processor frequency!

- Inflection points
  - Crossover causes rapid change
  - E.g. enough devices for multicore processor (2001)

- Current issues causing an “inflection point”
  - Power consumption
  - Reliability
  - Variability
Application Pull

- Corollary to Moore’s Law:
  Cost halves every two years
  
  In a decade you can buy a computer for less than its sales tax today. –Jim Gray

- Computers cost-effective for
  - National security – weapons design
  - Enterprise computing – banking
  - Departmental computing – computer-aided design
  - Personal computer – spreadsheets, email, web
  - Pervasive computing – prescription drug labels
Application Pull

• What about the future?

• Must dream up applications that are not cost-effective today
  – Virtual reality
  – Telepresence
  – Mobile applications
  – Sensing, analyzing, actuating in real-world environments

• This is your job
Abstraction

• Difference between interface and implementation
  – Interface: WHAT something does
  – Implementation: HOW it does so
Abstraction, E.g.

- 2:1 Mux (Digital Design)
- Interface

- Implementations
  - Gates (fast or slow), pass transistors
What’s the Big Deal?

- Tower of abstraction
- Complex interfaces implemented by layers below
- Abstraction hides detail
- Hundreds of engineers build one product
- Complexity unmanageable otherwise

Content:
- Quantum Physics
- Transistors & Devices
- Logic Gates & Memory
- Von Neumann Machine
- x86 Machine Primitives
- Visual C++
- Windows 7
- Firefox, MS Excel
Basic Division of Hardware

- In space (vs. time)
Basic Division of Hardware

• In time (vs. space)
  – Fetch instruction from memory  add r1, r2, r3
  – Decode the instruction – what does this mean?
  – Read input operands  read r2, r3
  – Perform operation  add
  – Write results  write to r1
  – Determine the next instruction  pc := pc + 4
Building Computer Chips

• Complex multi-step process
  ✓ Slice silicon ingots into wafers
  ✓ Process wafers into patterned wafers
  ✓ Dice patterned wafers into dies
  ✓ Test dies, select good dies
  ✓ Bond to package
  ✓ Test parts
  ✓ Ship to customers and make money
Building Computer Chips

Silicon ingot → Slicer → Blank wafers → 20 to 30 processing steps

Bond die to package → Tested dies → Die tester → Individual dies (one wafer) → Dicer → Patterned wafers

Packaged dies → Part tester → Tested packaged dies → Ship to customers
Performance vs. Design Time

• Time to market is critically important
• E.g., a new design may take 3 years
  – It will be 3 times faster
  – But if technology improves 50%/year
  – In 3 years $1.5^3 = 3.38$
  – So the new design is worse!
    (unless it also employs new technology)
Bottom Line

- Designers must know BOTH software and hardware
- Both contribute to layers of abstraction
- IC costs and performance
- Compilers and Operating Systems
About This Course

• Course Textbook

• Homework
  – ~5 homework assignments, unequally weighted
  – Some group, some individual
  – No late homework will be accepted

• Discussion: TBD
Thank You