ECE 471 – Embedded Systems
Lecture 21

Vince Weaver

http://www.eece.maine.edu/∼vweaver

vincent.weaver@maine.edu

19 November 2013

Announcements

- Project groups and topics!

HW#4 – Specifications

- For the HW the major spec is “it works”. There may be other concerns in the real world.
- Speed
- Size
- Development Time
- Portability/Maintainability

- Resources (how small the CPU? Floating point library? Stripped-down C library?)

HW#4

- Not compile?
- Give wrong result because still testing code
- “LCD” instead of “LED”
- Wrong value... multiply by (9/5) in conversion routine
- I wrote spec poorly. -1.0 to 0.0. Undefined values in cracks.

HW#4

- How did I write it?
- int_temp=temp*10.0;
- digit[0]=int_temp/100;
- int_temp-=(int_temp/100)*100;
- digit[1]=int_temp/10;
- digit[2]=int_temp%10;

- lookup table

HW#4

- Other ways?
- sprintf()
- ??
- Fixed point? How would you do it in assembly?

Notes on Process Technology

- 65nm – 2006
  p4 to core2, IBM Cell
  1.0v, High-K dielectric, gate thickness a few atoms
  193/248nm light (UV)

- 45nm – 2008
  core2 to nehalem
  large lenses, double patterning, high-k

- 32nm – 2010

sandybridge to westmere
immersion lithography

- 22nm – 2012 ivybridge, haswell
  oxide only 0.5nm (two silicon atoms)
  fin-fets

- 14nm and smaller – ??
  Extreme UV (13.5nm light, hard-vacuum required)?
  Electron beam?

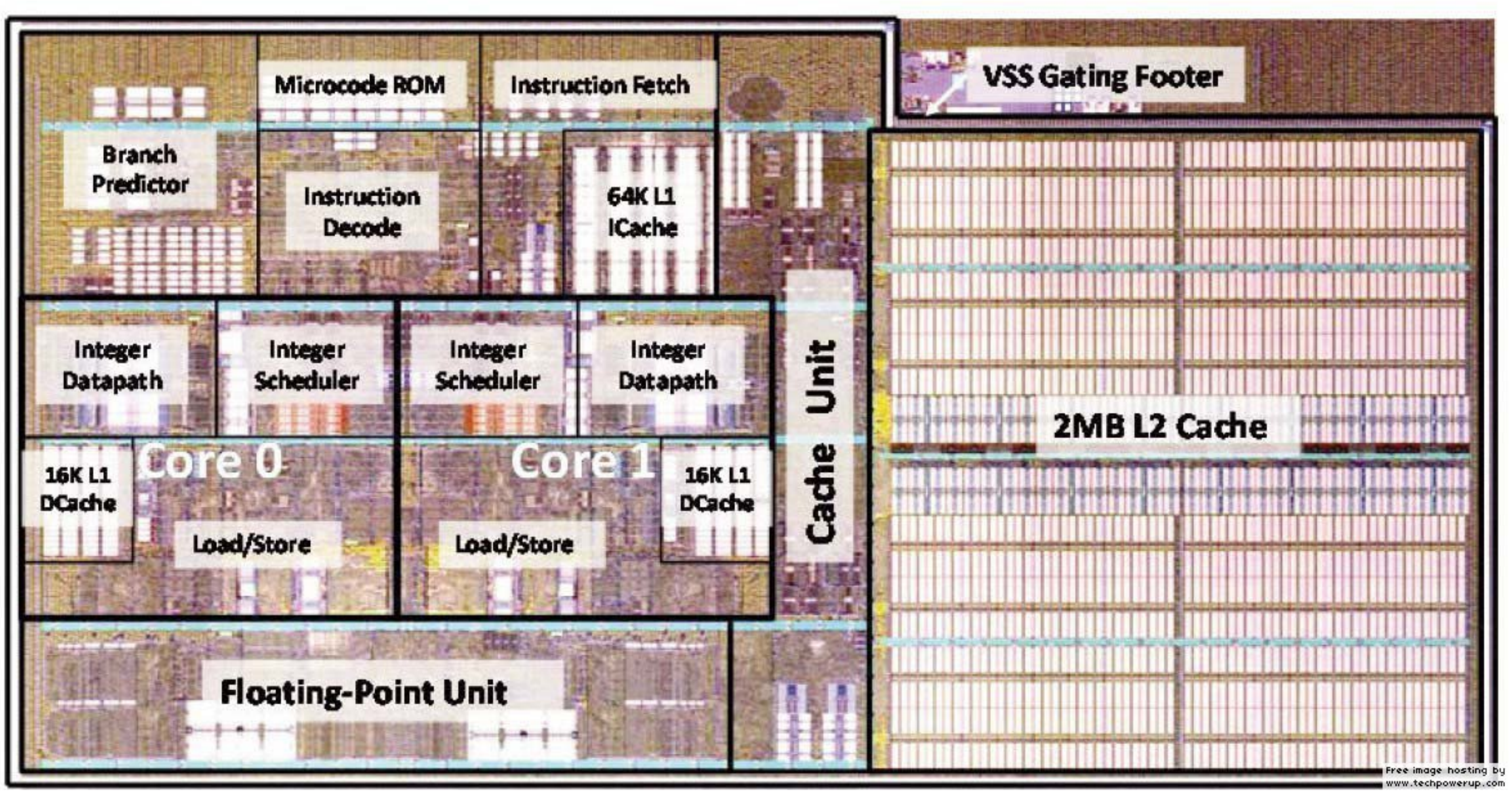
Notes on Process Technology

- TI-OMAP cell phone processor (more or less discontinued by TI, big layoffs in 2012)
  Beagle Board and Gumstix OMAP35?? – 65nm
- OMAP4460 (Pandaboard) 45nm
- Cortex A15 28nm
- Rasp-pi BCM2835 – 45nm?

CPU Power and Energy

- Became a trendy thing to research in 1999-2002
- Before that usually concern was with performance.
- These days energy results are often reported as a core part of any architectural proposal, not as a separate issue.
- The results discussed here are academic and may or may not be implemented in actual chips.

AMD Bulldozer Die Shot



Note which structures are big, using static power.

CPU Energy Breakdown

From Fan, Tang, Huan, Gao (ISLPED'05), Chinese Godson MIPS CPU

- Cache 36%
- TLB 13%
- FALU 10%
- ROQueue 7%
- FMUL 6%

- Float reg 5%
- Gen reg 5%
- MUL 2%
- MCUControl 2%
- ALU 1%
- Other 13%

Thermal Concerns Too

Power density exceed hot plate, approaching rocket nozzle

Methodologies Used in These Papers

It varies, but many of these are from simulations (sometimes validated). Anything from SPICE to “cycle-accurate” simulators.

Clock Generation

- Driving high-frequency load against capacitance, trying to keep whole chip in sync.
- Extreme Case: Alpha 21264 H-tree, 32% of power?
- Half-frequency clocks (on both edge, so clock run half as fast) (Mudge 2001)
- Asynchronous
- Locally Asynchronous (Divide to multiple clock domains)

Cache Power and Energy

Large area, low-hanging fruit

Decay Caches

- Kaxiras, Ho, Martinosi (ISCA 2001)
- Turn off cache lines not being used to reduce leakage
- DRAM cache with no refresh
- Decayed values can be re-fetched from memory. Tradeoff.

Drowsy Caches

- Flautner, Kim, Martin, Blaauw, Mudge. ISCA 2002.
- Move cold cache lines into “drowsy” mode.
  Lower power enough to hold state, not enough to lose contents. Reduce leakage. Better than decay as not lose data.

Adaptive Caches

- Albonesi (Micro 1999). Manually turn off ways in cache with an instruction.
- Size the caches

Cache Compression

- Dynamic zero compression for cache energy reduction (L Villa, M Zhang, K Asanović. Micro 2001).
- Cache Compression ("sign compression" – top bits) Energy savings 20% (simulated) (Kim, Austin, Mudge WMPI 2002)

Banking and Filtering

- Filter cache, banking (only have half of cache active) (Mudge 2001)
- Slowing Down Cache Hits, Banked Data Cache. (Huang, Renau, Yoo, and Torrellas. Micro 2000.)
- Vertical Banking, Horizontal Banking (Su and Despain, ISLPED 1995).

Code Scheduling

- Can Schedule code for lower power.
- Better cache rates lower power. performance/power can go hand in hand. (Kandemir, Vijaykrishnan, Irwin)

Branch Predictors

- Parikh, Skadron, Zhang, Barcella, Stan
- 4 concerns:
  1. Accuracy. Not affect power, but performance
  2. Configuration (may affect power)
  3. Number of lookups
  4. Number of updates
- Tradeoff power vs time.

- brpred can be size of small cache, 10% of power
- Can use banking to mitigate

Branch Predictors

- can watch icache, not activate predictor if nobranches
- Pipeline gating, keep track of each predicted branch confidence. If confidence hits certain threshold, stop speculating. Show this may or may not be good.
- Integer code, large predictors good
- FP, tight loops, predictors not as important.

Branch Predictor Evaluation

- (Strasser, 1999). Simulation, small branch predictor can help energy.
- (Co, Weikle, Skadron) Formula for break even point. Leakage matters, what brpred hides is stall cycles.
- SEPAS: A Highly Accurate Energy-Efficient Branch Predictor (Baniasadi, Moshovos. ISLPED 2004). Once a branch prediction reaches steady state (unlikely to change) stop accessing/updating predictor, saving

energy.

- Low Power/Area Branch Prediction Using Complementary Branch Predictors (Sendag, Yi, Chuang, Lija. IPDPS 2008)
  Complementary Branch Predictor to handle the tough cases.

TLB Energy

TLB Optimization – Assume in Same Page

- (Kadayif, Sivasubramaniam, Kandemir, Kandiraju, Chen. TODAES 2005).
  Don't access TLB if not necessary. Compare to last access (assume stay in same page) Circuit improvements

- (Kadayif,Sivasubramaniam, Kandemir, Kandiraju, Chen. Micro 2002)
  Cache page value.

TLB Optimization – Use Virtual Caches

- (Ekman and Stenström, ISLPED 2002) Use virt address cache. Less TLB energy, more snoop energy. TLB keeps track of shared pages.

TLB Optimization – Reconfiguring

- (Basu, Hill, Swift. ISCA 2012) Have the OS select if memory region physical or virtual cached.
- (Delaluz, Kandemir, Sivasubramaniam, Irwin, Vijaykrishnan. ICCD 2013) Reducing dTLB Energy Through Dynamic Resizing.
  Size TLB as needed, shutting off banks. Easier if fully-associative.

TLB Optimization – Memory Placement

- (Jeyapaul, Marathe, Shrivastava, VLSI'09) Try to keep as much in one page as possible via compiler.
- (Lee, Ballapuram. ISLPED'03) Split memory regions by region (text/data/heap). Better TLB performance, better energy.

Bus Protocols

- Bus Protocols
- Cache-Coherence Protocols

Busses

- Grey Code, only one bit change when incrementing. Lower energy on busses? (Su and Despain, ISLPED 1995).
- i2c – use addresses with lots of 1s

Prefetching

- Prefetching does not get looked at as closely. Various studies show it can be a win energy wise, but it is a close thing.
- (Guo, Chheda, Koren, Krishna, Moritz. PACS'04) HW Prefetch increase power 30%; have compiler help augment with hints, filters.
- (Tang, Liu, Gu, Liu, Gaudiot. Computer Architecture Letters, 2011).

Mixed results.

Metrics Clarification

- Energy delay = E*t (J*s), Energy delay squared = E*t*t (J*s*s), Smaller is better
- In related papers it is confusing, as no one shows formula (despite academic papers loving formulas). Often they use the inverse (so larger is better?) which also confuses things
- Other papers use MIPS/Watt = insn/J
  Not cross-platform. You really want to optimize for time,

not for instructions which can vary for lots of reasons and doesn't exactly equate to time.

DVFS and other CPU Power/Energy Saving Methods

- A lot of related work
- Will focus on actual implementations rather than academic papers this time

CMOS Energy Equation, Again

- $E_{tot} = [(C_{tot}V_{dd}^2\alpha f) + (N_{tot}I_{leakage}V_{dd})]t$

Delay, Again

- $T_d = \frac{C_L V_{dd}}{\mu C_{ox}(\frac{W}{L})(V_{dd} - V_t)}$
- Simplifies to $f_{MAX} \sim \frac{(V_{dd} - V_t)^2}{V_{dd}}$
- If you lower f, you can lower V_{dd}

DVFS

- Voltage planes – on CMP might share voltage planes so have to scale multiple processors at a time
- DC to DC converter, programmable.
- Phase-Locked Loops. Orders of ms to change. Multiplier of some crystal frequency.
- Senger et al ISCAS 2006 lists some alternatives. Two phase locked loops? High frequency loop and have programmable divider?

- Often takes time, on order of milliseconds, to switch frequency. Switching voltage can be done with less hassle.

When can we scale CPU down?

- System idle
- System memory or I/O bound
- Poor multi-threaded code (spinning in spin locks)
- Thermal emergency
- User preference (want fans to run less)

Adaptive Body Biasing

- Related to but not always considered part of DVFS
- Control voltage applied to body
- Change the threshold voltage
- Reduces leakage but slows performance