ECE 571 – Advanced Microprocessor-Based Design Lecture 17

Vince Weaver

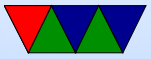
http://www.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

6 November 2014

Announcements

- HW4 due Friday
- Also project topic selection due Friday



NOMMU (uCLinux) Linux

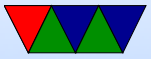
Good references:

`http://elinux.org/images/6/68/Porting_uClinux_CELF2008_Griffin.pdf`

`https://github.com/linux-test-project/ltp/blob/master/doc/nommu-notes.txt`

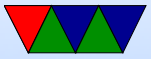
`http://www.linuxjournal.com/article/7221`

- First ported in 1998 (Kernel 2.0) to m68k processors (coldfire)
- Most of kernel same (drivers, etc). No memory

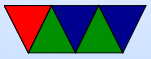


protection

- No demand loading of executables (all loaded at once, unless you do XIP)
- No on-demand stack growth, have to specify in advance (4k default)
- No auto-growing heap
- No memory over-committing. Ask for memory, you get it physical.

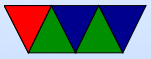


- Memory fragmentation can be a major problem.
- No paging/swapping
- No fork system call (use vfork). vfork creates a child like fork, but you are only allowed to _exit() or exec() in the child. No page tables copied. Also parent sleeps until child is done, and runs in parent's memory/stack space.
- RAM can get fragmented
- Special "bin flat" executable format. No dynamic stack, process has to be compiled re-locatable.

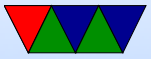


- XIP (execute in place) lets program text be read-only, execute out of ROM/RAM in one position, multiple programs share it (busybox?).

- Special memory allocator to not waste space

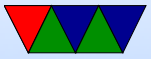


TLB Energy



Simple ideas

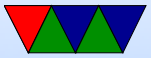
- TLB is similar to a cache, so use same optimization techniques (drowsy, etc)



TLB Optimization – Assume in Same Page

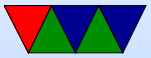
- (Kadayif, Sivasubramaniam, Kandemir, Kandiraju, Chen. TODAES 2005).
  Don't access TLB if not necessary. Compare to last access (assume stay in same page) Circuit improvements

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



TLB Optimization – Use Virtual Caches

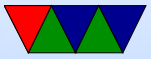
- (Ekman and Stenström, ISLPED 2002) Use virt address cache. Less TLB energy, more snoop energy. TLB keeps track of shared pages.



TLB Optimization – Reconfiguring

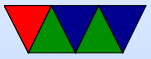
- (Basu, Hill, Swift. ISCA 2012) Have the OS select if memory region physical or virtual cached.

- (Delaluz, Kandemir, Sivasubramaniam, Irwin, Vijaykrishnan. ICCD 2013) Reducing dTLB Energy Through Dynamic Resizing.
  Size TLB as needed, shutting off banks. Easier if fully-associative.



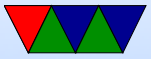
TLB Optimization – Memory Placement

- (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.

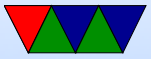


DRAM

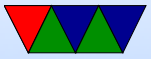
- Single transistor/capacitor pair. (can improve behavior with more transistors, but then less dense)
- Compare to SRAM that has 6 transistors (or 4 plus hard-to make resistors with high static power)
- In 90nm process, 30fF capacitor, leakage in transistor 1fA. Can hold charge from milliseconds to seconds.
- DRAMs gradually lose charge, need to be refreshed. Need to e conservative. Refresh every 32 or 64ms



- Found in DIMMs
- How many chips on DIMM? 8? 9? Why?
- Chips x1 x4 x8 bits, how many get output at a time. Grouped together called a "bank"
- Banks can mask latency, sort of like pipelining. If takes 10ns to respond, interleave the request.
- DIMM can have independent "ranks" (1 or 2 per DIMM), each with banks, each with arrays

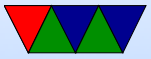


- Layout, multiple mem controllers, each with multiple channels, each with ranks, banks, arrays



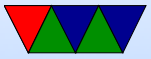
SIMMs/DIMMS

- Has SPD "serial prescence detect" chip that holds RAM timings and info. Controlled by smbus (i2c)
- SODIMM – smaller form factor for laptops



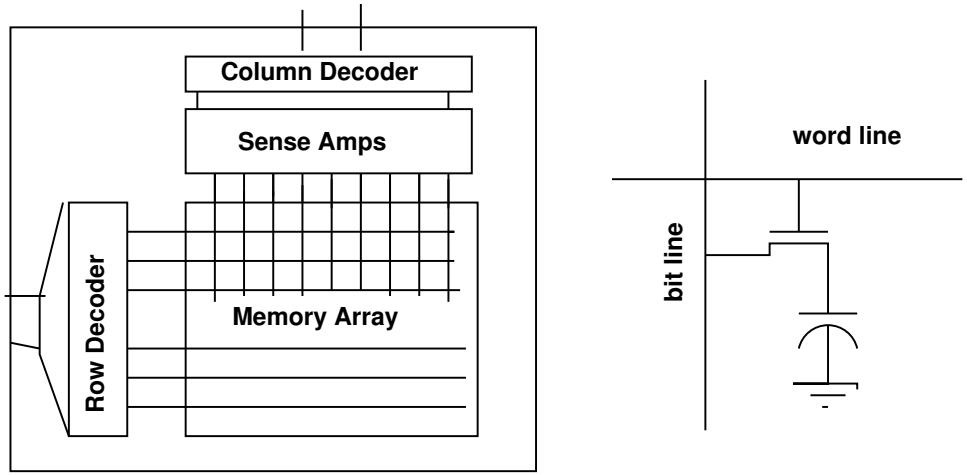
Read/Write

- DRAM read is destructive, always have to write back

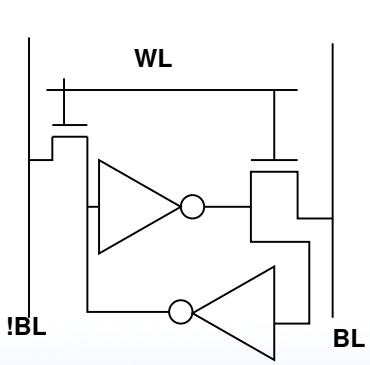


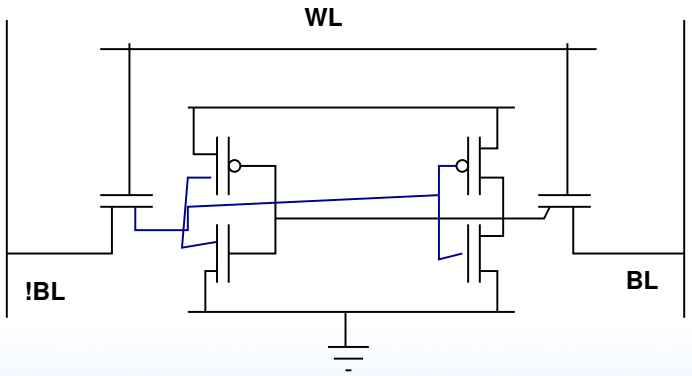
Diagram

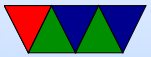
DRAM



SRAM

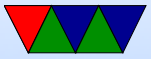






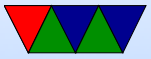
Refresh

- Need to read out each row, then write it back. every 32 to 64ms
- Old days; the CPU had to do this. Slow
- Newer chips have "auto refresh"

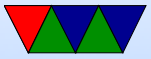


Memory Bus

- JEDEC-style. address/command bus, data bus, chip select
- Row address sends to decoder, activates transistor
- Transistor turns on and is discharged down the column rows to the sense amplifier which amplifies
- The sense amplifier is first "pre-charged" to a value halfway between 0 and 1. When the transistors are enabled the very small voltage swing is amplified.

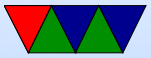


- This goes to column mux, where only the bits we care about are taken through



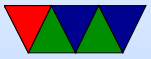
Memory Access

- CPU wants value at address
- Passed to memory controller
- Memory controller breaks into rank, bank, and row/column
- Proper bitlines are pre-charged
- Row is activated, then $\overline{RAS}$, row address strobe, is signaled, which sends all the bits in a row to the sense



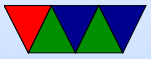
amp. can take tens of ns.

- Then the desired column bits are read. The $\overline{CAS}$ column address strobe sent.
- Again takes tens of ns, then passes back to memory controller.
- Unlike SRAM, have separate CAS and RAS? Why? Original DRAM had low pincount.
- Also a clock signal goes along. If it drives the device it's synchronous (SDRAM) otherwise asynchronous



Memory Controller

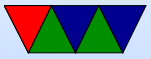
- Formerly on the northbridge
- Now usually on same die as CPU



Advances

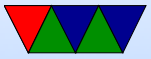
In general the actual bit array stays same, only interface changes up.

- Clocked
- Asynchrounous
- Fast page mode (FPM) – row can remain active across multiple CAS.
- Extended Data Out (EDO) – like FPM, but buffer



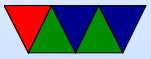
"caches" a whole page of output is the CAS value the same.

- Synchronous (SDRAM) – drives internal circuitry from clock rather than outside RAS and CAS lines. Previously the commands could happen at any time. Less skew.



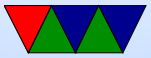
Memory Types

- SDRAM – 3.3V
- DDR – transfer and both rising and falling edge of clock 2.5V. Adds DLL to keep clocks in sync (but burns power)
- DDR2 – runs internal bus half the speed of data bus. 4 data transfers per external clock. memory clock rate * 2 (for bus clock multiplier) * 2 (for dual rate) * 64 (number of bits transferred) / 8 (number of bits/byte) so at 100MHz, gives tranfser rate of 3200MB/s. not pin

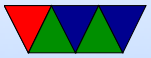


compatible with DDR3. 1.8 or 2.5V

- DDR3 – internal doubles again. Up to 6400MB/s, up to 8gigabit dimms. 1.5V or 1.35V
- DDR3L - low voltage, 1.35V
- DDR4 – just released. 1.2V , 1600 to 3200MHz. 2133MT/s, parity on command/address busses, crc on data busses.
- DDR4L – 1.05V

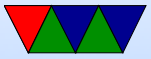


- GDDR2 – graphics card, but actually halfway between DDR and DDR2 technology wise
- GDDR3 – like DDR2 with a few other features. lower voltage, higher frequency, signal to clear bits
- GDDR4 – based on DDR3, replaced quickly by GDDR5
- GDDR5 – based on DDR3

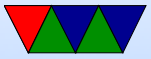


More obscure Memory Types

- RAMBUS RDRAM – narrow bus, fewer pins, could in theory drive faster. Almost like network packets. Only one byte at time, 9 pins?
- FB-DIMM – from intel. Mem controller chip on each dimm. High power. Requires heat sink? Point to point. If multiple DIMMs, have to be routed through each controller in a row.
- VCDRAM/ESDRAM – adds SRAM cache to the DRAM

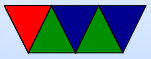


- 1T-SRAM – DRAM in an SRAM-compatible package, optimized for speed



Memory Latencies, Labeling

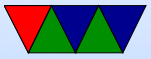
- DDR400 = 3200MB/s max, so PC3200
- DDR2-800 = 6400MB/s max, so PC2-6400
- DDR2 5-5-5-15
- CAS latency – T_{RCD} row address to column address delay – T_{RP} row precharge time – T_{RAS} row active time
- DDR3 7-7-7-20 (DDR3-1066) and 8-8-8-24 (DDR3-1333).



Memory Parameters

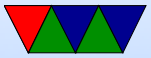
You might be able to set this in BIOS

- Burst mode – select row, column, then send consecutive addresses. Same initial latency to setup but lower average latency.
- CAS latecny – how many ccyles it takes to return data after CAS.
- Dual channel – two channels (two 64-bit channels to memory). Require having DIMMs in pairs



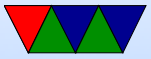
ECC Memory

- Scrubbing



Issues

- Truly random access? No, burst speed fast, random speed not Is that a problem? Mostly filling cache lines?



Future

- Phase-change RAM
- Non-volatile memristor RAM