DRAM

Today's topics:

Brief look at DRAM devices

Channel protocols & signalling

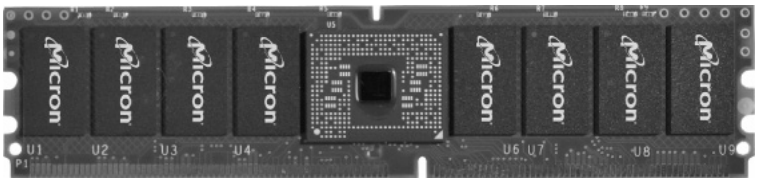
Memory controller issues

This is just a skim – CS7810 will have a more in depth treatment of lots of topics including this one

Plan Preview

- So far focus has been on-chip
  - processors, caches
  - and a bit of interconnect
  - brief look at parallel processing on 1 or more sockets
- Note that many big applications are I/O or memory bound
  - last 3 lectures before the 2nd midterm
    - » DRAM – this one
      - 2 standards
        - JEDEC – DDRn – focus on this one – 64 bit slower data bus
        - RAMBUS – RDRAM – fast skinny bus
      - pace will be rapid – high level understanding is the goal
    - » Disk and storage
    - » non-volatile RAM (e.g. not disk)
  - goal is to give you a high level understanding
    - » details are way too complicated to cover in 1.5 weeks

DRAM: Overview & Devices



Reference: "Memory Systems: Cache, DRAM, Disk

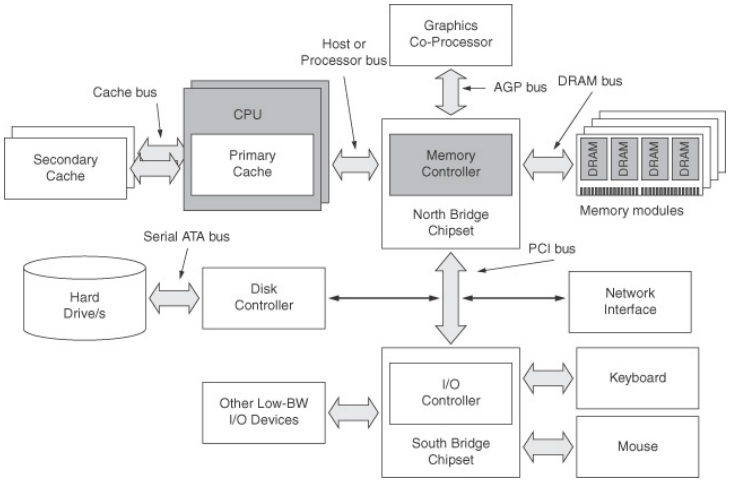
Bruce Jacob, Spencer Ng, & David Wang

Today's material & any uncredited diagram came from chapters 7 & 8

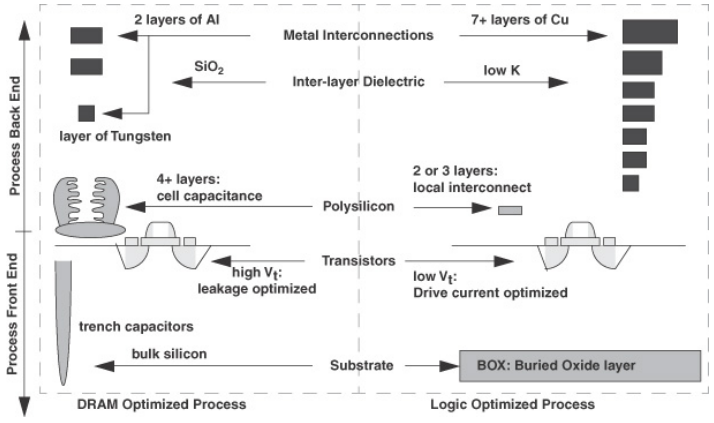
Key Item to Remember

- **It is easy to predict SRAM behavior**
  - **even though discrete SRAM may well disappear in this decade**
    - » **since cache buses (BSBs) are now extinct**
- **Hard to predict DRAM behavior**
  - **probabilistic resource availability**
  - **performance depends on controller and device model**
    - » **small controller differences show up as big performance differences**
    - » **access pattern has an even bigger effect than with caches**
      - **primarily since DRAM accesses are so much slower**
- **Disk performance is probabilistic as well**
- **Plus**
  - **lots of intermediate buffers, prefetch, … issues**

Typical PC



DRAM vs. Logic Process

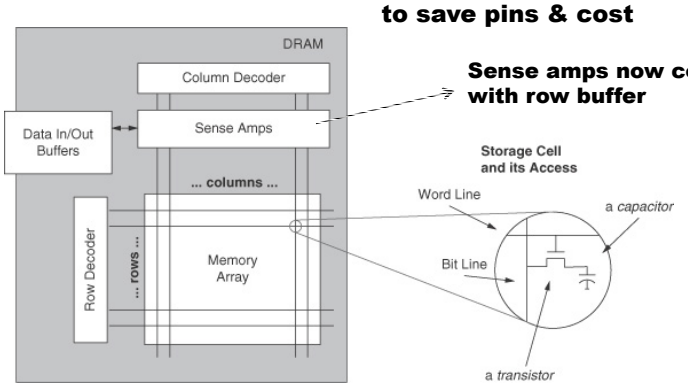


Hybrid Processes Coming

- **IBM was the pioneer**
  - **start with logic process**
  - **add extra layers to create high-C DRAM cells**
    - » **multiple oxide thicknesses**
      - **fast leaky transistors**
      - **slow less-leaky transistors**
    - » **enables eDRAM**
    - » **also helps with power issues**
      - **leakage is a big deal**
      - **only use fast transistors on the critical CPU path**
      - **use slow T's for non-critical path and memory blocks**
- **Current usage in transition**
  - **from high-performance SoC's to mainstream CPU**
    - » **issues do become more tricky as feature size shrinks**
    - » **but power is the nemesis so you do what you have to**

Basic DRAM Idea: bit/"mat"

Orthogonal address to save pins & cost



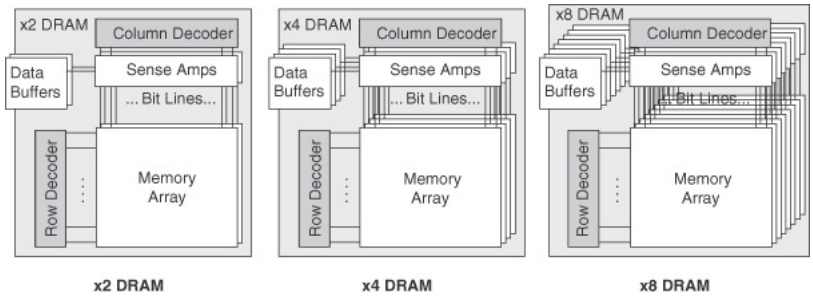
Sense amps now combined with row buffer

Storage Cell and its Access

It's All about Mats

- **DRAM devices come in several flavors**
  - **interface & speed: we'll deal with these later**
  - **width**
    - » **x4 & x8 are highest density die**
      - **used in price sensitive applications like PC's**
    - » **x16 & x32**
      - **higher per bit cost used in high performance systems**
- **DRAM chip = lot's of memory arrays (mats)**
  - **mats operate under several regimes**
    - » **unison**
      - **each access targets one bit/mat**
        - **x4 accesses 4 mats**
    - » **independent**
      - **mats organized as subsets to create banks**
        - **concurrent bank access is the idea**
      - **intra-bank mats operate in unison**
    - » **interleaved banks within a rank**

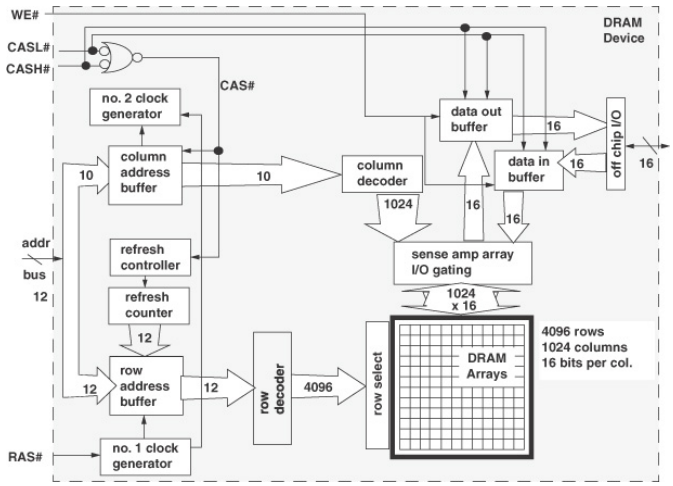
Mat & Width Organization



Slow Mat Problem

- **Mat access is slow**
  - **high-C word and bit lines**
    - » **bigger = slower**
      - **C for wire is linear in length at same width**
      - **Cgate is linear with size of row or column in the mat**
- **Interleave to speed up**
  - **mid-60's hack used on IBM 360/91 and Seymour's CDC 6600**
    - » **essentially a form of pipelining**
  - **if interface is n times faster than mat latency interleave n banks**
    - » **should be able to make things arbitrarily fast**
      - **in theory yes - in practice no**
        - **constraints: jitter, signal integrity, power**
  - **multiple on-die banks**
    - » **may be internally or externally controlled**

64 Mbit FPM DRAM (4096x1024x16)



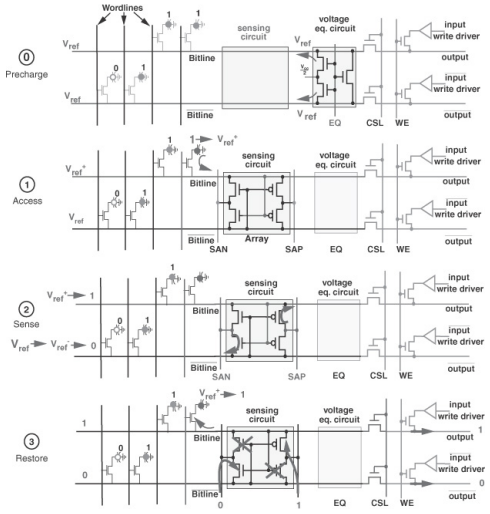
Sense Amps

- Small stored charge requires high sensitive amps
  - use differential model
    - » reference voltage precharged to half-way mark
    - » then look at which way the charge goes to determine value
      - noise margins must exist and trick is to keep them small
      - problematic as devices shrink
- Roles
  - 1: basic sense value
  - 2: restore due to the destructive read
    - » 2 variants in play
      - restore instantly or restore on row close
  - 3: act as a temporary storage element (row buffer)
    - » how temporary depends on restore choice

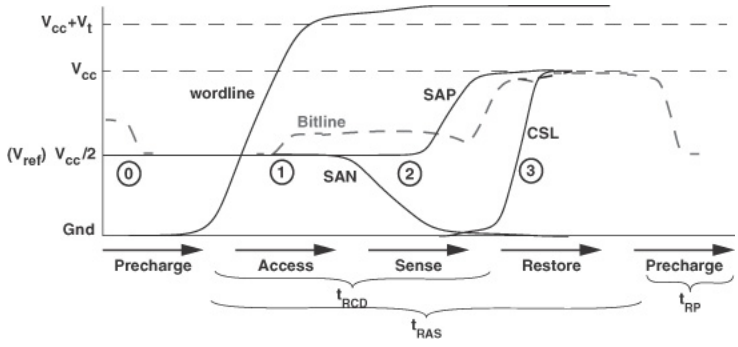
Decoders & Redundancy

- Defects occur and yields have to be high
  - rules of a low margin business
- Redundant rows, columns, and decoders
  - fuses are used to isolate defective components
  - appearance is a fully functional mat
  - fuse set
    - » burn in, test and then fuse set

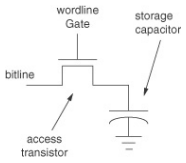
Sense Amp Operation



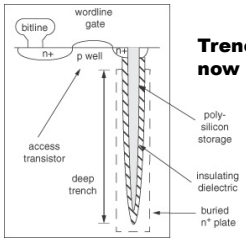
Sense Amp Waveforms



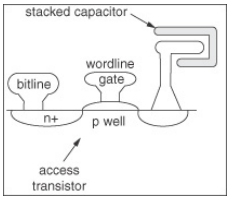
DRAM Cell



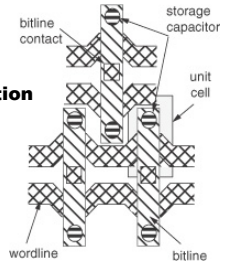
Logical View



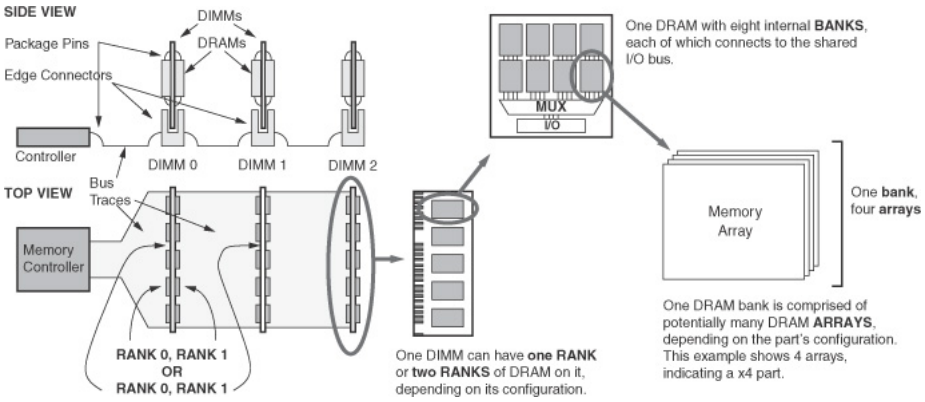
Trench implementation now primarily used in eDRAM



stacked implementation mainstream DRAM processes

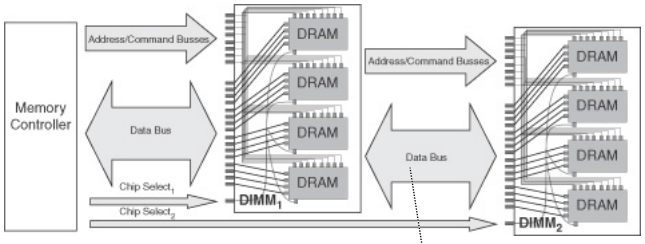


Ranks & Banks vs. DRAMs & DIMMs



JEDEC Interface

address width depends on DRAM capacity
control: RAS, CAS, Oenable, CLKenable, etc.



Chip select goes to every DRAM in a rank
Separate select per rank - 2 per DIMM common

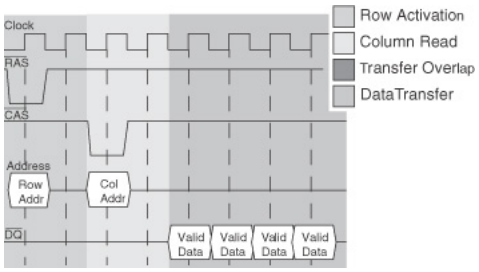
64 bits typical
wider in high-end systems

See any problems on the horizon with this model?

Memory Controller Issues

- DRAM control is tricky
  - CPU prioritizes memory accesses
    - » transaction requests send to Mem_Ctl
  - Mem_Ctl
    - » translates transaction into the appropriately timed command sequence
      - transactions are different
        - open bank then it's just a CAS
        - no open bank then Activate, PRE, RAS, CAS
        - wrong open bank then write-back and then ACT, PRE, RAS, CAS
        - lots of timing issues
      - result: latency varies
        - often the command sequence can be stalled or even restarted
        - refresh controller always wins
    - » now moving onto the CPU die
      - multi-core and multi-mem_ctl involves a lot of issues
  - Not as easy as you might guess if you want performance
    - » lots of device specific timing constraints

Simple SDRAM Timing



Note: pipelining possibilities

Timing Parameters (Micron Style)

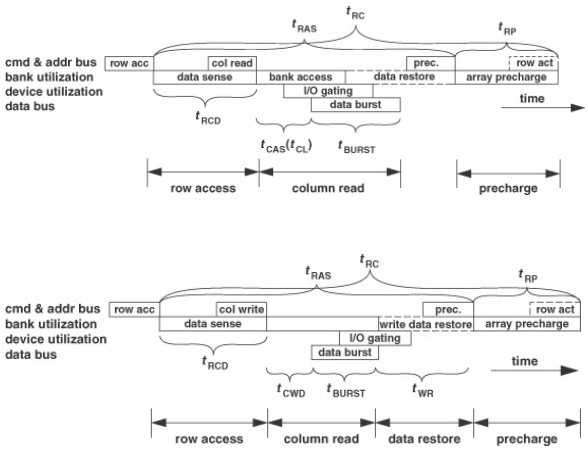
Parameter	Description
tAL	added latency to column accesses for posted CAS
tBURST	data burst duration on the data bus
tCAS	interval between CAS and start of data return
tCCD	column command delay - determined by internal burst timing
tCMD	time command is on bus from MC to device
tCWD	column write delay, CAS write to write data on the bus from the MC
tFAW	rolling temporal window for how long four banks can remain active
tOST	interval to switch ODT control from rank to rank
tRAS	row access command to data restore interval
tRC	interval between accesses to different rows in same bank = tRAS+tRP
tRCD	interval between row access and data ready at sense amps
tRFC	interval between refresh and activation commands
tRP	interval for DRAM array to be precharged for another row access
tRRD	interval between two row activation commands to same DRAM device
tRTP	interval between a read and a precharge command
tRTRS	rank to rank switching time
tWR	write recovery time - interval between end of write data burst and a precharge command
tWTR	interval between end of write data burst and start of a column read command

Minimal Timing Equations

A=row access
R=col_rd
W=col_wr
P=precharge
F=Refresh
s=same
d=different
a=any

Prev	Next	Rank	Bank	Min. Timing	Notes
A	A	s	s	tRC	
A	A	s	d	tRRD	plus tFAW for 5th RAS same rank
P	A	s	d	tRP	
F	A	s	s	tRFC	
A	R	s	s	tRCD-tAL	tAL=0 unless posted CAS
R	R	s	a	Max(tBURST, tCCD)	tBURST of previous CAS, same rank
R	R	d	a	tBURST+tRTRS	tBURST prev. CAS diff. rank
W	R	s	a	tCWD+tBURST+tWTR	tBURST prev CASW same rank
W	R	d	a	tCWD+tBURST+tRTRS-tCAS	tBURST prev CASW diff rank
A	W	s	s	tRCD-tAL	
R	W	a	a	tCWD+tBURST+tRTRS-tCWD	tBURST prev. CAS any rank
W	W	s	a	Max(tBURST, tCCD)	tBURST prev CASW same rank
W	W	d	a	tBURST+tOST	tBURST prev CASW diff rank
A	P	s	s	tRAS	
R	P	s	s	tAL+tBURST+ tRTP-tCCD	tBURST of previous CAS, same rank
W	P	s	s	tAL+tCWD+tBURST+tWR	tBURST prev CASW same rank
F	F	s	a	tRFC	
P	F	s	a	tRFC	

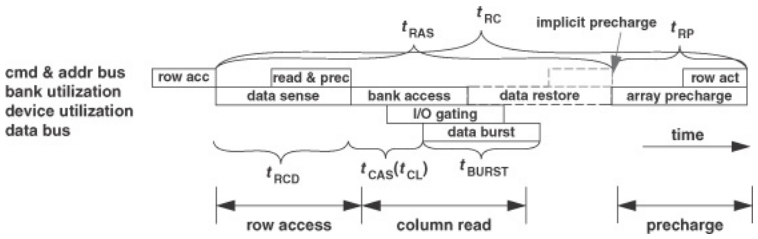
Read and Write Sequences



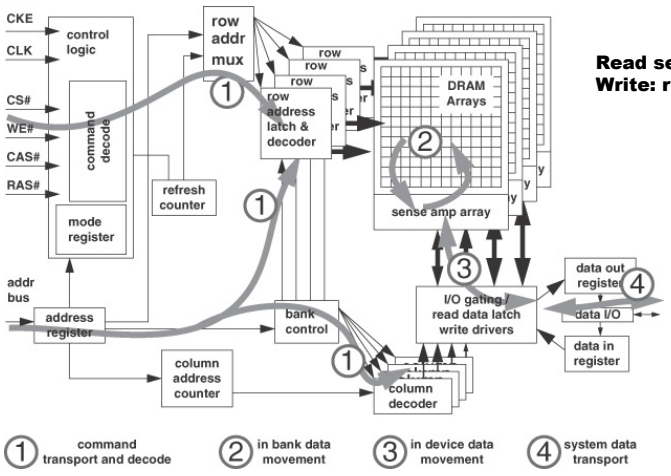
Note: % of time data bus bandwidth is utilized

Compound Commands

- **DRAM evolution**
  - **allows compound commands**
    - » **mem_ctlr options and scheduling complexity increase**
  - **column read and precharge**
    - » **use when next scheduled access is to a new row**
      - **2 commands rather than 3**
      - **timing constraints carried over however**



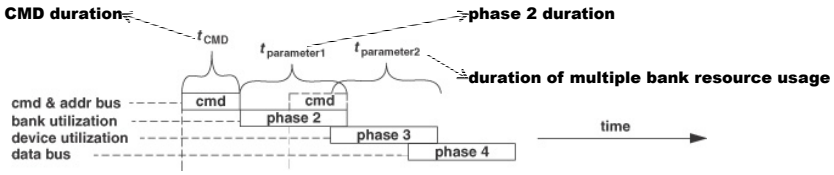
Generic Structure



Read sequence
Write: reverse 2,3,4

Abstract Command Structure

- **Reality**
  - **huge variety of command sequences possible**
    - » **all with heavily constrained timing issues**
      - **2 roles of timing**
        - **1) physical: latency, set-up and hold, signal integrity, lane retiming**
        - **2) power: limit concurrency to stay under thermal/power ceiling**
- **Start simple**
  - **command & phase overlap**



Note other overlaps - also specified by timing parameters

Mainstream Throughput Idea: DDRx

- **Use both clock edges**
  - **DDR transfers 2 bits per cycle per lane**
    - » **DDR2 transfers 4**
    - » **DDRn transfers 2^n**
    - » **signal integrity and power limit clock speeds**
      - **particularly on long FR4 wire traces**
- **Also add source synchronous clocking - enter DQS**
  - **timing variance creates synchronization issues**
    - » **DDR device uses DLL/PLL to synch with Mem_Ctl master clock**
      - **note skew depends on where the DIMM sits in the chain**
    - » **need to latch in the center of the data "eye"**
  - **other sources of timing uncertainty**
    - » **manufacturing variation, temperature, Miller side-wall effect, trace length**
      - **delay proportional to RC**
      - **power proportional to CV^2f**

Disturbing Trend

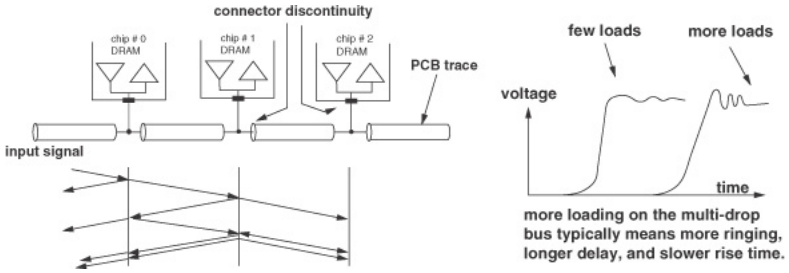
- DIMM capacity going up
  - process improvements yield more bits/die
- DRAM channel speed going up
  - DDRn
- # of DIMMs per channel going DOWN!!
  - SDR - 8 DIMMs/channel
  - DDR - 4 DIMMs/channel
  - DDR2 - 2 DIMMs/channel
  - DDR3 - 1 DIMM/channel and higher latency
    - » isn't this a lower bound?
    - » adding channels is expensive in CPU pins
      - remember mem_ctl is on chip now and for good reason
  - Why?
    - » stub electronics problem on a JEDEC broadcast bus
    - » gets worse if bus speed increases - it's the dI/dt thing
- Problem essence
    - » not enough memory capacity per socket
    - » huge server problem today

Signal Integrity

- Increasingly limiting in shrinking processes
  - gets even worse
    - » as speeds increase
    - » as trace length increases
- Multi-drop wires are a problem
  - very difficult to achieve perfect transmission line behavior in practice
    - » impedance changes with
      - temperature
      - manufacturing variability
      - L & C effects of the neighborhood
      - signal reflections
  - result is signal distortion
    - » made worse by noise
      - also a neighborhood problem
- DRAM systems
  - traces are long, and broadcast is the norm
    - » intra- and inter-device

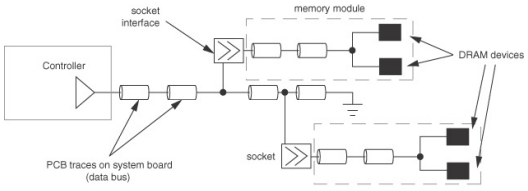
Multi-Drop Bus Complications

- **Result**
  - **as speeds increase**
    - » **#DIMMs per channel decrease**
    - » **delay added by slow rise time and let ringing settle**
      - **hmm - faster means more delay - huh?**
  - **socketed DIMM connector adds another discontinuity**
    - » **socket - PCB trace - connector - DIMM trace to DRAM die**



Other Complications

- **Skew**

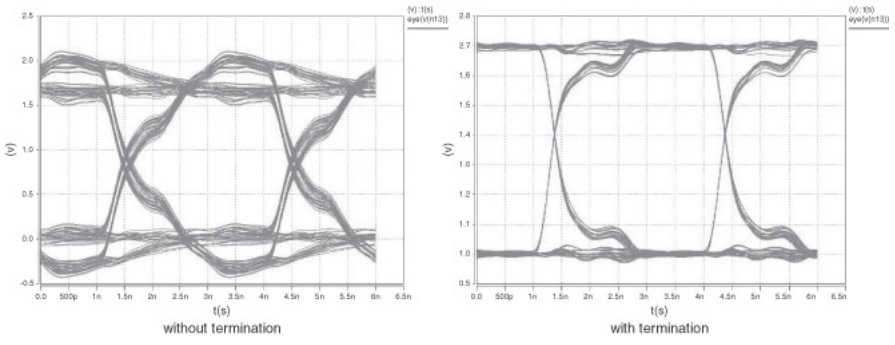


- **Jitter**
  - **small fluctuations in signal propagation velocity due to**
    - » **termperature, supply voltage, etc**
- **Inter-Symbol-Interference (ISI)**
  - **L & C induced cross-talk**
- **Bottom line**
  - **lots of practical barriers to increasing signal speed**

Termination

- **Key to minimizing reflections**
  - **but DRAM needs to be cheap**
    - » **cheap SOJ and TSOP packages**
      - **large pin C & L's - mismatched to trace impedance**
      - **OK for low freq - < 200 MHz**
  - **faster requires smaller pins ==> BGA (DDR) & FBGA (DDR2/3)**
- **Another termination issue**
  - **impedance inside vs. outside the package need to be isolated**
    - » **series termination (DDR)**
      - **damps internal DRAM component reflection effects on the DIMM trace**
    - » **programmable on die parallel termination (DDR2)**
      - **higher speeds ==> tighter reflection constraints**
      - **configuration register controls termination resistor switches**
      - **removes need to time for worst case configurations (max DIMMs)**

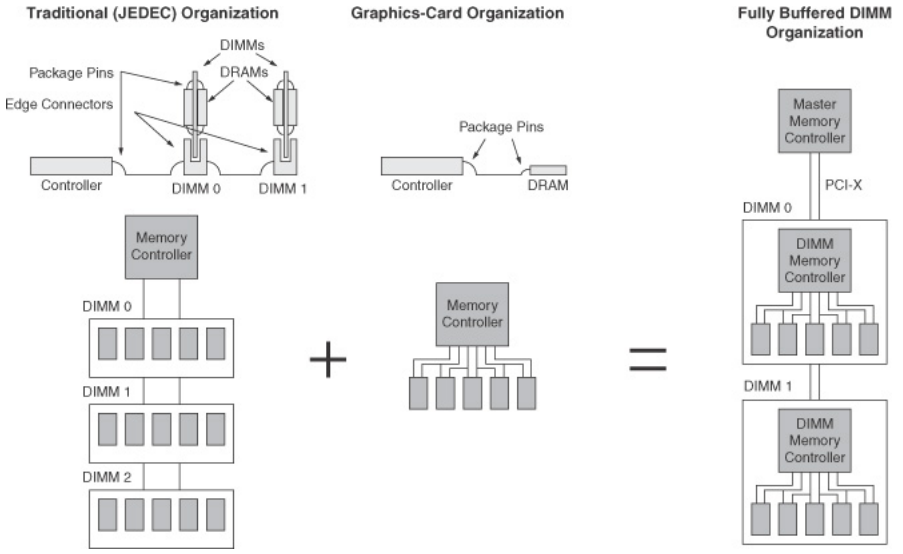
Termination: Eye Doctor



Voltage Issues

- **Low voltage swing**
  - **saves power and potentially improves speed**
  - **BUT: reduced noise immunity**
    - » **so do differential signalling**
      - **CACTI did this for all of your HW4 experiments**
    - » **problem - DRAM's have to be cheap**
      - **can't afford 2x data pins**
- **Vref**
  - **provide a common voltage reference used by all inputs**
    - » **adv: x+1 < 2x pins for interesting values of x**
    - » **disadv: lose the common mode rejection of differential**

Intel's FB-DIMM Compromise



FB-Dimm Problems?

- There are many
  - daisy chain causes varied response time
  - bit lane retiming additional latency problem
- Already considered a 1-trick pony
- Enter BoB - Buffer on Board - the new Intel hack
  - use a tree rather than a daisy chain for 4x DDR3
  - BoB placement
    - » motherboard or on a memory card riser
  - problem - another buffer stage in the memory hierarchy
    - » OK if prefetch strategy is working for you
- AMD has/had? a similar variant
  - Socket 3 Memory Extender (G3MX) micro-buffer
    - » effort now seems to have been cancelled

DRAM Systems Issues

- Power and Heat
  - the biggest concern now and in the future most likely
    - » early DIMMs consumed about 1W
    - » FB-DIMMs now at 10W
- Servers
  - goal
    - » 3x more channels and 8x more DIMMs per channel
  - looks like 250 W per socket just for memory
    - » huge problem now
  - definite time for a rethink
    - » problem
      - Industry momentum
      - standards
      - DRAM commodity ==> super low margins
        - rethinking is a costly proposition

Leakage & Refresh

- **Transistors are not ideal switches**
  - **leakage currents in DRAM processes are minimized**
    - » **but not to 0**
  - **leakage currents increase as Tsize goes down**
    - » **tricky balance of Vth, Vdd, and process**
    - » **additional increase with temperature**
  - **industry target - refresh every 32 - 64 ms**

Refresh Trends

- **t_{RFC} is going up**
  - **decreases availability ==> slower system memory**
  - **vendor choice**
    - » **keep inside the 64 ms refresh period**
      - **even though the number of rows goes up**

Family	Vdd	Device Capacity Mb	# Banks	# Rows	Row Size kB	Refresh Count	t_{RC} ns	t_{RFC} ns
DDR	2.5V	256	4	8192	1	8192	60	67
		512	4	8192	2	8192	55	70
DDR2	1.8V	256	4	8192	1	8192	55	75
		512	4	16384	1	8192	55	105
		1024	8	16384	1	8192	54	127.5
		2048	8	32768	1	8192	~	197.5
		4096	8	65536	1	8192	~	327.5

Other Refresh Options

- **All have control overhead**
  - **usually pushed to memory controller**
    - » **since device vendors need to minimize $/bit**
      - **device could do it**
        - **classic cost-performance dilemma**
- **Separate bank refresh**
  - **allow a bank to be refreshed**
    - » **while other bank accesses are still allowed**
      - **bandwidth win since memory bus can still be active**
      - **peak power win since 1 RAS on command bus at a time**
      - **mem_ctlr schedule gets harder**
  - **next step**
    - » **only refresh what is going to expire**
      - **huge scheduling problem - probably too hard**

DIMMs and DRAMs

DRAM chip type	DIMM Stick Type	Bus Clock Rate (MHz)	Memory Clock Rate (MHz)	Channel Bandwidth (GB/s)	non-ECC Channel Width	ECC Channel Width	Prefetch Buffer Width	Vdd	Read Latency Typical (bus cycles)	DIMM pins
DDR-200	PC-1600	100	100	1.6	64	72	2	2.5	2-3	184
DDR-266	PC-2100	133	133	2.133	64	72	2	2.5	2-3	184
DDR-333	PC-2700	167	167	2.667	64	72	2	2.5	2-3	184
DDR-400	PC3200	200	200	3.2	64	72	2	2.5	2-3	184
DDR2-400	PC2-3200	100	200	3.2	64	72	4	1.8	3-9	240
DDR2-533	PC2-4200	133	266	4.267	64	72	4	1.8	3-9	240
DDR2-667	PC2-5300	167	333	5.333	64	72	4	1.8	3-9	240
DDR2-800	PC2-6400	200	400	6.4	64	72	4	1.8	3-9	240
DDR3-800	PC3-6400	100	400	6.4	64	72	8	1.5	?	240
DDR3-1066	PC3-8500	133	533	8.53	64	72	8	1.5	?	240
DDR3-1333	PC3-10600	167	667	10.67	64	72	8	1.5	?	240
DDR3-1600	PC3-17000	200	1066	18.06	64	72	8	1.5	?	240

Additional Constraints

- Power - it's the biggest problem as things get "better?"
- Rules
  - first rule - things must work
  - second rule - things must get faster
  - third rule - devices must protect themselves
    - » Intel learned this the hard way
    - » for DRAM this is enforced via timing constraints
- Row activation in the main culprit
  - K's of bits moved to the sense amp latches
    - » question is how much of them do you use
      - multi-core land indicates a cache line
        - for large num's of cores
- Remember
  - large current profile changes
    - » cause timing delays
      - bit lane jitter depends on Vdd
      - Ohm's law V = I/R
        - not just a good idea - it's the law

Double Edged Sword

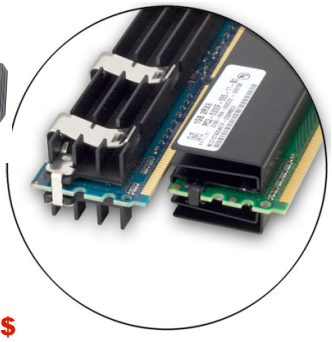
- Active power
  - $P_a = aCV^2f$
- non-adiabatic chargine regime
  - ~.5P given off as heat
    - » the other half is returned to the power supply
    - » Vdd variations on the power lines are an issue
      - also supply tolerance to high variance loads is a design issue
        - requires over provisioning
  - higher temps increase passive P component
- Faster is better
  - except for power since both f and a go up
    - » hence so does P and leakage
      - leakage impacts resource availability
      - can't ignore refresh and the 64 ms standard target

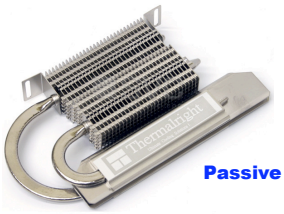
Hot DRAMs & Packaging

source: random web photos

Heat spreaders: DDR 1st step

Fins and Fans: DDR2 and beyond





Passive heat pipes

$$$ Ka-ching $$$

Memory Controller Requirements

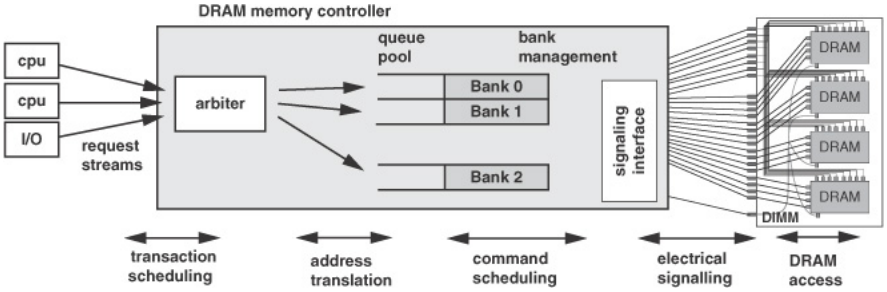
- Manage data movement to/from DRAM
  - device level
    - » electrical & timing restrictions
    - » error correction
      - typical parity just means retry and flag
  - system level
    - » arbitration fairness
      - will be necessary in multiple core/mem_ctlr configurations
    - » maximize system performance
      - command scheduling
      - multiple conflicting performance metrics however
        - heat, power consumption, latency, bandwidth
- Lots of options increase complexity
  - variety of timing parameters & command sequences
    - » specific to the target device
  - scheduling for some optimality target
    - » lots of queuing theory applies here

Top-Level View

- 3 top-level policy/strategies
  - row buffer management policy
  - address mapping scheme (MC, channel, rank, bank, row, col)
    - » what's the right swizzle?
  - memory transaction and command ordering strategy
- Large body of research
  - partially due to huge timing differences
    - » processors get faster & DRAM is fairly flat
  - seems to be reported primarily by the circuit community
    - » according to recent look by Dave and Manu
      - ISPLED – Int. SymP. on Low Power Electronics and Design
    - » and a bunch of reference cores put out by industry
    - » main game played by northbridge chipset vendors

Basic MC Components

- Note
  - as memory access cost increases w.r.t. compute on CPU's
    - » combining transaction and command scheduling is important
  - address translation targets rank and bank
    - » transaction turned into a series of DRAM commands
      - optimization options occur with interleaved transactions
        - while still respecting device timing restrictions



Row Buffer Management

- Open-Page
  - good
    - » both temporal and spatial locality exist in access pattern
      - spatial: amortizes large row activate energy cost
      - temporal: energy to keep row open results in improved bandwidth
        - latency limited by t_{CAS} only
  - bad
    - » energy: active row but no accesses
    - » time: precharge, activate, access if target row is inactive
      - better to perform a col-rd-precharge command when new row is known
  - scheduling issues
    - » similar to dynamic instruction issue
      - performance increases with a larger window
        - except when window is always slightly filled
        - multi-core/MC changes the probability
      - dependent and anti-dependent issues must be tracked
        - note write buffer in XDR (sound familiar?)

Concluding Remarks

- Whirlwind tour – phew!
- Take homes
  - understand role of MC, channel, rank, bank, row & column
  - large mat delay & broadcast commands
    - » MC role is to overlap commands optimally
    - » best bandwidth → keep data bus active
    - » open and closed row scheduling policy idea
  - challenges for the future
    - » signal integrity limits bus speed
    - » cpu pin count limits channel width
- Multi-core and improved process technology
  - only makes things worse
  - more compute power → higher memory pressure
    - » caches help and are critical
    - » but they can't catch everything
  - power is and will continue to be a fundamental constraint