Memory System Design Analysis

Reference: “Memory Systems: Cache, DRAM, Disk
Bruce Jacob, Spencer Ng, & David Wang
Today's material & any uncredited diagram came from Chapter 16

Conflicting Constraints

• DRAM industry cost vs. system performance
  • Moore's law hasn't been the norm for DRAM's
    » widening memory gap
    » 7% CAGR latency improvement
    » read bandwidth doubles every 3 years
Optimize

• For what metric
  • capacity
  • cost
  • performance
  • reliability

• For what workload
  • wide variance on memory pressure
    » OLTP, EMBC, SPEC, …
  • if caches & prefetching work
    » then things aren’t so critical

• CPU
  • OOO hides latency better than in-order
  • several memory ordering options, etc.

• Mem_Ctrl
  • scheduling policy impact

Workload Traces

• 3 basic transaction types
  • instruction fetch
  • data read
  • data write

• Key things to measure
  • rate at which various requests occur
  • isolated or bursty nature
    » bursty is the culprit
      • since CPU is much faster than main memory
**Gzip**

- Simple and caches work
  - 1 memory transaction/1000 instructions

**Gcc**

- Branch heavy – lots of fetches
  - No clear loop behavior
Fortran 77 mgrid Solver

- **Outer loop behavior repeats**
  - large data – small code so almost no fetches

Workloads

- **Measurement**
  - bus traces
    - traditional
    - both FSB and DRAM bus
    - now – memory controller moves on board
      - count and time L2 misses
        - possible probe effect

- **Duration**
  - billions of cycles usually necessary
    - usual heat up the cache issues
    - plus longer run times tend to capture the outer loop behavior
      - mgrid example
Analysis

• Efficiency
  • data transport / (data transport + overhead)
    » e.g. in a perfect world
    » all delay would be actually moving data
  • DRAM’s complicate this due to
    - command overhead
    - the large variety of timing constraints and the induced delay
    - multiplexing the RAS/CAS address
    - refresh
  • standards influence (DDRx, FB, Rambus)
  • some delays can be hidden
    - built in parallelism and a good scheduling policy

• 3 key limiters for DDR
  • inter-command constraints
  • row activation constraints
  • per-rank row activation constraints

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tAL</td>
<td>added latency to column accesses for posted CAS commands</td>
</tr>
<tr>
<td>tBURST</td>
<td>data burst duration on the data bus</td>
</tr>
<tr>
<td>tCAS</td>
<td>interval between CAS and start of data return</td>
</tr>
<tr>
<td>tCCD</td>
<td>column command delay - determined by internal burst timing</td>
</tr>
<tr>
<td>tCMD</td>
<td>time command is on bus from MC to device</td>
</tr>
<tr>
<td>tCWD</td>
<td>column write delay, CAS write to write data on the bus from the MC</td>
</tr>
<tr>
<td>tFAW</td>
<td>rolling temporal window for how long four banks can remain active</td>
</tr>
<tr>
<td>tGST</td>
<td>interval to switch DDT control from rank to rank</td>
</tr>
<tr>
<td>tRAS</td>
<td>row access command to data restore interval</td>
</tr>
<tr>
<td>tRCD</td>
<td>interval between accesses to different rows in same rank + tRAS + tRP</td>
</tr>
<tr>
<td>tRCC</td>
<td>interval between row access and data ready at sense amps</td>
</tr>
<tr>
<td>tRFC</td>
<td>interval between refresh and activation commands</td>
</tr>
<tr>
<td>tRP</td>
<td>interval for DRAM array to be precharged for another row access</td>
</tr>
<tr>
<td>tRRD</td>
<td>interval between two row activation commands to same DRAM device</td>
</tr>
<tr>
<td>tRTIP</td>
<td>interval between a read and a precharge command</td>
</tr>
<tr>
<td>tRRSW</td>
<td>rank to rank switching time</td>
</tr>
<tr>
<td>tWR</td>
<td>write recovery time - interval between end of write data burst and a precharge command</td>
</tr>
<tr>
<td>tWTR</td>
<td>interval between end of write data burst and start of a column read command</td>
</tr>
</tbody>
</table>
### And Command Scheduling Constraints

<table>
<thead>
<tr>
<th>Prev</th>
<th>Next</th>
<th>Rank</th>
<th>Bank</th>
<th>Min. Timing</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>A,a</td>
<td>a</td>
<td>A</td>
<td>a</td>
<td>tRC</td>
<td>A,a = any DRAM for all RAS same rank</td>
</tr>
<tr>
<td>R</td>
<td>a</td>
<td>A</td>
<td>a</td>
<td>tBF</td>
<td>tRRD plus tFAW for 5th RAS same rank</td>
</tr>
<tr>
<td>W</td>
<td>a</td>
<td>A</td>
<td>a</td>
<td>tBF</td>
<td>5th RAS unless posted CAS</td>
</tr>
<tr>
<td>F</td>
<td>a</td>
<td>A</td>
<td>a</td>
<td>tBF</td>
<td>8th RAS always based on Prev</td>
</tr>
<tr>
<td>P</td>
<td>a</td>
<td>A</td>
<td>a</td>
<td>tBF</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prev</th>
<th>Next</th>
<th>Rank</th>
<th>Bank</th>
<th>Min. Timing</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>d</td>
<td>A</td>
<td>a</td>
<td>IBURST + 10 TBE</td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>W</td>
<td>d</td>
<td>A</td>
<td>IBURST + 10 TBE</td>
<td></td>
</tr>
</tbody>
</table>

**Overhead =** \( \frac{(\text{minT}-\text{BURST})}{\text{BURST}} \)

### Efficiency vs. DRAM B/W

- based on simple Gzip trace
- open – F - #ranks - #banks
  - \( F = \text{Fifo, open page} \)
Gzip Bandwidth vs. DRAM B/W

Vortex Workload Differs
Averaging Multiple Workloads

Closed Page Systems

- 2 command ordering regimes
  - F = Fifo = no reorder
  - reorder Q of depth 4
    - notation
      - open – F - #ranks - # banks
      - close – 4 - #ranks - #banks
  - Note
    - command reordering has an even bigger effect on open page system performance
      - question is whether the workload issue rate & locality take advantage of it
      - downside is significantly more power consumption
    - for closed page systems
      - power disadvantage tends to disappear
      - question is what is the performance impact
Command Scheduling

- Bank round robin
  - cyclic schedule different banks, then change rank
- Rank round robin
  - cycle through ranks then banks
- Wang Rank Hop
  - alleviates \( t_{\text{FAW}}, t_{\text{RRD}}, \) and \( t_{\text{RTRS}} \) effects
  - distributes row activation commands to alternate ranks
  - while grouping CAS commands to an active rank
- Greedy
  - others depend on logical sequence
  - this one has per bank command queues
  - issues the one with the least wait time

Schedule Impact

Greedy is best, b/w increases w/ Q depth (minor) & increased burst size (major)
% gain past Q depth of 10-12 is modest
Note: greedy does not guarantee fairness, the others are fair
Sensitivity

- Short bursts and % of reads for DDR2 and DDR3
  - bo4 is burst length of 4 beats (8 is possible)
  - DRAM design optimizes for the common case

![Graph showing Maximum Sustainable Bandwidth (% of reads) vs. Read Transaction Percentage]

- 1.33 Gb/s DDR3 system is highly sensitive to bo4 ratio
- 800 Mb/s DDR2 less sensitive to bo4 ratio
- Basic channel config (2 ranks)
- Greedy scheduling
- Queue depth 6

DDR3 Wrinkles

- Burst chop in DDR3
  - 8-bit prefetch in DDR3 gives an 8-bit burst
    - if 4 bits are needed a burst chop mechanism is used
      - however the extra 4 beats are idled
      - hence the empty space can't be pipelined
  - hence the sensitivity in DDR3 to short burst transactions
    - currently cast in concrete in the JEDEC standard

- Proposals in the works
  - short write burst bank switching (WBS)
    - fill the short burst tail from another bank
    - writes are uncommon plus conflicts with the read b/w optimization
      - sim studies show this is a bust
  - short read-write burst bank switching
    - fill shorts by alternating read and write shorts in same rank
    - 14% gain w/ 100% short bursts and 90% reads
  - neither proposal is compelling
FB vs. DDR2

- **FB issues**
  - commands and write data share the southbound lane
  - but max bandwidth is 25% greater than a multi-rank DDR2 system

- **FB Latency**
  - increasing ranks from 1-2 has a big impact for both DDR2 & FB
    - due to reduced contention even though $t_{OST}$ comes into play for DDR
  - further increases
    - little impact on DDR2+
      - bus contention is the bottleneck
    - marginal improvement in moving from 2-4 ranks
    - no improvement after 4 ranks

Conclusions 1

- **Sustaining high bandwidth utilization**
  - harder w/ each successive generation due to:
    - relatively constant row cycle times
    - increasing b/w and shorter data transport times
    - higher overhead
    - for DDR3 where power is a concern
      - $t_{CAS}$ and $t_{RRD}$ constraints are more severe

- **Is there a fix for this conundrum**
  - e.g. the DRAM vendors are tightly bound
    - cost and performance conflict
    - cost has dominated the equation
    - standards both make and break the business
  - most recent gains have come from increased mem_ctlr sophistication
    - adds complexity for sure
    - latency is particularly sensitive to Q'ing delays
Conclusions 2

• Where do we go next
  • should vs. will is the question
    » will: Intel pushing FB and BoB_whatever
    » should: probably rethink the whole DRAM space
      * power is rapidly becoming cost
      * performance still matters of course
  • performance w.r.t. the CPU
    » more memory controllers on die – likely
      * probably a requirement in the multi-/many-core regime
    » BUT
      * CPU pin count is flat
      * per pin bandwidth is flat
  • personal view
    » the real culprit is wires – try something else
    » silicon nanophotonics is showing signs of life
      * In the ITRS now for inter-chip
      * could that mean cpu-dimm?

Conclusions 3

• That's a wrap for now on DRAM
• Lot's of hair
• Feedback appreciated
  • e.g.
    » useful
    » what should change
    » etc
• Disks are next