Negotiating the Maze
Getting the most out of memory systems
today and tomorrow

Robert Kaye
System on Chip Memory Systems

- Systems use external memory
  - Large address space
  - Low cost-per-bit
  - Large interface bandwidth required
  - Access to memory depends on accesses from other processing elements

- Challenges:
  - Manage the flow of data to and from external memory
  - Provide best bandwidth and latency characteristics to each processing element
  - Choose the memory solution that best matches the system requirements
Performance, Power & Cost

- Memory choice driven by performance, power and cost
- Relative importance of these factors varies between systems
- Available memories offer tradeoffs
  - There isn’t a panacea offering a single solution that is best for all scenarios
DRAM Choices Today

- **DDR2**
  - today’s benchmark

- **DDR3**
  - lower power than DDR2
  - higher performance
  - price cross over forecast: early 2011?

- **LPDDR2**
  - equivalent performance to DDR2
  - lower power/pin count than DDR2
  - price premium over DDR2/3

(not to scale!)
Predicting utilisation and bandwidth

MODELLING THE MEMORY INTERFACE
Memory Topology 101

- DRAMs are divided into BANKS. Typically 4 or 8 in current devices (older devices may have 2, future 4)
  - Each bank is subdivided into ROWs.
  - Each row contains multiple COLUMNS.
- To access a particular location the DRAM is sent a BRC (Bank, Row, Column) address.
- Before accessing the data the bank & row must be prepared.
Efficiency Measurement

- A bank access cycle is characterised by a number of timing parameters, and the variable transfer amount

- In order to ensure continuous streams of data, banks are accessed in an interleaved fashion

- As access length increases, efficiency improves
Utilisation Prediction

- Theoretical model used to predict utilisation and bandwidth

- Memory type & speed grade
- Interface Width
- Interface clock frequency
- Packet access length
- Bank distribution
- Read/write distribution

<table>
<thead>
<tr>
<th>Part:</th>
<th>DDR3-1600G</th>
</tr>
</thead>
<tbody>
<tr>
<td>Width:</td>
<td>16B</td>
</tr>
<tr>
<td>Frequency:</td>
<td>800 MHz</td>
</tr>
<tr>
<td>SampleHitLength:</td>
<td>32 bytes</td>
</tr>
<tr>
<td>SampleBankDistribution:</td>
<td>4 banks</td>
</tr>
<tr>
<td>Read/Write Distribution:</td>
<td>100% reads</td>
</tr>
<tr>
<td>Predicted Utilisation:</td>
<td>89%</td>
</tr>
<tr>
<td>Predicted Bandwidth:</td>
<td>2.8 GB/s</td>
</tr>
</tbody>
</table>

The graphs below show predicted utilisation for a range of SampleHitRates and SampleBankDistribution.
Bank Access & Hit Length Effects On Utilisation

Bytes per activate improves utilisation

Bank utilisation improves utilisation
Other Factors Affecting Utilisation

- Choice of memory, including interface widths
  - Different memories with different timing parameters
  - Wider interface widths & faster memories require more bytes per activate to achieve maximum efficiency
- Managing (minimising) the DQ turnaround overhead
- Predictive model assumes that memory controller can always order accesses for maximum efficiency
  - Almost always not possible: hazards, barriers, system prioritisation on requests will interfere
- A measure of memory controller quality
  - Percentage of predicted utilisation it can achieve across all system traffic profiles
Real world impact on memory interface efficiency

SYSTEM REQUIREMENTS
Quality of Service (QoS) Contracts

- **Minimum Bandwidth**
- **Minimize Latency**
- **Maximum Latency or Minimum Bandwidth**
- **Minimum Bandwidth**

### Minimize Latency Contract:
- Process requests to memory with minimum latency possible.
- Reduced latency = increased performance
- AKA “Best Effort”

### Minimum Bandwidth Contract:
- Provide master with minimum bandwidth.
- Low priority unless bandwidth is not being achieved.
- Provide additional bandwidth IF system loading permits.

### Maximum Latency Contract:
- Provide master with specified bandwidth within stated latency.
- Low priority until maximum latency approaches.
QoS Objectives

- Allocate system capacity (latency and bandwidth) to each master to meet the contract
  - Dynamically vary the priority to react to changes in bus traffic
- If there is excess capacity –
  - Allocate excess to where it can offer the most improvement
    - Usually reducing the CPU latency
    - Allocate excess to masters that can reduce performance later
- If there is insufficient capacity –
  - Remove capacity from masters that have the least impact on system performance
- Task of the memory controller
  - Meet these objectives whilst minimising the impact on memory efficiency, performance and power
System Latency

- Latency is added throughout the system in two forms:
  - **Static latency** – the delay through pipeline stages
    - Constant and specific to the path from master to slave
  - **Queuing latency** – the delay at arbitration points in the system

- Transaction delay depends on:
  - Number of transactions ahead of it in the queue
  - Rate at which they are processed

- Queue length dictated by
  - Capacity of the slave (memory type and efficiency)
  - Desired throughput

- Memory interface efficiency is a function of:
  - Queue length
  - Burst length
  - Read-write mix
  - Address distribution
Memory Controller Queue & Topology

- A queue is implemented in the memory controller
  - Allows re-ordering of transactions to maximize efficiency
  - If the queue fills it extends through the interconnect
  - Interconnect arbitration only operates when the queue extends through the interconnect
  - For effective QoS, the arbitration policy should be consistent throughout the system

- System topology influences the performance
  - CPU and LCD Ctrl placed close to the memory controller
    - Lower latency
  - Graphics and DMA on a separate level
    - Hierarchy improves performance
Memory Controller and QoS

- Memory controller utilizes QoS information passed by system
- Stratagem:
- Requests from minimum latency (best effort) contracts enter the queue with highest priority
  - Higher priority = less time in queue = lowest latency
- Requests from maximum latency (typically stream processing or real time) contracts enter queue with mid range priority
- Requests from minimum bandwidth contracts enter queue with lowest priority
Real Time Masters

- For real time masters adding latency does not affect performance
  - While latency is less than the maximum
- Real time requests carry maximum latency (time out) information into the controller queue
- If the transaction is still waiting after a time-out period
  - Promoted to highest priority
  - Only higher priority than best-effort masters when necessary

![Diagram showing survival rates and adjusted priority levels for different latency categories.]
**Batch Processing Masters**

- Average latency, bandwidth and queue length related by Little’s Law $E(L) = \lambda E(S)$
  - Hold queue length constant
  - Measure average latency and control priority
  - Priority controls latency which controls bandwidth
- Excess bandwidth is used
- Priority only exceeds other masters
  - Insufficient bandwidth obtained
  - Minimizes transactions prioritized over Best-effort masters

![Diagram showing bandwidth vs latency](image-url)
Memory Controller Design Considerations

- Many design considerations in order to achieve best possible efficiency and meet system QoS requirements

Examples
- Buffering and queuing
- System feedback
- Versatile re-ordering
  - What can be re-ordered without impacting utilisation?
  - When can utilisation considerations override QoS?
- Where, when and how to arbitrate between requests
  - Read/write prioritisation
  - Memory request scheduling

Measurement against efficiency model
- Assess alternatives
- Feedback on quality
Memory Controller Efficiency Example

- Results compare observed efficiency with predicted theoretical maximum
- Planar plot shows ideal (red grid) vs. observed
- Temperature map shows achieved percentage of ideal performance
  - Example considers >90% as pass (green) condition
- Test conditions: 50/50 read write mixture
  - Test system x32 LPDDR2-800
Memory Interface & AMBA 4

- AMBA® 4 brings changes the access pattern and controller requirements
- Coherency & snooping
  - Reduces the number of off-chip accesses
  - Increases the average hit length – higher proportion of cache fills and evictions
- QoS and Virtual Networks
- Long Bursts
- Barriers
  - Must be managed efficiently by the controller, or
  - Through blocking in the interconnect (less efficient)
Cortex-A15 in a CoreLink subsystem

- System coherency scaling to many cores
- I/O coherency reduces cache maintenance
- Low latency CPU to memory, and high bandwidth GPU to memory
- New highly efficient memory controller
  - 1/2/4 channels, up to 1066MHz
  - >90% interface utilization
  - Roadmap to future memory interfaces
- System MMU supports I/O virtualization
  - Efficient hardware access for multiple OS
**DMC-400 Dynamic Memory Controller**

- Fourth Generation ARM/AMBA Dynamic Memory Controller
  - High efficiency interface from AMBA 3 and AMBA 4 systems to shared off chip LPDDR2 and DDR3
  - Delivers the high bandwidth and low latency needed by high performance multimedia systems implementing Cortex-A15 and Mali-T604
  - Effective management of power in memory sub-system

- Complete master to memory traffic management for AMBA 4 systems
  - Designed and verified with AXI Network Interconnect and Banks CCI
  - System wide QoS
  - QoS Virtual Networks
  - Barrier Management
  - Compatible with AXI3 interconnects

- DMC-400 architecture scales to Future DRAM standards
New memory architectures are proposed to provide 4X available bandwidth without expending additional energy

FUTURE MOBILE MEMORIES
Future Mobile Memory Candidates

- **Wide-IO**: low speed (200MHz SDR), wide (4 x 128b) interface to stacked memory die (1Gb to 32Gb) through direct chip-to-chip connects

- **Serial** (aka **Low Power Serial I0**): high speed (5-8GHz), full duplex, multi-channel links to LPDDR2 core
  - Standardisation stalled – conflicting PHY specs; no consensus; no strong pull, replaced by…

- **Hybrid** (aka **Low Power Dual Mode Memory**): standard LPDDR2 at low bandwidth, high speed serial at high bandwidth over LPDDR2 bus

- **LPDDR3**: proposed evolutionary development of LPDDR2 – standard parallel DDR; spec development in progress
Wide-IO

- Achieves higher bandwidth through wider data bus: x512 I/O arranged as 4x128b channels
- Lower power through lower frequency (200MHz), SDR
- Stacking challenges: manufacturing, thermal
  - Controller and system design for multi-channel & large I/O count
- Spec finalisation target: 2011
  - Production forecast 2012-2013

Initial implementations: 1-die stack

Production – 4-die stacks
Low Power Dual Mode Memory (LPDMM)

- Hybrid parallel/serial interface using LPDDR2 pin-out
  - Parallel mode
    - LPDDR2 start-up latency and low power consumption in low bandwidth environments (single channel up to 1.6 GB/s)
  - Serial mode
    - high bandwidth (up to 6.4 GB/s) over same pins
- LPDMM challenges:
  - Hybrid PHY design / power
  - Managing parallel/serial switchover
- Spec finalisation target: 2011
- SPMT (Serial Port Memory Technology) developing solution
Comparing Wide IO & LPDMM with LPDDR2

- **LPDDR2**: Low Performance, Low Cost, High Power
- **Wide-IO**: Moderate Performance, Moderate Cost, Moderate Power
- **LPDMM**: High Performance, High Cost, Low Power

(not to scale!)
CONCLUSIONS
Conclusions

- Understanding memory interface efficiency is complex
  - Many variables and constraints
  - Getting the right solution is critical to system power, performance and cost

- System requirements can adversely effect efficiency
  - Memory controller can mitigate these when implemented as part of a system wide solution

- Many choices of memory
  - Future memory proposals will bring new design challenges

- ARM is committed to delivering memory controller solutions to enable optimal memory subsystems today and tomorrow
Thank You

Please visit [www.arm.com](http://www.arm.com) for ARM related technical details

For any queries contact < [Salesinfo-IN@arm.com](mailto:Salesinfo-IN@arm.com) >