


2011 ARM TechCon™  
*Join the community defining the future*



Chip Design

# Leveraging AMBA® 4

in Next Generation SOC Designs



Steve Hamilton  
Applications Architect  
Sonics, Inc  
hamilton@sonicsinc.com





2011 ARM TechCon™  
*Join the community defining the future*

Agenda

- ❑ What is new in AMBA4
- ❑ Long bursts
- ❑ AXI IDs
- ❑ QoS
- ❑ Summary

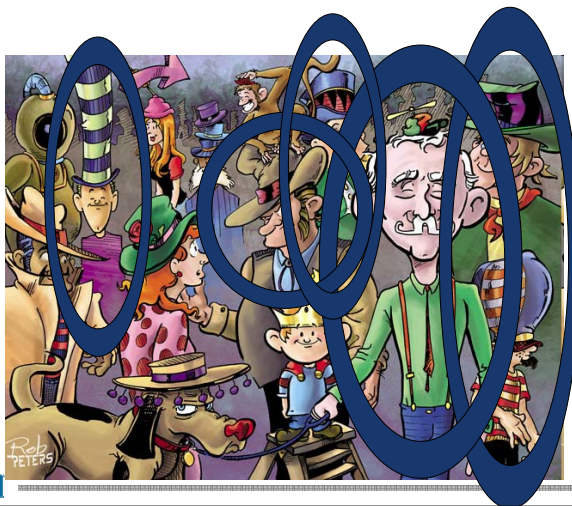
**Hamilton's Rules**



- **AMBA specifications**
  - **AMBA 4**
    - [AMBA AXI and ACE Protocol Specification](#)
    - [AMBA APB Protocol Specification](#)
    - [AMBA AXI4-Stream Protocol Specification](#)
    - [AMBA AXI Protocol Specification v2](#)

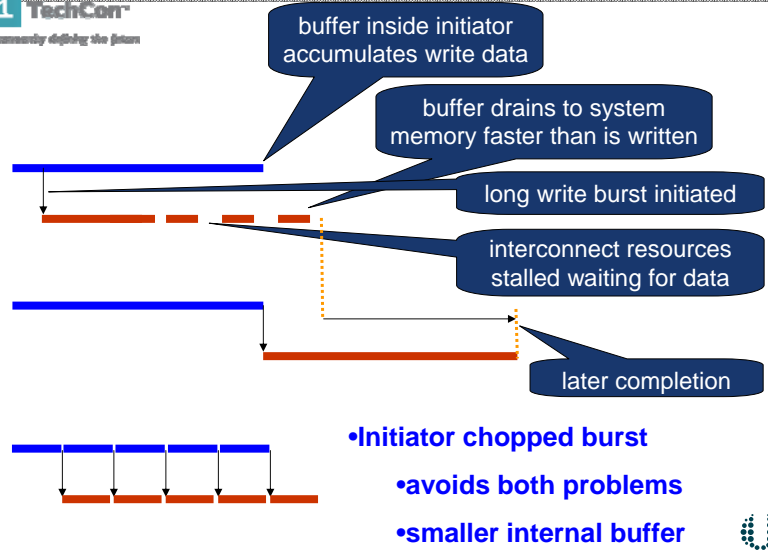
- new Stream Interface
- AHB absent
- new AXI-Lite spec
- new AXI Features
- AXI Feature removal
- AXI clarifications

- Clarifications
  - ordering, posting, memory types / cache impacts
- Removal of locked transactions
- Removal of write data interleave
- New features
  - longer bursts
  - QoS field
  - User fields



- Project Leader
- Architect
- Lead validation engineer
- SOC Integrator
- core and sub-system developers

- Can issue up to 256 beat bursts
  - performance, atomicity, simplicity
- Not guaranteed atomic beyond 16 beats
  - even in device space / non-modifiables
  - system (integrator) performance & QoS
- Atomicity may be undesirable for core as well



- Hamilton's High Performance (AMBA) Systems Rules
  - HHPS Rule #1: Don't issue long write bursts
  - HHPS Rule #2: Chop based upon address alignment

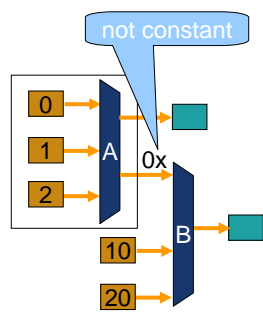
- Did not change in AMBA4
- Critical to High Performance Systems
  - Parallelism == Performance
  - Highly shared targets – especially DRAM
  - targets can exploit parallelism for performance
  - AMBA's way to identify concurrency / parallelism

- Exclusives “master ID”
- 8.2 Transfer ID Fields  
The AXI protocol provides an ID field to enable a master to issue a number of separate transactions, each of which must be returned in order.
- AMBA ID semantics
  - Different masters → no ordering
  - Same master, different ID → no ordering
  - Same master, same ID → in order completion and response
  - read and writes → different IDs

- Source Id
- Transaction Id
- Sequence Number

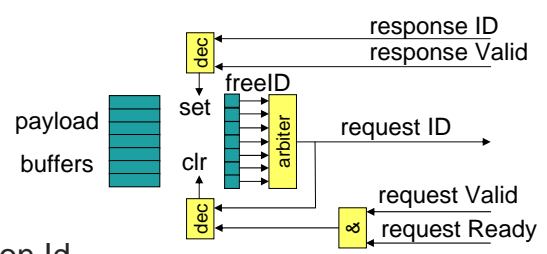
### Choice #1: Source Id

- Master Id or sub-master Id
  - Fixed source of transaction streams
    - can be used to id source - eg. security firewall
  - simple logic – constant ID
  - implies ordering required for ALL transactions from source
    - over constrains system
    - performance loss



### Choice #2: Transaction Id

- Transaction Id
  - simple logic – ID reservation system (above)
  - no ordering required for ANY transactions
    - maximum performance potential
    - explicit interlock or logic required for ordering



2011 ARM TechCon  
Join the community defining the future

## Load Balancing

The diagram shows two masters, 'master 1' and 'master 2', sending data to an 'interconnect'. Master 1 has a bandwidth of  $X$  MB/s and Master 2 has  $Y$  MB/s. The interconnect splits the traffic, sending  $.5X + .5Y$  to the DRAM controller from each master. The DRAM controller then outputs  $X$  and  $Y$  to the DRAM controller.

- Loading balanced independent of BWs / use cases
- Each master uses Transaction Ids fairly
- Integrator maps IDs to DRAM ports

ARM UBM Electronics

2011 ARM TechCon  
Join the community defining the future

## Choice #3: Sequence Number

The diagram illustrates a sequence number mechanism. A 'ctr' (counter) provides a 'request ID' to a 'set' of request IDs. The 'set' is controlled by 'set' and 'clr' signals. A 'dec' (decoder) outputs 'request Valid' and 'request Ready' signals. The 'request Ready' signal is also connected to another 'dec' block. The 'request ID' is also connected to a 'freelD' block. The 'response ID' and 'response Valid' signals are also connected to the 'dec' block.

- Sequence Number
  - transaction IDs
  - ID sequence indicates issue order
- Useful for high performance hazard resolution

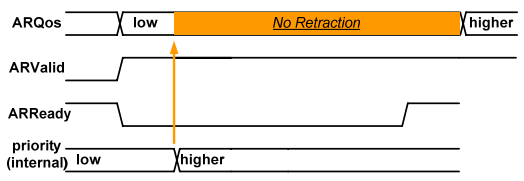
ARM UBM Electronics

- HGPS Rule #3:
  - Use IDs as Transaction IDs (or sequence numbers) to precisely express the ordering required by the initiator
  - (sequence numbers can preserve ordering hints for resolution of address hazards and write observability if the posting point how to interpret them)

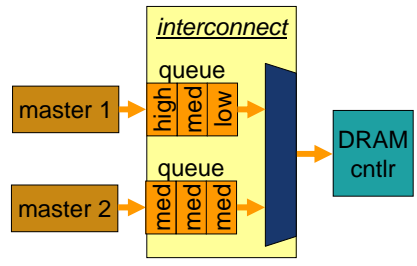
- Labeling of request with QoS class
- Semantics “not specified”
- spec “preferred use” (and ARM QoS products)
  - 16 levels of arbitration priority
  - dynamically managed by initiator
  - “programmable”
- Permits initiator to express “urgency” of request

- Decoupling philosophy
  - QoS/sharing is an “Integrator” issue, not “core developer” issue
- Implementation Problems

➤ retraction prohibition



➤ head of line blocking



- Mitigates Head-of-line blocking
- Still subject to panic failures
  - ignorant of Integrator preferences and allocations
- Ignorant of target state
  - contributes to down-stream panics

- Still dynamic priority scheme with sideband
- sideband flow is counter to data flow
  - defeats head-of-line blocking
- Metering target use against target allocations
  - follows integrator intentions
- aware of target state
  - avoids queuing things that cannot progress
  - lower latency

- HHPS Rule #5:
  - Do NOT dynamically manage QoS priority in the initiator
  
- HHPS Rule #6:
  - Use an interconnect that can enforce Integrator's QoS intentions / allocations (and discount Core Developer's "give me all i want!" intentions)

- AMBA4 AXI changes are aimed at making high performance systems more attainable
  
- Removal of Locked transactions avoids a big (decoupling) risk to performance
  
- Addition of long bursts (without write data interleaving) and of the QoS fields introduce new (decoupling) risks to performance
  - Use them carefully
  
- Follow Hamilton's High Performance Systems Rules