February 28 – March 1, 2012

Fabric Verification

By

Galen Blake
Senior MTS
Altera Corporation

Steve Chappell
Solutions Architect
Mentor Graphics
What is a Fabric?

183a.gif

cisco.com

575 × 278 - The Data BUS has a lower data forwarding capacity (32 Gbps) than the switch ... Similar - More sizes
Characteristics of a Bus Fabric

- Resists soda spills and chewing gum – oops wrong fabric...
- Connects multiple masters to multiple slaves.
- Masters can issue transactions simultaneously.
- Masters may issue multiple transactions while earlier transactions are still in flight.
- Some allow transactions to complete out of order.
- Performance impacted by data width and latency.
Verifying a real AMBA3 Switch Fabric

• Basic features must be tested:
  – Check protocol conformance on each port.
  – Check each master can talk to each slave.

• Basic features aren’t enough - these must also be verified:
  – Architectural parameters are correct
  – Bus arbitration on shared segments
  – Conditions that might lead to poor performance
  – Conditions leading to bus stalls or poor latency
  – Performance (bandwidth and latency)
Fabric Verification Goals

- Exhaustive protocol checking
  - Per port, supported features may differ
- Connectivity tests
  - Not a full cross-bar, non-trivial connectivity map
- Arbitration checking
  - Proper algorithms, no starvation scenarios
- Traffic scenario coverage
  - Sparse, Normal, Heavy and variable traffic loads,
- Performance measuring/validation
  - Bandwidth and latency monitoring under all conditions.
Fabric Topology

- Protocol Variety makes traffic very difficult to control.
Verification IP

- Will replace peripherals for enhanced controllability and synchronization.
- OVM-native VIP for all AMBA3 protocols (AXI, AHB, APB)
- Protocol monitors, coverage collection, importable test plan
- Support for directed, random, and graph-based sequences
- Compatible with acceleration tools (e.g. Veloce TBX) for really long tests
- Used Questa Verification IP
VIP Connections

- AMBA VIPs provide more control and monitoring
Beyond Constrained Random

- Constrained Random was a leap forward in verification productivity but there are limitations.
- Variables and constraints are sprawling.
- Constraint solvers are proprietary, not consistent.
- Important corner cases are left to chance.
- Coverage models required to determine results.
- If three 7’s pay out at 800:1, how many times must the constraint solver “lever” be pulled to cover 777?
Graph Based Coverage

- Achieves most efficient functional coverage
- No unnecessary repetition
- Provides clear visualization of state space
- Easier definition/review of coverage metrics
- If there are 32 “coverage bins” on this apparatus, cover all of them in 32 “spins”.
- Don’t gamble with your coverage.
Developing a Graph

• A particular protocol or coverage space is captured with a simple grammar into a single file.
• It is then compiled into a graph.
• The graph is reviewed and annotated.
• It gives the user strong visual cues about what is covered and what is not covered.
• The annotations inform as to the scale and practicality of the space defined.
• If necessary the user can make informed decisions to limit the scale to a manageable size.
• Alternatively, users can cover full scale large graphs by using coordinated farm servers or acceleration resources.
AXI Rules --> AXI Graph

```c
1 /*
 2  axi_full_protocol.rules
3 */
4 /*
5  Copyright 2011 Mentor Graphics Corporation. All Rights Reserved
6 */
7 #rule_graph axi_full_protocol {
8  import "impl_axi_param_defn.reg";
9  import "impl_axi_full_protocol.proto_reg";
10  import "impl_axi_full_protocol_burst_control.reg";
11  // various symbolic constants omitted for brevity...
12  // various symbolic definitions omitted for brevity
13  RV_options = [read or write]AXI_TRANS_WRITE | rand_data write_strobes) | read or write[AXI_TRANS_READ];
14  SelectBurstSizeLength = burst
15  if (burst == AXI_WRAP) (wrap_boundary | if (burst == AXI_WRAP) eta
16  burst_length = size;
17  UnlockedCacheProtOpts = unc_unlocked_cache_prot_start cache_prot unc_unlocked_cache_prot_end;
18  LockedCacheProtOpts = unc_locked_cache_prot_start cache_prot unc_locked_cache_prot_end;
19
20  case
21     UnlockedTrans =
22       lock[AXI_NORMAL, AXI_EXCLUSIVE]
23       addr_reg address
24       select_sd | if (EM_SAME_ID -- 1) eta
25     SelectBurstSizeLength
26     // constraint read or write c assures that an exclusive read is selected in RV_options
27     RV_options
28     UnlockedCacheProtOpts
29     do item
30     protocol unc_end unlocked
31     if (lock == AXI_EXCLUSIVE)
32        excl_transaction_gap // let excl read transaction register with QOR monitors
33        read or write[AXI_TRANS_WRITE] // exclusive access must be ended with a write
34     rand_data write_strobes
35     do item
36      lock
37     ) | if (lock == AXI_EXCLUSIVE) eta ;
38
39  LockedTrans =
40     lock[AXILocked]
41     addr_reg
42     select_sd | if (EM_SAME_ID -- 1) eta
43     lock_length
44     repeat required
45     SelectBurstSizeLength
46     RV_options address
47     protocol unc_end locked
48     LockedCacheProtOpts
49     do item
50     update_lock_length
51     lock_transaction_gap ;
52
53     // TOP of graph
54     axi_full_protocol - init repeat {
55        external_sync
56        create_item
57        protocol unc_start
58        LockedTrans | UnlockedTrans
59        check_env
60        transaction Gap ;
61    }
62"
```

Fabric Verification - Blake/Chappell
AHB Graph

- These graphs will cover the full spectrum of their respective protocols.
- Full protocol compliance is covered efficiently in a few hundred thousand clock cycles.
- Graphs can stop or continue looping to cover more address or data ranges generating long streams of useful bus traffic.
Graph-based Sequences & API

- Graphs drive OVM sequences for VIPs.
- Graphs drive CPU traffic from a C API.
Traffic Control and Modulation

- The goals are to first operate the fabric under normal and expected traffic conditions.
- Next, we want to put the fabric under as many conditions of duress as possible.
- We need to determine if there are any sets of traffic conditions that cause the fabric to:
  - Slow down to unacceptable performance
  - Block out certain masters or slaves
  - Lock up the system.
- Traffic modulation allows us to shape the traffic into scenarios that match light, normal or heavy traffic patterns.
Traffic Coordinating Graph

- Each protocol graph can be controlled by the traffic control graph.
- The traffic control graph can continuously adjust settings for each master to modulate local traffic.
- The traffic control graph can take over and precisely control the local graph giving it the ability to synchronize traffic on each master.
Traffic Control and Modulation

Traffic Control

CPU

axi

s

m

AMBA Fabric

axi

AXI VIP

axi

AXI VIP

axi

AXI VIP

ahb

AHB VIP

Scoreboard

mon

mon

mon

mon

mon

mon

DDR

AXI VIP

AXI VIP

APB VIP

Fabric Verification - Blake/Chappell
Traffic Scenarios

• This diagram shows a normal traffic scenario.
• Bus activity is dense during a buffer transfer.
• Bus activity goes quiet while waiting new for transfers to start.

• This diagram shows a heavy traffic scenario.
• There is very little quiet time between large transfers.
Traffic Control Scenarios

- The traffic control graph allows us to target specific features and performance of the fabric as a fabric instead of treating it as an array of ports.
Conclusion

• We have already proven the effectiveness of several components of this solution resulting in improved designs and schedules.

• As we scale up the small traffic control example shown here to the full chip level we expect continued design improvements and confidence.