Traffic Profiling and Performance Instrumentation For On-Chip Interconnects

Mark Peryer – Mentor Graphics, Bruce Mathewson - ARM

Overview and problem statement

Interconnects are the critical path in SoC design
- Functionality has to be correct
- Have to meet performance requirements

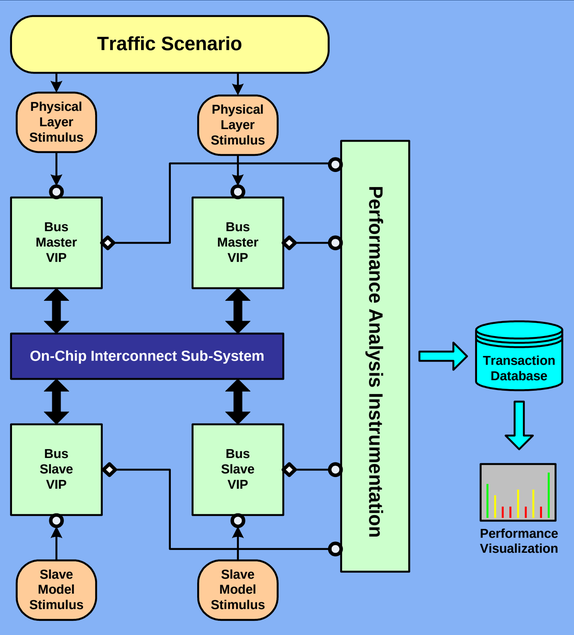
Interconnect functionality verification is a bound problem and can be solved efficiently using graph based techniques. Performance validation is an unbound problem especially with the trend to platforms where the end application is unknown.

Performance Validation Approaches
- Spreadsheets
  - Good first level approximation to budgets
- ESL/SystemC
  - Good for modelling the HW-SW architecture
  - Difficulty with building accurate interconnect models
- With the real design, running application code
  - Cannot happen until late in the development cycle
  - Difficult to generate and execute sufficient representative scenarios

Another significant issue is a lack of agreed performance metrics which means that even if the stimulus can be generated it is harder than it should be to analyze the interconnects performance capabilities.

The Approach We Have Explored
- Treat the interconnect RTL as a sub-system
- Use traffic generation to represent IP and SW activity
- Use abstraction for performance scenarios
- Develop performance analysis instrumentation and metrics

Traffic Generation And Analysis Environment

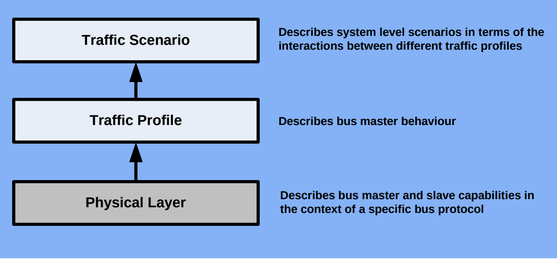


The traffic generation and performance analysis environment is represented by the diagram above. The salient features are:
- UVM testbench environment with bus interface VIPs
- Stimulus generation from an abstract Traffic Scenario
- Transaction level performance instrumentation
- Transaction DB for in, or post, simulation performance visualization

This environment allows complex use case scenarios to be quickly defined, run and analyzed to determine whether the interconnect has the required performance. The interconnect RTL can then be regenerated to address any short-comings and quickly re-validated within the environment before being integrated into the SoC.

ARM – Mentor Traffic Profile Proposal

On-chip bus traffic can be represented by 3 abstraction layers
- Physical Layer
  - A bus protocol (AXI3/4, ACELite, ACE)
- Traffic Profile
  - A description of a bus masters behavior
- Traffic Scenario
  - One or more interacting traffic profiles



The Physical Layer Descriptor

The physical layer descriptor is used to describe the bus transfer characteristics of a target design. This information can then be used to configure a VIP to reproduce a range of low level behaviors that the design would produce.

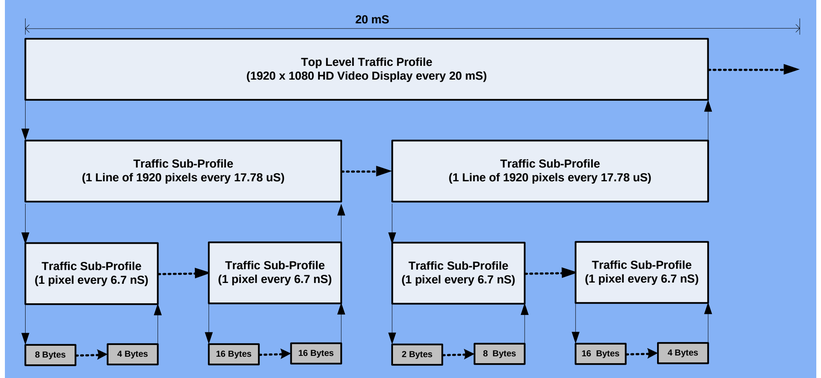
Certain aspects of the physical layer descriptor can be adjusted to allow "what-if" experimentation. The content of the physical layer descriptor is described in the table below:

Characteristic	Description	Variable
Bus Type	Which bus protocol	Fixed
Field Widths	Affects address range and data transfer quanta	Fixed
ID Width	Determines number of parallel transactions	Fixed
Burst Length	How many burst beats the master can support	Variable
Limitations	Sub-set of protocol supported	Variable
Primary timing	Timing parameters that directly affect performance	Variable
Secondary timing	Timing parameters with set defaults	Firm
Percent Error	Proportion of error responses returned by slave	Variable

The Traffic Profile Descriptor

The traffic profile descriptor describes the traffic that a bus master would generate. It specifies:
- The volume of data to be transferred
- The direction in which the data should be transferred
- The time to the next iteration of the profile
  - Also acts as a check that the profile has completed

The traffic profile descriptor can be broken down into any number of hierarchical sub-profiles, which again have periodicity. This is illustrated in the diagram below where the data fetch for a HD video display is described in terms of a traffic profile with two sub-profiles. The lowest layer of the traffic profile is delegated to the physical layer.



The parameters of the traffic profile descriptor are summarised in the following table:

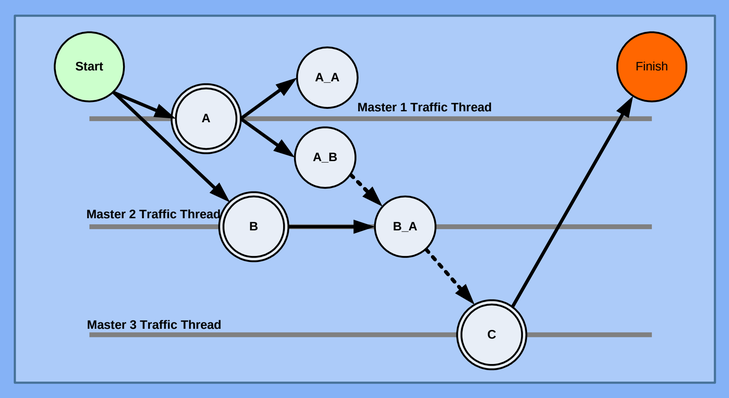
Parameter	Purpose
Name	Label used to identify the descriptor
Address Range	Determines when the traffic address wraps
Direction	Read, Write, R/W mix
Size	Number of bytes to be transferred
Stride	Offset in bytes to next occurrence
Period	Time to the next occurrence
Sub-profile(s)	List of sub-profiles. If empty, executes physical layer transactions

The Traffic Scenario

Traffic scenario combines the traffic profile descriptors together with the physical layer configuration to allocate traffic generation activity to the bus master VIPs in the environment. The activity between parallel traffic profiles can be synchronised by specifying trigger points.

The following inter-traffic profile relationships can be described:
- Dependencies
  - One activity has to complete before another
- Concurrency
  - More than one activity in progress

Traffic Scenario Example



In the above example, traffic profile A starts on master 1, and traffic profile B starts on master 2. Sub-profile B_A of B waits for sub-profile A_B of A to complete before starting. In turn, traffic profile C executes on master 3, but waits for sub-profile B_A of B to complete before starting.

Variations on this theme include having multiple copies of a traffic profile running on the same master, and repetitions of a traffic profile.

On-Chip Performance Metrics

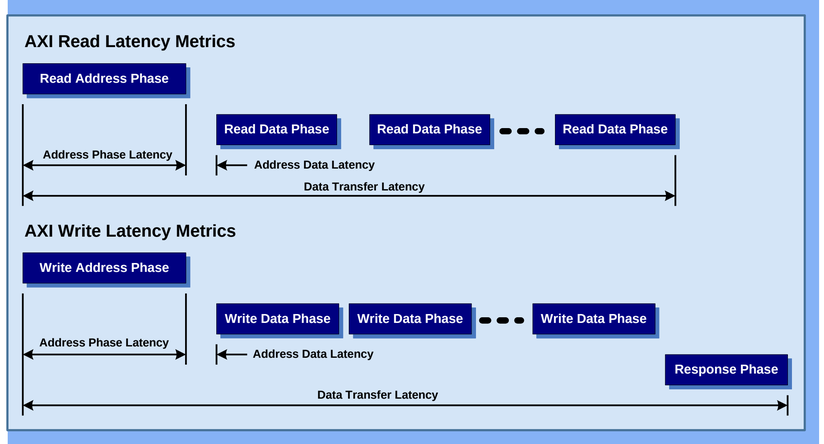
In order to interpret the behavior of an interconnect and determine that it meets its performance goals we propose the metrics in the following table:

Metric	Description
Address Phase Latency	Time for address phase to complete
Address data latency	Time from the address phase to the first data transfer
Data transfer latency	Overall time taken to complete a transfer
Data channel occupancy	% time data channel is busy
Outstanding accesses	Number of outstanding parallel accesses on channel
Data transfer bandwidth	Data Volume/Time either: • Instaneous • Windowed average • Overall average

For the ARM AXI protocol, the latency metrics for read and write data transfers are illustrated in the timing diagram below.

An AXI bus master can control:
- The rate at which read and write address phases occur
- The rate at which write data phases occur

An AXI bus slave controls:
- The rate at which data read phases occur
- The time to generate a write response



In addition to these latentcy and bandwidth metrics, there are a number of other performance metrics which can be calculated from the interconnect transaction data base contents for a traffic scenario – for instance:
- Between each master and slave
  - The number of read and write transfers
  - The min, max, and average latencies (as above)
  - The min, max and average bandwidth
- For each master:
  - The proportion of master traffic to each slave
- For each slave:
  - The proportion of slave traffic by originating master

DVCon 2013
Design & Verification Conference & Exhibition

mark_peryer@mentor.com, bruce.mathewson@arm.com

ARM® Mentor Graphics®