DVCon 2015 Paper

Automated Performance Verification to Maximize your ARMv8 pulling power

Author: Nick Heaton – Cadence Design Systems nickh@cadence.com Tel: +44 7876 198909
Co-Author: Simon Rance – ARM Ltd Simon.Rance@arm.com Tel: +1 (214) 585 3210

Introduction

This paper will introduce a fully featured ARMv8 mobile SoC design describing the key IP components, their salient features with specific emphasis on aspects that influence system performance and how they can be assembled into a CPU Subsystem. The challenge of tuning these components in order to maximize the system performance is described in detail including the introduction of performance characterization and the definition and use of example workloads for measuring, analyzing and debugging performance issues. Results from simulation of these various performance tests are introduced and discussed illustrating a systematic approach to ensuring maximum SoC performance is delivered.

ARMv8 CPU Subsystem Challenges

Multi-core, multi-cluster big.LITTLE^{TM} systems using the ARM Cortex^{TM}-A57 and Cortex-A53 typically contain a complex mix of high performance IP including GPU, High-Definition Display drivers and Video engines. In a mobile application the choice of DDR standard can have a big influence on cost and performance and in general performance is sacrificed to keep costs and power consumption down. However, regardless of which DDR option is chosen it is desirable to maximize the utilization of the memory system for the multiple, simultaneous demands placed upon it by the wide range of high-performance cores in the SoC.

Analyzing the system performance and tuning the infra-structure fabric, such as ARM Corelink^{TM} CCI-400 and Corelink^{TM} NIC-400 interconnects, in combination with the DDR controller presents designers with a big challenge. These challenges come in many forms; firstly the architecture needs to be defined, then high-level decisions need to be taken for example which IPs will be I/O coherent and hence need to have paths to memory passing through the cache-coherent interconnect and others which don't. Clock speeds and the clock domains of the various components need to be defined and the asynchronous bridges between these domains needing to be sized.

A UVM based approach is introduced as part of a systematic approach to performance characterization and performance tuning using synthetic traffic generators to model simulated system workloads. By defining workloads for specific system use-cases, corner case performance limitations can be found, understood and potentially eliminated.

The CPU Subsystem

Figure 1 shows a block diagram of the reference ARM v8 CPU Subsystem used throughout this paper. It contains 2x clusters of processors in a big.LITTLE configuration, one cluster contains 4x ARM Cortex A57 (big) processors, the second cluster contains 4x ARM Cortex A53 (little) processors. Each cluster is isolated in a separate Dynamic Voltage and Frequency Scaling (DVFS) domain to allow the system to optimally deliver the processing performance needed at a minimal power point. DVFS allows Voltage and Frequency to be raised or lowered in combination when more or less computing performance is needed, higher frequency will need a higher operating voltage and vice-versa.

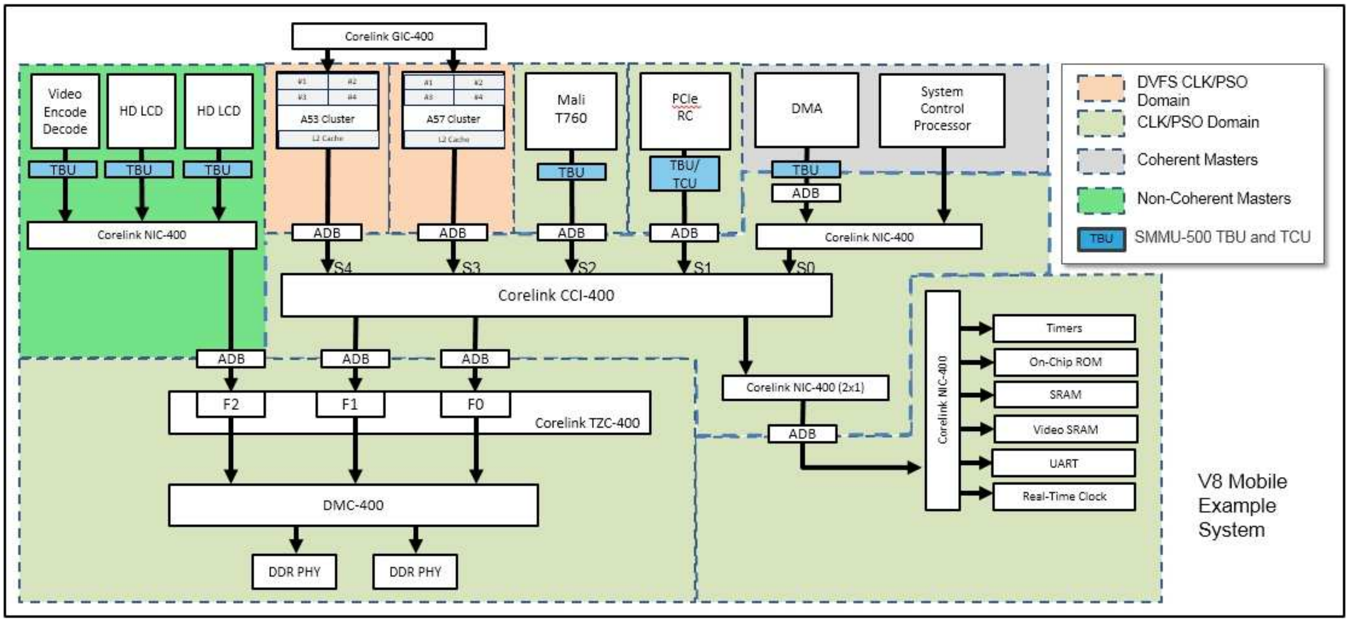


Figure 1: ARM v8 CPU Subsystem

The diagram shows the different clock and power domains as well as the AMBA Domain Bridges (ADB) that are needed when connecting two RTL modules in different domains. While the ADBs don't change the function they do introduce latency and they need to be sized as they contain FIFO structures to buffer transactions. The master IPs such as Video, LCD Controllers, GPU, PCIe, DMA and System Control Processor are all connected together and to the DDR memory system using a number of ARM Corelink System IP components.

The coherent domain in the system is managed by the CCI-400 Cache Coherent Interconnect which provides snooping support to allow the big.LITTLE clusters to share L2 Caches, but also provides snoop support for other I/O coherent masters such as the GPU to utilize the L2 caches in the clusters. The coherent domain is extended using a NIC-400 interconnect and in addition a NIC-400 is used to provide non-coherent access for three of the masters to connect to the DDR.

Two other components sit between the masters and the DDR controller, the first is the Corelink TZC-400 Trustzone Controller which provides a memory protection scheme for the system and the Corelink SMMU-500 System Memory Management Unit. The SMMU provides single or dual stage address translation to provide hardware support for virtualization. The SMMU is a distributed IP which is highlighted in blue in the diagram and comprises a number of Translation Buffer Units (TBU) connect to a Translation Control Unit (TCU) which provides common page-table walking for the Translation Look-aside Buffers (TLB) in each of the TBUs. Each TBU is connected to the TCU using an AXI streaming interface which is not shown on the diagram for simplicity sake.

Configuration Challenges

Many of the IPs in the infrastructure have a substantial number of configuration options, the NIC-400 interconnects for instance have configurable interfaces where performance related parameters like bus-width, read issuing, write issuing and Quality of Service (QoS) priority levels are set. For each AMBA interface the user can specify dynamic regulators to control bandwidth or latency, they automatically adjust QoS levels between defined limits based on how well the needs of the interface are being serviced. If for example the latency of transactions has got larger the user can configure the interconnect such that it's QoS value will be increased and increased until it is given sufficient priority to get the bandwidth/latency service it needs.

Similar configuration options need to be defined for the CCI-400, the ADB-400 components too need configuring to specify the FIFO depths inside them. An additional consideration is whether the system will use QoS Virtual Networks (QVN). This is an additional capability provided when architecting systems which contain QVN capable DDR controllers.

One of the most common ways to manage traffic prioritization is to use a DDR controller with a large number of AMBA interfaces and configure the DDR controller to prioritize the different interfaces. However this poses significant layout challenges as each AMBA AXI interfaces contains hundreds of signals and routing congestion becomes a limiting factor when performing SoC layout.

QVN alleviates the congestion problem by enabling multiple virtual channels to operate over one physical AMBA AXI interface. For example four virtual connections can be routed across one physical AXI connection. Further system-wide choices that need to be made such as clock speeds, bus-widths and power domains cut across all of the other IP configuration options presenting the designer with a bewildering array of choices.

The SMMU-500 has a number of additional configuration options that can dramatically affect the performance of the system. Inside each TBU is a TLB which needs to be sized, the user needs to define the number of entries in the TLB ideally to match the needs of the IP requiring translation. The decision on sizing the TLBs needs to take account of the likely page fault rate and the cost (in performance terms) of a page fault. This is complicated by again many choices in the page table setup choices.

Approach

Given all of the options described in the previous section it is paramount that a systematic approach is taken when selecting, configuring and assembling the system IP into a CPU subsystem. In order to get accurate results for analysis the chosen approach is to run RTL simulation and leverage Universal Verification Methodology (UVM) to create standard testbenches using a standardized High-Level Verification Language (HVL) SystemVerilog. In addition to the benefits of using a standard methodology and language, UVM enables the use of commercial grade Verification IP (VIP).

The chosen approach is to replace the master IP in the design, for example the processor clusters, LCD Controllers and Video engine with AMBA VIP. This has a double benefit, VIP is much more controllable than IP and hence it is far simpler to create very tightly defined AMBA traffic. Secondly, it generally improves simulation performance for large complex IP the most significant of which are the processor clusters. Figure 2 illustrates the concept where “Active” VIP is used to model AMBA ACE, ACELite and AXI4 Master IP and “Passive” VIP is used to monitor AMBA AXI4 Slave interfaces.

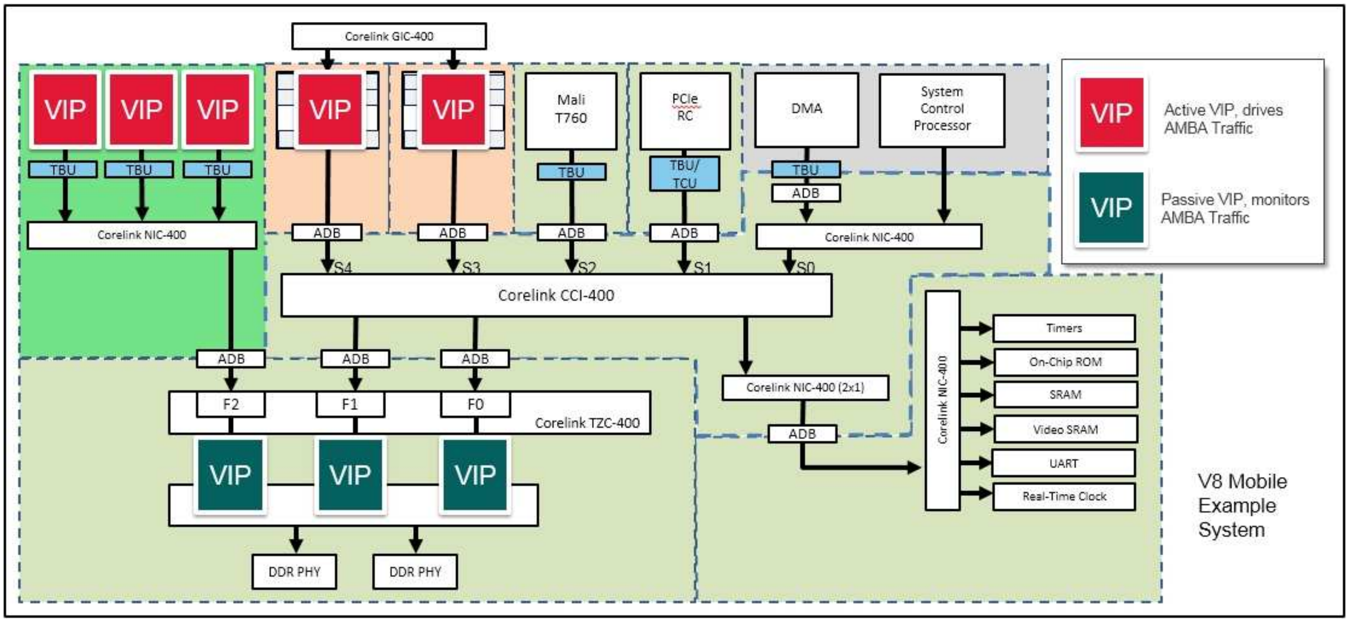
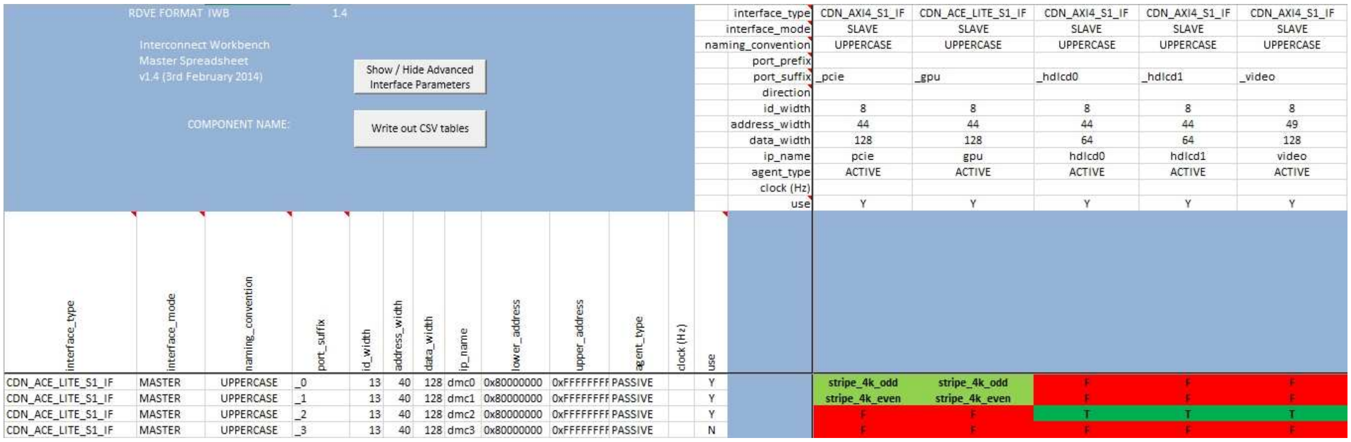


Figure 2 shows the conceptual UVM testbench

Using the Master VIP, traffic can be driven into the subsystem and used to target paths through all of the infrastructure IP. In order to drive legal paths through the system the testbench needs to have some additional information about the legal routes through it, this is provided by a routing model which allows tests to query paths either to or from any master to or from any slave. An additional benefit of this testbench structure is that tests can be written in a totally portable way. For example a test can be written which selects a single slave and systematically drives traffic from all the legal masters that are able to drive traffic to that slave.

Creating these testbenches can be achieved manually however substantial productivity gains can be made using testbench automation as provided by Cadence Interconnect Workbench (IWB). By defining input meta data using a Microsoft Excel Spreadsheet a testbench to perform subsystem verification and performance analysis can be automated. Figure 3 shows an example of the spreadsheet used to create testbenches for the CPU subsystem.



RDVE FORMAT IWB 1.4

Interconnect Workbench
Master Spreadsheet
v1.4 (3rd February 2014)

COMPONENT NAME:

interface_type	CDN_AXI4_S1_IF	CDN_ACE_LITE_S1_IF	CDN_AXI4_S1_IF	CDN_AXI4_S1_IF	CDN_AXI4_S1_IF
interface_mode	SLAVE	SLAVE	SLAVE	SLAVE	SLAVE
naming_convention	UPPERCASE	UPPERCASE	UPPERCASE	UPPERCASE	UPPERCASE
port_prefix					
port_suffix	_pcie	_gpu	_hdlcd0	_hdlcd1	_video
direction					
id_width	8	8	8	8	8
address_width	44	44	44	44	49
data_width	128	128	64	64	128
ip_name	pcie	gpu	hdlcd0	hdlcd1	video
agent_type	ACTIVE	ACTIVE	ACTIVE	ACTIVE	ACTIVE
clock (Hz)					
use	Y	Y	Y	Y	Y

interface_type	interface_mode	naming_convention	port_suffix	id_width	address_width	data_width	ip_name	lower_address	upper_address	agent_type	clock (Hz)	use					
CDN_ACE_LITE_S1_IF	MASTER	UPPERCASE	_0	13	40	128	dmc0	0x80000000	0xFFFFFFFF	PASSIVE		Y	stripe_4k_odd	stripe_4k_odd	F	F	F
CDN_ACE_LITE_S1_IF	MASTER	UPPERCASE	_1	13	40	128	dmc1	0x80000000	0xFFFFFFFF	PASSIVE		Y	stripe_4k_even	stripe_4k_even	F	F	F
CDN_ACE_LITE_S1_IF	MASTER	UPPERCASE	_2	13	40	128	dmc2	0x80000000	0xFFFFFFFF	PASSIVE		Y	F	F	T	T	T
CDN_ACE_LITE_S1_IF	MASTER	UPPERCASE	_3	13	40	128	dmc3	0x80000000	0xFFFFFFFF	PASSIVE		N	F	F	F	F	F

Figure 3: Spreadsheet for testbench automation

The spreadsheet defines buses by protocol, signal name, location in the design hierarchy and many more items for both masters and slaves and a routing table which can also include details about memory striping, a feature of the Corelink CCI-400 cache coherent interconnect.

The testbench automation provided by IWB uses this meta data to generate a testbench containing all the AMBA VIP configuration as well as Interconnect Validator, a system scoreboard that provide comprehensive coherency checks, data consistency checks and also records performance data for analysis. In addition IWB generates a suite of performance characterization tests which cover path by path maxBandwidth, minLatency, Outstanding Transaction sweeps and DDR Characterization tests. For example Figure 4 shows the results of driving write bursts with a sweep of burst lengths from the big Cluster to the striped DDR memory.

The test generates saturating transactions, in other words write bursts are driven into the system to the maximum write issuing level that the interconnect is capable of accepting. As can be seen as the burst length increases the corresponding bandwidth increases. Remember that only one path is active in any of these characterization tests and hence these figures are the best that the subsystem can ever achieve. The goal of these characterization tests are to find basic assembly and configuration bugs. For example clock configuration; write issuing configuration; data bus width configuration; DDR choice; all of these class of bugs can be found through this kind of automated test.

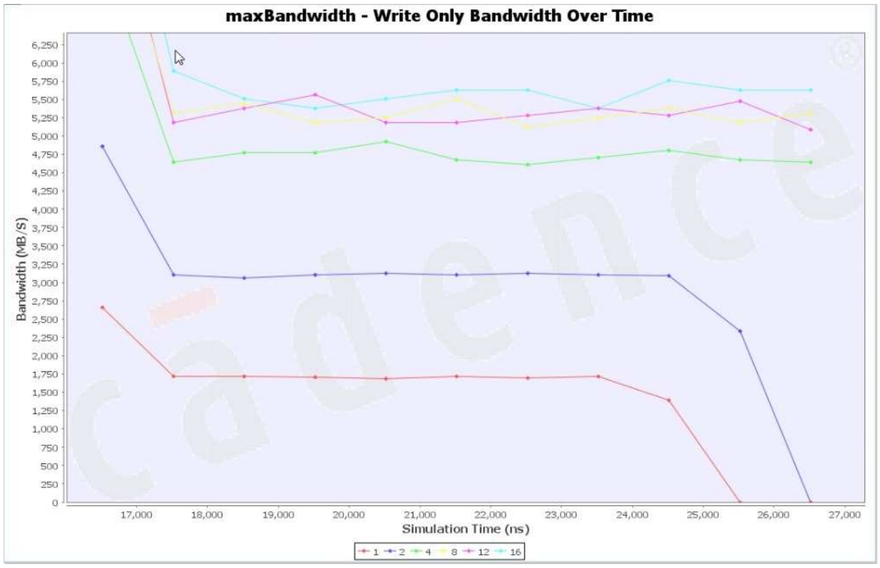


Figure 4: Write maxBandwidth from big cluster to DDR

Once characterization has been completed and all bandwidth and latency goals are comfortably met in single master -> slave paths, the next challenge is to create more realistic scenarios with multiple active masters generating AMBA traffic workloads that more closely resemble likely models of system operation.

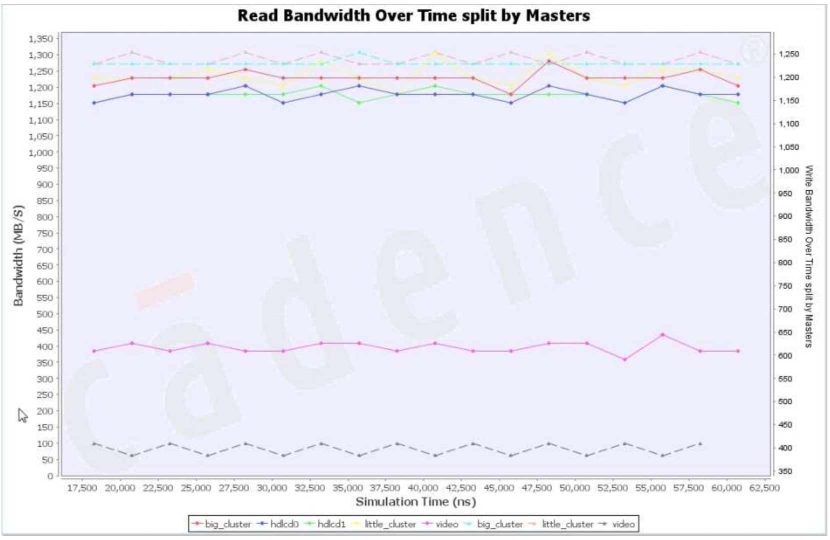


Figure 5: Read/Write Bandwidth per master

Figure 5 shows an example of a medium intensity workload. The chart shows the bandwidth that each of the masters achieves, it matches the requested bandwidth defined by the scenario. For example the VIPs standing in for the HD LCD controllers are both configured as if they were running 4K2Kp30 RGBa 8-bit (4-bytes per pixel), this represents a pixel rate of 297 Mhz. Although in a real system there would be gaps in the bandwidth requirement representing the blanking periods we have deliberately modelled a constant bandwidth requirement given predicting blanking across two displays is unlikely to align and the hence the worst case scenario is both displays driving pixels. Also notice how the LCD displays only have read bandwidth and they do not contribute any write bandwidth. In this scenario we can see that the big.LITTLE clusters generate 1.25 Gbytes/s of both read and write traffic; all the traffic is constrained to operate on 64 byte transactions as this represents a cache line. The roughly horizontal lines on the chart indicate that each and every master is getting the bandwidth they are demanding, hence the system can cope with this setup comfortably. This is confirmed when we look at the latency of transactions and figure 6 shows the latency of write transactions as well as the outstanding transaction values of the same masters.

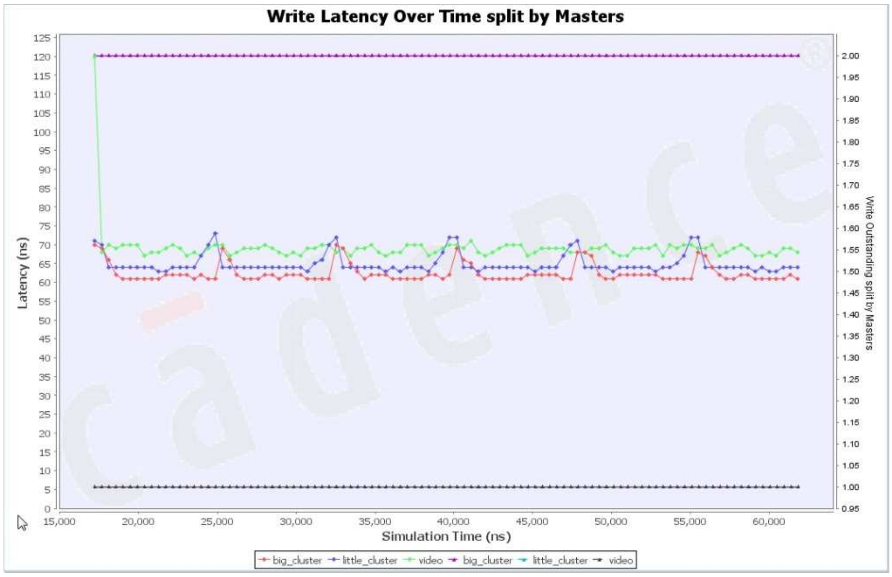


Figure 6: Write Latency with OT overlayed

Outstanding transactions are ones that have performed an address phase which has been accepted by the interconnect but are waiting for the data phase to complete. As we can see from the chart the latency of transactions is consistently under 75ns and the outstanding transaction levels never go above 2 for the big.LITTLE clusters and just 1 for the video. The outstanding transaction level is a great proxy for how much "back-pressure" there is in the system.

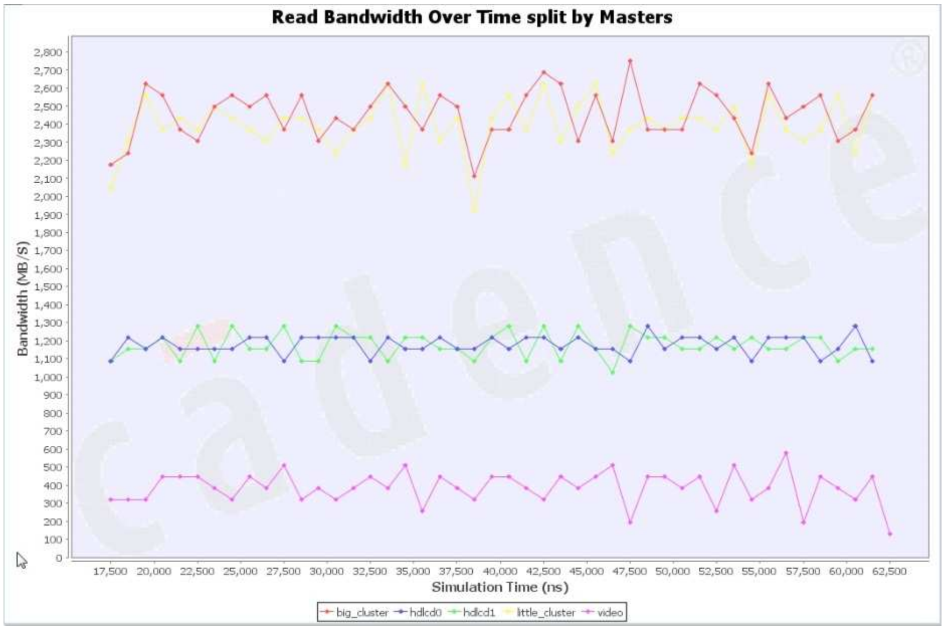


Figure 7: High-stress scenario, Read bandwidth by master

If we now look at a second scenario where the activity of the big.LITTLE clusters has been ramped up to 2.5Gbytes/s, looking at read data bandwidth in Figure 7 we see that although the cluster bandwidth is broadly achieved there is considerably more variability in the bandwidth levels.

Figure 8 however highlights that although the read levels of the system are close to being achieved this comes at the price of write bandwidth. As can be seen although requesting 2.5Gbytes/s the write bandwidth achieved is frequently below 1Gbyte/s for both big.LITTLE clusters. The spike in bandwidth towards the end of the simulation is caused by pent up demand in the masters that are being throttled by the system. Once all the read traffic has been completed the pent-up write traffic gets completed.

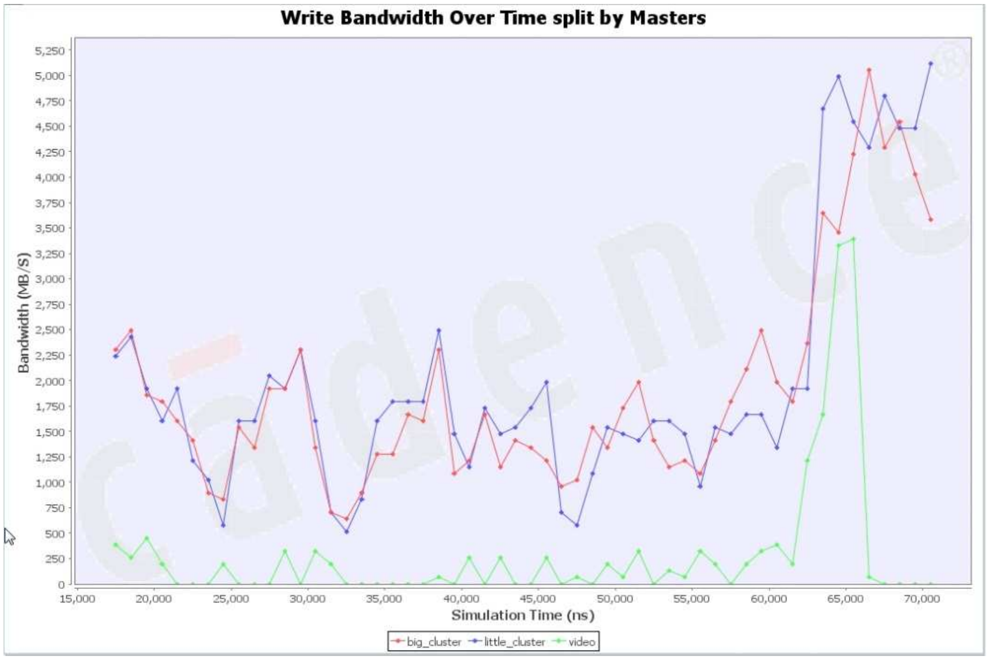
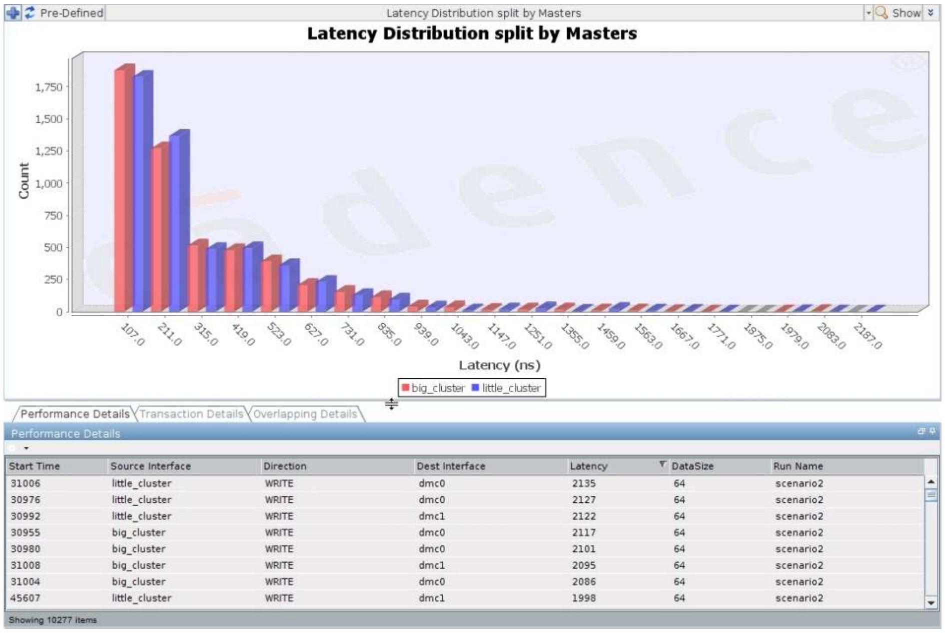


Figure 8: High stress Write Bandwidth by Master

One of the questions a system designer might ask at this point is "How bad do the read/write latencies on the big.LITTLE clusters get?"

The quickest way to get to this kind of information is to display a latency distribution filtered for just the big.LITTLE masters. Figure 9 shows the distribution as well as a table view to sort transactions by any metric, in this case latency. A distribution with a long "tail" is a strong indication that latency is out of control that the system is heavily loaded. Viewing, analyzing and debugging the system in this transactional view of the world provides the power of a TLM-like perspective but with the accuracy of RTL. These results are cycle-accurate and run on the real IP not abstract TLM models. Also, the transaction tables are linked to the signal level waveform viewer; this allows the user to jump into signal-level debug with markers to show the start and end of transactions.



Start Time	Source Interface	Direction	Dest Interface	Latency	DataSize	Run Name
31006	little_cluster	WRITE	dmc0	2135	64	scenario2
30976	little_cluster	WRITE	dmc0	2127	64	scenario2
30992	little_cluster	WRITE	dmc1	2122	64	scenario2
30955	big_cluster	WRITE	dmc0	2117	64	scenario2
30980	big_cluster	WRITE	dmc0	2101	64	scenario2
31008	big_cluster	WRITE	dmc1	2095	64	scenario2
31004	big_cluster	WRITE	dmc0	2086	64	scenario2
45607	little_cluster	WRITE	dmc1	1998	64	scenario2

Showing 10277 items

Figure 9: Latency Distribution of big.LITTLE Latency

From Figure 8 it can also be seen that the Video engine has very variable bandwidth that often dips close to zero. If we look at the Latency over time chart for this IP and overlay the Outstanding Transaction (OT) levels we can see in Figure 10 that the video engine (in green) is stalled for a considerable time and that the OT level goes up to over 20. Referring back to Figure 1 we can see that the Video Engine is connected to the DDR controller in a non-coherent way in combination with the HD LCD displays. It shares the same port which services display data. In general in a mobile system it is key that the displays don't lose data as this will cause flickering. Hence they are configured with a higher priority than other masters. In the first scenario the video got all of the bandwidth it requested, see Figure 5, however in the high intensity scenario, see Figure 10 the big.LITTLE clusters have such high bandwidth demands, double the original scenario, the video gets relegated to the lowest priority and hence it's bandwidth targets are missed and it's latency explodes by a factor of greater than 2000. The explosion of OT levels is a an excellent proxy for what is sometimes termed "back-pressure" in the system.

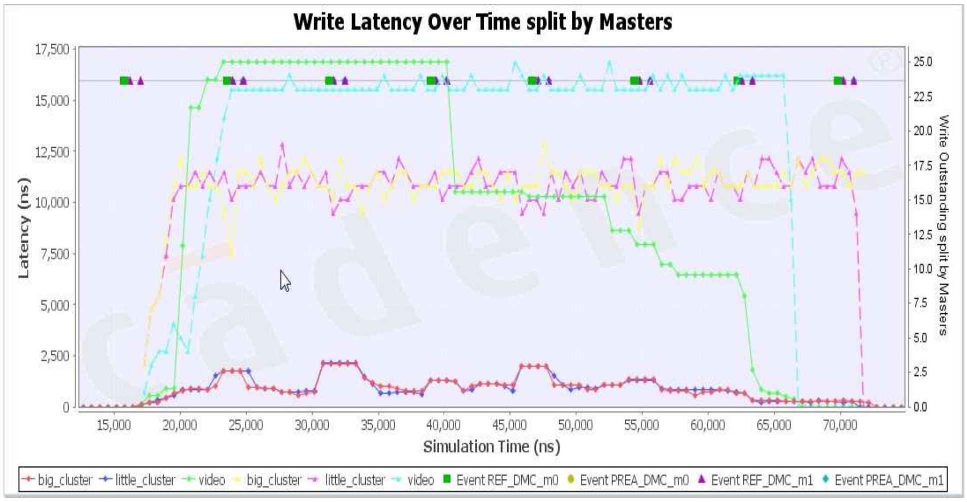


Figure 10: Write Latency over time with OT overlayed.

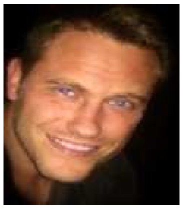
For added debug capability the chart also shows some key events from the DFI interface which is between the DDR controller and DDR PHY. The events shown (see the marks towards the top of the chart) are the refresh and pre-charge events from each DDR bank. This capability to display events from the Device Under Test (DUT) provides additional high-level information which can ease debugging performance issues. The events are logged using a simple monitor which can be user written. Visual correlation of system events and performance issues are a powerful tool in understanding how the system IP behaves under duress.

Summary

As the complexity of multi-core, multi-cluster CPU sub-systems continues to increase there is a corresponding increase in the complexity of performance tuning the sub-system design for best results. This paper introduces a systematic approach to achieving this goal. Functional verification methodology can provide assurance that the CPU sub-system will function correctly but this is not enough, a performance verification methodology is also needed which can provide an approach to ensure all performance requirements are validated against realistic workloads.

About the Authors

Nick Heaton is an ASIC and EDA veteran with more than 30 years of experience in the design and verification of complex SoCs. Nick graduated from Brunel University, London in 1983 with First Class Honors in Engineering and Management Systems. In 1993, he founded specialist ASIC Design and Verification Company Excel Consultants, servicing customers such as ARM® and Altera. In 2002, Nick joined Verisity (now Cadence) as Manager of Northern European Consulting Engineering. Nick currently works in the Cadence Research & Development organization as a Distinguished Engineer with special responsibility for Interconnect Workbench.



Simon Rance is Senior Product Manager at ARM and has been involved in chip design and automation for over 15 years. His career has spanned IC architecture, design & verification, software development, IP and EDA solutions. He has written and presented several papers and topics world-wide around IP design, chip assembly and automation techniques. In ARM, he is managing System IP Tooling world-wide.