

# A Methodology for Performance Analysis of Network-on-Chip Architectures for Video SoCs

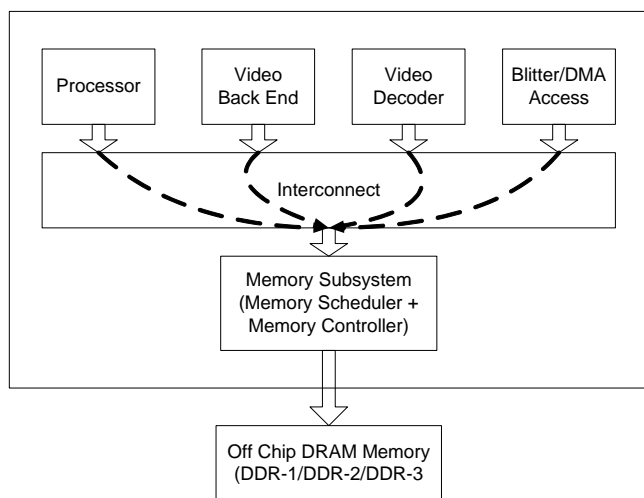
Krishnan Srinivasan, Performance Architect  
Sonics, Inc.

## 1. INTRODUCTION

A System-on-Chip (SoC) consists of several processing engines integrated onto the same chip. The traffic flow from each of the processing engines can have different performance requirements. For example, a processor whose traffic to the external DRAM is dominated by cache-line fills would require that the latency of the traffic is minimized. On the other hand, traffic flow from a video decoder engine requires that the traffic is serviced with the low jitter, such that the amount of buffering within the engine can be minimized.

Figure 1 depicts a typical SoC architecture for HDTV systems. For clarity, only the major blocks are included. The initiators are connected to the interconnection network by means of a communication interface, such as the Open Core Protocol (OCP) interface. The majority of traffic flows from the different processing engines (henceforth called Masters or Initiators) converge into the DRAM memory. In other words, the system operates as shared memory architecture and all bandwidth-intensive communication takes place through the external DRAM architecture. Therefore, the low bandwidth traffic not targeted at the DRAM architecture is not modeled. The memory subsystem consists of a memory scheduler, a memory controller, and the off-chip DRAM. The memory scheduler arbitrates among multiple traffic flows to generate requests to the controller, which are then issued to the DRAM.

Figure 1: SoC with DRAM



With each passing generation, the complexity of the system is increasing. The major artifacts of an increasingly complex system are:

- i) the number of masters is increasing from low tens, to the high tens or even hundreds
- ii) The bandwidth at the individual masters is increasing

Finally, in order to sustain the increasing bandwidth requirements, the number of memory channels is increasing from one to two to four and eight in the future. Therefore, a system with 50 initiators communicating with four DRAM channels (where each DRAM channel has its own NoC terminal) will introduce 200 traffic flows, each with its own bandwidth and latency requirement.

The increasing complexity of the SoC architectures has led directly to the increase in complexity of the interconnection architecture. The interconnection architecture has must solve multiple objectives: Satisfying bandwidth requirements,

satisfying latency requirements, minimizing gate count in the design, and minimizing power consumption. Such complex interconnection architectures have brought about a need for structured performance analysis of the same.

Increasingly, SoC designers are moving toward an intellectual property- (IP) based design methodology. A typical SoC may be comprised of a processor from a processor supplier, some locally designed blocks, an interconnect and memory subsystem from an interconnect vendor, and the DRAM memory subsystem from a DRAM vendor. For the interconnect vendors, the analysis of the interconnect with a system view has become a major problem. Often, the interconnect vendors do not have the access to complex initiators that constitute the SoC. Hence, they are often forced to design with the worst case scenario in mind, which can lead to unnecessary redundancy. The other option, which is to design the interconnection architecture based on random flows, can force re-architecting the interconnect late in the system design cycle, as the interconnect may not provide the required performance for the target SoC.

In this paper, we describe a methodology for the performance analysis of the interconnection architecture, with focus on SoC architectures for video applications, such as HDTVs, set-top boxes, etc. Our approach is to describe a typical video SoC, and provide a set of traffic profiles for the initiators that constitute the SoC. With the traffic profiles as basic building blocks, video SoC benchmarks of varying complexity can be constructed by varying the number of initiators and the amount of bandwidth per initiator.

We provide traffic profiles for various initiators so they can be modeled in an abstract manner, thus minimizing the development effort. In addition to the traffic profiles, we also provide essential measurement schemes designed for video benchmarking. With the recommendations of this paper, the interconnect designer can model traffic flows that mimic typical video systems and thus, can make the necessary architectural decisions early in the design cycle.

In the paper, we use the semantics of the OCP to explain the methodology. However, the methodology itself is not dependent upon the protocol. The paper is organized as follows. In Section 2, we describe the essentials to describe a video SoC. In Section 3, we describe the traffic profiles for the different traffic generators in the system. In Section 4, we describe the overall approach to generate a video SoC benchmark. In Section 5, we present details about the performance analysis of video SoC systems. Finally, in Section 6, we offer our conclusions.

## 2. Description of Video SoC Application

A typical video SoC consists of heterogeneous processing engines that primarily communicate with the external shared DRAM memory. The processing engines operate in a fully pipelined fashion, and for the purposes of the performance analysis of the interconnect, can be thought of as operating as independent initiators. The major processing engines in a video SoC are processors, transport engines, video decoders, audio processing units, graphics engines, display processors and a peripheral display cluster. Hence, there are seven distinct types of processing engines. The memory subsystem primarily consists of the DRAM memory, and can consist of one, two, four or eight DRAMs. With these units, a SoC can be integrated in a similar manner described in Figure 1, where all major communication takes place through the shared DRAM architecture. In order to describe the system, the system designer needs to consider the following parameters:

1. The number of initiators in the SoC
  - a. We note that a video SoC will be composed of at least one component of each initiator type, where an initiator type could be a processor, transport engine, video decoder, audio processing unit, graphic engine, display processor or a peripheral unit. However, it is possible for the designer to specify multiples of such units. For example, the SoC may be composed of two processors, two video decoders and one of each of the other initiator type.
  - b. For each initiator, the designer must specify the number of OCP threads and number of OCP tags per thread in the corresponding initiator OCP interface.
2. The performance requirements of each initiator
  - a. The performance requirements of each initiator can be specified based on:
    - i. worst case latency
    - ii. average latency
    - iii. the required bandwidth
  - b. The requirement of each initiator type will be different
3. The DRAM memory architecture
  - a. The DRAM architecture could be comprised of DDR-1, DDR-2, DDR-3 or XDR. For the DRAM architecture, the type of part, the number of parts, the DRAM burstlength, and the operating frequency of the DRAM bus is specified. Further, the number of DRAM channels in the design is also specified.
4. The width of the communication interfaces
  - a. The system designer specifies the width of the communication interfaces between the initiators and the interconnect, as well as the interconnect and the memory subsystem.
5. The operating frequency

- a. The system designer should specify the operating frequency of the communication interfaces. In this version, we do not explicitly define clock domain crossing requirements. We assume that the NoC handles the domain crossing, wherever required.

The interconnection architecture should satisfy the performance requirements of each initiator, subject to the other design parameters provided by the system designer. In addition, it should also be able to provide a measure of the number of gates that are consumed from an overall system perspective. For example, choosing a particular interconnection architecture can result in the minimization of the buffering in one of the initiators, while increasing the gate count of the network itself. Therefore, the overall evaluation must take into account the relative advantages and disadvantages taking the system as a whole, instead of evaluating the interconnect in isolation.

### **3. Traffic Profile Generation**

In this section, we describe the traffic profile generation for each initiator. The traffic profiles are generated in three levels of hierarchy. In the first level, the designer instantiates an abstract SoC system by specifying the number of initiators and their type, total bandwidth requirement of the system, the number of DRAM channels, the details about the DRAM architectures, and the details of the OCP interfaces.

In the second level of hierarchy, the designer specifies details about each initiator type. This includes the type of accesses, locality of addresses, etc. Finally, in the third level of hierarchy, the designer specifies the details about each initiator. For example, each initiator specifies the number of outstanding transactions per thread that the initiator can support.

In the remainder of the section, we describe the hierarchy level-1 specifications, followed by the traffic profile for each initiator type. In describing the traffic profile, we also describe the set of parameters associated with each initiator type at the second level of hierarchy, and initiator in the third level of hierarchy.

#### **3.1 Hierarchy Level-1 Specifications**

The level-1 specifications include the details about the entire system, without focusing on the individual initiators. The system designer provides the following parameters:

1. The number of initiators of each
2. The total bandwidth requirement of the system, which would be the sum of the individual bandwidth requirements at each initiator
3. The number of DRAM channels
4. The frequency of each DRAM bus
5. The number of parts in each DRAM
6. The type of part in each DRAM
7. The DRAM burstlength for each channel
8. The data width of each initiator OCP interface
9. The operating frequency of each initiator OCP interface
10. The number of threads in each initiator OCP interface
11. The number of tags per thread in each initiator OCP interface

#### **3.2 Processor Traffic Profile**

The CPU deals with cache-line sized chunks primarily. Cache line sizes are usually 32 bytes or 64 bytes. A dirty line replacement results in a Writeback (WB). The cache line fetch and writeback operations manifest as a read or write respectively, of burst-length depending on cache line size and OCP interface data width. Finally, depending on the cache organization, there can be separate instruction cache. If an instruction cache is present, then there are only read misses, since writebacks are absent.

In this paper, we abstract cache hierarchy away by expressing the miss rate coming to the interface as “misses per instruction” (MPI) or “misses per cycle,” rather than the misses per access. Therefore, we make the traffic profile insensitive

to the cache hierarchy of the system. Another advantage of the MPI measure is that it is, in part, an indicator of the CPI (cycles per instruction) performance measure. For our purposes, we can use the misses per cycle (MPC) measure instead of the MPI ( $MPC = MPI/CPI$ ).

### 3.2.1 Processor Characteristics:

1. Cache line size: (default 32B)
2. Read misses per cycle: 0.01 – 0.001
  - a. Therefore, a read miss should be issued every 100 – 1000 cycles, if the number of outstanding transactions is less than the number that can be supported.
3. Write misses per cycle (0.2 to 0.3) \* Read misses per cycle for data cache
  - a. 0 for instruction caches
4. Temporal distribution of accesses:
  - a. Uniform distribution (this is generated from a combination of read misses, writebacks)
  - b. With separate instruction cache, the misses are typically bursty because of high temporal and spatial locality, but the MPC itself is much lower (0.001 – 0.0001).
5. Spatial distribution of accesses (i.e., address generation)
  - a. Random address distribution
6. Bandwidth requirement
  - a. Bandwidth requirement of all CPUs together is about 15 % of the total bandwidth requirement of the SoC.

Based on the processor characteristics, the following hierarchy-Level-2 and hierarchy-Level-3 parameters can be derived:

#### **Level-2 Parameters**

1. Distribution of bandwidth among multiple processors
  - a. For example, if there are two processors in the system, the designer may choose to assign 33.33% of the processor bandwidth to processor-1, and the remaining 66.67% to processor-2. Therefore, the bandwidth on processor-1 will be 33.33% of the 15% of the total system bandwidth, which is 5% of the total system bandwidth. The remaining 10% of the total system bandwidth will be assigned to processor-2.

#### **Level-3 Parameters**

1. Cache line size 32 bytes or 64 bytes
2. Read miss and Writeback MPC per thread of the initiator OCP
  - a. The value should be within the specified range, and should meet the bandwidth requirement at the initiator
3. A value to indicate if instruction cache is available
  - a. If instruction cache is available, 1.5 % of the processor's traffic will be random reads. In addition to read miss every 100-1000 cycles
  - b. Without cache, no random reads

4. Outstanding transactions per thread. We measure the outstanding transactions in bytes. When this value is set to N, the processor generates at most N bytes of transactions per thread before stalling on that thread to receive the response to the first of the N transactions.
5. Average and worst case latency that the processor can tolerate
6. The percentage of traffic going to each thread, and with each tag ID, within each thread.
  - a. This value represents the distribution of traffic on each thread/tag.

### 3.3 Display Processor Profile

The display processor performs a variety of functions such as multi-field adaptive interlacing, motion interpolation, color correction, scaling, and noise reduction. This manifests in typically long bursts with spatial locality. The traffic typically consists of alternating periods of high activity, and no activity. During the period of activity, the initiator generates transactions at random intervals, such that the bandwidth in the active period is twice that of the total required bandwidth. The traffic generator stalls if it reaches the maximum number of outstanding transactions.

#### 3.3.1 Display Engine Characteristics:

1. Long bursts: burstlength of 128 – 384 bytes
2. Read to write ratio: ~2 – 3
3. Reads and writes go to different memory banks
4. Read addresses are correlated (spatial locality)
  - o When addresses are spatially correlated, the burst has incrementing addressing over large windows, for example. 512 bytes. This can be modeled by a weighted random selection of address. For example, start at a random address, and have incremental addresses for 512 bytes.
5. Write addresses are correlated (spatial locality)
6. Reads and writes are randomly distributed temporally, but they must satisfy the bandwidth requirement
7. The bandwidth is 40 to 50 % of the total system bandwidth

From the characteristics, the following Level-2 and Level-3 parameters can be derived:

#### **Level-2 Parameters**

1. Distribution of traffic among the multiple display engines
2. Amount of traffic for the display engine. It should be between 40 % and 50 % of total SoC traffic
3. Length of the one period of activity as a percentage of the total simulation, which is a measure of the number of frames being processed in one simulation

#### **Level-3 Parameters**

1. Burst length of the initiator
2. Number of outstanding transactions per thread
3. Number of OCP threads and OCP tags per thread on the initiator OCP interface
4. The percentage of traffic going to each thread, and with each tag ID, within each thread

### 3.4 Video Decoder Profile

The video decoder performs motion compensation/estimation, de-blocking filter, DCT transform, de-Quantization and entropy encoding. The operations are often performed on blocks of images, thus resulting in several OCP BLCK bursts. Each row in the BLCK burst is typically 16 to 64 bytes long. The number of rows is generally between 2 and 16, with the total size of the burst being 128 bytes to 384 bytes. Similar to the display processor, this unit also consists of alternating periods of high activity and quiet period. The period of activity has traffic flow which is twice the programmed bandwidth. Within an active period, the traffic is constrained randomly.

### 3.4.1 Video Decoder Characteristics:

1. BLCK bursts: burstlength of 16 – 64 bytes per row, 2 to 16 rows, and 128 to 384 bytes of total burst size
2. Read to write ratio: ~2 – 3
3. Reads and writes go to different memory banks
4. Read addresses per row are correlated (spatial locality)
5. Write addresses per row are correlated (spatial locality)
6. Two rows in a burst are separated by an address offset that is a power of 2. For example, 0x1000.
7. Reads and writes are randomly distributed temporally, but they must satisfy the bandwidth requirement
8. The bandwidth is 20 to 30 % of the total system bandwidth

From the characteristics, the following Level-2 and Level-3 parameters can be derived:

#### **Level-2 Parameters**

4. Distribution of traffic among the multiple decoder engines
5. Amount of traffic should be between 20 and 30 %
6. Length of the one period of activity as a percentage of the total simulation, which is a measure of the number of frames being processed in one simulation

#### **Level-3 Parameters**

5. Burst length of the initiator
6. Number of outstanding transactions per thread
7. The percentage of traffic going to each thread, and with each tag ID, within each thread.

## **3.5 Graphics Processor/ Blitter Profile**

The graphics processor unit performs color fill, rectangular copy, color space conversion, alpha blending, Gamma correction, resizing and clipping. The burst length of the transactions are typically large, about 256 bytes is size. Similar to the display processor, this unit also consists of alternating periods of high activity and quiet period. Within an active period, the traffic is constrained randomly. The period of activity has traffic flow which is twice the programmed bandwidth

### 3.5.1 Graphics Processor Characteristics:

1. Burstlength: 128-256 bytes
2. Read to write ratio: ~2 – 3
3. Reads and writes go to different memory banks
4. Read addresses are correlated (spatial locality)
5. Write addresses are correlated (spatial locality)
6. Reads and writes are randomly distributed temporally, but they must satisfy the bandwidth requirement
7. The bandwidth is 10 to 15 % of the total system bandwidth

#### **Level-1 Parameters**

1. Distribution of traffic among the multiple decoder engines
2. Amount of traffic should be between 10 and 15 %
3. Length of the one period of activity as a percentage of the total simulation, which is a measure of the number of frames being processed in one simulation

#### **Level-2 Parameters**

1. Burst length of the initiator
2. Number of outstanding transactions per thread
3. Number of OCP threads and OCP tags per thread on the initiator OCP interface
4. The percentage of traffic going to each thread, and with each tag ID, within each thread

### 3.6 Audio Processor, Transport and Peripheral Cluster

The audio processing unit and the peripheral clusters are low activity initiators, and make up the remaining of the traffic in the SoC. They can be randomly generated, within the available bandwidth.

## 4. Overall Approach

The overall approach to performance benchmarking of NoCs for video SoC is described in the following steps:

1. The level-1 parameters regarding the overall system are specified
2. For each initiator type, the Level-2 parameters are described
3. For each initiator, the Level-3 parameters are specified
4. A derivation algorithm should read the parameters of the system, and generate traffic for each initiator type, based on the characteristics of the initiator, and the Level-1/Level-2/Level-3 parameters.

The following enumeration describes the range of values to be used to specify each parameter in the system. Based on this, multiple designs can be generated, which can be used for performance analysis.

1. Number of initiators of each type : 1 to 4
2. Number of threads per initiator” 1 to 4
3. Number of tags per thread: 1- 8
4. Bandwidth of the system: 2 GBps – 10 GBps
5. OCP data width: 4 bytes to 16 bytes
6. OCP interface frequency: 266 MHz to 800 MHz
7. Number of DRAM channels: 1 – 8
8. Number of parts per DRAM: 1 to 4 “x16” parts, assuming 64Mbytes per part
9. Frequency of DRAM bus: 333 MHz (DDR-2 667), 533 MHz (DDR-2 1066), 800 MHz (DDR-2/3 1600)
10. DRAM burst-length: 4 or 8 DRAM words. If DDR-3 is used, the burstlength must be 8
11. Number of outstanding transactions per thread: 64 bytes to 512 bytes. Note that, the minimum value depends on the burst-length of the transaction.

## 5. Performance Analysis

In this section, we describe the essential performance analysis measures to ensure that the interconnection architecture is meeting the performance requirements of the system. A system is said to have met the performance only if **both** of the following are true:

1. The bandwidth requirement at each initiator is satisfied
2. The average and worst case latency constraints at the processor are satisfied

In addition to meeting the performance of the system, it is necessary to provide the designer with a measure of the cost of achieving this performance. The cost is mainly area and power, both of which can be estimated by the number of storage elements in the system. The total number of storage elements should not only take the storage elements within the interconnection network, but the storage elements in the initiators as well.

The total number of outstanding transactions in the initiator gives a measure of the amount of storage available in the initiator. For example, if an initiator can generate 128 bytes of outstanding transactions, it means that the initiator has a fifo

with a storage of 128 bytes, into which the transactions are temporarily stored. Therefore, we can associate each outstanding transaction as an entry in the fifo. One can estimate the total amount of storage required in the system by the following formula:

Measure of gate count = for all fifos in the network,  $\sum(\text{fifo depth} * \text{fifo width in bytes} +$   
for all initiators, for all threads  $\sum(\text{number of outstanding transactions per thread})$

## 6. Conclusion

In this paper, we addressed the performance benchmarking problem of Network-on-Chip architectures. We demonstrated the need for a performance benchmarking infrastructure, which interconnection network designers can utilize to measure the effectiveness of their architecture. Our focus here has been on performance benchmarking for SoCs targeted at video applications, such as HDTVs and set-top boxes. We also presented methods to analyze the performance of an interconnection architecture, by taking the performance goals and gate count into account. Future work will focus on other verticals in the scope of interconnection architecture design, such as wireless, automotive and multi-processor architectures.