Selecting memory controllers for DSP systems

By Deepak Shankar,
Mirabilis Design

DSP systems often include multiple embedded processors and hardware accelerators. The performance of these systems is typically limited by factors such as I/O bandwidth, memory distribution, and memory speed. This is particularly true when the system components share a memory interface. For such systems, it is critical to choose the right memory controller.

Different memory controllers offer latency distributions that make them suitable for specific applications. For example, a slot-based controller with fixed priorities can offer deterministic latencies, while buses such as PCI-Express and CoreConnect offer lower latency at peak loading but higher average latency.

It is extremely difficult to predict the performance of the memory system and the effect of contention without a solid model of the system. It is therefore important to invest in modelling before beginning development. This modelling should include allocating of threads/tasks to resources, identifying any custom hardware needs, and determining the size and speed of the I/O. The modelling can performed using a number of methods, including "back of the napkin" calculations, spreadsheet analysis, or by building a physical prototype.

In this article, we examine a unique "virtual prototyping" approach to modelling. We use this approach to model a MPEG II application in a Xilinx FPGA. We evaluate two memory access schemes for this application: the MPMC Memory Controller from Xilinx, and the CoreConnect Bus specification for FPGAs.

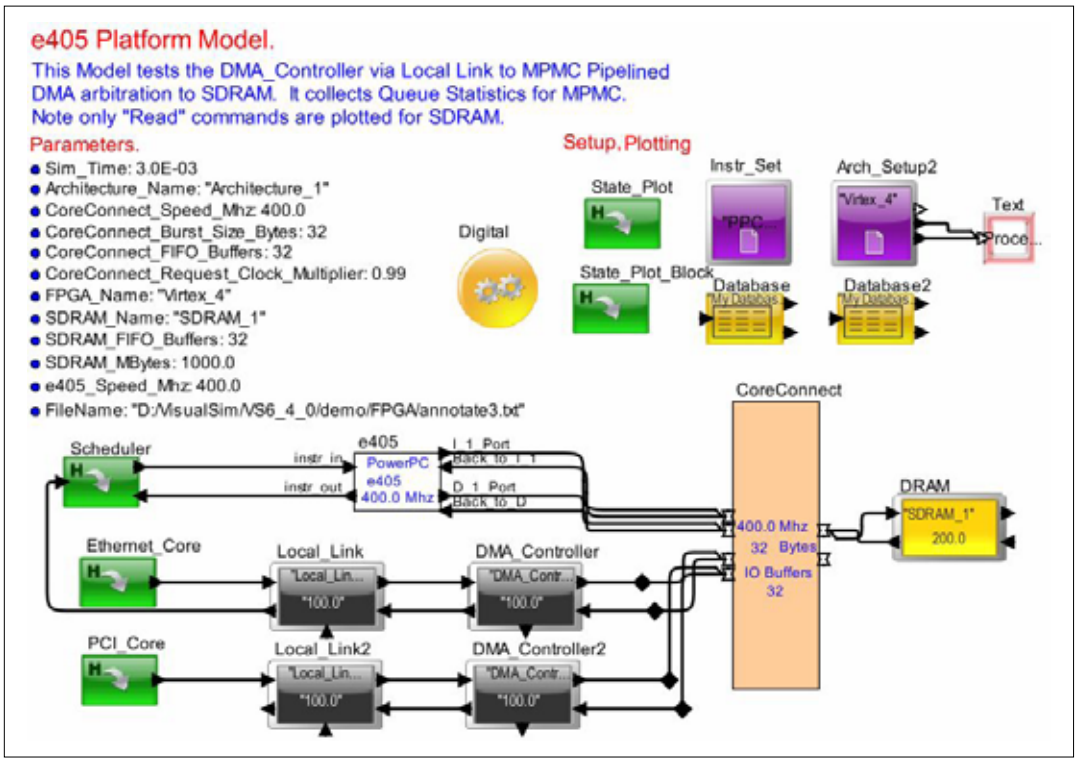

Figure 1: VisualSim model of the Xilinx Virtex 4 FX Platform: e405 with MPMC Memory Controller.

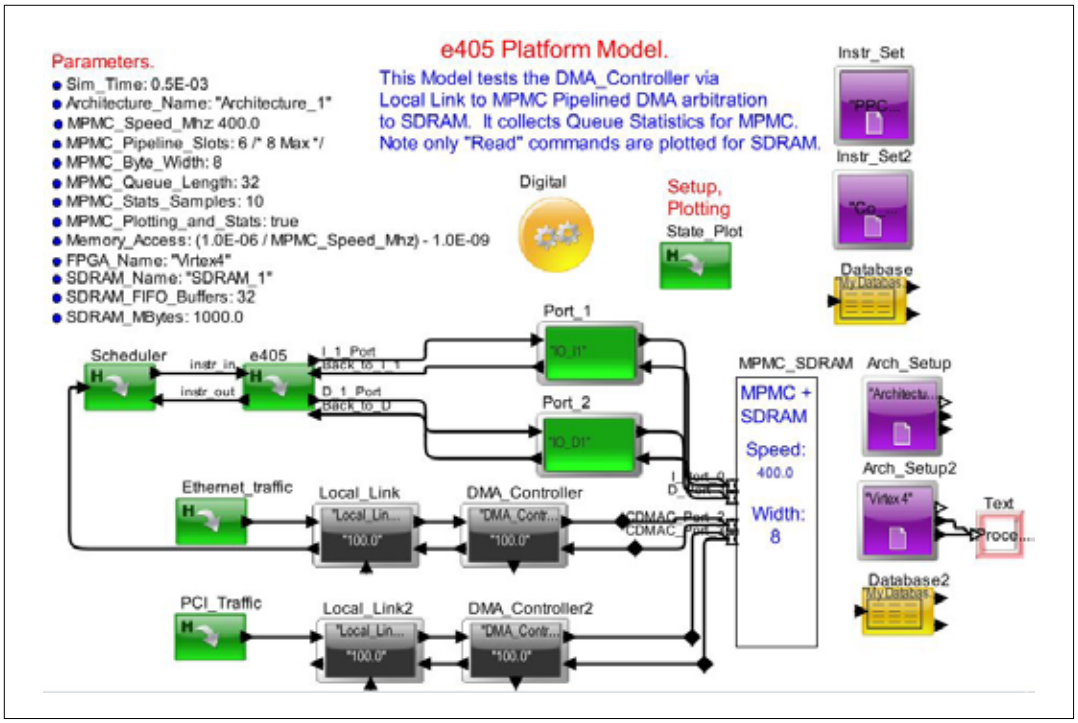

Figure 2: VisualSim model of the Xilinx Virtex 4 FX Platform: e405 with CoreConnect Bus Model.

Virtual Prototyping

The virtual prototyping approach discussed in the article is based on the VisualSim solution from Mirabilis Design. VisualSim is a graphical platform for analyzing the performance of hardware and software systems. It is based on a library of parameterized components including processors, memory controllers, DMA, buses, switches and I/Os. Using this library of building blocks, a designer can construct a specification-level model of a system

		Time Slot					
		1	2	3a	3b	4a	4b
Priority	1	P0	P1	P2		P3	
	2	P1	P0	P0	P1	P0	P1
	3			P1	P0	P1	P0

Figure 3: MPMC Memory Controller Arbitration Algorithm.

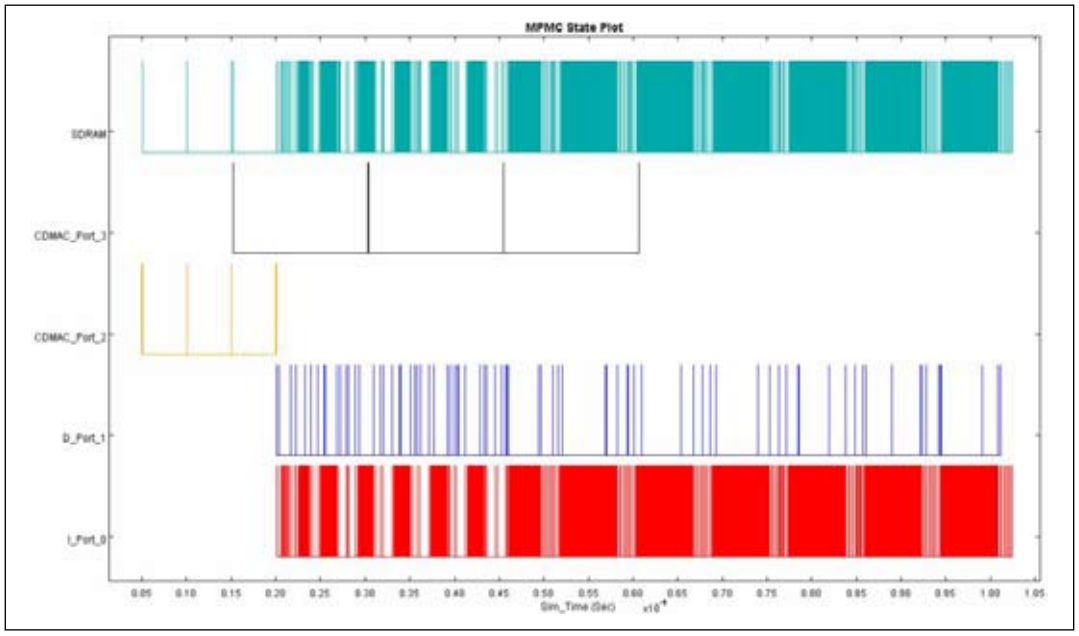

Figure 4: MPMC Memory Controller Activity with SDRAM at top.

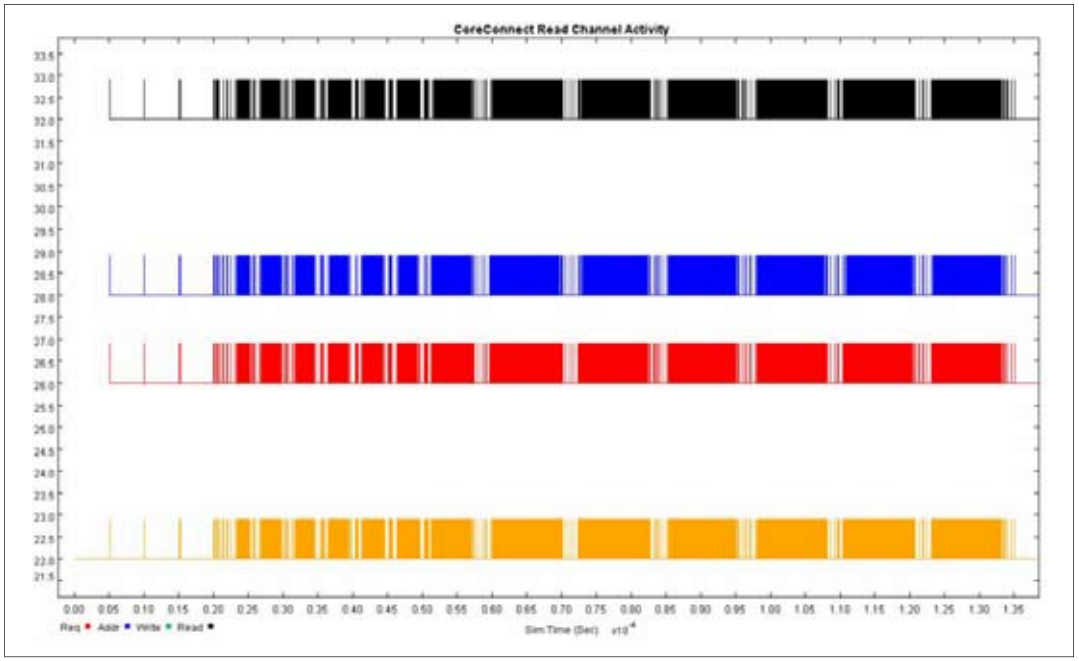

Figure 5: CoreConnect Bus Activity with SDRAM at bottom.

containing multiple processors, memories, sensors and buses. Model construction is a process of connecting icons that represent the IP in a graphical editor, as illustrated in Figure 1.

Simulations can be explored by varying parameters for the input scenarios, data rates, priorities, speed, or size. By analyzing the simulation results, the designer can choose the solution that achieves the required latency with the lowest power, smallest fabric configuration, and highest system throughput.

Project Overview

This project evaluates the performance of a MPEG II algorithm implemented in C. The target system is a Virtex-4 FX with up to two hardwired PowerPC e405 cores connected to DDR2 SDRAM. We evaluate two memory access schemes for this application: The Multi-Port Multi-Channel (MPMC) Memory Controller and the CoreConnect Bus.

The MPMC Memory

Controller (Figure 1) is a popular option for this type of application, as it provides an extremely efficient means of interfacing the processor and key high speed devices to SDRAM. The CoreConnect Bus (Figure 2) is another popular option. It supports multiple masters, including the PowerPC caches and key high speed devices, connected via a slave port to the SDRAM. The preferred technique depends on which approach will provide the best performance in terms of throughput, latency, and processor efficiency.

To investigate the similarities and differences between the two approaches, we constructed models of both configurations using VisualSim graphical modelling environment. The exploration models did not require any software coding. The designer only needs to connect the different modelling elements, create the right traffic mix from the processor and high speed devices, and select parameter settings that match the anticipated design.

Design Considerations

The MPMC Memory Controller and CoreConnect Bus are explored by using the same configuration. The instruction and data channels of the PowerPC e405, Ethernet interface and PCI interface are connected as Masters. The DDR2 SDRAM is a Slave. The Masters and Slaves were all maintained at the similar speed and size characteristics in both the models. The design considerations in this study are:

- What is driving the design—overall performance or a combination of power and performance?
- Do I need one or two e405 PowerPCs for my core tasks?
- If the e405 PowerPC is running at 400 MHz, then what might be the best MPMC or CoreConnect Bus clock ratio?

Statistics Name	CoreConnect		MPMC	
	Instruction Cache	Data Cache	Instruction Cache	Data Cache
Hit_Ratio_Max	95.05	96.23	94.98	97.37
Hit_Ratio_Mean	9.51	9.62	94.98	97.37
Hit_Ratio_Min	0.00	0.00	94.98	97.37
Hit_Ratio_StDev	28.52	28.87	0.00	0.00
KB_per_Thread_Max	1.80	0.04	1.80	0.04
KB_per_Thread_Mean	0.18	0.00	1.80	0.04
KB_per_Thread_Min	0.00	0.00	1.80	0.04
KB_per_Thread_StDev	0.54	0.01	0.00	0.00
Threads_Max	33.00	33.00	33.00	33.00
Threads_Mean	3.30	3.30	33.00	33.00
Threads_Min	0.00	0.00	33.00	33.00
Threads_StDev	9.90	9.90	0.00	0.00
Throughput_MIPs_Max	74.45	1.69	7.44	0.16
Throughput_MIPs_Mean	7.44	0.17	7.44	0.16
Throughput_MIPs_Min	0.00	0.00	7.44	0.16
Throughput_MIPs_StDev	22.33	0.51	0.00	0.00
Utilization_Pct_Max	18.61	0.42	1.86	0.04
Utilization_Pct_Mean	1.86	0.04	1.86	0.04
Utilization_Pct_Min	0.00	0.00	1.86	0.04
Utilization_Pct_StDev	5.58	0.13	0.00	0.00

Table 1: Instruction and Data Cache Statistics.

- Does my design have equal read and write memory activity, more reads than writes, or more writes than reads?
- What is the effect of the SDRAM speed on the processor/memory controller/bus clock ratio?
- What is the effect of the SDRAM speed on added application access to memory?

These design considerations overlap and are interdependent, making analysis complicated. The ability to model concurrency (that is, simultaneous events) in a deterministic manner within VisualSim allows for a consistent comparison of the MPMC Memory Controller and CoreConnect bus configurations.

Our exploration looked at processor stalls, cache hit-ratios, I/O throughput and latency per task. We modified a combination of clock speed, controller width and data loading rates to study the performance. This article discusses only the overall performance of the design. For more details on this design exploration or to view the detailed modelling effort, please contact the authors.

System Setup

The analysis was using the embedded solution on a Xilinx Virtex 4 FX. The system was modelled as follows.

PowerPC:

The PowerPC e405 core executed at 400MHz and the external SDRAM ran at 200MHz. For this analysis, we used a 200MHz DDR2 memory controller to interface to the SDRAM. The PCI received data at 33 MHz while the Ethernet ran at 100Mbps. The PCI stimulus was received at constant rate while the Ethernet received data every 2 ms in bursts of 4 packets of 256 bytes. The external SDRAM was 1GB and had 32 transaction-level FIFO buffers.

Floating-point co-processor

We used the Xilinx APU/FCM floating-point co-processor for this benchmark. The co-processor was needed due to the vector-oriented, heavy DSP nature of the workload. The APU ran at 400MHz to match the PowerPC pipeline speed. The PowerPC pipeline handled the instruction and data pre-fetch, and the APU simply executed the decoded instruction. Memory access from the APU was performed through PowerPC bus interfaces.

DMA: The PowerPC accessed memory using a direct memory call, while the PCI and Ethernet accessed memory using DMA. (The Ethernet and PCI have a lightweight DMA engine with a dedicated channel.) This was essential to fragment the incoming data and to prevent the network traffic from causing a bottleneck at the memory controller. To minimize the impact of instruction cache misses, software processing was triggered after 4 bursts of data from the Ethernet were saved in the SDRAM and the DMA triggered an interrupt in the processor.

Application software: There are a total of 33 unique DSP tasks in the application. The tasks running on the processor include: DFT, CS_Weighting, IR, and Q_Taylor_Weighting. These tasks are executed multiple times, and each loop has variable counts based on the size, depth and resolution of the incoming image.

MPMC Memory Controller

The MPMC Memory Controller arbitrates to SDRAM in a predictable fashion. As shown in Figure 3, the controller algorithm gives preference to the Processor Instruction (P0) and Data (P1) cache accesses for certain cycles, and secondary preference for application access ports 2 (P2) and 3 (P3) if these slots are already in use. Figure 4 illustrates the access patterns observed for the MPMC Memory Controller.

CoreConnect Bus:

The CoreConnect Bus arbitrates from the master to slave port using read, write, address and request channels. The slave port was connected directly to the SDRAM, and the bus arbitration provided high speed burst access to the SDRAM. The read channel has a capacity of 4 bursts and the write channel has a capacity of 2 bursts. For this benchmark, we used bursts of 16 bytes or two memory accesses of 8 bytes each. Figure 5 illustrates the access patterns observed for the CoreConnect Bus.

Analysis

The simulation ran on a 1.6 GHz Microsoft Windows XP (SP2 and Standard Edition) machine with 512 Mb of cache. We simulated 3.0ms of system execution. This simulation took 67 seconds to complete. The model was constructed in about 4 days using standard elements in the VisualSim library.

We simulated both 200 MHz and 400 MHz clock rates for the CoreConnect Bus and the MPMC Memory Controller. (All other model parameters were held constant.) Using the 200MHz MPMC, the end-to-end latency for application was 87.190 us,

Task Name	Cycles in Processor	
	MPMC Memory Controller	CoreConnect
DFT	108	109
DFT	735	742
DFT	716	712
CS_Weighting	448	523
IR	850	881
Q_Taylor_Weighting	439	523
CS_Weighting	437	523
IR	842	934
Q_Taylor_Weighting	463	492
CS_Weighting	460	549
IR	832	879
Q_Taylor_Weighting	459	515
CS_Weighting	466	523
IR	851	871
Q_Taylor_Weighting	486	502
CS_Weighting	521	513
IR	834	879
Q_Taylor_Weighting	533	482
CS_Weighting	518	518
IR	1565	1649
DFT	763	743
DFT	2671	2671
DFT	762	748
DFT	2676	2699
DFT	774	747
DFT	2676	2711
DFT	743	751
DFT	2681	2677
DFT	774	754
DFT	2671	2672
DFT	755	762
DFT	2678	2671
DFT	732	732

Table 2: Task latency in cycles for key tasks in the application benchmark software.

Statistics Name	CoreConnect	MPMC
Stall_Time_Pct_Max	19.64	1.99
Stall_Time_Pct_Mean	1.96	1.99
Stall_Time_Pct_Min	0.00	1.99
Stall_Time_Pct_StDev	5.89	0.00
Task_Delay_Max	6.823E-06	6.750E-06
Task_Delay_Mean	2.372E-06	2.374E-06
Task_Delay_Min	2.625E-07	2.625E-07
Task_Delay_StDev	1.822E-06	1.830E-06

Table 3: Latency and Stall Activity Statistics.

while the 400MHz CoreConnect achieved a latency of 86.42 us. Both numbers met our real-time threshold. All the discussion below will be based on these clock rates. We found that lower CoreConnect clock rates did not provide adequate performance.

We found that the MPMC offered uniform latency cycles for cache-to-SDRAM accesses while CoreConnect had more variable timing. The lowest cycle count was achieved using the CoreConnect but its average count was significantly higher. Table 1 shows the details of our results.

For 25 of the 33 tasks, the MPMC Memory Controller finished its tasks faster than the CoreConnect Bus. Some tasks took significantly longer time when running with the CoreConnect bus. Table 2 shows the details of our results.

As shown in Table 3, the mean processor stall for the application using both techniques was about 2.66%. While the average processor stall was 1.99% for the MPMC, it was 1.96% for the CoreConnect model. Unfortunately, the CoreConnect had a peak stall of 19.64% and this caused the excessive latency for certain tasks. This indicated that when the CoreConnect was stalled, the impact on the application latency was higher than with the MPMC. This also confirmed our finding that the CoreConnect had a higher overall latency and processor stalling percentage. Increasing the network traffic did not cause any noticeable increase in the MPMC latency, but external traffic increases did cause the overall CoreConnect latency to increase.

The MPMC Memory

Controller had a significantly better mean hit ratio for the instruction cache—94% vs. 18.84% for the Core Connect Bus. At certain times during the simulation, CoreConnect did get to 94% hit-ratio but the duration was very short. This indicated that the MPMC arbitrated SDRAM requests better than the CoreConnect Bus.

Looking at the number of threads (33) and the thread size (1.8 KB on average), we see that increasing the PowerPC caches to 64KB each would reduce SDRAM latency and achieve 100% hit-ratio for the instruction cache. The maximum value is a worst case estimate, based on cache access activity with no prefetching or considerations for looping in the application code.

Summary

The MPMC achieves a better SDRAM latency at almost half the speed of the CoreConnect bus. Moreover, the uniform cycle count offered scalability in places where the network traffic was more volatile. Our recommendation is to use MPMC if the lowest latency is required at the lowest clock rate and lowest power. In these two models one can see that the MPMC Memory Controller does a better job of arbitrating cache requests, resulting in better hit ratios and throughput for similar system conditions. Virtual prototypes and performance modelling allowed us to explore a number of scenarios and clock configurations. The statistics generated provided us full visibility into the operation of the FPGA design. Moreover the use of the fully validated VisualSim FPGA Modelling Toolkit components allowed us to save time without compromising overall accuracy. The toolkit has advanced simulation technology for FPGA system to make relative comparisons between different configurations.

References

IBM 64-Bit Processor Local Bus, Architecture Specifications, Version 3.5, SA-14-2534-01 FY 2001

Xilinx PLB Block RAM (BRAM) Interface Controller (v1.00b), DS420 October 25, 2005

Xilinx OPB Block RAM (BRAM) Interface Controller, DS468 December 1, 2005

Xilinx MCH_OPB Synchronous DRAM (SDRAM) Controller (v1.00a), DS492 April 4, 2005