WP459 (v1.0) January 13, 2015

Leveraging Data-Mover IPs for Data Movement in Zynq-7000 AP SoC Systems

By: Srikanth Erusalagandi

Moving large quantities of data, both off-chip and on-chip, requires careful selection of the interface technology best suited to the task. Data-mover IPs can help improve performance.

ABSTRACT

System-on-a-Chip (SoC) designs face increasingly aggressive power and performance system requirements. Systems designers are challenged to find the right architecture for their SoCs that satisfy these requirements. One of the most common operations in SoC-based design is the efficient implementation of data transfer functions, which can be critical to low-power or high-performance systems.

This white paper deals with some of the challenges that customers face while moving data in SoCs. Comprised of both a Processing System (PS) and Programmable Logic (PL), the Zynq®-7000 AP SoC architecture can leverage data-mover IP for efficient data movement within the PS, between the PS and PL, and within PL.

This white paper also describes some of the reference designs associated with data-mover IPs, guiding the designer to the best reference design for the envisioned application. This can help the designer reduce development time, associated design risks, and improve performance.

© Copyright 2015 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. PCI, PCIe, and PCI Express are trademarks of PCI-SIG and used under license. All other trademarks are the property of their respective owners.

Data Movement Challenges

Figure 1 illustrates the paths described in this section that impact data movement performance.

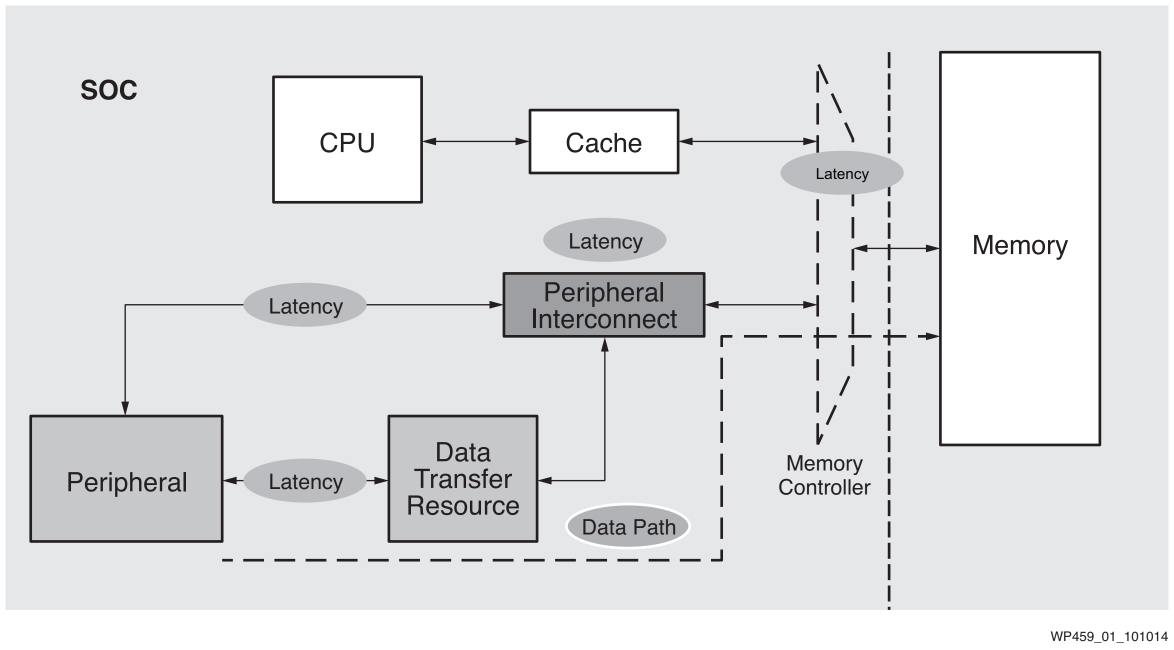

Figure 1: **Logical Diagram of Data Movement on an SoC**

Data movement in a System-on-a-Chip (SoC) can be on-chip from one functional block to another, or off-chip using a standard interface. Data transfers on-chip are normally done between standard peripherals or memory interfaces.

The key challenge a designer faces is finding a suitable data transfer method from a peripheral to a memory subsystem. While small on-chip data movement can be accomplished using software instructions, large data transfers can be done more efficiently using special data transfer resources.

Many SoCs use Direct Memory Access (DMA) controllers for this purpose, minimizing CPU overhead and increasing system performance. Many high-speed peripherals on SoCs (USB, Ethernet, etc.) include their own dedicated DMA engines that operate independently from the CPU. The CPU is involved only in setting up the transfer parameters and is interrupted when the transfer is completed. The CPU can perform other tasks while the data is being transferred, or it can be put into a low-power mode.

It is important that each peripheral on the SoC has the right bus protocol and data widths to optimally use the on-chip bus structures that minimize silicon infrastructure while supporting high performance, low power on-chip communications.

Another key element that impacts data movement performance is the on-chip bus structure that connects the peripheral to the memory subsystem. Today's SoCs also include advanced on-chip bus structures that help support multiple data transfers to occur simultaneously. Simultaneous data transfers not only enable higher data transfer bandwidths, but also enable lower power consumption when compared to doing sequential data transfers.

Apart from data movement, another key factor that impacts system performance is the data processing time, which indirectly affects the data movement operations in the system. Designers are looking for multi-processor solutions to increase system performance. The ideal solution involves not just *multiple* processors, but *different* compute engines for performing different tasks. Since all these engines do not work completely on their own, they need to pass data between them so each engine can perform its tasks and then hand the data set on to the next engine to perform its task. This particular data sharing mechanism mandates the data to be coherent between these multiprocessing engines. Data coherency can be managed using software, but doing so incurs the cost of executing cache maintenance routines whenever a processing engine updates the shared data. This has a direct impact on system performance.

Subsequent sections in this white paper describe how these challenges can be addressed while designing with the Zynq-7000 AP SoC.

Enable High-Performance Zynq-7000 AP SoC Systems with AXI

The Zynq-7000 AP SoC architecture comprises two sections, the Processing System (PS) and Programmable Logic (PL). The PS is an optimized silicon element consisting of a dual-core ARM® Cortex™-A9 MPCore™ with integrated peripherals. The PL enables designers to expand their hardware or accelerate the software by choosing either off-the-shelf IP or their own custom IP blocks that complement pre-built IP selections. Figure 2 illustrates the Zynq-7000 AP SoC block diagram.

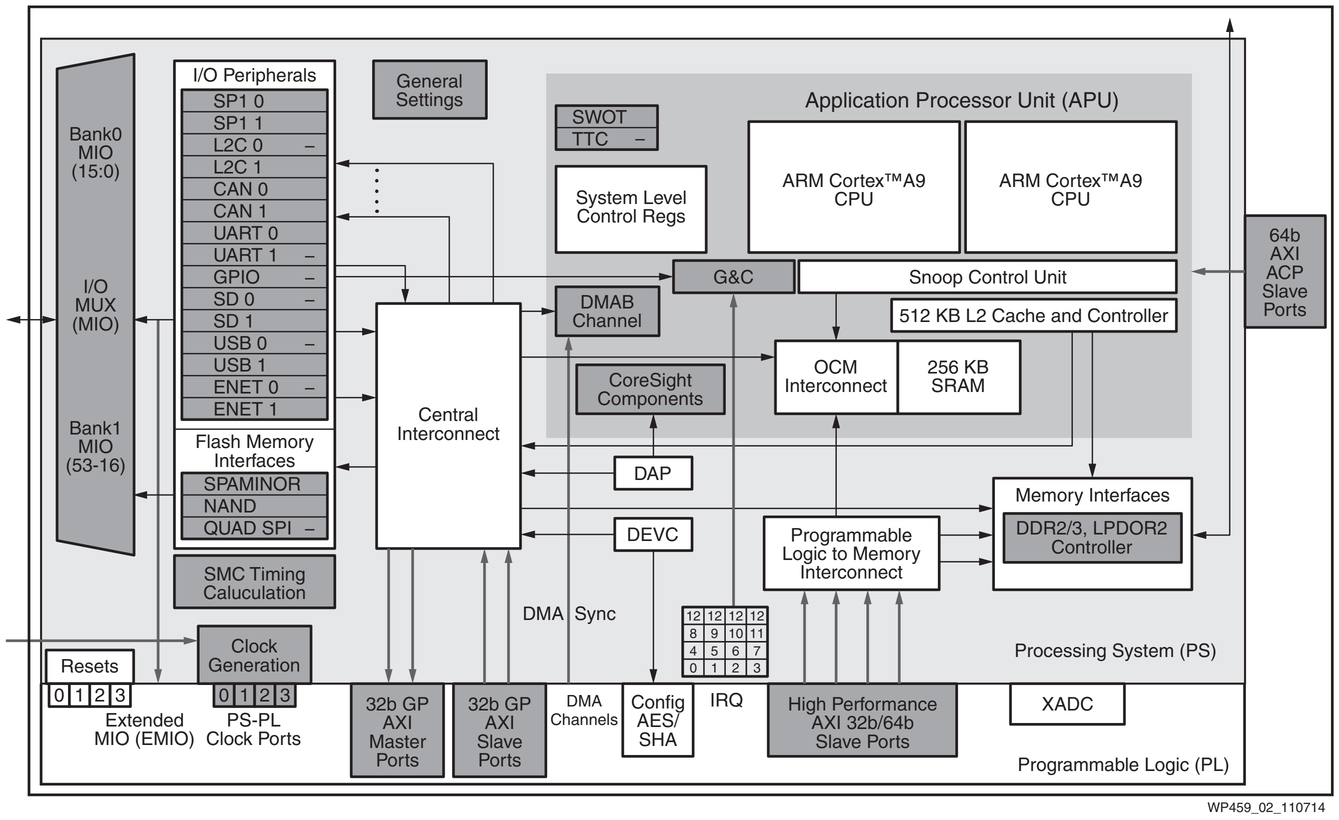

Figure 2: **Zynq-7000 AP SoC Block Diagram**

Xilinx has adopted the ARM AMBA™ AXI3 interface to facilitate communication between the PS and a peripheral or processor subsystem on the PL.

The PS-PL interface includes the following AMBA AXI3 interfaces for primary data communication:

- Two 32-bit AXI master interfaces (M_AXI_GP)
- Two 32-bit AXI slave interfaces (S_AXI_GP)
- Four 64-bit/32-bit configurable buffered AXI slave interfaces with direct access to DDR memory and OCM, referred to as high-performance (HP) AXI ports (AXI_HP)
- One 64-bit AXI slave interface (ACP port) for coherent access to CPU memory

The General Purpose (GP) AXI ports provide a simple means to move data between the PS and the PL. Access to the PL is provided through the two GP AXI Master ports (M_AXI_GP), which each have a memory address range. From a software perspective, any instruction that performs a read or write to this memory range will originate PL AXI transactions on this interface.

The two GP AXI slave ports (S_AXI_GP) allow a PL AXI master to access the PS memory or any of the PS peripherals on the PS subsystem.

The GP AXI master and slave ports are the low-bandwidth interfaces contributed by the 32-bit data width and the central interconnect switch in the PS subsystem that adds latency to a PS or PL peripheral access.

The two highest-performance interfaces between the PS and the PL for data transfer are the high-performance (HP) AXI ports and the ACP interface.

The high-performance AXI (AXI_HP) ports provide high-bandwidth PL slave interface to the OCM and DDR3 memories of the PS. A data-intensive PL master should be connected to these HP interfaces to achieve higher throughputs. Each AXI_HP contains control and data FIFOs to provide buffering of transactions for larger sets of bursts, making it ideal for workloads such as video frame buffering in DDR. This additional logic and arbitration does result in higher minimum latency than other interfaces. Any data that is transferred by the PL master to the DDR is not coherent with the Cortex-A9 CPUs and requires software instructions to make the data coherent with the processor.

The AXI ACP interface provides a similar user IP topology as the HP (AXI_HP) ports. The ACP differs from the HP performance ports due to its connectivity inside the PS. The ACP connects to the CPU L2 cache, allowing ACP transactions to interact with the cache subsystems. This potentially decreases the total latency for data to be consumed by a CPU. These optionally cache-coherent operations can obviate the need to invalidate and flush cache lines, improving application performance.

Choosing the Right IP Interface for Optimal Power and Performance

Xilinx has adopted AXI version 4 protocol for Intellectual Property (IP) cores that enable Plug-and-Play IP to use high-performance, device-optimized, on-chip interconnects. This is described in detail in *Vivado Design Suite: AXI Reference Guide*. [Ref 1]

AXI4 supports variable data widths up to 256 bits, enabling large data movements in burst modes. The AXI4 protocol supports three kinds of interfaces optimized for different application requirements.

- **AXI4-Full** — The AXI-Full interface is a high-performance, memory-mapped interface that allows bursts of up to 256 data transfer cycles with just a single address phase. This interface is recommended for data-intensive tasks that support both single-beat transfers for simple register read/write operations and burst transfers for large memory transactions.
- **AXI4-Lite** — The AXI4-Lite interface is a subset of the AXI4 protocol intended for communication with simpler, smaller control-register style interfaces in components. The AXI4-Lite interface supports single-beat transfers, ideal for writing control information or reading status information to/from an IP in the PL. Though the AXI4-Full interface supports single-beat data, the AXI4-Lite interface is more power-efficient because of its lower pin count, and as a result it consumes less routing and logic resources.
- **AXI4-Stream** — The AXI4-Stream interface provides point-to-point high-speed streaming data from a high-speed interface that supports unlimited data burst sizes. This interface is recommended for transferring high-bandwidth streaming data like video.

The Create or Package IP Wizard in the Vivado® Design Suite provides templates for AXI4-Full, AXI4-Lite, and AXI-Stream interfaces, facilitating development of custom hardware IP. Designers can modify these templates to add their own custom logic, then package the design as an IP and add it back to the Vivado IP Integrator tool's block design.

More information on the Create or Package IP Wizard is available in Chapter 4 of the "Migrating to AXI4 Protocols" section of the *Vivado Design Suite: AXI Reference Guide*. [Ref 1]

Leverage Data Mover and AXI IP to Obtain Ideal Results

After an interface has been chosen, the next step is to find a suitable data-mover IP that can be used to transfer the data from the peripheral to the system memory. The Xilinx IP Catalog contains and describes all the supported IP infrastructure that enables connections to the PS subsystem.

Subsequent sections in this white paper describe the recommended interface on the PS and the corresponding data-mover and AXI infrastructure IPs for moving the data between the PS and PL to achieve the best performance. Also illustrated are some key reference designs that can be used as templates for new designs.

Understanding the breadth and depth of the surrounding ecosystem support can help reduce development time and associated design risks.

Basic Overview

In its most basic form, an AXI4 interface connects one master device to one slave device. For device and system architectures that require more than one master or more than one slave, Xilinx provides AXI interconnect and infrastructure IPs to manage this type of communication requirement.

The PS–PL interface is based on AXI Version 3 protocol. AXI interconnect IP is the key infrastructure that performs protocol conversion and enables data movement between the PS and PL interfaces.

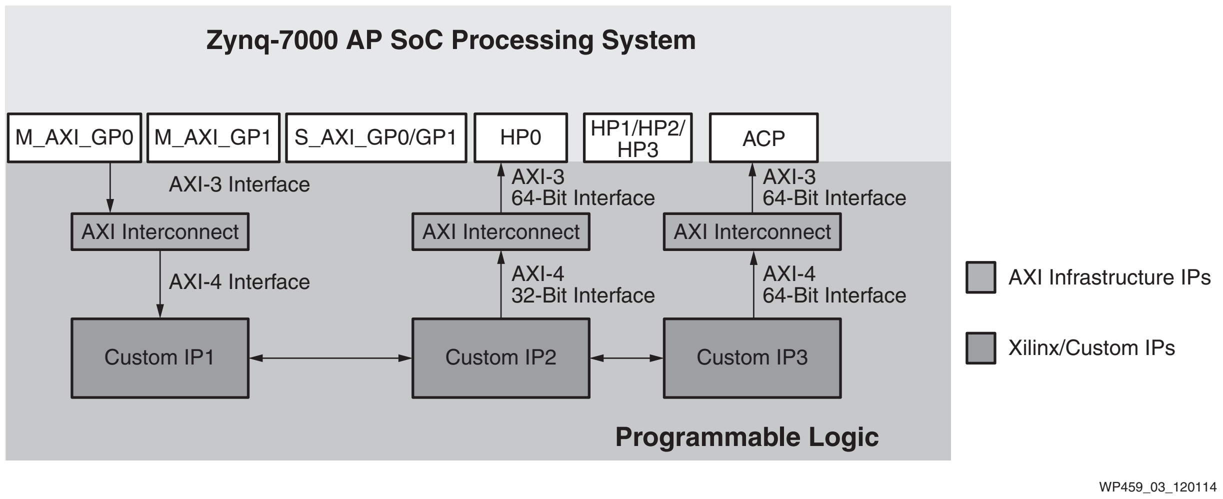

Figure 3: **AXI Interconnect Usage**

When connecting one master to one slave, the AXI interconnect can optionally perform address range checking. It can also perform any of the normal data width, clock-rate, pipelining, and protocol conversions. It also has built-in data-width conversion enabling the designer to easily hook up a 32-bit interface to a 64-bit HP port and ACP port of the optionally has built-in data FIFOs to manage the data rates to or from the PS and PL.

Xilinx's AXI4 Interconnect is also highly configurable to allow users to balance and tune for their design goal, e.g., area, timing, throughput, and latency. The Xilinx white paper *Maximize System Performance Using Xilinx Based AXI4 Interconnects* [Ref 2] provides recommendations on how to utilize the AXI interconnect IP for better performance.

Simple Register Read/Write

One of the most basic communication or data movements between the PS and PL is a simple read or write operation to a register available in the PL IP. These simple read/write transactions can be enabled via providing an AXI4-Lite or AXI4 interface on the IP. The AXI4 IP can be connected to GP AXI ports via the AXI Interconnect Infrastructure IP.

Most of the IPs in the Xilinx IP catalog have either an AXI4-Full or an AXI4-Lite interface to facilitate plug and play infrastructure capability using the Vivado IP Integrator tool. The IP registers can be accessed from the PS via the GP AXI ports, as shown in Figure 4.

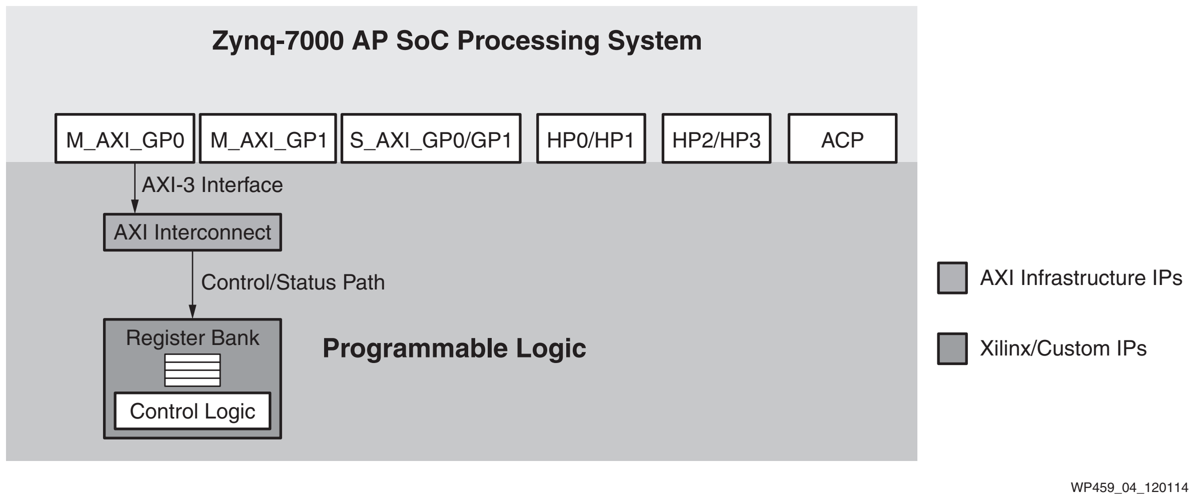

Figure 4: **AXI4-Lite Simple Register Read Write**

A Xilinx application note, *System Monitoring using the Zynq-7000 AP SoC Processing System with the XADC AXI Interface*[Ref 3], demonstrates a simple AXI-Lite register read/write interface. In this application note, a simple AXI-XADC IP is connected to the Master GP AXI Port 0 of the PS (Figure 5). The AXI4-Lite interface is used to configure the XADC for system monitoring applications.

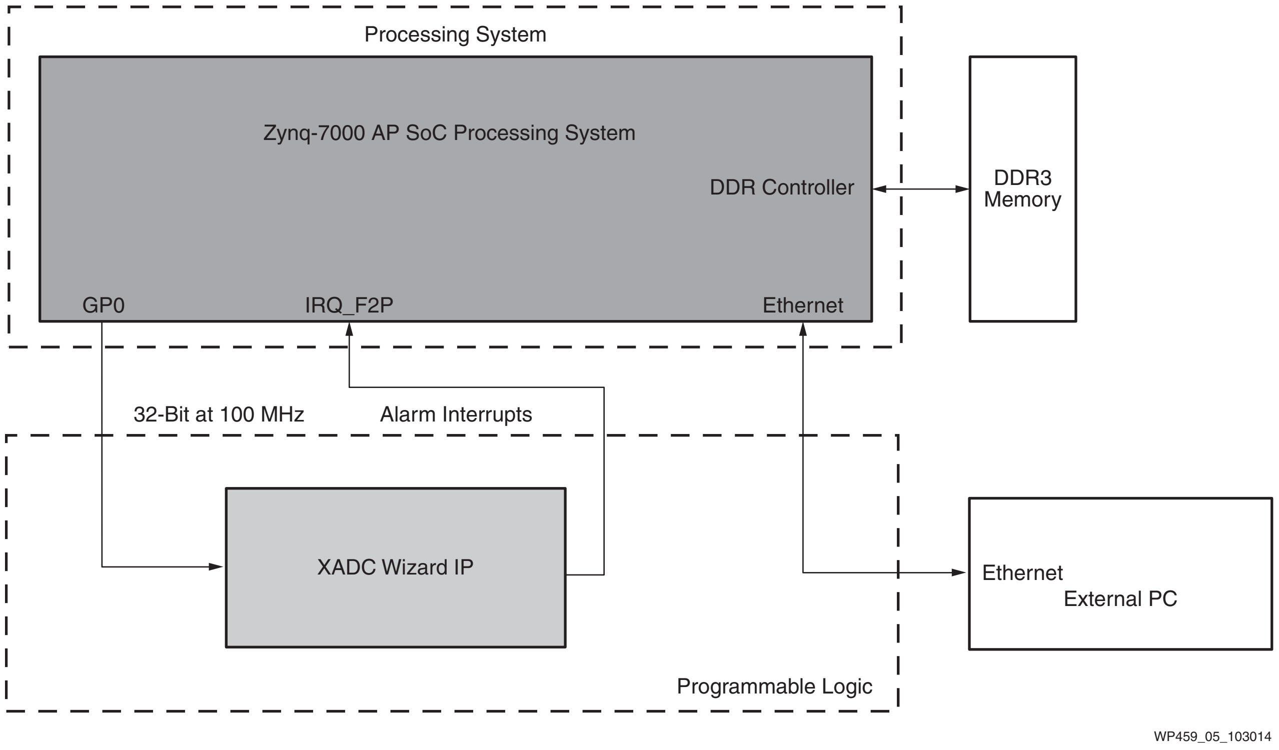

Figure 5: **System Monitoring Application using the Zynq-7000 AP SoC PS with the XADC AXI Interface**

For more information on this design, refer to the application note. [Ref 3]

AXI4-Memory Map to Streaming FIFO

The AXI4-Stream FIFO core converts AXI4/AXI4-Lite transactions to and from AXI4-Stream transactions, and can be used in applications that use packet communication.

The AXI4-Stream FIFO has three AXI4-Stream interfaces: one for transmit data, one for transmit control, and one for receive data. The AXI4-Stream transmit control interface can support the transmit protocol of AXI Ethernet cores.

The AXI4-Stream FIFO can be operated like a micro-DMA. This enables the core to be used to interface to AXI4-Stream IPs, similar to the LogiCORE™ IP AXI Ethernet core, without requiring a full DMA solution.

The registers of the AXI4-Stream FIFO can be accessed from the AXI4-Lite interface as well as the AXI4-Full interface. For data-intensive applications, it is recommended that:

- The AXI4-Stream FIFO register and datapaths be isolated
- AXI4-Lite be used for register access

AXI4-Full be used for the datapath. See Figure 6 for a block diagram of this implementation.

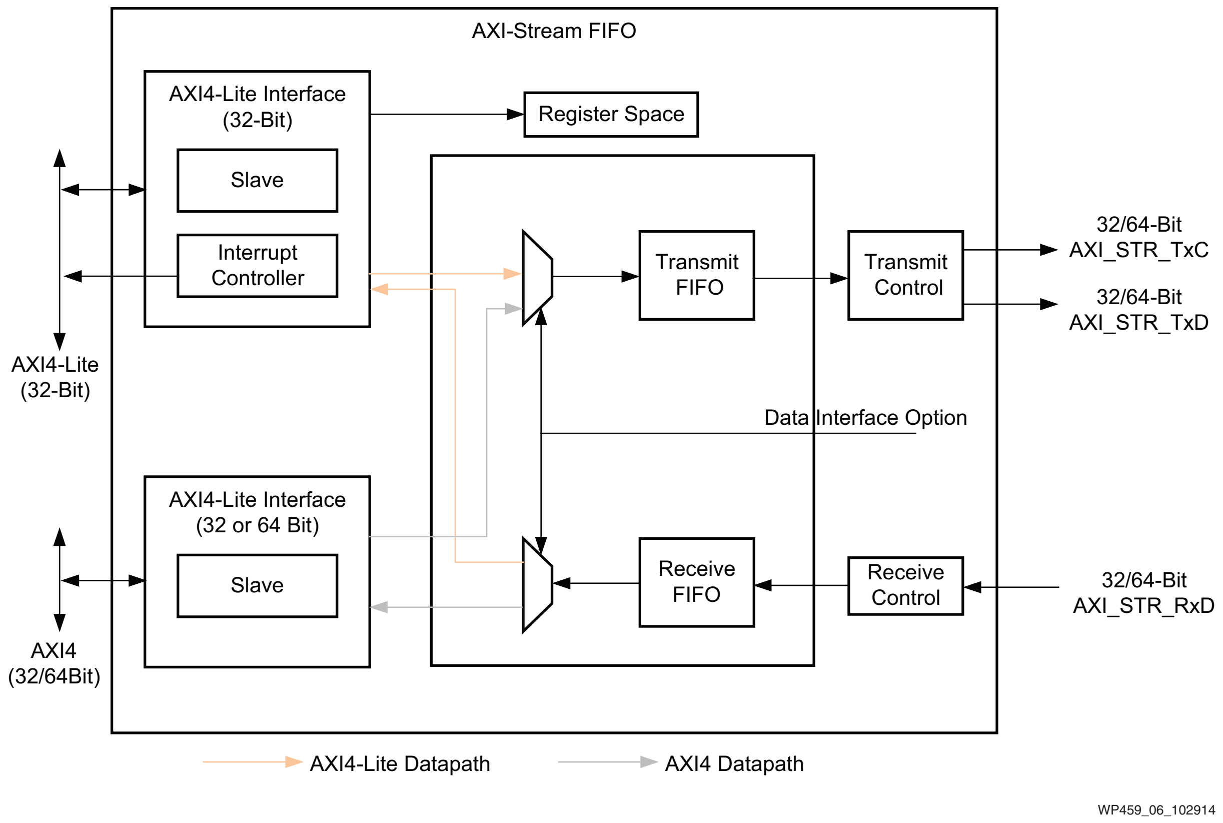

Figure 6: **AXI Streaming FIFO Block Diagram**

Figure 7 illustrates the recommended interconnection method for the AXI4-Stream FIFO.

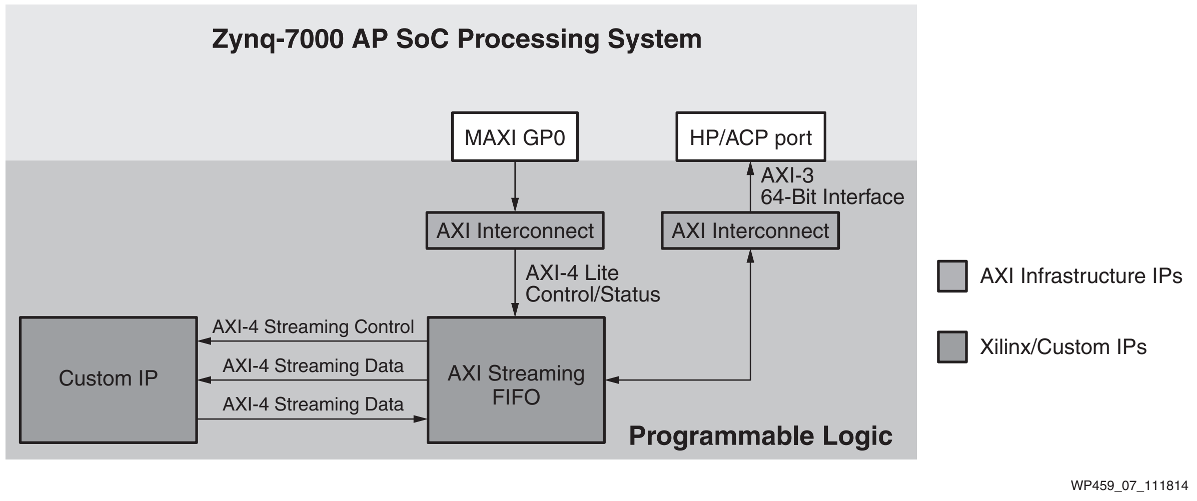

Figure 7: **Recommended Connection to AXI4-Stream FIFO**

The AXI4-Stream FIFO IP is recommended for medium-performance solutions, especially where light-weight packet transactions are required.

For more details on AXI4-Stream FIFO, see *LogiCORE IP AXI4-Stream FIFO Product Guide*[Ref 4].

AXI DMA

One of the key data-mover IPs in the Xilinx IP catalog is the AXI DMA IP core. The AXI Direct Memory Access (AXI DMA) IP provides high-bandwidth direct memory access between the AXI4 memory-mapped and AXI4-Stream IP interfaces. Designers who choose to use the DMA IP core should verify that they have the AXI-Stream interfaces for their IP.

The AXI DMA controller, moving primary high-speed DMA data between system memory and the stream target, routes through the AXI4 memory-mapped-to-stream master (MM2S); the AXI4 stream routes to the memory-mapped slave interface (S2MM) channels. *These channels operate independently*. The MM2S channel supports an additional AXI control stream for sending user application data to the target IP; an additional AXI status stream is provided for the S2MM channel for receiving user application data from the target IP. The DMA registers, accessed through the core's AXI4-Lite slave interface, should be connected to the PS GP master ports.

The optional scatter/gather port reads descriptors from system memory; it can be connected either to the GP slave ports or to HP ports.

This core block diagram is shown in Figure 8. For more information on AXI DMA, refer to the *LogiCORE IP AXI-DMA Product Guide*[Ref 5].

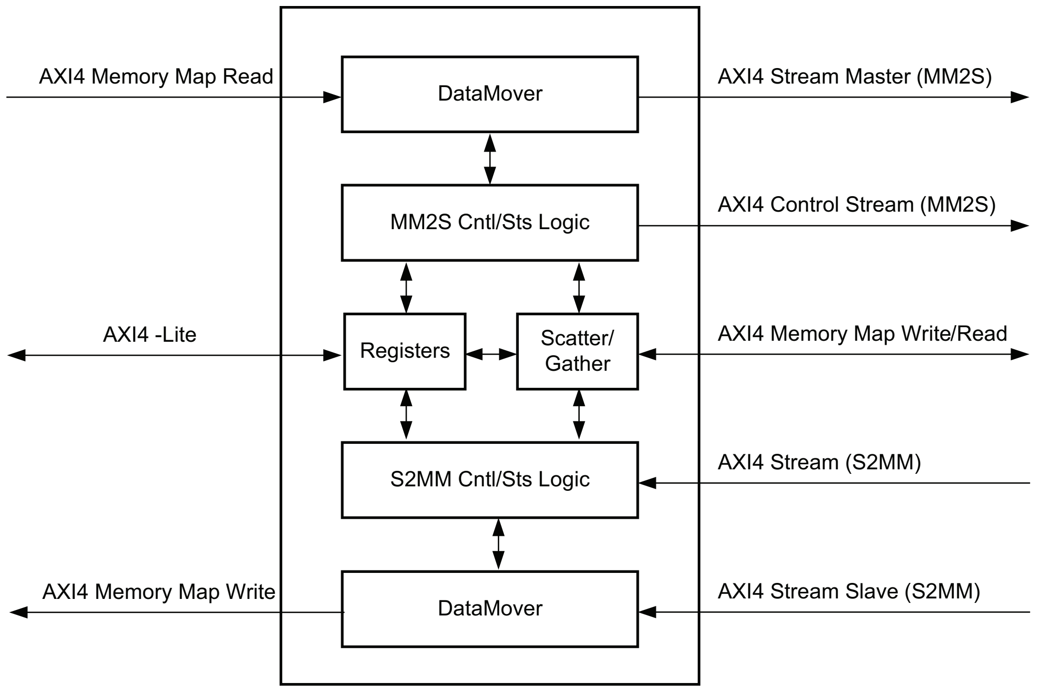

Figure 8: **AXI DMA Block Diagram**

When the AXI DMA core is used in a block design, the memory-mapped master port is connected to either of the PS AXI slave port interfaces. The slave interfaces of the PS include slave GP ports, HP ports, and the ACP port.

The slave GP ports provide access to PS peripherals and memory, but they have high latency and provide lower performance because of the data width and central interconnect delays. The PS ACP and high-performance AXI ports provide direct access to the PS memory, and provide lower latency than the GP ports. Therefore, it is recommended that the AXI DMA core memory-mapped master ports be connected to the ACP or to HP ports on the PS to achieve higher throughputs. See Figure 9.

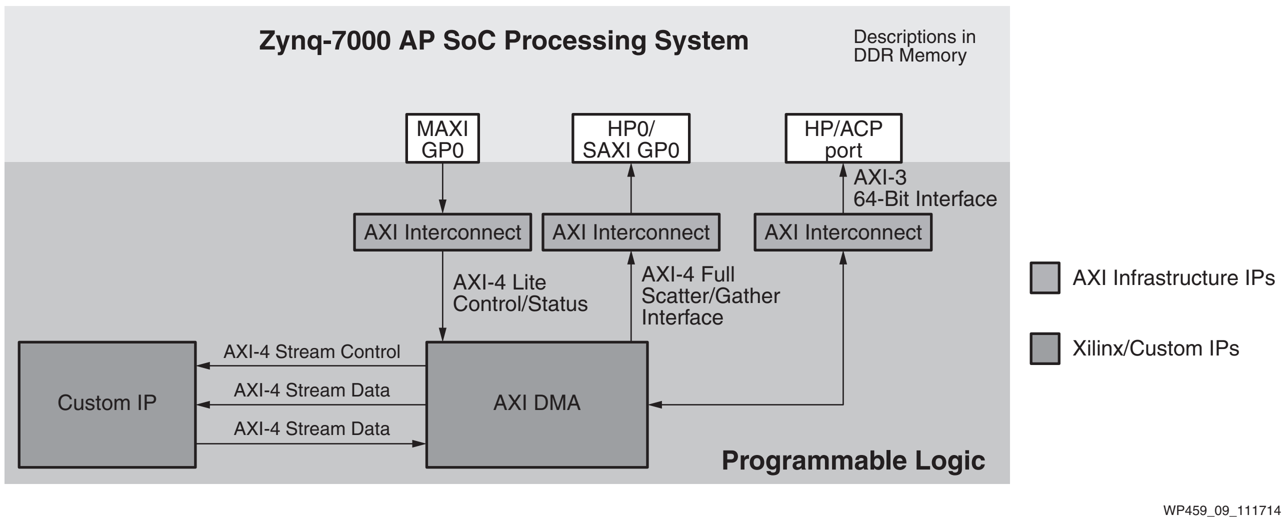

Figure 9: **Recommended AXI DMA Connections to Processing System**

The AXI DMA can operate in two different modes.

1. Register Direct Mode (Simple DMA)
2. Scatter/Gather Mode

Register Direct Mode (Simple DMA)

The AXI DMA in simple DMA mode can be used especially in cases where a high-speed transfer from a peripheral in PL is required. The AXI DMA in simple DMA mode provides configuration for doing simple DMA transfers that requires less FPGA resource utilization. In this mode, all the transfers are initiated by writing to the DMA control, source or destination address, and length registers. When the transfer is completed, an interrupt is asserted for the associated channel and, if enabled, generates an interrupt out.

There are multiple reference designs from Xilinx that illustrate the usage of AXI DMA in simple DMA mode.

Implementing Analog Data Acquisition using the XADC AXI Interface

A Xilinx application note, *Implementing Analog Data Acquisition using the Zynq-7000 AP SoC Processing System with XADC AXI Interface*[Ref 6], demonstrates a design in which the XADC data is transferred directly to system memory using the DMA IP core instead of by simple read/write operations.

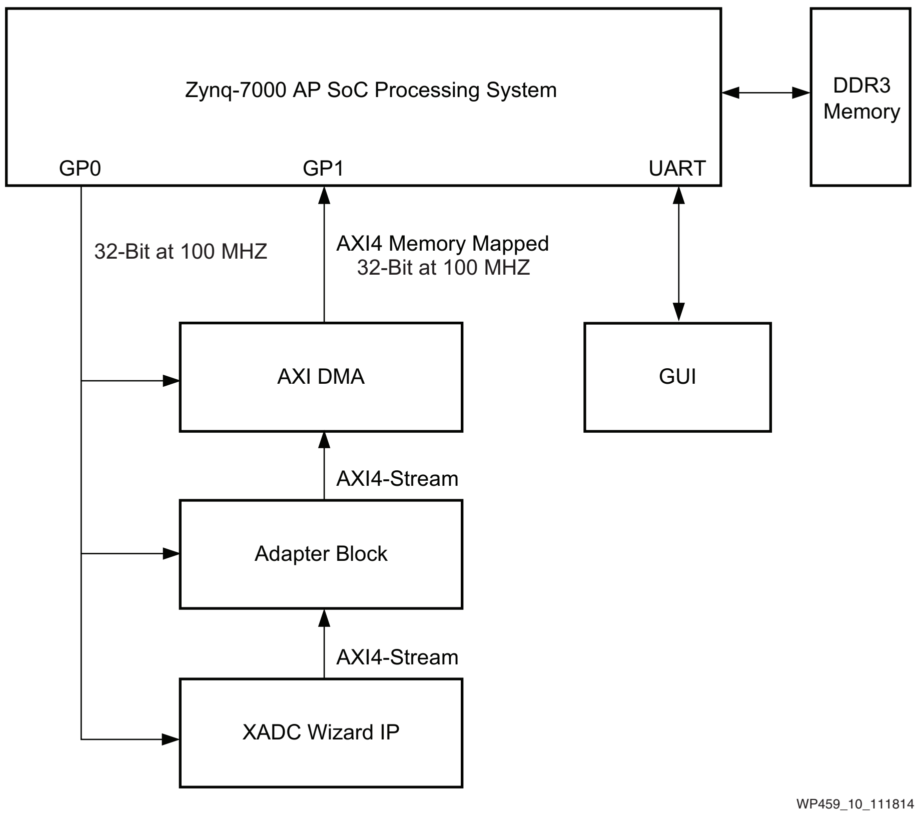

Figure 10: **Implementing Analog Data Acquisition using the PS with XADC AXI Interface**

For design and software implementation details, refer to the application note and related documentation.

Zynq-7000 AP SoC Accelerator for Floating-Point Matrix Multiplication using Vivado HLS

The AXI DMA IP core[Ref 5] also becomes a key infrastructure IP where the designer chooses to accelerate key software operations to hardware using high-level synthesis techniques. If the designer must move large amounts of data from the accelerated hardware implemented in PL to system memory, then DMA can serve the purpose. A Xilinx application note, *Zynq-7000 All Programmable SoC Accelerator for Floating-Point Matrix Multiplication using Vivado HLS*[Ref 7], demonstrates one such use case.

In this design (see Figure 11), AXI DMA performs all the burst formatting, address generation, and scheduling of memory transactions for communicating with the floating-point matrix multiplication accelerator IP core.

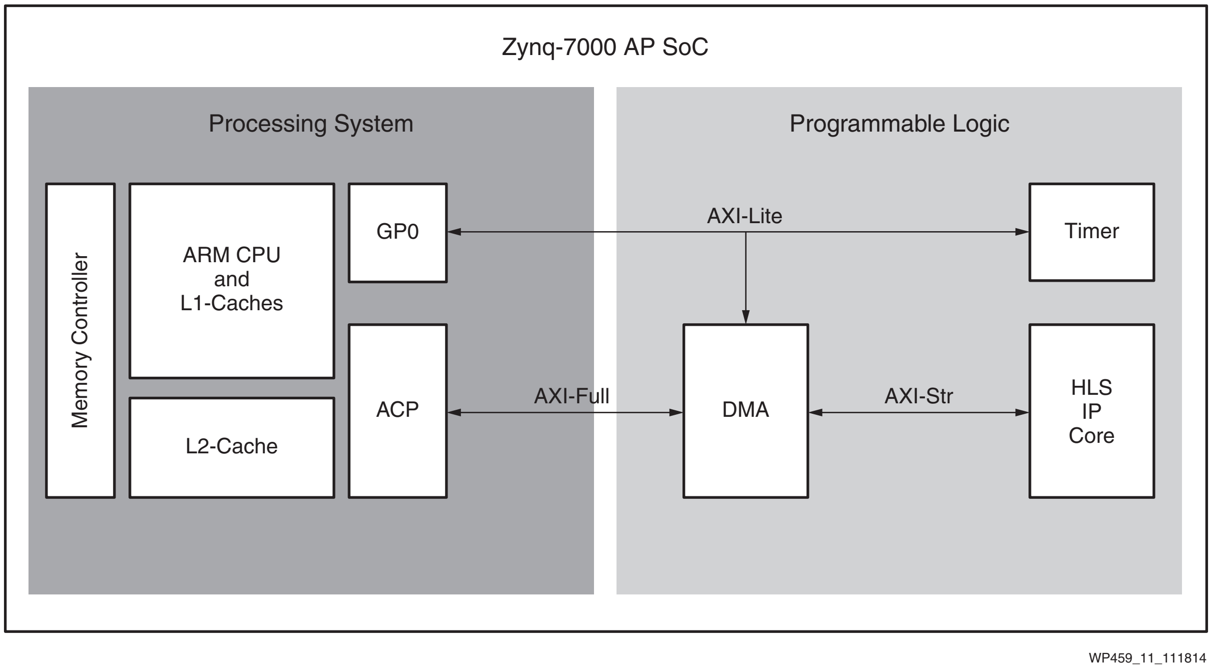

Figure 11: **Accelerator for Floating-Point Matrix Multiplication using Vivado HLS**

Refer to the application note to learn how a software program is modified to include AXI streaming interface, converted to an IP core, and connected to the PS using the DMA IP core[Ref 7].

Zynq-7000 AP SoC Spectrum Analyzer Reference Design

The Zynq-7000 AP SoC Spectrum Analyzer (see Figure 12) demonstrates a single-chip reference design for data acquisition and digital signal processing implementing a spectrum analyzer to demonstrate the capability of the Zynq-7000 AP SoC on the ZC702 development board.

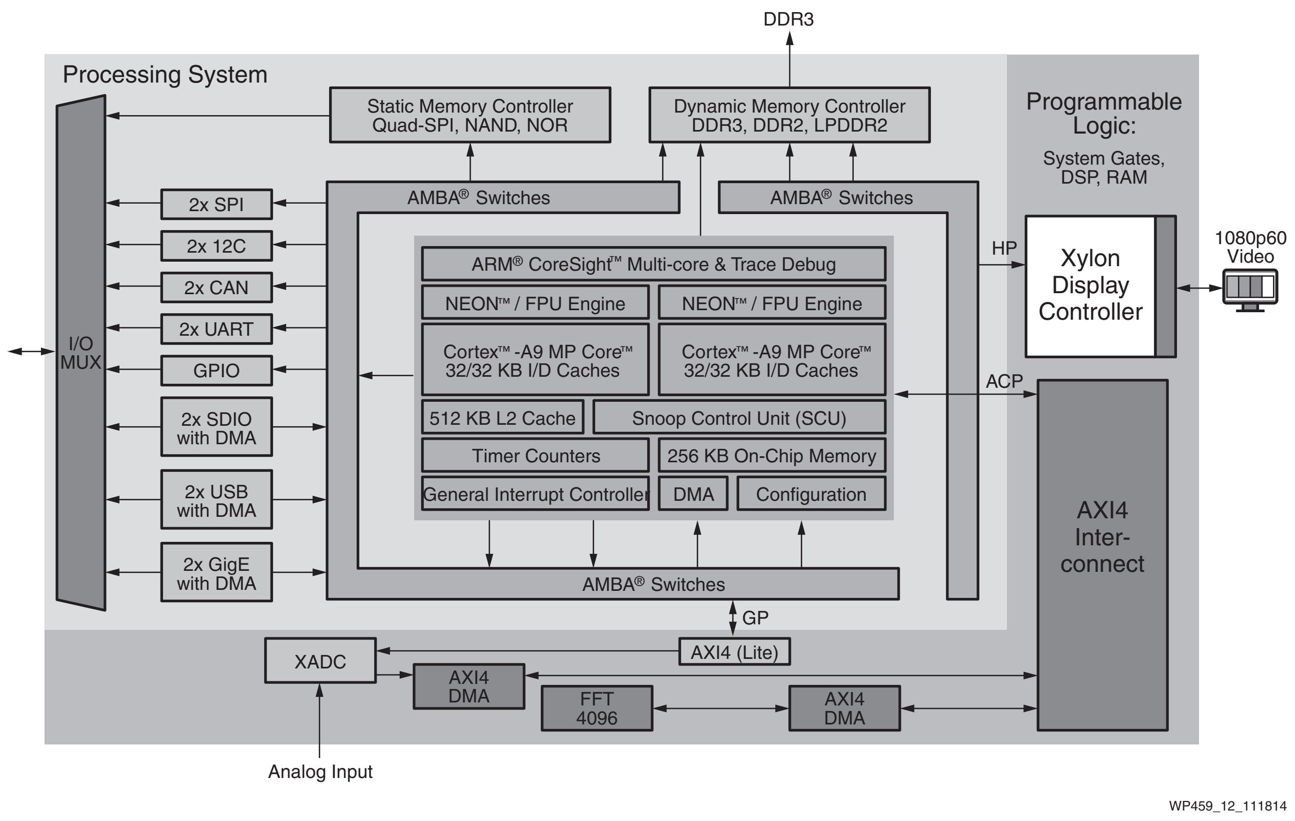

Figure 12: **Zynq-7000 AP SoC Spectrum Analyzer Reference Design**

In this design, one DMA IP core is used to transfer XADC samples directly to system memory via the ACP port, and another DMA IP core is used to transfer XADC samples from system memory to the Xilinx Complex FFT core to perform the complex FFT windows function.

Since the design demonstrates multiple technical references for building the spectrum analyzer design, the original design is split into multiple technical tips and can used for reference. The details and technical tips for this design are available in the technical articles section of the Xilinx Wiki page.

Scatter/Gather Mode

As described in the AXI DMA introduction, AXI DMA can be optionally configured to support the scatter/gather mode (see Figure 13). Unlike the simple DMA mode, the scatter/gather mode enables higher-performance DMA transfers at the cost of additional FPGA resource consumption.

The AXI DMA has an optional AXI4 scatter/gather read/write master interface through which the scatter/gather engine fetches and updates buffer descriptors from system memory. Unlike the simple DMA mode, the scatter/gather engine requires optional descriptors queued in the system memory. This offloads DMA management work (writing the address, length, and control registers) from the CPU, and instead fetches and updates the information automatically from the descriptor table, thus maximizing primary data throughput.

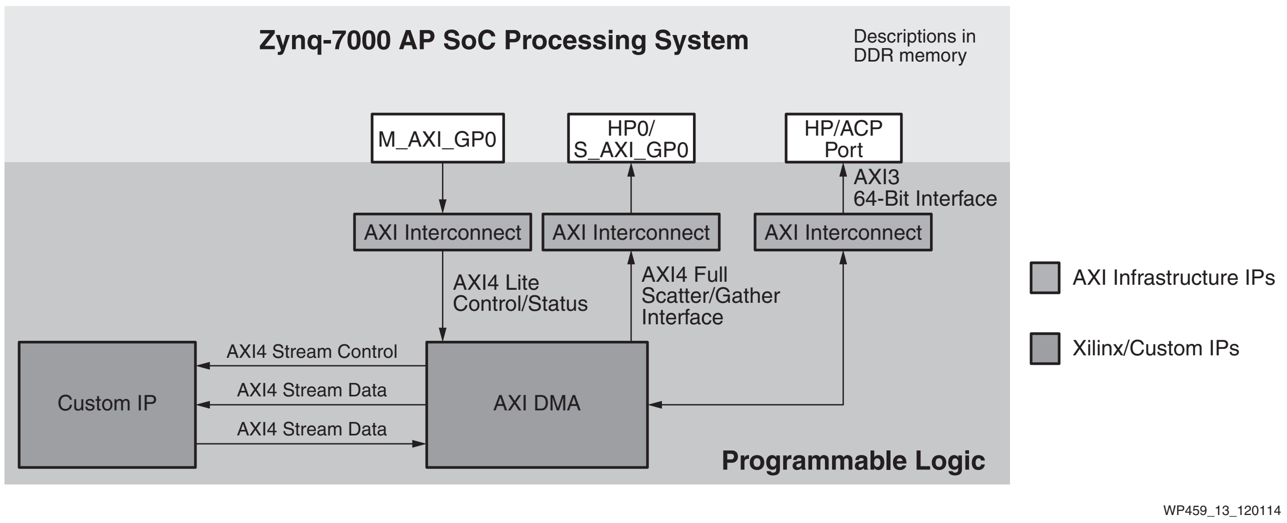

Figure 13: **Recommended Connections to AXI DMA Core**

The operation of the scatter/gather mechanism and descriptor formats is explained in the *LogiCORE IP AXI DMA Product Guide* (PG021).[Ref 5]

The following reference designs from Xilinx illustrate the usage of AXI-DMA in Scatter/Gather Mode.

Zynq-7000 AP SoC PL Ethernet Performance and Jumbo Frame Support

A Xilinx application note, *PS and PL Ethernet Performance and Jumbo Frame Support with PL Ethernet in the Zynq-7000 AP SoC*[Ref 8], demonstrates high-speed data movement between AXI Ethernet IP and system memory using the AXI DMA IP core.

The application note focuses on multiple aspects of the Ethernet peripheral usage in the Zynq-7000 AP SoC, like connecting the gigabit Ethernet MAC (GEM) through the extended multiplexed I/O (EMIO) interface with the 1000BASE-X physical interface using high-speed serial transceivers in PL.

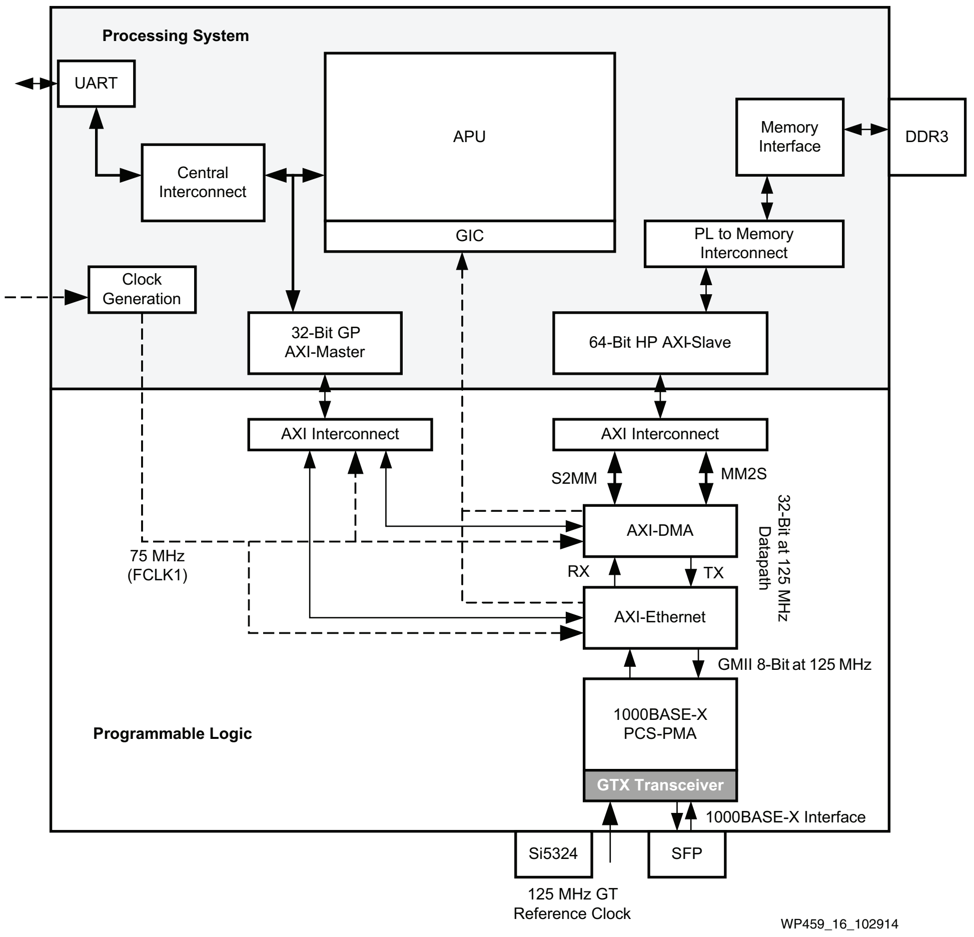

Figure 14: **Zynq-7000 AP SoC Ethernet Performance and Jumbo Frame Support with PL Ethernet**

In the PL implementation of Ethernet, the design consists of the AXI Ethernet, AXI DMA, and AXI Interconnect IP cores. The design uses the HP port for fast access to the PS-DDR memory; however, the GP slave port can also be used if the HP port is occupied with other peripherals. The application note also demonstrates the effective use of AXI-DMA control and status ports, which transfers the critical control and status information between the Ethernet MAC Controller and system memory required for the Ethernet device operation.

For design and software implementation details, refer to the application note.

AXI-Video DMA

High-performance video systems can be created using the Zynq-7000 AP SoC and available Xilinx LogiCORE IP AXI Interface cores. AXI interconnects and AXI video DMA IP cores enable the design of video systems capable of handling multiple video streams and multiple video frame buffers, sharing a common DDR3 SDRAM via the AXI3 ports on the Zynq-7000 AP SoC.

The AXI Video Direct Memory Access (AXI VDMA) core is designed to allow for efficient high-bandwidth access between AXI4-Stream video interface and AXI4 interface that can be connected to the HP ports of the PS interface.

The AXI VDMA operation is similar to the AXI-DMA configured in simple DMA mode but is optimized for video protocol. The core provides efficient two-dimensional DMA operations with independent asynchronous read and write channel operation. Initialization, status, interrupt, and management registers are accessed through an AXI4-Lite slave interface.

Figure 15 illustrates the blocks of the AXI VDMA IP core.

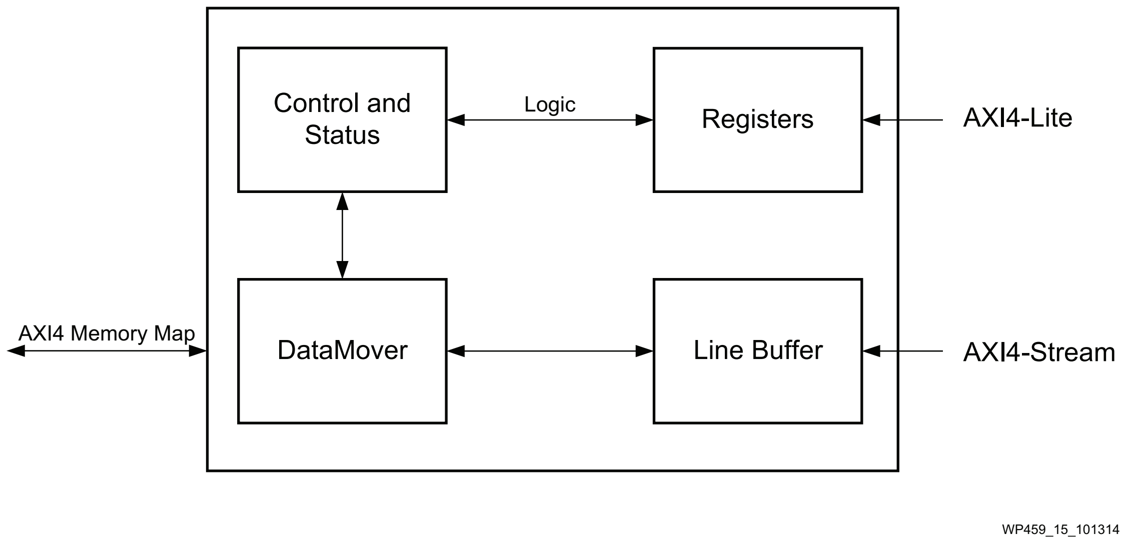

Figure 15: **Video DMA IP Core**

For more information on the AXI-VDMA IP, refer to the *LogiCORE IP AXI Video Direct Memory Product Guide*[Ref 9].

The following reference designs from Xilinx illustrate the usage of AXI-VDMA Core.

High-Performance Video Systems Using IP Integrator

A Xilinx application note, *Designing High-Performance Video Systems with the Zynq-7000 All Programmable SoC Using IP Integrator*[Ref 10], demonstrates the AXI VDMA core's video capability using the on Zynq-7000 SoC ZC702 development board. This design uses four AXI VDMA cores to simultaneously move eight streams (four transmit video streams and four receive video streams), each in 1920 x 1080p format at a 60 Hz refresh rate, with up to 24 data bits per pixel.

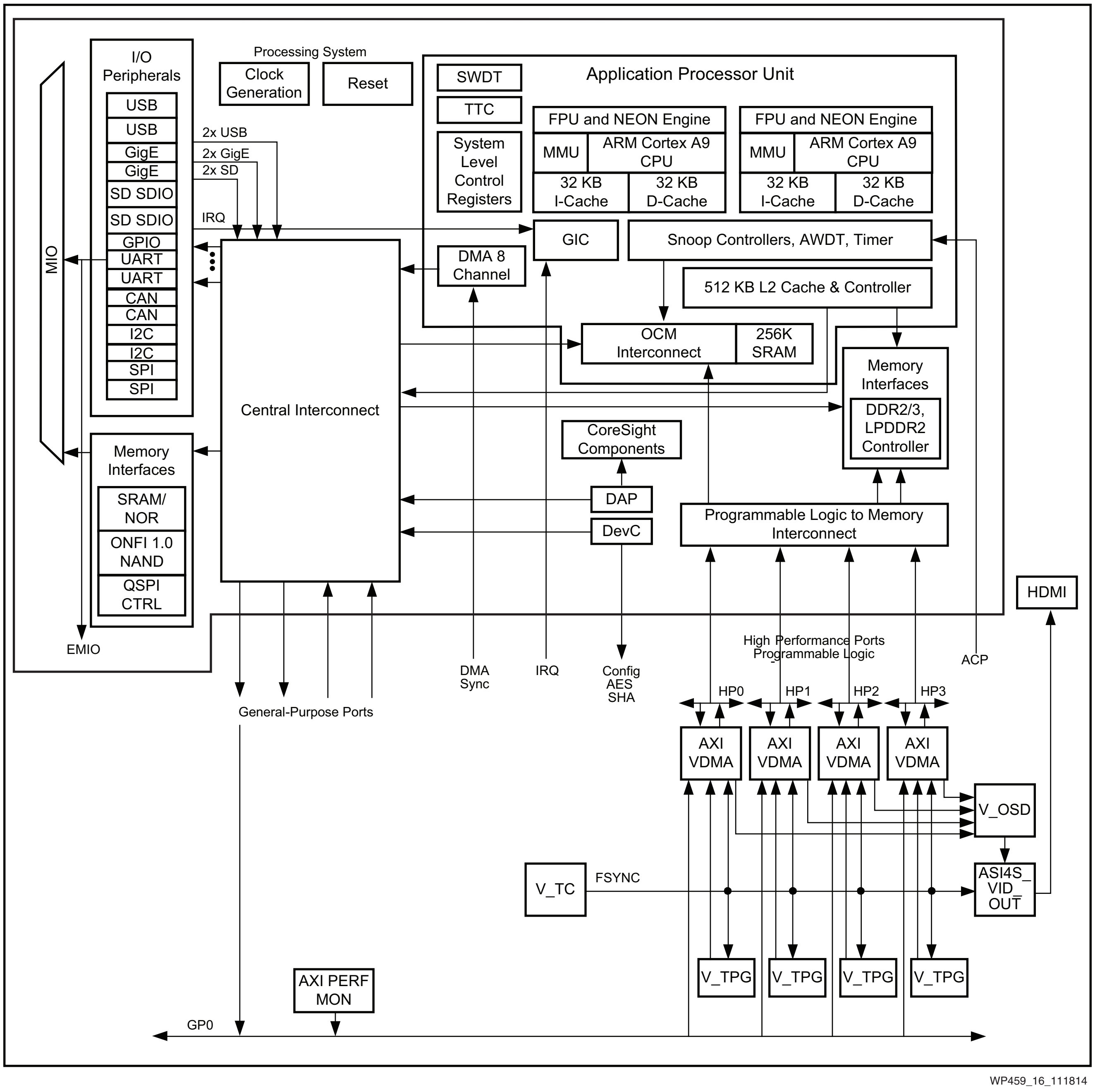

Figure 16: **Designing High-Performance Video Systems Using the Vivado IP Integrator**

The design is built using the Vivado Design Suite and the Vivado IP integrator feature. The design also includes software built using the Xilinx Software Development Kit (SDK). The complete IP integrator project and SDK tool project files are provided with this application note, allowing the designer to examine and rebuild this design or use the files as a template for starting a new design.

Zynq-7000 AP SoC Base Targeted Reference Design

The *Zynq Base Targeted Reference Design*[Ref 11], also built on the same principle, demonstrates the AXI VDMA usage and its connections. The Base TRD consists of two processing elements: the Zynq-7000 AP SoC PS and a video accelerator built in PL.

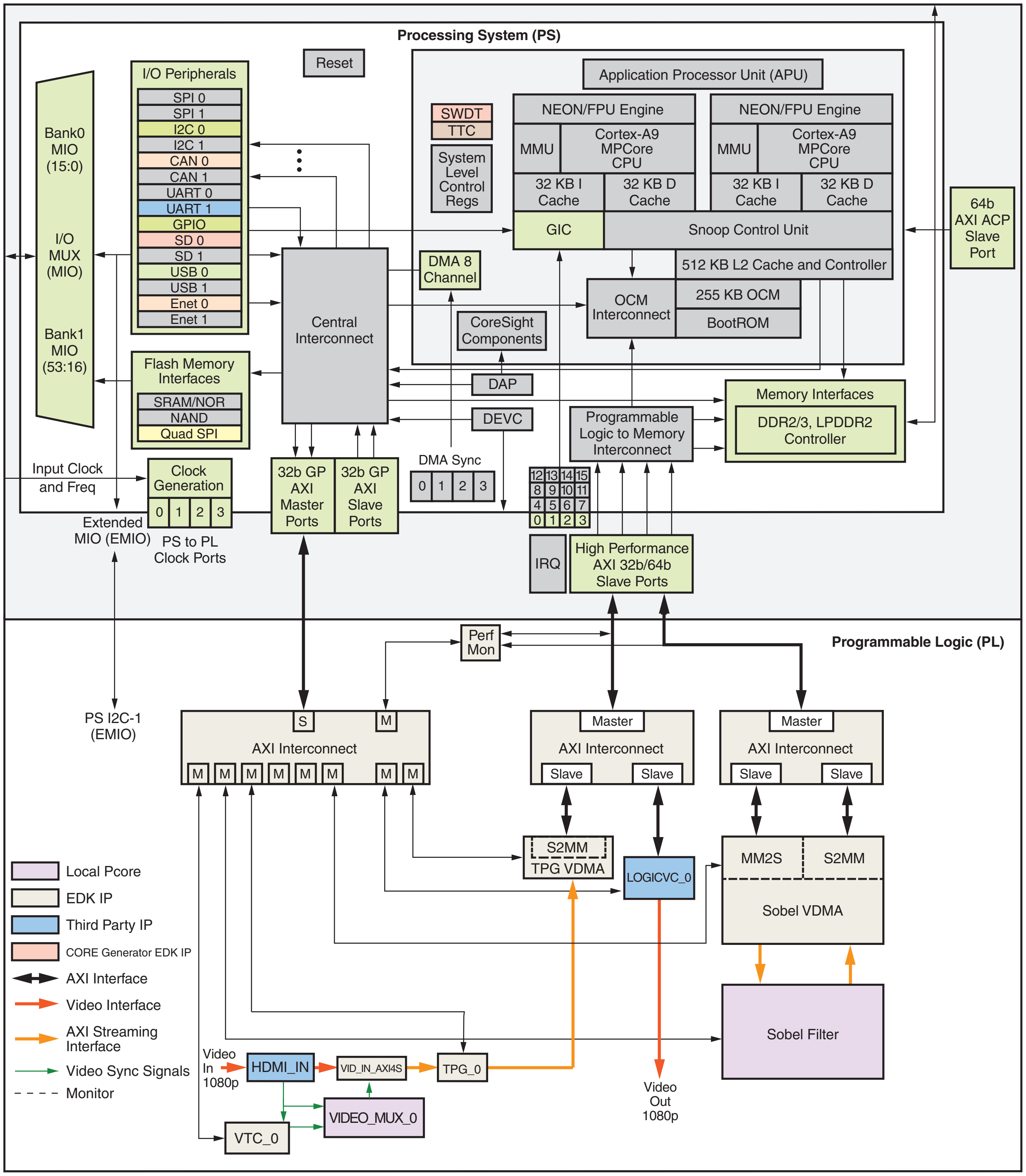

Figure 17: **Zynq-7000 AP SoC Base TRD**

The video accelerator in PL implements the AXI VDMA core which redirects video traffic generated by the Test Pattern Generator core and the Sobel Engine to DDR3 memory via the HP ports.

The AXI video DMA core in this design contains an AXI4-Lite slave interface with which the software configures the core for 1080p, where each horizontal line is set for 5,760 bytes (1,920 x 3 bytes per pixel), and each vertical line is set for 1,080 pixels by the processor. There are twelve frame buffers in this design (three frame buffers x four video pipelines).

Designers can leverage both the above reference designs to build a video application using the Zynq-7000 AP SoC.

The design files for this reference design can be downloaded from Xilinx Wiki Site at http://www.wiki.xilinx.com/Targeted+Reference+Designs#Zynq.

Zynq-7000 AP SoC PS DMA Controller (PL330 DMA)

The PS subsystem includes a DMA controller (DMAC) block that can be used to transfer data from a PS peripheral or PL peripherals to the system memory. The DMAC provides a flexible DMA engine that can provide moderate levels of throughput with little PL logic resource usage.

The DMAC supports up to eight channels. However, this flexible programmable model could increase software complexity relative to simple CPU transfer or specialized a DMA block implemented in PL.

The DMAC interface to the PL is through the GP AXI master interfaces, and a peripheral request interface also allows PL slaves to provide status to the DMAC on buffer state. This enables the PL peripheral to prevent stalled transactions due to interconnect and DMAC buffer management, as shown in Figure 18.

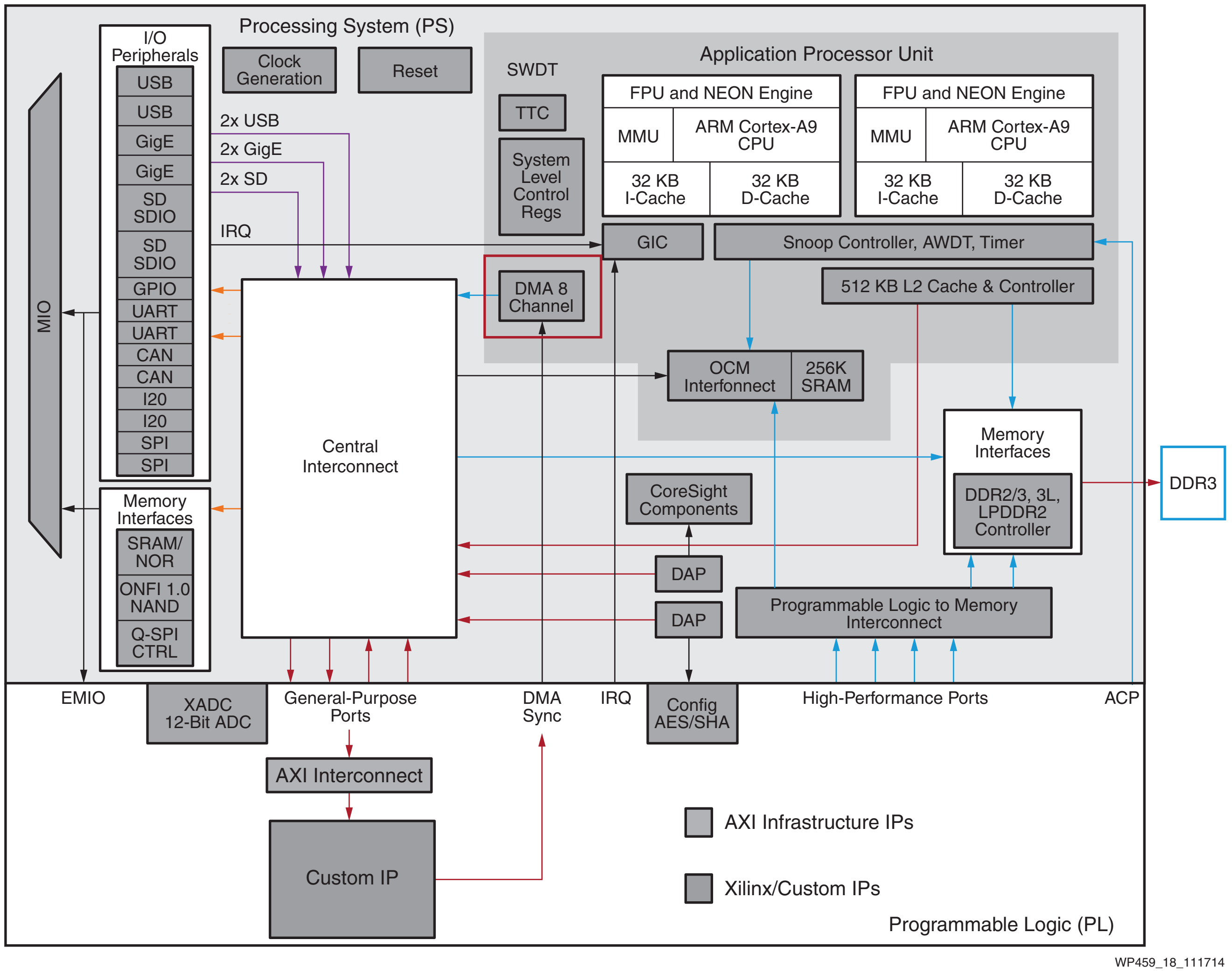

Figure 18: **Connections for PL330 DMA**

The PL330 DMA driver is integrated into the standard Linux DMA API framework. The hardware DMA components in the Zynq-7000 device are controlled through the standard Linux DMA API.

AXI-CDMA

The AXI CDMA is another key data-mover IP in the Xilinx IP catalog that performs provides high-bandwidth direct memory access between a memory-mapped source address and a memory-mapped destination address using the AXI4 protocol. The AXI CDMA IP provides an optional Scatter Gather (SG) feature to off-load control and sequencing tasks from the System CPU. The Initialization, status, and control registers of the CDMA are accessed through an AXI4-Lite slave interface.

The AXI CDMA is an ideal IP to move blocks of the data from buffers in the PL to the Zynq DDR memory via the Zynq PS-PL Interfaces. Chapter 6 of the Zynq-7000 All Programmable SoC: Concepts, Tools, and Techniques (CTT) guide provides a design example to configure the CDMA block to move data between the PS-PL interfaces.

For more information on AXI CDMA refer to the LogiCORE IP AXI CDMA Product Guide (PG034).

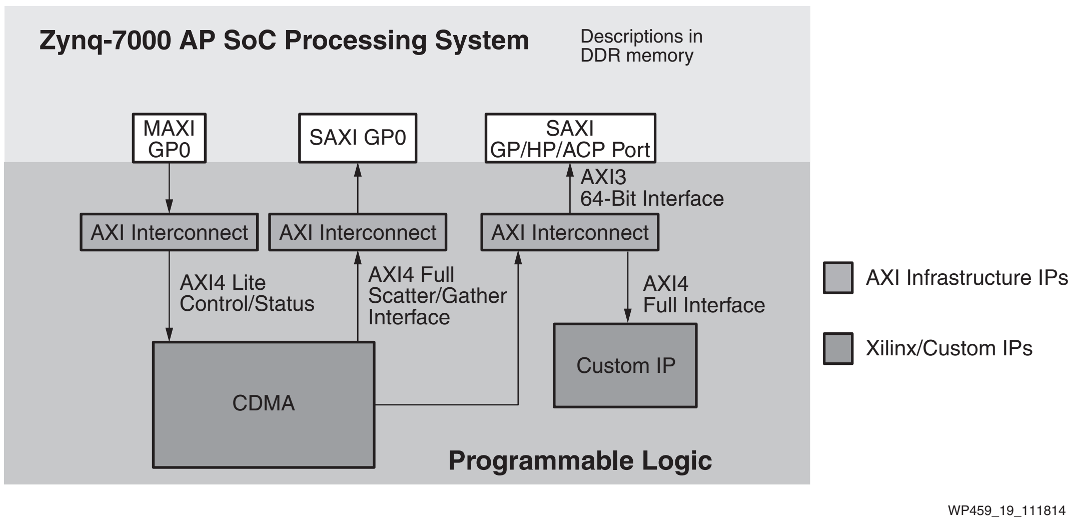

Figure 19: **Connections to AXI CDMA IP**

DMA for PCI Express Based Designs

To implement a PCIe Express® based design using Zynq-7000 AP SoC, Xilinx offers a couple of reference designs that use DMA to transfer data between the PCI Express core and system memory.

The PCI Express core in the PS can be used as Root Complex as well as Device Endpoint Block. The reference designs below illustrate PCI Express as an Endpoint Block.

PCI Express Endpoint – DMA Initiator Subsystem

A Xilinx application note, *PCI Express Endpoint-DMA Initiator Subsystem*[Ref 12], demonstrates a subsystem for Endpoint-initiated DMA data transfers through PCI Express. The provided subsystems target the indicated devices to initiate data transfers between DDR3 memory and an externally connected PCI Express Root Complex.

The block diagram for this reference designs and its connection to the device is shown in Figure 20.

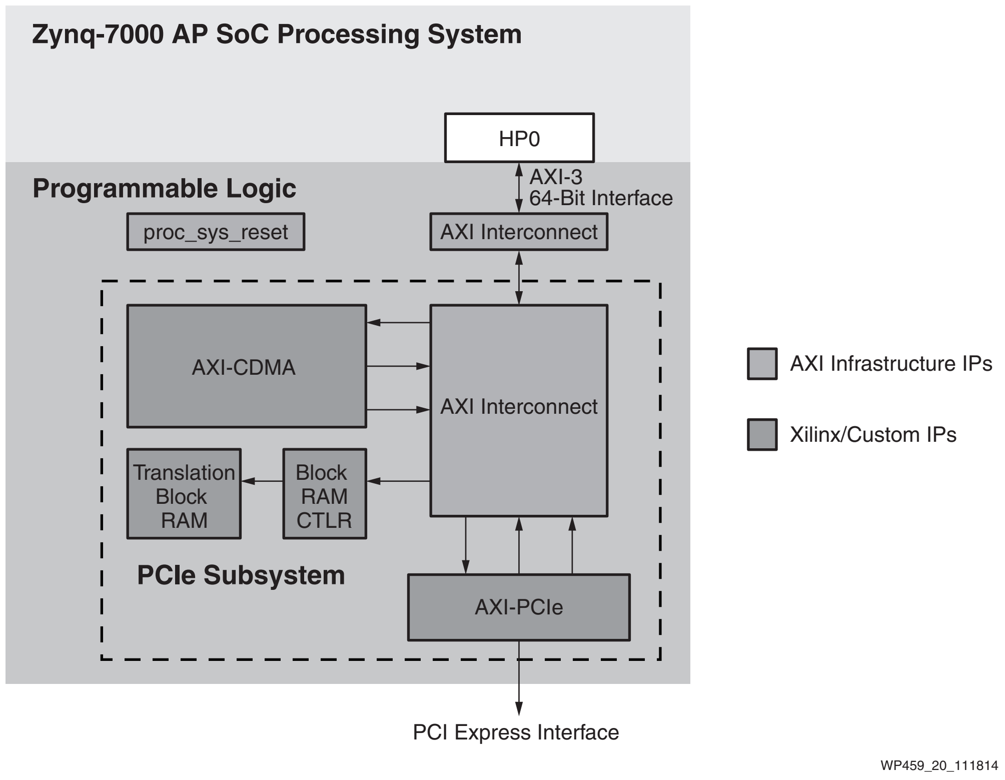

Figure 20: **PCI Express Endpoint-DMA Initiator Subsystem Reference Design**

To accomplish this, a scatter/gather-capable DMA engine is paired with the PCI Express IP along with AXI Interconnect IP cores This design demonstrates how to map AXI Memory to PCI Express IP to perform high-throughput data transfers over a PCI Express link. The DMA engine enables the system to manage data transfer over the PCI Express link to increase throughput and decrease processor utilization on the Root Complex side of the PCI Express link. This approach is important specifically for high-throughput PCI Express applications, which can include using the Zynq-7000 SoC PS high-performance ports or double-data-rate (DDR) memory.

For design and software implementation details, refer to the application note.

PCI Express Targeted Reference Design (TRD)

The *Zynq-7000 AP SoC ZC706 Evaluation Kit Getting Started Guide (ISE Design Suite 14.7)*[Ref 13] provides a reference solution to implement a PCI Express Base Targeted Reference Design (TRD), which expands the *Zynq-7000 Base Targeted Reference Design*[Ref 11] by adding PCI Express communication with a host system communicating at x4 Gen2 speed.

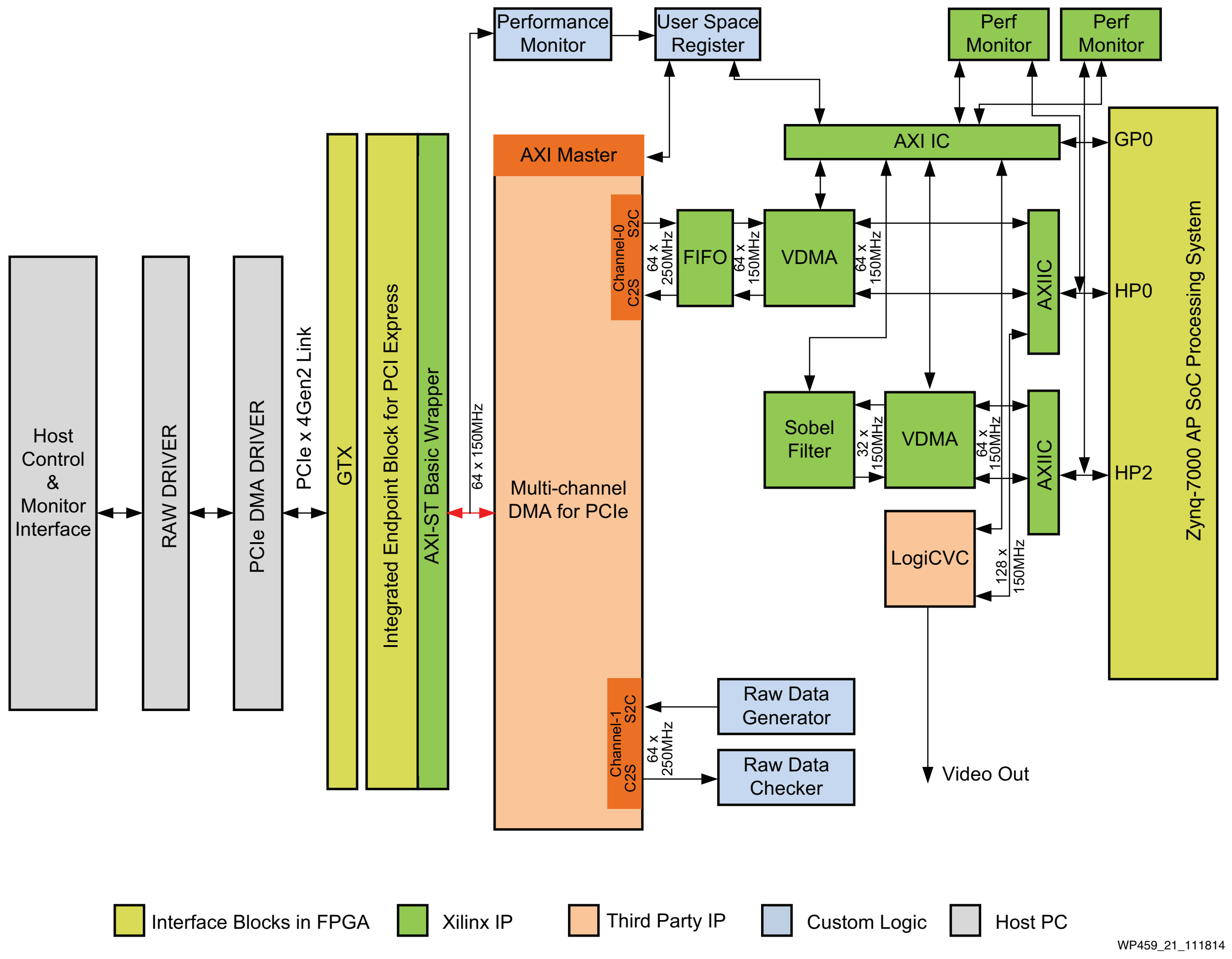

Figure 21: **PCI Express TRD**

In this reference design, Xilinx utilizes a third-party, multi-channel, bus-mastering scatter/gather PCIe Express DMA from Northwest logic. The Northwest Logic DMA Core provides high-performance, scatter-gather DMA operation in a flexible fashion.

The Northwest Logic DMA core provides an AXI4-Stream interface for data and an AXI4 interface for register space access. The Northwest Logic DMA core also provides independent transmit and receive channels for each channel, allowing full-duplex operation.

The PCI Express DMA translates the stream of PCI Express video data packets into AXI-Stream data, which is connected to a Video DMA (VDMA). Software running in the PS on the Cortex-A9 processor manages the AXI VDMA and transfers the raw video frames into the PS DDR3 memory.

Information regarding the Northwest Logic DMA core is available on the Xilinx website at http://www.xilinx.com/products/intellectual-property/1-8DYF-1689.htm.

The design files for this reference design can be downloaded from the Xilinx Wiki Site page at http://www.wiki.xilinx.com/Targeted+Reference+Designs#Zynq.

Summary

Table 1 summarizes the recommended interface on the PS and the corresponding data-mover IP for moving the data between the PS and PL. In using Table 1, first locate a body cell containing a description of the task and the performance required. Then refer to the row of headings above and the column of headings to the left to identify the best PS interface and the best data-mover IP for those requirements, respectively. If these interface results do not look quite right or are clearly inappropriate for the project at hand, try to find a similar task/performance description. The best interface models shown in the headings now might be more in line with the rest of the design.

Table 1: **Zynq-7000 SoC PS Interface Usage vs. Data-Mover IP**

Data-Mover IP	Zynq-7000 SoC PS Interface Usage			
	GP AXI Master Port	**GP AXI Slave Port**	**HP AXI Port**	**ACP (Accelerated Coherency Port)**
Simple Register Read / Write	Control and interrupt data to AXI PL peripheral.	Control and interrupt data to PS.	—	—
AXI FIFO MM2S	Control and interrupt data or low to medium performance data transfer between PS and streaming PL IP.	Control and interrupt data, or low to medium performance data transfer between streaming PL IP and PS memory.	Low to medium performance data transfer between streaming PL IP and PS memory.	Low latency data transfer between PL IP and PS Memory; data results to be in L2 cache, coherent with processors. Improves application performance.
DMA (Simple)	Control and interrupt data to PL.	—	Medium performance data transfer between streaming PL IP and PS memory.	Low latency data transfer between PL IP and PS Memory; data results to be in L2 cache, coherent with processors. Improves application performance.
DMA (Scatter / Gather)	Control and interrupt data to PL.	—	High performance data transfers between streaming PL IP and PS memory.	Low latency data transfer between PL IP and PS Memory; data results to be in L2 cache, coherent with processors. Improves application performance.
PL330 DMA	Control and interrupt data to PL, or low to medium performance data transfer between memory-mapped PL IP and PS memory.	Low to medium performance data transfer between memory Mapped PL IP and PS memory or PS peripherals.	—	—

Table 1: **Zynq-7000 SoC PS Interface Usage vs. Data-Mover IP *(Cont'd)***

Data-Mover IP	**Zynq-7000 SoC PS Interface Usage**			
	GP AXI Master Port	**GP AXI Slave Port**	**HP AXI Port**	**ACP (Accelerated Coherency Port)**
Video DMA	Control and interrupt data to PL.	—	High performance video pixel information between video IP in PL and PS memory.	—
PCI Express DMA	Control and interrupt data to PL.	—	High performance data transfers between PCI Express core in PL and PS memory.	Low latency data transfer between PCI Express Core in PL and PS Memory; data results to be in L2 cache, coherent with processors. Improves application performance.

Conclusion

This white paper lists the most popular data-mover IPs from Xilinx with the appropriate reference designs to which designers can refer to implement a wide range of systems. The design examples and recommendations provided in this document help designers choose the best data-mover IP much earlier in the design phase. Faster time to market is a competitive advantage; choosing Xilinx FPGAs, Targeted Design Platforms, AXI4 IP, and industry-standard AXI4 support provides that competitive edge.

References

1. *Vivado Design Suite: AXI Reference Guide* (UG1037)
2. *Maximize System Performance Using Xilinx Based AXI4 Interconnects* (WP417)
3. *System Monitoring using the Zynq-7000 AP SoC Processing System with the XADC AXI Interface* (XAPP1182)
4. *LogiCORE IP AXI4-Stream AXI FIFO Product Guide* (PG080)
5. *LogiCORE IP AXI DMA Product Guide* (PG021)
6. *Implementing Analog Data Acquisition using the Zynq-7000 AP SoC Processing System with XADC AXI Interface* (XAPP1183)
7. *Zynq-7000 All Programmable SoC Accelerator for Floating-Point Matrix Multiplication using Vivado HLS* (XAPP1170)
8. *PS and PL Ethernet Performance and Jumbo Frame Support with PL Ethernet in the Zynq-7000 AP SoC* (XAPP1082)
9. *LogiCORE IP AXI VDMA Product Guide* (PG020)
10. *Designing High-Performance Video Systems with the Zynq-7000 All Programmable SoC Using IP Integrator* (XAPP1205)
11. *Zynq Base Targeted Reference Design* (UG925)
12. *PCI Express Endpoint-DMA Initiator Subsystem* (XAPP1171)
13. *Zynq-7000 AP SoC ZC706 Evaluation Kit Getting Started Guide (ISE Design Suite 14.7)* (UG961)

Further Reading

1. *Zynq-7000 All Programmable SoC Technical Reference Manual* (UG585)
2. *Xilinx Design Tools: Vivado Design Suite, Embedded Development Kit (XPS)*

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
01/13/2015	1.0	Initial Xilinx release.

Disclaimer

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at http://www.xilinx.com/legal.htm#tos; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at http://www.xilinx.com/legal.htm#tos.

Automotive Applications Disclaimer

XILINX PRODUCTS ARE NOT DESIGNED OR INTENDED TO BE FAIL-SAFE, OR FOR USE IN ANY APPLICATION REQUIRING FAIL-SAFE PERFORMANCE, SUCH AS APPLICATIONS RELATED TO: (I) THE DEPLOYMENT OF AIRBAGS, (II) CONTROL OF A VEHICLE, UNLESS THERE IS A FAIL-SAFE OR REDUNDANCY FEATURE (WHICH DOES NOT INCLUDE USE OF SOFTWARE IN THE XILINX DEVICE TO IMPLEMENT THE REDUNDANCY) AND A WARNING SIGNAL UPON FAILURE TO THE OPERATOR, OR (III) USES THAT COULD LEAD TO DEATH OR PERSONAL INJURY. CUSTOMER ASSUMES THE SOLE RISK AND LIABILITY OF ANY USE OF XILINX PRODUCTS IN SUCH APPLICATIONS.