IP-ESC'09

Scatter-Gather DMA IP Core for PLDA EZDMA IP

Dmitry Smekhov, Instrumental Systems

Moscow Russia

Abstract :

Description of a new Scatter-Gather DMA component is presented. This DMA component uses a block of descriptors instead of a single one. This solution allows to increase a speed of data transfer with a fragment memory. The ADP201x1 module with this new Scatter-Gather DMA component provides a speed of input around 1522 Mbytes per second on the 1536 Mbytes memory block.

INTRODUCTION

"Instrumental Systems" company focuses mainly at the developing of modules like ADC, DAC and DSP. It has solutions for PCI as well as for Compact PCI buses. Several years ago we found also a solution for exploitation of a PCI-Express bus. Currently the realization of this PCI-Express bus with a use of Virtex 4 FPGA was accessible at several IP Cores:

- IP Core from Xilinx;
- DesignWare IP Core from Synopsys;
- EZDMA IP Core from PLDA.

For our developments we have chosen the EZDMA IP Core from PLDA [1]. This IP Core includes three layers of standard PCI-Express: Physical, Link, Transaction and additional Application layer. Application layer contains eight direct DMA channels. Reference design contains the Scatter-Gather DMA component.

In the Scatter-Gather DMA Transfer mode, the DMA start address is a pointer to a chained list of page descriptors. Each descriptor contains the address and the size of a data block, and also a pointer to the next descriptor block to enable circular buffers. Such mode is realized in PCI9054, PCI9056 and PEX8311 chips from PLX Technology [2], in EZDMA IP Core from PLDA [1] and IP Core from Northwest Logic [3]. This mode provides the maximum speed of a data exchange. For example, with the PCI9056 chip on the 32 bits PCI bus, 33 MHz speed of 110 Mbytes/s is provided at a limit of 125 Mbytes per second.

During operations with the EZDMA IP Core there are several problems:

1. The DMA channel during data transmission from the bus to the device provides a very slow speed in FIFO mode. It is related to a fact that the controller sends only one request for read operation and waits for its performance. Standard PCI-Express allows to send several requests, but answers can come in any order. It is admissible for memory, but is not admissible for FIFO.
2. The DMA request for reading can be ended with a "Completion Timeout" error. It is very uncommon error, but it happens. The DMA controller only informs on this error, but does not correct it.
3. The DMA controller reads out only one descriptor. It reduces the speed of the data exchange during operations with the fragment memory.

In operating system Windows there are two ways of memory allocation:

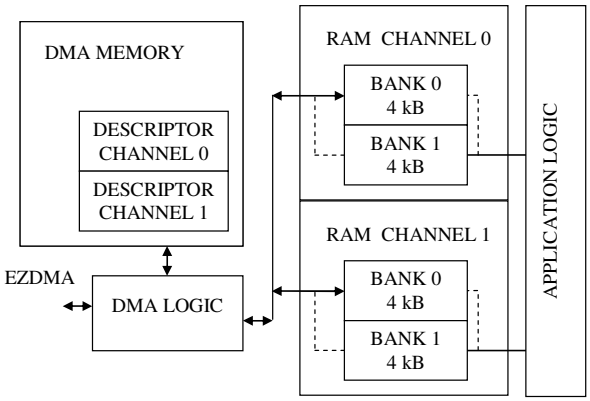
1. Allocation in the system memory.
2. Allocation in the user memory.

In the system memory a continuous memory block of physical addresses is allocated, but it is impossible to allocate a large block of memory. Typical value which could be allocated in this case is only 128 Mb. On the other hand, in the user memory it is possible to allocate the larger block of memory as, for example, 1536 Mb, but this memory will be fragmented on pages of 4 kilobytes. In the Scatter-Gather mode the DMA controller reads out a descriptor for each page. The reading of descriptor is a long operation and it slows down the exchange of the data. For example, on the PCI-Express x4 the data input in the system memories goes on with the speed of 710 Mb/s and in the user memory with only 550 Mb/s.

DESCRIPTION OF NEW CONTROLLER

The new DMA controller has been developed to resolve these problems.

This new controller replaces the "DMA_SG" component from the EZDMA reference design. The controller block diagram is shown below.



On the block diagram the next components are presented:

- EZDMA – connection to EZDMA
- DMA LOGIC – DMA channel operating logic.
- DMA MEMORY – memory of current parameters of the DMA channel.
- RAM CHANNEL0 – data memory of the channel 0
- RAM CHANNEL1 – data memory of the channel 1
- APPLICATION LOGIC – connection to other part of the project

The DMA controller is a two-channels controller, each of which is bidirectional. Thus, there is only one realization of operating logic and related memory, where work parameters of every channel are saved. This allows to reduce the amount of logic resources.

For each channel there are two banks of memory with the 4 kilobytes size for a bank. For the PCI-Express it looks like a memory and for device it looks as FIFO.

During data input from device, the device fills the first bank of memory, while controller transfers data to the PCI-Express from the other bank. Then banks are interchanged their positions.

During data output to device, the controller reads out the data from the computer memory and places it in the first bank of memory. The controller sends several read requests providing the maximum speed of data read out. Answers can come in any order and they are registered in the same memory bank according to their address. Meantime the data are transferred to FIFO from the second bank. After the end of process two banks are interchanged with their positions. In the case of occurrence the "Completion Timeout" error the repeated cycle of filling the memory banks starts.

The time diagram of data reading from the memory looks as follows:

1. There is an initial delay, approximately 1 us.
2. Package reception

The initial delay does not depend on the size of the required block. Also it turns out that reading of 8 or 512 bytes takes an approximately the same time.

In our work separate descriptors have been united in the special block of descriptors. This block has the size of 512 bytes and contains 63 descriptors with additional indexes of the following block of descriptors. This solution allows to increase the speed of work with the fragment memory.

Let's compare: a single descriptor provides input speeds of 710 Mb/s for the system memory and 550 Mb/s for the user memory. The solution with the block of descriptors provides the speed of input around 714 and 709 Mb/s in the similar conditions.

The block of descriptors is protected by the signature and the CRC. The DMA controller reads out the block of descriptors and checks both the signature and the CRC. If there is an error the controller stops its work. This feature considerably facilitates a software debugging.

There is moreover a possibility after finishing of the data exchange to pass to the first descriptor in the block. It allows do not access to the memory for the reading of descriptors.

After the end of the data exchange with a descriptor the controller can generate or not generate an interrupt for the CPU. It allows to work with the fragment memory. If, for example, the chain of descriptors can describe the memory block of 48 Mbytes, which consists of 12288 pages in 4 kilobytes, such interruption for the CPU will be generated only after reception of 48 Mbytes block.

The DMA controller can work also with cyclic buffers. Thus, there can be a problem with the speed of data processing in the CPU. If the processor processes the data more slowly, than the DMA controller transfers them, or if in the course of processing there is a pause, a loss of the data is possible. For solving this problem there is a special mode of data transfer, called "consistent mode". In this mode the interrupt handler on the CPU defines a parity between the accepted and processed blocks and if necessary pauses the data transmission.

The special components cl_test_check and cl_test_generate have been developed for testing the DMA component. During data input from the device, the cl_test_generate generates the special test sequence of data. The program checks this sequence in a real-time mode. Such a test can be executed within several hours. During data output to

the device, the program generates again the test sequence of data. The DMA controller transfers data to the device while the cl_test_check component checks data and remembers number of errors. Also it remembers the data for the first 16 errors. The test sequence includes several kinds of data block: running zero, running one, counter and pseudo-random sequence.

Up to now our company has developed two devices named "AMBPEX8" and "ADP201x1". AMBPEX8 uses the FPGA Virtex4 XC4VFX20 with the PCI-Express x4. ADP201x1 uses the FPGA Virtex5 XC5VLX50T with the PCI-Express x8. Descriptions of both modules can be found on a company Web-site: http://www.insys.ru.

The reached speeds of the data exchange are presented in the table below (Table 1). Measurements are executed on computer Intel Core I7 2.8 GHz with chipset P55. As it follows from the table, the speed of the data transfer with the user memory is the same as with the system memory.

Table 1. Comparison of characteristic speeds of input and output data transfers between different modules and computer.

Module	Input from device		Output to device	
	System 128 MB	User 1536 MB	System 128 MB	User 1536 MB
ADP201x1 Virtex 5 PCIE x8	1535 MB/s	1522 MB/s	1031 MB/s	1016 MB/s
AMBPEX8 Virtex 4 PCIE x4	714 MB/s	709 MB/s	521 MB/s	518 MB/s

CONCLUSIONS

The new Scatter-Gather DMA component have been developed. This DMA component provides the next opportunities:

1. Fast work with the fragment memory.
2. Correction of the "Completion Timeout" error.
3. Fast data transmission to device in the FIFO mode.

REFERENCES

1. "EZDMA IP for Xilinx Hard IP Reference Manual", PLDA

http://plda.com/download/doc/ip/ez_dma_v5lxt/EZDMA_Reference_Manual.pdf

2. "PCI 9056", PLX Technology

http://www.plxtech.com/download/file/668

3. "DMA Back-End Core", Northwest Logic

http://www.nwlogic.com/docs/PCIe_DMA_Back-End_Core.pdf