Organisation Européenne pour la Recherche Nucléaire
European Organization for Nuclear Research

CERN BE-CO-HT

Functional Specifications
Gennum GN4124 Core For FMC Projects

June 2010

Edited by:
Simon Deprez

Checked by:
Javier Serrano

Abstract

This functional specification describes the HDL controller for Gennum the GN4124 chip. This chip is used on the PCI express FMC carrier (www.ohwr.org/projects/fmc-pci-carrier). It provides a Wishbone master for control and status registers access and a DMA controller for high speed data transfers.

Revision History

Version	Date	Notes
0.1	21-05-2010	Initial release
0.2	29-05-2010	Add DMA master specifications
0.3	01-06-2010	Core diagram review + clocks

Contents

List of Figures

List of Tables

1 Introduction

The PCIe FMC carrier[1] will be associated with several FMC[2] mezzanines.

Each FMC mezzanine used with the PCIe FMC carrier will need a specific HDL configuration. The HDL (Hardware description language) code should be generic for reusing the carrier-specific modules with the other FMC mezzanines.

The communication between modules is essentially based on the Wishbone bus. Two separates Wishbones bus are used to performs communications.

- 1 Wishbone master interface that transforms memory reads and memory writes from PCI express bus in Wishbone single reads and single writes. (20 MB/s – 100 MB/s)
- 1 DMA master interface that preforms data transfers from local application to PCI express host. (up to 400 MB/s)

2 GN4124 core for PCIe FMC carrier

This core provides a Wishbone master to access control registers and read status. The Wishbone[3] bus is used for almost all communications between modules. The different soft cores are reusable.

A simple DMA master is used for high speed data transfers of mezzanine card acquisitions. This DMA master is also based on Wishbone and uses a new pipelined mode that was developed to increase throughput. It is expected that it becomes part of a new version of the Wishbone specification.

2.1 GN4124 Local Bus Interface

The GN4124 is a 4-lane PCI express to local bus bridge that is designed to work with an FPGA device to provide a high speed serial access for an application.

The GN412x PCI Express family Reference Manual[4] (52624-0 June 2009) describes the local bus interface at page 40.

The data exchange is based on packets similar to PCI express packets, but the headers are different. It is not possible to send MSI (Message Signaled Interrupts) packets with this bus so the interrupt request signal is

2.2 DMA Interface

The DMA interface is a pipelined Wishbone bus. It preforms data transfers from local application to PCI express host.

The DMA engine works with a linked list so that DMAs can be chained. The first item in the list is loaded by the host on the carrier and contains a pointer to the next one, which is in host memory. The DMA engine will fetch items from host memory and perform the corresponding DMAs until one of the items is recognized as the last one though the contents of the DMAATTRIBR register (see table 1).

Each item in the list is made of the following registers: DMACSTARTR, DMAHSTARTLR, DMAHSTARTHR, DMALENR, DMANEXTLR, DMANEXTHR and DMAATTRIBR. When reading these items

[1] See www.ohwr.org/projects/fmc-pci-carrier.
[2] See VITA FMC standard: www.vita.com/fmc.
[3] See http://opencores.org/downloads/wbspec_b3.pdf.
[4] See http://my.gennum.com/mygennum/view.php/gn4124-gullwing.

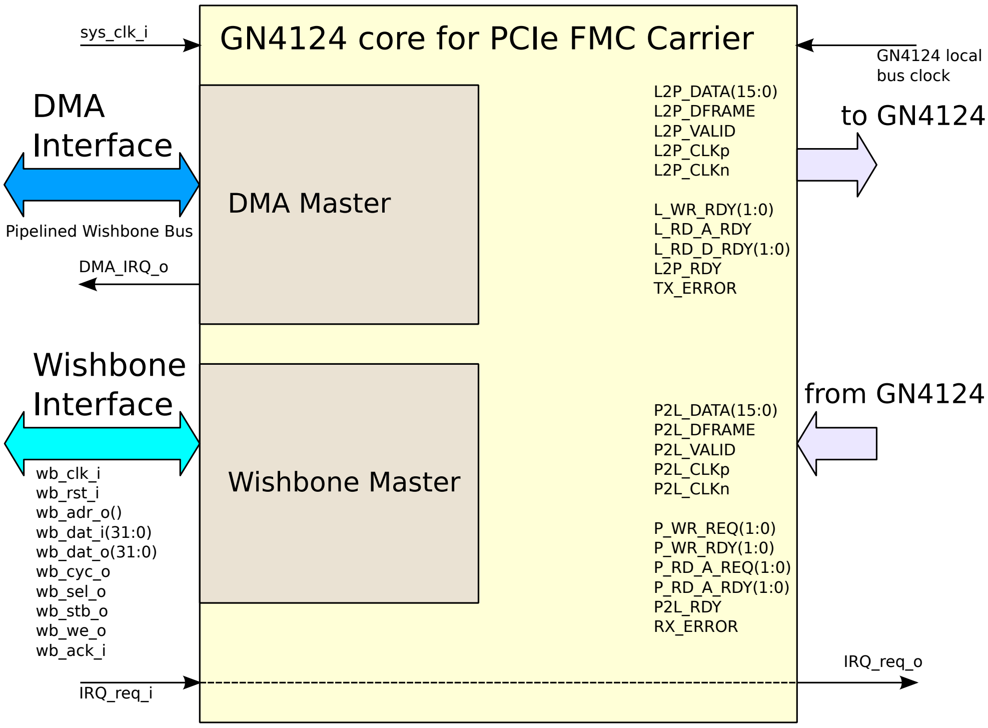

Figure 1: GN4124 core for PCIe FMC carrier

from the host, the DMA engine assumes a little-endian host. Big-endian hosts should shuffle data accordingly so that it is found in the same order as in a little-endian host. In addition, the DMA controller provides global DMA control and status registers.

The end of a chained DMA access generates an interrupt request (`DMA_IRQ_o` output) towards the interrupt controller.

NAME	OFFSET	MODE	RESET	DESCRIPTION
DMACTRLR		R/W		DMA engine control
DMASTATR		RO		DMA engine status
DMACSTARTR		R/W		DMA start address in the carrier
DMAHSTARTLR		R/W		DMA start address (low) in the host
DMAHSTARTHR		R/W		DMA start address (high) in the host
DMALENR		R/W		DMA read length in bytes
DMANEXTLR		R/W		Pointer (low) to next item in list
DMANEXTHR		R/W		Pointer (high) to next item in list
DMAATTRIBR		R/W		DMA endianness and control

Table 1: Register set for the DMA controller block

2.2.1 DMACTRLR

Writing 1 to this register starts a DMA transfer. Writing 2 aborts the ongoing transfer.

2.2.2 DMASTATR

This is a status register for the DMA engine. Possible contents are:

- 0: Idle (before any DMA transfer takes place).
- 1: Done (after successful DMA).
- 2: Busy.
- 3: Error (following a memory access error, either on the host or on the carrier). This also produces an interrupt.
- 4: Aborted (after receiving an abort command in DMACTRLR).

A DMA start command written into the DMACTRLR register takes this status out of Idle, Done, Error or Aborted into the Busy state.

2.2.3 DMACSTARTR

The DMACSTARTR register holds a byte address pointing to a location inside the local memory, at which the DMA access should start. Taking into account that the DDR is a 16-bit device, only even values are allowed in DMACSTARTR.

2.2.4 DMAHSTARTLR and DMAHSTARTHR

Registers DMAHSTARTLR and DMAHSTARTHR select the low and high parts of the 64-bit start address for the DMA access in the host.

2.2.5 DMALENR

Register DMALENR selects the length of the reading in bytes, i.e. twice the number of samples to be read by the host. This means DMALENR has to hold an even number.

2.2.6 DMANEXTLR and DMANEXTHR

These two registers contain the low and high parts of the 64-bit address of the next item in the linked list, in host memory.

2.2.7 DMAATTRIBR

This register contains several control features for the DMA engine:

- Bits [31..16] are reserved.
- Bits [15..8] are used to select how many bytes the DDR controller jumps after every RAM access.
- Bits [7..2] are reserved.
- Bit 1 is set to ’0’ for little-endian accesses and ’1’ for big-endian. This affects the way in which 16-bit samples can be stored in a 32-bit long word.
- Bit 0 is set to ’1’ to signal this is the last item in the linked list, ’0’ otherwise.

The end of a chained DMA access generates an interrupt request towards the interrupt controller.

2.3 Wishbone interface

The Wishbone master (see table 2) transforms a PCIe memory write into a 32 bits Wishbone write and a PCIe memory read into a 32 bits Wishbone read. Only single word reads and writes are supported.

The Wishbone bus is clocked by the `wb_clk_i` input.

WISHBONE DATASHEET for the 32-bit MASTER		
Description	Specification	
General description	Wishbone master controlled by PCI express	
Supported cycles	MASTER, READ/WRITE	
Data port, size: Data port, maximum operand size Clock frequency constraints:	32-bit 32-bit 100 MHz	
Supported signal list and cross reference to equivalent WISHBONE signals	Signal Name	WISHBONE Equiv.
	wb_clk_i	CLK_I
	wb_rst_i	RST_I
	wb_cyc_o	CYC_O
	wb_stb_o	STB_O
	wb_adr_o(10..0)	ADR_O()
	wb_dat_i(31..0)	DAT_I()
	wb_dat_o(31..0)	DAT_O()
	wb_we_o	WE_O
	wb_ack_i	ACK_I

Table 2: Wishbone master datasheet

2.4 Interrupts

When the GN4124 master gets a one-tick-long (`sys_clk_i`) positive pulse on the `IRQ_req_i` input, a one-tick-long (GN4124 local bus clock) positive pulse is sent on the `IRQ_req_o` output to the GN4124 chip. The interrupt signal is directly wired to GN4124 chip GPIO. When configured, the chip sends an MSI packet (Message Signaled Interrupts) to the host.

At the end of a chained DMA acces, the DMA master sends a one-tick-long (`sys_clk_i`) interrupt signal on the `DMA_IRQ_o` output toward the interrupt controller.