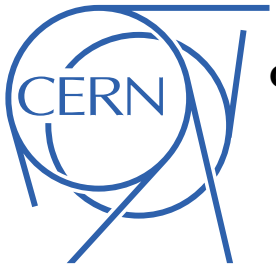

ORGANISATION EUROPÉENNE POUR LA RECHERCHE NUCLÉAIRE
EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

CERN BE-CO-HT

Functional Specifications
Gennum GN4124 Core For FMC Projects

July 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
0.4	28-06-2010	Add byte swapping for DMA interface
0.5	06-07-2010	Corrections after review of 25-06-10

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 (Hardware description language) configuration. The HDL 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.

- CSR (Control and status register) Interface
  - Wishbone[3] Master
  - 32 bit
  - Singles reads and writes
  - 40 MB/s (single reads) – 200 MB/s (single writes)
- DMA Interface
  - Pipelined Wishbone Master
  - 32 bit
  - Local to PCI Express data transfer
  - PCI Express to local data transfer
  - Byte swapping
  - 600 MB/s

2 GN4124 core for PCIe FMC carrier

This core provides a Wishbone master to access control registers and read status. The Wishbone 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.

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 wired to a GPIO input of the GN4124 chip.

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

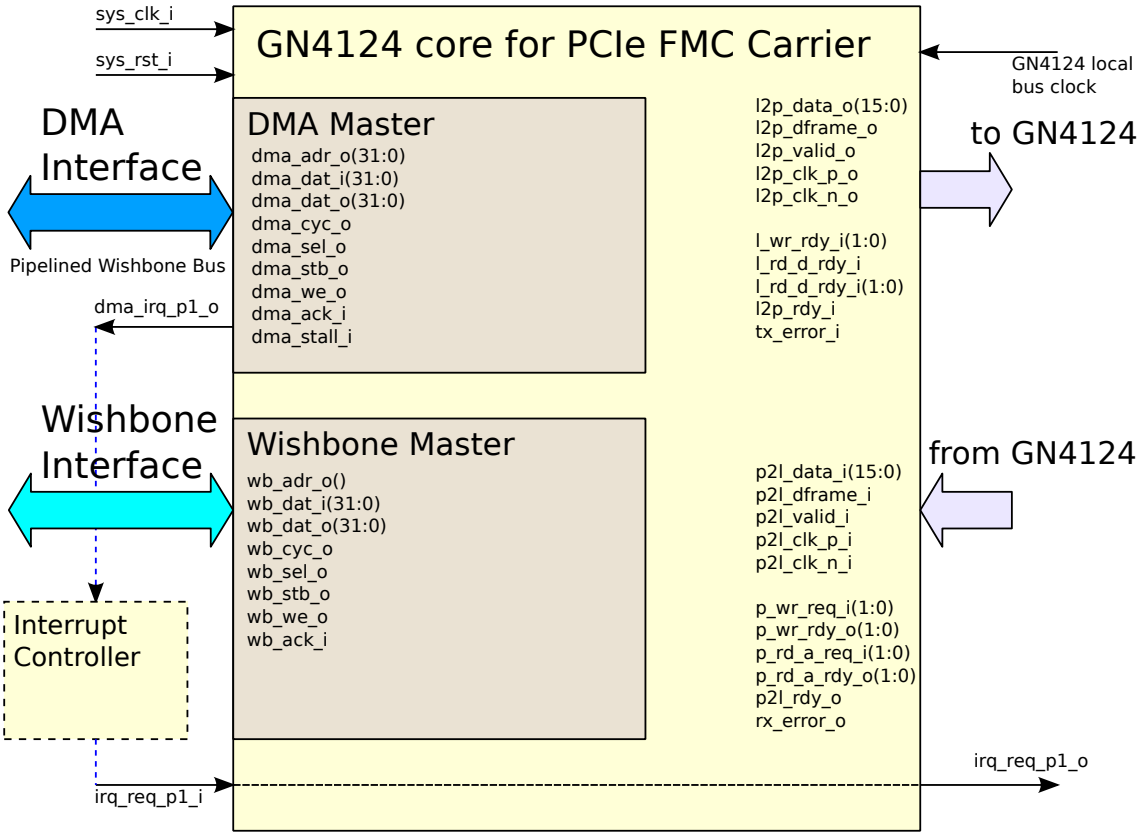

Figure 1: GN4124 core for PCIe FMC carrier

2.2 DMA Interface

The DMA interface is a 32 bit 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 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_p1_o` output) towards the interrupt controller.

NAME	OFFSET	MODE	RESET	DESCRIPTION
DMACTRLR	0x00	R/W	0x00000000	DMA engine control
DMASTATR	0x04	RO	0x00000000	DMA engine status
DMACSTARTR	0x08	R/W	0x00000000	DMA start address in the carrier
DMAHSTARTLR	0x0C	R/W	0x00000000	DMA start address (low) in the PCIe host
DMAHSTARTHR	0x10	R/W	0x00000000	DMA start address (high) in the PCIe host
DMALENR	0x14	R/W	0x00000000	DMA read length in bytes
DMANEXTLR	0x18	R/W	0x00000000	Pointer (low) to next item in list
DMANEXTHR	0x1C	R/W	0x00000000	Pointer (high) to next item in list
DMAATTRIBR	0x20	R/W	0x00000000	DMA chain control

Table 1: Register set for the DMA controller block

2.2.1 DMACTRLR

- Bits [31..4] are reserved.
- Bits [3..2] controls byte swapping of the DMA interface(see table 2). It works in the both DMA transfer directions:
  - "00" To swapping.
  - "X1" Swap the two bytes in the words.
  - "1X" Swap the two words.
- Bit 1 is set to '1' to abort the ongoing transfer.
- Bit 0 is set to '1' to start a DMA transfer, '0' otherwise.

Double word on PCI Express	Control bits of the swapping	Double word on Wishbone bus
A1 B2 C3 D4	"00"	A1 B2 C3 D4
	"01"	B2 A1 D4 C3
	"10"	C3 D4 A1 B2
	"11"	D4 C3 B2 A1

Table 2: Byte swapping

2.2.2 DMASTATR

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

- 0: Idle (before any DMA transfer takes place and after reset).
- 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.

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 PCI Express host.

2.2.5 DMALENR

Register DMALENR selects the length of the data transfer in bytes.

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 PCI Express host memory.

2.2.7 DMAATTRIBR

This register contains several control features for the DMA engine:

- Bits [31..2] are reserved.
- Bit 1: Transfer direction. This bit is set to '0' for local to PCI Express host data transfer an '1' for PCI Express host to local data transfer.
- 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 3) transforms a PCIe memory write into a 32 bits Wishbone write and a PCIe memory read into a 32 bits Wishbone read. Only single reads and writes are supported.

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

2.4 Interrupts

When the GN4124 master gets a one-tick-long (`sys_clk_i`) positive pulse on the `irq_req_p1_i` input, a one-tick-long (GN4124 local bus clock) positive pulse is sent on the `irq_req_p1_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_p1_o` output toward the interrupt controller (see figure 1). So the interrupt controller knows that the DMA master is the source of the interrupt.

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 sys_clk_i sys_rst_i wb_cyc_o wb_stb_o wb_adr_o(10..0) wb_dat_i(31..0) wb_dat_o(31..0) wb_we_o wb_ack_i	WISHBONE Equiv. CLK_I RST_I CYC_O STB_O ADR_O() DAT_I() DAT_O() WE_O ACK_I

Table 3: Wishbone master datasheet