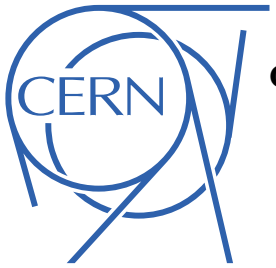

ORGANISATION EUROPÉENNE POUR LA RECHERCHE NUCLÉAIRE
EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

CERN BE-CO-HT

Gennum GN4124 to Wishbone bridge

April 2010

Edited by:

Simon DEPREZ

Checked by:

Abstract

Gennum is providing an IP core that can be used for hardware design with the chips GN412x. This documentation describes how to add a simple wishbone master to the IP and use it for a specific application.

Revision History

Version	Date	Notes
0.1	10-03-2010	Created
0.2	01-03-2010	First draft
0.2	14-04-2010	Add Wishbone master description
0.4	27-04-2010	Add Wishbone slave description
0.5	04-05-2010	More about Wishbone master

Contents

Introduction **2**

1 Gennum IP core **2**
1.1 Gennum documentation . . . 2
1.2 Gennum ISE project . . . 3

2 Wishbone master **3**
2.1 Application Attachment layer modifications . . . 3
2.1.1 Arbiter . . . 3
2.1.2 Packet decoder . . . 4
2.2 Wishbone master module . . . 4
2.2.1 Wishbone master limitations . . . 5
2.2.2 Write request . . . 5
2.2.3 Read request . . . 5
2.3 Wishbone slave . . . 7

3 DMA controller **8**
3.1 Gennum Pinto project . . . 8

4 Gennum Simulation Testbench **8**
4.1 Some issues . . . 8
4.1.1 Path error . . . 8
4.1.2 Error with address in BFM scripting using C . . . 8
4.2 Makefile . . . 8
4.3 BFM script . . . 9
4.4 Simulation . . . 9

Summary **9**

List of Figures

1 Gennum core with Wishbone master . . . 4
2 Write request . . . 5
3 Read request . . . 6

Introduction

This project aims to provide a Wishbone master generic interface for FMC[1] projects controlled by a PCI express access.

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

This file explains how to start with the GN4124 IP core. It describes the testbench environment provided by gennum (Beta version) to simulate the IP.

The Wishbone master is wroten as simple as possible

1 Gennum IP core

1.1 Gennum documentation

Gennum documentation files can be obtained from the following url:

`http://my.gennum.com/mygennum/view.php/gn4124-gullwing`

A registration is required. The registration must be validated.

The reader is required to be acquainted with the following documentation:

- GN412x PCI Express family Reference Manual (52624-0):
  - Overview (p 11–12),
  - Timing diagrams (p 41, 52–55),
  - DMA core diagram (p 84);
- GN412x FPGA IP Hardware Design Guide (51860-2):
  - Overview (p 7),
  - DMA sequencer limitations (p 9),
  - DMA core diagram (p 22),
  - Project register map (p 50);
- GN412x RDK Software Design Guide (51859-1): Describes the software furnished with the development kit;
- GN412x Simulation Test Bench User Guide (53716): Tutorial for the simulation environment;
- GN412x BFM Reference Manual: Describes the Bus Functional Model used for the Testbench.

The testbench zip archive (`GN412x_TestBench_ModelSim(Beta_0.2.0).zip`) contains the last two documents.

[1] See http://www.vita.com/fmc.html.
[2] See http://www.ohwr.org/projects/fmc-pci-carrier.

1.2 Gennum ISE project

The Xilinx ISE project name is Lambo. The project must be migrated to a new version of Xilinx ISE. The project files can be obtained from the following url (a registration is required):

`http://my.gennum.com/mygennum/view.php/gn4124-gullwing`.

Select `GN412x FPGA IP Xilinx Version 2009-05-26`.

After synthesis, the bitstream can be downloaded on the Gennum development kit with the `gendiag` program.

The `gendiag` program can be downloaded from the Gennum web site.

Select `GN412x RDK SW 1.3 Windows`. A linux version is existing.

Edit the file `C:\Program Files\GN412x_RDK\GenDiag\gendiag.ini`. The last lines contains the path of the Xilinx bitstream file :

```
<path>\lambo.bit
```

Run `gendiag`.

2 Wishbone master

This project is based on the Gennum FlexDMA IP core.

Gennum IP provides a CSR interface similar to Wishbone but not totally compatible. There is no signal to differentiate read cycle of a wait state. There is no acknowledge that permit to slave to indicates normal termination of a bus cycle.

The original target controller modification is not easy because the the DMA sequencer is driven with this interface.

2.1 Application Attachment layer modifications

The Application attachment layer is the main part of the Gennum core. It contains DMA masters and the target controller. A Wishbone master interface is added. See Figure 1.

The `./fpga_project/lambo/design/rtl/` folder in the testbench environment contains all the Gennum IP files.

That transforms PCIe write into Wishbone write and a PCIe read into a Wishbone 'delayed' read.

2.1.1 Arbiter

Add an entry for `wbmaster` module. The file `arb.v` is modified to accept packets of the `wbmaster` module. The added signals are :

- `wbm_arb_data` : input;
- `wbm_arb_valid` : input;
- `wbm_arb_dframe` : input;
- `wbm_arb_req` : input;

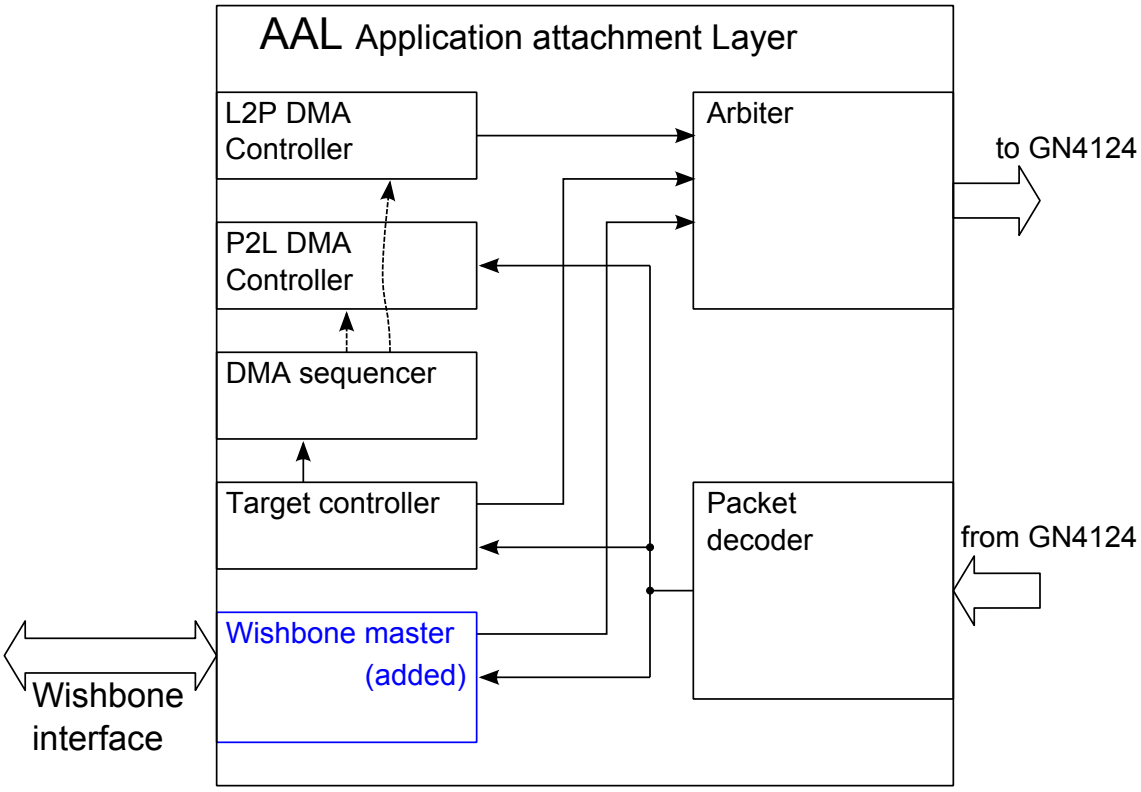

Figure 1: Gennum core with Wishbone master

- `arb_wbm_gnt` : output.

2.1.2 Packet decoder

The packet decoder (`packet_decoder.v`) is not modified.

2.2 Wishbone master module

A Wishbone master based on Gennum Target module (`tar.v`) is added. It supports single read and single write of 32 bit data. The Wishbone bus clock frequency is 100 MHz. The Gennum FlexDMA local clock is used.

Addresses between 400h and 7FCh are routed to the wishbone master (see `dma.v`). On the Wishbone bus the address range is 100h – 1FFh for 32 bit registers. This can be redefined in the file `dma.v`.

Wishbone signals are added to the application attachment layer entity (`dma.v`) :

- `wb_adr_o` : output;
- `wb_dat_i` : input;
- `wb_dat_o` : output;
- `wb_cyc_o` : output;
- `wb_sel_o` : output;

- `wb_stb_o` : output;
- `wb_we_o` : output;
- `wb_ack_i` : input.

This signals are also present in `wbmaster` module and replace original target interface(see `tar.v`).

2.2.1 Wishbone master limitations

Data transfers are limited to single writes and single read. During a wishbone cycle, new requests are ignored.

2.2.2 Write request

When a single write packet is addressed to the wishbone master, the module drive the wishbone bus signals (see figure 2). Then, the module wait for an acknowledge from a wishbone slave to terminate the cycle. During the wishbone cycle, the other PCIe requests are ignored. Wishbone cycles have a timeout of 7 clock cycles.

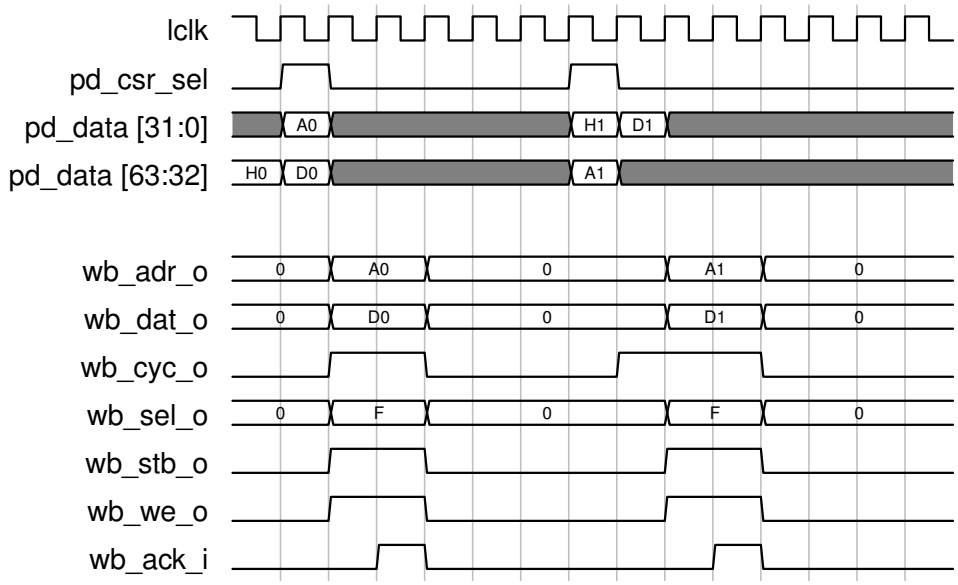

Figure 2: Write request

2.2.3 Read request

The process is the same than for a write request but the write enable signal isn't put high. When the wishbone cycle is terminated, a simple read completion packet is sent to the arbiter (see figure 2).

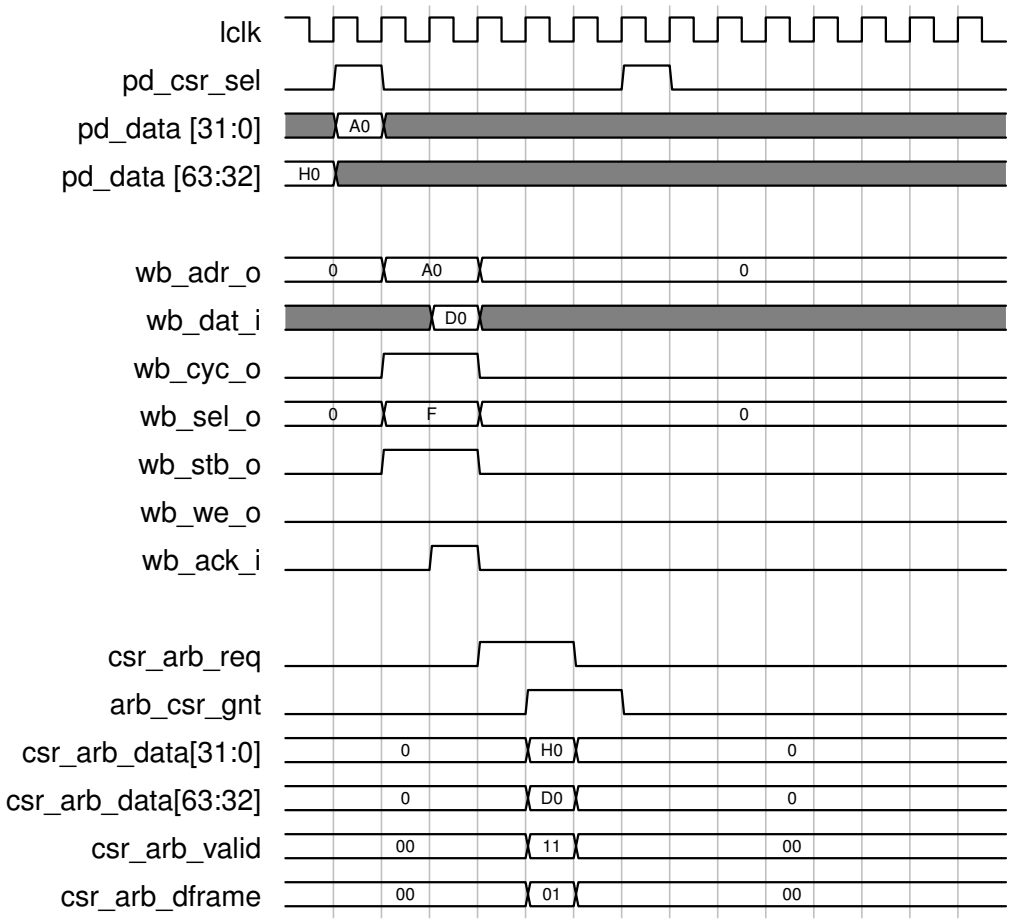

Figure 3: Read request

2.3 Wishbone slave

A Wishbone slave module (`wb_debug.vhd`) was generated with the CERN`wbgen2`[3] script with 3 registers of 8 bits for tests. It was connected to debug leds and buttons of the gennum kit and to the wishbone master.

The input description file for `wbgen2` is `wb_debug.wb` :

```
-- here comes our peripheral definition
peripheral {
-- short (human-readable) name for the peripheral.
  name = "Gennum card debug Wishbone slave core";
-- a longer description, if you want
  description = "An 8-bit output port";
-- name of the target VHDL entity to be generated
  hdl_entity = "wb_gpio_debug";
-- prefix for all the generated ports belonging to our peripheral
  prefix = "gpio";

-- Ouput register
  reg {
    name = "LED port";
    description = "Output port";
    prefix = "led";
    field {
      name = "Port output value";
      description = "Reflects the state of LEDs.";
      type = SLV;
      size = 8;
      access_bus = READ_WRITE;
      access_dev = READ_ONLY;
    };
  };

  reg {
    name = "DEBUG port";
    description = "Input port";
    prefix = "debug";
    field {
      name = "Port input value";
      description = "Reflects the state of the DEBUG switch.";
      type = SLV;
      size = 8;
      access_bus = READ_ONLY;
      access_dev = WRITE_ONLY;
    };
  };

  reg {
    name = "Test register";
    description = "Test register";
    prefix = "test";
    field {
      name = "Test register";
      description = "Test read and write register";
      type = SLV;
```

[3] See Wishbone slave generator.

```
      size = 8;
      access_bus = READ_WRITE;
      access_dev = READ_ONLY;
    };
  };
};
```

3 DMA controller

3.1 Gennum Pinto project

This project contains a Xilinx DDR2 ram controller used for DMA transfer.

The project must be migrated to a new version of Xilinx ISE. If the file `C:\pinto\pinto.xcf` cannot be found by the synthesizer, change the path by click-right on `Synthesize - XST` and choose `Process properties`. Set the good path for the Synthesis Constraints File.

4 Gennum Simulation Testbench

4.1 Some issues

4.1.1 Path error

The file `modelsim.ini` contain an absolute path reference to the file `vlog.opt` at line 256.

Change the line to :

```
OptionFile = vlog.opt
```

4.1.2 Error with address in BFM scripting using C

The `long long int` variables make errors in the format string of `printf` commands when the MinGW (Windows) compiler is used. In the address placeholders of the format string, the length parameter `ll` must be replaced by `I64`.

Replace `%016llX` by `%016I64X` in the file `./Test_Builder/lib/model.c` at lines 64, 73, 288 and 298.

4.2 Makefile

Rules are added for both Wishbone master `wbmaster.v` (line 51-53) and wishbone slave `wb_gpio.vhd` (line 49-51).

Dependencies are added in the top module (`lambo.v`, line 45) rule in the Makefile.

4.3 BFM script

Gennum provides a Testbench environment that emulate the local bus between the GN4124 chip and the FPGA. The Bus Functional Model is used to simulate PCI access for de FPGA system.

The file `wbmaster_test.c` contains a script with read and write requests for the wishbone bus.

4.4 Simulation

Read the GN412x Simulation Test Bench User Guide (53716).

The file `./Test_builder/wbmaster_test.c` is a simple BFM script used for the testbench with the wishbone master.

Open a terminal and use the command :

```
test wbmaster_test
```

In Modelsim, run the simulation :

```
do lambo.do
```

Summary

The wishbone master is simple and allows single read and single write cycles of a 32 bit data.