



Development of Dual Port Block RAM for ngFEC module

DESY Summer Project Report 2013

By

Jaiky Kumar

COMSATS Institute of Information Technology, Pakistan

Supervised by

Alan Campbell, Ozgur Sahin

Abstract

For the upgrade process of HCAL, we are replacing the VME with μ TCA. There is a setup of μ TCA at DESY in which we have a GLIB card and this card is use for the control, triggering and data acquisition of the front end electronics components on the detectors. Inside the GLIB card we have a Virtex 6 FPGA which is the most powerful FPGA in market now days and it controls all the functionality of GLIB card. So In this project I have developed a dual port block RAM module which will be implemented inside the Virtex 6 FPGA during the upgrade of HCAL.

Contents

1. Introduction	4
1.1 CMS	4
1.2 μ TCA.....	4
1.2.1 μ TCA @ HCAL.....	5
1.2.2 ngFEC.....	5
1.3 GLIB	5
1.3.1 Architecture of GLIB.....	6
1.3.2 The GLIB Board.....	6
1.4 FPGA.....	7
1.4.1 Classification of FPGAs	8
1.4.2 FPGA Programming Languages	9
1.5 Types of RAM	9
1.5.1 SRAM (static).....	9
1.5.2 DRAM (dynamic)	10
1.5.3 Block RAM.....	10
2 My work towards GLIB.....	10
2.1 Goal of the project	10
2.2 Design.....	11
2.2.1 Dual Port Block RAM	11
2.2.2 Controller for Dual Port Block RAM	13
2.2.3 Wrapper	14
3 Simulation Results.....	15
4 Conclusion.....	16
5 Acknowledgements.....	16
6 References	17

1. Introduction

1.1 CMS

CMS (Compact Muon Solenoid) is one of the biggest experiments of CERN and its main purpose behind this is to explore physics at TeV scale.

CMS is a detector which has built for more than one purpose; it is developed to measure the energy and momentum of photons, electrons, muons and other particles after the collisions and it is also used for monitoring and alignment of data to the detector.

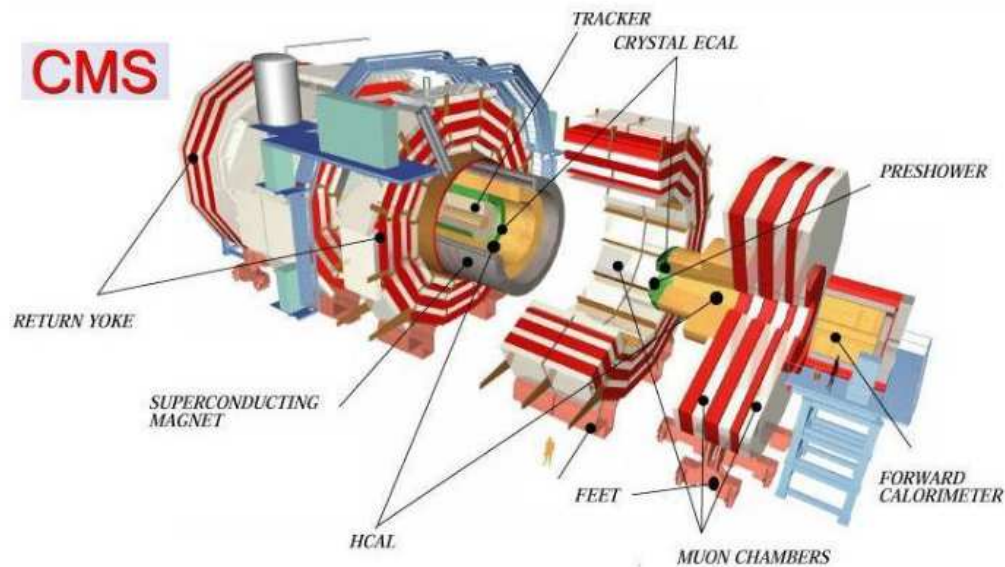


Fig 1: CMS detector

1.2 μ TCA

In high energy physics experiments, the control system built of distributed systems and communication between these systems is a challenge. Data Acquisition requires large amount of data with very low latency. μ TCA is a micro version of Telecommunications Computing Architecture and it started with advanced Telecommunications Computing Architecture which offers various types of data communication channels with many features like high bandwidth, redundancy, high reliability and very flexible backplane connectivity.

1.2.1 μ TCA @ HCAL

μ TCA will replace VME because it has the capacity of more data than VME. It will reduce the complexity because in VME we have many cards and in μ TCA we have reduced the number of card. It also have the better cooling process and contains the star topology.

1.2.2 ngFEC

ngFEC stand for next generation front end controller and its purpose is to control all the front end electronics on the detector during the operation. In the next generation FEC crate, the μ TCA standard will be used.

1.3 GLIB

The Gigabit Link Interface Board (GLIB) is an FPGA based evaluation platform and in high energy physics experiments it is an easy entry point for users of high speed optical links. It can be used for Control, triggering and data acquisition from front end electronics on detectors.

GLIB major hardware component is an FPGA based framework which will be used inside a μ TCA crate in the current project, FPGA controls mostly all the functions of GLIB card.

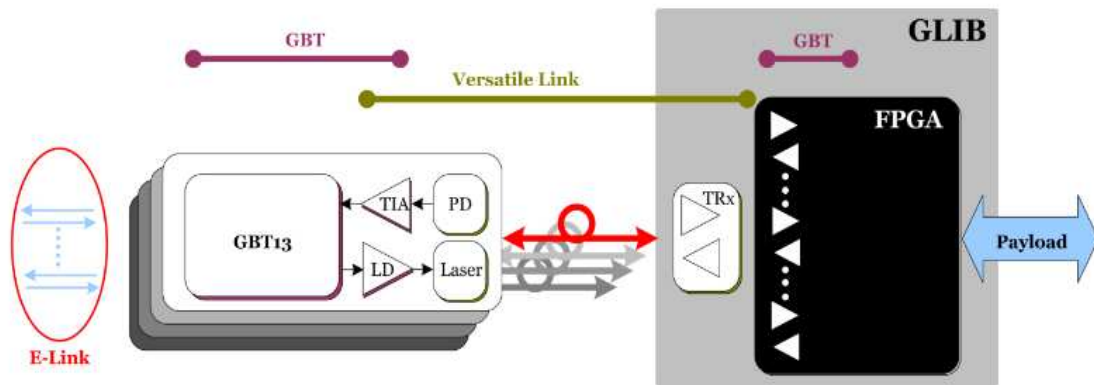


Fig 2: The GLIB board in a GBT-Versatile Link system

Each GLIB card can process the data to/from four SFP modules at 6.5 Gbps. This performance meets the specifications of the GBT/Versatile Link project with data rate of 4.8 Gbps.

1.3.1 Architecture of GLIB

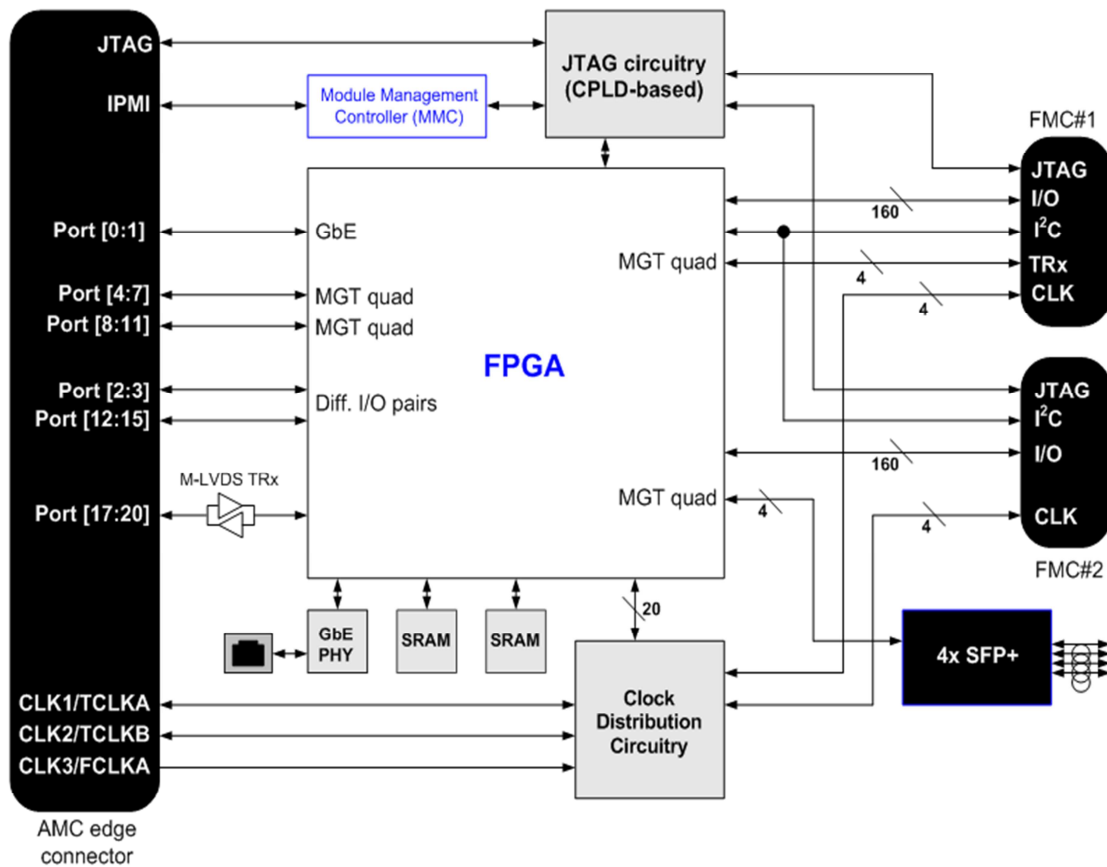


Fig 3: The GLIB Architecture

1.3.2 The GLIB Board

The main component of GLIB is FPGA which is controlling all the function done by GLIB card like sending the acknowledgment to the network when to read the data from Block RAM to avoid misreading of data and also send some information from network/detector to Block RAM.

There are two Static RAMs on GLIB card which can also be used for storing some information. The special feature of this card is that it contains the FPGA Virtex 6 which is most powerful indeed in the market and one other important feature of this FPGA is that it contains a block RAM module .There is a microcontroller which is controlling some functionality of GLIB card as well.

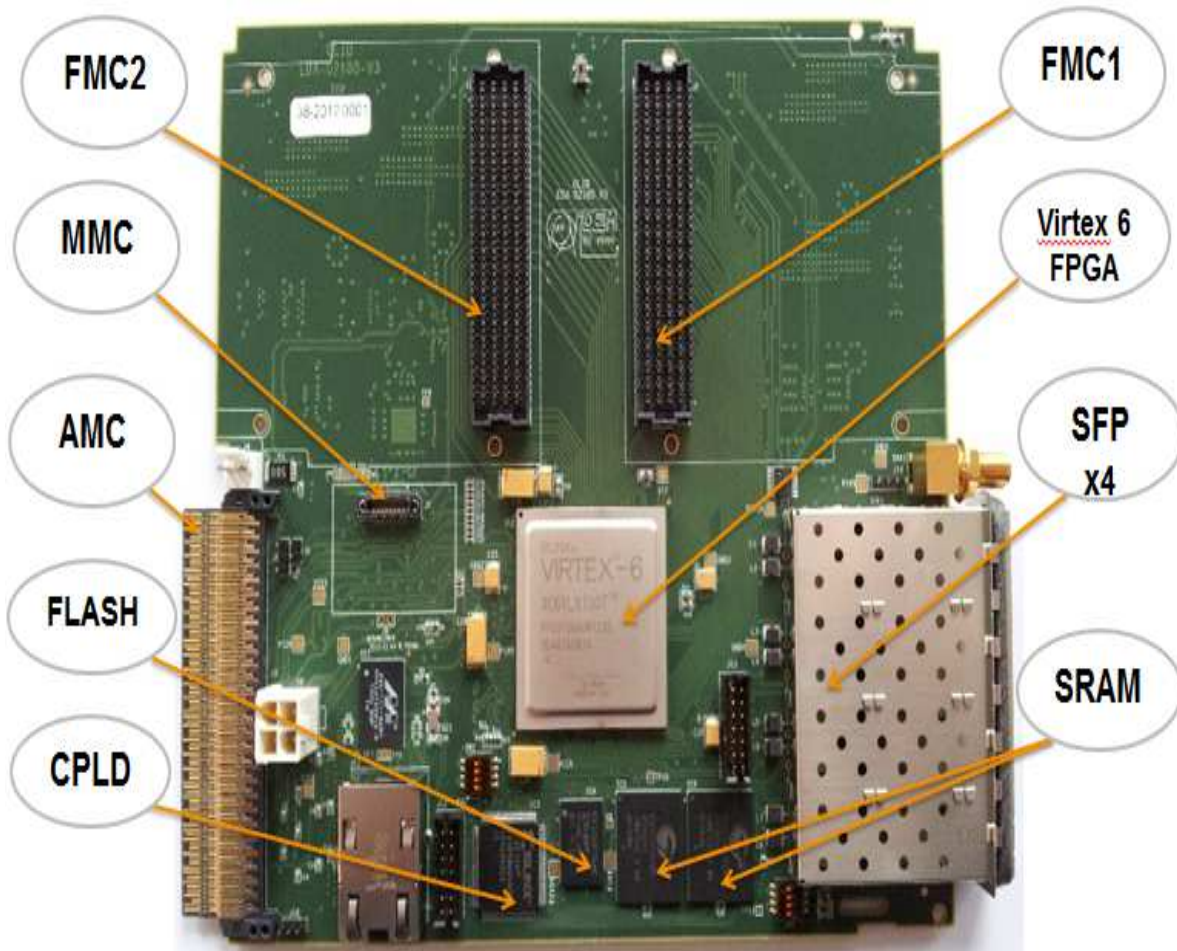


Fig 4: The GLIB Board

1.4 FPGA

FPGA stand for Field Programmable Gate Array which can be envisaged as a grid of ICs with programmable interconnections as shown in figure 5. FPGA consists of three basic components:

- Logic Blocks (like ICs)
- I/O Units
- Interconnections

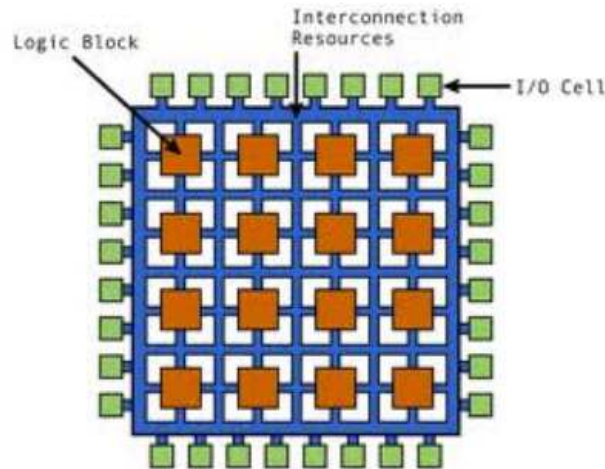


Fig 5: Schematic diagram showing the arrangement inside an FPGA

It can be programmed by the user that is why it is called 'Field Programmable Gate Array'. This feature gives more importance to FPGA over ASIC (Application Specific IC), which are specially manufactured for a particular task. User can configure the function which they want in logical blocks, choose the desired I/O Units to be used and the way both these components are be interconnected. Parallel Execution distinguishes FPGAs from microcontrollers.

1.4.1 Classification of FPGAs

FPGAs differ in their architecture and functionality based on the companies they are manufactured from.

Currently the major FPGA vendors are:

1. Xilinx - most popular (In this project we are using Virtex-6)
2. Altera
3. Actel

These differ in:

- In manner of speed, Xilinx FPGAs are considered the speediest FPGA now days.
- Implementing programmability (VHDL/Verilog).

1.4.2 FPGA Programming Languages

FPGA are programmed using Hardware Descriptive Languages (HDLs). There are two HDL languages, namely:

1. Very High Speed Integrated Circuit Hardware Description Language (VHDL)
2. Verilog Hardware Description Language.

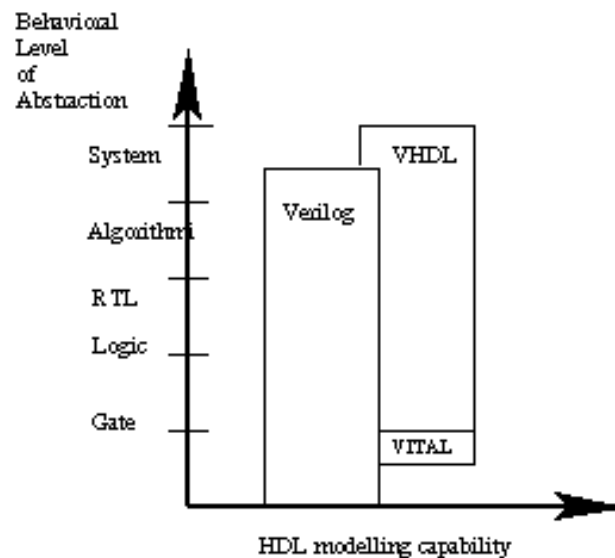


Fig 6: Comparison between VHDL and Verilog HDL

VHDL is based on Pascal and is very old. Verilog is relatively recent and is based on C language. VHDL is not case sensitive whereas Verilog is case sensitive so this make VHDL more friendly. Verilog has simple data types whereas VHDL allows users to make more complex data types. We have used VHDL in this project to program the Virtex 6 FPGA.

1.5 Types of RAM

1.5.1 SRAM (static)

- Formed from internal latches – a typical SRAM is made up of 6 MOSFETS.
- Latch will store information as long as power supplied.

- Very fast as compared to DRAM.
- Expensive.
- Consume less power
- External chip on GLIB

1.5.2 DRAM (dynamic)

- Storage of charge on a capacitor gated by a transistor.
- High packing density, large cheap memory,
- Very slow as compared to SRAM.
- Cheap + economies of scale = very cheap.
- Consume more power

1.5.3 Block RAM

- Made up of SRAMs
- Dual port.
- Very Fast.
- Generated from Core Generator as a real RAM.
- BRAM is a special feature on Virtex 6 FPGA of GLIB

In this project we are using the Virtex-6 device with 36 Kb block RAM size. This 36 Kb blocks are cascade able which will implements a deep and wide memory.

2 My work towards GLIB

2.1 Goal of the project

The main objective of this project was to developed a Dual Port Block RAM, buffering mechanism using VHDL language for GLIB Project which allows to use the internal BRAM block resources as dual ported memory inside the virtex 6 FPGA, that is capable of storing/retrieving the information to/from one port of BRAM to/from Network via 1 Gb Ethernet connection and also do the same operation on another port of RAM to/from front end components on detectors via signal of multiple I2C buses at 100 kHz and LHC clock of 40 kHz. To initialize the front end electronics components first we will send the data from network to dual Port block, the data will be stored there and then the data is first serialized by I2C controller and I2C protocol is implemented on it after that data will be digitized and will be read by the front end electronics components to initialize them. If the network wants to read the data which we used for the initialization of front end electronics components then it

can read from the dual port block memory because each of the buses can read and write the data.

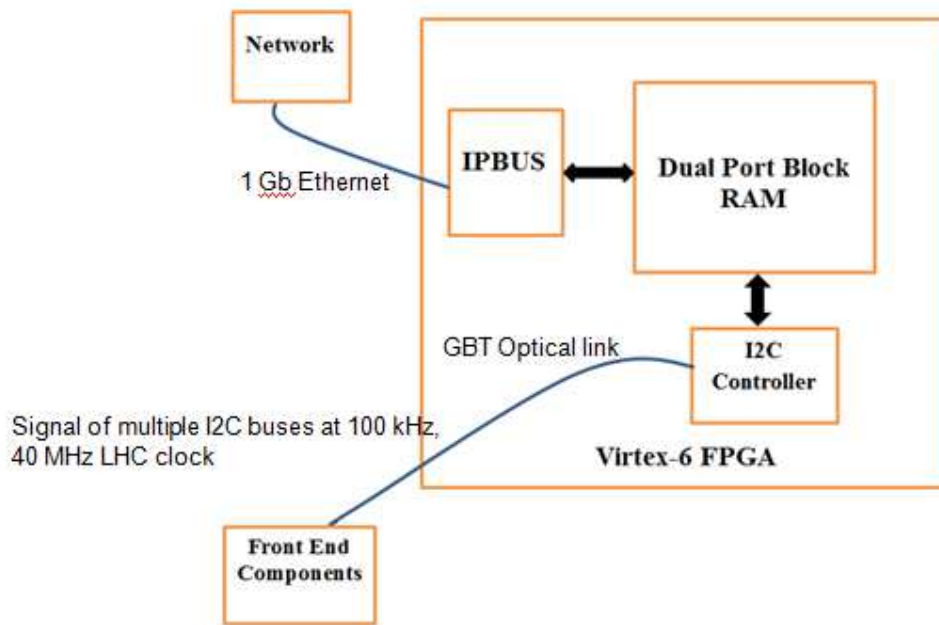


Fig 7: Block Diagram of BRAM interfacing with virtex-6 FPGA

2.2 Design

2.2.1 Dual Port Block RAM

The true dual port BRAM consists of 36 Kb size and it contains two independent ports, A and B. Data can be written to either port or both of the BRAM and can also read from either/both port of the BRAM. Read and Write operations are synchronous. Each port has its own data in, address, write enable, clock and data out.

We have used this RAM in this project because it contains dual port for simultaneously reading and writing at different address (a write operation from GLIB Card and a read operation from Frontend module at the same time) and also it is too fast and also not too much expensive as SRAM.

Write Mode

The data available on the output port is dependent on the the write mode and there are three write modes namely: **WRITE_FIRST**, **READ_FIRST**, and **NO_CHANGE**. We have used **NO_CHANGE** mode for our module, in this the data output during the write operation remains the last read data.

The RAM module writes the data only when the write enable pin is high and there should be an appropriate address for writing. If the write enable pin is low then it reads the data from the address which the user gives to the module.

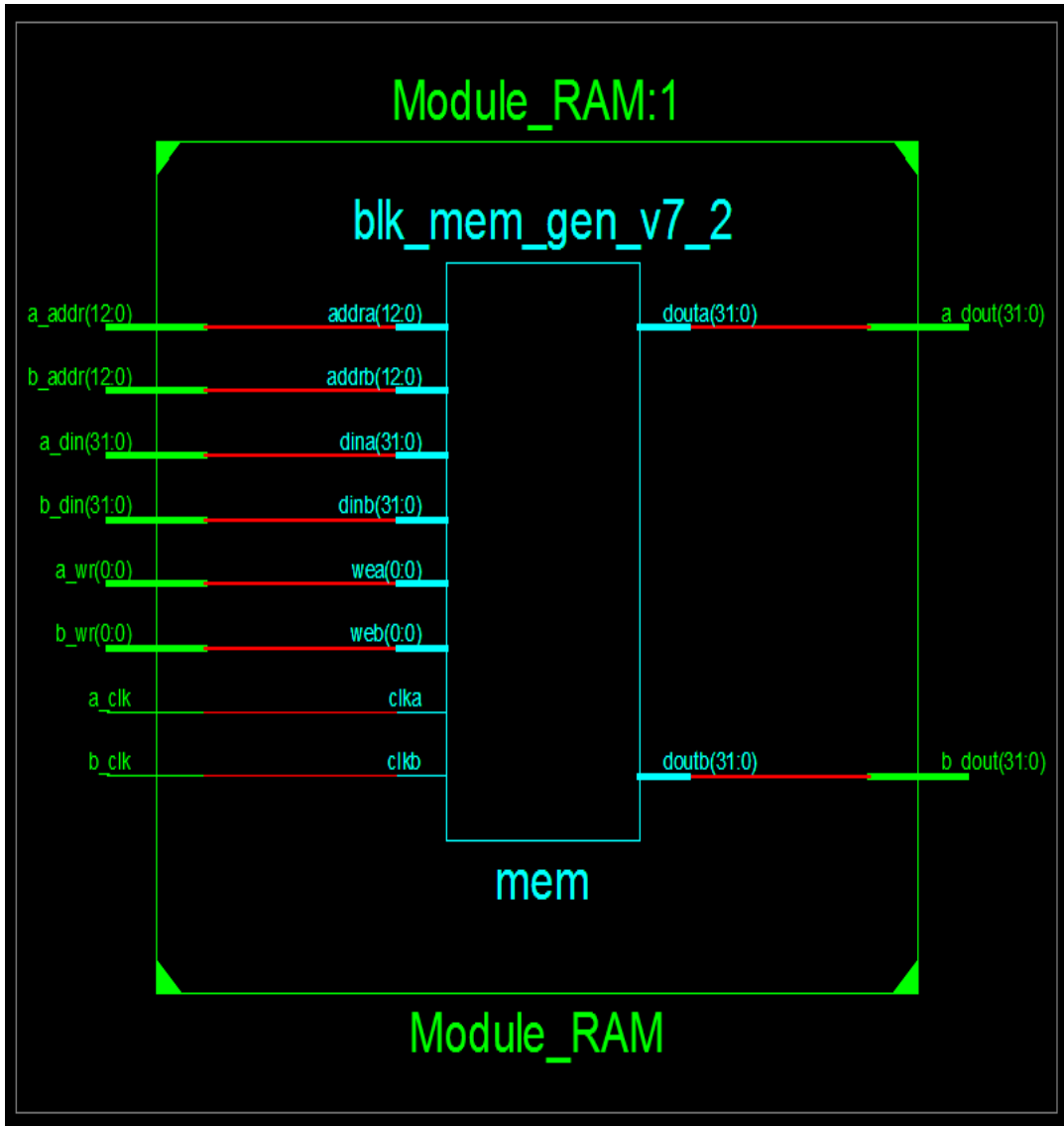


Fig 8: RTL Schematic of Block dual Port RAM

Port Function	Description
[a b]_addr[13]	Address for Write/Read Cycle
[a b]_din[32]	Input data for Write Cycle
[a b]_wr	Write Enable Pin for write cycle
[a b]_clk	Clock
[a b]_dout	Output Data from BRAM

Table 1: True Dual-Port Block RAM Functions and Description

2.2.2 Controller for Dual Port Block RAM

The controller for dual port Block RAM manages the flow of data going to and from the block RAM. Controller is made up of three states: reset, write and read. The reset is the state when all the inputs and outputs will be set to zero means to reset the RAM. This state can be achieved by enable the reset pin as high.

The second state is write state and it will send signal the RAM module whether to write or read, depending upon the write enable pin. If the write enable pin is high then it will send the write enable signal as high, address and data to be written on RAM module for the writing process.

The third state is read state, if the write enable pin is low then controller will send the write enable pin as low and it will also send the address to read the data from RAM module.

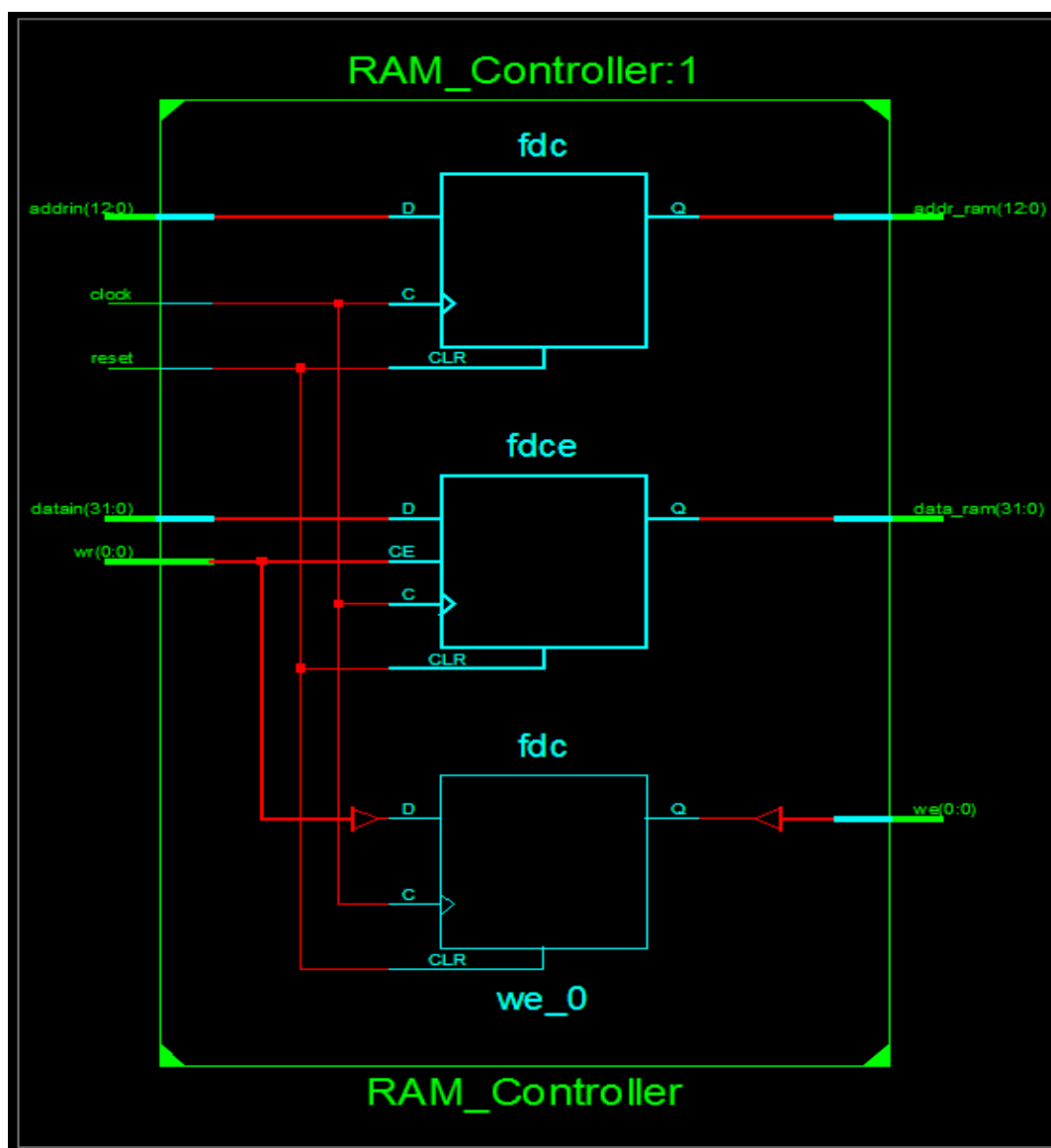


Fig 9: RTL Schematic for Controller of Block dual Port RAM

Port Function	Description
Addrin[13]	Address for Write/Read Cycle to the controller
Datain[32]	Input data for Write Cycle to controller
Reset	Reset the Memory
Clock	Clock
Wr	Write Enable Pin for write cycle
Addr_ram[13]	Address for Write/Read Cycle to the BRAM
Data_ram[13]	Output Data from Controller to BRAM for Writing
We	Write Enable Pin for write cycle to BRAM

Table 1: True Dual-Port Block RAM Controller Functions and Descriptions

2.2.3 Wrapper

This module serves as the top module which contains the controller and BRAM modules inside of it. The input from the wrapper goes to the controller which gives control signal, data and address to BRAM for reading or writing operation.

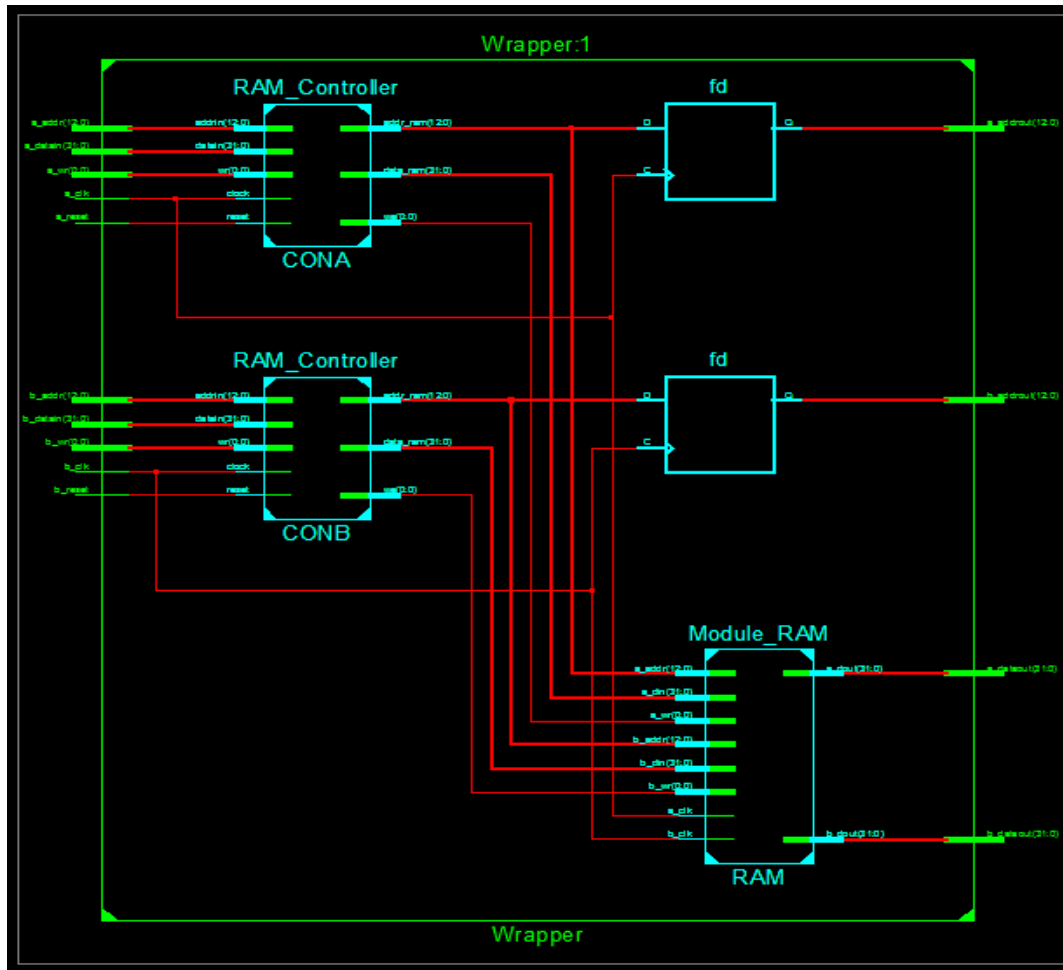


Fig 10: RTL Schematic for Wrapper

Port Function	Description
[a b]_addr[13]	Address for Write/Read Cycle
[a b]_din[32]	Input data for Write Cycle
[a b]_wr	Write Enable Pin for write cycle
[a b]_clk	Clock
[a b]_dout	Output Data from BRAM

Table 1: Wrapper Functions and Descriptions

3 Simulation Results

For the first cycle the Reset signal is enable high which will reset all the inputs and outputs of both the Ports of RAM module, as you can see for the first 10ns all the inputs and outputs are zero as Reset is enable high.

For the next 10ns the write enable signal is high for the Port A of RAM and also we give the address and data to be written on Port A of RAM, during the write cycle the output at the Port A is undefined because at the time of writing no data is available at the output Port A.

In the next cycle we are reading the data from Port B of the RAM module that's why the write enable of Port B is low and the same address is given on Port B to get the data which we wrote on the Port A during 2nd cycle.

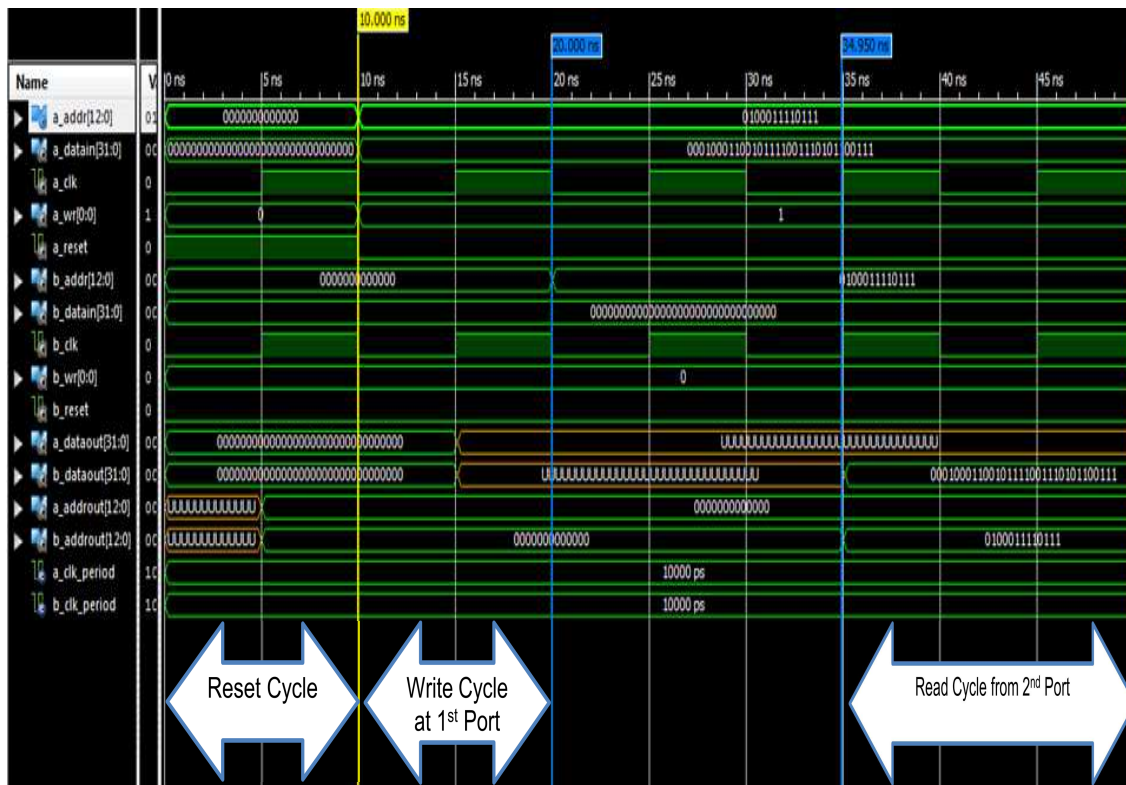


Fig 11: Simulation Results

4 Conclusion

The dual port Block RAM module which has been developed allows the use of the internal BRAM blocks as dual port memory inside the Virtex 6 FPGA of the GLIB card. The project will find application in the ngFEC firmware for the upgrade of HCAL. The author was involved in programming the dual port block RAM using very high speed integrated circuits hardware description language (VHDL).

5 Acknowledgements

I would like to thank to my supervisors Dr. Alan Campbell and Mr. Ozgur Sahin for providing me such a friendly and conducive environment for carrying my project that made my summer research experience wonderful.

I would like to thanks Frau Andrea Schrader, Dr. Olaf Behnke and others in DESY, Hamburg for organizing such a great summer programme. I am highly indebted to my parents for their constant encouragement and support.

6 References

[1]

<http://indico.cern.ch/getFile.py/access?contribId=197&sessionId=81&resId=1&materialId=slices&confId=170595>