



Implementation of FPGA Based Communication Network using High Speed PCIe and Multi Gigabit Transceivers (MGT)

Presentation by:

Jayant Sharma

(2011059)

Diksha Moolchandani

(2011046)

(Indian Institute of Information
Technology Jabalpur)

Under Supervision of:

Mr. Abhishek Bajpai

(Bhabha Atomic Research
Centre, Mumbai)

Mr. Saket Saurav

(Indian Institute of Information
Technology Jabalpur)

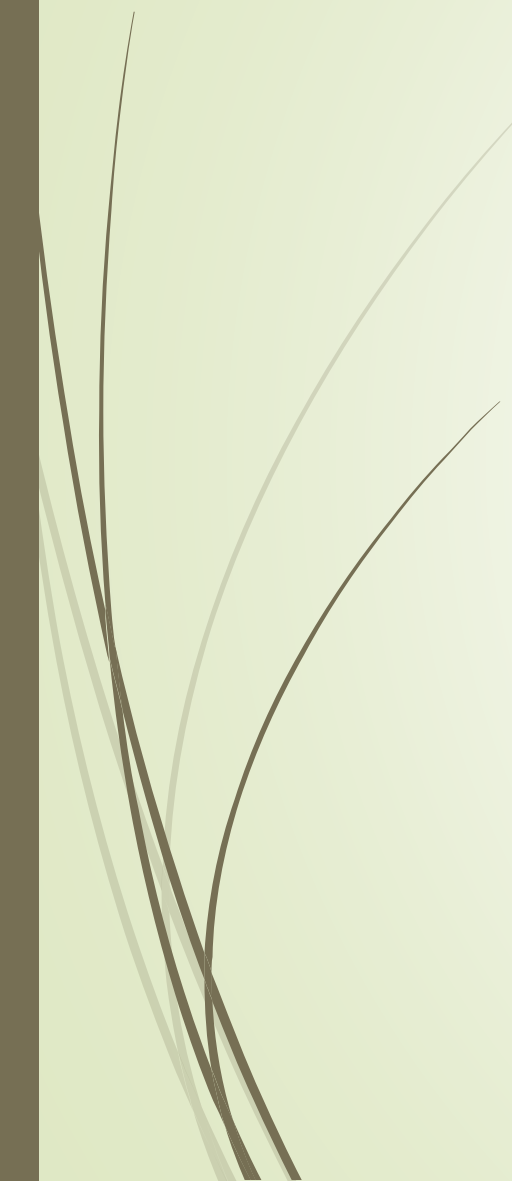


Agenda

- Problems and Needs
 - Purpose
 - Introduction to FPGA
 - Existing Solutions
 - Previous Work
 - Proposed Architecture
 - High Speed Communication
 - Layered Structure
 - Features
 - Implementation Details
 - Future Scope
- 



Problems and Needs

- Increase in data and network security requirements created demand for powerful cryptography algorithms, e.g. AES, ECDLP, RC4 etc.
 - Such applications are computationally intensive.
 - Supercomputing applications.
 - Complete dependence on supercomputers leads to high costs.
 - Need hardware software trade-off for efficient implementation.
- 



Purpose

We developed a hardware architecture that could serve these purposes:

I. High Throughput

- ▶ High speed communication.
- ▶ Cluster of devices used to increase performance.
- ▶ Large no. of operations per unit time.

II. Parallelism

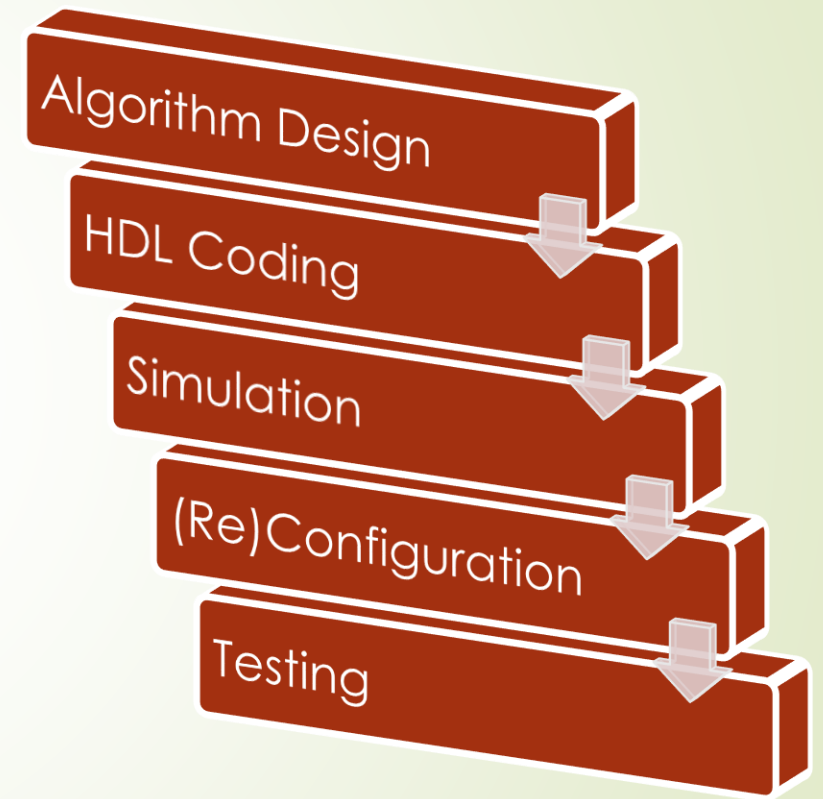
- ▶ Task level parallelism.
- ▶ Data level parallelism.
- ▶ Algorithms running simultaneously on parallel cores.

III. Customizable

- ▶ Hardware suitable for many applications.

Introduction to FPGA

- Field Programmable Gate Array (FPGA)
- CMOS SRAM Cell based semiconductor device
- Re-programmable (after fabrication)
- HDLs like Verilog and VHDL



FPGA Design Process

Existing solutions

CPU Cluster

- ▶ High frequency of operation (upto 4GHz).
- ▶ Smaller number of generic cores per node.
- ▶ High communication overhead.
- ▶ High power dissipation (88W, Core i7-4790).

CPU-GPU Cluster

- ▶ Parallel processing due to hundreds of cores in a GPU.
- ▶ Ease in programming using High Level Languages.
- ▶ High communication overhead.
- ▶ Power (410W, GeForce GTX 570).

FPGA Cluster

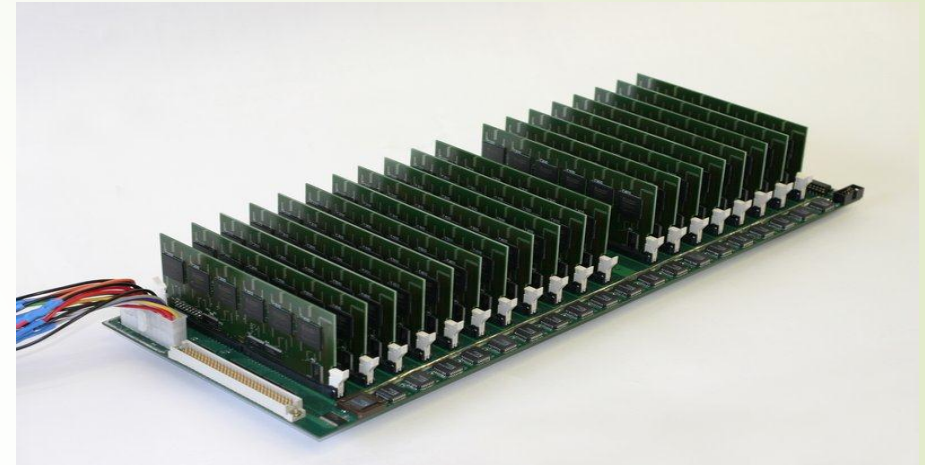
- ▶ Customizable, reconfigurable and flexible.
- ▶ Thousands of application specific cores per node.
- ▶ Power efficient (22W, Virtex-7)
- ▶ Time consuming development cycle and difficult to program.
- ▶ Low frequency (~400MHz).

We chose FPGA based cluster due to a generic requirement in which applications of any complexity and type can be implemented without inducing much costs.

Related Work

COPACOBANA

- ▶ **Cost Optimized Parallel Code Breaker** is a Virtex-4 FPGA cluster.
- ▶ 16 plug-in modules each hosting 8 FPGAs.
- ▶ Plug-in modules connected by 64 parallel bus at 20MHz.
- ▶ Breaks DES in less than a week.
- ▶ Limitations:
 - ▶ Low data rate (1.2 Gbps)
 - ▶ Routing complexity and EMI issues with parallel buses.
 - ▶ Design not scalable.



Source: www.copacobana.org

Proposed Architecture

RING

- Ring network with FPGAs as nodes

DEVICE

- FPGA devices
- **Source node FPGA**

HOST

- Master Control CPU
- Controls operations as needed by applications

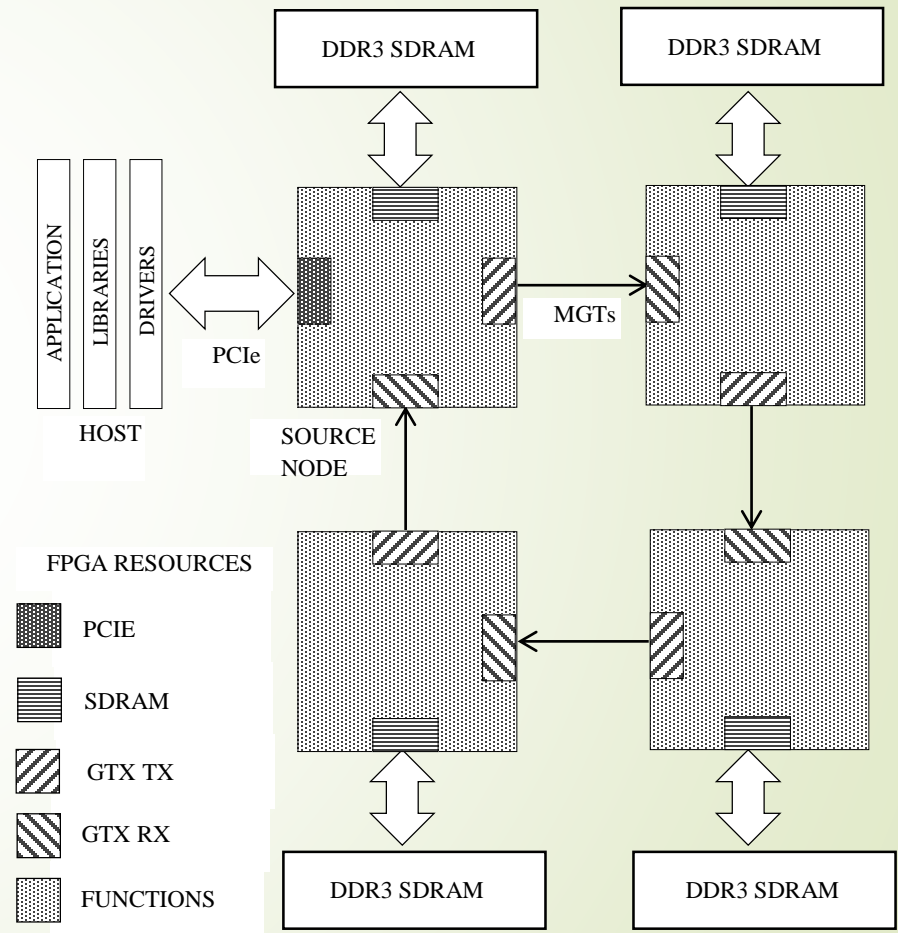
FUNCTION

- Modules on each FPGA
- Each Function allotted a address
- User design specified by application

DDR3 SDRAM Memory

Data Transfer

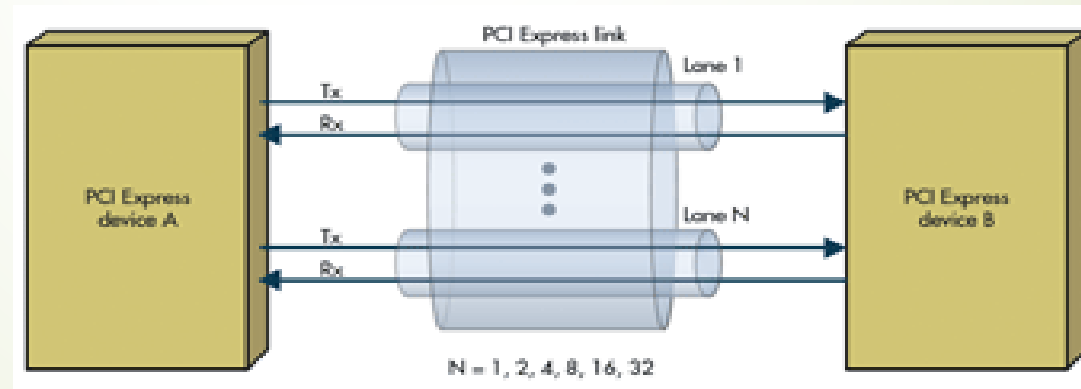
- PCI Express Bus
- Multi Gigabit Transceivers



High Speed Communication

PCI Express (PCIe)

- High performance system bus providing chip-chip & board-board interconnect
- Used to connect Host PC and FPGA device.
- Serial bus using differential signals.
- x4 lane Gen1 PCIe bus offers upto 2.5 Gbps data rates per FPGA (for Xilinx FPGAs).
- Uses 8b/10b encoding for data integrity and CRC.

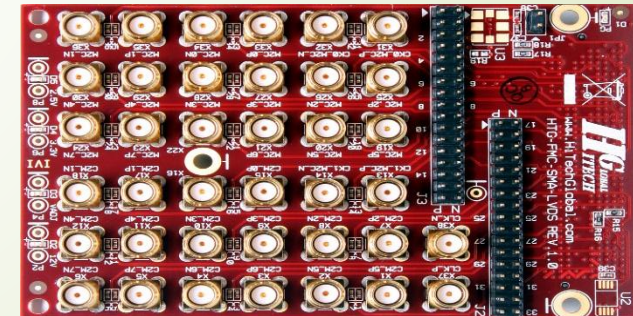
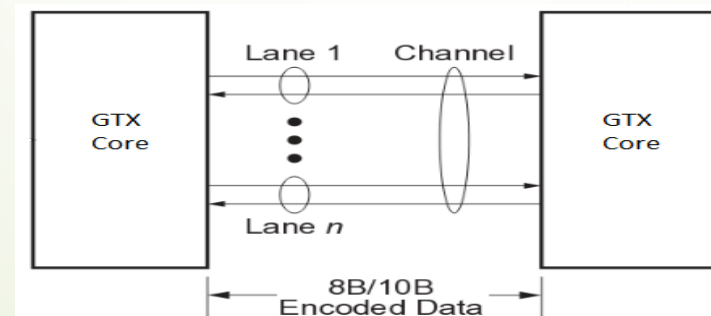


Source: www.google.com

High Speed Communication

Multi-Gigabit Transceiver (MGT)

- Implemented as a Serializer/Deserializer (SERDES) IP Core.
- SERDES is functional block containing Serial In Parallel Out (SIPO) and Parallel In Serial Out (PISO).
- It is used for low cost and high speed communication with serial/differential interface.
- FPGAs can be interconnected at a maximum line rate of 3.125Gbps.
- Transceivers can be physically connected to SMA (SunMiniature version A) ports to give differential signals for duplex communication.



Host to FPGA

Python Application

- ▶ Takes data from user to be read/write to FPGA.
- ▶ E.g. n samples of a signal to be processed.

C Driver

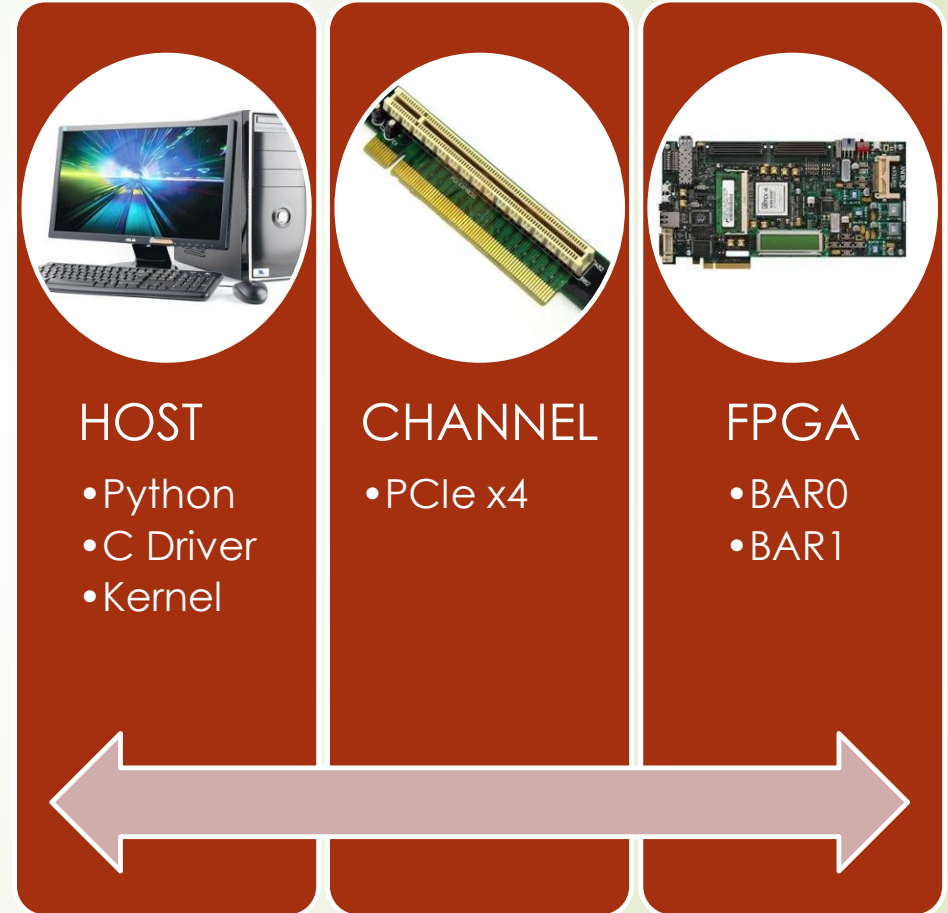
- ▶ Copies this data to kernel memory.

Kernel

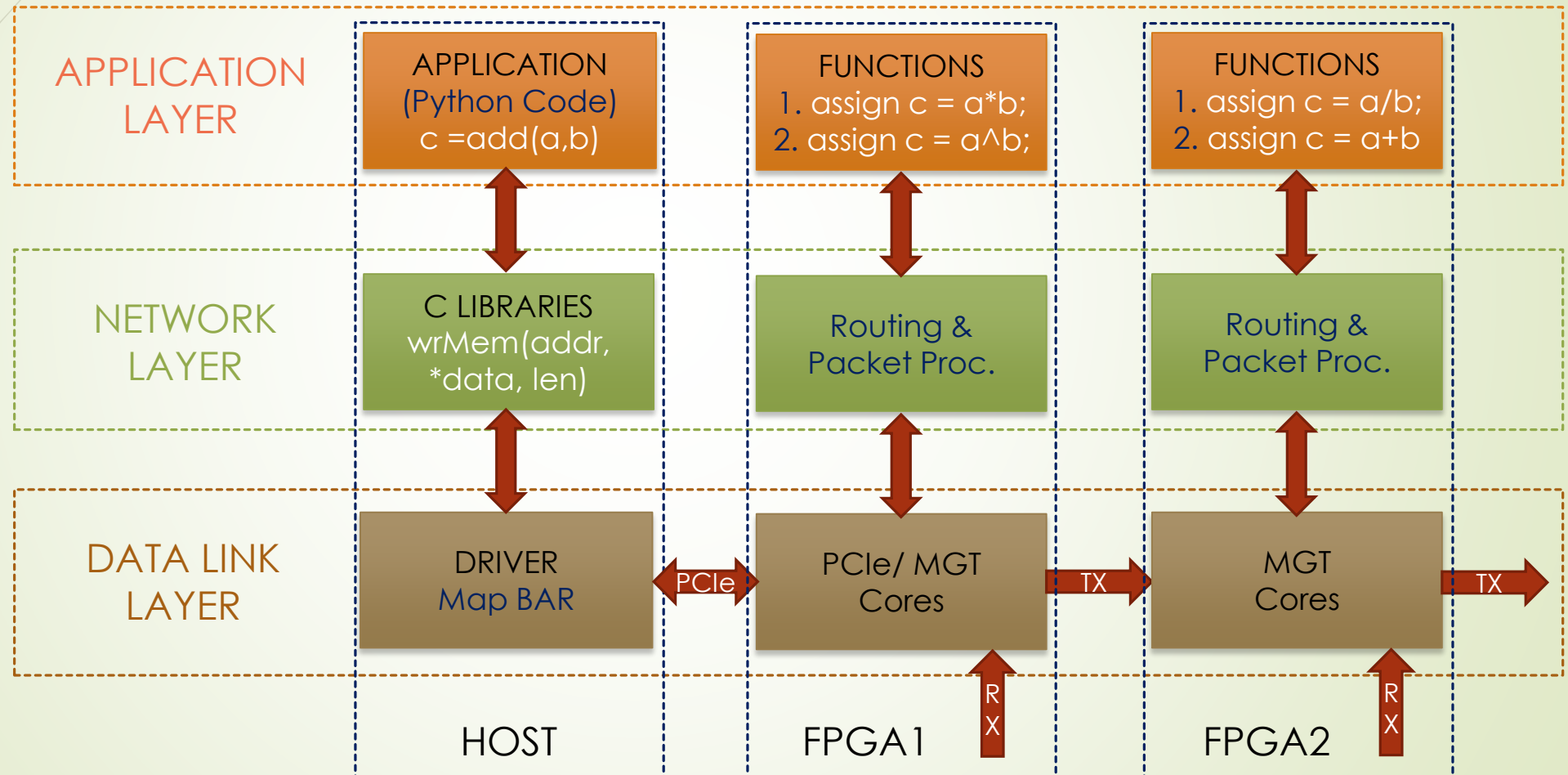
- ▶ Provides abstraction of hardware
- ▶ Application interacts with I/O, memory via Kernel.

Base Address Registers (BARs)

- ▶ Configuration and I/O registers



Data Flow through Layers with Simple Example



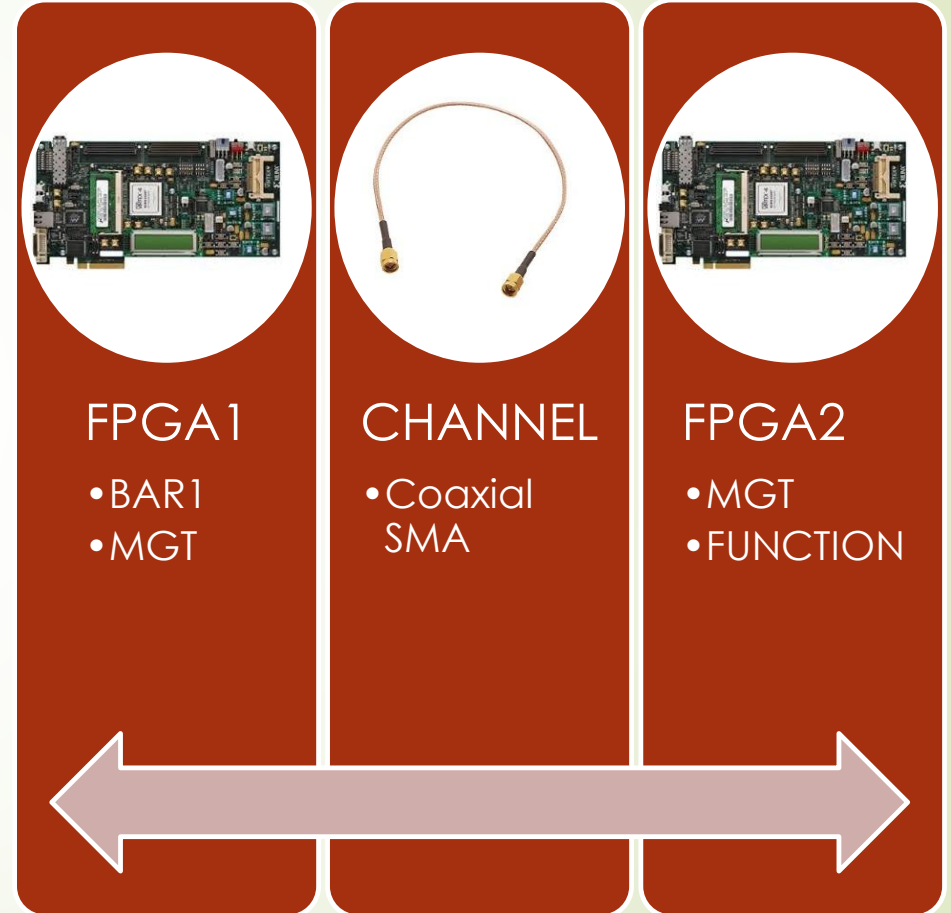
FPGA to FPGA

FPGA1

- Data from Host is stored in BAR1.
- This data sent to FPGA2 in form of packets.
- MGT used for sending and receiving data.

FPGA2

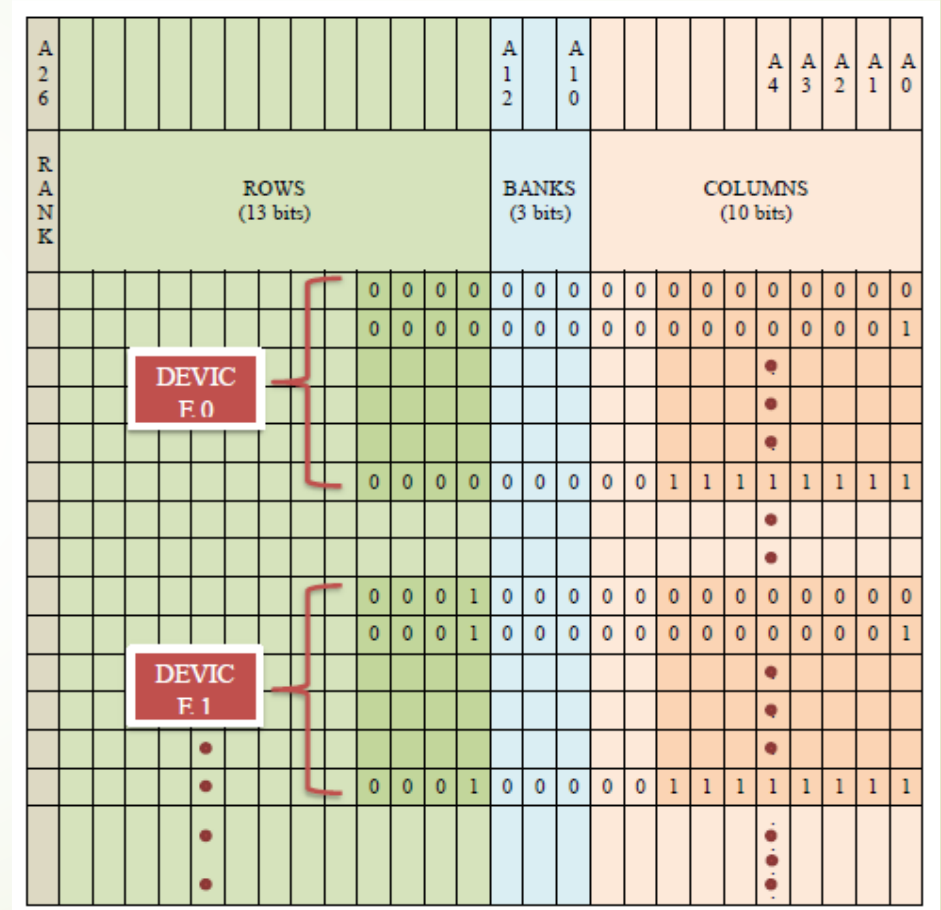
- Receives data from FPGA1.
- Processes data according to the application.
- Function can be processor, filter, any parallel algorithm.
- Sends processed data back to FPGA1.



DDR3 Memory

DDR3 Synchronous Dynamic RAM

- R/W 32 Bytes per cycle
- 1 Row contains 1024 (2^{10}) columns
- 1024x64 (8KB) memory space to each function
- Total 8192 rows i.e. 8192 Functions



DDR3 SDRAM Operation





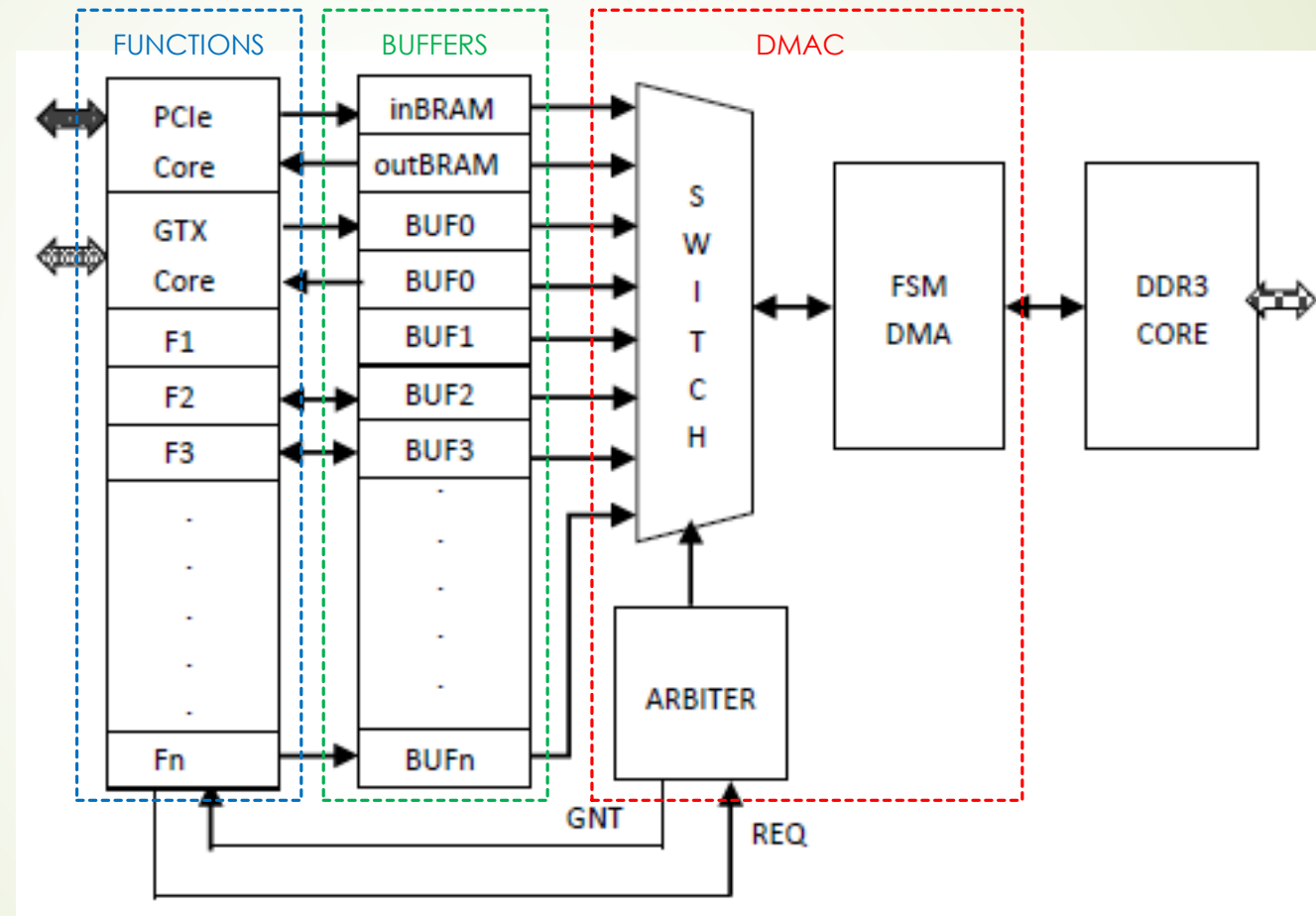
Network Layer

- ▶ Provides data flow path and routing infrastructure on FPGAs using DMA Controller.
- ▶ Handles incoming and outgoing data from Data Link Layer.
- ▶ Data in the form of packets is transferred from Host PC to the source node.
- ▶ Packet processing is done by network layer.
- ▶ Packet contains data to be transferred and identification information of the targeted function.
- ▶ Packet from Host is written into dual-port BRAM called **inBRAM** of size 8KB. **outBRAM** of 8KB size holds outgoing packets.
- ▶ Device ID (DID) of every packet stored in inBRAM is checked to find out which FPGA device it belongs to.

Packet Structure

Order	Packet content	Size (Bytes)	Description
1	PKT START	1	Start byte for packet
2	DID	1	Device ID of targeted FPGA
3	UFID	2	Unique Function ID of targeted function
4	COMMAND	1	Commands like READ/WRITE/RESET etc.
5	LENGTH	2	Total number of data bytes (data_sz)
6	ADDRESS	2	Starting address for memory operation
7	DATA	data_sz	Data payload of max. 1024 bytes
8	PKT END	1	End byte of packet

Components of Network Layer





Network Layer (contd.)

1) Local Buffers

- ▶ Dual port BRAM of size 8KB
- ▶ Needed to store temporary data from functions.
- ▶ SDRAM can be accessed by only one function at a time through DMA.
- ▶ If all functions require such buffers, a total of 230 functions can be implemented per Virtex-6 FPGA.
- ▶ But maximum number of functions is limited by logic used by single core of algorithm function and device resources available.
- ▶ Main resources which are required include total number of BRAMs and Slices supported by FPGA, and external DDR3 memory storage.

Network Layer (contd.)

2) DMA Controller

- ▶ Data conflicts occur when all functions on a single FPGA try to access memory.
- ▶ DMA Controller (DMAC) is implemented which controls access to 256 bit DDR3 memory bus based on requests by functions using a priority scheme.
- ▶ DMAC consists of:
 - i. **Priority Arbiter** - PCIe request from Host is given the highest priority followed by SMA and then other FPGA functions being of same priority are handled in round robin manner.
 - ii. **FSM DMA** - . On grant of R/W request Finite State Machine transfers data between local buffer of device and its corresponding address space in SDRAM. Also, this FSM implements SDRAM as a FIFO buffer for each device so that it cannot be read or write when buffer is empty or full respectively.

Network Layer (contd.)

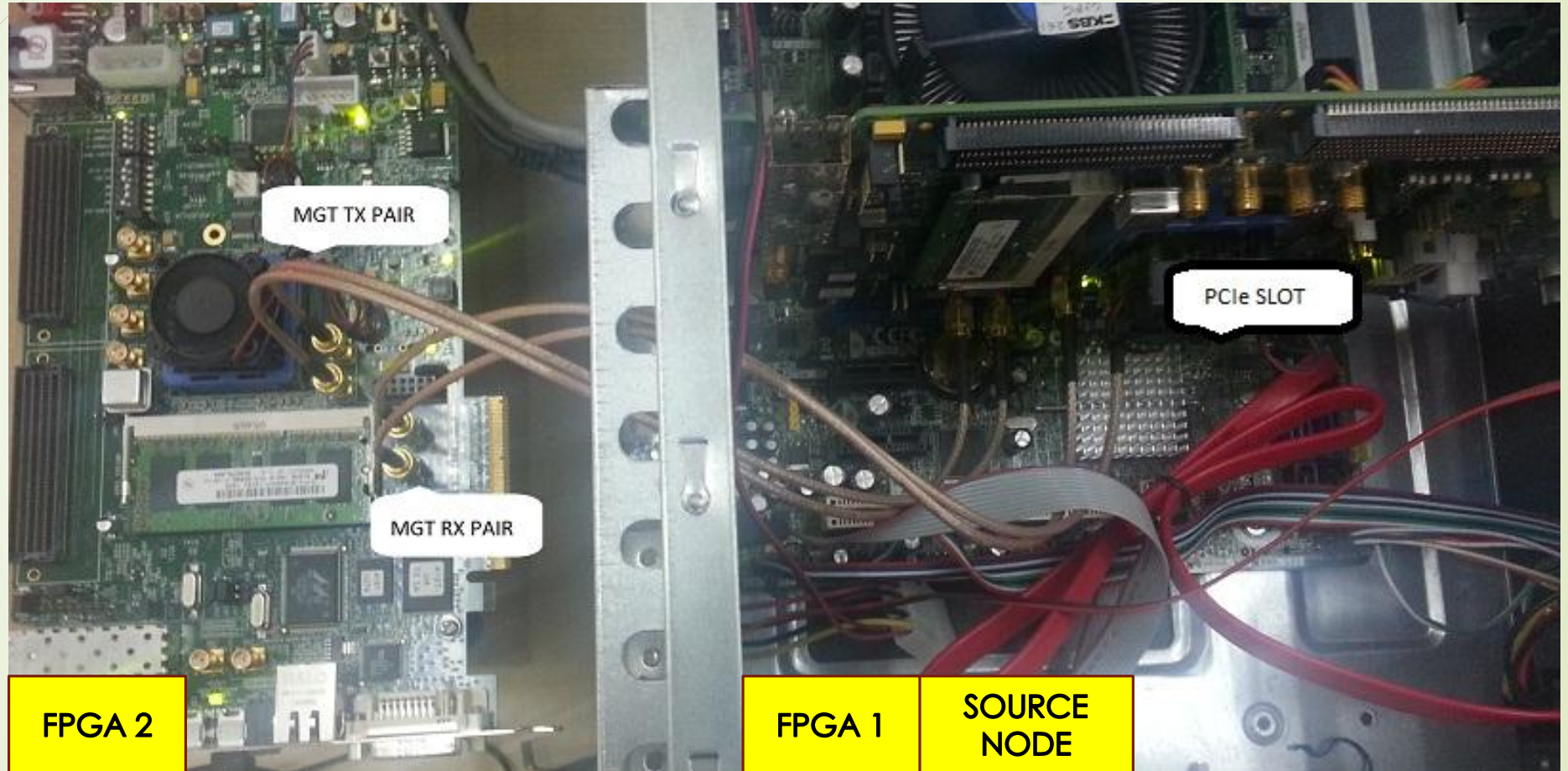
3) Functions List:

- ▶ Each function writes back its status after completion of operation.
- ▶ Status registers of all devices are stored on the source node FPGA. 16KB space is assigned to BAR1 for Status Registers which is directly accessed by Host.
- ▶ These registers are organized on the source node FPGA as a doubly Linked List structure *FuncList*. The Status pointer of each function points to its Status Registers.

```
struct FuncList {  
    struct FuncList;  
    char DID;  
    int FID, UFID;  
    int *Status, *Command;  
    struct FuncList *next , *prev;  
} *function;
```

- ▶ New devices can be added to the ring just by updating this list on source node.

Communication Between Host, FPGA1 and FPGA2





Implementation Details

- ▶ **Hardware:** Xilinx Virtex-6 XCV6LX240T FPGA
- ▶ **Results:**
 - ▶ Host to FPGA communication achieved a measured throughput of above 700 MBps.
 - ▶ 256 bit data read/write every cycle from DDR3 memory at a clock of 200MHz providing a data rate of about 47Gbps.
 - ▶ Data transfer using GTX Transceivers and SMA Cables occurs at line rate of 3.125Gbps with single line being used.
- ▶ **Features:**
 - ▶ Expandable ring upto any no. of devices.
 - ▶ For a ring of 100 devices and 256 functions per device, all 25600 functions can be initialized in 0.8s.



Read and Write Throughput using PCIe

DATA SIZE (kB)	WRITE (MBps)	READ (MBps)
1	153.98	103.25
2	234.05	189.42
4	330.95	246.87
8	496.09	367.78
16	542.57	446.33
24	627.34	463.55
32	756.37	516.78

Resource Utilization of a Single FPGA

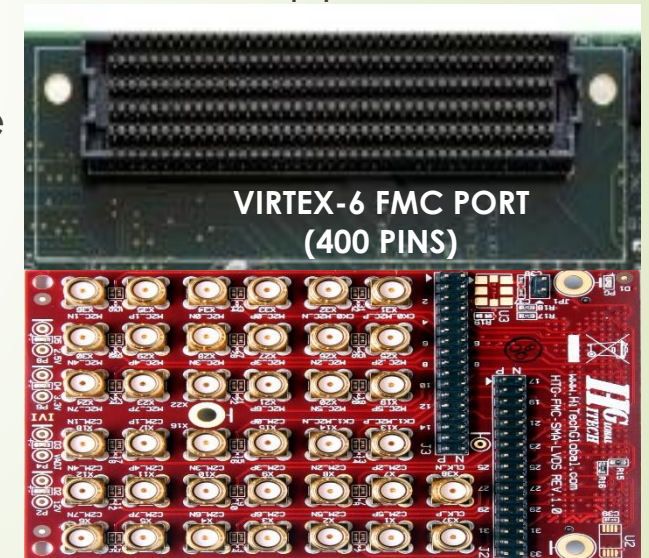
Cores	Slices		LUTs		Registers		Bram 36Kb
PCIe Core	3109	8%	10221	6%	8772	2%	30
GTX IP Core	98	1%	204	1%	163	1%	10
DDR IP Core	2291	5%	3551	2%	5322	1%	0
DMA with 1 Function	2180	5%	3592	2%	5302	1%	2
DMA with 10 Functions	2188	5%	3583	2%	5302	1%	20



Our paper titled **“FPGA Cluster Based High Throughput Architecture for Cryptography and Cryptanalysis”** has been accepted and presented at **National Workshop on Cryptology 2014.**

Future Scope

- ▶ Many Host PCs can be connected in a network with each host controlling a ring of FPGAs.
- ▶ Latest FPGAs like Virtex-7 provide PCIe gen3 with line rate of 8Gbps.
- ▶ Functions can be separately configured if FPGA supports Partial Reconfiguration.
- ▶ Virtex-6 provides 20 GTX pairs of which 8 are present on **FMC** port.
- ▶ Achievable FPGA-FPGA throughput:
 - ▶ FMC: $3.125 \times 8 = \mathbf{25Gbps}$
 - ▶ SMA: 3.125Gbps





Thanks