

CoMA: Towards a Configurable Many-core Accelerator for FPGA-based Embedded Systems

Marco Ramirez, Masoud Daneshtalab, Pasi Liljeberg, Juha Plosila

Abstract

Hardware accelerators release the general purpose processor of a system from very compute-demanding tasks. This work presents a Configurable Many-core Accelerator for FPGA-based systems, named CoMA. Its architecture combines an array of processing cores interconnected by an NoC, with an I/O interface based on the AXI protocol. CoMA provides the designer with a system abstraction layer that facilitates task partitioning and peripheral access. The implementation of the I/O interface was verified through simulation, and synthesized for an FPGA.

Main features

- CoMA explores the concept of utilizing an array of processing cores for accelerating the execution of the most compute demanding tasks of an application.
- Its architecture comprises an array of processing and I/O-specialized cores interconnected by a Network-on-Chip (Figure 1).
- Connectivity with other system components is provided through the industry-standard AXI bus thus assuring compatibility with third-party IP cores.
- Message passing communication model that facilitates the implementation of data-flow applications and maximizes task parallelism.
- A System Abstraction Layer that provides transparent inter-task communication and access to I/O devices (Figure 2).
- External RAM and I/O peripherals are mapped onto a single memory address space.

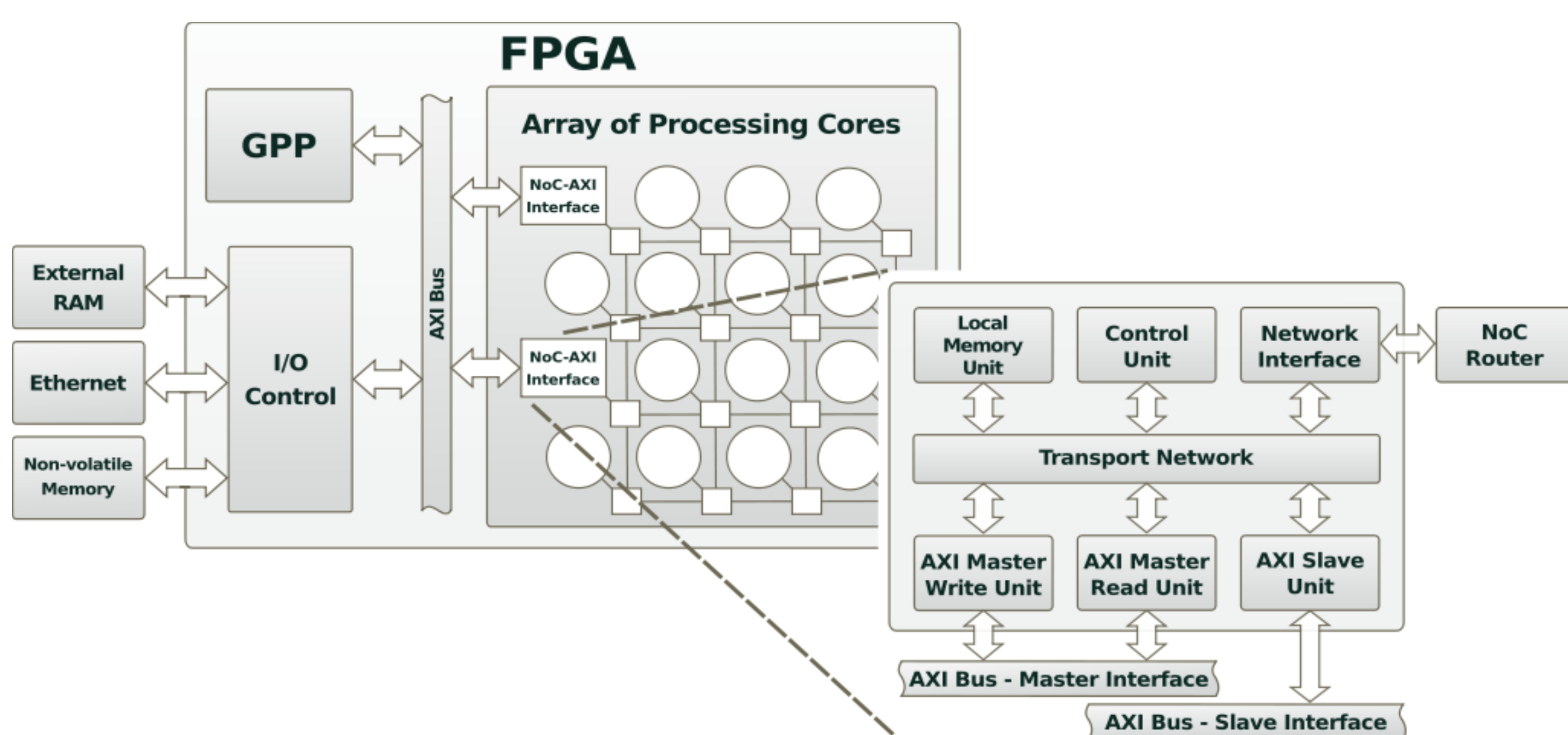


Figure 1. Architecture of a CoMA-based system.

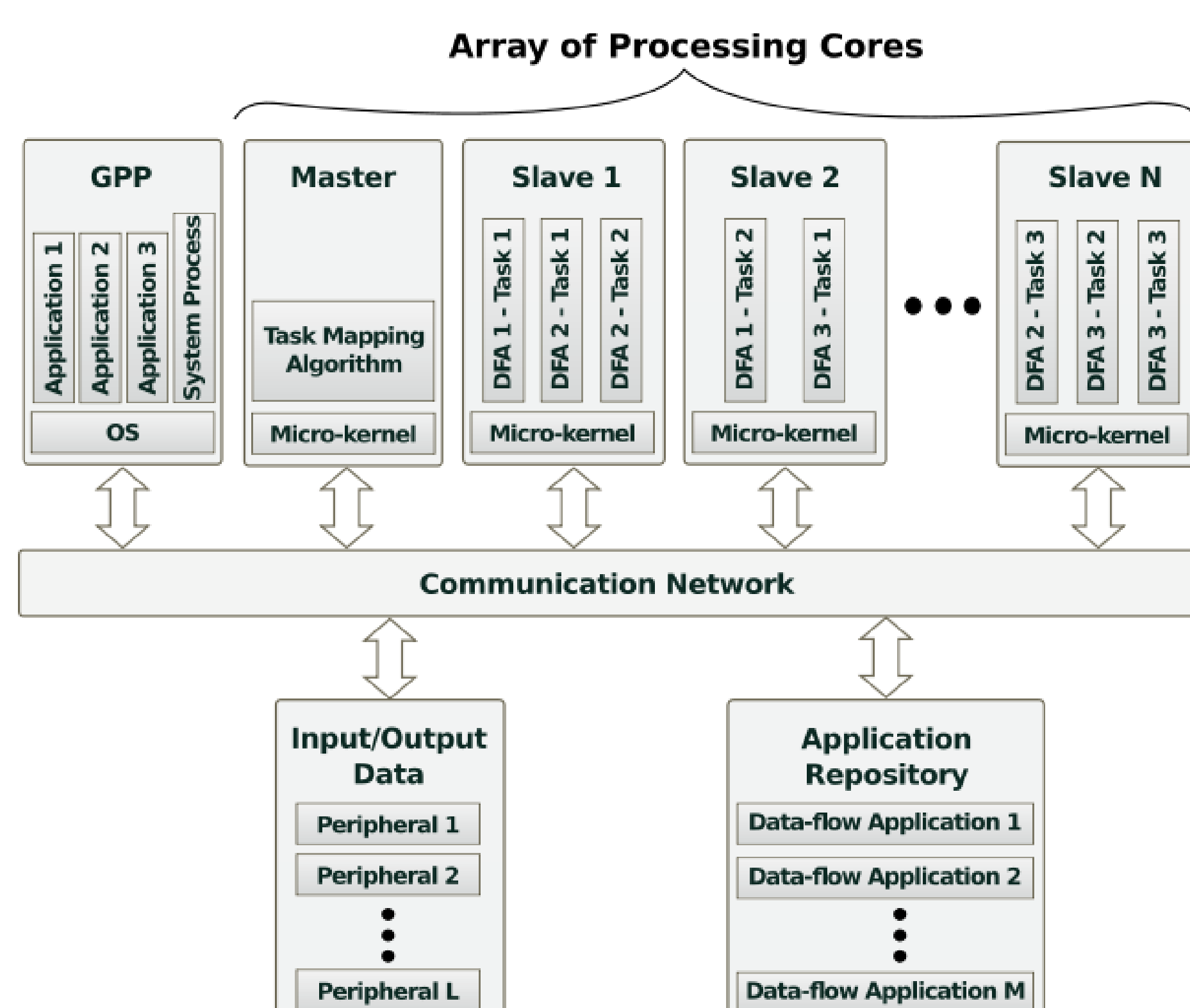
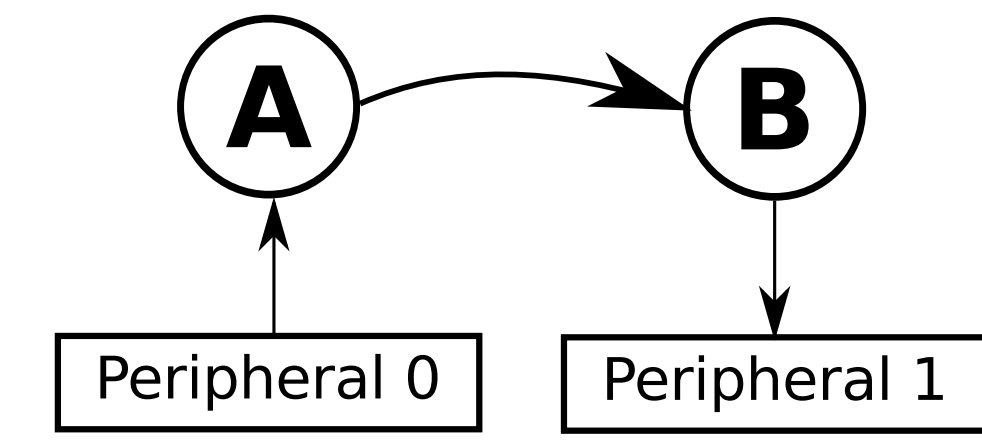


Figure 2. System Abstraction Layer.



Task A pseudo code

```
void main() {
    Read_IO(ioAddr0,buffer,bufferSize);
    // Computations
    Send(taskB,msg);
}
```

Task B pseudo code

```
void main() {
    Receive(taskA,msg);
    // Computations
    Write_IO(ioAddr1,buffer,bufferSize);
}
```

Figure 3. Application example.

System Abstraction Layer

CoMA is designed to accelerate data-flow applications, which are typically represented as task graphs. An application comprises a set of tasks (nodes) that pass messages among them (edges). The System Abstraction Layer defines a very simple application programming interface (API) through which the tasks of an application can exchange messages and get access to the I/O devices. The developer only needs to create a file for each task and use the API to implement all the communication functionality of the task. Figure 3 shows an example of a small application, while Figure 4 depicts the typical work flow.

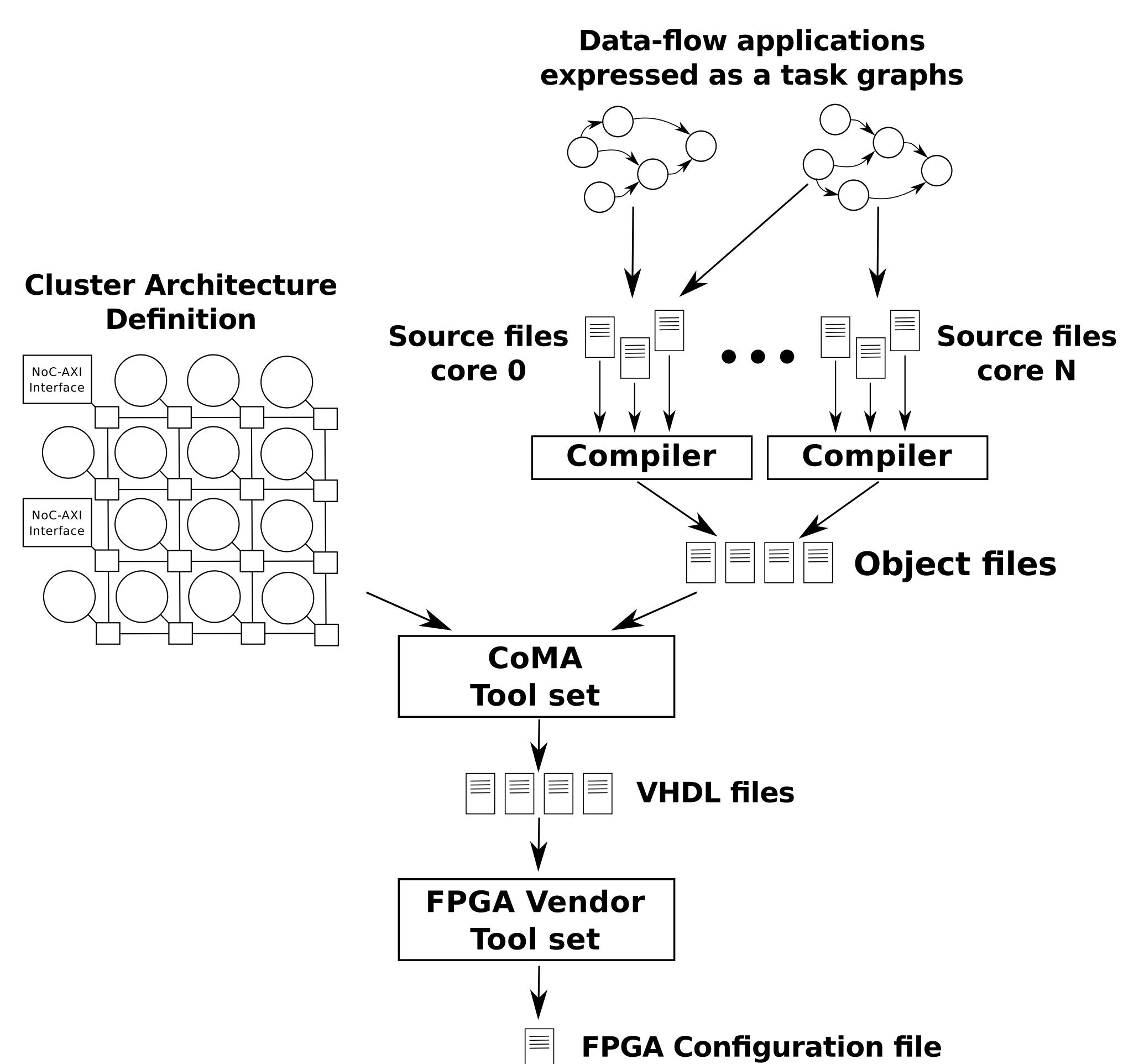


Figure 4. Typical work flow for a CoMA-based system.

NoC-AXI Interface

The NoC-AXI Interface is the key component of CoMA, therefore the implementation efforts have been focused on it. Figure 5 shows the amount of resources utilized by the NAI on a Virtex-6 FPGA.

Functional Unit	LUTs
AXI-Master Write Unit	238
AXI-Master Read Unit	261
AXI-Slave Unit	66
Control Unit	59
Local Memory Unit	457
Network Interface	237
Glue Logic	185
Total	1,503

Figure 5. Amount of resources required by the NAI.

Contact Information

Marco Ramirez
Email: maanrl@utu.fi