EMBEDDED PARALLEL SOFTWARE FRAMEWORK FOR ON-CHIP DISTRIBUTED PARALLEL COMPUTING

Jaume Joven Murillo

Dep. MiSE
Universitat Autònoma de Barcelona (UAB) / Edifici Q - Campus UAB (ETSE)
08193 Bellaterra (Barcelona)
jaume.joven@uab.es, jjoven@microelec.uab.es

Jorge Luis Zapata Muga

CEPHIS – Dep. MiSE
Universitat Autònoma de Barcelona (UAB) / Edifici Q - Campus UAB (ETSE)
08193 Bellaterra (Barcelona)
jlzapata@microelec.uab.es

David Castells Rufas

CEPHIS – Dep. MiSE
Universitat Autònoma de Barcelona (UAB) / Edifici Q - Campus UAB (ETSE)
08193 Bellaterra (Barcelona)
david.castells@uab.es

Jordi Carrabina Bordoll

CEPHIS – Dep. MiSE /
Universitat Autònoma de Barcelona (UAB) / Edifici Q - Campus UAB (ETSE)
08193 Bellaterra (Barcelona)
jordi.carrabina@uab.es

ABSTRACT

This paper addresses a new hardware/software System-on-a-Chip (SoC) co-design methodology to map PVM/MPI parallel software framework to a developed multiprocessor architecture for distributed parallel computing on a chip. This methodology is based on two concurrent phases. The first one is the software development of the embedded parallel framework for on-chip platforms. Parallel Virtual Machine (PVM) and Message Passing Interface (MPI) are two traditional parallel software frameworks for distributed parallel computing, so we need to translate one of them towards the on-chip environment. The goal of the second phase is to develop distributed parallel on-chip hardware architecture, based on Multiprocessor-System-On-a-Chip (MPSoC) that includes a Network On-a-Chip (NoC) strategy, together with the corresponding distributed memory subsystem. Later, this hardware architecture will be synthesised in a reconfigurable device (FPGA) or Application-Specific Integrated Circuit (ASIC).

Once parallel software framework is loaded into each processor of our hardware architecture, we can develop typical parallel distributed applications to run over the SoC/NoC platform. The result of this work will be complete parallel on-chip architecture for an embedded distributed parallel computing.

1. COMPONENTS OF GENERIC EMBEDDED ARCHITECTURE FOR DISTRIBUTED PARALLEL COMPUTING

The whole set of components needed to generate the on-chip distributed parallel computing architecture. All required components are:

- Soft/Hard IP core processors (P_i): NIOS-II + Avalon bus
- Distributed memory subsystem (M_i)
- Network Interface Controller (NIC_i)
- Network on-chip (NoC)
- Porting of parallel software framework (PVM or MPI)

Figure 1 shows a block diagram of all the components of our distributed parallel processing architecture and the existing relationship between them.

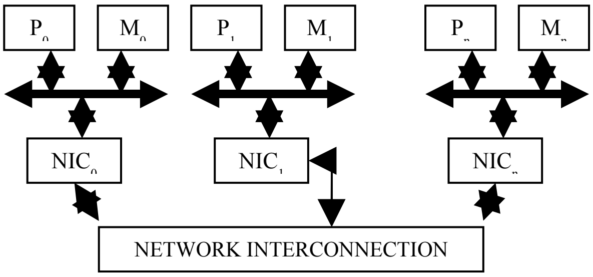


Fig 1. MPSoC/NoC architecture for distributed parallel processing (MPP)

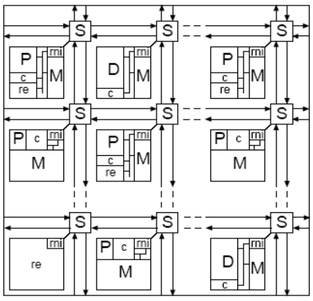
We must remark that the architecture shown in figure 1 is valid for any type of embedded processors (ARM, MicroBlaze, PowerPC, Leon…), of on-chip buses (AMBA, CoreConnect…) and architectural development tools (UltraWizard to develop ARM-AMBA based MPSoC designs, Xilinx ISE). For every case the components of the parallel distributed architecture are the same, and the proposed methodology is still valid.

2. HW DEVELOPMENT

HW design is essentially based on the MPSoC/NOC design and all the communication resources (NICs, on-chip network). The NIC implements the interface following VSIA rules between mono-processor system and the interconnection network, and also the message passing interface between software framework to the hardware bus transactions. So, the NIC implementation has a strong relationship with the porting of parallel software framework over an on-chip environment. There are two different ways to develop a NIC inside the on-chip bus, as bus slave or bus master-slave.

2.1.MPSoC/NoC

Mapping components to a distributed NoC-based MPSoC is straight forward. Every tile of the MPSoC/NoC will have its own NIOSII embedded soft core processor with the memory subsystem with its NIC. On the other hand, interconnection network resources must include a switch or router to communicate the NoC tiles. On that kind of architectures a large number of processors can be interconnected through a complex communication protocol. (store & forward, virtual cut through or wormhole for packet switching or circuit switching).
So, the routing resource implementation depends on the selected topology, in our case a Mesh. In figure 2a and 2b is shown an example of a NoC with Mesh topology, and the associated resource of routing.



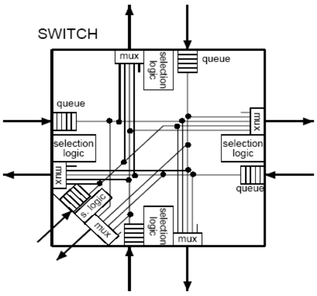


Fig 2a. MPSoC/NoC Mesh architecture

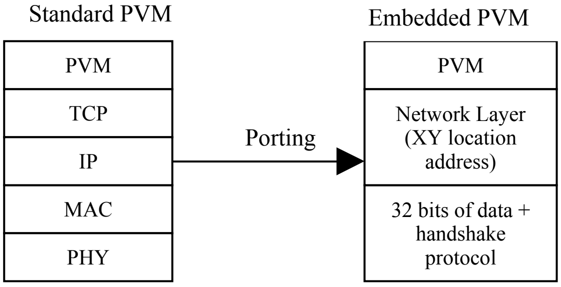
Fig 2b. Switch resource to Mesh architecture

3. PORTING OF SW FRAMEWORK

Both PVM and MPI parallel software frameworks are well know, and they are based on TCP/IP Protocol because the conventional distributed parallel system usually transmit messages over a Local Area Networks (LAN) environments. In our NoC-based MPSoC the interconnection between software an NIC goes through the Avalon bus. Therefore, we have to the lower layer of PVM framework to manage in a different way the message passing mechanisms to the interconnection network through the NIC.

Examining PVM , the selected distributed parallel software framework for this paper we learn that offers lots of routines but only a subset useful for on-chip systems. The main difference relays in the fact that the lowest level interface will interact with a different architecture and components in a transparent way from the application perspective.

In MPSoC/NoC architectures the transport layer is technology independent and the packet size has usually variable size. This layer is in charge of managing the building at NIC of the packets coming from PVM through the network layer. The network layer defines the way the packet is transmitted to the interconnection network from any producer to a given receiver, by obtaining the corresponding information from the network assigned address. (XY address location inside MPSoC/NoC). Data-link and physical layer define the signals for the physical interfaces and the packet formats at bit level.



Standard PVM
Embedded PVM
PVM
TCP
IP
MAC
PHY
Porting
PVM
Network Layer (XY location address)
32 bits of data + handshake protocol

Fig 3. On-chip embedded OSI protocol comparison

4. CONCLUSIONS AND FUTURE WORK

In this paper we present the results obtained using the NIOSII processor, Avalon bus and the presented MPSoC/NoC based architecture. Wherefore, we used an Altera development kit as experimental development platform, based on a EP1S25 Stratix FPGA, with 25660 LEs and 1.944.576 bits of on-chip RAM. The development board also includes two independent banks of 1MB of extern SRAM, 32Mbits of flash memory and other resources. Dual core MPSoC architecture requires 7.539 LEs (29% of FPGA resources), and we achieve a message passing bandwidth of 5.2Mbps using on-chip RAM.

The parallel software framework we developed an embedded based version on JPVM to an on-chip environment. Our embedded PVM framework takes small overhead of memory footprint usage.

The presented hardware/software co-design methodology to develop distributed parallel computing on-chip architecture over a FPGA devices offer several benefits. The selected components make this methodology and its development robust and capable to be reused.

Nowadays a lot of huge computations problems exists and must be parallelized in a distributed way in order to get the results more quickly. Some of those problems, like distributed fractal generation, molecular dynamics simulations, superconductivity studies, or matrix & sorting algorithms could be useful to compute on embedded on-chip environments. For this reason, the methodology presented and developed in this paper is good approach to solve these special problems on embedded environment.