A Survey on Existing MPSOCs Architectures

Med Aymen SIALA
Department of Electrical Engineering
LECAP - INSAT/EPT
University of Carthage, TUNISIA

Slim BEN SAOUD
Department of Electrical Engineering
LECAP - INSAT/EPT
University of Carthage, TUNISIA

ABSTRACT
The majority of recent embedded systems are based on MPSOCs (Multi Processors System On Chip) architectures. This is explained by the possibilities that offers this kind of architectures, as it ameliorates performances by duplicating computing units on the same chip. Besides, this tendency is boosted by technological advances allowing a very large integration scale which is necessary to MPSOC fabrication. As a consequence, the challenge for MPSOCs has changed: Now, the calculation capacity and the number of processors on the same chip are more and more increasing and become often higher than requests. The priority has became then to focus on communication and synchronization between theses processors in order to ensure better performances of the whole system. In this survey we propose to make a detailed study about different architectural aspects of existing MPSOCs: First of all, we will deal with the topologies and the interconnections inside multi processor systems, with comparisons between PtoP (Point To Point), buses and NOCs (Networks On Chip) based communications. Then we will talk about GALS (Globally Asynchronous Locally Synchronous Systems). Finally, we will end with introducing memory architectures of MPSOCs

General Terms
Embedded systems

Keywords
MPSOC, interconnections, point to point, bus, NOC, GALS, memories

1. INTRODUCTION
We ask that authors follow some simple guidelines. In essence, we ask you to make your paper look exactly like this document. The easiest way to do this is simply to download the template, and replace the content with your own material. Historically, the first appearance of MPSOCs has been since the early 1990s, while the symmetric multiprocessing designs of these first MPSOCs were well used for servers and workstations and were promoted by the huge increasing of the integration capacity. Since this time, MPSOCs has known a remarkable advancement and diversification. In 2005, the first personal computers dual-core processors were announced, and as of 2009 dual-core and quad-core processors are widely used in servers, workstations and PCs (Personal Computers). The number of cores inside a chip is increasing a day after the other, and it is expected that it will reach some tens in few next years. At another hand, and facing to the trend of MPSOCs in computer industry, many works have tried to survey the different architectural aspects of this type of architectures. We quote [18], [11] which deal with buses, [21] which deals with different NOCs topologies and [9] which chooses to compare point to point, bus and NOC communications. In fact, the design of MPSOCs presents several important choices, we talk about processors selection, topologies that should be used and routing strategies that have to be adopted. All theses factors have a direct impact on system performances. In the first section of this survey we treat all theses aspects by presenting different MPSOCs communication topologies and strategies inside multi processors system on chip, namely point to point, buses and NOCs. In addition we adopt several comparisons between different interconnections techniques as well as between different implementations of the same technique. Besides, we give different examples extracted either from academic or from industrial world. This first paragraph treats also different NOCs routing protocols in addition to their topologies. The other following sections give a brief overview about either "Globally Asynchronous Locally Synchronous" systems and MPSOCs memory organization. We finish in the last by a conclusion in which we summarize all treated points and we propose some challenges to be studied in next works. The goal of this study is to help designers to decide about architectures that should be adopted according to application domain and different factors that influence the design challenge.

2. TOPOLOGIES AND INTERCONNECT

2.1 Not Communicating Processor
It is a very basic topology (just a duplication of resources) formed by processors that are completely independents and not communicating. In this topology each processor has its own local memory and its system device connected via the local bus (example PLB: processor local bus). For this architecture, processors can not coordinate to perform the same function (that's why this topology is rarely used), however, each one of them can make a specific function [1].

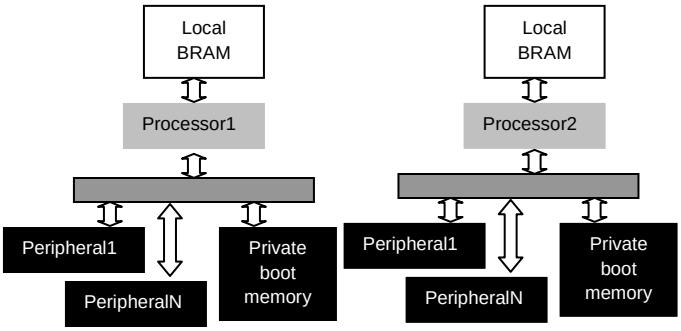


Fig. 1 Not communicating processors architecture

2.2 Point To Point Communication Between Processors

The simplest way to communicate system components is to connect them. This communication is done using direct connections between each pair of communicating IPs (Intellectual Properties). "Point to point" is a simple to implement and efficient (a fast data exchange) topology. But in the same time, it is limited in terms of scalability and flexibility (some rigidity in the system), very complicated and very expensive [9].

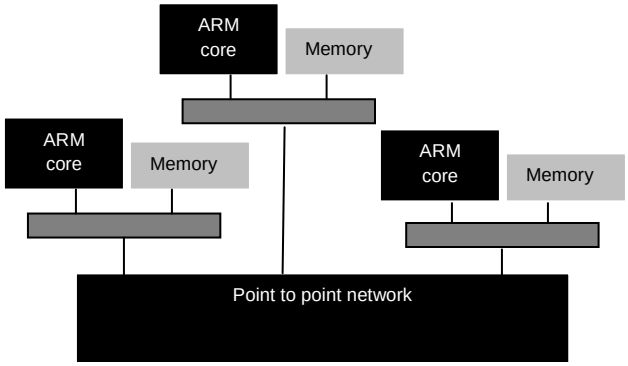
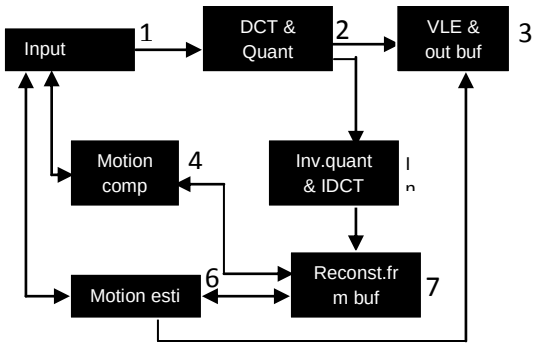
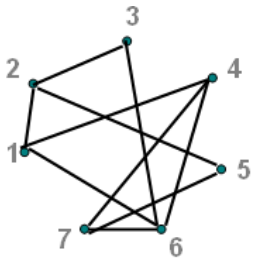


Fig. 2 Point to point communication between resources

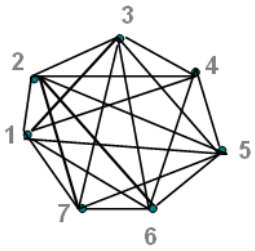
During the "point to point" communication, the throughputs of data transfer are very high between relatively a little number of modules: In fact, we cannot greatly increase the number of modules for a "point to point" communication: This will certainly increase exponentially the complexity of the system and its size, since each resource must have a direct connection with each one of the others resources. In [9], "Lee et al." presented a concrete example of a "point to point "communication: An implementation of an MPEG2 encoder using this type of communication (Fig. 3). The example shows the complexity of the topology despite the low number of "point to point" communications in this architecture: 7 nodes and 10 connections.



(a) MPEG2 encoder using a PtoP communication



(b) PtoP connections between nodes



(c) and if we connect all nodes

Fig. 3 An implementation of an MPEG2 encoder using point to point communication [9]

The implementation will be even more complex if it binds all the nodes, especially, if we want theses communications to be bidirectional. In literature, as well as in the industrial world, many works and industrialized systems and protocols have used "point to point" communication. We quote:

• Coware [10]: An architecture and its associated design flow for MPSOCs with a point to point communication.

• "Rendezvous" protocol: ancient and well known for point to point communication. For this protocol, the sender should be blocked until the receiver is ready to receive and inversely. This ensures that the two sides are synchronized before the transfer takes place.

• OCP (Open Core Protocol) [2]: A point to point interface that provides a standard set of data, control and test signals which allow to different cores of MPSOCs to communicate.

2.3 Processors connected over a bus

The traditional architecture of interconnections in a MPSOC is the bus; it is the most used since it is inspired from the monoprocessor architectures. Some changes (like adding priority and arbitration rules for bus access) make it suitable for multiprocessor systems. For the bus architectures, the arbitration policy has a direct impact on the performances of the MPSOC. The major advantage of the communication by bus is its

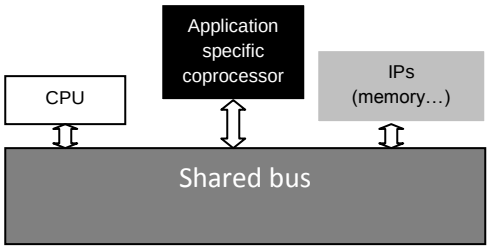


Fig. 4 MPSOC architecture: processors connected over a bus

simplicity (A single channel of communication), therefore a relatively reduced design time. Consequently, the architecture is not very demanding in cost and surface (much less connections than point to point communication)[9]. In the other hand, bus architecture is inefficient [9], with a limited bandwidth and a throughput available between the units on the bus inversely proportional to the number of theses units. In conclusion, the choice of the communication by bus is good for the architectures with small number of units. Otherwise, this communication is characterized by its low flexibility/scalability [9] and by its high energy consumption.

2.3.1 Examples Of Industrial Bus Systems

2.3.1.1 Core Connect Of IBM

"Coreconnect" is an embedded bus architecture; it has a free license owned by IBM [4]. This architecture is based on 3 synchronous buses (PLB : Processor local bus, OPB : On Chip Peripheral Bus and DCR : Device Control Register), a bridge and 2 arbiters (see Fig. 5 below).

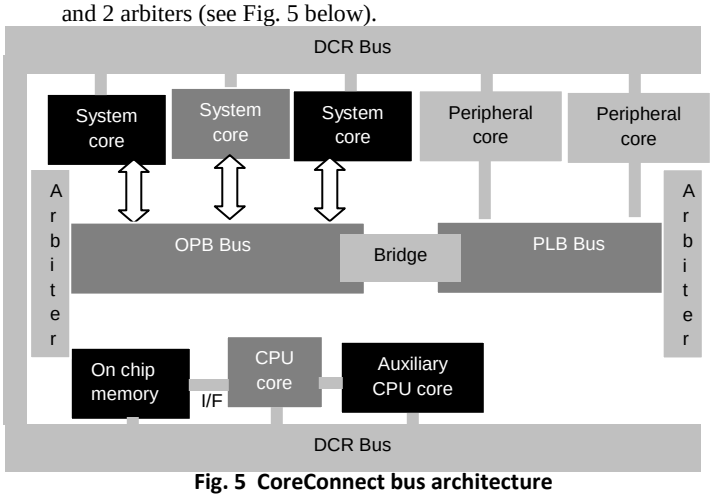


Fig. 5 CoreConnect bus architecture

2.3.1.2 STbus Of ST:

It is a communication architecture developed by "STMicroelectronics", it presents 3 types of protocols[14]:

• Type 1: (The simplest): simple load and store operations.

• Type 2: (comparing to type1): Transfers are more complex and "pipelined".

• Type 3: (comparing to type 2):

– The form of packages has changed.

– Allowing initiators (eg: processors) to have answers in an order different to the one of the requests sequence.

ST bus is characterized by its remarkable flexibility: It can arrange any kind of communication: From the simple shared bus (as AHB of AMBA) to the complete "crossbar"[15]. Physically, "STBus" is formed by 2 channels of data communication: From initiator to target (memories, specific "hw" ...) and inversely [26]. This allows the initiator to send a request while the target is sending the response, therefore, it will be a remarkable improvement in performance. "ST bus" is also characterized by a very effective arbitration policy (eg: It can complete a simple reading transfer in just two cycles while three cycles are needed in the case of "AMBA")[15].

2.3.1.3 AMBA Of ARM

The AMBA bus is a product of "ARM" [5], [11] designed for the family of processors ARM [12]. It is simple and very used. It allows connecting high-performance modules ("ARM" cores, "RAM"s...) via the high-performance bus protocol "AHB" [11]. "AMBA" has an architecture with:

• 3 bus protocols: Advanced High-performance Bus (AHB), Advanced System Bus (ASB) and Advanced Peripheral Bus (APB) (the last is designed for transactions with low performance dedicated to peripherals).

• A bridge which is a connection between two entities operating with different protocols (and/or different frequencies) whose role is to adapt the protocols and to enable the buffering and the synchronization needed to connect the different clock domains[22].

The transaction of "AMBA" bus is as follows [11]:

• Master starts the transaction by requesting access to central arbiter.

• Arbiter decides priorities in case of access conflicts and allows to master to access when the bus is ready (The arbiter uses the "ASB" protocol). Arbitration address and data transfer are pipelined by the "AHB" of "AMBA" in order to increase the effective bandwidth of the bus.

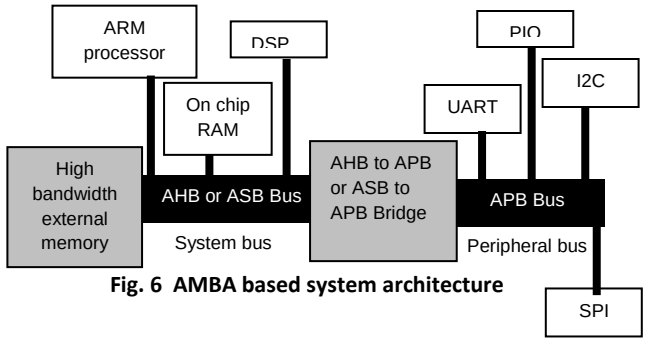


Fig. 6 AMBA based system architecture

2.3.1.4 SiliconBackplane Of SONICS

SiliconBackplane is a highly reconfigurable communication system for ”SOC” developed by ”Sonics”[6], [11], [36]. Its architecture includes a pair of protocols properties of “Sonics”: ”TDMA” and “micro SiliconBackplane network”, plus an open specification interface for ”IP” component compliant with ”OpenCore” protocol.

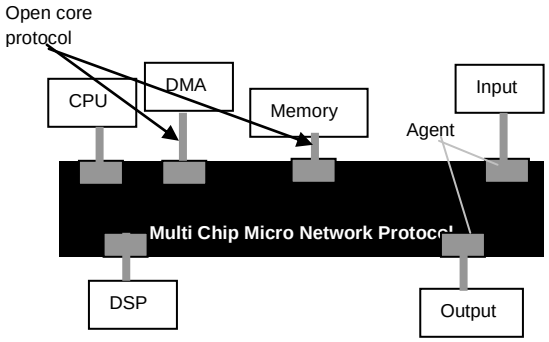


Fig. 7 Architectural model network of Sonics

2.3.1.5 CoreFrame Of Palmchip Corp [18], [19]

Core- Frame is a high performance, low energy consumption architecture[18] formed by:

• 3 synchronous ”bus”:

– CPU bus

– Palm bus: A master slave interface allowing communication between CPU and peripherals (not used for memory access), designed for transfers at low speed and low power consumption [19].

– M bus: Designed for shared memory transfers by CPU or peripherals: High speed transfers.

• PalmBus controller: To connect the two buses: Palm and CPU.

• A cache or bridge to connect the two bus “M” and “CPU”.

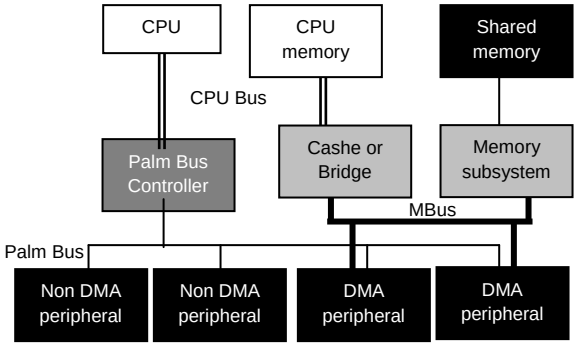


Fig. 8 CoreFrame based architecture system structure

2.3.1.6 Wishbone [17]

Whishbone is a specification of an open source interconnection property of OpenCores [44], [45]. Its objective is to allow to designers to create flexible interfaces between different types of “IP”s, while being independent of semiconductor technology used [17], [45]. ”Wishbone” specification allows consequently promoting the reuse of “IP”s which accelerates the task of MPSoC designers. ”Wishbone” defines two simple and efficiently implemented interface types:

• Masters: Cores able to generate bus cycles

• Slaves: Components receiving bus cycles.

This communication is characterized by:

• Only one bus architecture for all applications

• Multi master mode supporting

• User defined arbitration methodologies

• Flexible Data bus width and operands size

2.3.1.7 Avalon Of Altera Corporation

Avalon bus is mainly used to connect different embedded modules (processors, peripherals) of programmable system on chip, typically for FPGAs designs based on ”NIOS” processor[18]. The bus supports multi master and uses an arbitration technique called slave-side or distributed arbitration and a data transfer of 8, 16, 32, 64 or 128 bits of width.

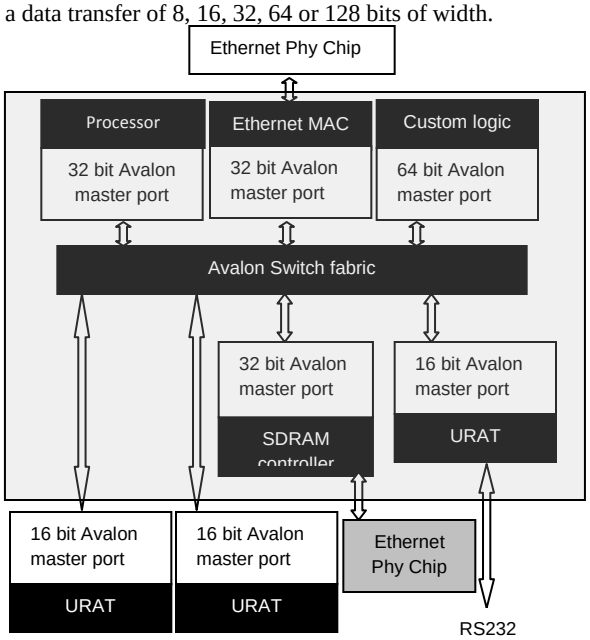


Fig. 9 Avalon bus based system

2.3.1.8 Comparison between different bus characteristics:

Many works have invested in a comparison between various buses described above. We cite [18] which compare more than 6 Buses according to 5 main criteria which are Topology, synchronicity, arbitration, bus width (data and address) and operating frequency:

• Concerning the topology [18] confirm that AMBA is using hierarchical bus topology, Avalon is using the point to point one, Coreconnect has the data lines shared and the control lines forming a point to point ring, wishbone which can be used as a point to point, a ring, an universal shared bus or an interconnection network, silicon backplane which is an interconnection network and coreframe which is formed by bus system(palm bus and Mbus are both point to point)

• When talking about synchronicity [18] precise that al these buses are synchronous

• The arbitration in these buses is as follow: Amba and wishbone are using application specific arbitration (except APB of AMBA which doesn't require any arbitration), Avalon is using slave side arbitration, Coreconnect is using programmable priority fairness and Silicon backplane which using tow level of arbitration, the 1st is TDMA,the 2nd is "round roubin" taken passing.

• The Data bus width is 32, 64, 128 or 256 bits for AHB and ASB of AMBA, 8, 16 or 32 for its APB, 1 to 128 for Avalon, 32, 64, 128 or 256 bytes for PLB of CoreConnect, 8, 16 or 32 bytes for its OPB, 32 bytes for its DCR, 8, 16, 32, or 64 for Wishbone and Silicon Backplane.

• The address bus width, is 32 bits for AMBA, 1 to 32 bits for Avalon, 32 bytes for PLB and OPB of Coreconnect, 10 bytes for its DCR, 1 to 64 for Wishbone and not applicable for the rest(Coreframe and Silicon Backplane).

• Operating frequency which is user defined for AMBA and Wishbone, depending on PLB width for Coreconnect.

2.4 Processors connected over a "crossbar" [8], [16], [17]

A "crossbar" is a switch with multiple inputs / outputs thatbinds each time inputs to outputs. For this purpose a "crossbar" should be formed by a parallel bus matrix [8]. "Crossbar"s are used to overcome bandwith limits of shared "bus". The switch "crossbar" allows more than one master to use the "bus" provided that they not access to the same slave [17], it allows also each master to access to two or more slaves[17]. The operating principle of a "crossbar" is to chain several actions: It starts with requesting a channel in the switch by the master. Once this is done, the data will be transferred in a "point to point" communication. The transfer rate of the "crossbar" is higher than a shared "bus": The "crossbar" can withstand even the very high speeds. However, the major drawback is the complexity of the interconnection logic and routing resources. This affects the cost of the "crossbar" to be so high because of the connections inside the matrix[8]. The "crossbar"s are widely used in industry, they are for example used for "ARM PL300/301", "AMBA3" and "SonicsMX" [13].

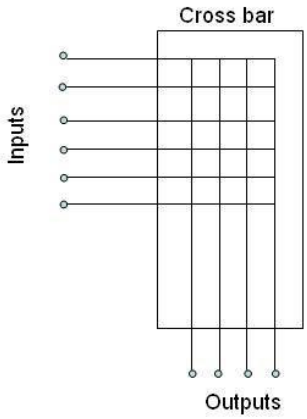


Fig. 10 A crossbar is formed by a matrix of buses in //

2.5 MPSOCs: hybrid communication

There are some interconnection systems that can contain several types of communications. Among theses systems we quote:

• "Wishbone" that defines 4 types of interconnections: Point to point, data flow, shared bus, and "crossbar" ([17], [18]).

• "STbus" which can operates as a shared "bus", a partial or a complete "crossbar" [15], [18].

• "Amba" which operates as a shared "bus", a bridge entities (bridged clusters), a partial "crossbar", a complete "crossbar" or even a complete network on chip (NoC)[20].

2.6 NOCs

2.6.1 Introduction: "NOC" vs "BUS" vs "Point to Point (PtoP)"

According to [21] a "NOC" is defined by :

• Its topology: Nodes positions and connectivities.

• Implemented protocol: How theses links and nodes are used.

Topologies differences between "bus", "point to point" and "NOC" communications are illustrated in the following fig. 11. So why choosing a "NOC" instead of a "bus" or a "point to point" communication? As an answer to this question, many works have focused on a comparing study between different communication strategies:

• In [21], we remark by the mean of comparatives tables between "bus" and "NOC"s that for "bus", every attached unit

add a parasitic capacitance, therefore electrical performance of system degrades with growth. In addition, there is a possibility

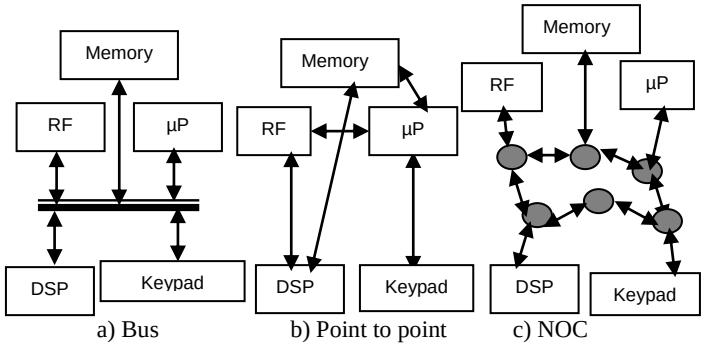


Fig.11 Topologies differences between a) bus b) point to point and c) NOC communications [21]

of deadlock in arbitration with a shared bandwith between all units. In contrary, the major advantage of "bus" is its simple concepts. Concerning "NOC"s, the risk of performances degradation which may appear when scaling doesn't exist: Network wires can be pipelined because links are "point-to-point".

• [9] has also compared the three types of communication: by "bus", "point to point" and by "NOC" and he has concluded that "bus" are simples and not very demanding in cost and surface but not very efficients, with a bad scalability and a high energy consumption. In another hand, "point to point" communication is very efficient but very expensive in surface, above all, in case of complex systems (a lack of scalability). "NOC"s have similar performances than "point to point" communications: More efficient than the "bus" with less power consumption. They are less demanding in surface than "point to point" and they are more scalable than the two others communication techniques. In conclusion, for NOCs we invest more in conception to have a better compromise cost/performance.

2.6.2 NOC Components:

Physically, a "NOC" is essentially constituted of a network adapters, a routing nodes, and links allowing to connect the routing nodes[21], [26]. A network adapter is the implementation of the interface connecting "NOC" "IP"s to routing nodes. Nodes mission consist of routing data according to chosen routing protocol.

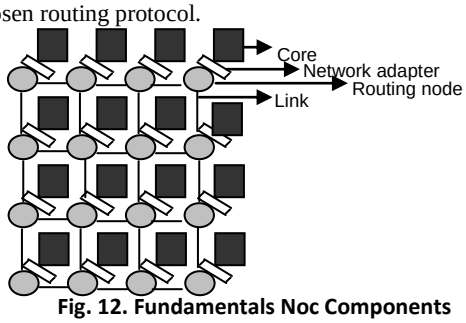


Fig. 12. Fundamentals Noc Components

2.6.2.1 Network Adapters:

The role of a network adapter consist if interfacing cores and network in a standardized way: Network characteristics become transparent from cores side. They allow consequently a simple reuse of "IP"s and offer a high level communication service with minimum of effort. A network adapter contains two sub blocs: A core interface and a network interface. Theses interfaces allow it to assure many functionalities like decoupling cores from network, end to end network flow control implementation, layers system conception approaches facilitation and packetizing/ depacketizing of messages sent by "IP"s in order to be routed by the networks. In literature, many examples have tried to implement network adapters. As examples: "Radulescu et al." [22] and "Bejerregaard et al."[23].

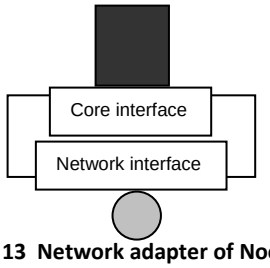


Fig. 13 Network adapter of Noc

2.6.2.2 The Routers [26]

Routers constitute the most important element in the "NOC" architecture. They allow the implementation of the routing strategy and the flow control algorithm. Additionally, the routers contain buffers to save coming packets. The routing algorithm implemented in the routers determinate destination of each entering packet. It usually implements simple minimal-path functions.

2.6.3 NOC's Network Layer

NOC's network layer means the topology of the "NOC" as well as the routing protocol implemented on it.

2.6.3.1 NOC topologies:

NOC's topology means its form. It can be regular like "Spidergon", "Mesh", "Torus" and "Tree" or irregular. It means also interconnections between the different nodes, which can be unidirectionals or bidirectionals. NOC's topology has a direct impact on its performances[21], especially on its power consumption: In [25], "Pande et al." assert a 20 percent to 40 percent of reduction of energy/bit for a same debit only by an optimal traffic localization.

• Regular "NOC" topology: Many regular topologies more or less efficient have appeared. Find some examples:

– "Spidergon" [7], [26], [28]: The "Spidergon" topology is based on an even number of nodes, where each node is connected, rather than to its two neighbours (from the two sides: left and right) via unidirectional links[7], to the opposite by the

center node of the "Spidergon" (see figure 14). According to [7], the index of the three nodes xj connected to a node xi in a "Spidergon" with a number of nodes equal to

N = 2n nodes (0 < i < N) are:

j = (i + 1)mod N (1)

j = (i − 1)mod N (2)

j = (i + N/2)mod N (3)

A "Spideron" is characterized by its low messages latency. It is composed of homogeneous blocs where we use the same type of router to form the totality of the network. So it has a low cost [7] with a simple routing and symmetrical peaks (an optimized traffic [28]).

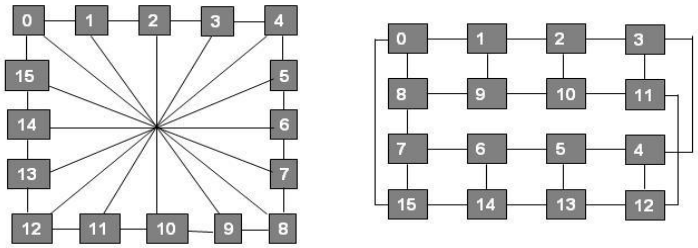


Fig. 14 "Spidergon" Topology and its on chip layout [7]

Many works have used the "Spidergon" topology: In [27], "Zitouni et al." have presented a derivation of "Spidergon" with a node in the center (see Fig. 15).

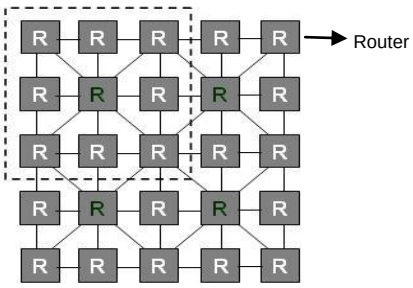


Fig. 15 The "Spidergon" of Zitouni et al. [27]

In [7], "Moadeli et al." gived a model to calculate average latency of messages in the interconnections of "Spidergon" network. In [28], "Bononi et al." proved the performances of a "Spidergon" topology, they concluded after a comparative survey with "Ring" and "Mesh" topologies that "Spidergon" offers the best compromise between performance and flexibility (scalability). In [26], "Concer et al." presented a new routing algorithm in order to better optimize "Spidergon" "NOC"s. "ST microelectronics" has also presented in [28] its new "Spidergon". A well known example of "Spidergon" is the "Octagon" [24]. With 8 nodes and 12 bidirectional links, "Octagon" has a communication between each pair of nodes, and produces a throughput higher than the one of a shared "bus" or a "crossbar", with lower connections than a "crossbar" (see Fig 16).

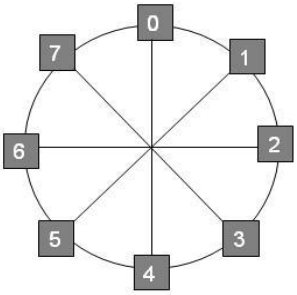


Fig. 16 "Octagon" configuration: 8 nodes, 12 bidirectional links [24]

– "Mesh" [33]: "Mesh" is a simple topology which allows access to all resources. It is characterized by its scalability. A n*n 2D "Mesh" topology is formed by:

1) R = n*n routers [33]

2) Each router (except those on the sides) is connected to 4 neighbours routers and to a core (a processor or a memory) via its input/output channels (figure17).

3) An input/output channel consists of two "point to point" unidirectional communications between two routers or between a router and a resource. The number of communication channels of a n*n 2D "Mesh" is

C = 3n^2 − 2n (4)

Processors (ie nodes) of a "Mesh" communicate by exchanging messages [39].

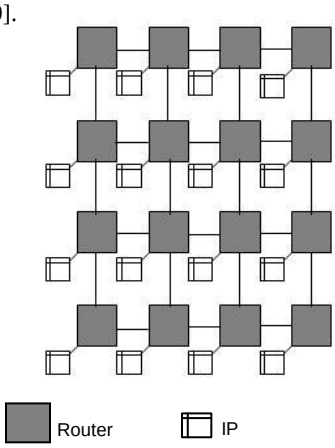


Fig. 17 "Mesh" 4 _ 4 topology [33]

– "Torus" [34]: The only difference between the two topologies "Mesh" and "Torus" is that for "Torus", edge routers of one side are connected to edge routers of the opposite side, which is not the case of "Mesh"(see figure 18). "Torus" is proposed to reduce latencies of "Mesh" topology while retaining its simplicity. Each

router in this topology is connected to 4 neighbours routers and to a core (processor, memory..) through input/output channels. In [34], ”Aghatabar et al.” have presented an alternative to ”Torus” topology: ”folded Torus” to avoid ”Torus” latencies.

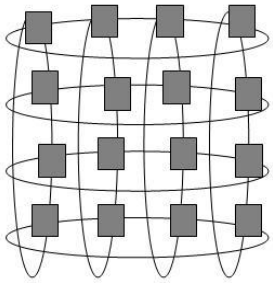


Fig. 18 “Torus” 4*4 topology [21]

– “Tree” [36], [37], [38]: In the ”Tree” topology, we place the routers on the nodes and the terminals on the leaves [37]. Each node is defined by its two coordinates (n, p): n to indicate the node level, p to indicate its position. The father’s number of a node and its childern’s number are not defined by a rule.The connection between routers is bidirectional “point to point”. According to [38], ”Leiserson” has proved formally that ”Tree” topology has the best cost comparing to other regular topologies. Other than [38], several studies have focused on the “Tree” topology. As example we quote: [36] who presented a special “Tree” topology called “Butterfly Fat Tree” where each router (or switch) is connected to 2 parents and 4 children as well as [37] in which “Adriahantenaina et al.” have adopted a “SPIN”(Scalable Programmable Integrated Network) NOC with a “Tree” topology (fat tree) and show that this architecture has better performances in terms of latency than a ”PI” bus especially for a large number of cores.

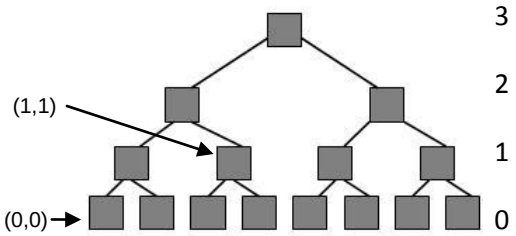


Fig. 19 “Tree” topology

– Regular topologies comparison: Face to theses multiple regular ”NOC”s topologies, several works have focused on a comparative study between them: We quote in this context the work of ”Aghtabar et al.” [34] who tried to compare the two topologies ”Mesh” and ”Torus” in different aspects: Latency, energy/debit ratio, energy consumption ... and this with different routing algorithms and traffic models. They conclude that ”Torus” has always better performances than ”Mesh”, but at the expense of higher energy consumption. ”Pande et al.” Compare in [35] Mesh and Butterfly Tree topologies with an evaluation methodology that they proposed (based on known performance metrics: Latency, throughput and area): we get the result: The Tree topology has less latency until saturation and occupies less surface, but the “Mesh” has more throughput for uniform traffic.

• “NOC”s irregular topologies (called also customized):

Irregular topologies are considered more realistic than regular ones [28] with less constraints on network form. According to [21], theses irregular topologies are always obtained by mixing different regular forms with hierarchical, hybrid or asymmetric way (see Fig.20 below)

• Regular vs. Irregular topologies: Facing to this variety of forms of on-chip networks, the trend is in line to make comparative studies between the two types of topologies. Several works got involved:

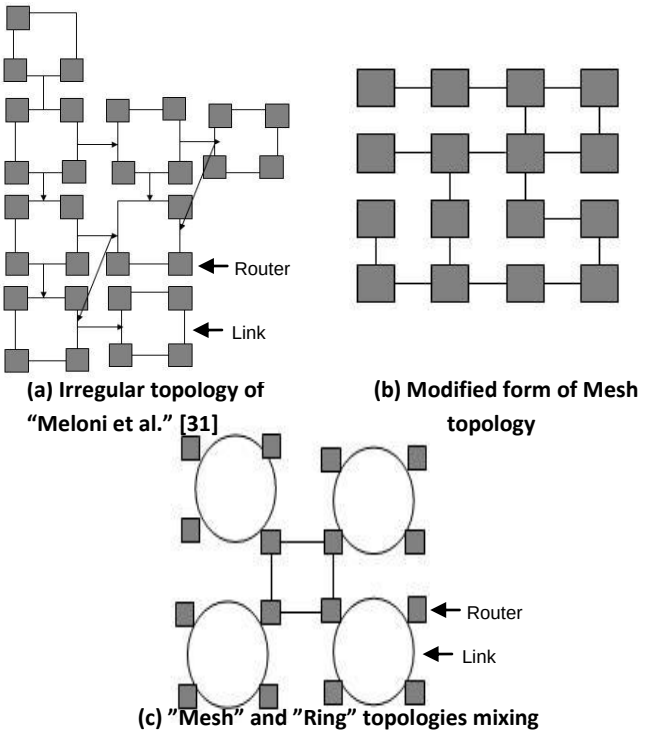
– According to [30], the type of topology (regular or irregular) goes with the scope of the MPSOC and nature of the cores used: The regular topologies are suitable for general purpose architectures with homogeneous cores. Under these conditions (general architecture and homogeneous cores) regular topologies lead to a regular and predictable “layout”s. Instead, irregular topologies are more appropriate for MPSoC’s specific applications with heterogeneous cores and memories and having different sizes. For such systems, the irregular architectures is more efficient than regular in term of energy consumption, area and performance.

– ”Meloni et al.” in [31] confirm this statement by applying a benchmark (Mult benchmark) on a regular topology and then on a customized one and scored much better performances for the second case.

– Similarly, according to [21], the work of ”Jalabert et al.” on video applications shows that irregular networks are more beneficial to XpipesCompiler than regular ones.

– On the other hand, between an irregular Mesh, a second regular Spidergon, and a third ”Ring” topologies, ”Bononi et al.” in [28] prefer the regular Spidergon which gives results comparable to other topologies in terms of performance and flexibility, but much easier in its implementation.

– Regular topologies are also easier to generate: According to [32], with “XpipesCompiler” it is faster to generate a regular topology than another irregular, whatever that is not of great importance since all the generation of any type of “NOC”s does not take with “XpipesCompiler” more than a few minutes.



(a) Irregular topology of "Meloni et al." [31]

(b) Modified form of Mesh topology

(c) "Mesh" and "Ring" topologies mixing

Fig. 20 Examples of irregular topologies

2.6.3.2 "NOC"s: Routing Protocols

"NOC"s routing protocols determinate the sequence of channels that a message packet traverses from its source to its destination [39]. The challenge is to minimize latency while avoiding deadlocks or any other kind of blocking due to the fact that messages are waiting each other. It is also obviously necessary to avoid losing messages when they circulate through the network indefinitely and never reach their destination [39], [41]. Routing protocols can be deterministic or adaptive [34]:

• Deterministic protocols: Route a packet from its source to its destination by a single predefined path [39]. Theses protocols lack flexibility and do not respond dynamically to the network state (do not adapt to network). This has a negative impact on performance by decreasing throughput and increasing NOC's latency. As examples of deterministic algorithms we quote "XY" ([34], [39], [42]) which is a deterministic routing algorithm without deadlock or virtual channels (CVs) proposed for "2D Mesh" topology. In this algorithm, a packet takes his first way in the direction "X" and then in the direction "Y", that's how it allows only 4 types of rotations (see Fig. 21 and Fig.22 below).

• Adaptive protocols: Take into account information on network traffic and channel status to avoid congested regions of the network [34]. This gives more chance to messages to arrive in an optimal way to their destinations [39].

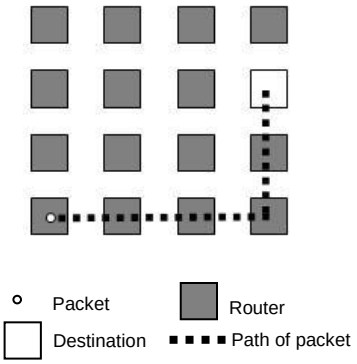


Fig. 21 Deterministic routing protocols route a packet from its source to its destination by a single predefined path. Example: The XY

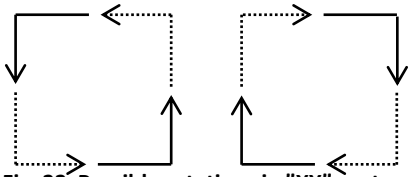


Fig. 22 Possible rotations in "XY" protocol

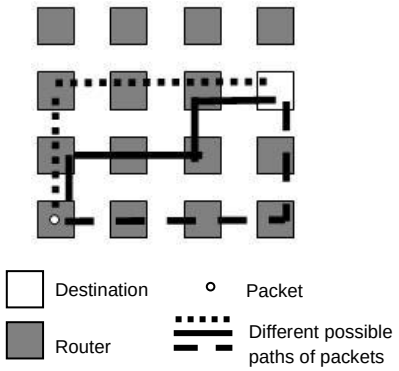


Fig. 23 Adaptive routing protocol: A message can be routed to its destination through different possible physical paths according to nodes traffic status

Among adaptive routing protocols, we distinguish fully adaptive routing protocols (FARP) and partially adaptive ones (PARP). The firsts (the fully adaptive routing protocols) allow a packet to use all possible physical paths from its source to its destination [40]. For these protocols deadlocks are avoided by the use of virtual channels (CVs). We quote as an example [40]which classifies fully adaptive routing protocols according to their number of CVs: There are those with more than 2 CVs, those with exactly 2 "CV"s and those with less than 2 CVs. The seconds, (the partially adaptive routing protocols), unlike the totally adaptive ones, don't allow all possible paths, they allow only a few of the possibilities [40]. Several examples are dealing

with this type of routing. The "Turn Model" [41], The "Even-Odd" and the "DyXY" in belong:

– The "Turn Model" avoids deadlocks without using virtual channels. This protocol dedicated for "Mesh" topology [39] is based on the analysis of directions in which packets can rotate and cycles that make up these turns [41]. One tour of each cycle will then be banned to guarantee the impossibility of deadlocks (that's why the protocol is partially adaptive). Depending on their banned tours, we have 3 "Turn Models": The "west first" (Fig. 24), the "north last"(Fig. 25) and the "negative first"(Fig. 26) [41]. According to [41], at least one quarter of the turn shall be prohibited to avoid deadlocks.

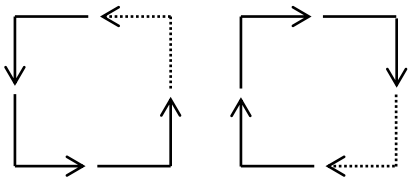


Fig. 24 One tour of each cycle is prohibited: West first [41], (Tours to the west (dotted) are prohibited)

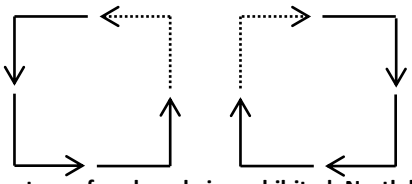


Fig. 25 One tour of each cycle is prohibited: North last [41] (Tours having the north as destination (dotted) are prohibited)

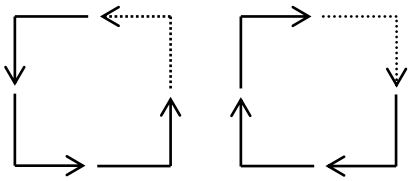


Fig. 26 One tour of each cycle is prohibited: Negative first [41] (Tours to a negative side (dotted) are prohibited. Negatives sides are south and west)

– The "Even-Odd turn model" [39] constitute also a good example of partially adaptive routing protocols. It is a model of wormhole routing algorithm without virtual channels allowing to avoid deadlocks for the "Mesh" topology. Unlike the old "turn models" that prohibit certain rotation directions, the "even -odd" applies some restrictions on where some turns can be made:

Fig. 27 Even-Odd turn model

1) (a) and (c) (Fig. 27(a), 27(c)) are prohibited on even columns

2) (b) and (d) (Fig. 27(b), 27(d)) are prohibited on odd columns

This implicates that (a) and (b) are prohibited in a same column, the same case for (c) and (d). With this, we avoid deadlocks caused by packets waiting each other in an infinite cycle.

– "DyXY" protocol [42] is also one of best known

partially adaptive routing protocols. It's a deadlock free routing protocol that adapts to congestions. This protocol is dedicated to "Mesh" topology.

• Virtual channels "CV" s: "CV"s is a widespread technique for routing protocols based on creating abstractions (the "CV"s) which share the same physical channel, as well as routers in the two proximities of this physical channel [39]. Virtual channels are first introduced to help deterministic routing protocols, after that they are generalized to adaptive routing protocols. It is valid for different NOCs topologies [39]. CV's technique has many advantages such as minimizing (up avoid) deadlocks. Additionally, adding a virtual channel to a physical one is much less expensive than adding a second physical channel [41]. However, adding a virtual channel has its drawbacks, it requires addition of buffer space and complex control logic to the two routers of proximity [41]. Adding a virtual channel to a physical one also reduces the bandwidth of virtual channels that already share the same physical channel [41].

2.6.4 "NOC"s Examples

In literature, many "NOC" examples are existing, we quote :

• Mango (Message-passing Asynchronous Network-on-chip providing Guaranteed Services over OCP interfaces) [23]: "MANGO" is a GALS (Globally Asynchronous Locally Synchronous) system in which cores reside in independent clocks areas connected by an heterogeneous "NOC" across network adapters(see figure 28). The three "MANGO" characteristics are:

- A clockless implementation.

- A guaranteed communication service.

- Standard socket access points.

• "AEthreal"[21], [43]: "AEthreal" is a "NOC" developed by "Philips" that offers a certain quality of services providing a guaranteed debit and a best effort. "AEthreal" "NOC" is very flexible in slot allocation, ports number and many other characteristics.

• SPIN (Scalable Programmable Integrated Network)[21],[37]

• "Chain" (CHip Area INterconnect) [21], [29].

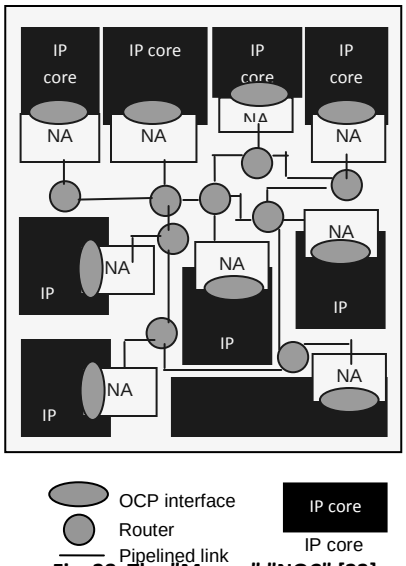


Fig. 28 The "Mango" "NOC" [23]

3. GLOBALLY ASYNCHRONEOUS LOCALLYSYNCHRONEOUS SYSTEMS

"Globally Asynchronous Locally Synchronous" systems are considered as an alternative to the limitations of previous asynchronous systems (multi-clocks) and to conventional synchronous "bus" [27]. "GALS" are based on islands (blocs) locally synchronous connected by asynchronous links [21]. For theses systems the packetization is taken by synchronous modules [21], while the sending is done through asynchronous links. The re-synchronization occurs after the receipt. "GALS" have many advantages like power consumption minimization [27], [21]: No power consumption when links are in standby. Transfers in "GALS" systems have a low latency. "Globally Asynchronous Locally Synchronous" systems favor expanded systems based on blocs of "IP"s s with different temporal characteristics [21] and reduce synchronization problems to smaller sub-problems [21]. CHAIN" [29], "Mango" [23], "ANOC", "QNOC " and "Proto" are examples of "NOC"s dedicated to "GALS" systems [21]. "GALS" announce a number of challenges like Resynchronization and possibility of introducing errors [21].

4. MPSOCS: MEMORIES ARCHITECTURES

"MPSoC"s memories architecture has a direct impact on overall system performance: In fact, while the processor frequency increases by approximately 75 percent per year, "DRAM"'s performance increases by only 7 percent [47]. If we continue on this pace, system performance will be saturated: Time spent by the processor waiting for a response from the main memory will be greater than computation time. Consequently there is real need to create and use new hierarchical memories organizations in "MPSOC"s.

4.1 Types of memories

There are many types of memories which can be used in embedded multi processors systems. We quote:

- Scratch pads memories: "SPM"s
- Caches memories
- Externals memories

4.1.1 Scratch Pad Memories (SPMs)

SPMs: "Scratch Pad memories" are small "SRAM" on chip data memories characterized by their speed. With only one processor clock cycle latency [46], "SPM"s can contribute to resolve cache conflicts [46]. In [46], "Panda et al." have presented a technique to efficiently exploit "SPM"s by dividing up scalar variables and arrays between off-chip "DRAM"s and on-chip "SRAM"s "SPM"s to minimize the total execution time of embedded applications. Advantages of "Scratch Pad Memories" are many: As all other memories internals to the chip, "SPM"s are characterized by their low consumption, low latency, small number of pins and high debit[49].

4.1.2 Caches Memories [46]

"Caches" are fast local Memories with only one processor clock cycle latency [46]. They represent on chip interfaces between processors and external memories [46].

4.1.3 SPMs Vs Caches

The comparable utility between Caches and "SPM"s push us to do this little comparison between the two types of memories which proves that there are some differences between them. Indeed, caches are using a "HW" controller to decide what data to put on and what data to fetch. In contrary, "SPM"s doesn't need any "HW" as they are controlled by "SW". In the other hand, according to [48], "SPM" s are more efficient than caches in terms of energy consumption, because they are not using a "HW" controllers. So it is better to use this memory type especially for systems that have regular memory access speed that can be predicted and analyzed statically at compile time.

4.1.4 External Memories (Off Chip)

External memories are "DRAM"s. Access to theses memories take many processor clock cycles (typically between 10 and 20). Example in [49], "Gossens et al." have referred to an implementation of a "SOC" dedicated to video decoding : "VIPER". The "SOC" requires a large memory space, so an internal memory is insufficient and it is more appropriate to gather data and instructions in a single external memory. To hide the latencies of the two "Viper" processors (which are "MIPS" and Trimedia), the authors use two caches, one for instructions and the other for data.

4.2 Memories strategies for MPSOCs

4.2.1 Shared Memory Systems

Shared memories are the most known and reliable mean for exchanging information between the computational units. They also represent typically the fastest asynchronous way of communication. This way of communication is dedicated to large data transfers. According to [1], a shared memory system has the following properties:

• Each processor can address directly all the memory space (data are seen by all processors).

• Shared memory access is synchronized with hardware as well as software protocols between processors; this is done via some instructions like leoad/store.

• Data memory space is transparent to the programmer who has to define the memory regions which will be shared and those which will not be shared. According to [3], communication based on shared memory is used in many application fields like signal processing, data and image processing, multimedia and other applications demanding a large size of data.

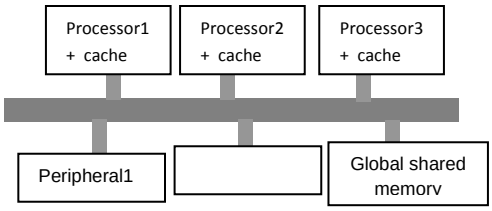


Fig. 29 Global shared memory architecture

It is also possible to share:

• Internal ”BRAM”s

• DMA dedicated for important data transferring

As well as memories, we can also share (see Fig. 30)

• ”Mailbox”s

• ”Mutex”s

• PLBv46 bridges

4.2.2 Distributed Memory Systems

The distributed memory architectures are constituted by many independent blocs. Each one has its own processor as well as its local memory and its own Input/Output module. Programming into this architecture is particularly complex, as programmer should take into account the distribution of the data between different processors. The “message passing via mailboxes” communication technique is the one that should be used in this case. In this technique, the communicating blocs exchange messages which should be received in a synchronous or in an asynchronous way. Example ”Xilinx” offers an inter processes communication peripheral named ”OPB-Mmailbox” to ensure this kind of communication.

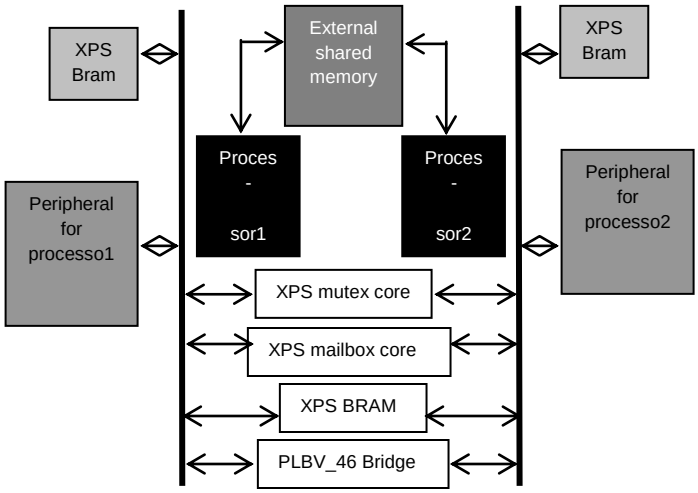


Fig. 30 Generic architecture with 2 processors (many shared resources: Bram, bridge, mailbox, memories)

5. CONCLUSIONS

In this paper we have presented a generic study on different “MPSOC”s aspects. First of all, we have talked about topologies and communications inside the chip. In this first section we have presented different types of interconnections, from the very old (the not communicating processors, the point to point communications or the ”bus”es), to the most recent ones (the “NOC”s). All this, with many industrial examples of each type of communications, especially for “bus”es and “NOC”s. In the second part of this survey, a brief introduction to ”Globally Asynchronous Locally Synchronous” systems was done, as this type of systems is very used for ”MPSOC”s researches and industrial world examples. Finally, a view on different memory organizations of ”MPSOC”s has been presented, since memories represent a real challenge for futures ”MPSOC”s. After this study, future works can be devoted to give some solutions to actual existing problems. Among these problems we quote memory organization strategies which present a big challenge, face to the huge increasing of the computing capacities which hasn’t been followed by an equivalent amelioration in memories latencies.

6. REFERENCES

[1] V. Asokan, ”Designing Multiprocessor Systems in Platform Studio”, White Paper: Xilinx Platform Studio (XPS), 2007

[2] OCP-IP, System-on-Chip (SoC) design, http://www.ocpip.org/

[3] S. Meftali, ”An Optimal Memory Allocation for Application-Specific Multiprocessor System-on-Chip” System Synthesis, 2001. Proceedings: The 14th International Symposium On page(s): 19 -24, 2001

[4] IBM, ”The CoreConnect Bus Architecture” available at https://www01.ibm.com/chips/techlib/techlib.nsf/techdocs/852569B20050FF7785256 991004DB5D9

[5] ARM, ”AMBA Specification Rev 2.0”, available at http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ihi0011a/index.html1

[6] Sonics, SoC design, http://www.sonicsinc.com/

[7] M.Moadeli ”An Analytical Performance Model for the Spidergon NoC”, 21st International Conference on Advanced Networking and Applications, 2007

[8] S. Pasrisha et al. ”Constraint-Driven Bus Matrix Synthesis for MPSoC”, Proceedings of the 2006 Asia and South Pacific Design Automation Conference Yokohama, Japan SESSION: Interconnect for high-end SoC, Pages: 30 - 35, 2006

[9] H.Lee et al., ”On-Chip Communication Architecture Exploration: A Quantitative Evaluation of Point-to-Point, Bus, and Network-on-Chip Approaches”, ACM Transactions on Design Automation of Electronic Systems,Vol. 12,No. 3, Article 23, 2007

[10] K. Rompaey et al. ”CoWare - A design environment for heterogeneous hardware/software systems”, Proceedings of the conference on European design automation, Geneva, Switzerland Pages: 252 - 257,1996

[11] L.Bennini et al. ”Networks on chips: A new paradigm for componentbased MPSoC design”, in A. Jerrraya and W. Wolf Editors, Multiprocessors Systems on Chips, Morgan Kaufman, pp. 49-80 , 2004

[12] P. Aldworth, ”System-on-a-Chip Bus Architecture for Embedded Applications”, IEEE International Conferenceon Computer Design, pp. 297- 298, 1999.

[13] S.Na et al. ”Low-Power Bus Architecture Composition for AMBA AXI”, journal of semiconductor technology ans sciences, VOL.9, NO.2, June, 2009

[14] Scandurra et al., ”STBus communication system: Concepts and sefinitions”, Reference Guide, STMicroelectronics.(stbus,2003)

[15] M.Loghi et al. ”Analyzing On-Chip Communication in a MPSoC Environment”, Proceedings of the conference on Design, automation and test in Europe - Volume 2, Page: 20752, 2004

[16] S.Murali et al. ”An Application-Specific Design Methodology for On-Chip Crossbar Generation”, Computer-Aided Design of Integrated Circuitsand Systems, IEEE Transactions, Volume: 26 Issue: 7, on page(s): 1283 - 1296, Sonoma, CA, USA July 2007

[17] ”WISHBONE System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores”, rev B.4, 2010 available at http://cdn.opencores.org/downloads/wbspec

[18] M. Mitic et al. “An overview of on chip buses”, FACTA Universitatis Series: Electronics and Energetics, vol.19, no.3, pp 405-428, Dec 2006

[19] “Overview of the coreframe architecture”. Palmchip Corporation. [Online]. Available: http://www.palmchip.com

[20] F.Angiolini et al. ”Contrasting a NoC and a Traditional Interconnect Fabric with Layout Awareness”, Proceedings of the conference on Design, automation and test in Europe, Munich, Germany, SESSION: Application-specific network on chip design Pages: 124 - 129, 2006

[21] T. Bjerregaard et al. ”A Survey of Research and Practices of Networkon- Chip ”, ACM Computing Surveys, Vol. 38, March 2006

[22] A. Radulescu et al. ”An Efficient On-Chip Network Interface Offering Guaranteed Services, Shared-Memory Abstraction, and Flexible Network Configuration”, Proceedings of the conference on Design, automation and test in Europe - Volume 2, Page: 20878, 2004

[23] T. Bjerregaard et al. ” An OCP Compliant Network Adapter for GALS-based SoC Design Using the MANGO Network-on-Chip”, ConferenceProceedings of the International Symposium on System-on-Chip (SoC’05), pp. 171-174, November 2005

[24] F.Karim et al. ”An interconnect architecture for networking systems on chips”, Micro IEEE, Volume: 22, Issue: 5, On page(s): 36 - 45, Sep/Oct 2002

[25] P.Pande et al. ”Effect of traffic localization on energy dissipation in NoCbased interconnect. In International Symposium on Circuits and Systems “(ISCAS).IEEE, 1774-1777, 2005

[26] N.Concer et al. ”aEqualized: a Novel Routing Algorithm For The Spidergon Network On Chip”, Design, Automation and Test in Europe Conference and Exhibition, 2009. DATE ’09, On page(s): 749 - 754, Nice, 20-24 April 2009

[27] A.Zitouni, “A Generic and Extensible Spidergon NoC ”,World academy of science, engineering and technonogy, issue 31, july 2007

[28] L.Bononi, ”Simulation and Analysis of Network on Chip Architectures: Ring, Spidergon and 2D Mesh”, Proceedings of the conference on Design, automation and test in

Europe: Designers' forum, Munich, Germany, SESSION: On chip communication networks, Pages: 154 -159 ,2006

[29] J.Bainbridge et al. "CHAIN: A delay-insensitive chip area interconnect", Micro, IEEE, Volume: 22, Issue: 5, On page(s): 16 - 23, Sep/Oct 2002

[30] K.Srinivasan et al. "An Automated Technique for Topology and Route Generation of Application Specific On-Chip Interconnection Networks", Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design, San Jose, CA, Pages: 231 - 237 2005

[31] P. Meloni etal. "Routing Aware Switch Hardware Customization for Networks on Chips", Nano-Networks 2006, Lausanne, Switzerland, September 14-16, 2006

[32] D.Bertozzi et al. "NoC Synthesis Flow for Customized Domain Specific Multiprocessor Systems-on-Chip", IEEE transactions on parallel and distributed systems, vol.16, No.2, february 2005

[33] M.Jamali et al. "MinRoot and CMesh: Interconnection Architectures for Network-on-Chip Systems" , World Academy of Science, Engineering and Technology 54 2009

[34] M.Aghatabar et al. "An Empirical Investigation of Mesh and Torus NoC Topologies Under Different Routing Algorithms and Traffic Models", Digital System Design Architectures, Methods and Tools, 2007. DSD. 2007. 10th Euromicro Conference, On page(s): 19 - 26, Lubeck, 29-31 Aug. 2007

[35] P.Pande et al. "Evaluation of MP-SoC Interconnect Architectures: a Case Study", Proceedings of 4th IWSOC, Banff, Alberta, Canada, 19th-21st July, 2004

[36] P.Pande et al. "Design of a switch for network on chip applications", Proceedings of the 2003 IEEE International Symposium on Circuits and Systems (Cat. No.03CH37430), IEEE, Volume vol.5, Bangkok, Thailand, p.217-20, 2003

[37] A.Adriahantenaina et al. "SPIN: a Scalable, Packet Switched, Onchip Micro-network", Proceedings of the Design,Automation and Test in Europe Conference and Exhibition (DATE'03), Munich, Germany, March 2003 [38] CH.Leiserson, "Fat-trees: Universal networks for hardware efficient supercomputing", IEEE transaction on computers, volC-34 N10, October 1985

[39] G.Chiu "The Odd-Even Turn Model for Adaptive Routing", transactions on parallel and distributed systems, VOL. 11, NO. 7, JULY 2000

[40] P.Mohapatra, "Wormhole Routing Techniques for Directly Connected Multicomputer Systems", ACM Computing Surveys, Vol. 30, No. 3, September 1998

[41] Ch.Glass et al. "The Turn Model for Adaptive Routing", ACM SIGARCH Computer Architecture News, Volume 20 , Issue 2 (May 1992), Special Issue: Proceedings of the 19th annual international symposium on Computer architecture (ISCA '92), Pages: 278 - 287, 1992

[42] M.Li et al. "DyXY - A Proximity Congestion-Aware Deadlock-Free Dynamic Routing Method for Network on Chip", DAC 2006, July 24- 28, 2006, San Francisco, California, USA, 2006

[43] K. Goossens et al. "AEthereal Network on Chip: Concepts, Architectures, and Implementations", Copublished by the IEEE CS and the IEEE CASS IEEE Design and Test of Computers, 2005

[44] OpenCores, Open source hardware IP-cores, http://opencores.org/

[45] OpenCores, SoC Interconnection: Wishbone http://opencores.org/opencores,wishbone

[46] P.R.Panda "Efficient Utilization of Scratch-Pad Memory in Embedded Processor Applications", European Design and Test Conference, Proceedings of the 1997 European conference on Design and Test, Page: 7, 1997

[47] S.A.MCKEE, "Reflections on the Memory Wall". In Proc. of the Conference On Computing Frontiers, 2004

[48] I.Issinen et al. "Multiprocessor System-on-Chip Data Reuse Analysis for Exploring Customized Memory Hierarchies" DAC 2006, July 24-28, 2006, San Francisco, California, USA

[49] K.Goossens et al "Interconnect and memory organization in SOCs for advanced set-top boxes and TV" chapter 15 of the book "Interconnect- Centric Design for Advanced SOC and NOC " of J. Nurmi et al., 2004, VII, 453 p., Hardcov