A Multi Level Functional Verification of Multistage Interconnection Network for MPSOC

Yassine AYDI, Ramzi TLIGUE, Maïssa ELLEUCH, Mohamed ABID
CES Laboratory
National Engineering School of Sfax
Email: yassine.aydi@oous.rnu.tn, ramzitligue@ieee.org, mohamed.abid@enis.rnu.tn

Jean-Luc DEKEYSER
LIFL and INRIA-Futurs,
University of Lille, France
Email: dekeyser@lifl.fr

***Abstract*— Network on chip (NOC) has emerged as a promising alternative to ensure communication for Multiprocessor systems on chip (MPSoC). This paper proposes a hybrid verification approach of Delta multistage interconnection networks for MPSoC. At the generic level, we propose a formal specification of the network in the ACL2 theorem proving environment. We will ensure the soundness of our verification approach by using programmable logic circuits for fast performance verification of Delta MIN. We thus show the utility of the hybrid approach to give a more realistic model describing the communication architectures.**

I. Introduction

Modern embedded systems integrate a potentially large number of applications or functions in a single chip. An increasing number of processors and data memory units are being integrated into a single chip to build Multi-Processor Systems on Chip (MPSoC) [1]. Therefore, researches were focused mainly in squeezing computing and controlling power on embedded systems. As a result, many MPSOC platforms have emerged [2].

Nevertheless, one of the most critical area of MPSoC design is the choice of the suitable interconnect platform. Indeed, this communication architecture must support the entire inter-component data traffic and has a significant impact on the overall system performance [3]. As a promising alternative, Networks on Chip (NoC) have been proposed by academia and industry to handle communication needs for the future multiprocessor systems-on-chip [4].

Multistage Interconnection Network (MIN) has been used in classical multiprocessor systems. As an example, MINs are frequently used to connect the nodes of IBMSP [5] and CRAY Y-MP series [6]. Further on, MINs are applied for networks on chip to connect processors to memory modules in MPSOC. A MIN is defined by its topology, switching strategy, routing algorithm, scheduling mechanism, fault tolerance, and dynamic reconfigurability [7].

An essential step in the design of an MPSoC is the verification of the whole system, and especially of the selected communication architecture. Traditionally, this verification is synonym with simulation which consists on the performance evaluation of the system [8]. However, such technique provides partial verification, so it cannot cover all design errors or detect undesirable situations (deadlock, starvation). The trend is then to adopt formal verification, which is based on using methods of mathematical proof to ensure the quality of the design, improve the robustness of the system, and speed up the development [9]. To compare and contrast different communication architectures, a standard set of performance metrics should be evaluated, such as area, energy consumption, execution time and latency. Therefore, the easy programmability and the large integration capacity of FPGA provide a faster performance evaluation through emulation which complements the formal verification process of the communication architecture.

A hybrid approach for functional verification of Delta multistage interconnection networks for MPSoC is investigated in this paper. Section 2 introduces MIN architecture. Next, a formal approach to specify Delta-MIN based on-chip communications is detailed. Section 4 describes Delta networks model implemented on FPGA. Finally, we conclude the paper and we give directions for future work.

II. MIN ARCHITECTURE

In this section, we present an overview of the networks used for the specification and the verification of the interconnection platform for MPSOC.

A. *MIN Components*

The common multistage interconnection networks (MINs) used, have N inputs and N outputs nodes and are built using $r \times r$ switches. Such MINs have N/r switches at each stage, and $log_r N$ stages of switches denoted d. The different stages are connected by links generated by applying permutation functions. In a MIN, a path between a source and a target is obtained by operating each corresponding switch of the stage i in straight mode if the i^{th} bit of the destination address is equal to 1, otherwise in exchange mode.

B. *MIN with Banyan property*

Banyan MIN is a multistage interconnection network characterized by one and only one path between each source and destination. A banyan MIN of size $N \times N$ consists of $r \times r$ crossbars. An interesting subclass of Banyan MINs is composed of Delta networks. Let denote by: o_i the i^{th} output of a crossbar in a MIN, and by C_j, a crossbar belonging to the stage j. So, the Delta property can be defined as follows: if an input of C_j is connected to the output o_i of C_{j-1}, then all other inputs of C_j must be connected to the stage $(j-1)$ on outputs with the same index i.

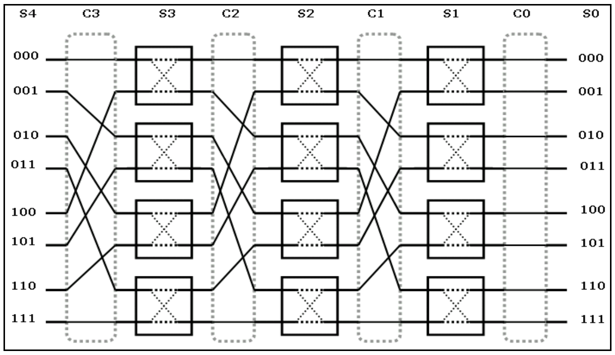

Fig.1. A Delta network (8,2)

The difference between each of the existing MINs is the topology of interconnection links between the crossbar stages. A study of equivalence of a variety of Delta MINs has been detailed in [10]. An example of Delta networks (a subclass of MINs) illustrated in figure 1 connects 8 processors to 8 memories by means of 3 stages of 4 switches each. Processors and memories are represented by 3-bit number $(d_2\ d_1\ d_0)_2$. The interconnection stages denoted C_i $(0 \leq i \leq 3)$ are generated by applying permutation functions.

III. Specification and verification of a Delta-MIN based architecture

We describe below the methodology adopted to specify in formal notations the Delta MIN network. We detail a generic topology and extended routing components as extension of the Generic Networks on Chip model denoted GeNoC [11]. This model takes into account the common components of any on-chip interconnection architecture, and models them in a functional style through four functions: "Send", "Recv", "Routing" and "Scheduling". The GeNoC model has been implemented in the ACL2 theorem proving environment [12].

A. The Delta MIN topology component

The Delta MIN topology as described above (fig.1) is composed of nodes and connections.

-***The set of nodes****:* a pair of coordinates ***(x y)*** is used to represent a node in a Delta MIN. The coordinate x is decimal. It represents the stage of nodes to which belongs the node. The Y coordinate is binary and it describes the position of the node within the same stage. The function ***gen-nodes-dmin*** generates all nodes of the network. It takes as parameters N, the size of the network, and r (r=2), the degree of switches. The validity of these parameters is recognized by the predicate ***ValidParamsp-dmin***. We define also another predicate called ***dmin-nodesetp*** for the whole nodes validity. The nodes set generation is constrained by the theorem.1.

Theorem.1 Nodes set generation

```
(defthm gen-nodes-dmin-correct
      (implies (ValideParams-dmin pms)
      (dmin-nodesetp(gen-nodes-dmin pms))))
```

-***Connections****:* we represent a connection cnx in a Delta MIN by a list ***((x px) (y py))***, where x is the origin of cnx, px is the port involved in cnx, y is the second extremity and py is the port of y. For example, the connection (((3) (0 1)) L0) (((2) (1 0)) R0) denotes that the port L0 of the switch ((3) (0 1)) is connected to the port R0 of ((2) (1 0)). In the case of Delta MIN, the connection functions are always a list of three permutations to apply respectively on the first stage of connection, the middle stages and finally, on the last stage. In the ACL2 logic, we define the function ***gen-cnx-node*** that generates all connections of one node n. It takes as arguments the node n origin of connections, the list of permutation functions, the parameter d denoting the stages number of the network, and r the degree of the switches. The theorem.2 checks that every node ext2 produced by the permutation function σ_{n-1} (modelled by sigmak) belongs to the set of nodes (nodes). The same constraint must be also verified for the other two permutation functions. We define below the function ***gen-top-dmin*** (definition.1). It generates all the connections of a Delta MIN by taking as inputs N and r previously defined, and the type of the Delta MIN. The last parameter is used by ***gen-topology*** to select the types of permutations corresponding to this network.

Definition.1. Generation of Delta MIN topology

```
(defun gen-top-dmin (pms-t)
 (let* ((x1 (car pms-t))(x2 (cadr pms-t))
       (x3 (caddr pms-t))
       (S (car x1))
       (pms-s (cadr x1))(fp-s (caddr x1))
       (Sw (car x2))(pms-sw (cadr x2))
       (fp-sw (caddr x2))
       (D (car x3))(pms-d (cadr x3))
       (fp-d (caddr x3))
     ;;(S pms-s fps)
  (top-S (gen-top-dmin-src_sw S pms-s fp-s))
    ;; (Sw pms-sw fpsw)
  (top-Sw (gen-top-dmin-sw_sw Sw pms-sw fp-sw))
             ;;(D pms-d fpd)
  (top-D (gen-top-dmin-sw_dest D pms-d fp-d)))
  (append top-S (append top-Sw top-D))))
```

Theorem.2

```
(defthm ext2-gen-one-cnx-in-nodes
  (let* ((nodes (gen-nodes-dmin pms))
       (c (gen-one-cnx x i fp))
       (ext2 (y-node c)))
  (implies (and (ValidParamspD pms)
            (Validfp-dmin fp)
            (member-equal x nodes)
            (valid-node-dmin ext2))
        (member-equal ext2 nodes)))
:hints (("GOAL" :in-theory
             (disable gen-nodes-inv1))))
```

Finally, theorem.3 proves the constraint that expresses the validity of the Delta-MIN topology and check the compliance of the definitions with the generic topology component extended.

Theorem.3

```
 (defthm valid-gen-top-dmin
   (let* ((pms-t (params-top-t pms))
         (top (gen-top-dmin pms-t)))
(implies (and (ValidParamspD pms)
           (valid-params-t pms-t))
           (valid-top-dmin top)))
 :hints (("GOAL" :in-theory (disable Validfp-dmin GEN-NODES-DMIN-S-
pms gen-nodes-dmin-sw-1-pms GEN-NODES-DMIN-d-pms ))))
```

B. *The Delta MIN routing component*

The routing algorithm used in Delta MINs is the self routing. It depends only on the destination address, called also control sequence. If the corresponding digit of the control sequence is equal to *i*, the message to deliver will be switched to the output *i* of the current crossbar. Here, the routing algorithm must take into account connections. Indeed, the only information of the port through which the message must be switched is not enough. Thus, we must look in the topology for the connection with the current switch as origin. As defined in ACL2, the routing function ***routing-dmin*** takes as arguments the list of missives to be routed through the Delta MIN, and the parameters to generate the whole topology. For each missive, ***routing-dmin calls*** the following function *compute-rte* (definition.2) to compute the route between the origin (*from*) and the destination (*to*).

Definition.2 Function compute-rte.

```
(defun compute-routes-dmin (from to cdrto top)
(if (endp cdrto)
    nil
(let*((bit_rtg (car cdrto))
      (from-a (adapt-node from bit_rtg))
      (next-node (ext2 (rech-top from-a top))))
(cond
;; destination bit equals 0
((equal bit '0)
 (list* (list from-a next-node)
 (compute-routes-dmin next-node to (cdr cdrto) top)))
;; destination bit equals 1
((equal bit '1)
 (list* (list from-a next-node)
 (compute-routes-dmin next-node to (cdr cdrto) top))))))
```

The ACL2 theorem proving environment provides also an execution engine. Thus, we can simulate the execution of the definitions. We present below a simulation of the function ***routing-dmin*** showing the progression of a list of missives (table 1) through an omega network 8x8, using 2x2 crossbars.

TABLE I. THE LIST OF MISSIVES

id	*origin*	*content*	*destination*
1	((4) (0 0 1))	frm1	((0) (1 0 0))
2	((4) (1 0 1))	frm2	((0) (0 0 1))

The simulation result of the missive number 1 is shown below. We can notice that the routing algorithm make use of connections like ((((3) (0 1)) O1) (((2) (1 1)) I0)), to compute a route.

TABLE II. SIMULATION RESULTS

Id	1	2
C3	((((4) (001)) L) (((3) (01)) I0))	((((4) (110)) O) (((3) (10)) I1))
C2	((((3) (01)) O1) (((2) (11)) I0))	((((3) (10)) O1) (((2) (01)) I1))
C1	((((2) (11)) O0) (((1) (10)) I1))	((((2) (01)) O1) (((1) (11)) I0))
C0	((((1) (10)) O0) (((0) (100)) I))	((((1) (11)) O0) (((0) (111)) L))

In this section, we have proposed an approach to specify and verify the Delta multistage interconnection networks by identifying inherent properties of all topologies and connections. These properties, which are called also constraints, have been validated using the ACL2 theorem proving environment. To achieve the routing extension, we have formalized the general common relation between topology and routing.

IV. DESIGN AND PERFORMANCE ANALYSIS OF A DELTA-MIN BASED ARCHITECTURE ON FPGA

The main idea in this work is to combine hybrid verification techniques by enhancing coverage of the state space traversed. Therefore, we proposed in this section a design of a Delta multistage interconnection network for MPSoC architecture on FPGA which provides fast verification through emulation by evaluating a set of performance metrics.

A. *Design of Delta-MIN on FPGA*

The configurable Delta MIN provides support for a variety of network topologies which play an important role in designing routing strategy, network latency, throughput and area. We will restrict study to Delta MINs networks (fig.1). A generic connection block is developed to involve various MINs topology by switching links between the crossbar stages.

- Data exchanged: Traffic passed through the networks composed of fixed size packets. The packet format has three parts (fig.2).

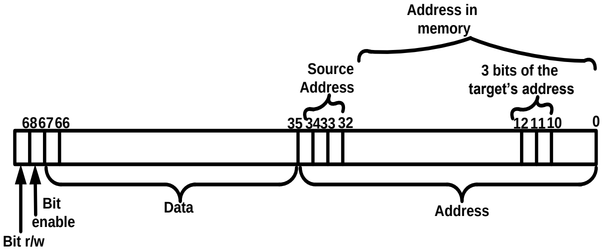

Fig.2. Packet format

- Router architecture: The router is composed of 2x2 crossbars, a control component (Scheduler), a couple of input and output ports (fig.3). Each input port of the router has dedicated buffer storage. Packets are buffered in input port until the output port of the next stage is ready to accept the packets; the width of each buffer is equal to the packet length in order to facilitate the routing strategy.

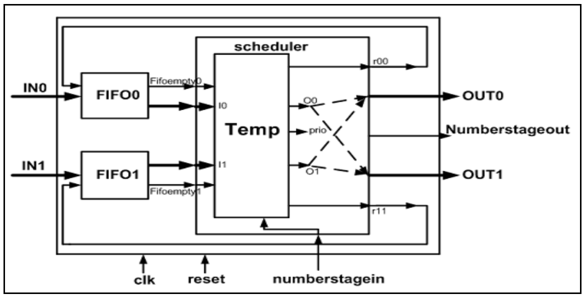

Fig.3. Router architecture

-MIN implementation: A model of a MIN for N processors and N memory has been implemented on a prototyping platform *Xilinx virtex4*. The following figure represents an example of an omega MIN (8, 2) described at RTL level.

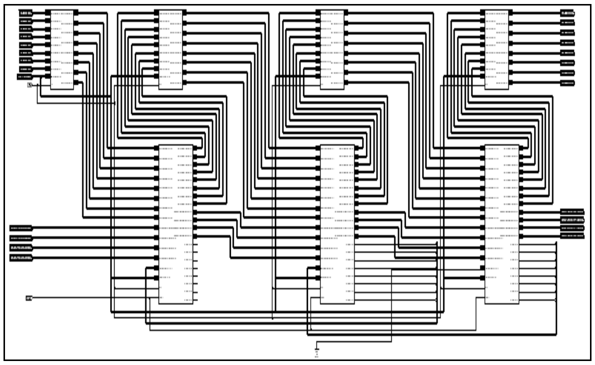

Fig.4. An Omega network (8,2)

B. Simulation results

This section details the implementation results of the configurable Delta MIN as well as complete MIN-based multiprocessor system. Our platform performs communication to N Mini-MIPS processors and N sharing data memories. The utility of the Delta MIN is shown here using a Finite impulse response (*FIR*) filter application. In order to demonstrate the effectiveness of the proposed design methodology, the application is parallelized on MPSoC architecture with 4 to 32 processors.

TABLE III. RESOURCES UTILIZATION FOR N*N NODES

Logic Utilization								
N	Number of Slices	%	Number of Slice Flip Flops	%	Number of 4inputLUTs	%	Number of FIFO16/ RAMB16s	%
4	2281	2	2364	1	3993	2	16	4
8	6839	7	7092	3	12257	6	48	14
16	18220	20	18912	10	33057	18	128	38
32	37566	42	48560	27	66657	37	320	95

Table III details the resource utilizations for N*N nodes implementations on a prototyping platform Xilinx Virtex4. Based on these results, we find that area on FPGA is increasing while increasing the size of the network.

V. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a hybrid approach for functional verification of multistage interconnection network for MPSOC. Our objective in this work is to use several verification techniques to validate selected communication architecture in different level abstraction. The framework presented in this paper opens promising trend for further development as complement to traditional verification techniques. We plan to extend this work to employ formal notations to validate the implementation of the communication architecture based on multistage interconnection network.

VI. REFERENCES

[1] A. Jerraya, and W. Wolf, “Multiprocessor Systems-on-Chips”, Morgan Kaufmann Publishers, San Francisco, 2004.

[2] W. Wolf, “The future of multiprocessor systems-on-chips”, Proc. of the 41st annual conference on Design automation, ACM Press, New York, 2004, pp. 681–685.

[3] S. Pasricha, “Floorplan-aware automated synthesis of bus-based communication architectures”, Proc. of the 42nd annual conference on Design automation, ACM Press, New York, 2005, pp. 565–570.

[4] L. Benini, “Networks on chips: A new SoC paradigm”, Computer, Vol. 35, N° 1, IEEE Computer Society Press, Los Alamitos, California, 2002, pp. 70–78.

[5] C. B. Stunkel, “The SP2 High-Performance Switch”, IBM Systems Journal, Vol. 34, N° 2, IBM Corp., Riverton, USA, 1995, pp. 185–204.

[6] T. Cheung, “A simulation study of the CRAY X-MP memory system”, IEEE Transactions on Computers, Vol. 35, N° 7, IEEE Computer Society, Washington, 1986, pp. 613–622.

[7] Y. Aydi “Design and Performance Evaluation of a Reconfigurable Delta MIN for MPSOC”, In 19th International Conference on Microelectronics (ICM ’07), 2007.

[8] Y. Aydi, “Dynamicity Analysis of Delta MINs for MPSOC Architectures”, STA'07, 2007.

[9] M. Elleuch., “Formal Specification of Delta MINs for MPSOC in the ACL2 Logic”, in Proceedings of Forum on Design and specification Languages - FDL ’08, 2008.

[10] C. Kruskal, “A unified theory of interconnection network”, Theoretical Computer Science, Vol.48, N°1, Elsevier Science Publishers Ltd., essex, 1986, pp. 75-94.

[11] J. Schmaltz, and D. Borrione, “Towards a Formal Theory of Communication Architecture in the ACL2 Logic”, Proc. of 6th international workshop on the ACL2 theorem prover and its applications, ACM Press, New York, 2006, pp. 47- 56.

[12] M. Kaufmann, and J S. Moore, “ACL2: An industrial strength version of nqthm”, IEEE Transactions on Software Engineering, Vol. 23, N°4, IEEE Press, New York, 1996, pp. 23-34.