Based B Extraction of QNoC architecture properties

Abdelhamid Hariche*, Mostefa Belarbi*, Hayat Daoud [+]
Ibn Khaldoun University, Computing sciences department
LIM , Tiaret, Algeria
*haricheabdelhamid@gmail.com **mbelarbi@mail.univ-tiaret.dz [+]hayat.daoud@hotmail.fr

***Abstract*—Embedded systems become more and more present in our daily life. The validation of this kind of systems with time their evolution became fast and complex we focus on multiprocessor systems on chip (MPSoC) and exactly Qaulity of Service of Network on chip architecture (QNoC). The interconnection of communication modules (IP - Intellectual Property) constitutes a fundamental part during the design of such systems expressed in terms on band-width, latency, power consumption and reliability. The validation currently for MPSoC (with QNoC's basis) based on the logical simulation which it can't allow a global validation for this system even it is not adapted for the design of high level integration complex systems (Handicaps with respect to the concept *time to market*).**

The new validation approach using the formal technics using B event method consists of suggesting aspects and constraints related to the reliability of NoC and the over-cost related to the solutions of tolerances on the faults (a design of NoC tolerating on the faults for SoC containing configurable technology FPGA) by extracting the properties of the QNoC architecture existed in the VHDL code associated and prove these last using the prover that used in B event method. This approach makes it possible to exploit.

***Keyword*- MPSoC, Formal Technics, Generating Model, QNOC, VHDL, B Event, incrementale**

I. INTRODUCTION

Systems integrated on chips (or Systems On Chips) are more and more complex and integrate an increasing number of processing units to answer needs of new applications. This growth entails an increase of communication needs in circuits for exchanges of data and for the processing control. The evolution of the silicon technologies makes possible this density of integration. With scales sub-Micronics news constraints appear however. The interconnections cost becomes greater to the one used in the logic. Communications become major goal for performances of systems.

Having made this report, different groups of research have been brought to propose a new paradigm: the network on chip (NoC). Because of the complexity of several applications and the integration, inventors get on more and more resource of calculation (i. e. IPs) in a system on chip. However, it returns the manufacture test of these systems more difficult, notably systems on chip based on networks on chip QNoC. To perform the routing communication, we must use several routing algorithms **[01]**, but this can make some main problems:
the failure routing decision errors and the dropped node on the way problem which yielded the failing of the material.**[02][03]**
Using the embedded systems concept, we can process microelectronic aspects of integrated circuits on the SOC, we propose in this document to develop a methodology that used the B method for the verification of the system on chip field. The objective is to propose a formal model **[04]**with the help of the language B-event to verify the network on chip architecture (QNoC), and the safety working of this last, and That is innovating these systems on chips.

There are research works which allow to test of NoC [**05**] using declarative assertions to specify expected functional and temporal properties of modules and\or their environment by the recognizing what's valuable such as the constraint for the correctly-use of a node (or IP) and it delivered result even the correct behavior from the design. PSL is a formalism ease writing temporal and logical properties, the online embedded testing using technique for synthesis from the assertion and properties for a monitor, this work use the notion of monitor to pinpoint erroneous transactions between modules that belong to different clock domains concept suggested which is coded using VHDL or Verilog as language.

In some researches**[06]** they think for avoiding the risk of functionality they need a specification analysis and modeling techniques in software community just like SLOOP or System Level design with Object Oriented Process .SLOOP employs four UML (Unified Modeling Language) models which detailing those three aspects the target system functionality , structure and timing . Each model is used to develop a system before software and hardware implementation. Conceptual Model is the result of the analysis of requirements to avoid non-functional constraint for a costumer. Functional model for the representation of structure of function and the task of level of parallelism carrying of computing workload and communication workload. Architectural model represent the physical resources

of architecture it consist the processing resources and communication resources also it parameterize each resource using the concept of class. Performances model which maps process of functional model onto processing resources of architectural model, this model valuates the performances of the selected architecture using statistics, it helps the designer to improve the system which satisfies the requirements of the design.

Some solutions [**07**] that used formal technology which extract properties from the existed coded need five steps, the first step is to run automatic formal to explore the reachability of the design, the next step is to run simulation to capture code coverage results, the third step in the flow is to run automatic formal on the design with the merged simulation-coverage database to make the modification of the design more easily verifiable, the final step in the flow is to prune out the coverage goals proven unreachable in Step 3 from the simulation database and generate accurate code coverage metrics that reflect what is truly reachable in the coverage model of the design.

There are a several solutions to verify any NoC design but all the method had to make in the final way an implemented code with the VHDL language and here the point which our work will start to translate the VHDL code to B event model to ease the formal checking for this design [**08]**.

I. Based B method

B is a method for specifying, designing and coding software systems. It is based on Zermelo-Fraenkel set theory with the axiom of choice, the concept of generalized substitution and on structuring mechanisms (machine, refinement, implementation). The concept of refinement is the key notion for developing B models of (software) systems in an incremental way. B models are accompanied by mathematical proofs that justify them. Proofs of B models convince the user (designer or specifier) that the (software) system is effectively correct. We provide a survey of the underlying logic of the B method and the semantic concepts related to the B method; we detail the B development process partially supported by the mechanical engine of the prover.

1. B event method

B event is the extension of B method **[09]** for the design of different systems with the replacing of concepts for machines and operations by the models and events. These models can be seeing as a package for modeling the environment of any modeling systems which is evaluating its state using events. These last are specify by the use of guarded actions which activated when their guard were true . In this case each event had a function to act this notion is inspired from the guarded command of Dijkstra[**10]** or the action for Back**[11]**.

Every event for B event is representing by a state predicate called guard, a substitution for the goal of modify the values of variables of the system. For a given state many system guards can be true, one of them can be triggered when its substitution can be carry out. B event models are composing of two based constructors Contexts and machines, the first represent the static part of model when the second is for the dynamic one.

2. Context structure

The Context contain many clauses introduced by a specific keywords as they are shown within Rodin platform, some clauses are introducing with modeling elements with labels (theorem axioms) which is generated automatically in the Rodin platform, such as “Sets” which defines the carrier set of the Context, “Constants” is the list of various constants introduced in the context, “Axioms” lists of the various predicates which will be present as hypotheses in all the proof obligations, “Theorems” lists of theorem which have to be proved within the context, “Extend” defines if a context is the extension of another (if exist) **[11]**.

3. Machine structure

As the same of context, the machine had a specific keywords with labels introducing and automatically generated in the Rodin platform; “Refines” contains (if any) the machine which this machine refines; “Sees” list of contexts referenced by the machine, “Variables” lists the various variables introduced in a machine, “Invariants” the list of predicates which the variables must obey, “Events” lists various event in a machine(and they had a predefined syntax on the Rodin platform)**[12]**.

4. Proof obligation

The proof obligations define what is to be proved for an Event-B model. They are automatically generated by rodin platform tool called the proof obligation generator, just to check contexts and machines texts and decide what is to prove in these texts, there are eleven rules for the proof obligation all defined and labeled inside the Rodin platform **[12]**.

III. Case Study

1. Architecture presentation

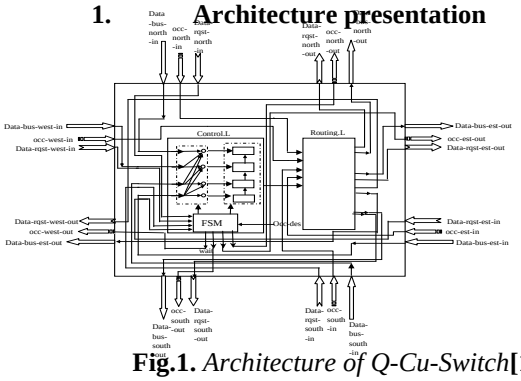


Fig.1. *Architecture of Q-Cu-Switch***[12]**

In this report we focused on the Q-Cu-Switch formalization which is a combination between two existed switches(Fig.1). This next table demonstrates the difference between a Q-Switch and a Cu-Switch:

Q-Switch **[13]**	Cu-Switch **[13]**	Q-Cu-Switch **[13]**
Unidirectional bus	Bidirectional bus	Bidirectional bus
One buffer for each input	one buffer for 4 inputs	A single buffer for 4 inputs.
The priority on right for the arbitration Policy.	Arbitration Policy is based on the priority on right.	Arbitration Policy is based on the priority on right.

Remark: It may that several flits take the same way-out direction. In this case the switch takes 3 flits at maximum.

A policy of arbitration must be adopted for the logic routing which manages the priority of sending of the flits. This policy is based on the rule of "the priority on right". It is built individually for each port of Q-Switch (Fig.1).

2. Routing algorithm

The flit is initially conveyed in direction X and in addition following the direction Y (Fig.2) until the final destination This classic routing algorithm is not adapted to the networks dynamic evolution. Using this module based on the policy of routing of Q-Switch with its algorithm of

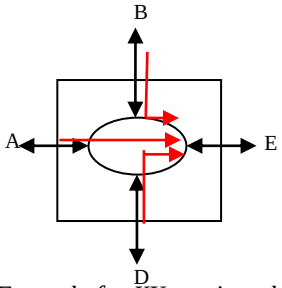


Fig.2. *Example for XY routing algorithm* **[01]**

adaptive routing modified XY. The flits are conveyed as the same case in the classic XY routing algorithm. If during routing of X, a flit meets a processing element, it will modify the routing temporarily of X with Y. With this intention, Q-Switch uses its coordinates and the signals control of the input which indicate the nature of its neighbors (Q-Switch or IP). The flits cannot take the arrival directions this restriction simplifies a little bit on the logic routing to each port.

This routing algorithm which makes the flits always takes the shortest way between two IPs, i.e. All the sending flits of one IP to the other will cross the same number of Q-switches. This fact avoids the regrouping situations of the flits to its destination node. In order to avoids situations of deadlock for some Forbidden cases.

IV Model Generation

The new validation approach using the formal technics based on the generation of model of a high level design of the MPSoC for the architecture NoC. The formal translation which is the step of getting a formal model in event B from the code VHDL representing the Q-Cu-Switch architecture.

In the next section we try to exploit some important points we had remark during the phases used on the auto-translate model.

1.Formal translation model

Two method are involved in this phase the translation of the code blocks (total blocks approach) to a model in event-B or just the checked blocks (checked blocks approach) using the proof obligation.

The main concept of making formal translation in the first case to the event-B model is the composition of the event B model itself that it made of several components: machines containing the dynamic parts for a model and contexts containing the static parts for a model **[11]**. So the extraction from a VHDL code of the dynamics parts and the statics ones required the declarations in the code and the behavior of the code used i.e. the extraction of relations between different instructions blocks.

The declarations on ENTITY even on ARCHITECTURE can represent the constant or variable or set for each event B model, so the axioms, theorems, invariant and different predicates are the result of the translation of instruction blocks.

This the event B model **[04]** obtaining for Q-Cu-switch which has toke as an illustrated example to show the first approach for the formal auto-translate model step knowing that this last have been improved and it assured the well function for some characteristic of the Q-Cu-switch architecture programmed on VHDL**[08]**.

It is the first model (Fig.3) which generally contains predicates when just the human may be able to model but in this case (Q-Cu-switch architecture) the possibility of the translation was clearly present and thus can be a positive point for this kind of translation approach

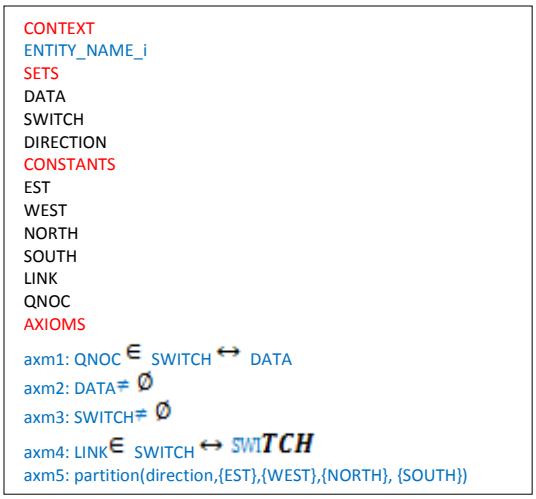


Fig.3.a *First Event B model for QNOC system* **[04]**

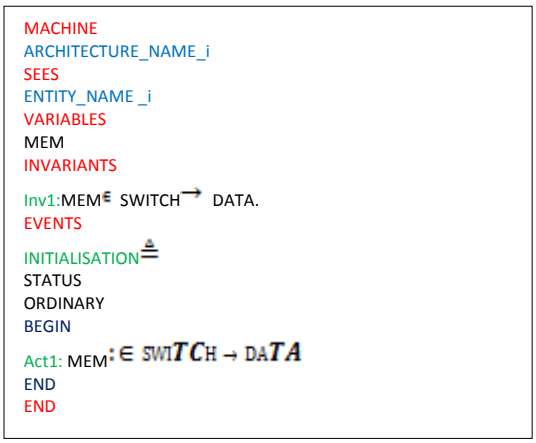


Fig.3.b *First Event B model for QNOC system* **[04]**

2. Complete blocks step

a) Context generation

The figure (Fig.4) summarizes the reason to consider the existence of DATA as set and this is the manner used:

-In the part Generic a great value with the type integer has allocated for a variable called N, which lead to check it in all the declarations which had used this variable.

Generic integer N:=20 ; -- this instruction

Constant integer N:=20; -- or even this instruction

- From all the checked declarations there are 4 Input ports and 4 Data output ports and that means the existence of a set called DATA :

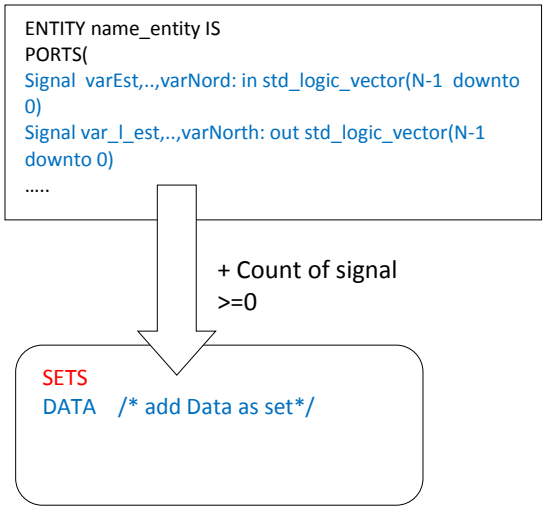


Fig.4. *Example of translating to a set*

VHDL syntax:

Signal var1,var2,var3,var4: In std_logic_vector (N-1 downto 0)

Signal var_out1,var_out_2,var_out_3,var_out_4: Out std_logic_vector(N-1 downto 0)

And all the other sets had been extracted by the similar method at almost.

For the case of constants we take LINK which had extracted because of the variables which represent ports in the reason that they had Std_logic as type (Input and output) thus oblige the communication using at least a link between switch, when it assumed that the set SWITCH had extracted with the other sets using the explained method previously (Fig.5).

VHDL syntax:

Var_io_1,…….,var_io_n : in Std logic ;

Var_io_1,…….,var_io_n : ou t Std logic ;

Therefore it is similar to translate from the VHDL code all the constants with the particularity of:

- The enumerated type may toke as Constants.
- Some relation or function (picked from instructions) which must already exist as a constants e.g. QNOC.

As what we explain about axioms previously, from the falling edge for each process, the variable used is for a SWITCH (or its component) or DATA in bidirectional sense (in switch register, data out of the switch). And the picking up is doing for any element from of each set.

VHDL Syntax

- Case for the relation SWITCH → DATA :

data out=different_Values; -- different_vaules can take register

- Case for the relation DATA → SWITCH :

Register=different_values; -- different_values can take data in

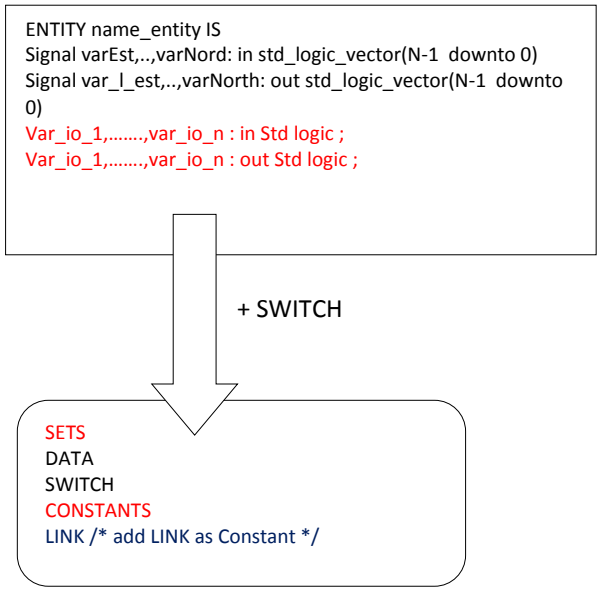


Fig.5. *Example of translating to a constant*

Extracting from the falling edge implies to specify all the system when it assumes that the constant QNOC was extracting i.e. at result:

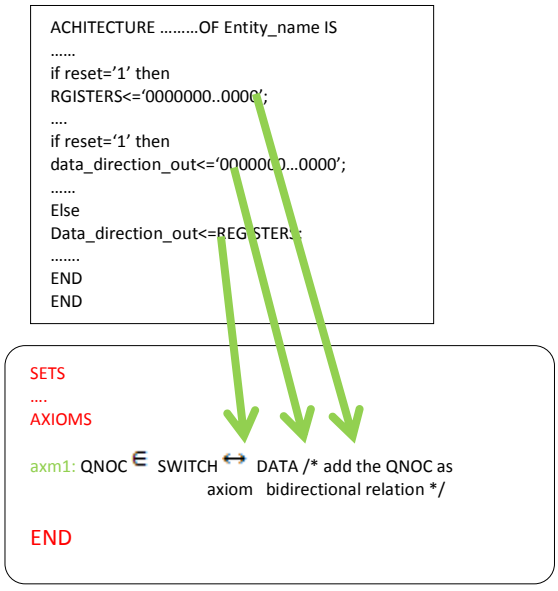
axm1: QNOC $\in$ SWITCH $\leftrightarrow$ DATA

It is similar to translate from the VHDL code all the Axioms which are:

- A typing for a constant or sets.
- Set definition for constant or sets as predicate for either the relation type or the set construction.

Remark: Founding out that the events in the machine contain guards and substitutions which are predicates thus lead to use the same method for the axioms translation.

This next figure (Fig.6) illustrates the manner to extract the predicate *axm1*.



Taking example(Fig.7) the event *evt* there is a test used for the *nb_paquets* for a safety routing with conditions:

VHDL syntax

- Condition(or guard) 1:

Data_to_route <=data_direction_in; -- and this is the action

Registers <=anothers_differentes_data_direction_in;

And this condition for the no-duplicate copies for a same data thus is translating as:

$$MEM(SWITCH_{RECEIVER}) \neq DATA1$$

- Condition(or guard)2:

Router_occ_out=0000|1111;

⋮

Occ_west_out<=router_occ_out(i);-- when $0 \leq i \leq 3$

And that means thesending information to a Nabors about the state of the SWITCH_ROUTER:

$$SWITCH_{ROUTER} \mapsto SWITCH_{RECEIVER} \in LINK$$

These conditions summarize how to routing a data and that present in the action:

$$MEM(SWITCH_{RECEIVER}) := DATA1$$

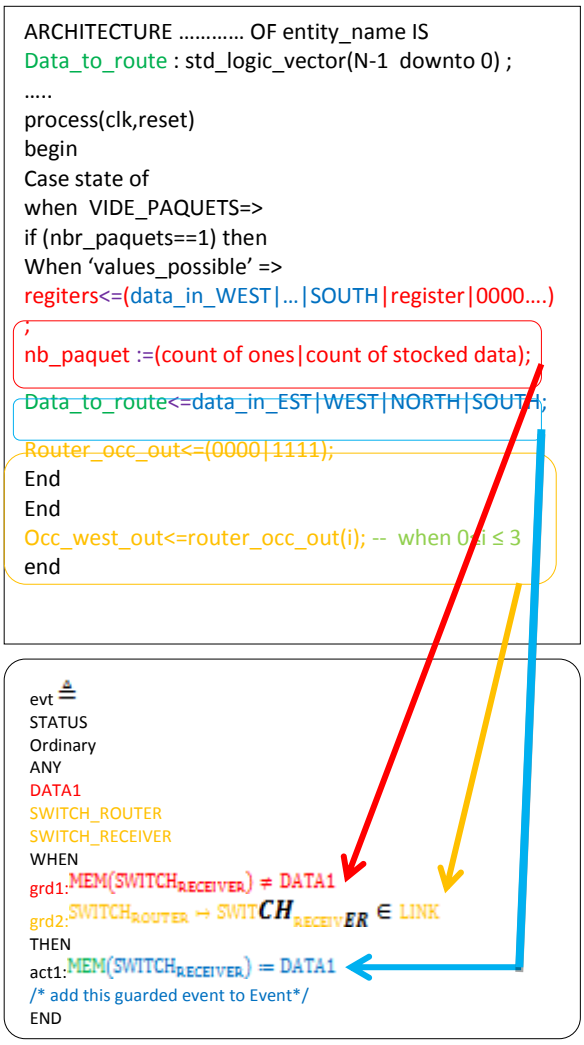


Fig.7. *Example of translating to a guarded event*

And all the events have the same manner to being extracted taking as consideration:

- The type of each event.
- The use of sets in each event.

c) **Approach disadvantage**s

This approach has as negative points:

- Un-use interpreting for some corposants which can added intuitively.
- The conflict on the order during the translation either sequentially or concurrently or following the system characteristics.
- The dissimilarity in the translation using the behavior makes the complexity for this process increase.

These reasons imply to use the second approach which is our Future work.

3. Checked blocks step

For the auto-translate model which uses the checked blocks the manner of translation is the same in the particularity of:

- The focalization on the events and the set of predicates proved in each model.
- The verification of the existence for the declaratives parts for each model in the VHDL code.
- The addition for the messing declarative parts checked.

V. Conclusion

This research work is the combination of several translators event B, other languages but in the particularity of the use of the semantic basis and it take as advantages:

- The possibility of translation of VHDL code onto proved event B model which will assure the function of the QNOC system.
- The automation of the translation decreases the modeling time rate.

However this work will be developed to be adopted for any QNOC architecture even in the future several systems and also to obtain the well-structured VHDL code assured by the well-proved Event B model to decrease the *time to market.*

REFERENCES

[01] S. Jovanovic, C. Tanougast, C. Bobda and S. Weber: *A New Deadlock-free Faulttolerant Routing Algorithm for NoC interconnections*, The 19th int. Conference on Fiold Programmable Logic and Applications, IEEE Circuits and Systems Society,.Pp. 326-331. 2009.

[02] N. karimi, A. Alaghi, Z. Navabi, *online network-on-chip switch fault detection and diagnosis using functional switch faults.*Journal of universal computer Science, vol.14, no. 22, pp. 3716-3736. 2008.

[03]S. Y.Lin, W. CShon, C. C. Hsu and A. Y. Wu, *Fault-Tolerant router with built-in self-Test/-daignosis and fault-isolation circuits for 2D-mesh based Chip multiprocessor systems*, Int. Journal of Electrical Engineering. Vol. 16, No 3,pp213-222, 2009.

[04] T.B. Meziane. *For modeling a NOC switch using B event*, theme for Master Degree Computing ingenery University of Tiaret. 2011.

[05] K. Morin-Allory L.Fesquet D.Borroine *Asynchronous Assertion Monitors For Multi-clock domain system verification* 2006.

[06] Q. Zhu, A. Matsuda, S. Kuwamura, T. Nakata *.An Object-Oriented Design Process for System-on-Chip using UML* IEEE article , 2002.

[07] R. Sabbagh H. Foster *Using Formal Technology to Improve Coverage Results* April 23th 2012.

[08] A. Hariche, *Design and of a verification environment for the implementation of embedded applications*, theme for Master Degree Computing ingenery University of Tiaret. 2012.

[09] J.-R. Abrial. *Extending B without changing it*, for developing distributed systems, Proceedings of the 1st Conference on the B method, pp. 169–191. November 1996.

[10] E. Dijkstra. *Guarded commands*, nondeterminacy and formal derivation of programs. *Communications of the ACM CACM 18*. 1975.

[11]J.-R. Abrial. Modeling in Event-B ,system an software engeenring.Book published by university of Cambridge pp 176-203 .2010.

[12] R.J. Back *Stepwise refinement of action systems*,CREST 1991.

[13] H. Daoud *Modeling and validating for architecture of network on chip NOC*, theme for Master Degree Computing ingenery University of Tiaret .2011.