Functional Verification of the RCMP Egress Routing Logic

Murugesh Palanisamy and Sofiène Tahar

Technical Report

Dept. of Electrical & Computer Engineering, Concordia University
1455 de Maisonneuve W., Montreal, Quebec, H3H 1M8 Canada
E-mail: {m_palani,tahar}@ece.concordia.ca

September 1998

Abstract

In this report, we present our results on the functional verification of an Egress cell traffic module using both simulation and model checking. The design considered is a commercial product used in a network port interface for an ATM switch fabric. The available behavioral level VHDL design was translated in BLIF-synthesizable Verilog set and verification was carried out by simulation as well as using model checking in VIS (Verification Interacting with Synthesis) tool. One design error which could lead the system into a hangup state was detected through Synopsys-VSS and Verilog-XL simulations and it was also confirmed through model checking in VIS. The verification in VIS was done in a reasonable CPU time as the design consisted of only 131 latches in total.

Contents

1 Introduction

ATM (Asynchronous Transfer Mode) provides a high-speed, low-delay, multiplexing and switching network to support user traffic, such as, voice, data and video applications. In an ATM network, the RCMP (Routing, Cell counting, Monitoring, Policing) process both the Ingress and Egress cell traffic in a network port interface for an ATM switching fabric. Main application areas for the RCMP are in : 1) Edge switches, 2) Enterprise switches, 3) Core switches, and 4) Access Muxes and Residential Broadband switches.

This project particularly deals with the functional verification of the Egress Routing logic of the RCMP using both simulation and Symbolic Model Checking. Symbolic Model Checking provides a fully automatic verification method and allows certain properties expressed in a branching-time temporal logic, CTL (Computational Tree Logic), to be verified completely automatically [2]. Although symbolic model checking algorithms provide efficient property checking results, the automatic verification is limited by the size and the complexity of the design.

Simulations were performed on the available Synopsys-VHDL code (a behavioral level description obtained from PMC-Sierra Inc.), first using Synopsys-VSS and then using Verilog-XL on the translated Verilog code. Using a Symbolic Model Checking tool, VIS, property checking was carried out on the Verilog behavioral description. Through both simulation and verification, a small design error which could violate the system behavior was successfully detected and corrected.

The rest of the report is organized as follows. Section 2 gives an overview of the function of the Egress Routing logic. Section 3 describes its behavioral specification. The VHDL/Verilog RTL modeling of the routing logic is presented in Section 4. Section 5 discusses about the model checking of the logic using property checking in VIS and its results. Finally, Section 6 concludes the report.

2 Egress Routing Logic

The RCMP performs following ATM layer functions [4]:

(1) Header translation and cell append for routing,
(2) Policing,
(3) Cell counting, and
(4) OAM cell processing and routing for 64K virtual channels.

The RCMP supports Ingress function (cell transfer from PHY device to the Switch fabric) primarily, but can be configured for use in Egress direction (cell transfer from Switch fabric to PHY devices) through Egress logic. The Egress Routing logic of RCMP discussed in this report does only header translation of ATM cells and does not perform traffic shaping and RM (Resource Management) cell insertion. A typical Egress application is shown in Figure 1.

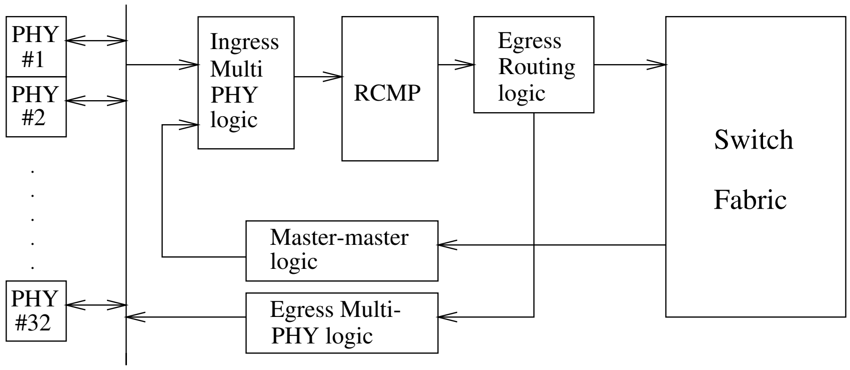

Figure 1: Block Diagram of Egress Routing Logic of RCMP

The cell traffic coming from the Switch fabric are looped back into the input cell interface of the RCMP, which treats these cells as if they are sourced from another PHY port. The RCMP takes care of the extended cell format used in switch core by removing the appended cell octets and producing 53-octet ATM cells for the PHY layer. This is done by VC table lookup and performing the address translation and multicasting if appropriate. The advantage of doing multicasting in Egress is that it avoids congesting the Switch. The Egress Multi-PHY glue logic decodes the PHY address to generate the write-enable to the appropriate PHY port.

The Egress routing glue logic performs the routing function by using the first prepend word of a cell, which contains a routing bit that indicates whether the cell should be routed to the Switch or back to the Egress direction. It also contains a 5-bit PHY address for the Egress cells. The reason for using the first prepend word is to allow the glue logic to determine the routing of the cell before the remainder of the cell arrives, thus avoiding the need to buffer the cell.

The main features of the Egress Routing logic are [4]:

- Supports Backward Routing of cells from the RCMP output. Decodes the prepend word to determine whether a cell is destined to the Switch fabric or one of the PHY devices in the Transmit direction.

- Supports either Direct Multi-PHY interface or Indirect-addressing Multi-PHY interface for upto 32 PHY's.
- Supports cell format with one prepend and 27-words (16-bit wide). Prepend word is stripped off.
- Supports back-pressure flow control (referred to as stuttering in the code) from the Switch.

The assumptions made on the Egress logic are [4]:

(1) The same clock is used for the PHY interface and the Switch interface. This is because the Egress Routing glue logic does not perform synchronization between two different clock domains.

(2) Both the PHY interface and the Switch interface have the same data bus width (16-bit). This greatly simplifies the Egress Routing logic.

3 The Behavioral Specification

The Egress Routing logic should perform the following for every cell transfer [4]:

(1) It asserts RRDENB (read-enable signal) to the RCMP to read out the first Prepend word, and then halts the cell read by de-asserting RRDENB.

(2) Examines the routing bit to connect the control signals from either the Switch or the Egress Multi-PHY glue logic.

(3) Strips off the prepend word for Egress cells. The Prepend word is stripped off to empty this word from the RCMP output FIFO but not actually transferring the data to the destination.

(4) Starts the cell transfer. If the cell is destined to the Switch, the Switch controls the RRDENB signal. Otherwise, the Egress logic controls both the RRDENB and PHYWRENB (write-enable signal for PHY) signals.

(5) The Postpend word required for other purposes is not appended to the Egress cells by the RCMP. So the Egress logic stops the cell transfer without postpend stripping and waits for the RCA (Cell Availability signal) from the RCMP.

The functional specification can be represented as a state diagram as shown in Figure 2. The state diagram additionally incorporates a word-counter, which is used to detect the end of cell transfer. For the Egress cell flow, both the RCMP and the

PHY ports have slave interfaces, and the glue logic acts as the master, providing both the RRDENB and PHYWRENB signals. The Switch interface acts as a master by providing MRDENB signal.

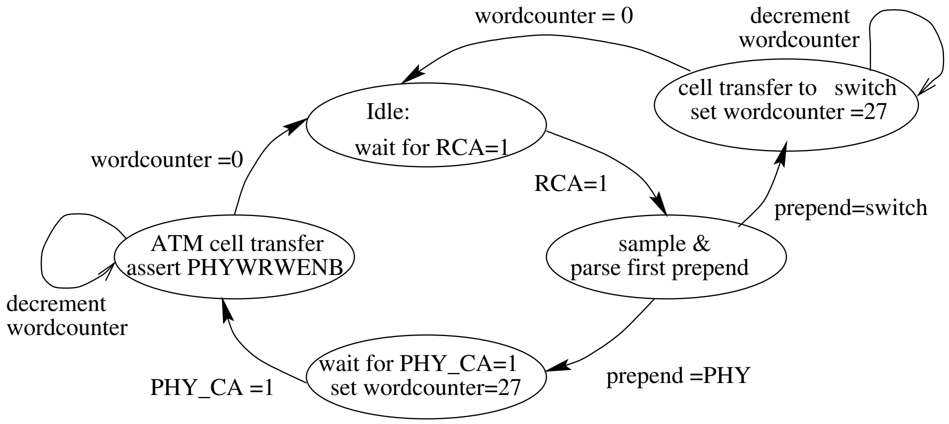

Figure 2: State Diagram Representation of the Egress Behavior

4 VHDL and Verilog Implementation

This section explains the implementation of the Egress Routing logic in VHDL and Verilog.

4.1 VHDL Design

The VHDL design was a total behavioral description of the Egress logic. The logic was designed using six concurrent processes, doing the functions of output assignment, next-state transition and assignment, state decoding, PHY and switch interface decoding and the word-counting.

The block diagram in Figure 3 represents the VHDL implementation. The state-machine handles all the cell-transfer handshaking. The interface to the RCMP consists of the control signals RCA, RSOC, and RRDENB, plus 16-bit data RDAT and the parity RPRTY. RCMP acts as a “slave” interface which receives the read-enable signal, RRDENB. The interface to the Switch includes MCA and MRDENB. The Switch acts as a “master” sourcing the read-enable signal, MRDENB. The interface to the PHY devices consists of the 4 PHYTCA’s, 4 PHYWRENB’s (in Direct mode), and the 5-bit PHY address called PADD (in Indirect-addressing mode). The output data, ODAT and the parity OPRTY are connected to both the Switch and the PHY devices. Note that signals ending with ’B’ are active low.

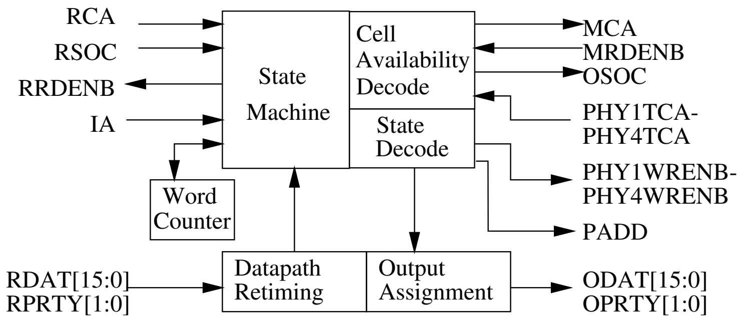

Figure 3: Block Diagram of the Egress VHDL Design

There are two modes of operation, direct and indirect modes, which correspond to the interface mode to the PHY devices. The interface to the Switch is identical in both modes. The state diagram shown in Figure 4 represents the sequence of operations.

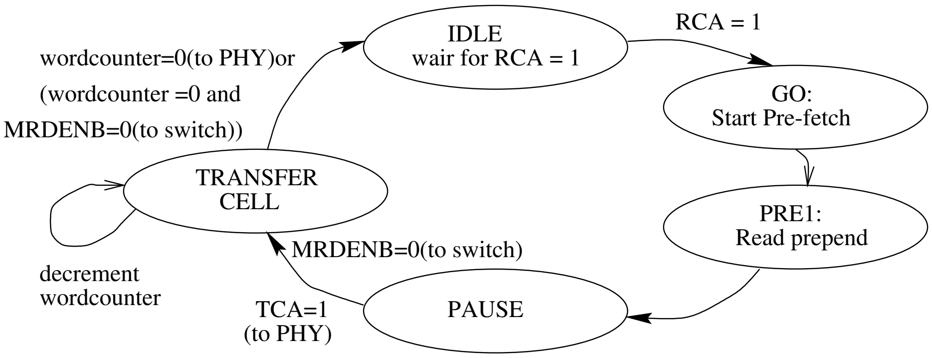

Figure 4: State Diagram Representation of the Egress VHDL Design

Direct Mode Operation

In this operation mode, cell transfer occurs in the Egress direction to the 4 PHY devices connected directly to Egress Multi-PHY glue logic. The timing diagram for this mode operation is given in [4]. The following explains the state-machine sequence for each cell transfer:

Clock#1: RCMP indicates a cell is available by asserting RCA.

Clock#2: Egress logic samples the RCA.

Clock#3: Egress asserts RRDENB to read out the first prepend. P_STATE→ GO, Cellcount=28.

Clock#4: The first prepend word (RDAT[15:0]) along with RSOC read out from RCMP.

P_STATE→ PRE1, Cellcount=27.

Clock#5: The routing tag, BWD_TAG, is decoded from the prepend word, which indicates that the cell is destined to one of the four PHY devices.

P_STATE→PAUSE. Waits for the PHY device to assert PHY_CA (cell availability signal).

Clock#6: If the PHY_CA of the corresponding PHY device is asserted, RRDENB continue to be asserted, and the rest of the cell is transferred to the device. Otherwise, P_STATE remains in PAUSE till PHY_CA becomes high. Word 1 and OSOC are presented on the output to the PHY device.

Clock#7-31: P_STATE→TRANSFER_CELL, and the remaining of the cell (26 words) are transferred from RCMP to the PHY device. After the transfer, RCA_S is forced low.

Clock#32: P_STATE→ IDLE, Cellcount = 0.

Clock#33: Egress logic once again samples RCA for any cell availability.

Indirect Mode Operation

In indirect addressing mode, upto 32 PHY devices can be interfaced with Egress Multi-PHY glue logic. It is identical to the direct mode operation. Only difference is that, instead of 4 direct PHYWRENB signals, a 5-bit PHY address (PADD) is output. PADD decodes to address a maximum of 32 PHY devices. The timing diagrams for this mode of operation are given in [4].

Switch Mode Operation

The switch mode operation is the same in either direct or indirect addressing mode. The operation is identical to that of direct mode upto prepend word fetch (till Clock#4).

Clock #5: The decoded BWD_TAG indicates that the cell is destined for the switch. Thus, MCA is asserted. The Egress logic waits for the assertion of MRDENB from the switch.

Clock#6-8: With the assertion of MRDENB, the control is directly transferred to the switch. Egress outputs word 1 and OSOC. P_STATE→ TRANSFER_CELL.

Clock#9-32: The rest of the ATM cell is transferred. RRDENB is deasserted. RCA_S is forced low.

Clock#9-32: P_STATE→ IDLE, Cellcount = 0.

Clock#33: Egress logic samples RCA.

Detailed timing diagrams for the switch mode operation are given in [4].

Stuttering Switch Mode Operation

When the switch deasserts just after the first word or just after the first word, an internal buffering mechanism is used in the Egress glue logic which holds the data till the switch becomes ready again to read. This is done by sampling the input and output data in stuttering situation. It is illustrated in the timing diagram given in [4]. The stuttering mechanism is only provided for the first and last words of a cell transfer in switch mode.

4.2 VHDL Simulation

Simulation in Synopsys-VSS was carried out on the VHDL design by constructing test benches for each mode of operation. The system entered irreversible "PAUSE" state as the processes were not able to detect the changes occurring in the variables through their sensitivity lists of the processes, namely "output assignment" and "ca_decode". After this modification, the simulation results conformed to the timing diagram specifications.

4.3 Verilog Design

The given VHDL code was converted to a Verilog description with the modification and correctness-by-simulation was established against the timing diagrams using Verilog-XL once again.

5 Formal Verification Using VIS

VIS (Verification Interacting With Synthesis) is a tool that integrates the verification, simulation, and the synthesis of finite state hardware systems [1]. It uses a Verilog front end and supports fair CTL model checking, language emptiness checking and hierarchical synthesis [1]. VIS operates on an intermediate format called BLIF-MV, which is an extension of BLIF (Berkeley Logic Interchange Format). VIS includes a stand-alone compiler from Verilog to BLIF-MV, called VL2MV.

The source Verilog program was converted to a VL2MV-synthesizable subset of Verilog by introducing non-determinism and symbolic variables. For producing inputs (RDAT, RPRTY, MRDENB, PHY1TCA, PHY2TCA, PHY3TCA, PHY4TCA), a stimuli-process was incorporated into the Verilog code, which monitors the P_STATE variable and generates the corresponding inputs. The following section describes the CTL properties that were formed to cover the behavior of the Egress Routing logic for verification in VIS.

5.1 Properties Description

CTL properties comprising liveness and safety properties were considered for the verification, similar to the verification work in [3]. The safety properties included language emptiness checking and invariant checking. In the following CTL expressions, "!", "$\rightarrow$ ", "$*$" and "+" denote logical "not", "imply", "and" and "or", respectively.

Safety Properties

Three fairness constraints (Buchii type) were specified, covering the fair paths. Similarly, invariant properties, which must be satisfied throughout the behavior of the system, were specified. They are described below:

Fairness Constraints: The following fairness constraints must be satisfied infinitely often along all the fair paths in the system.

Fair#1: $!(p\_state = pause)$;

Fair#2: $!(cellcount[7:0] = 28)$;

Fair#3: $!(rca\_s = 1)$;

Invariants: The invariant properties, which model the invariant behavior of the system, must be satisfied for the proper synchronization of every cell transfer either to the PHY device or to the Switch.

Inv#1: $AG(cellcount[7:0] = 0)$;

Inv#2: $AG((p\_state = transfer\_pre1) \rightarrow (cellcount[7:0] = 28))$;

Liveness Properties

Liveness#1: In the direct addressing mode, if the routing (bwd_tag) corresponds to the PHY device #1, and system is in the "pause" state, then eventually, the system must move to a "transfer_cell" state and the write-enable signal (phy1wrenb) for that device must be asserted.

$AG((bwd\_tag[7:0] = b00000011)*(p\_state + pause)) \rightarrow$
$AF((p\_state = transfer\_cell)*(phy1wrenb = 0)))$;

Three more similar properties can be specified for the other three PHY devices in the direct mode.

Liveness#2: After the fetch of the first prepend word (rsoc=1), the rest of the ATM cell is transferred to the output with the first word notification (osoc=1) in all the fair paths of the system.

$AG((rsoc = 1) \rightarrow AF(osoc = 1))$;

Liveness#3: In the indirect addressing mode for the PHY devices, the transfer of the ATM cell takes place along with the correct decoded address at the address port (padd4:0]=bwd_tag7:3]).

$AG((bwd\_tag[2:0] = 3) \rightarrow$
$AF((p\_state = transfer\_cell)*(padd4:0] = bwd\_tag7:3])))$;

The above property takes care of the addressing of 32 PHY devices in the indirect addressing mode in the Egress direction.

Liveness#4: With the routing tag carrying the addressing of the Switch and cell availability signal (mca) to the switch enabled, the system transfers the rest of the cell with the assertion of the read-enable signal from the switch.

$AG((bwd\_tag[7:0] = b00000000)*(mca = 1)) \rightarrow$
$AF((p\_state = transfer\_cell)*(mrdenb = 0)))$;

Liveness#5: This property checks for the proper transition of the system from a "pause" state to a "transfer_cell" state.

$AG((p\_state = pause) \rightarrow AF(p\_state = transfer\_cell);$

5.2 Property Checking and Results

Initially, the Egress Routing logic was successfully verified against the fairness constraints to remove the presence of any non-trivial behavior. Similarly, the invariants were checked successfully. The liveness properties modeling ATM cell transfer in all the three modes of operation (Direct, Indirect Multi-PHY modes and Switch mode) were checked against the Verilog model. The verification of all the properties were done employing dynamic variable ordering, which reduces the CPU time by choosing an optimized order for the variables. The CPU time, memory usage, and the nodes allocated for every property checking are shown in Table 1.

Verification	CPU time (sec)	Memory used (MB)	Nodes allocated
Fair#1	89.2	6.33	4820221
Fair#2	88.2	8.2	7626808
Fair#3	302.4	12.2	22424387
Inv#1	20.1	3.3	8286368
Inv#2	15.3	2.9	5132624
Liveness#1	70.2	5.8	1989467
Liveness#2	23.4	3.9	3985431
Liveness#3	25.5	4.5	2436219
Liveness#4	29.6	6.3	8065623
Liveness#5	400.4	6.0	7341856

Table 1: Property Checking Results

The system hangup error in the "pause" state notified during VSS and Verilog-XL simulations was reproduced during property checking of the Liveness *Property*#5, by VIS producing an exact error trace.

To obtain a possible latch reduction, abstraction was applied wherever necessary. The input and output data were abstracted out to 8 bits from 16 bits. The cell count register was reduced from 8 bits to 5 bits as it takes integer values from 0 to 28. Further reduction in the input/output data path was not possible because of the routing tag transferred in the same data path [3]. The latch reduction yielded through above approach is summarized in Table 2.

	Combinational Nodes	Latches
Original Model	1460	131
Reducted Model	1088	87

Table 2: Latch Reduction using Partial Abstraction

Even with partial latch reduction employed, the verification time (CPU time) was not considerably reduced for the properties described in the previous section. Table 3 illustrates two verification examples obtained using the reduced model with latch reduction.

Verification	CPU time (sec)	Nodes allocated
Fair#3	263.1	15826146
Liveness#5	373.4	5611521

Table 3: Property Checking using Latch Reduction

6 Conclusions

The formal verification of Egress Routing logic of the RCMP through property checking was done using VIS tool. A design error which could lead the system to a deadlock "pause" state, was detected and rectified through simulation (both Synopsys-VSS and Verilog-XL) and through property checking using VSS. The expressed CTL properties covered all the behaviors of the system in all the addressing modes except the "stuttering mechanism". This was not possible because it involved the introduction of timing delays in the input assignments to the model and correspondingly, VIS does not accept a timed Verilog model. However, the correctness of the stuttering mechanism was established through Verilog-XL simulation.

As a future work in this direction,the synthesized Verilog model of the Egress logic can be verified against the behavioral Verilog model used in this project, using sequential equivalence checking in VIS.

References

[1] R. Brayton et al.: VIS: A System for Verification and Synthesis, Technical Report UCB/ERL M95, Electronics Research Laboratory, University of California,Berkeley, December 1995.

[2] E.M.Clarke and E.A.Emerson: Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic, *ACM Transactions on Programming Languages and Systems*, Vol.8, No.2, April 1986, pp. 244-263.

[3] J. Lu and S. Tahar: On the Verification and Reimplementation of an ATM Switch Fabric Using VIS, Technical Report No.401, Dept. of Electrical and Computer Engineering, Concordia University, September 1997.

[4] PMC Sierra, Inc.: PM7322, RCMP Egress Routing Logic in VHDL, March 1997. http://www.pmc-sierra.com/register/rcmp800.asp