A Markov Model to Control Heat Dissipation in Open Multi-Frequency Green Routers

Alfio Lombardo, Diego Reforgiato,Vincenzo Riccobene, Giovanni Schembra
Dipartimento di Ingegneria Elettrica, Elettronica e Informatica (DIEEI)-University of Catania
{lombardo, diegoref,vincenzo.riccobene, schembra}@dieei.unict.it

Abstract—**One of the main goals of telecommunications engineering today is to make Internet routers green. A positive side effect of this is the reduction of the working temperature of their internal components. This idea is encouraging the realization of smaller scale routers, but this process has to be supported by the confidence that temperature remains in a given range. For this reason the target of this paper is to propose an analytical discrete-time Markov model that allows green router designers to choose green router parameters with constraints related to the average working temperature. Other important constraints are loss probability during clock frequency switches and energy saving gain. The proposed model is applied to a case study to show how it can be used to design some router Governor parameters to achieve the target of maintaining the average temperature below a given threshold.**

Keywords-component; Green Routers, Heat Dissipation, Power Consumption, Performance Evaluation, Markov model, NetFPGA

I. INTRODUCTION

Telecommunications networks are often provisioned for worst-case or busy-hour load, and this load typically exceeds their long-term utilization by a wide margin; moreover, as shown in [1], current network nodes have a power consumption that is practically constant and does not depend on the actual traffic load they face. The implication of these factors is that most of the energy consumed in networks today is wasted [2-3].

A non-marginal side effect of high energy dissipation is the increment of the temperature of the places where network devices reside, with a consequent further waste of energy used by cooling machines to maintain the temperature of the local environment constant.

The steadily rising energy cost and the need to reduce the global greenhouse gas emission make this occurrence unsustainable: today, in fact, 37% of the total ICT emissions are due to telecommunications companies infrastructures and devices [4]. For this reason, addressing energy efficiency challenges in wireline networks is receiving considerable attention in the literature today [5-6]; moreover many research projects have been started on this topic (see for example [7-9]). Thus, some novel hardware devices, so-called "green routers", are expected in the near future to allow to enter different power states [10] according to the input traffic. A lot of work was done in the past, focusing on the definition of power management techniques, like for example the static techniques described in [24-27], and the adaptive policy proposed in [28]. Two approaches have been proposed to reduce energy consumption in network components [6]. The first is based on putting network components in sleeping state during idle intervals, reducing energy consumed in the absence of traffic. The second one is based on adapting the rate of network operations to the offered workload. Rate adaptation in particular is usually achieved by scaling the processing power according to the data rate the router has to manage; at this purpose the clock frequency driving the router processes can be modified according to the input data rate [11]. The energy aware technique to be used in a green router depends on a number of factors, including the role of the router in the network, the profile of incoming traffic, the hardware complexity and the related costs with respect to the energy we can potentially save and the QoS we want to guarantee to the users [12].

Now, let us note that the introduction of green management techniques to make network routers green has an important consequence on the decrease of working temperature of the hardware device. As known, temperature is one of the major factors which must be considered and addressed in the design and the manufacture of electronic devices, and specifically routers, since operating at higher temperature degrades system reliability, causes performance degradation and leads to higher cooling and packaging costs [13].

Moreover, "smaller and faster" are the chief demands driving today's electronic design. These issues translate into high power densities, higher operating temperatures and lower circuit reliability. Therefore, greening a router can be considered as a leveraging approach to move towards this direction. In other words, reduction of the average temperature in green routers due to the application of algorithms aimed at reducing energy consumption will allow designers to modify hardware, reducing its size and the size of the passive and active cooling systems, since a package designed for the worst case is excessive. However, the above hardware modifications can make again router circuits heat beyond their designed thermal limits. For this reason, the working range of temperature becomes again an important issue in green router design.

With all this in mind, the paper target is to introduce an analytical model of a green router with given requirements in terms of temperature, QoS and energy consumption. This model allows the designer to determine in advance if the device will operate within recommended thermal ranges when the green router governor uses a given energy saving policy. In

addition, the same model can be used to evaluate the achieved amount of energy saving.

The proposed model is a multi-dimensional discrete-time Markov model. A case study based on the open NetFPGA Reference Router [14] is considered to show how the proposed model can be easily applied to a real case scenario. At this purpose we modified the NetFPGA Reference Router leveraging on the facility of the NetFPGA platform that allows designers to reduce the clock rate from 125 to 62.5 MHz by changing the value of an ad-hoc hardware register.

The paper is structured as follows. Section II introduces the NetFPGA Reference Router architecture. Section III shows the measurements that we have conducted on both temperature and energy consumption in the modified version of the Reference Router. Section IV describes the Markov model that we have defined. All the results of our analysis are shown in Section V. Finally, Section VI ends the paper with authors' conclusions and future directions.

II. NETFPGA ROUTER ARCHITECTURE

The NetFPGA [14] is an accelerated network hardware that augments the functions of a standard computer. A user-programmable FPGA (with two PowerPC processors) is hosted on the board together with SRAM, DRAM, and four 1 Gbit/s Ethernet ports[1]. The FPGA circuitry directly handles all data-path switching, routing, and processing operations of Ethernet frames and IP packets, leaving software to handle control-path functions only [14].

The open router we consider in this paper is the Reference Router [15] implemented on the NetFPGA platform (as it is in the version 1 Gbit/s). Its scheme is shown in Fig. 1. The reference pipeline consists of the user data path, eight input queues and eight output queues (each port has a CPU queue and a MAC queue). In the figure the input and output MAC queues are shown: the input MAC queues contain the packets coming from the physical Ethernet ports and forwarded to the packet process engine, whereas the output MAC queues forward the packets from the process engine to the Ethernet ports.

It is possible to change the NetFPGA clock rate from 125 MHz to 62.5 MHz and vice-versa by changing the value of the CPCI_CNET_CLK_SEL_REG hardware register. The clock frequency switch we have developed is performed by the host computer accessing the hardware register via software. This clock frequency variation affects only a part of the NetFPGA board. The physical ports still run at 125 MHz. There are shallow (2KB) FIFOs between the two clock domains that manage the clock domain crossing. This clock variation facility constitutes the base of our work.

As stated in [16], when the Reference Router works at 62.5 MHz, it is able to forward traffic without loss only when the bit rate is less than or equal to 2 Gbit/s. During each frequency switching interval, the NetFPGA loses all the incoming packets. We are currently working on a hardware version of the clock frequency switch where the use of some buffer and

[1] 10 Gbit/s Ethernet ports are already available in the newer version.

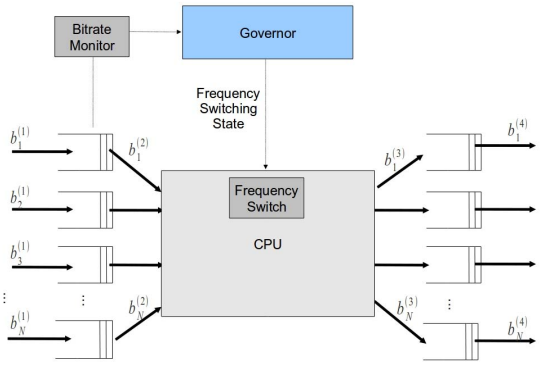


Figure 1. Architecture of the NetFPGA router

an adequate policy will prevent packet loss during switch; besides, within the NetFPGA platform we can add FIFOs to the NetFPGA pipeline. The management of such FIFOs introduces a power consumption though, and for this reason we have decided to not include them in this paper. However, the buffer may be included in our study as our analytical model is general and does not depend on whether the frequency switch is realized in hardware or software. In this case, the same proposed model could be used with the only difference that the QoS parameter loss probability will have to be replaced with the average delays introduced using the FIFOs.

In order to manage frequency switches maintaining quality of service (QoS) acceptable while decreasing energy consumption and at the same time controlling the CPU temperature, we introduce a Router Governor, that is an entity which decides the router policy to be adopted.

Quality of Service is defined by the following parameters:

- Probability of packet loss during frequency switching intervals;
- Energy saving gain;
- Mean temperature on the CPU surface.

Energy saving gain is defined as follows:

$$\rho = \frac{P_{MAX} - P_{MEAN}}{P_{MAX}} \cdot 100\% \quad (1)$$

where P_{MAX} is the maximum power consumed, i.e. if no saving policy is applied, while P_{MEAN} is the mean value of the consumed power when the Router Governor works to save energy.

Let us note that other traditional QoS parameters characterizing the router, like for example loss probability and delay, are not considered here because they are not altered by the presence of our Router Governor. To this purpose, the clock frequency of 62.5 MHz is not used when the input traffic bit rate is greater than 2 Gbit/s. This information is provided by the Input Bitrate Calculator block [18,19] we developed on the NetFPGA router, with the aim of measuring

the input bit rate $B_{IN} = \sum_{i=1}^{4} b_i^{(1)}$, and communicating it to the Router Governor run-time.

More specifically, the Router Governor has the following tasks:

1. Controlling the CPU temperature, in order to avoid that the mean temperature exceeds a given threshold;
2. Avoiding that the probability of packet loss during clock frequency switches exceeds a given threshold;
3. Maximizing the energy saving percentage while respecting the two above items.

With this in mind, the Router Governor policy is defined as follows:

- RULE 1: if the clock frequency was previously set to 62.5 MHz and the current input bit rate is greater than 2 Gbit/s, then the clock frequency is switched to 125 MHz, in order to avoid losses due to the low processing power;
- RULE 2: if the clock frequency was previously set to 125 MHz and the current input bit rate is lower than 2 Gbit/s, then the clock frequency is switched to 62.5 MHz with a probability $p_G(B_{IN})$, which is adaptive to the current input bit rate: the greater the distance between B_{IN} and the threshold of 2 Gbit/s (of course for $B_{IN} \leq 2$ Gbit/s), the lower the risk of a new frequency switch needed to set the clock frequency to 125 MHz again, and therefore the higher the value we can use for $p_G(B_{IN})$; on the contrary, the lower the value of $p_G(B_{IN})$, the rarer the use of the clock frequency 62.5 MHz, and then the lower the energy saving.

It is easy to argue that the probability function $p_G(B_{IN})$ plays a very important role in the router performance in terms of loss probability, temperature and energy saving. The design of this function will be assisted by the analytical model that will be described in Section IV.

III. Temperature and Power Consumption Measurement

In this section we will show the measurements we have carried out in order to calculate the power consumption and the temperature behavior of the NetFPGA platform. Section III.A introduces the power model we have used to compute the energy measurements, whereas Section III.B describes the temperature measurements we have taken. The reference measurement scenario, shown in Fig. 2, is described below. In order to avoid that measurements were affected by external factors, all the measurements have been done in a laboratory with a constant environmental temperature of 22°C.

A. Energy consumption measurement

Power consumption has been evaluated using the following configuration: the Reference Router project has been loaded on the NetFPGA card along with SCONE (the NetFPGA control software). Ultraview PCI Smart Extender PCIEXT-64UB card has been used to isolate the power consumption of the NetFPGA card from the consumption of the NetFPGA host computer, An Agilent MSO7054A series 7000 InfiniiVision oscilloscope has been used for power measurements. Finally, we have used the IXIA, a high-precision commercial-grade hardware traffic generator with sophisticated capabilities for configuring traffic profiles, synchronizing traffic on multiple ports, and accumulating statistics based on pattern filters.

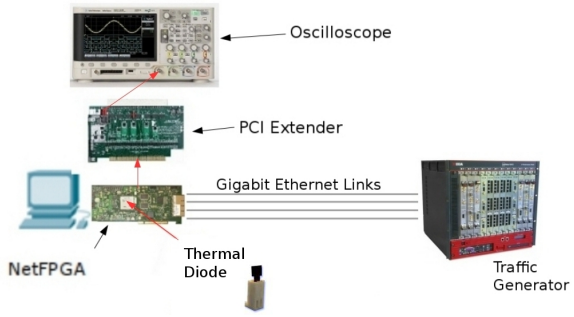


Figure 2. Reference measurement scenario

Previous studies in [17] and [20] have provided some power models and energy measurements accounting the NetFPGA router functions. Using those results, we have extended the NetFPGA power characterization to the two possible clock frequencies the NetFPGA is able to work with, i.e. 125 MHz and 62.5 MHz. Therefore, we have introduced the dependency on the clock frequency in the linear model of [17]:

$$\Psi(f_C, B_{IN}) = P_C(f_C) + K\,P_E(f_C) + \\ + N_I(B_{IN})\cdot E_p(f_C) + R_I(B_{IN})\cdot E_r(f_C) + R_O\,E_t(f_C) \quad (2)$$

where f_C is the NetFPGA clock frequency that can be set to either 125 MHz or 62.5 MHz; $P_C(f_C)$ is the constant baseline power consumption of the NetFPGA card (without any Ethernet ports connected); $P_E(f_C)$ is the power consumed by each Ethernet port (without any traffic flowing); $E_p(f_C)$ is the energy required to process each packet (parsing, routing lookup, etc.); $E_r(f_C)$ is the energy required to receive, process and store a byte on the ingress Ethernet interface; $E_t(f_C)$ is the energy required to store, process and send a byte on the egress Ethernet interface; K is the number of Ethernet ports connected (1 to 4); $N_I(B_{IN})$ is the input traffic bitrate to the NetFPGA card in packets-per-second (pps); $R_I(B_{IN})$ is the input rate to the NetFPGA card in bytes-per-second; $R_O(B_{IN})$ is the output rate from the NetFPGA card in bytes-per-second.

We have carried out several measurements to evaluate each of the variables for each clock frequency. To such a purpose, the measures have been organized in steps of work:

1. Measure of the baseline power consumed by the NetFPGA card when none of the Ethernet ports are connected;

2. Measure of the baseline power consumption connecting each Ethernet port with no traffic flowing;

3. Measure of the energy cost related to the packet processing and in particular:

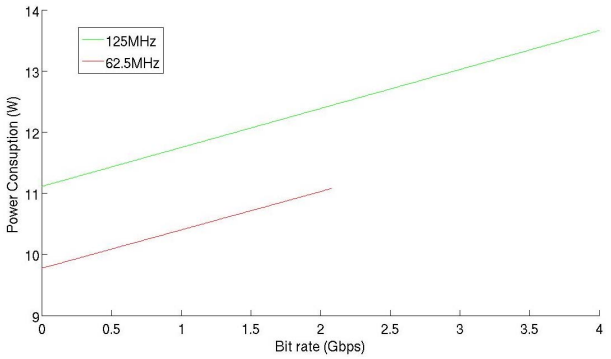


Figure 3. Power consumption model for NetFPGA platform.

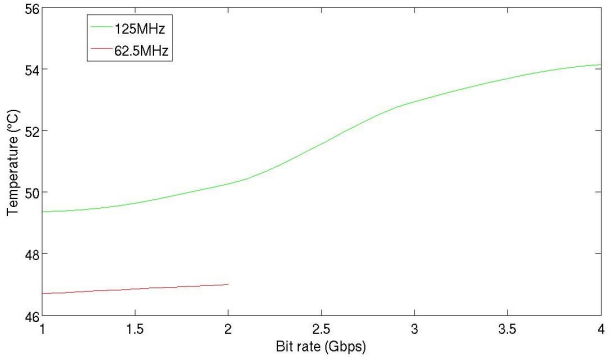


Figure 4. Steady values of temperature for different values of the bitrate.

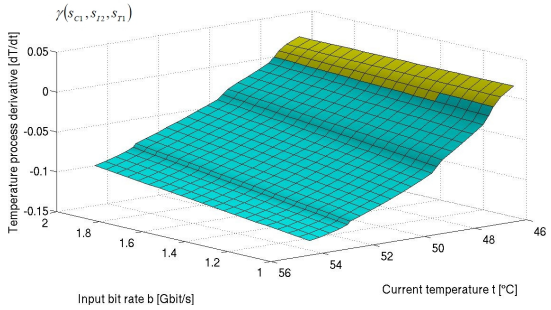
Table I. Component values using 62.5 MHz and 125 MHz clock frequency.

Energy Component	62.5 MHz	125 MHz
Power consumed by unconnected NetFPGA card: $P_C(f_C)$	5.8315 W	7.1683 W
Power consumed per connected Ethernet port: $P_E(f_C)$	1.10 W	1.10 W
Per-Packet processing energy: $E_p(f_C)$	104.23 nJ	181.07 nJ
Per Byte receive-side energy: $E_r(f_C)$	1.1310 nJ	1.1372 nJ
Per Byte transmit-side energy: $E_t(f_C)$	2.20 nJ	2.29 nJ

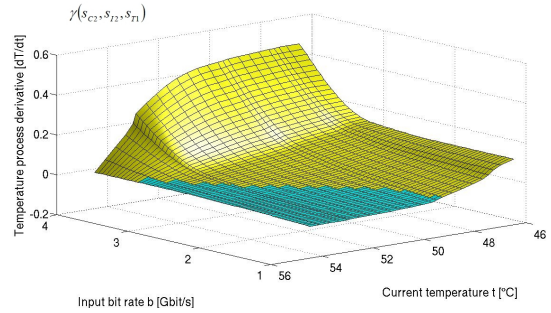
(a) Measure of the per-byte processing energy (receive/process/store a byte) on the ingress Ethernet interface, versus the arrival rate;

(b) Measure of the per-byte processing energy (receive/process/store a byte) on the egress Ethernet interface, versus the service rate.

Table I compares the final results we have obtained for the used frequencies. The results we have obtained for 125 MHz correspond to those obtained in [17]. These occurrences have given the basic idea to manage the NetFPGA hardware platform in order to save energy according to the input load of the Reference Router. A power management policy aimed at this purpose, however, has to take into account the QoS decrease that the use of a lower clock frequency determines.



(a) Clock frequency: 62.5 MHz



(b) Clock frequency: 125 MHz

Figure 5. Temperature process derivative.

In Fig. 3 we have summarized the power consumption model described in (2) characterizing the current NetFPGA Reference Router. The *x*-axis shows the input bit rate, whereas the *y*-axis the power consumption in Watt. The red line indicates the power consumption for 62.5 MHz whereas the green line shows that for 125 MHz. Specifically, when the bit rate is lower than or equal to 2 Gbit/s (in such a case the clock frequency of 62.5 MHz may be used with no loss), the consumed power is lower than 11 W; for bit rates higher than 2 Gbit/s, the clock frequency of 125 MHz has to be used, and the power consumption goes up to 14 W. More details about such measurements produced by our analysis can be found in [16].

B. Temperature measurement

To measure the temperature, we have used a thermal diode on the NetFPGA CPU surface. The temperature we have measured depends on both the network traffic processed and the clock frequency. We have calculated, for each value of the clock frequency and for different values of the input bit rate, the steady temperature, shown in Fig. 4. Moreover, for each pair (bit rate *B*, temperature *t*), we measured the derivative of the temporal temperature behavior to reach the steady temperature value when, starting from an initial temperature *t*, we load the router with an input traffic with bit rate *B*. Figs. 5a and 5b show such values when the clock frequency is 62.5 MHz and 125 MHz, respectively.

According to the steady temperature value shown in Fig. 4, the slope could be either positive (if the starting temperature *t* is lower than the steady value) or negative (in the other way around). For example, if the starting temperature is 46.7 °C and

the input bit rate is $B = 4$ Gbit/s, then the current temperature will grow up to the steady value of the state with bit rate equal to 4 Gbit/s, i.e. 54.14°C. Conversely, in the same clock frequency and bitrate conditions, if the starting temperature is 55 °C, then the current temperature will have to decrease in order to reach the same steady temperature value.

IV. MARKOV MODEL

In this section we define a discrete-time model of the temperature behavior of the CPU residing on the considered router. Since it depends on both the clock frequency and the input traffic bit rate, we use a 4-dimensional Markov model whose state is $S^{(\Sigma)}(n) = \left(S^{(C)}(n), S^{(I)}(n), S^{(T)}(n), S^{(S)}(n)\right)$, where:

- $S^{(C)}(n) \in \Im^{(C)}$ is the clock frequency process at the generic slot n;
- $S^{(I)}(n) \in \Im^{(I)}$ represents the quantized input traffic bit rate at the generic slot n;
- $S^{(T)}(n) \in \Im^{(T)}$ is the temperature at the generic slot n;
- $S^{(S)}(n) \in \Im^{(S)}$ is the indicator variable of a switch at the generic slot n: $S^{(S)}(n) = 1$ if a clock switch occurred in the slot n.

The sets $\Im^{(C)}$, $\Im^{(I)}$, $\Im^{(T)}$ and $\Im^{(S)}$ are the state spaces for the four above state variables.

In the NetFPGA router case, $\Im^{(C)} = \{62,125\}$ MHz; the set $\Im^{(S)}$ is, by definition, $\Im^{(S)} = \{0,1\}$. The set $\Im^{(I)}$ contains the considered quantized input traffic values (in the case study described in Section V $\Im^{(I)} = \{0.5,1.0,1.5,2.0,2.5,3.0,3.5,4.0\}$ Gbps). The set $\Im^{(T)}$ is constituted by evenly spaced values ranging from the minimum to the maximum values the temperature can assume when the router works. These values are chosen according to the measures shown in Section III.B.

Now, in order to define the system model, we have to decide the slot duration and the time diagram of each slot. As far as the slot duration is concerned, we use the duration of a clock frequency switch, and we will indicate it as Δ. In our case $\Delta = 2$ ms. In order to define the model time diagram, we consider two generic states: $s_{\Sigma 1} = (s_{C1}, s_{I1}, s_{T1}, s_{S1})$ in the slot n, and $s_{\Sigma 2} = (s_{C2}, s_{I2}, s_{T2}, s_{S2})$ in the slot $n+1$. Fig. 6 represents the time diagram of the generic slot. We assume the following event sequence:

1. The first action at the beginning of the slot $n+1$ is the evaluation of the new value of the input traffic bit rate. This value is obtained sampling the bit rate values smoothed with an EWMA filter with a time constant equal to the time slot.
2. Then, according to the new value of the input traffic bit rate, the Governor decides the clock frequency for the new slot. As discussed so far, if a clock frequency modification occurs, since a slot interval is needed to apply the clock frequency switch, the new clock frequency will be actually operative from the slot $n+2$;

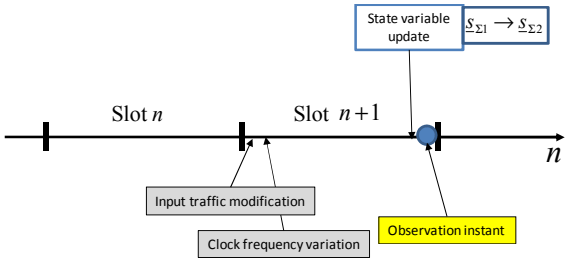


Figure 6. Model time diagram

during the slot $n+1$ the router is frozen and all the input traffic arriving in this slot is lost.

3. Then, at the end of the slot $n+1$, the system state variables are observed.

Now we can define the generic element of the state transition probability matrix as follows:

$$Q^{(\Sigma)}_{[s_{\Sigma 1}, s_{\Sigma 2}]} = \text{Prob}\left\{S^{(\Sigma)}(n+1) = s_{\Sigma 2} \middle| S^{(\Sigma)}(n) = s_{\Sigma 1}\right\} = \\ = Q^{(I)}_{[s_{I1}, s_{I2}]} \cdot \eta^{(C)}_{[s_{C1}, s_{C2}]}(s_{I2}) \cdot Q^{(T)}_{[s_{T1}, s_{T2}]}(s_{C2}, s_{I2}) \cdot \vartheta^{(S)}_{[s_{S2}]}(s_{C1}, s_{C2}) \quad (3)$$

where:

- $\vartheta^{(S)}_{[s_{C1}, s_{C2}]}$ is a clock switch indicator function. It is defined as follows:

$$\vartheta^{(S)}_{[s_{S2}]}(s_{C1}, s_{C2}) = \begin{cases} 1 & \text{if } (s_{C2} \neq s_{C1} \text{ and } s_{S2} = 1) \\ & \text{or } (s_{C2} = s_{C1} \text{ and } s_{S2} = 0) \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Its goal is to set (3) to 0 when the values of s_{C1} and s_{C2} are not compatible with the state s_{S2} (i.e. they are equal to each other, but $s_{S2} = 1$, or different but $s_{S2} = 0$).

- $\eta^{(C)}_{[s_{C1}, s_{C2}]}(s_{I2})$ is a boolean function depending on the clock frequency switching law used by the Governor to decide the clock according to the input traffic bit rate. Its goal is to set equation (3) to 0 when the value of s_{C2} is different from the clock frequency the Governor policy would choose according to the input values s_{I2} and s_{C1}. In other words, $\eta^{(C)}_{[s_{C1}, s_{C2}]}(s_{I2}) = 1$ only if the Governor policy decides to use the clock frequency s_{C2} when the clock frequency and the input traffic bit rate are s_{C1} and s_{I2}, respectively. Otherwise, $\eta^{(C)}_{[s_{C1}, s_{C2}]}(s_{I2}) = 0$. In our case study, we have:

$$\eta^{(C)}_{[s_{C1}, s_{C2}]}(s_{I2}) = \\ = \begin{cases} 1 & \text{if } s_{C2} = 125 \text{ MHz and } s_{I2} > 2\text{Gbit/s} \\ 1 - p_G(s_{I2}) & \text{if } s_{C1} = s_{C2} = 125 \text{ MHz and } \\ & s_{I2} \leq 2\text{Gbit/s} \\ p_G(s_{I2}) & \text{if } s_{C1} = 125 \text{ MHz and } \\ & s_{C2} = 62.5 \text{ MHz and } s_{I2} \leq 2\text{Gbit/s} \\ 1 & \text{if } s_{C1} = 62.5 \text{ MHz and } \\ & s_{C2} = 62.5 \text{ MHz and } s_{I2} \leq 2\text{Gbit/s} \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

The term $p_G(s_{I2})$ is the probability that the Governor decides to switch down the clock frequency from 125 MHz to 62.5 MHz when the input traffic is not greater than 2 Gbit/s. As said in Section II, it is adaptive with the current value of the input bit rate. The design of this function will be discussed in Section V.

- $Q^{(I)}$ is the state transition probability matrix for the quantized input traffic. It is an input of the problem, because it characterizes the traffic crossing the router;
- $Q^{(T)}(s_{C2}, s_{I2})$ is the state transition probability of the temperature. It will be calculated below.

Let $s_{\Sigma 1} = (s_{C1}, s_{I1}, s_{T1}, s_{S1})$ be the state at the end of the previous slot, and let us indicate the array containing the derivative of the temporal temperature behavior when the system state is $s_{\Sigma 1}$ as $\gamma(s_{C2}, s_{I2}, s_{T1})$. In our case study, Fig. 5a and 5b represent the value of that function for each value $t = s_{T1}$ of the current temperature, and $b = s_{I1}$ of the given input bit rate b, for both the values of clock frequency 62.5 MHz and 125 MHz, respectively. Thus, the new CPU temperature in the next slot $n+1$ is:

$$T_2 = s_{T1} + \gamma(s_{C2}, s_{I2}, s_{T1}) \cdot \Delta \tag{6}$$

Therefore, the new state of the temperature, s_{T2}, will be one of the two most adjacent states to T_2 belonging to $\Im^{(T)}$. Let us indicate the most adjacent state with a temperature greater than T_2 as $\lceil T_2 \rceil$, and the most adjacent state with a temperature lower than T_2 as $\lfloor T_2 \rfloor$. The new temperature state s_{T2} will be either $s_{T2} = \lceil T_2 \rceil$ or $s_{T2} = \lfloor T_2 \rfloor$ with a probability dependent on the distance between the real temperature calculated as in (6) and the temperature associated to the adjacent states $\lceil T_2 \rceil$ and $\lfloor T_2 \rfloor$. More specifically:

$$S^{(T)}(n+1) = \begin{cases} \lceil T_2 \rceil & \text{with prob}: \ (T_2 - \lfloor T_2 \rfloor)/(\lceil T_2 \rceil - \lfloor T_2 \rfloor) \\ \lfloor T_2 \rfloor & \text{with prob}: \ (\lceil T_2 \rceil - T_2)/(\lceil T_2 \rceil - \lfloor T_2 \rfloor) \end{cases} \tag{7}$$

Now, from the matrix $Q^{(\Sigma)}$ we can derive the system steady-state probability array $\underline{\pi}^{(\Sigma)}$ by solving the following system:

$$\begin{cases} \underline{\pi}^{(\Sigma)} Q^{(\Sigma)} = \underline{\pi}^{(\Sigma)} \\ \underline{\pi}^{(\Sigma)} \cdot \underline{1}^T = 1 \end{cases} \tag{8}$$

where $\underline{1}^T$ is a column array with all the elements equal to one. Its generic element, $\pi^{(\Sigma)}_{[s_\Sigma]}$, is the steady-state probability of the state $\underline{s}_\Sigma = (s_C, s_I, s_T, s_S)$.

Now, we can derive the marginal steady-state probability array for the temperature process and its mean value:

$$\pi^{(T)}_{[s_T]} = \sum_{\forall s_C \in \Im^{(C)}} \sum_{\forall s_I \in \Im^{(I)}} \sum_{\forall s_S \in \Im^{(S)}} \pi^{(\Sigma)}_{[s_C, s_I, s_T, s_S]} \quad \text{and}$$
$$E\{\underline{\pi}^{(T)}\} = \sum_{\forall s_T \in \Im^{(T)}} s_T \cdot \pi^{(T)}_{[s_T]} \tag{9}$$

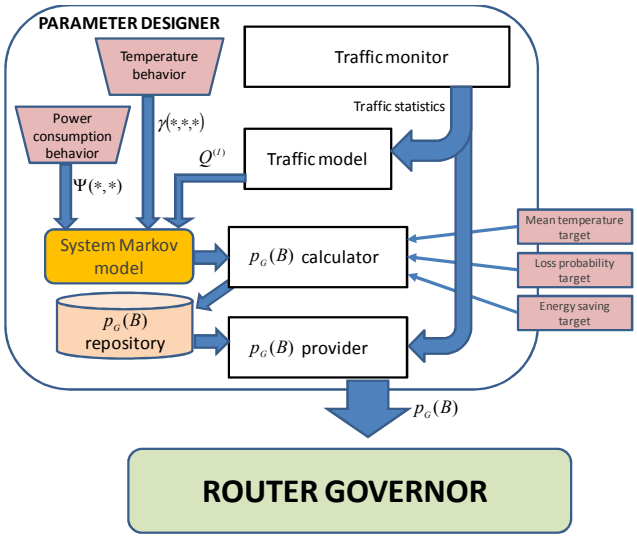


Figure 7. Switching probability function $p_G(B)$ calculation scheme

Let us now calculate the probability of loss occurring during the switching slots. It is defined as:

$$P^{(Loss)} = \lim_{m \to +\infty} \frac{L(m)}{V(m)} \tag{10}$$

where $L(m)$ and $V(m)$ are the cumulative number of lost bits and arrived bits in m consecutive slots. Equation (10) can be rewritten as follows:

$$P^{(Loss)} = \lim_{m \to +\infty} \frac{L(m)}{m} \frac{m}{V(m)} = \frac{\overline{L}}{\overline{V}} \tag{11}$$

The term $\overline{V}$ is the mean value of arrived bits per slot, and can be calculated from the input bit rate traffic statistics as follows:

$$\overline{V} = \sum_{s_\Sigma \in \Im^{(\Sigma)}} s_I \pi^{(\Sigma)}_{[s_\Sigma]} \tag{12}$$

The term $\overline{L}$ represents the mean value of bits lost per slot. Since in our case bits are lost only during clock frequency switches, we have:

$$\overline{L} = \sum_{s_C \in \Im^{(C)}} \sum_{s_I \in \Im^{(I)}} \sum_{s_T \in \Im^{(T)}} s_I \pi^{(\Sigma)}_{[s_C, s_I, s_T, 1]} \tag{13}$$

Finally, we can derive the energy saving percentage as follows:

$$\rho = \frac{P_{MAX} - E\{p\}}{P_{MAX}} \cdot 100\% \tag{14}$$

where P_{MAX} is the power consumed when no saving policy is applied by the Router Governor, while $E\{p\}$ is the mean value of the consumed power calculated as follows:

$$E\{p\} = \sum_{\forall s_C \in \Im^{(C)}} \sum_{\forall s_I \in \Im^{(I)}} \Psi(s_C, s_I) \cdot \sum_{\forall s_T \in \Im^{(T)}} \sum_{\forall s_S \in \Im^{(S)}} \pi^{(\Sigma)}_{[s_C, s_I, s_T, s_S]} \tag{15}$$

The term $\Psi(s_C, s_I)$ represents the power consumed when the router is loaded with an input traffic bit rate of s_I and the clock frequency is s_C.

V. SYSTEM PARAMETER DESIGN

In this section we apply the proposed model to design the switching probability function $p_G(B)$ used by the Router Governor to decide whether to switch or not according to the current bit rate, B ($B \leq 2$ Gbit/s). The working scheme is depicted in Fig. 7.

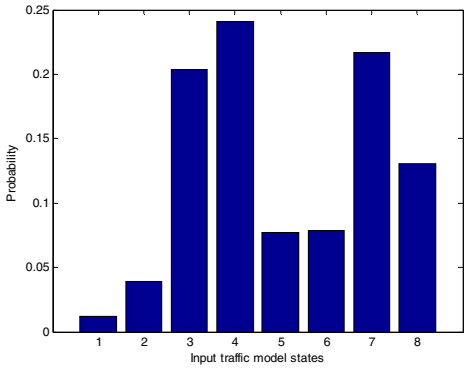
According to this scheme, traffic is continuously monitored from the *Traffic Monitor* block, whose task is to measure first- and second-order statistics of the incoming traffic, in terms of probability density function and autocorrelation function. These statistics are passed to both the *Traffic model* block and to the $p_G(B)$ *provider* block. The first one, solving an inverse eigenvalue problem [21-22], derives the input traffic model, providing the *System Markov Model* block with the matrix $Q^{(I)}$. This last block implements the Markov model described in Section IV: using the temperature behavior derivative $\gamma(*,*,*)$ and the power consumption behavior $\Psi(*,*)$ described in Section III, it provides the $p_G(B)$ *calculator* block with a tool to derive the best set of $p_G(B)$ values satisfying the system targets. The achieved values are stored in the $p_G(B)$ *repository* block. The $p_G(B)$ *provider* block, according to the traffic statistics received by the *Traffic Monitor* block, searches for the $p_G(B)$ set associated to the current traffic statistics, and passes it to the Router Governor to decide the clock frequency in the next slot. Let us note that the $p_G(B)$ *calculator* block requires some second to calculate a $p_G(B)$ set, but this is needed only for a new set of traffic statistics. This occurrence can affect router performance, but this can happen during the router startup phase only. When the repository is well populated, the $p_G(B)$ *provider* block finds the $p_G(B)$ set available in the repository, and therefore it has not to wait the work of the $p_G(B)$ *calculator* block.

Let us now describe how the $p_G(B)$ *calculator* block works, using a case study. To this purpose we installed our NetFPGA green router as the Internet access router of the DIEEI Lab at the University of Catania. Quantizing the traffic in eight different bit rate levels, ranging from 0.5 Gbit/s to 4 Gbit/s with steps of 0.5 Gbit/s, the traffic transition probability matrix produced at a given instant by the *Traffic model* block was a tri-diagonal matrix whose main diagonal, superior and inferior pseudo-diagonals are shown in Table II. Fig 8 shows the traffic statistics received as input by the *Traffic monitor* block. The considered traffic has a mean value of 2.54 Gbit/s and a standard deviation of 0.965 Gbit/s.

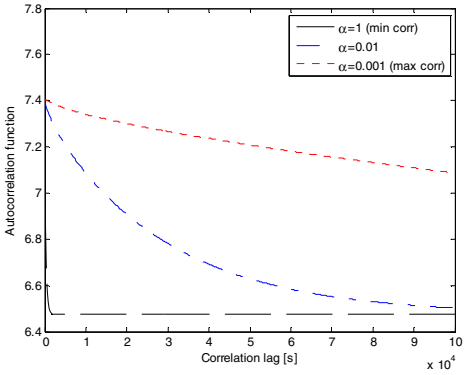
Moreover, in order to analyze the impact of the traffic correlation on the achieved performance, we considered two more cases of input traffic, characterized by transition probability matrices derived from the one listed in Table II by multiplying the terms of the pseudo-diagonals by a coefficient $\alpha = 10^{-2}$ and $\alpha = 10^{-3}$. The terms of the main diagonals are then calculated such that the sum of each row is equal to one. In this way first-order statistics remained unchanged, while traffic becomes more correlated for decreasing values of α .

Table II: Non-null elements of the Input traffic transition probability matrix

INFERIOR PSEUDO-DIAGONAL		MAIN DIAGONAL		SUPERIOR PSEUDO-DIAGONAL	
Pos	**Value**	**Pos**	**Value**	**Pos**	**Value**
		(1,1)	9.9990e-001	(1,2)	1.0000e-004
(2,1)	3.1569e-005	(2,2)	9.9993e-001	(2,3)	3.5098e-005
(3,2)	6.7811e-006	(3,3)	9.9994e-001	(3,4)	4.8774e-005
(4,3)	4.1255e-005	(4,4)	9.9995e-001	(4,5)	6.3636e-006
(5,4)	1.9848e-005	(5,5)	9.9994e-001	(5,6)	3.8975e-005
(6,5)	3.8314e-005	(6,6)	9.9992e-001	(6,7)	3.8609e-005
(7,6)	1.3970e-005	(7,7)	9.9990e-001	(7,8)	8.6030e-005
(8,7)	1.4286e-004	(8,8)	9.9986e-001		



(a) Probability density function



(b) Autocovariance function

Figure 8. Input traffic first- and second-order statistics

The task of the $p_G(B)$ *calculation block* is not easy, because it has to solve an optimization problem run-time. However, its solution is an implementation detail which is out of the scope of this paper (for example, in our implementation we used a genetic algorithm). However, for the sake of clarity, in the following we describe the brute force solution of this problem. According to this algorithm, all the possible tuples of values of the $p_G(B)$ array, for each $B \in \{0.5, 1.0, 1.5, 2.0\}$ Gbit/s are compared. More specifically, we applied the model for each value of the 4-uple $[p_G(0.5\,\text{Gbit/s}), p_G(1.0\,\text{Gbit/s}), p_G(1.5\,\text{Gbit/s}), p_G(2\,\text{Gbit/s})]$ obtained by varying each

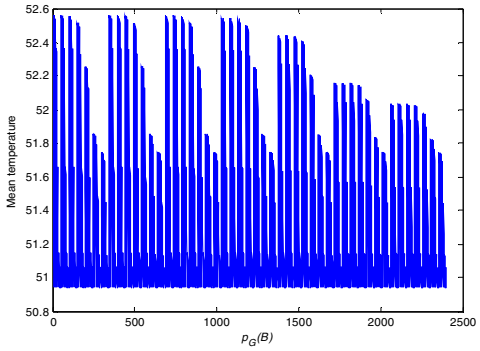


Figure 9. Mean temperature calculated for each $p_G(B)$ set

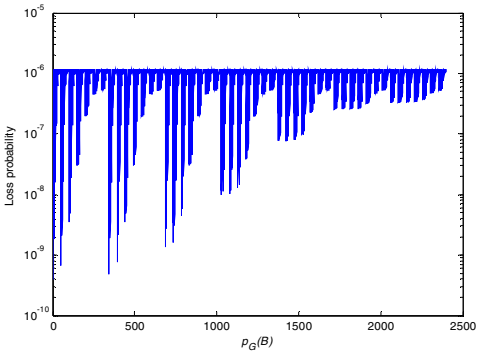


Figure 10. Loss probability calculated for each $p_G(B)$ set

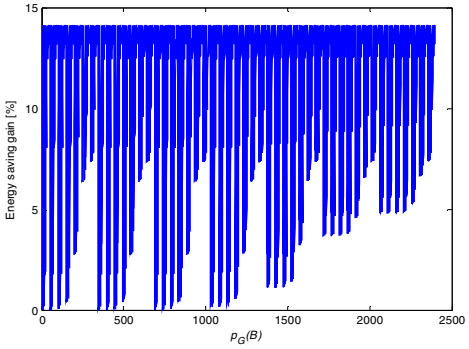


Figure 11. Energy saving gain calculated for each $p_G(B)$ set

element of the array in the set $\{10^{-9},10^{-8},10^{-7},\ldots,10^{-3}\}$. Therefore we applied the model for 2401 times. Figs. 9, 10 and 11 show system performance in terms of mean temperature, loss probability and energy saving gain, for each of the 2401 sets of $p_G(B)$, calculated for the original traffic matrix, i.e. the one characterized by the values listed in Table II. Then the above figures have been combined using two of them at a time, obtaining the plots in Figs. 12 and 13. These figures allow the $p_G(B)$ *calculation block* to calculate $p_G(B)$ according to the given requirements. For example, if the mean temperature target is of 52°C, from Fig. 12 we can deduce that the minimum loss probability is $3.5\cdot10^{-7}$. However, the lower the loss probability, the lower the energy saving gain, as shown in Fig. 13. In particular, using Fig. 13, we can find the working point that satisfies all the requirements. If, for example, the loss probability target is equal to $8.0\cdot10^{-7}$, we have an energy saving gain of 10%. The values of the $p_G(B)$ array characterizing the working point chosen as described so far are the result of the $p_G(B)$ *calculation block*.

VI. Conclusions and Future Work

In this paper we have proposed an analytical discrete-time Markov model that allows green router designers to control the temperature statistics of a router, for given input traffic statistics and energy saving law applied by the Router Governor. Moreover, the model allows the manufacturers to evaluate the energy saving gain which is possible to obtain. In the case study we have shown how the model can be used to design the Router Governor parameters to achieve the target of maintaining the mean temperature below a given threshold. The future directions that we will pursue are related to an extension of the model to capture both the dependence of the power on the environmental temperature, and a Governor policy based on the behavior of input and output queues. Moreover, we will take into account the leakage power and how that affects router energy consumption.

Acknowledgement

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n. 257740 (Network of Excellence "TREND").

References

[1] P. Barford, J. Chabarek, C. Estan, J. Sommers, D. Tsiang, and S. Wright, "Power awareness in network design and routing," in Proc. of IEEE INFOCOM 2008, Phoenix, USA, April 2008, 2008.

[2] Ipmon Sprint, "The applied research group. http://ipmon.sprint.com/,"2007.

[3] A. P. Jardosh, *et al.*,"Towards an energy-star wlan infrastructure," in Proc. ofthe 8thIEEE Workshop on Mobile Computing Systems andApplications, Washington, DC, USA 2007, pp. 85–90.

[4] The Climate Group, "The climate group: Global e-sustainability initiativereport, smart 2020 enabling low carbon economy in the informationage, http://www.smart2020.org/publications/," 2008.

[5] M. Gupta and S. Singh, "Greening of the Internet," in Proc.of the 2003 conference on Applications, technologies, architectures,and protocols for computer communications, SIGCOMM '03, New York, NY, USA:

[6] S. Nedevschi, *et al.*, "Reducing network energy consumption via sleeping and rateadaptation," in Proc. of the 5th USENIX Symposium onNetworked Systems Design and Implementation, 2008.

[7] Econet, "http://www.econet-project.eu/," 2010.

[8] Trend, "http://www.fp7-trend.eu/," 2010.

[9] Greentouch, "http://www.greentouch.org/," 2011.

[10] Cisco, "Ciscoenergywise,http://www.cisco.com/," 2009.

[11] R. Bolla, *et al.*, "Energy efficiencyin the future internet: A survey of existing approaches and trends inenergy-aware fixed network infrastructures," in IEEE Comm. and Tutorials (COMST) 2011.

[12] DC. Hu, C. Wu, W. Xiong, B. Wang, J. Wu, and M. Jiang, "On thedesign of green reconfigurable router toward energy efficient internet, in Commun. Magazine, IEEE, v. 49, n. 6, 2011, pp. 83–87.

[13] Massoud Pedram Shahin Nazarian, "Thermal Modeling, Analysis and Management in VLSI Circuits: Principles and Methods", Proceedings of the IEEE, vol 94, Issue 8, pages 1487-1501, 2006.

[14] G. Gibb, J. W. Lockwood, J. Naous, P. Hartke, N. McKeown, "NetFPGA – An Open Platform for Teaching How to Build Gigabit-Rate Network Switchs and Routers", IEEE Transactions on Education, Vol. 51, N. 3, August 2008.

[15] http://netfpga.org/foswiki/bin/view/NetFPGA/OneGig/ReferenceRouterWalkthrough

[16] A. Lombardo, C. Panarello, D. Reforgiato, G. Schembra, "Power control and management in the NetFPGA Gigabit Router,"Proc. of Future Network & Summit 2012, 4 - 6 July 2012, Berlin, Germany.

[17] V. Sivaraman, A. Vishwanath, Z. Zhao, and C. Russel, "Profiling perpacket and per-byte power consumption in the netfpga gigabit router," IEEE INFOCOM Workshop on Green Communications and Networking (GCN), 2011.

[18] Bit rate input output ewma calculator, http://netfpga.org/foswiki/bin/view/NetFPGA/OneGig/InputOutputEwmaBitrate

[19] Alfio Lombardo, Diego Reforgiato, Vincenzo Riccobene, Giovanni Schembra, "NetFPGA Hardware Modules for Input, Output and EWMA Bit-Rate Computation", International Journal of Future Generation Communication and Networking, June 2012.

[20] A. Vishwanath, Zhi Zhao, Vijay Sivaraman, Craig Russell, "An Empirical Model of Power Consumption in the NetFPGA Gigabit Router", IEEE 4th International Symposium on Advanced Networks and Telecommunications Systems (ANTS) 2010.

[21] "SMAQ: A measurement-based tool for traffic modeling andqueuing analysis Part II: Network applications," *IEEE Commun. Mag.*, vol. 36, p. 66, Aug. 1998.

[22] "SMAQ: A measurement-based tool for traffic modeling andqueuing analysis Part II: Network applications," *IEEE Commun. Mag.*,vol. 36, p. 66, Aug. 1998

[23] A. Lombardo, G. Morabito, G. Schembra "Modeling Intramedia and Intermedia Relationships in Multimedia Network Analysis through Multiple Time-scale statistics,". IEEE Transactions on Multimedia,vol. 6, pp. 142-157 ISSN: 1520-9210.

[24] L. Benini, G. Paleologo, A. Bogliolo, and G. De Micheli, "Policy optimization for dynamic power management," IEEE Trans on Computer-Aided Design, Vol. 18, pp.813-833, Jun. 2001.

[25] T. Simunic, L. Benini, and G. De Micheli, "Event-driven power management," IEEE Trans on Computer-Aided Design of Integrated Circuits and Systems, Vol. 20, pp.840-857, Jul. 2001.

[26] H. Jung, and M. Pedram, "Dynamic power management under uncertain information," in Proceedings of Design Automation & Test in Europe (DATE'07), Apr. 2007.

[27] Q. Qiu, Y. Tan and Q. Wu, "Stochastic Modeling and Optimization for Robust Power Management in a Partially Observable System," in Proceedings of Design Automation & Test in Europe (DATE'07), Apr. 2007.

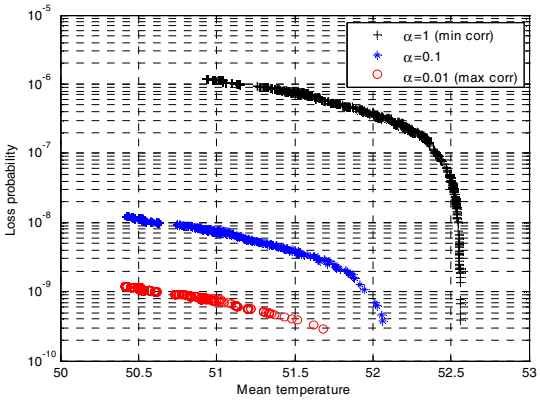


Figure 12. Loss probability vs. mean temperature

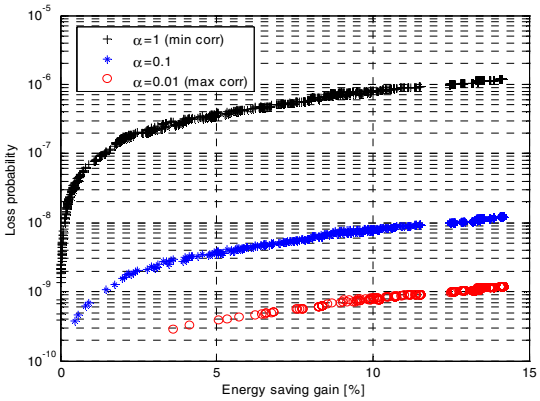


Figure 13. Loss probability vs. energy saving gain

[28] Wei Liu, Ying Tan and Q. Qiu, "Enhanced Q-learning algorithm for dynamic power management with performance constraint", in Proceedings of the Conference on Design, Automation and Test in Europe, pp. 602-605, 2010.