Forecasting-Based Dynamic Virtual Channel Management for Power Reduction in Network-on-Chips

Amir-Mohammad Rahmani[1], Masoud Daneshtalab[1], Ali Afzali-Kusha[1]*, and Massoud Pedram[2]

[1]Dept. of Electrical and Computer Eng., Univ. of Tehran, Tehran, Iran, {am.rahmani, m.daneshtalab}@ece.ut.ac.ir, afzali@ut.ac.ir

[2]Dept. of EE-systems, Univ. of Southern California, Los Angeles, CA90089, pedram@usc.edu

Abstract *— In this paper, a forecasting-based dynamic virtual channel allocation technique for reducing the power consumption of network on chips is introduced. Based on the network traffic as well as past link and total virtual channel utilizations, the technique dynamically forecasts the number of virtual channels that should be active. It is based on an exponential smoothing forecasting method that filters out short-term traffic fluctuations. In this technique, for low (high) traffic loads, a small (large) number of VCs are allocated to the corresponding input channel. To assess the efficacy of the proposed method, the network on chip has been simulated using uniform, transpose, hotspot, NED, and realistic GSM voice codec traffic profiles. Simulation results show that up to a 35% reduction in the buffer power consumption and up to 20% savings in the overall router power consumption may be achieved. The area and power dissipation overheads of the technique are negligible.*

Keywords — Network-on-chips, Dynamic Power Management, Dynamic Virtual Channel Allocation

1 INTRODUCTION

With the advances in the semiconductor technology, it has become possible for designers to integrate tens of Intellectual Property (IP) blocks together with large amounts of embedded memory on a single chip. Integrating all these computational resources (such CPU or DSP cores, video processors) requires demanding communication resources as well [1]. Additionally, scaling of the CMOS feature size into nano-scale regime makes the interconnect delay and power consumption critical parameters in the optimization of the digital integrated circuits. The interconnect optimization in this regime is a challenging task due to the worsening of the crosstalk and electromagnetic interference effects [1].

The network-on-chip (NoC) approach has been proposed as a solution to the complex on-chip communication problems where routers are used between IP cores [2][3][4][5]. Among different components of routers, buffers consume a large amount of dynamic power which increases rapidly as the packet flow throughput increases [6][7]. Increasing the buffer size greatly improves the performance of the interconnection network at the price of a higher power consumption, and hence, the buffer size should be optimized [7]. To reduce the buffer size, one may use the wormhole switching [8].

The latency of data communication in NoCs is a key design parameter which should be minimized. One method to minimize the latency is to use virtual channels (VCs), which provide virtual communication path between routers as the main elements for the data communication in NoCs [9]. The VC decouples buffer resources from the transmission resources, which in turn enables an active message to bypass blocked messages thereby utilizing the available network bandwidth that would otherwise be left idle [9]. Consequently the network throughput is increased by up to 40% over wormhole router without VCs while concurrently the dependence of the throughput on the depth of the network is reduced [9]. Additionally, VCs may prevent the occurrence of deadlock situations [10]. The use of VCs, however, increases the power consumption of NoCs. The power consumption is also a crucial parameter by invoking efficient router designs.In this paper, a dynamic power

management technique for reducing power consumption of NoCs with virtual channels is proposed. The technique optimizes the number of active VCs for the router input channel based on the traffic condition and past link utilization.

The remainder of the paper is organized as follows. In Section 2, we briefly review the related works while Section 3 presents the NoC switch structure used in this work. Section 4 describes the proposed forecasting-based dynamic virtual channels allocation architecture. The simulation results are discussed in Section 5 while Section 6 concludes the paper.

2 RELATED WORK

There have been several previous work results on the VC optimization to save buffers and exploit performance. In [11], the VCs is customized to achieve a 40% buffer saving without any performance degradation. The approach, however, is static, requiring *a priori* detailed analysis of application-specific traffic patterns. A dynamically allocated multiqueue structure for communication on multiprocessor is presented in [12]. Because this architecture has a complex controller, limited channel count, and three-cycle delay for each flit arrival/departure, it does not appear to be a good candidate for NoC's .

In [13], a dynamic VC regulator which allocates a large number of VCs in high traffic conditions, called ViChaR, is suggested. In this technique, the number of VCs in each input channel is variable and depends on the number of flit slots in the input buffer, making the VC allocation a complex task. This scheme does not utilize any history monitoring for the VC allocation. In addition, several tables in the input channel controller of the technique should be updated whenever a flit enters or exits the router. Also, using a shared memory in the ViChaR increases the complexity of the controller further. Because use of the *Unified Buffered Structure* is proposed, a large number of MUX/DEMUX components per input channel are required. To alleviate the problem, the virtual channel buffers are grouped together, further increasing the hardware complexity of the controller. The technique may

not be scalable in terms of the packet size. Finally, the ragraph. technique may not be used for applications requiring variable packet size (e.g., multicasting communication in chip-multi-processing.) ViChaR invokes a two-stage arbitration.

3 SWITCH STRUCTURE

Fig. 1shows the general architecture of the switch which contains two generic modules, namely, an input channel and an output channel. The main structure is based on *RASoC* switch proposed in [14]. We have however made some modifications to its buffering, routing, and flow control parts and added support for Virtual Channels (VCs) based on the work discussed in [9]. In this section, the switch is described in some detail.

3.1 Communication Model

The switch utilizes a handshaking mechanism for its communication i/e/. it communicates with its neighbor switches or cores by sending and receiving request and response messages. Each link (Fig. 1) includes two unidirectional channels in the opposite direction of one another to transmit data, framing, and flow control signals. In addition to *n* bits for the data, two bits are used for the packet framing which are *bop* (begin-of-packet) and *eop* (end-of-packet). The bop is set only at the *packet header* while *eop* is set in the last payload word, which is also the *packet trailer*. Therefore, the variable packet length feature is supported.

3.2 Switching

The switch uses the wormhole packet switching approach [15] where messages are sent by means of packets composed of flits. A flit (flow control unit) which is equal to the physical channel word (or phit – physical unit) has *n*+2 bits. It is the smallest unit over which the flow control is performed.

3.3 Routing and Arbitration

The proposed switch supports different deterministic or adaptive routing algorithms such as XY [14], DyXY [16], and Odd-Even [17] used in a 2-D mesh topology. In addition, exhaustive round-robin

[17] and priority-based [19] arbitration schemes are implemented in the switch. Note that the switch supports the locking mechanism required for the wormhole packet switching.

3.4 Flow Control and VCs Management

Since a handshaking mechanism is used for the communication, when a sender puts some data on the link, it activates the *val* (valid) signal. When the receiver receives the data, it activates the *ack* (acknowledge) signal. In the proposed VC management approach, after the reception of each packet, a VC Controller unit allocates one of the free VCs to this packet and locks the allocated VC until the packet leaves it.

3.5 Input Channel and Output Channel Modules

The input channel module shown in Fig. 2consists of four important units which are the VC Controller, IB (Input Buffer), IC (Input Controller), and IRS (Input Read Switch). Note that signal names at the interface of the router include the prefix "*in_*", for the input channel modules, and "*out_*", for the output ones and all the internal signals connecting the input and output channel modules use the prefix "*x_*" [14]. In Fig. 2, four VCs are used for each input channel. The VC Controller utilizes the handshaking protocols for the flow control (to provide a smooth traffic flow by avoiding buffer overflow and packet drops), allocation/deallocation of each VC to the input flow, and DVCA (Dynamic Virtual Channel Allocation) mechanism which will be explained in Section III. The IB block is a $p\times(n+2)$-bit FIFO buffer which is responsible for storing the flits of the incoming packets while they cannot be forwarded to an output channel. The number of the VCs is n. The IC block of each VC performs the routing function while its IRS block receives *x_rd* and *x_gnt* signals. A bit value of 1 in the *x_gnt* signal shows that which output has been selected while a bit value of 1 in the *x_rd* signal indicates that the output port has completed reading the current flit. Since this bit is connected to the *rd* signal of the IB block, this block transfers its input data to the input of its IC block for sending the next flit to the output module.

The output channel architecture of the proposed switch, which is similar to the RASoC switch [14], is depicted Fig. 3It is composed of four blocks which are OC (Output Controller), ODS (Output Data Switch), ORS (Output Read Switch), and OFC (Output Flow Controller). The OC block runs the arbitration algorithm to select one of the requests sent by the VCs. Then, it activates the grant line of the selected request which induces the proper switching of the ODS and ORS blocks. They connect the *x_din* (the data part of the current flit) and *x_rok* (where each bit of this signal corresponds to *rok* of the corresponding VC IB block and is set when the data is ready at the output of the IB block) signals of the selected virtual channel to the external output channel interface. Note that *x_din* contains the *x_out* of all VCs. ODS and ORS blocks connect the *x_din* and *x_rok* signals of the selected virtual channel to the *Out_data* and *Out_val*, respectively, of the external output channel. *Out_val* is the signal that shows the data on *Out_data* is valid. The OFC block connects the bit with value 1 of *x_rok* to the *Out_val* and sets the proper bit of *x_rd* to 1 when *Out_ack* is received.

4 DYNAMIC VIRTUAL CHANNEL ALLOCATION (DVCA) ARCHITECTURE BASED ON DYNAMIC POWER MANAGEMENT (DPM)

Buffers are the single largest power consumers for a typical switch in an on-chip network [20]. Therefore, reducing the power consumption of the VC input buffers may considerably lower the power consumption of the network. This power minimization should be performed with a minimal degradation of the throughput. In this work, we propose to use a dynamic power management (DPM) technique to dynamically determine the number of active VCs. The DPM technique provides us with the flexibility to precisely tune the trade-off between the power dissipation and the performance. The DPM technique is based on a *distributed forecasting-based policy*, where each router input port predicts its future communication workload and the required virtual channels based on the analysis of the prior traffic.

4.1 Communication Traffic Characterization

Characteristics of the communication traffic in an input channel may be captured by using several potential network parameters. Different traffic parameters such as link utilization, input buffer utilization, and input buffer age have been proposed for simple input channels (without VCs) [21]. None of these parameters (indicators) alone can correctly represent the communication traffic in VC-based input channels. Thus, we employ a combination of these parameters to predict the network load in VC-based input channels. More precisely, we use the link utilization and the virtual channel utilization parameters as the network load indicators as is explained below.

The Link Utilization (LU) parameter is defined as [21]

$$LU = \frac{\sum_{t=1}^{H} A(t)}{H}, \qquad 0 \le LU \le 1 \quad (1)$$

where $A(t)$ is equal to 1, if the traffic passes through the link i in the cycle t and 0 otherwise, and H is the window size. The link utilization is a direct measure of the traffic workload. When the network is lightly loaded or highly congested, the link utilization is low. At low traffic loads, the link utilization is low due to the fact that the flit arrival rate is low. When the network traffic increases, the flit arrival rate between the adjacent routers and the link utilization of each link increases. When the network traffic approaches the congestion point, the number of free buffer spaces in the upstream router will become limited causing the link utilization to start to diminish [21]. This observation reveals that the link utilization alone will not be sufficient for assessing the network traffic. The forecasting-based DVCA policy, therefore, requires more information for making the right decision. In this work, we use the utilization of each virtual channel to complement the link utilization indicator for the proposed forecasting policy.

As mentioned earlier, after receiving each packet, the VC Controller allocates one available VC to the corresponding received packet and locks the VC (by setting L to 1). After the packet leaves the

buffer, the controller releases the VC (by resetting L to 0). The virtual channel utilization tracks how many locks of VC in the router input channel occur.

Let us denote the number of cycles that each packet uses a VC (if it is sent without interrupt) by z where $L(s)$ is set to 1 during these cycles. The virtual channel utilization which is the lock rate of each VC during a window of H cycles is denoted by VCU and calculated as

$$VCU = \frac{\sum_{s=1}^{H} L(s)}{H}, \quad 0 \le VCU \le 1 \quad (2)$$

The overall VCU, denoted by OVCU, is defined as the sum of VCUs of each input channel obtained from

$$OVCU = \frac{\sum_{i=1}^{n} VCU}{n}, \quad 0 \le OVCU \le 1 \quad (3)$$

where n is the number of VCs per input channel. Table I shows a simple example of calculating the virtual channel utilization when H is 5 cycles and n is 4. The data values in each VCx ($1 < x < 4$) column denote the packet number (ID) that locks a given VC at cycle i. The numbers in each Lx column correspond to the locking status of each VCx at that cycle. The VCU of each VC in the window is given in the shaded part of the last row. The last number in each VCx corresponds to the number of packets that used this VC during this window. For this input channel, the $OVCU$ is equal to 15/20 or 3/4.

We use the link utilization as the primary traffic indicator, while the virtual channel utilization is used as a litmus test for detecting the network congestion. Next, we will show the usage of these indicators for the DVCA policy.

4.2 Forecasting-based DVCA Policy

In the proposed technique, the DVCA unit uses the LU and OVCU parameters for measuring the past communication traffic. Based on this analysis, the communication traffic of the next period and the number of required active VCs are determined. To reduce the area and power dissipation overheads

of the proposed solution, we must simplify the forecasting equation. To this end, we combine the two measures using a simple weighted equation given by

$$CT = LU/n + W \times (OVCU - OVCU_{\min}),\ 0 \le W \le 1/2,\ 0 \le CT \le 1 \quad (4)$$

where CT is the communication traffic parameter, W is the forecasting weight, $OVCU_{\min}$ is the sum of all $VCU_{\min}$ (the VC smallest possible lock rate) for each input channel in each history interval. Since $VCU_{\min}$ occurs when there are no stalls for the packets passing through the corresponding VC, $OVCU_{\min}$ is equal to LU/n. In the above example, dividing the last number in each VCx column by $n \times H$, we obtain the $VCU_{\min}$. As this equation implies, the communication traffic is defined as a function of LU and the network load (congestion). The load is obtained by subtracting the actual lock rate from the minimum lock rate of the VCs (multiplied by the coefficient of W.) In this equation, we set W to 0.5 which simplifies Eq. (4) to a straightforward average equation (because $OVCU_{\min} = LU$).

To make the forecasting formula reliable, we use an exponential smoothing function. This is a simple and popular forecasting formula, which is commonly used in programming and inventory control science and is defined as [22]

$$CT_{predict} = CT_{past} + \alpha \times (CT_{actual} - CT_{past}) = \alpha \times CT_{actual} + (1 - \alpha) \times CT_{past} \quad (5)$$

where α is the forecasting weight, $CT_{predict}$ is the predicted communication traffic, CT_{actual} is the actual communication traffic in the current history period, and CT_{past} is the predicted communication traffic in the previous history period. The accuracy of the prediction is a strong function of α, and hence, its value should be selected such that the prediction error is minimized (see Section IV).

The network traffic profile has short-term and long-term fluctuations. The proposed technique in this work filters out the short-term traffic fluctuations, adapting the number of active VCs based on the long-term traffic transitions. Based on the prediction, the controller decides to increase, decrease, or keep the same the number of active VCs. The pseudo-code for our proposed Forecasting-based DVCA policy for the case of n virtual channels per input channel is shown in Fig. 4.

4.3 Hardware Implementation

The hardware realization of the proposed forecasting-based DVCA policy which relies on the local link and buffer information is shown in Fig. 5. Because the communication overhead of relying on global information is avoided, a simple hardware implementation can be invoked. To measure the link utilization (*LU*), a counter at each input port gathers the total number of packets that are passed from the link in each history interval. Similarly, there is a counter for each VC, calculating the number of occurred locks (*VCU*). To calculate the *OVCU*, an adder block is used to sum up the *VCU*s in each input channel. The *CT* Calculator carries out CT using Eq. (4). The Forecasting Unit uses the calculated CT and previously predicted CT from the prior interval (CT_{past}) to predict the new *CT* ($CT_{predict}$) for the next *H* cycle interval. A register stores the $CT_{predict}$ to be used as the CT_{past} in the next interval.

To reduce the hardware overhead, we set α to 12/16, which is very close to the optimal values (see Section IV) of this parameter for the traffic profiles used in this work. We implemented the division and multiplication operations by right and left shifts, respectively.

The Decision Logic unit determines the number of required active virtual channels (*Required_VCs*) using $CT_{predict}$, CT_{past}, and the required number of the virtual channels for the previous history window. Based on the *Required_VCs* value, the number of virtual channels in each input channel may be changed via a clock gating technique where the VC clock is gated. Other power-saving techniques may also be used. Finally, note that we simplify the division and multiplication operations by setting *H* and $H \times n$ values as power-of-two numbers. Otherwise, the overhead of the DVCA unit will become slightly larger. For example, we set both *n* and *H* to 4.

5 Results and Discussion

To assess the efficiency of the proposed technique, we have compared NoCs with the forecasting-based DVCA and conventional virtual channel controllers. The comparison is performed in terms of power and latency for different traffic profiles. We used VHDL to develop six switches with 2, 4, and

8 virtual channels based on the conventional (non-DVCA) and DVCA. The XY routing algorithm was invoked in this study where the simulations were performed for a 6×6 mesh NoC. The switches are labeled as XY-2VCs, XY-2VCs-DVCA, XY-4VCs, XY-4VCs-DVCA, XY-8VCs and XY-8VCs-DVCA, respectively. Also, we set W, α, and H to 1/2, 12/16 (75%), and 4, respectively.

The performance of the network is evaluated using latency curves as a function of the packet injection rate (*i.e.*, the number of packets injected into the network per cycle). The packet latency is defined as the time duration from when the first flit is created at the source core to when the last flit is delivered to the destination core. For each simulation, the packet latencies were averaged over 250,000 packets. Latencies were not collected for the first 30,000 cycles to allow the network to stabilize. It was assumed that the packets had a fixed length of five flits, the buffer size of each virtual channel was five flits, and the data width was set to 32 bits. To perform the simulations, we used uniform, transpose, NED [27], and hotspot traffic patterns [23].

In the uniform traffic pattern, a core sends a packet to any other cores with an equal probability while in the hot spot traffic pattern, messages are destined to a specific node (the core at (3, 3)) with a certain (higher than average) probability and are otherwise uniformly distributed. The NED is a synthetic traffic model based on Negative Exponential Distribution where the likelihood that a node sends a packet to another node exponentially decreases with the hop distance between the two cores. This synthetic traffic profile accurately captures key statistical behavior of realistic traces of communication among the cores. Finally, in the transpose traffic profile, a core at the position (i, j) only send packets to the core at the position ($6 - j$, $6 - i$).

The NoC performances for the two virtual channel management schemes under uniform, hotspot 10%, Transpose, and NED traffics are given in Fig. 6. As is observed from the figure, each pair of DVCA and non-DVCA with 2, 4, and 8 VCs have almost the same performance at low traffic loads. As the load increases, the packet latency rises dramatically due to the network congestion. The results show that the conventional VC controller performs slightly better than the DVCA VC controller. This

originates mainly from the prediction error.

The power consumptions of the switches which are computed by Synopsys Power Compiler using a 0.13µm standard CMOS technology are presented in Fig. 7 . As the results reveal, the average power consumptions of the switches with the DVCA VC controller are considerably lower than those of the corresponding conventional switches at low traffic loads. The reason is that when the DVCA VC controller is used, some of the VCs are clock-gated leading to power saving. As the traffic load increases, the difference between the average power consumptions of the switches with the same number of the VCs decreases until they eventually become nearly the same at the congestion injection rate. Significant power saving is achieved at the price of slightly higher latency for the NoC with the DVCA VC controller. Indeed, summing up the power consumption at each injection rate leads to power saving of 35% in the buffer power consumption and up to 20% savings in the overall router power consumption.

To assess the performance of the proposed technique under more realistic traffic loads, we have used GSM voice CODEC traffic profile [26]. As shown in Fig. 8, the GSM voice CODEC was partitioned into 9 cores mapped manually onto a 6×6 mesh NoC. The communication traces between the cores were recorded for an input voice stream of 1000 frames. Due to the space limitation, only the communications in the first 1000 cycles are shown. The encoder and decoder were partitioned into core_0 – core_4 and core_5 – core_8, respectively. The directed edges between the cores represent communication volumes. For example, the numbers shown on the edge between core_0 and core_1 express that there is a communication volume from core_0 to core_1 with a size of 320 bytes starting from clock cycle 0. The CODEC cores generate packets based to the recorded communication traces. The other cores in the NoC generate packets based on a uniform traffic, with the same average packet injection rate as the CODEC cores. The packet injection rate is increased incrementally to obtain the latency-throughput and the power consumption characteristics shown in Fig. 9, respectively [25]. As seen in Fig. 9, the switch with the conventional VC controller performs

slightly better than the switch with the DVCA VC controller at high traffic loads because of the prediction error. The average power consumptions of the switches with the DVCA VC controller are better than the corresponding conventional switches at low traffic loads.

Next, to see the effect of α on the prediction accuracy, we have used the sum square error (SSE) [24]

$$SSE = \sum_{i=1}^{k} \sum_{j=1}^{H} \left(y_{ij} - \hat{y}_{ij} \right)^2 \quad (6)$$

where y_{ij} is the actual number of the required VCs, $\hat{y}_{ij}$ is the predicted number of the required VCs, and k is the total number of the prediction windows that the simulation has been run. As shown in Fig. 10, for the uniform traffic, the optimum forecasting weights for DVCA-2VCs, DVCA-4VCs, and DVCA-8VCs were equal to 0.89, 0.75 and 0.83, respectively while for the realistic traffic as shown in Fig. 10, the optimum forecasting weights for DVCA-2VCs, DVCA-4VCs and DVCA-8VCs were equal to 0.86, 0.76 and 0.78, respectively. The error, however, will not be much higher than the minimum value if we select α as 0.75 for all the cases.

Finally, to determine the area and power overheads of the proposed technique, we synthesized the DVCA and conventional switches using the Synopsys Design Compiler. Note that the controller is not on the critical path of the router, and hence, its delay overhead does not need to be considered. The synthesis results which are obtained for a 0.13μm standard CMOS technology are reported in Table 2 and Table 3. The figures given in these tables reveal that the area and power overheads of the proposed forecasting-based DVCA VC controller are negligible.

6 Conclusion

In this paper, we introduced a forecasting-based dynamic power management technique for controlling the number of active virtual channels (VCs). The approach makes use of the link and VC utilizations in predicting the communication traffic. Based on the predicted traffic, the number of the active virtual channels may be increased, decreased, or remain the same. The clock-gating power

management technique is used to activate/deactivate the VCs. To determine the efficacy of the proposed technique NoCs with conventional and DVCA VC controller for 2, 4, and 8 VCs were simulated. The simulation results which performed for the uniform and transpose traffic profiles showed considerable power savings with a minimum impact on the latency for the proposed technique. The technique was implemented using a simple hardware to make the power and hardware overheads very small.

REFERENCES

[1] J. Hu and R. Marculescu, "Application-specific buffer space allocation for networks-on-chip router design," Proceedings of International Conference on Computer Aided Design (**2004**), pp. 354-361.

[2] L. Benini and G. D. Micheli, "Networks on chips: A new SoC paradigm," IEEE Computer (2**002**) , vol. 35, pp. 70–78.

[3] W.J. Dally et al., "Route Packets, Not Wires: On-Chip Interconnection Networks," Proceedings of Design Automation Conference (**2001**), pp. 684-689.

[4] S. Heo and K. Asanovic, "Replacing global wires with an onchip network: a power analysis," Proceedings of the 2005 International Symposium on Low Power Electronics and Design (**2005**), pp. 369-374.

[5] R. Kumar, V. Zyuban, and D. M. Tullsen, "Interconnections in multi-core architectures: understanding mechanisms, overheads and scaling," Proceedings of the 32nd International Symposium on Computer Architecture (**2005**), pp. 408-419.

[6] W. Hangsheng, L. S. Peh, and S. Malik, "Power-driven design of router microarchitectures in on-chip networks," Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture (**2003**), pp. 105-116.

[7] T. T. Ye, L. Benini, and G. De Micheli, "Analysis of power consumption on switch fabrics in network routers," Proceedings of the 39th Design Automation Conference (**2002**), pp. 524-529.

[8] L. M. Ni and P. K. McKinley. A survey of wormhole routing techniques in direct networks. IEEE Computer (**1993**), 26:62{76.

[9] W. J. Dally, "Virtual-channel flow control," In Proc. of the 17th Annual International Symposium on Computer Architecture (**1990**), pp. 60-68.

[10] W. J. Dally and C. L. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," IEEE Transactions on Computers (**1987**), pp. 547-553.

[11] Ting-Chun Huang, Umit Y. Ogras, and Radu Marculescu, "Virtual Channels Planning for Networks-on-Chip," Proceedings of International Symposium on Quality Electronic Design (**2007**), pp. 879-884.

[12] Y. Tamir and G. L. Frazier, "High-performance multiqueue buffers for VLSI communication Switches," Proceedings of International Symposium on Computer Architecture (**1988**), pp. 343-354.

[13] C.A. Nicopoulos, Dongkook Park, Jongman Kim, N. Vijaykrishnan, M.S. Yousif, and C.R. Das, "ViChaR: A Dynamic Virtual Channel Regulator for Network-on-chip Router," Proceedings of International Symposium on Microarchitecture (**2006**), pp. 333-346.

[14] C.A. Zeferino, M.E. Kreutz, A.A. Susin, "RASoC: A Router Soft-Core for Networks-on-Chip," Proceedings of Design, Automation and Test in Europe Conference and Exhibition (**2004**), pp. 198- 203.

[15] W. J. Dally and C. L. Seitz, "The torus routing chip," Journal of Distributed Computing (**1986**), vol. 1(3), pp. 187-196.

[16] M. Li, Q.-A. Zeng, and W.-B. Jone, "DyXY – a proximity congestion-aware deadlock-free dynamic routing method for network on chip," Proceedings of Design Automation Conference (**2006**), pp. 849–852.

[17] G. M. Chiu, "The odd-even turn model for adaptive routing," IEEE Trans. on Parallel and Distributed Systems (**2000**), 11:729 – 738.

[18] E.S .Shin, V.J. Mooney, G.F. Riley, "Round-robin Arbiter Design and Generation," Proceedings of International Symposium on System Synthesis (**2002**), pp. 243-248.

[19] A. Bystrov, D. J. Kinniment, and A. Yakovlev, "Priority Arbiters," Proceedings of International Symposium on Advanced Research in Asynchronous Circuits and Systems (**2000**), pp. 128-137.

[20] S. Banerjee and N. Dutt, "FIFO Power Optimization for OnChip Networks," Proceedings of the 14th ACM Great Lakes symposium on VLSI (**2004**), pp. 187-191.

[21] L. Shang, L. S. Peh, and N.K. Jha, "Dynamic Voltage Scaling with Links for Power Optimization of interconnection Networks," Proceedings of Ninth International Symposium on High-Performance Computer Architecture (**2003**), pp. 91-102.

[22] R .J. Tersine, Principles of Inventory & Material Management, Fourth Edition, Prentice Hall PTR (**1994**).

[23] M. Rezazad and H. Sarbazi-azad, "The Effect of Virtual Channel Organization on the Performance of Interconnection Networks," Proceedings of 19th IEEE international Parallel and Distributed Processing Symposium (**2005**), pp. 264-271.

[24] Albert H. Bowker and Gerald J. Lieberman, Engineering Statistics, Prentice Hall PTR, 2nd edition (**1972**).

[25] D. Wu, B. M. Al-Hashimi, M. T. Schmitz, "Improving Routing Efficiency for Network-on-Chip through Contention-Aware Input Selection," Proceedings of Asia and South Pacific Conference on Design Automation (**2006**), pp. 36-41.

[26] M. T. Schmitz, B. M. Al Hashimi, and P. Eles, "System level design techniques for energy-efficient embedded systems," Kluwer Academic Publishers (**2004**).

[27] Amir-Mohammad Rahmani, Iman Kamali, Pejman Lotfi-Kamran, Ali Afzali-Kusha, Saeed Safari, “Negative Exponential Distribution Traffic Pattern for Power/Performance Analysis of Network on Chips,” Proceedings of 22nd International Conference on VLSI Design (**2009**), pp.157-162.

FIGURES AND TABLES

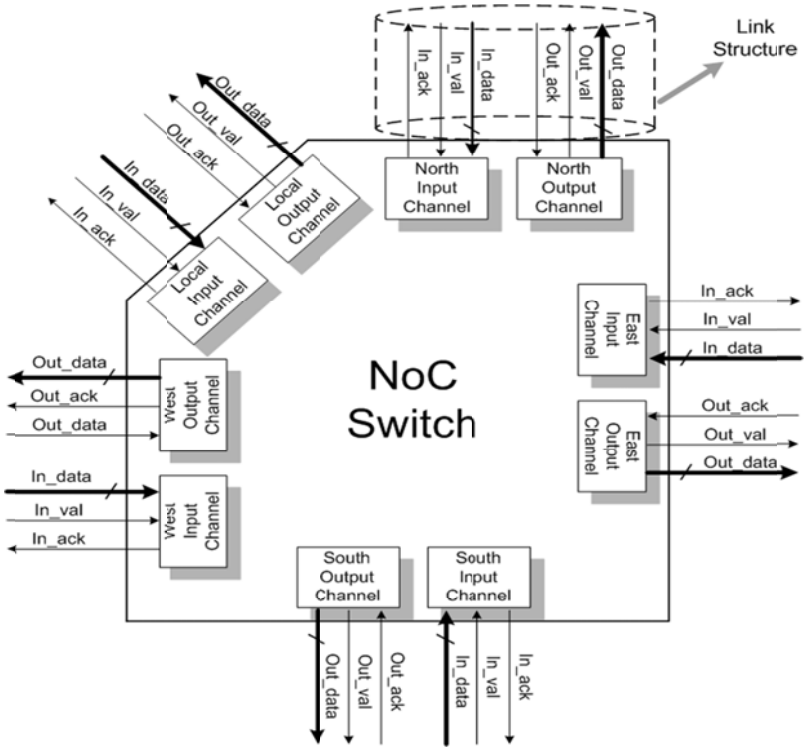


Fig. 1 Proposed switch interface and architecture.

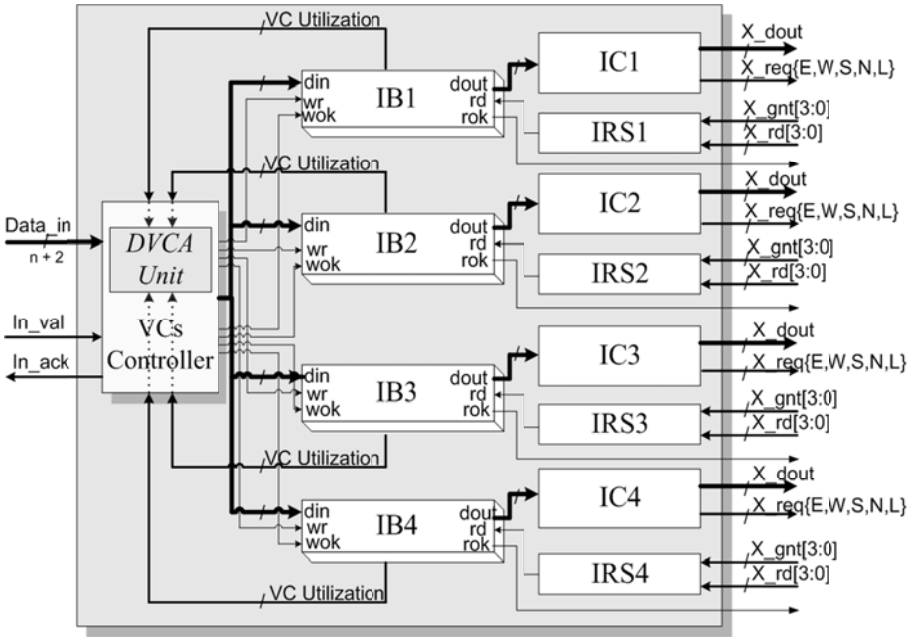


Fig. 2 Input channel architecture.

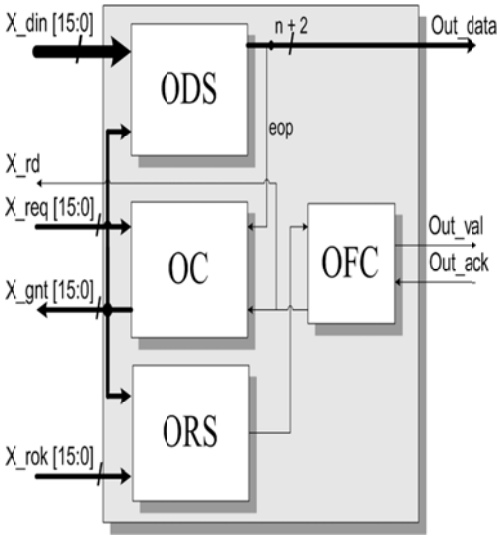


Fig. 3 Output channel architecture.

$CT_{actual} = LU/n + (W \times (OVCU - LU))$
$CT_{predict} = CT_{past} + \alpha \times (CT_{actual} - CT_{past})$
if ($CT_{predict} > CT_{past}$) **then**
  if ($required\_VCs = 1$ **and** $CT_{predict} > (\frac{H \times (1) - 1}{H \times n})$) **then**
    $required\_VCs = required\_VCs + 1$
  else if ($required\_VCs = 2$ **and** $CT_{predict} > (\frac{H \times (2) - 1}{H \times n})$) **then**
    $required\_VCs = required\_VCs + 1$
  …
  …
  else if ($required\_VCs = n - 1$ **and** $CT_{predict} > \frac{H \times (n-1) - 1}{H \times n}$)**then**
    $required\_VCs = required\_VCs + 1$
  end if
else if ($CT_{predict} < CT_{past}$) **then**
  if ($required\_VCs = n$ **and** $CT_{predict} < \frac{n-1}{n}$) **then**
    $required\_VCs = required\_VCs - 1$
  else if ($required\_VCs = n\text{-}1$ **and** $CT_{predict} < \frac{n-2}{n}$) **then**
    $required\_VCs = required\_VCs - 1$
  …
  …
  else if ($required\_VCs = 2$ **and** $CT_{predict} > \frac{1}{n}$) **then**
    $required\_VCs = required\_VCs - 1$
  end if
end if
$CT_{past} = CT_{predict}$

Fig. 4 Pseudo-code for our proposed Forecasting-based DVCA policy.

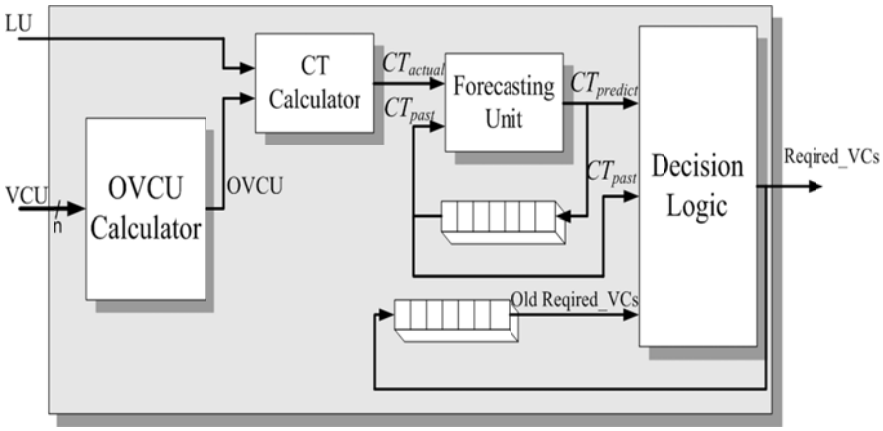
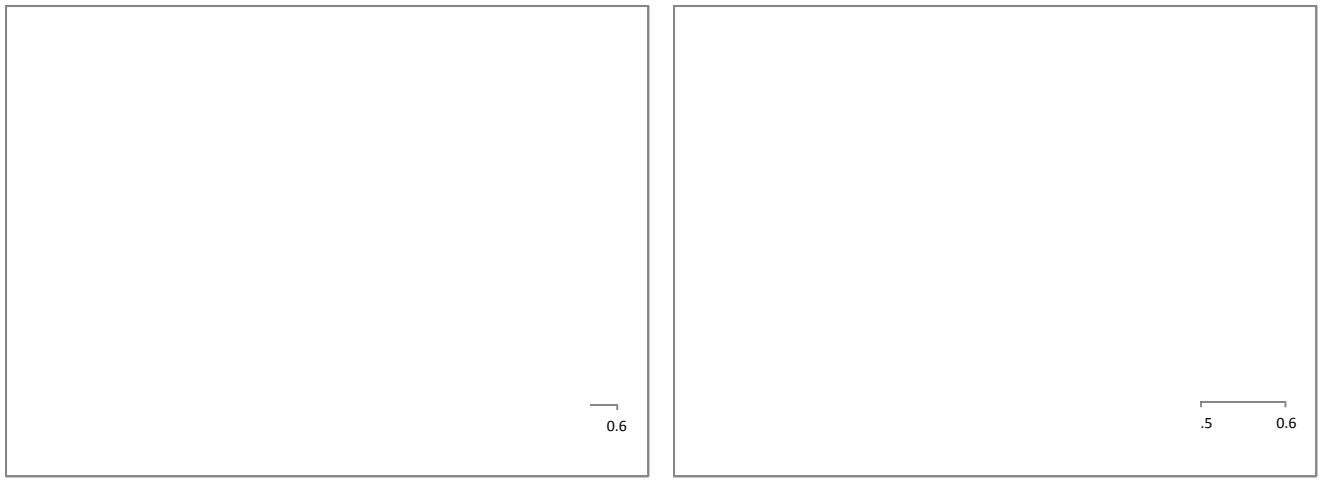
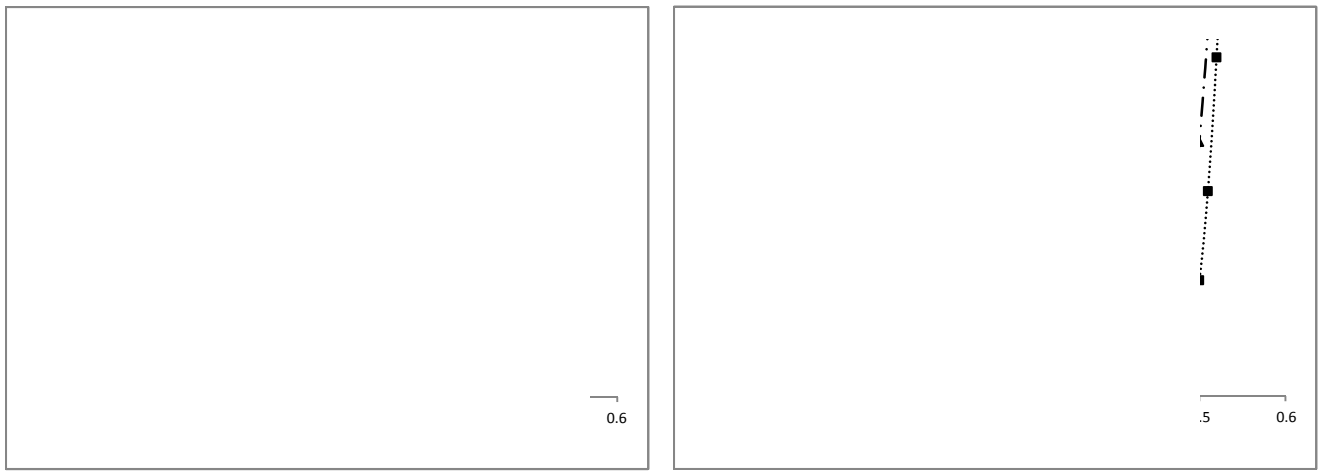


Fig. 5 The hardware diagram of the forecasting-based DVCA policy.

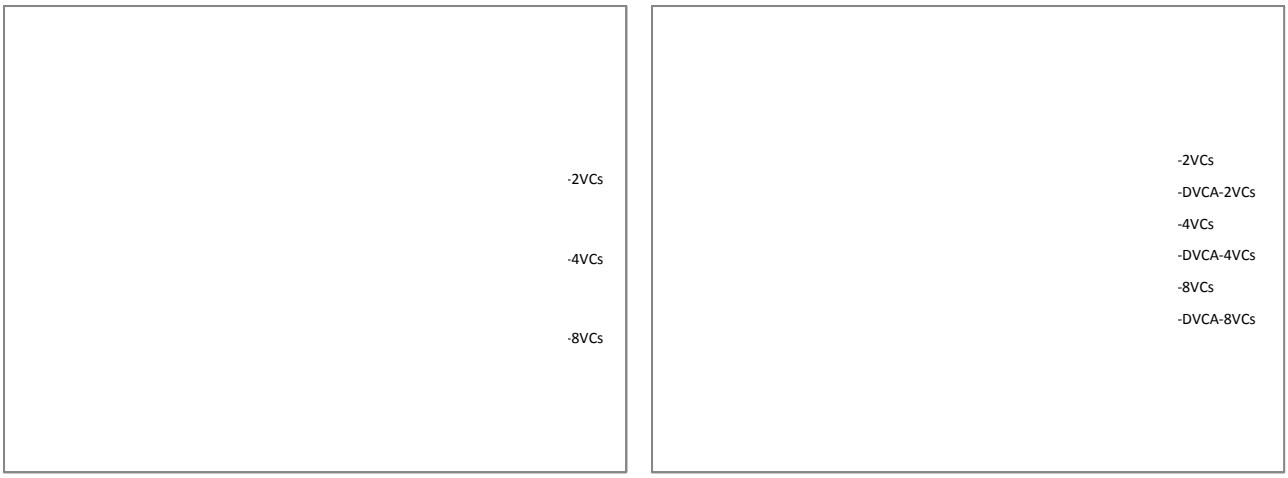


(a)

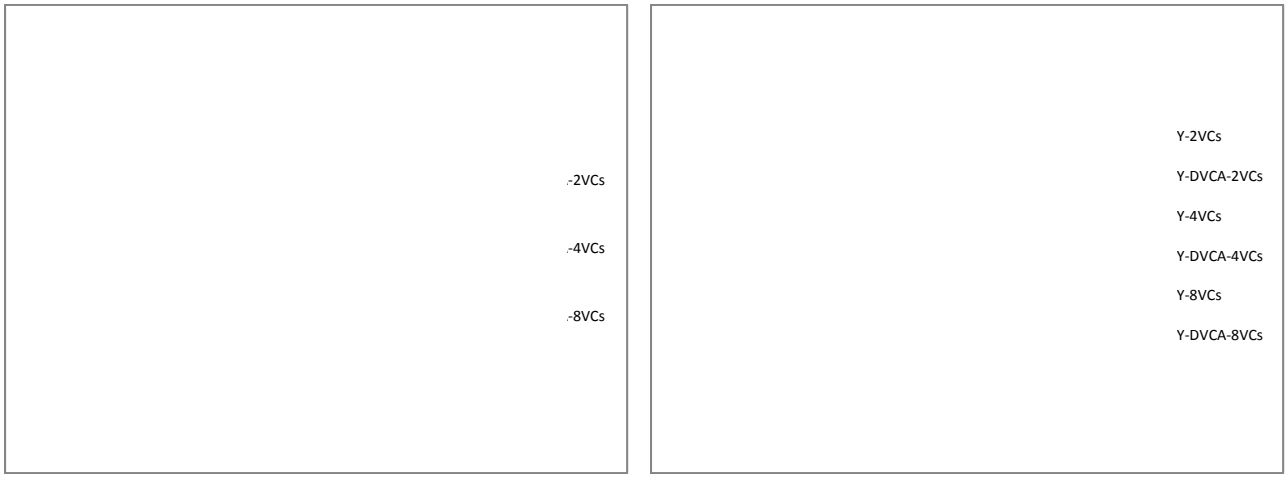


(c)

Fig. 6 Average latency for 6×6 2D mesh under (a) NED, (b) Transpose, (c) Hotspot, (d) Uniform traffic profile



(a)



(c)

Fig. 7 Average power consumption for 6×6 2D mesh under (a) NED, (b) Transpose, (c) Hotspot, (d) Uniform traffic profile

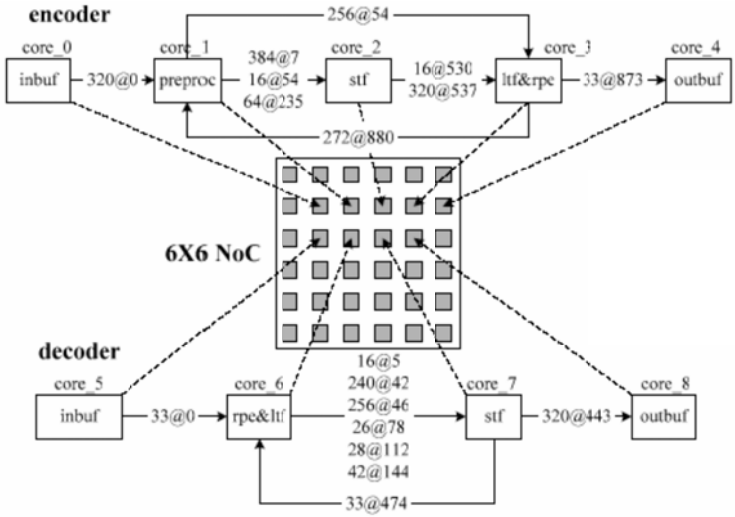
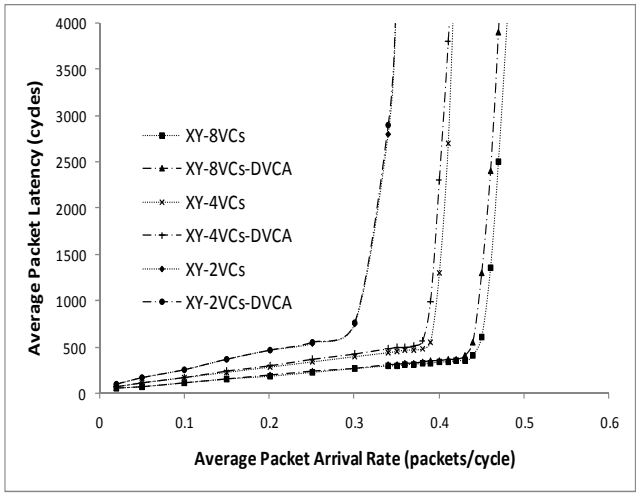
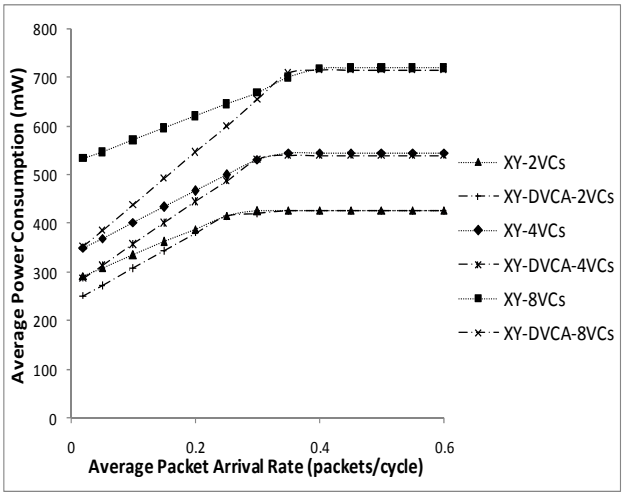


Fig. 8 Partition, communication trace and core mapping of a GSM voice CODEC [25]

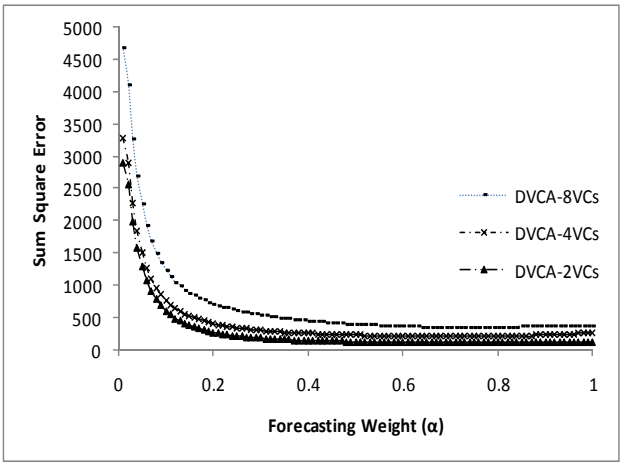


(a)



(b)

Fig. 9 (a) Average latency for 6×6 mesh size under CODEC traffic, (b) Average power consumption for 6×6 mesh size under CODEC traffic.



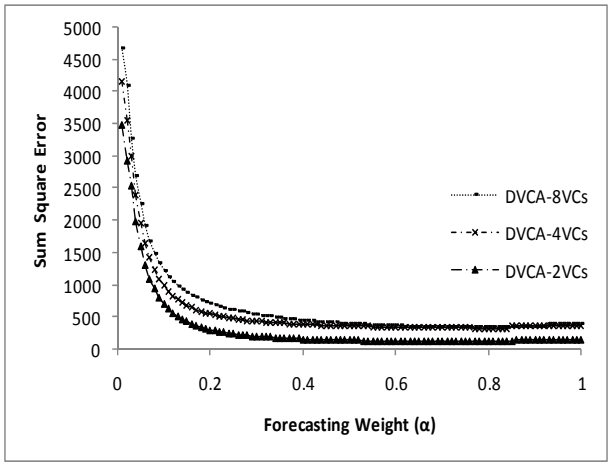


Fig. 10 Forecasting sum square error under (a) uniform traffic, (b) GSM voice codec as a function of α.

Table I. An example of calculating the total VCU.

cycle	*VC1*	*VC2*	*VC3*	*VC4*	*L1*	*L2*	*L3*	*L4*
1	1	2	-	-	1	1	0	0
2	1	3	4	-	1	1	1	0
3	1	3	5	-	1	1	1	0
4	6	3	5	7	1	1	1	1
5	-	3	5	7	0	1	1	1
	2	2	2	1	**4**	**5**	**4**	**2**

Table II. Area overhead of the DVCA unit

Component	**Area (μm^2)**	**Overhead (%)**
DVCA UNIT FOR 2 VCS	9296.81	2.78
DVCA UNIT FOR 4 VCS	11918.68	1.99
DVCA UNIT FOR 8 VCS	19863.11	1.09

Table III. Power overhead of the DVCA unit

Component	**Power (mW)**	**Overhead (%)**
DVCA UNIT FOR 2 VCS	4.86	1.7
DVCA UNIT FOR 4 VCS	5.27	1.23
DVCA UNIT FOR 8 VCS	8.55	0.89

BIOGRAPHIES

Amir-Mohammad Rahmani received B.S. degree from Mashhad Branch, Azad University, Iran, in 2006, and M.S. degree from the University of Tehran, Tehran, Iran, in 2009, both in Computer Engineering. He is currently pursuing his research in Low-Power High-Performance Nanosystems Laboratory, University of Tehran, Iran. His research interests include low power design, Network-on-chips, CAD, and test of digital circuits.

Kwanho Kim received his M.S. and Ph.D. degrees in Computer Engineering …

Massoud Pedram received a B.S. degree in Electrical Engineering from the California Institute of Technology in 1986 and M.S. and Ph.D. degrees in Electrical Engineering and Computer Sciences from the University of California, Berkeley in 1989 and 1991, respectively. …