A New Modeling Method for Vector Processor Pipeline Using Queueing Network

Tie Qiu[*] , Lei Wang[†] ,
He Guo, Xiaoyan Liu
School of Software
Dalian University of Technology
Dalian, China, 116620

Lin Feng[‡]
School of Innovation Experiment
Dalian University of Technology
Dalian, China, 116024

Lei Shu[§]
Nishio Lab
Osaka University 1-1
Yamadaoka
Suita Osaka, Japan 565-0871

ABSTRACT

Embedded vector processor is a kind of high-performance parallel processor. Pipeline design is a key technology in embedded vector microprocessors. This paper proposes a new modeling method for vector processor pipeline using open queueing network by instruction set feature of vector processor. According to instruction set distribution of vector processor in the practical projects and flowing in the pipeline modeling, the model of pipeline queueing network is analyzed. Total delay and mean delay are computed in every path. A better solution of pipeline is put forward as a result of delay data. Serving time of server nodes is averaged by partitioning for pipeline modeling and adding processing nodes in executing model. In conclusion, the delay data before and after improvement pipeline scheme are analyzed: the delay distributing of improvement scheme is almost equality and choke points with long delay and unequal are avoided.

Keywords

Vector Processor, Queueing network, Pipeline Modeling, Delay

1. INTRODUCTION

Embedded vector processor is a Single Instruction Multiple Data (SIMD) processor [1], which is a kind of high-performance parallel processor [2]. Repeated computing of the large amounts of data is solved by vector processor, which is used in embedded graphics image processing [3], data compression and data encryption in the communication field, and high-performance computing [4], etc. Application of pipeline technology [5] is an effective way, which is used to improve processing capacity of vector processor in the data operations. Processor pipeline is divided into multiple modules according to instruction execution time and process, so that multiple instructions operate in a parallel way within the different modules in the pipeline. The utilization of internal components in the CPU is increased, therefore processing speed is improved. At present, most of the pipeline technologies are proposed based on scalar processor architecture. These technologies are not fully suitable for pipeline of vector processor, resulting complicated design of pipeline architecture and difficult scheduling control, thus its efficiency is not high in vector processor pipeline. Therefore, research for modeling and optimization methods based on pipeline of vector processor have theoretical significance and practical value.

Queueing network is an effective system-level modeling method, which is widely used in modeling and performance analysis of computer systems and communication systems [6], [7]. In the queueing network models of pipeline technology, the task processing speed of queuing network nodes corresponds to the delay of pipe-segment, and the queueing network topology corresponds to link relationship in the pipe-segment in pipeline [8]. In recent years, the queueing network delay has been explored. Bolot and others in the reference [9] gave a method of network path delay, in a relatively short time period detecting the shortest delay. In the literature [10], Gurewitz, Cidon, and others suggested an improvemented objective function based on the least-squares difference, and the estimation method of deterministic time-delay is discussed. In the reference [11], Papagiannaki, Moon, Fraleigh *et al.* measured the shortest delay of queuing system based on maximum entropy methods. Literature [12] employed queueing network in order to model opportunistic multi-hop packet forwarding along the street with respect to the specifications of MAC and routing schemes, and evaluated the average delay and the maximum stable throughput. The successful use of queuing network solves the extension of all fields, but in the field of embedded microprocessor pipeline modeling and delay analysis is rarely mentioned. This paper presents a method of designing and modeling for pipeline based on vector processor. Through the establishment of an open queueing network model, the

[*] Corresponding author: qiutie@dlut.edu.cn

[†] Corresponding author: lei.wang@dlut.edu.cn

[‡] Corresponding author: fenglin@dlut.edu.cn

[§] Corresponding author: lei.shu@ieee.org

CHINACOM 2010, August 25-27, Beijing, China

DOI 10.4108/chinacom.2010.125

.

end-to-end delay of queueing network paths and the whole system delay are modeled and analyzed. The best solution of instruction pipeline for vector processor was obtained.

The remainder of this paper is organized as follows. Section 2 describes the architecture of vector processor pipeline and its function modules. Section 3 details related work and background knowledge, which include analyzing open queueing network, modeling of the vector processor pipeline using open queueing network, path delay calculation and average delay calculation for entire queueing network. Subsection 4.1 analyzes the original model of pipeline queueing network. Subsection 4.2 presents the modeling improved pipeline queueing network and re-calculates the path delays. Subsection 4.3 analyzes the average network delays with M paths and the comparison of maximum average node delay of before and after improvement. Finally, we conclude this paper in Section 5.

2. THE ARCHITECTURE OF VECTOR PROCESSOR PIPELINE AND QUESTIONS

We design a vector processor, which is 8 x 32bits parallel processor, using 32-bit instruction word length of the RISC instruction system [13]. Data bus is 32 bits, vector length can be configured from 1 to 8. The overall architecture is shown in Fig. 1.

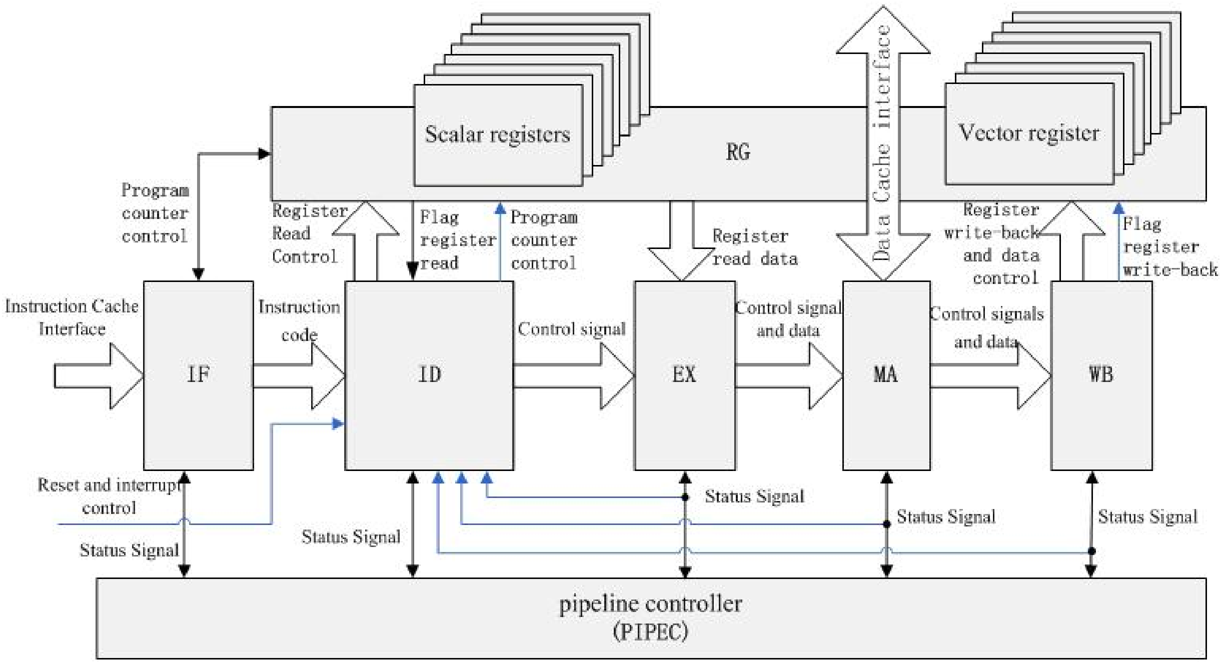

Figure 1: The pipeline architecture of vector processor

The vector processor pipeline has seven components, including: 1) Registers Group (RG), 2) Pipeline Controller (PIPEC), 3) Fetching Module (IF), 4) Decoding Module (ID), 5) Executing Module (EX), 6) Memory Access Module (MA), and 7) Write-back Module (WB). Where, the registers group includes scalar registers and vector registers. The instruction Cache is loaded by the instruction memory interface. The instruction operation code has been identified in the decoding stage and the data been read from registers. Arithmetic and logic operation functions and the address calculation function are realized in the executing stage of instruction pipeline. This module also includes the scalar data, vector data, and multiplication of data processing and computing, thus needs more time-consuming. Only the memory access instruction flows through the MA, where it can include byte, half word, word and multi-word memory access instructions. If instruction needs to be written back, the result is written back to the corresponding registers. Each instruction in accordance with the order of instruction address is read out in the whole pipeline operations, flows through the IF, ID, EX, MA, WB five stages in turn, under the control of the pipeline controller.

When the vector processor is running, the loaded instructions are processed by pipeline. The instruction's path through the pipeline is different for instruction flows of different types, and the time delay of processing instruction is different for different paths. One critical issue in designing effective pipelines is how to get the delay of shortest paths and nodes by repartitioning the pipeline modules according to different instruction flowing paths, which will greatly shorten the average delay. In the following work, the queuing network modeling method is used to calculate the path delay and improve vector processor pipeline.

In order to better describe the modeling and analysis methods, the following symbols are defined as Tab. 1.

3. RELATED WORK AND BACKGROUND KNOWLEDGE

3.1 Open Queueing network

Open queueing network of a typical queue is displayed as shown in Fig. 2. When a task reaches node i, there are three conditions in an open queueing network of a typical queue: (i)Independent external Poisson arrival λ_e. (ii)From queue k with probability p_{ki} to reach node i. (iii)With probability p_{ii} in the network loop. When a task leave from node i, there are two conditions: (i)To reach node j with probability p_{ij}.

Table 1: The symbol definition

Symbol	Description
i, j, k, d	Node No. in the queueing network.
m	Queue length of node.
n	Path No. in the queueing network.
N	Total number of node in the paths.
M	Total number of path in the queueing network.
λ_i	Average task arrival rate of node i.
μ_i	Service rate of node i.
ρ_i	Utilization of Node i.
λ_e	Independent external Poisson arrival.
p_{ie}	Probability of leaving the node i.
p_{ij}	Probability of from node i to node j.
γ	Pure arrival rate, which enter the queueing network.
K	Node state.
$E(k_n)$	Average tasks number in the path n.
$E(k)$	Average tasks number in the queueing network.
$E(T)$	End to end delay with N-level nodes .
$E_M(T)$	Average network delay with M paths.

(ii)To leave the node i with probabilit p_{ie}.

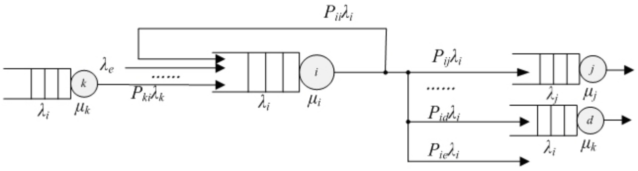

Figure 2: The queue modeling of queueing network

For the open queueing network, Theorem1 has been established.

Theorem 1. For any open queueing network, all possible speed of leaving the node i equal the arrival speed in the state.

$$\left[\lambda_i + \sum_{i=1}^{m} \mu_i\right] p(k) = \sum_{i=1}^{m} \lambda_e p(k - I_i) + \sum_{i=1}^{m} p_{id}\lambda_i p(k + I_i) + \\ + \sum_{i=1}^{m}\sum_{j=1}^{m} p_{ki}\lambda_k p(k + I_j - I_i) \qquad (1)$$

Where I_i and I_j is a unit vector, that show per unit change of a node starting from a state . More details on Theorem 1 can be found in [14].

By Theorem 1, we can give corollary 1:

Corollary 1. For open queueing network of pipeline, number of all possible instruction of leaving the node i equal the arrival number in the state.

$$\lambda_e + \sum_{i=1}^{m-1} \lambda_k p_{ki} = \sum_{f=j}^{d} \lambda_i p_{if} + p_{ie}\lambda_i \qquad (2)$$

Proof: Using reductio ad absurdum, suppose the number of instructions entering the node i is not equal to the number of instructions leaving the node i, then according to Theorem 1 of the flow balance equation, there must be instruction with probability $p_{ii} \neq 0$ in the self-loop. This is in contradiction with that each node service is the order of one-way in a microprocessor instruction pipeline. Therefore, the Corollary1 is established. (End proof).

3.2 Node delay calculation

Each task delay is formed by queuing delay (wait delay) and transmission delay in the queueing network of pipeline. Where, transmission delay has linked with groping length and capacity of transmission path.

Utilization ρ_i of node i by the formula (3) is given.

$$\rho_i = \frac{\lambda_i}{\mu_i} \qquad (3)$$

End to end delays caused by M/M/1 queueing models of the N-level nodes, by the formula (4) is given.

$$E(T) = \sum_{i=1}^{N} \frac{1}{\mu_i - \lambda_i} = \sum_{i=1}^{N} \frac{1/\mu_i}{1 - \rho_i} \qquad (4)$$

3.3 Average delay calculation for entire queueing network

In the pipeline queueing networks, each node is considered as a M/M/1 queue, and each path is considered as a queueing model with N-level nodes, service rate μ_n, the average task arrival rate λ_n and pure arrival rate γ of entering the queueing network. Then the average task number $E(k_n)$ in the path n equals sum of servicing and queuing tasks, $E(k_n)$ can be the formula (5) obtained.

$$E(k_n) = \frac{\lambda_n}{\mu_n - \lambda_n} \qquad (5)$$

The average number of tasks $E(k)$in queueing network can be the formula (6) obtained.

$$E(k) = \sum_{n=1}^{M} E(k_n) \qquad (6)$$

The average number of tasks $E(k)$ and the average delay $E_M(T)$ of entire queueing network accord with the following relationship (7) by Little's formula [15].

$$\gamma E_M(T) = E(k) \qquad (7)$$

The average network delay with M paths is obtained by the equation (5), (6), (7), that shown in formula (8).

$$E_M(T) = \frac{1}{\gamma}\sum_{n=1}^{M} \lambda_n T_n = \frac{1}{\gamma}\sum_{n=1}^{M} \frac{\lambda_n}{\mu_n - \lambda_n} \qquad (8)$$

4. PERFORMANCE EVALUATION USING QUEUEING NETWORK MODELING

4.1 Modeling for vector processor pipeline

According to Reference [16] used in the method description of queuing system, we will build the pipeline of vector processor as an open queueing network. Various types of instruction arrival rate approximates Poisson distribution, and interval of service time approximates exponentially distributing. The pipeline queueing model is formed by a queueing network of M/M/1 queues. The queueing network of five level pipeline is shown in Fig. 3. The nodes 1 to

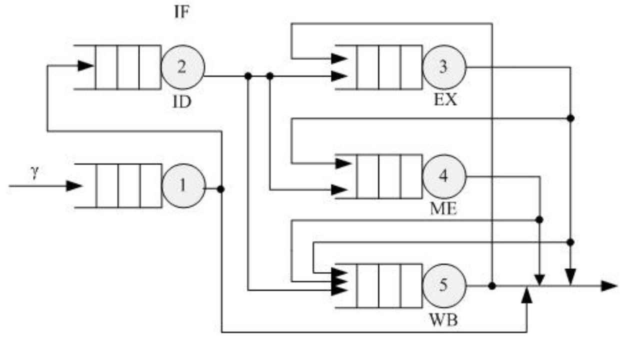

Figure 3: The Queueing network of five level pipeline

5 represent the fetching module, decoding module, executing module, memory access module and register write-back module in the queueing network of pipeline.

Embedded vector processor instructions can be classed into branch instructions, scalar and vector data processing instructions, testing instructions, load/store instructions for single memories, load/store instructions for multiple memories and scalar multiplication instruction. Instructions of different function types are served differently in the pipeline. For example, data processing instructions need write-back results, and test instructions do not need to write back the results. In this paper, to achieve the highest performance and efficiency for the pipeline components of embedded vector processor, different function instructions are classified and statistics by practical project tasks that vector processors can process, thus the transfer probability of pipeline queueing networks is obtained. The instruction flows of the five level pipeline queueing network are shown in the transfer matrix (9) .

$$P = \begin{bmatrix} 0.0 & 0.99 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.81 & 0.05 & 0.14 \\ 0.0 & 0.0 & 0.0 & 0.11 & 0.80 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.61 \\ 0.0 & 0.0 & 0.07 & 0.0 & 0.0 \end{bmatrix} \qquad (9)$$

Vector processor has been simulated using FPGA programming techniques. Processing cycle of node service rate μ_1, μ_2, μ_4, μ_5 is $10ns$, and processing cycle of node service rate μ_3 is $18ns$. Tab. 2 shows the calculation results of path delay. According to the calculation results, node delay distribution is uneven. For example, E(T15), when path delay through the nodes (1, 2, 3 , 5) is $88.129ns$, when path delay through the nodes (1, 2, 4, 5) is $41.340ns$. This is because the bottlenecks segment of pipeline results in that overall system efficiency is not high.

4.2 A case study

From the node service rate queueing network of five level pipeline, The processing time of executing module is larger than the processing time of the other four modules. As can be seen from Tab. 2, the path containing node 3 has a larger delay. Thus the problem about leading to the uneven distribution of the node delay and becoming a bottleneck in the pipeline segment need to be improved.

In the pipeline architecture of vector processors, the amount of tasks of the executing module can be divided into scalar data processing, multiplication and vector operations. If there are multiplication and vector operations, then increase service time delay in the executing module (EX). As for service time of any task, three operations can be completed within 10ns. Therefore, node 3 (EX) is divided into scalar computing module (EXI), vector computing module (EXV) and multiplication module (MUL). The service rate of each node is approximately the same in the queueing network, the improved pipeline queueing network model is shown in Fig. 4.

Table 2: The path delay of five level pipeline

delay	Paths	Paths Dealy (ns)	Average Node Delay (ns)
E(T13)	Node1,2,3	74.058	24.686
E(T14)	Node 1,2,4	30.433	10.144
E(T14)	Node 1,2,3,4	84.974	21.244
E(T15)	Node 1,2,3,5	88.129	22.032
E(T15)	Node 1,2,4,5	41.340	10.335
E(T15)	Node 1,2,3,4,5	95.880	19.176
E(T15)	Node 1,2,5	30.490	10.163
E(T15)	Node 1,2,3,4,5,3,4,5	117.700	14.713

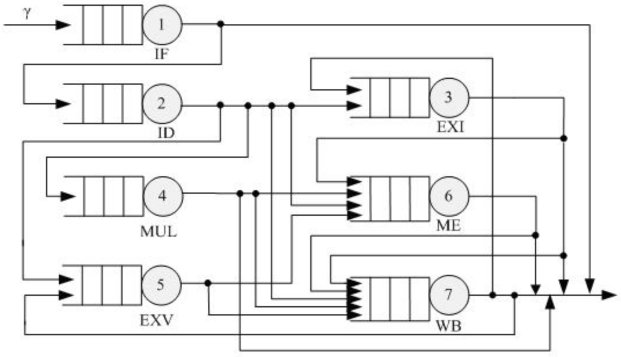

Figure 4: The improved pipeline queueing network

According to the practical engineering application of vector processor, implementation code are classified. Statistic results show that the probability of using the multiplication instruction is about 0.19, and the probability of using vector instruction about 0.22. The instruction flows of the pipeline queueing network were shown in the transfer matrix (10) after improvement.

$$P = \begin{bmatrix} 0.0 & 0.99 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.34 & 0.19 & 0.22 & 0.12 & 0.13 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.23 & 0.65 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.58 & 0.31 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.48 & 0.52 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.54 \\ 0.0 & 0.0 & 0.07 & 0.0 & 0.04 & 0.0 & 0.0 \end{bmatrix} \qquad (10)$$

Tab. 3 shows the path delay calculation results, which is obtained by instruction stream of the different running paths after improved queueing network. The average node delays tend to balance in all the paths from calculation results in Tab. 3.

4.3 Contrast of system average delay calculation

The comparison is obtained by maximum average node delay of before and after improved pipeline queueing network as Fig. 5. Where, $N = 3, 4, 5, 8$ means 3, 4, 5, 8 nodes through path, respectively. The average node delays in all paths tend to balance after improvement.

The pipeline average delay of queueing network with 8 paths before improvement (as shown in Fig. 3) is calculated by formula (9) and the result is $E_8(T) = 66.666ns$. The pipeline average delay of queueing network with 13 paths after improvement (as shown in Fig. 4) is $E_13(T) = 29.780ns$. Improved pipeline average delay of queueing network delay for vector processor is reduced to 44.67% with the result before the improvement.

Table 3: The path delay calculation results after improvement

delay	Paths	Paths Dealy (ns)	Average Node Delay (ns)
E(T13)	Node 1,2,3	31.826	10.609
E(T14)	Node 1,2,4	30.840	10.280
E(T16)	Node 1,2,3,6	42.299	10.425
E(T16)	Node 1,2,4,6	41.417	10.354
E(T16)	Node 1,2,5,6	41.663	10.416
E(T17)	Node 1,2,3,7	43.291	10.823
E(T17)	Node 1,2,4,7	41.140	10.285
E(T17)	Node 1,2,5,7	41.720	10.430
E(T17)	Node 1,2,3,6,7	53.642	10.728
E(T17)	Node 1,2,4,6,7	52.760	10.552
E(T17)	Node 1,2,5,6,7	53.006	10.601
E(T17)	Node 1,2,3,6,7,3,6,7	85.750	10.719
E(T17)	Node 1,2,5,6,7,5,6,7	85.152	10.644

5. CONCLUSION

This paper studies the efficiency factor of vector processor pipeline and improved methods. Queueing network model for vector processor pipeline is established, and performance parameters of delay are analyzed. Finally, the method for eliminating bottleneck of pipeline delay is proposed. System delay problem of uneven distribution is solved by the method, which execute module (EX) is divided into scalar computing module (EXI), vector computing module (EXV) and multiplication module (MUL). The path delay of vector processor pipeline and the average delay of entire queueing network have been reduced after improved queueing network model. This study is provided with some guidance for the future design and modeling pipeline architecture of vector processor.

The queueing network model of single-server system is discussed in this paper. Multi-server situation (i.e., multi-core processors and on-chip network), and performance optimization for internal module of pipeline, as well as processing blocking between the modules, will be follow up research.

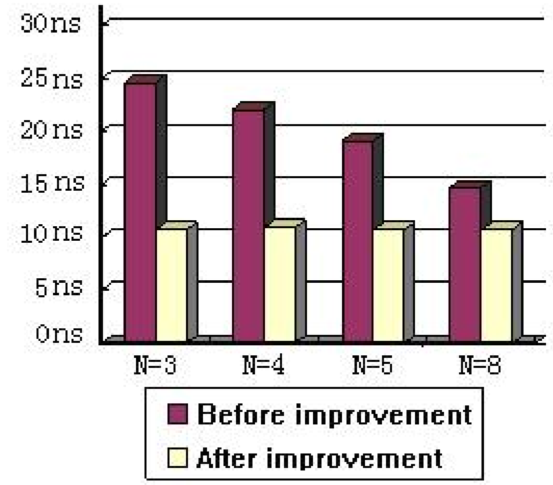

Figure 5: The comparison of maximum average node delay of before and after improvement

Acknowledgment

(1) Lei Shu's research in this paper was supported by Grant-in-Aid for Scientific Research (S)(no. 21220002) of the Ministry of Education, Culture, Sports, Science and Technology, Japan.

(2) Lin Feng's research in this paper was supported by the National Science Foundation of China(no. 60773213)

6. REFERENCES

[1] Jakub K, Wesley A, Jack D, "Optimizing matrix multiplication for a short-vector SIMD architecture-CELL processor," *Parallel Computing*, vol. 35, no. 3, pp. 138–150, 2009.

[2] Robelly J P, Cichon G, Ahlendorf H, Fettweis G, "A HW/SW design methodology for embedded SIMD vector signal processors," *International Journal of Embedded Systems (IJES)*, vol. 3, no. 3, pp. 160–169, 2008.

[3] Jae-Sung Yoon, Donghyun Kim, Chang-Hyo Yu, Lee-Sup Kim, "A 3D graphics processor with fast 4D vector inner product units and power aware texture cache," *Custom Integrated Circuits Conference(CICC)*, pp. 539–542, Sep. 2008.

[4] Vijaya Y, Raghavan S, Michael S, "Techniques to improve motion compensation performance of H264 video decoder using a vector processor," *International Symposium on Communications and Information Technologies Proceedings(ISCIT)*, pp. 1082–1087, Oct. 2007.

[5] Michael F, "Evaluating dataflow and pipelined vector processing architectures for FPGA co-processors," *Proceedings of the 9th EUROMICRO Conference on Digital System Design: Architectures, Methods and Tools*, pp. 127–130, Aug. 2006.

[6] Vidhyacharan B, "State diagrams and steady-state balance equations for open queuing network models," *Computers and Electrical Engineering*, vol. 31, pp. 460–467. 2005.

[7] John N, Daigle, *Queueing theory with applications to*

packet telecommunication, Boston, MA: Springer Science and Business Media, Inc., 2005.

[8] Lei Wang, Zhi-ying Wang, Kui Dai, “An approximate method by queueing network modeling for performance evaluation of asynchronous pipeline rings,” *2006 IEEE Internal Conference on Computer and Information Technology,Seoul,Korea*, pp. 244–249, Sep. 2006.

[9] Bolot J C, “Characterizing end-to-end packet delay and loss in the internet,” *Proceedings of the ACM SIGCOMM'93, New York: ACM Press*, pp. 289–298, 1993.

[10] Gurewitz O, Cidon I, Sidi M, “One-way delay estimation using network-wide measurements,” *IEEE Transactions on Information Theory*, vol. 52 no. 6, pp. 2710–2724, 2006.

[11] Papagiannaki K, Moon S, Fraleigh C, et al, “Analysis of measured single-hop delay from an operational backbone network. Proceedings of IEEE INFOCOM'02,” *New York : IEEE*, pp. 535-544, 2002.

[12] Adel J, Farid A, “Analytical evaluation of average delay and maximum stable throughput along a typical two-way street for vehicular ad hoc networks in sparse situations,” *Computer Communications*, vol. 32, pp. 1768–1780, 2009.

[13] Tomas B C, Rene C, Claudia F U, “On the design and implementation of a RISC processor extension for the KASUMI encryption algorithm,” *Computers and Electrical Engineering*, vol. 34, no. 6, pp. 531–546, 2008.

[14] Sheng Youzhao, *Queuing Theory and Its Application in Modern Communication*, Bei Jing: Posts and Telecom Press, pp. 2007.

[15] Buetler, Frederick J, “Mean sojourn times in markov queueing networks – Little's formula revisited,” *IEEE Transactions on Information Theory*, vol. IT–30, no. 2, pp. 233–241, 1983.

[16] Tahilramani H, Manjunath D, Bose S K, “Approximate analysis of open network of GE/GE/m/N queues with transfer blocking,” *Proc. of the 7th Seventh International Symposium on Modeling Analysis and Simulation of Computer and Telecommunication Systems*, University of Maryland, pp. 164–172, Oct. 1999.