Required Delay of Packet Transfer Model For Embedded Interconnection Network

Mohd. Kalamuddin Ahmad
Research Scholar
Mewar University
Rajasthan

Prof (Dr.) Mohd. Husain
Director (AIET)
Lucknow

***Abstract*— We proposed a model based on M/M/1 queuing theory for delay analysis in wormhole routing network on an embedded architecture of torus network with hypercube (K-ary n cube).The product is generated from embedded interconnection network to good interconnection network can be designed for parallel computing systems .The proposed model follow as input an application graph, topology, and packet latency. This model can estimate the flow of average latency. It is, show latency and delay of packet, for higher dimensions interconnected networks.**

***Keywords- Torus network, Embedded network, Hypercube network,* Network *parameters,* Queueing theory.**

I. INTRODUCTION

Delay of packet transfer is recognized as one of the most critical design characteristics for torus embedded hypercube interconnection network architectures [12]. In this work, we propose a model which predicts the latency of flows in a network- in torus embedded hypercube based system. Models are frequently employed by system designers for early architecture and design decisions. To this end, application and architecture models are first developed separately.

An analytical model which can estimate the desired performance metrics in a fraction of time. Analytical models can be used to prune the large design space in a very little time compared to simulate. Thus, it is justified to derive accuracy analytical models for performance predicts of torus embedded hypercube to eliminate the necessity for time consumption. The information provided during the performance analysis step can be used in any optimizing loop for torus embedded system such as topology selection, application mapping, and buffer allocation. Although the use of high-level models conceals a lot of complex technological aspects, it facilitates fast explore of the torus embedded design space. Accurate simulations can be setup at later steps of design process when the design space is reduced to a some practical choices.

In this research a performance queuing model, is proposed and evaluated for torus embedded. The performance queuing model, which is based on a M/M/1 queuing model, has been developed for deterministic routing and wormhole routing. The proposed model is topology-independent and assist any kind of spatial and temporal traffic patterns. The estimated performance metrics such as average latency can be conveniently used for optimizing purposes to find appropriate design parameters. This gives us confidence that we can utilize the model in the early

design phase of high performance torus embedded hypercube interconnection networks.

The interconnection network is an important component in a parallel computer. A better interconnection is expected to have number of links, topological network cost and more reliable [10]. The interconnection network must be able to scale up with a few building blocks and with minimum redesign. The hypercube is a network with high connectivity and simple routing but the node degree grows logarithmically with number of vertices making it typical task to built scalable architecture Torus is a network with constant node degree and is highly scalable architecture but has greater network diameter [6]. The advantages of hypercube and torus network can be superposed on to an embedded architecture [1], [10], [12]. called torus embedded hypercube scalable interconnection network.

Computing delay time for all nodes and channels, the average packet latency between any two nodes in the network, latency can be calculated. The average packet latency is the weighted mean of these latencies; start-uptime is also called latency.

II. PROPERTIES OF ARCHITECTURE

Suppose $l \times m$ be the size of several concurrent torus networks with l no. of rows and m no. of columns and N being the no. of nodes connected in the hypercube, the torus embedded hypercube network can be designed with the size of (l, m, N). Nodes with identical positions in the torus networks will be a group of N no. of nodes connected in the hypercube configuration and can be addressed with three parameters such as row no. i, column no. j of torus and address of node k in hypercube where the addressed node is residing. Hence, a (l, m, N)–torus embedded hypercube network size will have $l \times m \times N$ no. of nodes and a node with address as (i, j, k) where $0 \le i < l$, $0 \le j < m$ and $0 \le k < N$. The data routing functions of torus embedded hypercube as :

$T_1 (i, j, k) = (i, (j+1) \ mod \ m, k)$ (1)

$T_2 (i, j, k) = (i, (m+j-1) \ mod \ m, k)$ (2)

$T_3 (i, j, k) = ((i+1) \ mod \ l, j, k)$ (3)

$T_4 (i, j, k) = ((l+i-1) \ mod \ l, j, k)$ (4)

$T_C (k_{n-1} .....k_{d+1} \ k_d \ k_{d-1} .....k_0)$

$= (k_{n-1} .....k_{d+1} \ \overline{k}_d \ k_d .....k_0)$ (5)

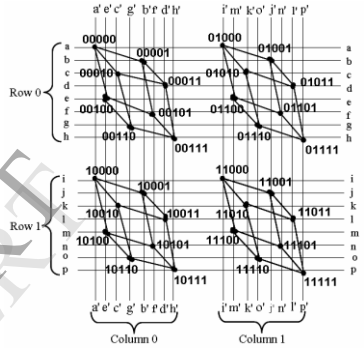


Fig:1. A (2, 2, 8)-torus embedded hypercube Network

Two nodes (i_1, j_1, k_1) and (i_2, j_2, k_2) are said to be connected if following connection rules are satisfied:

Rule 1: A *hypercube link* in the network will exist if

(i) $i_1 = i_2$ and

(ii) $j_1 = j_2$ and

(iii) k_1 and k_2 differ by one bit position in their binary address.

Rule 1 generates $l \times m$ hypercube with dimension N and these hyper cubes are separated from each other until the *rule* 2 is applied.

Rule 2: A *mesh link* will exist if

(i) $k_1 = k_2$ and

(ii) i and j differ by one in one component while the other component is identical.

Rule 2 generates *N* meshes with dimension *l*×*m* and these meshes are isolated from each other if *Rule* 1 is neglected.

The end to end connections of row and column of each torus are not shown in Fig1 for simplicity. A wraparound connection is done along every row or column if they have same label as a completion of (2, 2, 8)-torus embedded hypercube network. The proposed network is a highly scalable network. Scalability is achieved in either ways. Firstly, the dimension of the hypercube can be increased by keeping the size of concurrent torus same but increasing the number of concurrent torus accordingly. Secondly, dimension of torus is expanded by keeping the size of the hypercube constant. Scaling up the system using latter method in which expanding the size of torus without affecting the node degree of existing nodes is preferred than the case in former method of hypercube expansion [4], [5].

The total number of links, topological network cost, the scalability and the reliability are the parameters considered in evaluating the performance of this network. The result obtained shows that the torus embedded hypercube favors the scalability of interconnection network.

III. FOUNDATION

Queuing theory is an appropriate and useful modeling tool for system analysis and performance evaluation in computer and telecommunications network [14]. Since our proposed model has been constructed on the M/M/1 priority queue [3], [22], in this section we give a quick review on the M/M/1 queue and priority queue concepts.

A. M/M/1 Queue

The M/M/1 queueing model has a single service facility with one server, unlimited waiting room and the first-come first-served queue discipline. The service times are independent and identically distributed with a general distribution, the inter_arrival times of customers are also independent and identically distributed with a general distribution, and the inter_arrival times are independent of the service times. It is assumed that the general inter_arrival time and service time distributions are each partially specified by their first two moments. We should remind here that the nth moment of a random variable X is defined as the average of $X^n(\overline{X^n} = \Sigma_{i=1}^{k}(X_i)^n/k)$. All descriptions of this model thus depend only on the basic parameter four-tuple $(\bar{a}, a^2, \bar{s}, s^2)$, where $\bar{a}$ and $\overline{a^2}$ are the first and second moments of the customers' inter_arrival time, and similarly, $\bar{s}$ and $\overline{s^2}$ are the first and second moments of the service time. Also in this work we consider the arrival rate and service rate as $\lambda = 1/\bar{a}$ and $\mu = 1/\bar{s}$, respectively. The mean delay time of a M/M/1 queuing model system can be approximate by Allen – Cunneen formula [3]

$$\bar{W}_{M/M/1} \approx \frac{\rho\,(C_A^2 + C_S^2)}{2\mu(1-\rho)} \qquad (1)$$

Where ρ is the utilization factor of the server and equal to λ/μ, C_A and C_S are the coefficient of variation of the inter_arrival time and service time respectively [3].We remind that the relationship between Coefficient variation of random variable X and its moments is represented by

$$C_X^2 = \overline{x^2}/\bar{x}^2 - 1.$$

B. Priority Queue

We consider a system with one server in which the customers have preferential treatment based on priorities associated with them. We assume that the priority of a customer is an integer fixed at arrival time, and a customer with priority i (i=1,2,3……..,p) belongs to class i . We say one customer has higher priority than another if it belongs to a priority class with lower index.

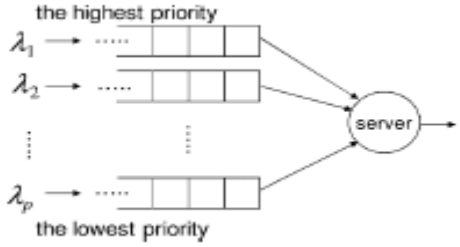


Fig.2 Priority queueing system.

In other words, the lower the index, the highest priority. The priority queuing system to be studied is depicted in Fig. 2, where the different queue levels correspond to the different priority classes. For the service discipline, we assume that whenever a customer is completed, the server is next assigned to that customer at the head of the highest priority nonempty queue. Once a customer begins on the server, it is allowed to run to completion; i.e., the service discipline is non preemptive. Independent and identically distributed arrivals and service times are assumed for the ith class with the arrival and service rate denoted by and μ_i, respectively. The mean delay time of random arrivals to the ith queue $\bar{W}_i$ can be written as [22]:

$$\bar{W}_i = \frac{\bar{R}}{\left(1 - \Sigma_{k=1}^{i-1}\rho_k\right)\left(1 - \Sigma_{k=1}^{i}\rho_k\right)} \qquad (2)$$

Where $\bar{R}$ is the residual service time seen by an incoming customer. In a M/M/1 queuing system, is approximated by [3]

$$\bar{R} \approx \Sigma_{k=1}^{p} \frac{\rho_k}{2\mu_k}\left(C_{A_k}^2 + C_{S_k}^2\right) \qquad (3)$$

Where μ_k and ρ_k are average service rate and utilization factor of class k , respectively. Also C_{Ak} , and C_{Sk} are CV of inter_arrival time and service time of class k, respectively.

In all the analysis we have reviewed so far, the queue size of each class was infinite. However, in the case of wormhole switching this is not a true assumption, because in wormhole switching each buffer can hold only finite number of flits.

IV. PERFORMANCE ANALYSIS

The following assumptions are made when developing the proposed performance model.

• The Performance queuing model works for deterministic routing algorithms which may be minimal or non-minimal.
• The switching method is wormhole and messages are broken into packets (flits).
• There is one finite FIFO queue per channel and channels are allocated per packet. It means that the channel is released when the entire flits has transferred via channel.
• Flits are consumed immediately by the destination node. In order to characterize network performance, architecture and application models are essential.

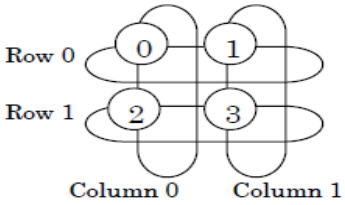


Fig :3(a)

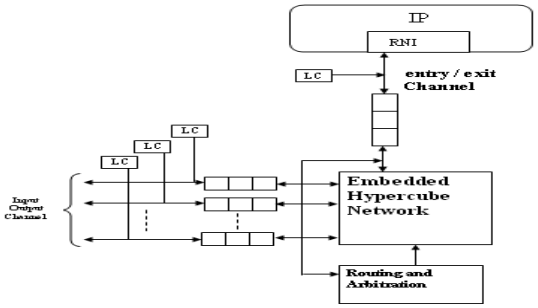


Fig :3(b)

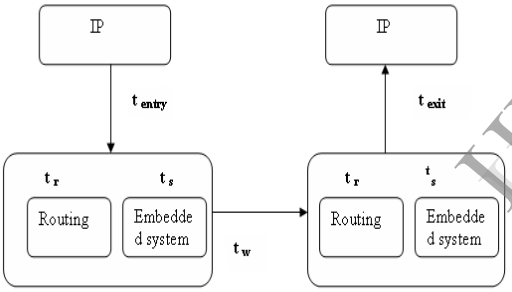


Fig :3(c)

A. Model of Architecture

As shown in fig. 3(a) graph can represent the topology of torus and fig1. Torus embedded hypercube architecture. Vertices and edges of the graph show nodes and channels of the Hypercube, respectively. The structure of a single node is depicted in Fig. 3(b). Every node contains an intellectual property core and a router with input channels and output channels. Each intellectual property core performs its own computational, storage or I/O processing functionality, and is equipped with a resource-network-interface. The resource-network-interface translates data between intellectual property cores and routers by packing/unpacking data packets and also manages the packet entry process. Packets are entered into the network on the entry channel (input port 1) and leave the network from the exit channel (output port 1). Generally, each channel connects output port of node to input port j of node N to input port i of node M. Therefore, we denote this channel OC^{N}_{j} (jth output channel of router N) or IC^{M}_{i} (ith input channel of router M). We consider the general reference architecture for routers in [7] and it comprises the following major components.

• ***Buffer.*** This is a finite FIFO buffer for storing flits in transit. In the model shown in Fig. 3(b), a buffer is associated with each input physical channel and each output physical channel. In alternative designs, buffers may be associated only with inputs (input buffering) or outputs (output buffering).

•***Link controller*** **(LC).** The flow of flits through the physical channel between adjacent routers is implemented by the link controller. The link controllers on either side of a channel coordinate to transfer flits.

•**Embedded hypercube.** This component is responsible for connecting router input channels to router output channels.

•***Routing and arbitration unit.*** This component implements the routing algorithms, selects the output channel for an incoming packet, and accordingly sets the crossbar switch. Routing is only performed with the head flit of a packet.

If two or more packets simultaneously request the same output channel, the

arbiter must provide for arbitration among them.

In this work, we suppose that input channels have a descending order of priority in a clockwise direction for each output channel. The incoming packets from entry channel have the highest priority in every priority set. Usually, a control mechanism impedes the network from being overloaded. Therefore, it is assured that the router is never overloaded, and incoming packets from lower priority channels do not face starvation. If the requested output channel is busy, the incoming head flits remains in the input buffer. It will be routed again after the channel is freed and if it successfully arbitrates for the channel. Similar to the network model in [11], we suppose that the routing decision delay for a packet, crossing time of a flit over the torus embedded hypercube, and transmitted time of a flit across a wire between two adjacent routers are t_r, t_s, and t_w respectively. Also the transfer times of a flit through the entry and exit channels are considered to be t_{ent} and t_{exit}, respectively. We can infer that the latency of a head flit of a one hop packet in the absence of contention includes entry channel delay (t_{ent}), first router delay ($t_r + t_s$), inter-node wire (t_w) delay, second router delay ($t_r + t_s$), exit channel delay (t_{exit}). So, we can write it as $t_{ent} + (t_r + t_s) + t_w + (t_r + t_s) + t_{exit}$

In this study, we consider the wormhole-routing under deterministic routing algorithm. Although adaptive routing algorithms avoid congested channels and result in more balanced load on the network, they may cause out-of-order packet delivery. The reorder buffers needed at the destination for ordering the packets impose large area and power on system [15]. Deterministic routers not only are more compact and faster than adaptive routers, but also assure in-order packet delivery.

B. Model of Application

The weight of the edge represents the communication rate between source and destination. We consider the communication graph , Although generation of data packets in hypercube nodes has dependence, especially in application-specific platforms, the studies in [10], [20] show that compared to real traffic traces in hypercube:

t_r : Time spend for packet routing decision (cycles)

t_s : Time spend for routing (cycles)

t_w : Time spend for transfer a flit between two adjacent routers (cycles)

m: Average size of packets(flits)

ρ : Standard deviation of packet size (flits)

L: Average packet latency in the network from source to destination

L_a: Average packet latency in the network (cycles)

IP^N: The IP core located at address N

R^N: The router located at address N

IC^N_i : The ith input channel in router R^N

OC^N_j : The jth input channel in router R^N

IB^N_i : Capacity of the buffer in IC^N_i (flits)

OB^N_j : Capacity of the buffer in OC^N_j (flits)

P: Probability of a packet is generated in IP source and is delivered to IP destination $\sum_S \sum_D$ **P=1**

λ^N : Average packet entry rate of IP^N (packet / cycles)

$\lambda^N_{i\to j}$: Average packet rate of IC^N_i to OC^N_j (packet / cycles)

λ^N_j : Average packet rate to OC^N_j (Packet / cycles)

$P^N_{i\to j}$: Probability of a packet entered From IC^N_i to exited from OC^N_j

μ^N_i : Average service rate of OC^N_j

$\bar{R}^N_j$: Residual service time seen by an incoming flow (cycles)

$C_B{}^N_j$: Coefficient of variation for service time OC^N_j

$C_A{}^N_{i\to j}$: Coefficient of variation for iter_arrval time of packet from IC^N_i to OC^N_j

$\rho^N_{i\to j}$: Fraction of time occupied by packet

$W^N_{i\to j}$: Average waiting time for a packet from IC^N_i to OC^N_j (cycles),

The incoming packets will be still accurate to model their traffic generation separately as independent bursts of packets with statistical characteristics.

We assume that the packet entry process to the router has a general distribution with mean value of packets/ cycle and coefficient of variation of C_A . Also, the probability of packet transfer from the source node S to the destination node D is P.

V. COMMUNICATION ANALYSIS

To have a better view of the proposed model, the main idea of the analysis approach is summarized here.

a) To calculate the average latency of flows, it is essential to estimate the packet waiting times for network channels.

b) Every channel is modeled as a M/M/1 priority queue and the delay time to access every channel is calculated based on the set of flits (packet) arrival rate and channel service time which are calculated in (c) and (d), respectively.

c) Communication volume among IP cores and routing algorithm, the packet arrival rate to each channel is determined.

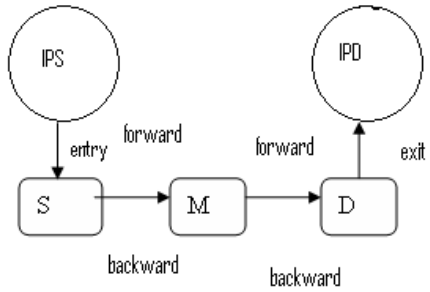


Fig. 4. Two-hop flow from source to destination.

d) The channel service time, which is part of the delay time, is calculated recursively for every communicate way starting from the destination node.

A. Model of Latency

The average packet latency is used as the performance metric. We assume that the packet latency spans the instant when the packet is created, to the time when the packet is Delivered to the destination node . We also assume that the packets are consumed immediately once they reach their destination nodes.

In Fig. 4, consider a flow which is generated in IP source, and reaches its destination IP destination after traversing R^S , R^M and R^D. The latency of this packet L consists of two sections: the latency of head flit L_h and the latency of body flits L_b . In another words

$$\mathbf{L = L_{h\ (source\ to\ destination)} + L_b} \qquad \mathbf{(4)}$$

$L_{h\ (source\ to\ destination)}$ is the time since the packet is created in IP source, until the head flit reaches the IP destination , including the queueing time spent at the source node and intermediate nodes. In

Fig. 4, $L_{h\ (source\ to\ destination)}$ can be computed as

$$L_{h\ (source\ to\ destination)} = t_{ent} + (t_r + W^S_{entry\rightarrow forw} + t_s) + t_w + (t_r + W^M_{back\rightarrow forw} + t_s) + t_w + (t_r + W^D_{back\rightarrow exit} + t_s) + t_{exit} \quad (5)$$

Where $W^N_{i\rightarrow j}$ is the mean waiting time for a packet from IC^N_i to OC^N_j. Note that in Fig. 4, the channel between S and M can be addressed with OC^S_{forw} or IC^M_{back}.

Once the head flit arrives at the destination, the flow pipeline cycle time is determined by the maximum of the switch delay and wire delay. For an input-only or output-only buffered router, this cycle time would be given by the sum of the switch and wire delays [7]. In other words, in an input-output buffered router L_b is given by

$$L_b = (m-1)x \max(t_s, t_w) \quad (6)$$

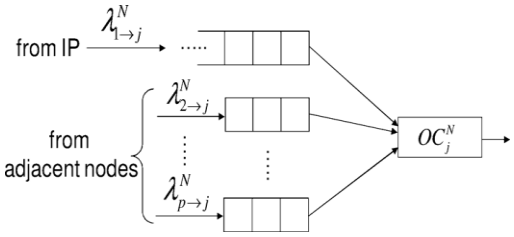
and in an input-only or output-only buffered router it is

$$L_b = (m-1)x \max(t_{s+} t_w) \quad (7)$$

The only unknown parameter for computing the latency is $W^N_{i\rightarrow j}$. This value can be calculated using a queueing model.

B. Waiting Time Estimation

A router is primarily modeled based on non-preemptive priority queuing system. Let us consider, for instance, the jth output channel of $R^N(OC^N_j)$. As can be seen in Fig. 5, this



Channel is modeled as a server in a priority queueing system with p classes (IC^N_1 to IC^N_p), the arrival rate $\lambda^N_{i\rightarrow j}(1 \le i \le p)$, and served by one server (OC^N_j) of service rate μ^N_j. Both inter_arrival and service times are independent and identically distributed with arbitrary distributions. The queueing model for output channel represented in Fig. 5 is different from traditional priority queue model in Fig. 1. Since in the wormhole routing every input buffer can hold finite number of flits, we cannot use (2) and we have to compute the average waiting time for the head of class in this special case of priority queues. Using a technique similar to that employed in the literature for general priority queues [3], [22], we can write

$$W^N_{i\rightarrow j} = \begin{cases} \bar{R}^N_j / \left(1-\rho^N_{1\rightarrow j}\right), & i=1 \\ \bar{R}^N_j \Big/ \left(1-\Sigma^{i-1}_{k=1}\rho^N_{k\rightarrow j}\right)^2, & 2 \le i \le p \end{cases} \quad (8)$$

Where $\rho^N_{k\rightarrow j}$ is the fraction of time that the is occupied by packets from and equals

$$\bar{R}^N_j \approx \rho^N_j \left(C^2_A + C^2_{S^N_j}\right) \Big/ 2\mu^N_j. \quad (9)$$

It is obviously, average packet rate to an output channel of is equal to sum of the average packet rate from all input channel of to this output channel. Therefore, we can write delay time by substituting in (8) . As a result, (8) can be rewritten as

$$W^N_{i\rightarrow j} = \begin{cases} \frac{\rho^N_j \left(C^2_A + C^2_{S^N_j}\right)}{2\left(\mu^N_j - \lambda^N_{i\rightarrow j}\right)}, & i=1 \\ \frac{\lambda^N_j \left(C^2_A + C^2_{S^N_j}\right)}{2\left(\mu^N_j - \Sigma^{i-1}_{k=1}\lambda^N_{k\rightarrow j}\right)^2}, & 2 \le i \le p. \end{cases} \quad (10)$$

Therefore, to compute the $W_{i\to j}^{N}$ we have to calculate the arrival rate from IC_i^N to $OC_j^N(\lambda_{i\to j}^N)$ and also first, and second moments of the service time of $OC_j^N(S_j^N,(S_j^N)^2)$. In the following two subsections, packet arrival rate and channel service time are computed.

C.Packet Arrival Rate

Assuming the network is not overloaded, the arrival rate from to can be calculated using the following general equation

$\lambda_{i\to j}^{N}$ =∑∑ $^{\lambda}$ X P X R (source → destination, IC^N_i to OC^N_j) **(11)**

In (11) , the routing function R (source → destination , IC^N_i to OC^N_j) =1 if a set of flits from IP source to IP destination transfer from IC^N_i to OC^N_j , otherwise 0.Note that we suppose a deterministic routing algorithm, the function of R (source → destination , IC^N_i to OC^N_j) can be represented ,regardless of topology and routing The average set of flits rate to OC^N_j can determined as

$$\lambda_j^N = \Sigma_i \lambda_{i\to j}^N. \quad \textbf{(12)}$$

D. Channel Service Time Estimation

After calculating the set of flits arrival rates, now we emphasize on the calculation of the moments of channel service times. At first, we assign a positive integer index to every output channel. Let D^N_j be the set of all possible destinations for a packet which transfers through OCNj. The index of OC^N_j is equal to the maximum of distances among N and M each where M € D^N_j. Obviously, the index of a channel is between one and diameter of the network. In addition, the index of all exit channels is assumed to be zero. After that, all output channels are divided into some groups k based on their index numbers, so that group contains all channels with index k.

Determination of the channel service time moments starts at group zero (exit channels) and works in increasing order of group numbers. Therefore, the waiting time from lower numbered groups can then be thought of as adding to the service time of packets on higher numbered groups' k-1. In other words, to determine the delay time of channels in group, we have to calculate the delay time of all channels in the group. This approach is independent of the network topology and works for all kind of deterministic routing algorithm where minimal or non-minimal. Finally, channel service time for OC^M_j [8]

$$C^2_{s_i^M} = \overline{\left(S_i^M\right)^2} \Big/ \left(\bar{S}_i^M\right)^2 - 1. \quad \textbf{(13)}$$

Now, we are able to compute the average waiting time of all output channels using (10). Computing delay time for every nodes and channels, the average packet latency between any two nodes in the network, , latency can be calculated. The average packet latency is the weighted mean of these latencies

$\mathbf{L_a}$ = ∑∑ **P** (source → destination) x **L**(source → destination) **(14)**

Let we assume that a node i (source) ,wishes to send a packet of size m to annoter node j (destination). The time to need the transfer a packet along a network link is roughly linear in the packet length . Therefore, the time required to transfer a packet along a given route is also linear in the packet length. Required to transfer a packet of

size m from node i →j is often modeled as

$$C_{i,j}(m) = L_{i \to j} + m/B_{i,j}$$
$$= L_{i \to j} + m\, b_{i,j} \quad (15)$$

Where **L** start-uptime is also called latency expressed in second and $B_{i,j}$ in bytes per second is the bandwidth expressed in bytes per second

VI. CONCLUSION

The use of analytical models can potentially address these limitations under certain assumptions. We propose the Performance Queueing model for predicting the communication performance of wormhole-routing in embedded hypercube platforms. This queueing theory based model takes as input: 1) an application communication graph; 2) a topology graph; 3) a length route, and 4) a communication protocol and no. of links can be used, and calculates some performance metrics of the system such as average set of flits (packet) latency. The model independency on network topology and workload type makes it a robust tool to explore the huge design space of embedded -based systems. In many applications such as real-time systems, the worst case execution time is of particular concern since it is important to know how much time might be needed in the worst case to guarantee that the task will always finish its jobs before the predetermined deadline. Therefore, we plan to advance this research by integrating the proposed average case model with an analytical worst case model.

REFERENCES

[1] William J.Dally " performance analysis of k-ary n-cube interconnection network", IEEE Transaction comp. vol 39, No. 6 June 1990..

[2] A. E. Kiasari, A. Jantsch, and Z. Lu, "A framework for designing congestion-aware deterministic routing," in *Proc. Int.Workshop Netw.-on-Chip Arch. (NoCArc), Held in Conjunction With IEEE/ACM Int. Symp. icroarch. (MICRO-43)*, 2010, pp. 45–50

[3] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains: Modeling and Performance Evaluation With Computer Science Applications*, 2nd ed. New York: Wiley, 2006.

[4] Ahmed Louri and Hongki Sung, "An Optical Multi-Mesh Hypercube: A Scalable Optical interconnection Network for Massively Parallel Computing," Journal of Lightwave Technology, Vol.12, No.4, Apr.1994, pp.704-716.

[5] Ahmed Louri and Hongki Sung, "A scalable optical hypercube-based interconnection network for massively parallel computing," Applied optics, Vol.33, No.11, Nov.1994.

[6] N. Gopalakrishna Kini, M. Sathish Kumar and Mruthyunjaya H.S., "A Torus Embedded Hypercube Scalable Interconnection Network for Parallel Architecture," IEEE explore conference publications, Mar.2009,

[7] J.Duato, C.Yalamanchili, and L.Ni, *Interconnection Networks: An Engineering Approach*. San Francisco, CA: Morgan Kaufmann, 2003.

[8].Abbas Eslami Kiasari,Zhonghai Lu, Axel ,"An analytical latency model for networks on chips ",IEEE transaction on VLSI.

[9] S. H. Kang and D. K. Sung, "Two-state MMPP modelling of ATM

superposed traffic streams based on the characterisation of correlated interarrival times," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, 1995, pp. 1422–1426.

[10] J. H.Bahn andN.Bagherzadeh, "A generic trafficmodel for on-chip interconnection networks," in *Proc. Int. Workshop Netw.-on-Chip Arch.(NoCArc), Held in Conjunction With the IEEE/ACM Int. Symp. Microarch.(MICRO-41)*, 2008, pp. 22–29.

[11] J.Duato, C.Yalamanchili, and L.Ni, *Interconnection Networks: An Engineering Approach*. San Francisco, CA: Morgan Kaufmann, 2003.

[12] N. Gopalakrishna Kini "*Performance Metrics Analysis of Torus Embedded Hypercube Interconnection Network",* International Journal on Computer Science and Engineering Vol.1(2), 2009, 78-80

[13] J. Kim and C. R. Das, "Hypercube communication delay with wormhole routing," *IEEE Trans. Comput.*, vol. 43, no. 7, pp. 806–814, Jul. 1994.

[14] L. Kleinrock, *Queueing Systems*. New York: Wiley, 1975, vol. 1.

[15] S. Murali, T. Theocharides, N. Vijaykrishnan, M. J. Irwin, L. Benini, and G. De Micheli, "Analysis of error recovery schemes for networks on chips," *IEEE Design Test Comput.*, vol. 22, no. 5, pp. 434–442, Sep./Oct. 2005.

[16] U. Y. Ogras, P. Bogdan, and R. Marculescu, "An analytical approach for network-on-chip performance analysis," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 29, no. 12, pp. 2001–2013, Dec. 2010.

[17] A. E. Kiasari, H. Sarbazi-Azad, and M. Ould-Khaoua, "An accurate mathematical performancemodel of adaptive routing in the star graph," *Future Generation Comput. Syst.*, vol. 24, no. 6, pp. 461–474, 2008.

[18] M. Palesi, R. Holsmark, S. Kumar, and V. Catania, "Application specific routing algorithms for networks on chip," *IEEE Trans. Parall. Distrib. Syst.*, vol. 20, no. 3, pp. 316–330, Mar. 2009.

[19] P. P. Pande, C.Grecu,M. Jones, A. Ivanov, and R. Saleh, "Performance evaluation and design trade-offs for network-on-chip interconnect architectures," *IEEE Trans. Comput.*, vol. 54, no. 8, pp. 1025–1040, Aug. 2005.

[20] K. Pawlikowski, "Steady-state simulation of queueing processes: A survey of problems and solutions," *ACM Comput. Surveys*, vol. 22, no. 2, pp. 123–170, 1990.

[21] V. Soteriou, H. Wang, and L.-S. Peh, "A statistical traffic model for on-chip interconnection networks," in *Proc. IEEE Int. Symp. Model., Anal., Simulation Comput. Telecommun. Syst.*, 2006, pp. 104–116.

[22] H. Takagi, "Queueing Analysis. vol. 1: Vacation and Priority Systems," Amsterdam, 1991.