High level Hardware/Software Communication Estimation in Shared Memory Architecture

Sujan Pandey, Heiko Zimmer, Manfred Glesner, Max Mühlhäuser†
Institute of Microelectronics Systems, Karlstr. 15, D-64283, Darmstadt
†Telecooperation Group, Hochschulstr. 10, D-64283, Darmstadt
Darmstadt University of Technology, Germany
{pandey,heiko,glesner}@mes.tu-darmstadt.de, max@informatik.tu-darmstadt.de

***Abstract*— This paper presents a method of modeling on-chip communication behavior of a system as a set of communicating processes to find an optimal bus width and interface buffer size for the communication bus. An assumption for the modeling is that the system has already been partitioned and mapped onto the appropriate components of a SoC. We parameterize the communication behavior of a mixed Hw/Sw system considering the amount of data to be transferred, on-chip bus width, computation time of synthesized hardwares, bus topology and bus protocol. With these parameters we estimate the transition probabilities between the communicating processes and model the overall communication behavior of a system by Markov chain. This model is used to estimate the round trip communication delay and buffer size in the bus-interfaces for different bus widths. From the estimated figures we select the optimal values for those parameters that satisfy given design constraints. The results of applying this approach to an Ogg Vorbis decoder clearly demonstrate the utility of our techniques for modeling communication behavior in order to estimate the bus width and buffer requirements of a complex system.**

I. Introduction

After partitioning of a complex system into hardware and software, and mapping onto the appropriate blocks of SoCs, part of the system functions are implemented in software that runs on a standard processor core while the rest of the system functions are implemented in (a synthesized) hardware. At several time instances in software program a hardware routine is called to transfer a control from software to hardware and after the execution of certain task in a hardware control is given back to the software. When the control is transferred either from hardware or software, data has also to be transferred to software or hardware since both sides may depend on the same data. An abstract model of hardware and software communication [1] can be seen in Fig. 1. In this model, set of communicating processes at left and right side represent the software and hardware model of a system, respectively. Between hardware and software there is a shared memory (which is used for the communication between them).

In this paper we propose a method to model such a communication behavior of Hw/Sw as a set of communicating processes in Markov chain, where each process communicates with other process with a certain probability, which we call a transition probability. A process can have transition probability to itself or to other process or both of them. We define that a process with a higher transition probability to itself and a lesser to other processes means that the process spends most of its time by doing a certain task and interacts very seldom with other processes. These process transition probabilities depend directly on the available on-chip communication resources so we parameterize them with parameters like size of data to be transferred, bus width, computation time of a hardware, bus topology and bus protocol. We estimate the transition probabilities of an individual resource by taking into account those parameters and model the communication behavior in Markov chain. In Markov chain we estimate expected round trip communication delay (RTCD) for mixed Hw/Sw system, which includes time spend in hardware, bus interface, bus and shared memory. Hence the total Hw/Sw communication delay of a system can be calculated by multiplying RTCD with the number of times hardware is called. The number of calls time is estimated by simply profiling a software of an application in a standard profiling tool.

To show the feasibility of our approach, we model Hw/Sw communication behavior of Ogg Vorbis decoder [4] and estimate different RTCD and buffer size of an interface for several possible bus width. From the estimated RTCD and buffer size we choose an optimal size of bus width and buffer size at input and output of interface to keep overall Hw/Sw communication delay less than or equal to the given run time delay constraint.

The remainder of this paper is organized as follow. In section II, we described related work and compare our approach with current approaches. In section III, we give an introduction to Markov chain for modeling system behavior. In section IV, we show how to estimate expected time spend in a process, expected round trip communication delay (in term of cycle and transition probabilities) of hardware and communication interface in Markov chain model. In section V, we present an experimental results to approve our method of Hw/Sw communication delay estimation for selecting an optimal on-chip communication parameters. In section VI, we give conclusion of this paper and possible future research.

II. Related work

Recent approaches to on-chip communication synthesis and analysis mainly focus to get the best partitioning result of a complex system [1,2,3]. For instance, Henkel and Ernst [1] present Hw/Sw communication delay estimation for a shared memory architecture by separating a system into cluster of hardware and software. They estimate the communication delay by analyzing variables which are defined in a process and going to be used by another process. In terms of communication model our approach is similar but they do not estimate the delay considering on-chip resources of the target architecture. These approaches are, therefore, more concerned to the Hw/Sw partitioning.

In [5] Lahiri et al. propose a method to find a communication architecture after the system has been partitioned into Hw/Sw and mapped onto the appropriate components. Their approach estimate the communication behavior by mapping a system into several available communication templates, and takes the one which fulfills the requirement best. This approach focus mainly the problem of finding communication topology and protocol.

In [2] Knudsen and Madsen present a communication protocol selection method to be used before partitioning of a complex system. They compare several communication protocols by modeling system bus and interfaces, which is something related to the model we consider but still lacks estimating overall Hw/Sw communication

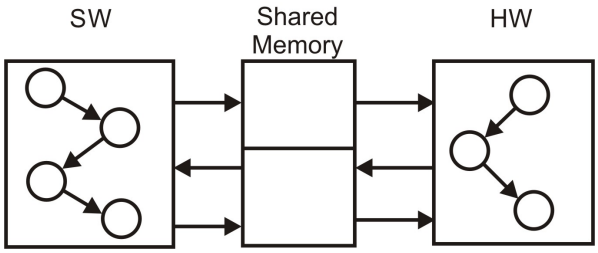

Fig. 1. Shared memory based Hw/Sw communication model

delay to find an optimal size of bus width and buffer size requirement at input and output of bus interfaces. Our approach presents a method to model on-chip Hw/Sw communication behavior which is parameterized considering bus protocol, bus interface, hardware, software, bus width and bus structure to find an optimal size of bus width and buffer size by analyzing round trip communication delay for different design alternatives.

III. Markov chain based system modeling

In this section, we present the mathematical formulation of our MC (Markov Chain) based system modeling approach. Specifically, we show that the specification of a system can be modeled as a set of communicating processes P as shown in Fig. 2, with their transition probability from one process to other process(es), which is represented by the transition matrix M. We then present a method to estimate expected cycles spent in a process and expected communication delay between two processes. The details about these terms are explained in the following section.

A. Discrete time Markov chains

Consider a *finite state space* $S = \{1, 2, \cdots, N\}$. In the *discrete-time* domain, a random process X on state space S is a sequence of random variables each taking values in S, i.e., $X = \{X_n \in S : n \geq 0\}$. We call the value assumed by the random variable X_n the state of X at time instant n. Process X is a *Markov chain* if it satisfies the *Markov property*, i.e., its future evolution depends only on the current state. More formally, X is a Markov chain if for all n ≥ 1 and all $j, i, x_{n-1}, \cdots, x_1, x_0 \in S, Prob(X_{n+1} = j | X_n = i, (X_{n-1} = x_{n-1}, \cdots, X_1 = x_1, X_0 = x_0)) = Prob(X_{n+1} = j | X_n = i)$, where Prob(A|B) denotes the conditional probability that A occurs given that B occurs. When x_i takes only finite discrete values, a Markov sequence of a system can be generated by constructing a state transition graph (STG), in which every distinct discrete value is represented by a vertex. In this paper we assign each vertex as a process of a system which performs a certain task. The Markov chain model of a system with three processes a, b and c with their corresponding transition probability matrix M is,

$$M = \begin{pmatrix} p_{a,a} & p_{a,b} & p_{a,c} \\ p_{b,a} & p_{b,b} & p_{b,c} \\ p_{c,a} & p_{c,b} & p_{c,c} \end{pmatrix}$$

Where the matrix element $p_{a,b}$ is the transition probability of the process a to the process b.

IV. Communication delay estimation methods

A. Estimation of expected time spent in a process

Suppose X_n be a chain of processes with transition matrix M. Consider the amount of time spent in process j up through time n is,

$$Y(j, n) = \sum_{m=0}^{n} I\{X_m = j\} \tag{1}$$

Here notation I denotes the "indicator function" of an event, i.e., the random variable which equals 1 if the event occurs and 0 otherwise.

Suppose π is a limiting probability vector, i.e., for some initial probability vector v, $\pi = lim_{n\to\infty} v M^{n+1} = (lim_{n\to\infty} v M^n) M$. We call a probability vector π an invariant probability distribution for M if $\pi = \pi M$. Such a π is also called a stationary probability distribution and can be estimated for all processes,

$$lim_{n\to\infty} \frac{1}{n+1} E(Y(j, n) | X_0 = i) = \pi(j) \tag{2}$$

i.e., $\pi(j)$ represents the fraction of time that the chain spends in process j.

B. Expected number of cycles estimation

Let M be the transition matrix of a system S which has cluster of transient and recurrent process. These clusters are obtained from the process classification method [6,7]. In this section we estimate an average number of steps require to reach process j from process i. This can be estimated by arranging the matrix M to the group of sub matrices A, S and Q so that sub matrix Q is a group of processes, which visit process j with probability 1.

$$M = \begin{pmatrix} A & R \\ S & Q \end{pmatrix}$$

Let i be a process of sub matrix Q and consider Y_i, the total number of visits to i is,

$$Y_i = \sum_{n=0}^{\infty} I\{X_n = i\}. \tag{3}$$

As the matrix M is re-arranged assuming that all the processes of Q will visit the processes of A, i.e., $\forall i \in Q$ act as transient process and $Y_i < \infty$ with probability 1. Suppose $X_0 = j$, where j is another process of Q then,

$$\begin{aligned} E(Y_i | X_0 = j) &= E[\sum_{n=0}^{\infty} I\{X_n = i\} | X_0 = j] \\ &= \sum_{n=0}^{\infty} P\{X_n = i | X_0 = j\} \\ &= \sum_{n=0}^{\infty} p_n(i, j). \end{aligned} \tag{4}$$

In other words, $E(Y_i | X_0 = j)$ is the (j, i) entry of the matrix $I + M + M^2 + \cdots$ which is the same as the (j, i) entry of the matrix $I + Q + Q^2 + \cdots$. However, a simple calculation shows that,

$$\begin{aligned} (I + Q + Q^2 + \cdots)(I - Q) = I \\ or\ (I + Q + Q^2 + \cdots) = (I - Q)^{-1} = M \end{aligned} \tag{5}$$

Let T_i be the number of steps require to reach process i. In other words, T_i is the smallest time n such that $X_n = i$. For any other process k let $T_{i,k}$ be the number of visits to k before reaching i (if we start at process k, we include this as one visit to k). Then,

$$\begin{aligned} E(T_i | X_0 = j) &= E(\sum_{k \neq i} T_{i,k} | X_0 = j) \\ &= \sum_{k \neq i} M_{jk}. \end{aligned} \tag{6}$$

Hence, the expected number of steps for process j until it reaches process i is the sum of M_{ji}.

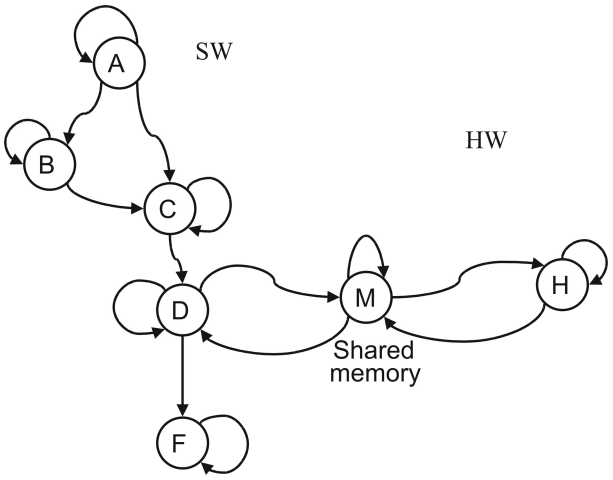

Fig. 2. Communication model based on Markov chain

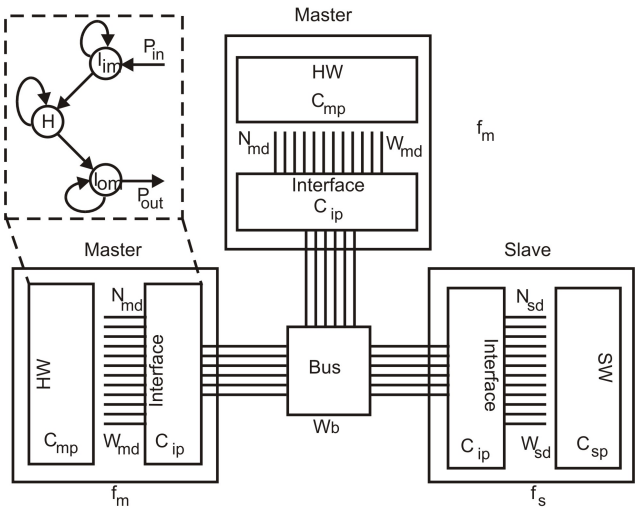

Fig. 3. On-chip communication resources and their interconnection

C. Hardware and communication interface modeling

In this section, we present a mathematical expression for estimating transition probability of on-chip communication resources. Before that we define our on-chip communication model as shown in Fig. 3, which consists of one global shared memory as a slave and synthesized hardwares as a master. These masters and slave communicate with each other by sharing a single communication bus. Each master and slave have their own bus interface to establish communication between them. We consider each interface as a set of two communicating processes, where one process is the input buffer to receive the incoming data and other process is the output buffer to send computed data to the slave as shown in Fig. 3. For the simplicity of modeling communication between master and slave we keep parameters such as size of data to be transfered and cycles need for a hardware to compute certain data constant. That means resources of hardware are fixed and cannot be changed. That's why transition probabilities of hardware in master and slave are constant. But, parameters such as on-chip bus width and size of buffer requirement in interfaces are variable, which we need to find by comparing estimated communication delay with the given run time constraint.

Assume that P[h|h] and P[I_{om}|h] are the transition probabilities to hardware and out terminal of interface given the condition that it was in hardware. Similarly, P[m|m] and P[I_{os}|m] are the transition probability to memory and output terminal of slave's interface given the condition that it was in memory. Similarly, other probabilities can be denoted and estimated as,

$$P[h|h] = \frac{C_{hp}}{C_{hp} + C_{I_{om}}} \tag{7}$$

$$P[I_{om}|h] = 1 - P[h|h] \tag{8}$$

$$P[I_{is}|I_{om}] = \frac{C_{ipsm} \times f_m/f_s \times W_{md}/W_b \times P_{bg}(m_i)}{C_{gw} + C_{ipsm} + C_{syn}} \tag{9}$$

$$P[I_{om}|I_{om}] = 1 - P[I_{is}|I_{om}] \tag{10}$$

$$P[I_{im}|I_{os}] = \frac{C_{ipss} \times f_s/f_m \times W_{sd}/W_b \times P_{bg}(m_i)}{C_{ipss} + C_{gw} + C_{syn}} \tag{11}$$

$$P[h|I_{im}] = P[I_{im}|I_{os}] \times W_b/W_{md} \times P[I_{om}|h] \tag{12}$$

$$P[I_{im}|I_{im}] = 1 - P[h|I_{im}] \tag{13}$$

where,
C_{ipsm} = N_{md}/W_b, $C_{ipss} = N_{sd}/W_b$

C_{hp} : data processing cycles for hardware
C_{ipsm} : data send cycles for I_{om} in master
C_{ipss} : data send cycles for I_{os} in slave
$C_{I_{om}}$: data write cycles from hardware to I_{om}
C_{syn} : data transfer sync. cycles
N_{md} : size of data to be transfered to master per session
W_{md} : width of data storage in master
W_b : bus width
C_{gw} : wait cycles for granting bus
N_{sd} : size of data to be transfered to slave per session
W_{sd} : width of data storage in slave
$P_{bg}(m_i)$: bus granting prob. for master m_i
f_m, f_s : operating frequency of master and slave

V. Experiments and results

For an experimental purpose we chose the Ogg Vorbis decoder which is widely used for audio compression in multimedia applications. The Ogg Vorbis decoder includes four main decoding steps which are inverse quantization, channel decoupling, reconstruct curve and IMDCT (Inverse Modified Discrete Cosine Transformation) [4]. Since our approach considers a system that has already been partitioned and mapped onto the hardware and software part of system-on-chip components, the software part includes inverse quantization, channel decoupling and reconstruct curve which are stored in a memory and the hardware part includes IMDCT (Inverse Modified Discrete Cosine Transformation). The overall on-chip modules and their interconnection for an implementation of Ogg Vorbis decoder is shown in Fig. 3. It has two masters and one slave which are connected to a single on-chip bus. One master is a processor which does inverse quantization, channel decoupling and reconstruct curve; and other master is the IMDCT which recovers data from frequency domain to time domain. These two masters communicate with each other by sharing the data through shared memory, which acts as a slave in our example.

Since IMDCT is the computation intensive part of the Ogg Vorbis decoder, it is called several times to compute the huge amount of data. Obviously, if the width of the data bus is high enough, only a small number of cycles need to transfer certain amount of data, other side we need to also insert the same proportion of buffer at the input and

$W_b(i)$	P[m|I_{is}]	P[I_{os}|I_{os}]	P[I_{im}|I_{os}]	P[I_{is}|I_{is}]	P[I_{is}|I_{om}]	P[I_{om}|I_{om}]	P[I_{im}|I_{im}]	P[h|I_{im}]	E[Cyc]	E[Buf-I_{im}]	E[Buf-I_{om}]
8 bit	0.206	0.79	0.825	0.79	0.527	0.47	0.871	0.129	153	8	4
16 bit	0.139	0.721	0.278	0.86	0.864	0.13	0.913	0.008	118	12	3
24 bit	0.104	0.86	0.139	0.89	0.936	0.06	0.934	0.006	111	15	2
32 bit	0.084	0.915	0.084	0.91	0.963	0.03	0.947	0.005	107	19	2
40 bit	0.007	0.943	0.056	0.92	0.976	0.02	0.956	0.044	102	23	2
48 bit	0.006	0.959	0.040	0.93	0.983	0.016	0.962	0.037	99	26	1
64 bit	0.047	0.97	0.023	0.95	0.99	0.009	0.97	0.029	97	33	1

TABLE I

TRANSITION PROBABILITY FOR DIFFERENT BUS WIDTH

$W_b(i)$	cycles	E[Cyc]	E[Buf-I_{im}]	E[Buf-I_{om}]	Sec.
8 bit	8700704	153	8	4	26.62
16 bit	8700704	118	12	3	20.53
24 bit	8700704	111	15	2	19.31
32 bit	8700704	107	19	2	18.61
40 bit	**8700704**	**102**	**23**	**2**	**17.74**
48 bit	8700704	99	26	1	17.22
64 bit	8700704	97	33	1	16.87

TABLE II

ROUND TRIP COMMUNICATION DELAY ESTIMATION

output of bus interface if the processing speed of IMDCT is fixed. Here, we focus to analyze a round trip delay due to communication between shared memory and IMDCT by assuming first, amount of data transfer between them is fixed in each session and the processing speed of IMDCT and memory are kept constant. i.e. the transition probabilities of hardware and memory are P[h|h] = 0.75, P[$I_{om}|h$] = 0.25, P[m|m] = 0.3 and P[$I_{os}|m$] = 0.7. These data are taken from the HDL simulation of the IMDCT and the shared memory together. In addition to this, the operating frequency of both master and slave are equal and the size of data to be transferred each time from slave to master and master to slave are 384 and 128 bit, respectively, which are constant for each session. Average wait cycles for the bus synchronization C_{syn} and average wait cycles for a master to get the bus granted C_{gw} are 33 and 11 in each session, respectively. Based on the described method in section IV (c), we evaluated the transition probabilities of each interface and hardware for different size of on-chip bus width W_b as shown in Table I. With these transition probabilities we modeled the communication behavior of IMDCT and shared memory in Markov chain and estimated expected round trip communication delay in term of cycles E[Cyc], expected buffer size in input and output of interface E[Buf-I_{im}] and E[Buf-I_{om}] respectively, based on the method described in section IV. Where the expected number of cycles E[Cyc] is generalized to the system clock and expected number of buffer E[Buf-I_{im}] and E[Buf-I_{om}] are generalized to size of 8 bit register. In Table I, it can be seen clearly that when size of bus width increases, expected number of round trip communication cycles E[Cyc] and amount of data flow per cycle also increases. This results requirement of more buffers at the input of bus interface in master and slave. Unlike this requirement of buffer size at the out of interface decreases with increasing size of bus width. This is because rate of data flow from the hardware to the output of interface is constant.

We profiled an audio data of 38 s length using Ogg Vorbis decoder and found that the IMDCT is called 8700704 times which needs almost 48 % of the total processing time, i.e. 18.24 s is a run time constraint for IMDCT. We assume that in real implementation of Ogg Vorbis decoder the hardware (IMDCT) is also called approximately 8700704 times with system frequency $f_{m,s}$ = 50 MHz and average bus grant to IMDCT is 63 % then the total communication time for hardware and shared memory with different bus width is shown in Table II. Where column Sec. gives different communication delay for different width of bus. According to the given run time constraints of IMDCT, bus width of 40 bit, expected cycles 102, buffer size at input and output of interface are 23 and 2 were chosen, respectively. In this case, the total expected communication delay (17.74 s) between shared memory and IMDCT is less than the given run time constraint 18.24 s.

VI. CONCLUSION

At every abstraction level of a complex system design flow, analysis of communication behavior is an important task to identify the key design decisions like low power, adequate performance, small size etc. In this paper, we focused our work to analyze communication behavior of a system that has been already partitioned and mapped onto the appropriate blocks of SoC. We presented a method to model on-chip communication behavior of hardware/software components, which estimates round trip Hw/Sw communication delay and size of buffer at the input and output terminal of communication interface for different on-chip bus width, and selects optimal size of bus width and buffer size to satisfy the run time constraints of a system. We used our method to model on-chip communication behavior of Ogg Vorbis decoder by modeling its communication resources such as shared memory, bus interface, bus and hardware in Markov chain and estimated communication delay, buffer size for different bus width. The results shown in this paper clearly demonstrate, this as an efficient method for finding an optimal size of bus width, buffer size by fulfilling run time constraint.

REFERENCES

[1] J. Henkel and R. Ernst, *High-Level Estimation Techniques for Usage in Hardware/Software Co-Design*, In IEEE/ACM Proc. of Asia and South Pacific Design Automation Conference, pages 353-360 February 1998.

[2] P. V. Knudsen and J. Madsen, *Integrating Communication Protocol Selection with Hardware/Software Codesign* In IEEE Transactions on Computer Aided Design of Integrating Circuits and Systems, Vol. 18, No. 8, pages 1077-1095 August 1999.

[3] T. Y. Yen and W. Wolf, *Communication Synthesis for Distributed Embedded Systems* In IEEE/ACM Proc. of International Conference on Computer Aided Design, pages 288-294, 1995.

[4] Xiph.org Foundation *"Vorbis I Specification."* http://www.xiph.org/ogg/vorbis/doc/Vorbis_I_spec.html.

[5] K. Lahiri, A. Raghunathan and S. Dey, *Design Space Exploration for Optimizing On-Chip Communication Architecture* In IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, Vol. 23, No. 6, pages 952-961, June 2004.

[6] G. F. Lawler, *Introduction to Stochastic Processes*, Chapman and Hall, 1996.

[7] G. Ciardo, *Distributed and Structured Analysis Approaches to Study Large and Complex Systems*, Springer Verlag New York Inc., pages 344-374, 2002.