Bandwidth Tracing Arbitration Algorithm for Mixed-Clock Systems with Dynamic Priority Adaptation

Young-Su Kwon and Chong-Min Kyung
VLSI Systems Lab., CHiPS, KAIST
GuSong-Dong, YuSong-Gu
Taejon, Korea
yskwon@vslab.kaist.ac.kr, kyung@ee.kaist.ac.kr

Abstract

As the processing capabilities and operating frequency of embedded system are growing, so is the needed data bandwidth to fully utilize the processing capability. The ability to transfer huge amount of data between the embedded core and external devices is required for efficient system operation. In this paper, the data communication architecture for the mixed-clock system is proposed. The dynamic priority adaptation algorithm for bus arbitration is proposed for bandwidth guarantee. The communication architecture that incorporates the proposed arbitration algorithm adapts the priority of communication components dynamically based on the information from FIFO. The experiments show that the measured bandwidth of each component traces the required bandwidth well compared to the other arbitration algorithms.

1. Introduction

With the advent of system-on-a-chip (SOC) time-to-market pressure and the requirement of many components for the system functionality made it difficult for a single design team to design the whole system. Therefore, design reuse, or use of IP's (intellectual Properties) has become a nearly mandatory design methodology for the design of electronic systems including embedded systems. The most important requirement of the embedded systems is the performance and cost efficiency. As the embedded systems incorporate a larger number of components and IP's into a single design, the communication time between heterogeneous components such as programmable processors and dedicated hardware components has become the dominant part of the system execution time. The architecture for optimized inter-component communication determines the amount and type of communication between different components of a system.

To reduce the communication overhead in the embedded system design, many communication architectures have been proposed. In [1], the scheduling algorithm for the interacting processes and bus access is proposed. The process interaction includes both data flow and the control flow. The execution time for the flow of control between processes is considered for scheduling of the system execution. The inter-processor communication model and direct memory access was applied to the rapid prototyping of multi-DSP systems in [2]. The buffered communication model for direct inter-processor communication and direct memory transfer is used to schedule the tasks on many processors. These techniques focus on the static scheduling usually based on the control flow graph under the assumption that the system behavior is predetermined and the execution time of each process is predictable.

The arbitration mechanisms for several components that shares common bus were proposed in [3] and [4]. The communication architecture tuner [3] is the additional layer of circuitry that monitors the current status of communication and predicts the relative importance of each communication transaction. The information about the relative importance enables the dynamic scheduling of communication transactions. The Lotterybus in [4] consists of a random arbitration algorithm and lottery manager. It provides low latency and guaranteed bandwidth for the communication architectures. In [5], the data prefetch is applied to the communication between components. The bus wrapper that incorporates the controller and prefetch unit transfers data between components to reduce the read latency. Direct memory access (DMA) architecture was applied to the real-time communication mechanism implemented by software for high-speed data communication in [6]. The pre-programmed DMA that removes the control overhead improves the communication speed of scheduled processes. In [7], the DMA controller is synthesized for a specific application although the synthesis mechanism does not consider the system performance.

In this paper, we propose the communication architecture to guarantee the bandwidth requirement for each communication component in the mixed-clock embedded system. The novel arbitration mechanism, *dynamic priority adaptation* (DPA) is introduced. The proposed communication architecture incorporates the enhanced DMA architecture with dynamic priority adaptation. In section 2, the communication architecture for the mixed-clock system is shown and limitations are explained. In section 3, the communication architecture with the proposed dynamic priority adaptation algorithm is proposed. The experiments are shown in section 4 and the conclusion is in section 5.

2. The Communication Architecture for the Mixed-Clock System

The data transfer between communication components that have difference clock frequencies is usually implemented by First-In First-Out (FIFO) hardware. DMA is a technique by which blocks of data can be transferred from one part of the memory or peripherals to the other part without control of the embedded core. The DMA controller is usually used to transfer data between several FIFO's and the memory of communication component. The conventional data communication architecture for several communication components with DMA controller is shown in Fig. 1.

The clock of DMA is synchronized to the parent communication component in Fig. 1 and moves data for one of the FIFO's. Therefore, the arbitration mechanism that determines priority of FIFO's is important for the high-performance data communication in the mixed-clock system. It is required to design the efficient arbiter that selects the winner FIFO for DMA transfer.

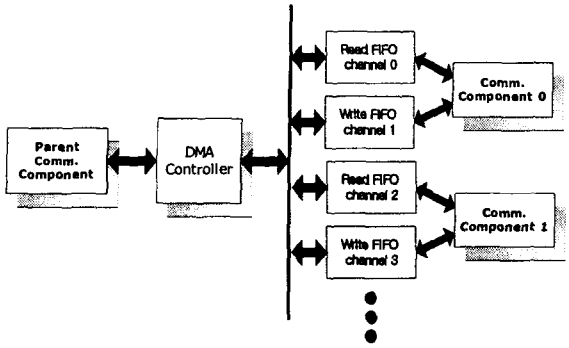

Fig. 1. The mixed-clock data communication system. The DMA controller moves data between FIFO's of several communication components and the parent communication component.

The important issue of the arbiter in the communication architecture is minimization of overall execution time. The scheduling algorithms proposed in [1] and [2] determine the sequence of the execution of each communication component at the compile time. This static scheduling algorithm is not applicable to the general embedded system where the state of each communication component is determined dynamically by external environment. The dynamic behavior requires the arbiter to select the winner for the data communication based on current status of each communication component. Therefore, it is required for the arbiter of the general embedded system to guarantee the bandwidth of each communication component that changes dynamically.

In the mixed-clock system where FIFO is used, it is important to check the FIFO status to guarantee the bandwidth of the communication component.

3. Communication Architecture with Dynamic Priority Adaptation

We designed a communication architecture that incorporates DMA architecture with novel arbitration mechanism, *dynamic priority adaptation*. It arbitrates between a multitude of FIFO's and the parent communication component and determines the transfer parameters by itself. The child FIFO channels are usually the component that comprises the embedded core such as peripheral subsystems and custom ASIC cores operating at different frequencies. For the mixed-clock operation between the child channels and the parent channel, FIFO's are used for burst data transfer. The FIFO for the DPA algorithm has additional signals to report the current status of FIFO to the DMA controller. In this section, we describe the proposed dynamic priority adaptation (DPA) mechanism for FIFO arbitration.

3.1. Architecture

The block diagram of the proposed communication architecture is shown in Fig. 2 with several communication components (CC). The DPA arbiter requests current status of each FIFO at each clock cycle. The selected FIFO channel by DPA arbiter sends the information about internal status of FIFO to FIB. DPA arbiter gathers the information and selects one of them based on dynamic priority adaptation algorithm. The winner candidate information is sent to DMA controller.

The burst length calculator (BLC) computes current burst length of the winner candidate based on the FIFO information of the selected winner candidate.

The DMA controller selects the winner candidate as current winner if the processed data transfer is finished. If the winner candidate is determined, the new data transfer is started with the burst length computed by BLC.

3.2. Dynamic Priority Adaptation

We have defined the current status of FIFO as follows:

- d_i : The depth of i-th FIFO. It is the same to the count of buffer cells inside of the FIFO.
- $c_{i,t}$: The current fill count of i-th FIFO at time t. It is the occupied spaces in the FIFO. Time t is the clock cycle count.
- s_i : The status sampling period. It is the clock cycle count of the interval of checking the current status of i-th FIFO.
- $v_{i,t}$: The fill speed of i-th FIFO. It is the rate of filling of FIFO at clock instant t. The fill speed is defined as

$$v_{i,t} = \frac{\partial c_i}{\partial s_i} \cong (c_{i,t} - c_{i,t-1})\frac{1}{s_i}.$$

  The fill speed is initialized at each sampling point.
- $f_{i,t}$: The fail count of i-th FIFO at time t. When the FIFO becomes full or empty, the communication component fails to transfer data to/from the FIFO when it tries to transfer data. The fail count of FIFO, $f_{i,t}$ is defined as the fail count of communication component for the i-th FIFO during s_i. It is reseted at the sampling point.

The priority of i-th communication component is determined by the following heuristic equation,

$$p_{i,t} = c_{i,t} + (c_{i,t} + v_{i,t}d_i)f_{i,t}s_i.$$

The priority is proportional to the sum of the current fill count, $c_{i,t}$ and the fill speed, $v_{i,t}$. The fill speed is the gradient of the fill count at each sampling point. If fill speed for i-th component is larger, it means that the communication component is likely to fail to transfer the data in the near future. The arbitration algorithm should increase the priority for that component to trace the required bandwidth. In other words, the current gradient field is required to estimate the increment or decrement of the bandwidth of i-th communication component. Therefore, the priority of i-th FIFO should be proportional to the current fill count and the current gradient of fill count.

The priority of the communication component that failed to transfer data with the FIFO is increased by $f_{i,t}$ that represents the frequency of data transfer failure. When $f_{i,t}$ equals to 0 at the beginning of the system operation, the current fill count $c_{i,t}$ becomes the priority value.

To decrease the hardware complexity, the equation can be modified as follows if $s_i << d_i$ (The sampling period in cycle count << The FIFO depth),

The term $(s_i + d_i)$ is constant value for i-th FIFO. Therefore, $p_{i,t}$ can be computed using two multipliers for each FIFO. The decision of sampling period depends on the average variance of the required bandwidth. The large variance means that the bandwidth of each FIFO changes frequently. The sampling period should be small so that DPA arbiter can adapt to the change of bandwidth. The small sampling period does not show good adaptation at all

times. If the sampling period is so small, the fill speed shows the instant behavior. DPA arbiter can not consider the gradual behavior of required bandwidths in this case.

$$
\begin{aligned}
& p_{i,t} \\
& \cong c_{i,t} f_{i,t} s_i + (\Delta c_{i,t} f_{i,t}) d_i \\
& = c_{i,t} f_{i,t} (s_i + d_i) - c_{i,t-1} f_{i,t} d_i \\
& \cong c_{i,t} f_{i,t} (s_i + d_i) - c_{i,t-1} f_{i,t-1} (s_i + d_i)
\end{aligned}
$$

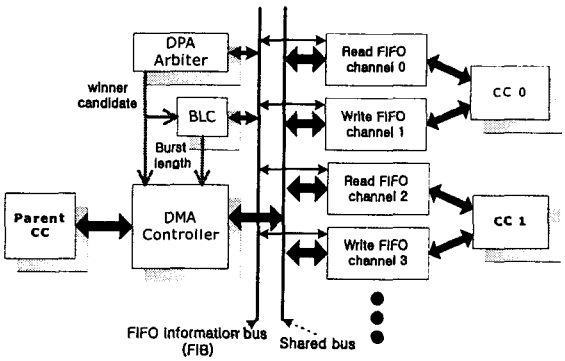

Fig. 2. The proposed communication architecture. The FIFO information bus (FIB) carries the information about FIFO's. DPA arbiter and BLC compute the winner candidate and the burst length based on FIB.

The burst length is also computed based on the FIFO information. The heuristic equation for the burst length computation is as follows.

$$b_{i,t} = \frac{c_{i,t}}{2}\left(1 + \frac{v_{i,t} s_i}{d_i}\right)$$

We can use the fill count of FIFO as the current burst length but it is not preferred because it can starve the other FIFO's. Therefore, the fill speed to the FIFO depth is used to scale the current fill count. The burst length becomes smaller than the current fill count.

This equation can be modified as follows to remove the division operation.

$$b_{i,t} = (c_{i,t} >> 1)(1 + (\Delta c_i >> n_i))$$

where n_i equals $\log_2 d_i$.

4. Experiments

To demonstrate the excellence of the proposed DPA arbitration mechanism, we have implemented the simulator for the mixed-clock data communication. We have simulated several communication components with four arbitration mechanisms.

- ARBIT 0 : Round-Robin arbitration. The fill count for the selected arbitration candidate is used as the burst length.
- ARBIT 1 : Fixed-Priority arbitration. The fill count is used as the burst length.
- ARBIT 2 : Dynamic priority arbitration based on the current fill count that selects the FIFO channel with the maximum fill count as the arbitration winner. The fill count is used as the burst length.
- ARBIT 3 : DPA arbitration.

The example communication architecture has four communication components (CC). The required bandwidth of communication components varies randomly. The bandwidth of DMA controller is 528Mbytes/sec. The sampling period s_i is 32 clock cycles.

The DMA controller starts the transaction as soon as possible the DPA arbiter and BLC finishes the selection of the arbitration winner and the computation of the burst length. The overhead for the arbitration is not large because the arbitration operation is executed with the data transfer in parallel.

The average bandwidth of each component is shown in Table 1.

Table 1. The average bandwidth for each communication component in the example system.

	Required average bandwidth
CC #0	33 data transfers/1500ns
CC #1	50 data transfers/ 3750ns
CC #2	100 data transfers/6000ns
CC #3	63 data transfer/4500ns

The graphs for the required bandwidth and the measured bandwidth of the communication component #1 are shown in Fig. 3. The X axis is time in μs and the Y axis is the bandwidth in data count/μs. The required bandwidth is drawn as the solid line and the measured bandwidth is in dotted line.

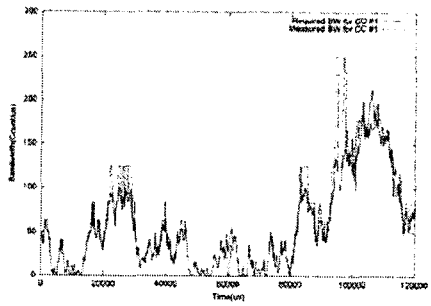

Fig. 3. The trace of the measured bandwidth of CC #1, where DPA is used for the arbitration of communication components.

The measured bandwidth between 80ms and 120ms does not trace the required bandwidth well in case of (a), (b) and (c). The measured bandwidth fluctuates because the DMA controller is in service for the other interfaces and does not serve for interface #1.

MSE (Mean Square Error) is the measure of control and quality and is used as the level of the quality of bandwidth guarantee. MSE equals the mean of the squares of the deviations from target. In this experiment, MSE is the average of square of bandwidth difference at each sampling point. MSE of bandwidth for each component is shown in Table 2. The percentage of MSE compared to ARBIT0 is shown in parenthesis. For each experiment, average bandwidth and bandwidth variance are specified in unit of 1 data transfer/100ns. The cycle count of each experiment is 120000 cycles.

Table 2. MSE of bandwidth for each component.

	CC0	CC1	CC2	CC3
Experiment 1				
BW(1/100ns), Variance (1/100ns)	2.2 , 1.4	1.3, 0.8	1.7, 0.1	1.4, 0.5
ARBIT0	695.1 (100%)	4223.1 (100%)	6453.9 (100%)	81.1 (100%)
ARBIT1	694.2 (99. 8%)	4251.6 (100.6%)	6443.1 (99.8%)	83.1 (115%)
ARBIT2	681.3 (98.0%)	3786.5 (89.7%)	6489.0 (100.5%)	79.8 (98.4%)
DPA	602.5 (86.6%)	3672.1 (86.9%)	5938.2 (92.0%)	38.2 (47.1%)
Experiment 2				
BW(1/100ns), Variance (1/100ns)	3.2, 3.3	1.9, 0.9	2.0, 1.1	1.6, 0.5
ARBIT0	6094.2 (100%)	4478.1 (100%)	5160.2 (100%)	34.2 (100%)
ARBIT1	6180.8 (101%)	4478.2 (100%)	5192.2 (100.6%)	32.9 (96.2%)
ARBIT2	5085.3 (83.4%)	4435.5 (99.0%)	4519.4 (87.6%)	36.6 (107.0%)
DPA	5140.7 (84.3%)	3350.5 (74.8%)	4297.7 (83.3%)	20.4 (59.6%)
Experiment 3				
BW(1/100ns), Variance (1/100ns)	2.2, 1.4	1.9, 2.2	2.0, 1.4	1.6, 1.0
ARBIT0	263.9 (100%)	3840.3 (100%)	14140.5 (100%)	769.3 (100%)
ARBIT1	263.4 (99.8%)	3926.5 (102.2%)	13951.1 (98.6%)	772.9 (100.4%)
ARBIT2	212.3 (80.4%)	4053.3 (105.5%)	13433.2 (94.9%)	205.8 (26.8%)
DPA	152.3 (57.7%)	44.31 (1.1%)	14477.2 (102.4%)	191.2 (24.9%)

5. Conclusion

We have proposed a communication architecture that enables the efficient data communication in the mixed-clock system. The proposed DPA algorithm arbitrates between communication components in the proposed communication architecture. DPA algorithm adapts the priority of communication component and traces the dynamic behavior of bandwidth of each communication component. The graph of the measured bandwidth in example system with DPA shows that the bandwidth using DPA traces the required bandwidth very well. The measured MSE for DPA is about 70% of that of the other arbitration algorithms.

6. References

[1] Petru Eles, Alex Doboli, Paul Pop and Zebo Peng, "Scheduling with Bus Access Optimization for Distributed Embedded Systems," *IEEE Transactions on Very Large Scale Integration Systems*, pp. 472-491, 2000.

[2] Claudia Mathis, Bernhard Rinner, Martin Schmid, Reinhard Schneider and Reinhold Weiss, "A New Approach to Model Communication for Mapping and Scheduling DSP Applications," *International Conference on Acoustics and Speech*, pp. 3354-3357, 2000

[3] Kanishika Lahiri, Anand Raghunathan, Ganesh Lakshminarayana and Sujit Dey, "Communication Architecture Tuners : A Methodology for the Design of High-Performance Communication Architectures for System-on-Chips," *Design Automation Conference*, pp. 513-518, 2000.

[4] Kanishika Lahiri, Anand Raghunathan, Ganesh Lakshminarayana and Sujit Dey, "LOTTERYBUS : A New High-Performance Communication Architecture for System-on-Chip Designs," *Design Automation Conference*, pp. 15-20, 2001.

[5] Roman L. Lysecky, Frank Vahid and Tony D. Givargis, "Techniques for Reducing Read Latency of Core Bus Wrappers," *Design Automation and Test in Europe*, 2000.

[6] Sujaya Srinivasan and David B. Stewart, "High Speed Hardware-Assisted Real-Time Interprocess Communication for Embedded Microcontrollers," *Real-Time Systems Symposium*, pp. 269-279, 2000.

[7] Mattias O'Nils and Axel Jantsch, "Synthesis of DMA Controllers from Architecture Independent Descriptions of HW/SW Communication Protocols," International Conference on VLSI Design, pp. 138-145, 1999.

[8] Dave Comisky, Sanjive Agarwala and Charles Fuoco, "A Scalable High-Performance DMA Architecture for DSP Application," *International Conference on Computer Design*, pp. 414 – 419, 2000.

[9] T. Chelcea and S. Nowick, "Robus Interfaces for Mixed-Timing Systems with Application to Latency-Insensitive Systems," *Design Automation Conference*, pp. 21-26.

[10] L. Carloni, K. McMillan, A. Saldanha and A. Sangiovanni-Vincentelli, "A Methodology for Correct-by-Construction Latency Insensitive Design," *International Conference on Computer-Aided Design*, pp. 307-314, 1997

[11] Kanishika Lahiri, Anand Raghunathan and Sujit Dey, "Efficient Exploration of the SoC Communication Architecture Design Space," *International Conference on Computer-Aided Design*, pp. 424-430, 2000

[12] Roman L. Lysecky, Frank Vahid and Tony D. Givargis, "Experiments with the Peripheral Virtual Component Interface," *International Symposium on System Synthesis*, pp. 221-224, 2000.

[13] Amit Goel and William R. Lee, "Formal Verification of an IBM CoreConnect Processor Local Bus Arbiter Core," *Design Automation Conference*, pp. 196-200, 2000.

[14] Sanjive Agarwala, Charles Fuoco, Tim Anderson, Dave Comisky, and Christopher Mobley, "A Multi-Level Memory System Architecture for High Performance DSP Application," *International Conference on Computer Design*, pp. 408-413, 2000.

[15] J. Rowson and A. Sangiovanni-Vincentelli, "Interface-Based Design," *Design Automation Conference*, pp. 178-183, 1997.

[16] V. Madisetti and L. Shen, "Interface Design for Core-Based Systems," *IEEE Design & Test of Computers*, pp. 42-51, 1997.

[17] F. Vahid and T. Givargis, "Incorporating Cores into System Level Specifications," *International Symposium on System Synthesis*, pp. 43-48, 1998.

[18] Ken Hines and Gaetano Borriello, "Dynamic Communication Models in Embedded System Co-Simulation," *Design Automation Conference*, pp. 395-400, 1997

[19] Kanishka Lahiri, Anand Raghunathan and Sujit Dey, "Fast Performance Analysis of Bus-Based Systems-On-Chip Communication Architectures," *International Conference on Computer-Aided Design*, pp. 566-572, 1999

[20] Drew Wingard, "MicroNetwork-Based Integration for SOCs," *Design Automation Conference*, pp. 673-677, 2001

[21] Drew Wingard and Alex Kurosawa, "Integration Architecture for System-on-a-Chip Design" *Design Automation Conference*, pp. 85-88, 1998

[22] Bill Cordan, "An Efficient Bus Architecture for System-on-Chip Design," *International Conference on Computer Design*, pp. 623-626, 1993.