Design and Implementation of A Reconfigurable Arbiter

YU-JUNG HUANG, YU-HUNG CHEN, CHIEN-KAI YANG, AND SHIH-JHE LIN

Department of Electronic Engineering
I-Shou University,
Kaohsiung, Taiwan, ROC

Abstract: -The SOC design paradigm relies on well-defined interfaces and reuse of intellectual property (IP). Because more and more IPs are integrated into the design platform, the amount of communication between the IPs is on the increase and becomes the source of the performance bottlenecks. The arbiter plays a very important role to manage the resource sharing on the SOC platform. This paper presents a reconfigurable arbiter with various combinations of arbitration algorithms. The performance analysis for the various combinations of the arbitration algorithms under different traffic loads is simulated. The reconfigurable arbiter was implemented by FPGA and synthesized by Synopsys Design Complier with a TSMC 0.18 μm cell library. In addition, the power analysis of the reconfigurable arbiter at various arbitration states is reported. The reconfigurable arbiter can be custom-tuned to obtain high bandwidth utilization, low latency, and power effective for on-chip bus communication.

Key Words:-Arbiter, Reconfigurable, System-on-chip, Arbitration Algorithm, FPGA, AMBA

1. Introduction

Recently, the application of SIP (Silicon Intellectual Property) reuse methodology for system on chip (SOC) has proved its validity by reducing the productivity gap and meeting critical time-to-market objectives, reducing design errors, decreasing system complexity, and easing verification and testing [1]. On-chip bus communication is one of the critical components in a SOC platform. An efficient on-chip communication system has to satisfy the interface behavior of each IP block integrated within the complex SOC. With the increasing number of system components in SOC design, it becomes that an efficient arbiter is one of the most critical factors for high system performance. The conventional bus arbitration algorithms, the static fixed priority and the round robin, show several drawbacks on bus communication such as bus starvation [2].

Currently there exists no system bus standardization. However, communication architectures defined by commercial standards are widespread and available in the market. For example, the PI-Bus [3] of OMI, the AMBA bus [4, 5] of ARM, the FISPbus [6] of Mentor Graphics, the CoreConnect of IBM [7], the SiliconBackplane of Sonics [8], the Wishbone of Silicore [9] and others. The CoreConnect and AMBA make use of a fixed priority arbiter. Lotterybus defines an arbitration method that does not presume any fixed communication topology [10]. Silicon Backplane uses an arbitration method by means of TDMA-based arbitration. Based on the AMBA AHB protocol and a more complex interconnection matrix, the Multi-layer AHB is a different realization of the bus architecture, which enables data transfers between several masters and slaves in a system [11]. Although the arbitration protocol is fixed, the choice of an arbitration scheme is usually depending on the application requirements.

By implementing an efficient arbitration algorithm the system performance can be tuned to better suite the applications. This paper presents the design and implementation of an arbiter with a reconfigurable hybrid arbitration algorithm. The rest of this paper is organized as follows. The architecture of the reconfigurable arbiter is presented in the next section. The performance analysis methodology is described in Section 3. The simulation and implementation results are provided in Section 4 and Section 5, respectively. Finally, we conclude the paper in Section 6.

2. Reconfigurable Arbiter

The Advanced Microcontroller Bus Architecture (AMBA) is an open System-on-Chip bus protocol for high-performance buses on low-power devices. The AHB is a pipelined system backbone bus, designed for high-performance operation. It can support up to 16 bus masters and slaves that can delay or retry on transfers. It consists of masters, slaves, an arbiter and an address decoder. It supports burst and split transfers. The address bus can be up to 32 bits wide, and the data buses can be up to 128 bits wide. The AMBA uses conventional fixed priority arbiter. Fig. 1 shows the architecture of the reconfigurable arbiter presented in this work. As shown in Fig. 1, this reconfigurable arbiter can serve up to a maximum 16 masters and all of them are divided into four groups F1-F4 for the first level competition. According to different arbitration algorithms assigned in F1 ~ F4 blocks, each block will select one master from four input masters for the second level competition. The final granted master will then be determined by the arbitration algorithm chosen in block 5.

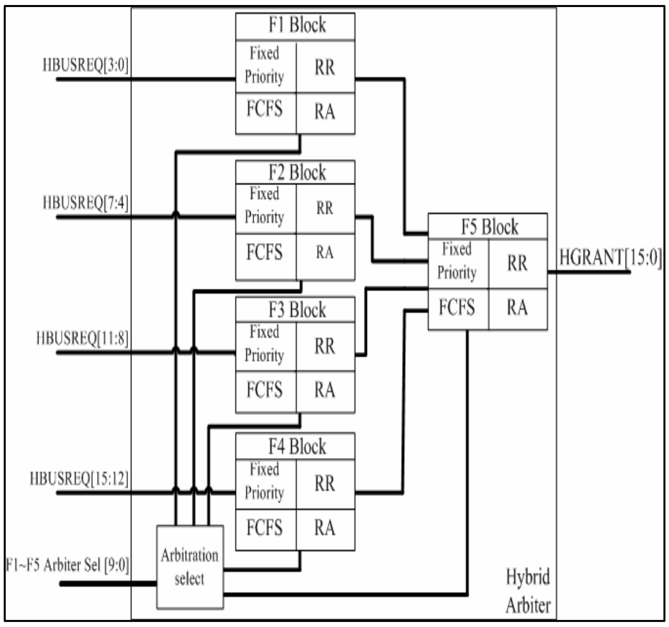


Fig.1 Architecture of the reconfigurable arbiter

In Fig. 1, each group (F1 ~ F5) can be reconfigurable to adapt a specific dynamic algorithm. For example, either one of the round-robin algorithm, random access algorithm or first-come-first-serve algorithm can be combined together to decide the functionality of the reconfigurable arbiter. With the reconfigurable functionality, it can assign different arbitration algorithms for specific groups of masters. In the following, each block (F1 ~ F5) will be assigned a number to denote the status of the reconfigurable arbiter. The notation 1, 2, 3, and 4 are used to represents fixed priority, round robin, first come first serve and random access algorithm, respectively. For example, the arbitration state (12141) denotes the fixed priority algorithm is assigned to functional blocks F1, F3 and F5, round robin algorithm is assigned to functional block F2, and random access algorithm is assigned to functional block F4.

3. Performance Analysis Methodology

A system performance analytical module is proposed to examine the efficiency of the hybrid arbitration algorithm. The simulation strategy for performance analysis of the arbiter with hybrid arbitration schemes is divided into two parts: Traffic Pattern Generation (TPG) and Arbitration Simulator Generation (ASG) as shown in Fig. 2. The traffic patterns are generated based on the statistic distribution such as Bernoulli, Binomial, equilikely, Geometric, Pascal, Possion, Uniform, Exponential, Erlang, Normal, lognormal, Chisquare … etc. The simulations in this paper are based on the TPG which assign various distributions to indicate the data amount and bus request time for each master. For ASG, a behavior bus arbiter model based on Fig. 1 is used to take into account the effect of the shared bus communication architecture on system performance.

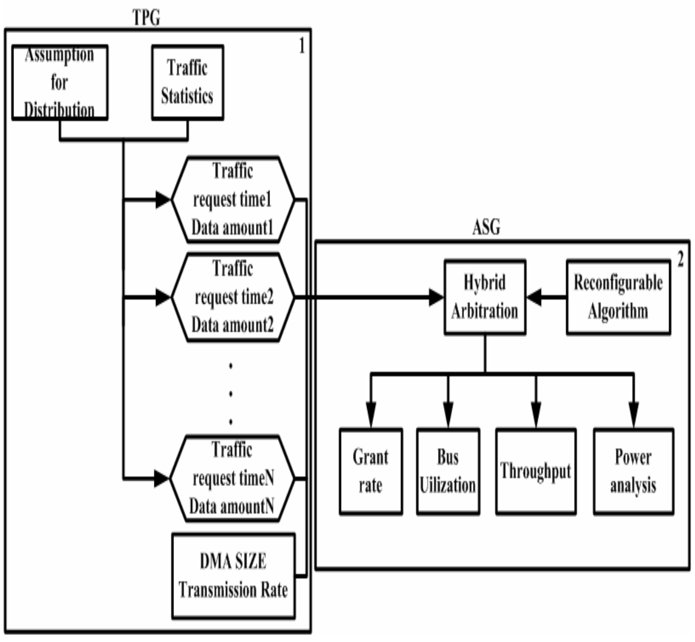

TPG
1
Assumption for Distribution
Traffic Statistics
Traffic request time1 Data amount1
Traffic request time2 Data amount2
Traffic request timeN Data amountN
DMA SIZE Transmission Rate
ASG
2
Hybrid Arbitration
Reconfigurable Algorithm
Grant rate
Bus Uilization
Throughput
Power analysis

Fig. 2 System performance analytical module

The reconfigurable hybrid arbitration algorithms can be achieved by assigning different arbitration algorithms into functional blocks F1 – F5. To study the effects of the various reconfigurable arbitration schemes, the performance analysis are further simulated based on various permutations of arbitration algorithms for blocks F1 ~ F5 under the same traffic distribution pattern. For the evaluation of relative performances of different configuration states, the average waiting time, bus granted rate, throughput, bus utilization can then be calculated.

4. Simulation Results

The grant rate and average waiting time are further simulated based on 1024 permutations of arbitration algorithms for blocks F1 ~ F5 under the same traffic distribution pattern. The request time adopts Bernoulli distribution and the probability is 0.5. The amount of traffic data adopts Equilikely distribution, the mean is 12 and the variance is 6.67. The data transmission rate is 32 bit/s, and the period is 16 clock cycles. Results of the software modeling for various combinations of the arbitration algorithm in function block F1 to F5 are reported in Fig. 3. The x-axis depicts all the possible arbitration combinations where 1, 2, 3, and 4 represent fixed priority, round robin, first come first serve and random access algorithm, respectively. From Fig. 3, it shows that the hybrid arbitration algorithm has a predictable average waiting time. So, user can assign the optimal combination for the application specific tasks. It also shows that the hybrid arbitration scheme with a proper assignment of the arbitration algorithm to F's functional blocks can achieve better performance as compared to traditional arbitration schemes such as fixed priority and round robin.

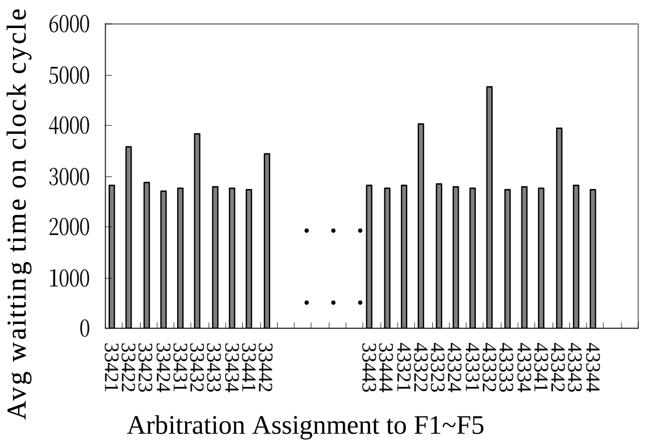


Fig. 3 Average waiting time under hybrid arbitration

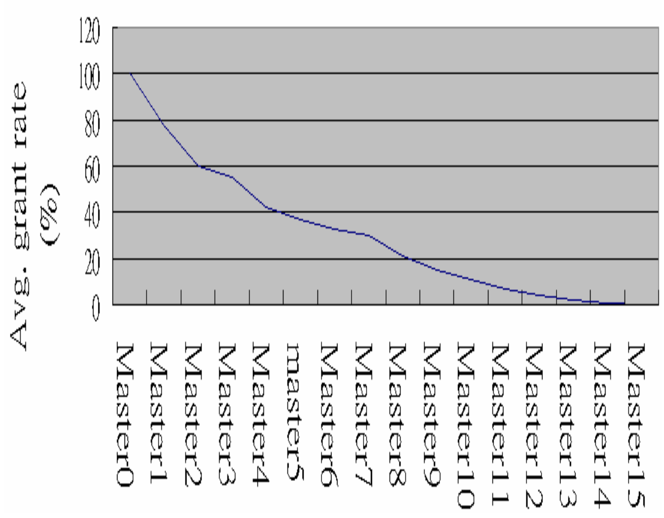


Fig. 4 Grant rate for arbiter configuration state as 11111

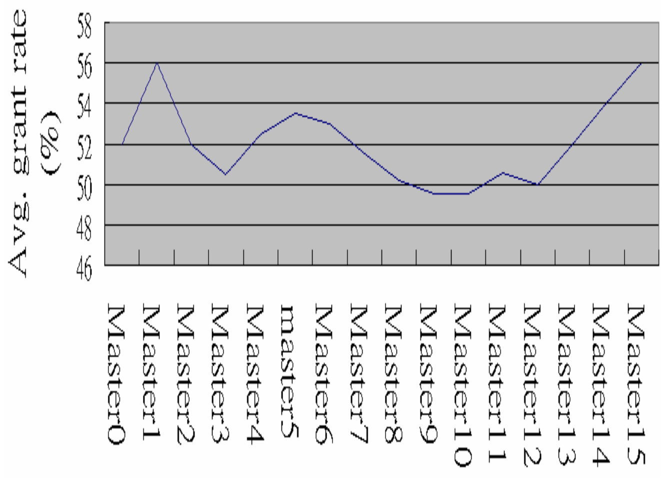


Fig. 5 Grant rate for arbiter configuration state as 23242.

The reconfigurable arbiter with arbitration algorithms for F1 to F5 are 11111 and 23242 in Figs. 4 and 5, respectively. From the average grant rate shown in Fig. 4, the starvation occurs for low priority master (master 15) under fixed priority arbitration scheme. A lower priority master wanting to use a shared resource gets blocked when a higher priority master holds the resource. However, this starvation issue can be resolved by reconfiguring the arbitration state. As indicated in Fig. 5, master 15 can obtain 56 % grant rate with arbitration state 23242, which is much better than the arbitration state 11111.

The arbiter configuration states for F1 to F5 in Figs. 6 and 7 are 22222 and 13133, respectively. As shown in Fig. 7, the average waiting time is much shorter as compared with conventional round robin arbitration scheme indicated in Fig. 6.

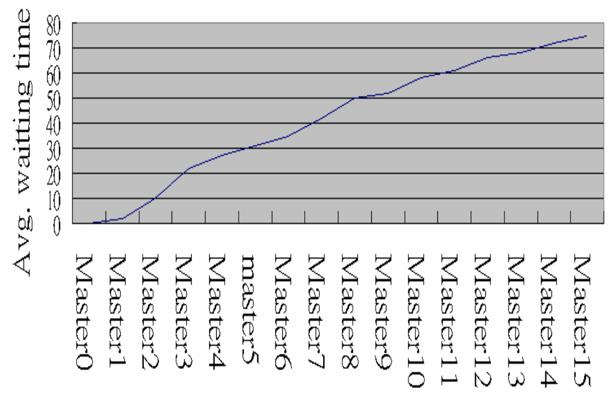


Fig. 6 Waiting time for arbiter configuration state as 22222

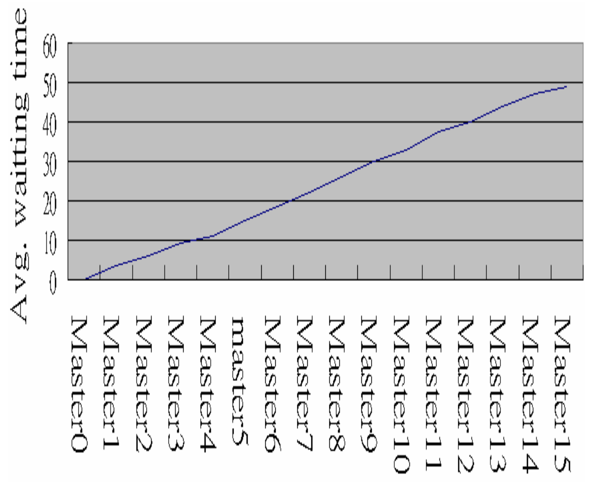


Fig 7 Waiting time for arbiter configuration state as 13133

The average waiting time for arbiter configuration as 13141, 23232, and 11111 is shown in Fig. 8. The longest waiting time is obtained when the arbiter configuration is 23232. The average waiting of 13141 can gain the shortest waiting time, which is better than the conventional fixed priority arbitration scheme.

In order to speed efficient transmission of large bursts of data, buses usually provide a block transfer mode for multiple bus cycles. Fig. 9 shows the effect of block size on performance. The worst case occurs when the arbitration configuration is set to be 43332 for F1~F5 block and the block size is chosen

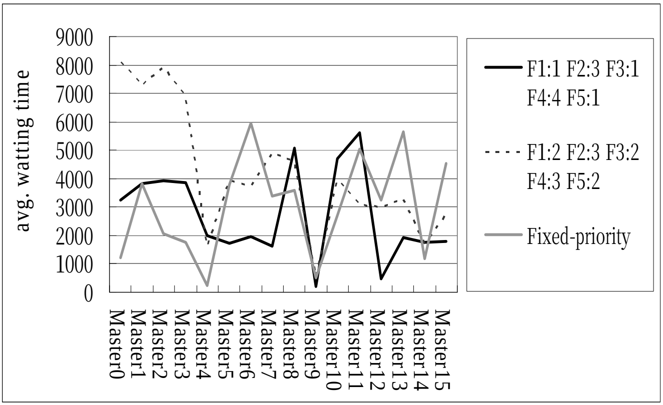


Fig. 8 Waiting time for various arbiter configuration states

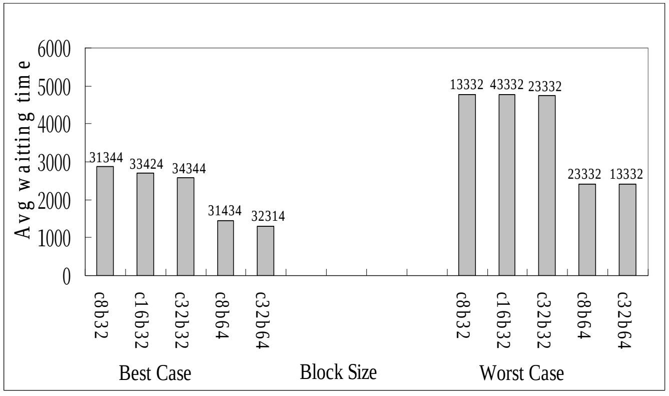


Fig. 9 Block size effect for various arbitration scheme

to be 16x32 bits/s. The block size c16b32 denotes in the x-axis of Fig. 9 indicating the data transmission rate is 32 bit/s, and the period is 16 clock cycles. The best performance is obtained by 32314 assignment state and the block size is 32x64 bits/s.

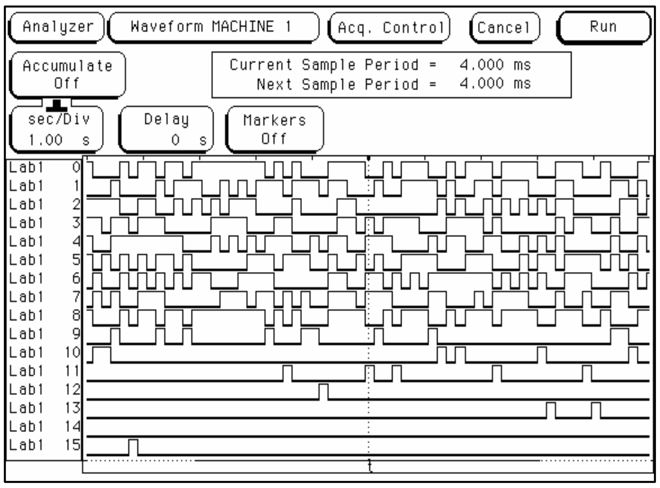
Simulation tests using normal distribution for data amount and exponential distribution for master request are conducted for 1000 times and results are average out. Based on 1024 permutations of arbitration algorithms for blocks F1 ~ F5 under the same traffic distribution pattern, Table 1 shows the simulation results of the grant rate and through put at various configurations. For the present case study, it indicates that the reconfigurable arbiter with 22221 configuration has the best bus utilization rate, 13431 configuration has the best average grant rate, 22221 configuration has the best throughput performance.

Table. 1 Simulation results of the performance analysis at various configuration schemes

Arbitration configure	Avg. Bus Uilization (%)	Avg. Grant Rate (%)	Avg. Throughput (bit/s)
12314	78.582281	21.847687	25.10701
12413	78.70257	12.832688	25.144301
13431	78.708633	22.378313	25.146475
13411	78.741391	21.03675	25.156859
34341	78.879539	21.743875	25.200553
22224	81.990609	13.027	26.195668
12214	83.080711	16.205313	26.543143
12411	84.082711	18.282812	26.862924
12424	84.190094	17.3065	26.897707
12421	84.772875	18.18225	27.083648
22221	85.31818	12.873	27.257682
11111	78.558523	19.754125	25.09885
22222	60.248195	10.371563	19.24841
33333	75.091805	15.752938	23.990609
44444	73.05175	19.795437	23.339273

5. Implementation Results

The reconfigurable arbiter was synthesized using Altera's Quartus II 4.2, which is an integrated environment for logic design and synthesis. After finishing the verilog simulation, we download the program into EPF10K100ARC240-1chip. An 8051 microcontroller was used to drive the test signal into the chip. The arbiter can be reconfigured many times by modifying the program code for the 8051 microcontroller, and, thus, can be changed and updated real-time. The Tektronix logic analyzer is connected to the FPGA pins to check the functionality of the reconfigurable arbiter. Fig. 10 shows the reconfigurable arbitration state (12341) measured by the logic analyzer.



```
Current Sample Period =   4.000 ms
   Next Sample Period =   4.000 ms
```

Fig. 10 Verification of 12341 configuration by logic analyzer

In this study, the RTL model of the reconfigurable arbiter is synthesized to gate-level using Synopsys Design Vision with TSMC 0.18 μm technology file. The synthesized verilog files are simulated in ModelSim. In order to provide a fast evaluation of the energy impact of various arbitration states, the switching activities are then used by Synopsys PrimePower for power calculation. The same RTL model is also used and mapped into Xilinx Vitrex and Altera Straitx GX development tools. Fig. 11 shows the synthesis and power calculation flow.

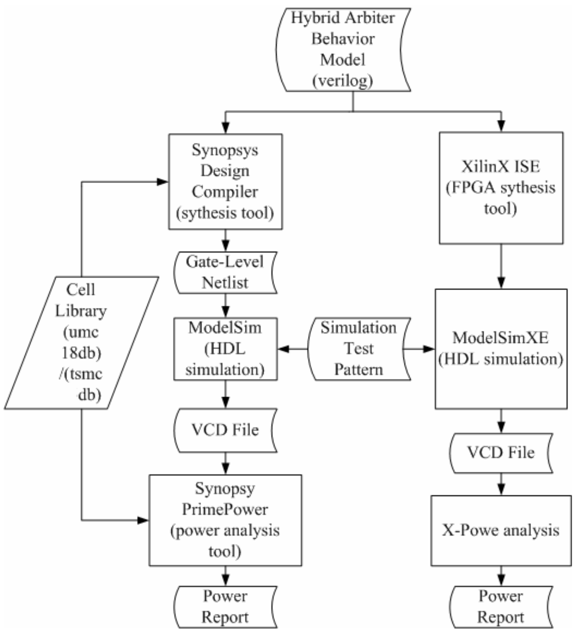

Hybrid Arbiter Behavior Model (verilog)
Synopsys Design Compiler (sythesis tool)
XilinX ISE (FPGA sythesis tool)
Gate-Level Netlist
Cell Library (umc 18db) /(tsmc db)
ModelSim (HDL simulation)
Simulation Test Pattern
ModelSimXE (HDL simulation)
VCD File
VCD File
Synopsy PrimePower (power analysis tool)
X-Powe analysis
Power Report
Power Report

Fig. 11 Synthesis and power calculation flow

Table 2 shows the FPGA implementation results of the reconfigurable arbiter for single layer share bus system. Table 3 lists the area and power performance summary of the design complier report based on TSMC 0.18 μm technology file.

Table 2 FPGA Implementation Results

Xilinx VirtexE XCV2000e-BG560			Atera StratixGX EP1SGX25CF672C5	
Delay (ns)	Slice Flip Flops	4 input LUTs	Delay (ns)	Logic elements
16.758	657	1153	9.286	1108

Table 3 Design Complier Report

Combinational area (μm^2)	28956.04883
Noncombinational area (μm^2)	36467.35547
Net Interconnect area (μm^2)	882564.1875
Total cell area (μm^2)	65423.63672
Total area (μm^2)	947987.8125
Cell Internal Power (mW)	3.9
Net Switching Power (mW)	2.9415
Total Dynamic Power (mW)	6.8415
Cell Leakage Power (μW)	2.0263
NAND2X1 : 5.04 µm (height) x 1.98 µm (width)	

The power performance of the reconfigurable arbiter under various configuration schemes for TSMC 0.18 μm technology is shown in Fig. 12 It can be seen the maximum total power is 59.92 μW for arbitration state (32332) in the shared bus system. The minimum power is 47.73 μW for arbitration state (11111) configuration.

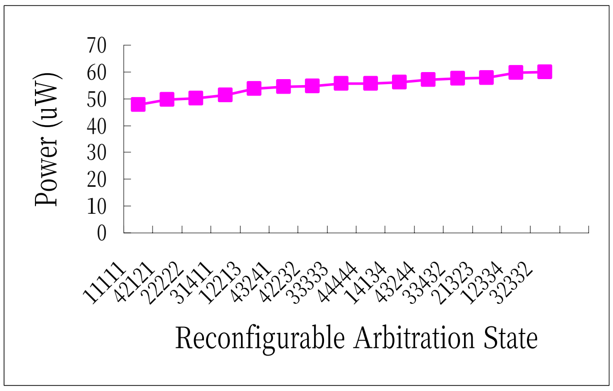


Fig. 12 Power performance at various reconfigurable arbitration states

6. Conclusion

An arbitration mechanism is usually employed to resolve the bus contention. The arbiter with hybrid arbitration algorithm for single layer shared bus system is presented in this study. The simulation results not only provide performance analysis for the various combinations of the arbitration algorithms. The gate-level power analysis is also applied to explore power dissipation in various reconfigurable arbiter architectures. These results can be feed into the reconfigurable arbiter to obtain the optimal condition under different system workloads. The reconfigurable arbiter can be custom-tuned to obtain high bandwidth utilization, low latency, and power effective for on-chip bus communication. The architecture-level power estimation that provides a fast evaluation of the energy impact of various optimizations early in the design cycle is essential. The results obtained show that the framework of the reconfigurable arbiter can be used to explore the space of possible configurations to evaluate the performance/power trade-off.

Acknowledgments

This research was partially supported by National Science Council, R. O. C., under Grant NSC95-2221-E-214-082.

References

[1] Wen-Shiu Liao and Pao-Ann Hsiung. FVP: A Formal verification Platform for SoC. In Ganesh Gopalakrishnan and Philip Windley, editors, the 16th IEEE International SoC Conference: Portland, Oregon, volume 1522 of Lecture Notes in Computer Science. Springer–Verlag, September 2003.

[2] Noguera, J. and Badia, R. M. “HW/SW codesign techniques for dynamically reconfigurable architectures”, *IEEE Trans. VLSI Syst.* Vol. 10, No. 4, 2002, pp. 399 – 415.

[3] Open Microprocessor systems Initiative, DRAFT STANDARD OMI 324: PI-Bus Rev. 0.3d, 1994.

[4] AMBA 2.0 Specification. [Online]. Available: http://www.arm.com/armtech/AMBA.

[5] D. Flynn. AMBA: Enabling Reusable On-Chip Design. IEEE Micro, July 1997, pp. 20–27.

[6] http://www.mentor.com

[7] http://www-3.ibm.com/chips/techlib/techlib.nsf/productfamilies/CoreConnect_Bus_Architecture

[8] “Sonics Integration Architecture, Sonics Inc.” Available: http://www.sonicsinc.com.

[9] W. Peterson. Design Philosophy of the Wishbone SoC Architecture. In Silicore Corporation, 1999. Available: http://www.silicore.net/wishbone.htm.

[10] Lahiri, K.; Raghunathan, A.; Lakshminarayana, G.; “ The LOTTERYBUS on-chip communication architecture”, IEEE Trans. on Very Large Scale Integration (VLSI) Systems, Vol. 14, No. 6, 2006, pp. 596 – 608.

[11] www.arm.com/pdfs/DVI0045B_multilayer_ahb_overview.pdf