The Implementation and Evaluation of a Low-Power Clock Distribution Network Based on EPIC

Rong Ji, Xianjun Zeng, Liang Chen, Junfeng Zhang

School of Computer Science,National University of Defense Technology
Changsha, Hunan, China, 410073
{rongji,xjzeng,liangchen,jfzhang}@nudt.edu.cn

Abstract. The multiply clock domain (MCD) technique is a novel technique to compromising between synchronous systems and asynchronous systems to reduce the power. However, most present studies of MCD are based on superscalar architectures. In this paper, MCDE, a MCD technique based on explicitly parallel instruction computing (EPIC) architecture is designed and implemented to reduce the power of clock distribution network. In addition, a series of experiments have been done to evaluate it. The result of the experiments show that, using a MCDE clock network microarchitecture with a fine-grained adaptive dynamic adjustment algorithm, can effectively decrease the microprocessor power by 40%, compared with the original EPIC clock network microarchitecture.

1. Introduction

The clock network can dissipate 20–50% of the total power on a chip [1]. The clock network power dissipation limits have emerged as a major constraint in the design of microprocessor. Thus the optimization of the clock network becomes one of the important issues of high-performance microprocessor design.

At present, most microprocessors are implemented by using fully synchronous clock distribution mode. The clock distribution network is designed very carefully to meet the constraints of clock skew. It contributes to the complexity of clock interconnection and the significant increase of microprocessor power. So the designers present asynchronous system which need no clock. But there are so many difficulties in design of the complete asynchronous signals. MCD based on GALS style [2], which is a compromise between asynchronous systems and synchronous systems, has been focused on. Most current researches on MCD are based on superscalars [3,4,5,6,7,8], and not been applied to EPIC architecture, because there are much difficulties in its implementation. For example, the uncertainty of asynchronous communication brings significant challenges for data dependence and instruction completion in order in EPIC.

This paper describes the implementation and design of MCDE clock distribution network, and experimental simulation to evaluate it. Section 2 presents the conventional EPIC architecture, and analyzes what limits power optimization of microprocessor. Section 3 describes the design and implementation of the MCDE clock distribution network. Section 4 details the simulation framework and experimental setup. The results of all the tests are given and analyzed in Section 5. Conclusions are in Section 6.

2. The Conventional EPIC Architecture

2.1. The Conventional EPIC with One Clock Domain

An EPIC architecture emphasizes co-operating compiler with hardware to exploit instruction-level parallelism (ILP). It combines the advantages of superscalar architectures and the advantages of very long instruction word (VLIW) architectures [9]. Itanium 2 [10] is a classical processor based on EPIC architecture. Thus the EPIC architecture of Itanium 2 micoprocessor is a comparable architecture to our design. Fig. 1(a) shows a conventional EPIC architecture with single clock domain. The front-end of the microprocessor obtains the addresses of fetching instructions, by means of Bpred predicting branch objects. The instructions are fetched from instruction Cache (I-Cache), and reserved into instruction buffers (I-Buffer). They are decoded by the dispatch logic (Dispatch), and sent to corresponding execution units. ALUs and FPUs can execute multiply independent integer instructions and floating-point instructions per cycle, respectively. The operands are respectively given by integer registers (I-Reg) and floating registers (F-Reg). The results are reserved in corresponding registers. Load/store operations will access Level 1 data Cache (D-Cache) and level 2 Cache (L2 Cache). There are abundance of functional units and Cache levels to provide adequate hardware supports for exploiting ILP.

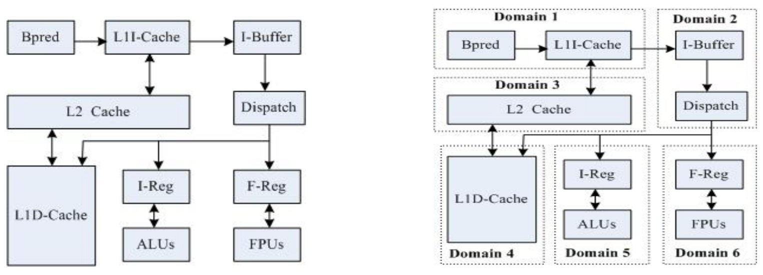


Fig. 1. (a) A conventional EPIC architecture **(b)** The clock domain partition of the EPIC

2.2 The Analysis What Limits Power Optimization of the Conventional EPIC

Currently the microprocessors based on the conventional EPIC architecture often use the fully synchronous clock strategy. The clock signal is generated by a clock generator, and is sent to each unit within the microprocessor. All units synchronously operate under control of this global clock signal. The clock distribution network is usually designed carefully to meet the constraints of clock skew. It contributes to the complexity of clock interconnection and the significant increase of microprocessor power. Thus, it is necessary for EPIC architecture to implement the MCD clock distribution network, in order to decrease the power of the microprocessor.

3 The Design and Implementation of MCDE Processors

A MCD architecture based on EPIC, which need no global clock, is called MCDE architecture. The microprocessor based on EPIC is divided into several clock domains. Each domain uses an independent clock. Among different domains, there is asynchronous communication, which can save the power of microprocessors.

3.1 The Partition of Clock Domains

In MCDE architecture, the entire chip can be divided into several domains according to the EPIC architecture characteristic of the microprocessor. There are two rules in the partition: 1) this partition can't change the organization structure of the microprocessor pipelines too much; 2) the domain boundaries are set between the components having a loose coupling with each other to the best of our abilities. The components having a close coupling with each other are placed in the same domain to reduce communication operations among different domains.

The EPIC microprocessor is partitioned into six clock domains according to the rules described above: a fetching instruction domain (Domain1), a dispatch domain (Domain2), and a L2 Cache domain (Domain3), a load/store domain (Domain4), an integer domain (Domain5), and a floating-point domain (Domain6). Fig.1(b) shows the partition detail. The function of Domain1 contains branch prediction, instruction address generation, and I-Cache read. Domain2 accomplishes dispatch of instructions. Domain3 includes L2 Cache read/write operation. L1D-Cache read/write is completed in Domain4. Domain5 completes the load/store operation of integer operands, and execution of arithmetic logic. Domain6 consists of the load/store operation of floating-point operands, and execution of floating-point computing. Domain1 and Domain2 are called front-end. The remaining domains are called back-end. Each domain has its own local clock. The units within same domain operate in synchronous mode. The asynchronous communication is used among different domains.

3.2 Communication Mechanisms among Domains

In the MCDE, the queue structures within the microprocessor is choosen as the synchronous points of the different domains. In this case, when the queues are not full and not empty, the latencies of synchronous overhead can be hidden. The design of the queue structures is based on the hybrid clock FIFO, which is shown as Fig. 2. The

FIFO queue has two ports: input and output. The function of input is to write data to the FIFO, which is controlled by *CLK_w*. *Req_w* is the control signal of write requirement. *Data_w* is the input data bus. When the FIFO is full, the signal *full* is set one. The function of the output of the FIFO is to read data from the FIFO, which is controlled by *CLK_r*. *Req_r* is the control signal of read requirement. *Data_r* is the output data bus. *Valid_r* is the valid bit of output data.

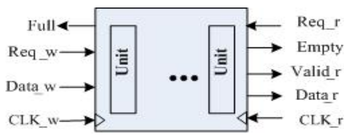


Fig. 2. The hybrid clock FIFO

The process of data stored at input is done as follows: When the rising edge of *CLK_w* arrives, *Req_w* that is the requirement signal and the data in *Data_w* are sampled. The data items will be written into the queue at the next clock period. If the *FIFO* is full, the signal *Full* is set before the next clock period arrives, and receiving the data at the input is prohibited. The process of reading data from the output is described as follows: When the rising edge of *CLK_r* arrives, *Req_r* that is the requirement signal is sampled. The data items will be put on *Data_r* bus, and the valid bit *Valid_r* is set. If the *FIFO* is empty at this clock period, the signal *Empty* is set, and reading operation at the output is prohibited until the *FIFO* is not empty.

3.3 The Design of the Register Scoreboard

A register scoreboard is used to resolve the problems of the uncertain latencies. In order to assure the compatibility with EPIC programs, it allows the compiler to schedule the instructions to deal with the certain latencies. Moreover, the hardware scoreboard, shown as Fig.3, is designed to handle dynamically the problems of the uncertain latencies. In Fig.3, r/w represents the read/write flag of the instruction; *r*D/*w*D denotes the domain number of read/write.

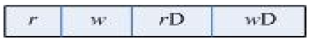


Fig. 3. The format of the register scoreboard

The scoreboard only determines the Read After Write (RAW) dependences and Write After Write (WAW) dependences among the instructions, which are dispatched to different issue groups in different clock domains. Because the EPIC compiler ensures having no RAW and WAW dependences within an issue group, the instructions, dispatched to the same clock domains, can be synchronously executed in order. For the Write After Read (WAR) dependences within an issue group, the scoreboard doesn't detect it at the time of the instructions being issued. Since the EPIC execution model of Non-Unit Assumed Latency (NUAL) semantic [11] allows the instructions to write-back out of order, which is premised on committing in order, the WAR dependences are detected when the instructions' results are written-back.

Assume that, for a instruction, S is a set of the source register numbers; D represents a set of the destination register numbers. E is the domain number to which the instruction will been dispatched to execute. For the register r, SB[*r*].*w*, SB[*r*].*w*D, SB[*r*].*r*, and SB[*r*].*r*D, respectively denote each field in Fig.4. When the instruction is dispatched, the operation of the register scoreboard is described as the Dispatch Algorithm. in Fig.4. When read the operands of the instructions, the operation of the scoreboard is designed as the Read Algorithm. in Fig.4. When write the result of the instructions, the operation of the scoreboard is described as the Write-back Algorithm in Fig.4.

The Dispatch Algorithm	The Read Algorithm	The Write-back Algorithm
if the resource queue is busy	**for each** $s \in$ S	**for each** $d \in$ D
return resource dependencies	SB[s].r = 0	**if** SB[d].r
for each $s \in$ S	SB[s].rD = 0	**return** WAR dependencies wait
if SB[s].w **and** (E $\neq$ SB[s].wD)		**else**
return RAW dependencies wait		SB[d].w = 0
for each $d \in$ D		SB[d].wD = 0
if SB[d].w **and** (E $\neq$ SB[d].wD)		
return WAW dependencies wait		
for each $s \in$ S		
SB[s].r = 1		
SB[s].rD = E		
for each $d \in$ D		
SB[d].w = 1		
SB[d].wD = E		

Fig. 4. The related algorithms of the register scoreboard

3.4 The Implementation of Instruction Commit in Order

According to the NUAL semantic [11], EPIC would guarantee the instructions issue and commit in order. However, the execution of instructions and the results write-back of instructions are out of order. Although the dispatch process is completed within a clock domain to assure the instructions issue in order, the execution process of the instructions is completed in several different clock domains, which can't guarantee the instruction completion in order. Thus we design an instruction order buffer (OB).The structure of the OB is shown as Fig.5.

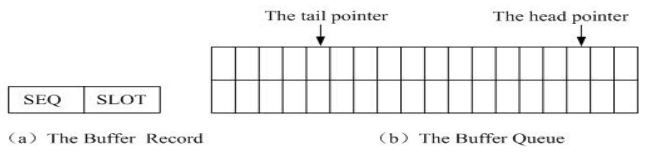


Fig. 5. The structure of the order buffer

The OB is a FIFO queue. However, it might to write or read multiple records. When the instructions are dispatched, the dispatch logic allocates two numbers to each instruction: SEQ, which is the instruction word number, and SLOT, which is the serial number of the instruction in this instruction word. The SEQ and the SLOT solely determine all instructions which are being processed by the processor, and mark the instruction rank required by the program. When the instructions are dispatched, the instructions which are dispatched within the same cycle will write SEQ and SLOT into the OB in order. And then, The SEQ and the SLOT enter the pipelines of the back-end clock domains, keeping company with the instruction. Once the instruction is completed, the OB is searched by means of the SEQ number and the SLOT number. If all instructions ahead of this instruction can be committed, the instruction could be committed. The mechanism designed above can guarantee sufficiently the instruction commit in order.

3.5 A Dynamic, Adaptive Control Algorithm of Clock Domain's Frequency

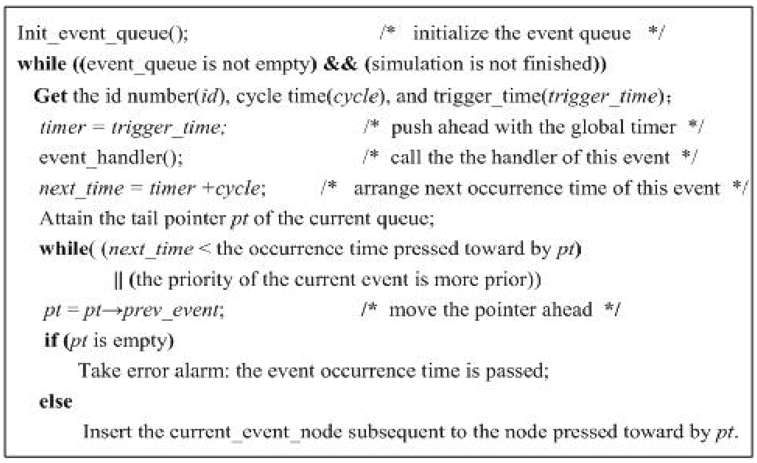
Queue utilization is an appropriate metric for dynamically determining the desired domain frequency. The dynamic, adaptive control algorithm of clock domain's frequency is based on this idea to reduce the power of the clock network.

The dynamic, adaptive control algorithm of clock domain's frequency is described as follows: In each back-end domain of the MCDE architecture, the attack/decay algorithm [6] is used independently. When the entries in the domain issue queue is in excess of 10,000-instructions interval, the hardware counts. Using the number and the corresponding number from the previous interval, the algorithm determines whether there is a significant change that threshold is 1.7 percent, in which case the algorithm uses the attack mode: The frequency changes by 7 percent. If no significant change occurs, the algorithm uses the decay mode: It reduces the domain frequency slightly by 0.17 percent.

But when the instructions per cycle (IPC) changes by more than a certain threshold that is 2.3 percent, the frequency remains unchanged for that interval. It is used to identify natural decreases in performance that are unrelated to the domain frequency, and to prohibit the algorithm from reacting to them.

4 The Simulation Framework and Experimental Setup

To evaluate our MCDE clock network architecture, we use basic technology parameter of CACTI 0.8μm [12], and scaling down method to implement power consumption evaluation model [13]. Since IMPACT [14] provides a comprehensive infrastructure for modeling and simulation of EPIC architecture feature, the Lsim simulator [14,15] of the IMPACT compile framework is adopted as the simulation engine. In addition, in order to simulate the MCDE processor, the event-driven algorithm based on multi-clock domains, described as Fig. 6, is designed.



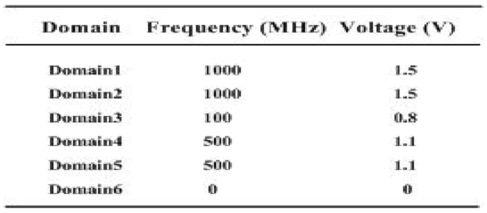
```
Init_event_queue();                          /*  initialize the event queue  */
while ((event_queue is not empty) && (simulation is not finished))
  Get the id number(id), cycle time(cycle), and trigger_time(trigger_time);
   timer = trigger_time;                   /*  push ahead with the global timer  */
   event_handler();                        /*  call the the handler of this event  */
   next_time = timer +cycle;         /*  arrange next occurrence time of this event  */
   Attain the tail pointer pt of the current queue;
   while( (next_time < the occurrence time pressed toward by pt)
            || (the priority of the current event is more prior))
    pt = pt→prev_event;                    /*  move the pointer ahead  */
    if (pt is empty)
        Take error alarm: the event occurrence time is passed;
   else
        Insert the current_event_node subsequent to the node pressed toward by pt.
```

Fig. 6. The event-driven simulation schedule algorithm

To evaluate objectively the MCDE, we design three groups of experiments:

- **SVF Strategy**: Each of clock domains works on condition that the voltage is same, and frequency is also same, called SVF for simple, which represents same voltage and frequency. That is to verify the performance and power consumption characteristic of MCDE relative to the EPIC.
- **DVF Strategy**: Each of clock domains works on condition that the voltages may be different, and the frequencies may be different, too. The frequencies and voltages are not adjusted during the system running. This strategy is called DVF for simple, which denotes different voltage and frequency. It can verify advantages of fine-grained setting of voltage and frequency in MCDE. The voltage and the frequency of each of domains are set as Table 1.

Table 1. Different voltage and frequency settings



Domain	Frequency (MHz)	Voltage (V)
Domain1	1000	1.5
Domain2	1000	1.5
Domain3	100	0.8
Domain4	500	1.1
Domain5	500	1.1
Domain6	0	0

- **DVF+DAA Strategy**: Each of clock domains works on condition that voltage and frequency maybe different with each other domain. They are also set as Table 1. Furthermore, during the system running, the frequency of each back-end domains is adjusted dynamically and adaptively as the control algorithm described above. This strategy is called DVF+DAA, which is an abbreviation for Different Voltage and Frequency + Dynamic Adaptive Adjustment Strategy. This group of experiment is used to verify the potentiality for fine-grained dynamic adaptive frequency adjustment decreasing the power of the microprocessor in MCDE.

5 The Analysis of Experimental Results

Fig.7 shows the power of MCDE processors relative to the basic EPIC processor. The experimental results reveal that all three MCDE strategies can decrease the microprocessor's power. But the power decrease of SVF is only 6 percent. After DVF used, the processor's power is decreased by 32 percent owing to further exploiting the advantage of MCDE. Comparing with SVF and DVF, DVF+DAA strategy can more effectively decrease the processor's power, which decreases the power by 40 percent.

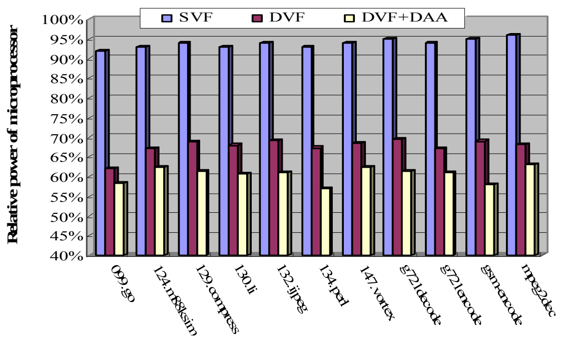


Fig. 7. The power consumption of MCDE relative to the EPIC

Fig.8 shows that all these three MCDE strategies can lead to performance degradation relative to EPIC, since they use the asynchronous communication mechanism. The SVF strategy results in a slight degradation, within 1 percent. Comparing with SVF, DVF and DVF+DAA result in more performance degradation owing to clock frequency being decreased. For DVF, the average performance degradation is approximately 6.5 percent. For DVF+DAA, it is about 7.3 percent.

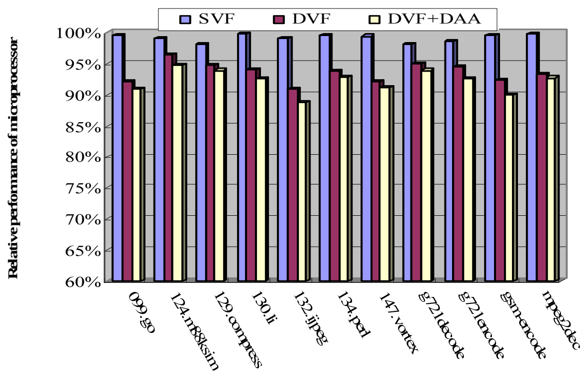


Fig. 8. The performance of MCDE relative to the EPIC

Energy delay product (EDP) is a popular metric to evaluate comprehensively the performance and the power consumption. Fig.9 shows the EDP of MCDE relative to

EPIC, corresponding to three MCDE strategies. The results of the experiment indicate that, for DVF+DAA, a significant overall EDP improvement is achieved, about 17 percent, owing to immensely exploit the potential for MCDE reducing the power consumption of the microprocessor. In addition, for DVF, the average EDP improvement is approximately 14 percent. Finally, the EDP improvement brought by SVF is not significant, only 7 percent, as a consequence of the conservative strategy. But comparing with EPIC, all three MCDE strategies are obvious to improve the EDP. Thus, there is a significant advantage of decreasing the power consumption by using MCDE, comparing with conventional EPIC.

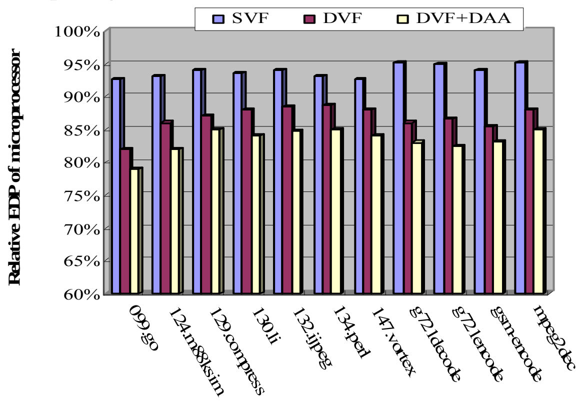


Fig. 9. The EDP of MCDE relative to the EPIC

6 Conclusions

The MCD is a novel technique that combines the advantages of synchronous systems and the advantages of asynchronous systems to resolve the problems of high-performance microprocessors, such as the significant power, and the complex clock distribution network. However, the current studies of MCD are almost based on superscalar. In this paper, a MCDE clock network architecture is implemented, and evaluated comprehensively. The experimental results show that, the MCDE clock network has a significant potential for reducing the power consumption of the high-performance EPIC microprocessor. Using the DVF+DAA MCDE strategy, the power of the EPIC microprocessor can be reduced by 40%, and about a 17 percent EDP improvement is achieved.

Acknowledgements

This work is sponsored by the National Science Foundation of China under the grant NO. 60676016.

References

1. Jatuchai Pangjun et al.: Low-Power Clock Distribution Using Multiple Voltages and Reduced Swings. IEEE Trans.on VLSI systems, Vol.10, No.3 (2002)309-318
2. Chapiro, D.M.: Globally Asynchronous Locally Synchronous Systems. PhD thesis, Stanford Univ (1984)
3. Iyer, A., Marculescu D.: Power and Performance Evaluation of Globally Asynchronous Locally Synchronous Processors. Proceedings of the 29th Annual International Symposium on Computer Architecture (2002)158-170
4. Magklis, G., et al.: Profile-based Dynamic Voltage and Frequency Scaling for a Multiple Clock Domain Microprocessor. Proceedings of the 30th Annual International Symposium on Computer Architecture (2003) 14-27
5. Semeraro, G., Magklis, G., et al.: Energy-Efficient Processor Design Using Multiple Clock Domains with Dynamic Voltage and Frequency Scaling. Proc. 9th Int'l Symp. High-Performance Computer Architecture (2002) 29-40
6. Semeraro, G., Albonesi, D.H., et al.: Dynamic Frequency and Voltage Control for a Multiple Clock Domain Microarchitecture. Proc. 35th Ann. IEEE/ACM Int'l Symp. Microarchitecture (2002) 356-370
7. Iyer A., Marculescu D.: Power efficiency of voltage scaling in multiple clock, multiple voltage cores. Proceedings of the International Conference on Computer-Aided Design (2002) 379-386
8. Eswaran, A., Chen, S.: All-domain fine grain dynamic speed/voltage scaling for GALS processors., URL http://www.ece.cmu.edu/~schen1/ece743/proposal_up.pdf (2003)
9. Schlansker, M., Rau, B.: EPIC: Explicitly Parallel Instruction Computing. IEEE Computer, Vol.33, No.2 (2000)37-45
10. Naffziger, S., Colon-Bonet, G., Fischer, T., et al.: The Implementation of the Itanium 2 Microprocessor. IEEE Journal of Solid-State Circuits, Vol.37, No.11 (2002)1448-1460
11. Schlansker, M., Rau, B.: EPIC: An Architecture for Instruction-Level Parallel Processors. Technical Report HPL-1999-111, HP Laboratories (2000)
12. Reinman G., Jouppi N.: An Integrated Cache Timing and Power Model. Technical report 2000/7, Western Research Laboratory, USA (2000)
13. Wang Yongwen, Zhang Minxuan: Microarchitecture-Level Power Modeling and Analyzing for High-Performance Microprocessors. Chinese Journal of Computers, Vol.27, No.10 (2004) 1320-1327
14. Wang Yongwen, Zhang Minxuan: IMPACT XP: An integrated performance / power analysis framework for compiler and architecture research. Chinese Journal of Electronics, Vol.13, No.2 (2004) 250-253
15. Chang, P., Mahlke, S., Chen, W., et al.: An Architectural Framework for Multiple-instruction-issue Processors. In Proceedings of the 18th Annual International Symposium on Computer Architecture(1991) 266-275