

# A Model of Implementable SMT Processor on FPGA

Ippei Tate, Yoshiyasu Ogasawara, Mikiko Sato, Koichi Sasada, Kaname Uchikura,  
Kazunari Asano, Satoshi Watanabe, Mitaro Namiki and Hironori Nakajo  
Tokyo University of Agriculture and Technology  
2-21-16 Naka-cho, Koganei-shi, Tokyo, Japan.

## Abstract

*Recent improvements of FPGA technology in its clock frequency, density and configuration cost enable to implement large-scale, complex circuit. In these days, many systems adopt FPGA so that the processor in FPGA will be required higher performance. This paper focuses on Simultaneous Multithreaded (SMT) architecture, one of multithreading technology that improves performance. We suggest implementable model of SMT processor on FPGA designing SMT processor and memory controller in hardware description language. As a result, a model that contains SMT processor, cache memory, memory controller is able to implement on single FPGA. Furthermore, we indicate efficient hardware use with internal Block RAM.*

*Keywords:* FPGA, SMT processor, multithreading, cache memory, DDR-SDRAM

## 1 Introduction

Reconfigurable devices such as FPGA have been improved its clock frequency, density and configuration cost. In these days, the significant improvement enables to implement large-scale, complex circuit on a FPGA device. There are a lot of commercial products that combining a processor and dedicated circuits to make up a System on a Chip (SoC), or soft-core processors have been released for FPGA-based SoC development by FPGA vendors [1] such as MicroBlaze from XILINX, Inc. [2] and Nios released by Altera, Co.[3]. Comparing to high-performance processor such as Pentium4 released from Intel, Co.[4], these soft-core processors are very small and lower performance. However, the increase of demand systems using reconfigurable device in the future will lead soft-core processors to improve.

To achieve the performance improvement, multithreaded architecture has got a lot of attention currently. The multithreaded architecture aims to exploit thread-level parallelism (TLP) rather than instruction-level parallelism (ILP). Simultaneous Multithreaded (SMT) processor, one of multithreaded processor, executes multiple threads in parallel allowing them to share hardware resources in the processor such as ALU, cache memory and so on. The availability of SMT processor has already shown in the document [5]; it suggests that SMT architecture achieves multithreading with lower resource comparing with a Chip Multiprocessor (CMP), another one of multithreaded architecture.

This paper focuses on SMT processor as high-efficient multithreaded processor. We suggest implementable model of SMT processor on FPGA designing SMT processor and memory controller in hardware description language. We in this paper show the model of stand-alone SMT processor and the model contains SMT processor, cache memory and DDR-SDRAM controller. Implementing these models on FPGA, we indicate implementing method that uses FPGA resource efficiently.

The paper is organized as follows. We first describe multithreaded processor in next section to explain why we chose SMT processor. Then section 3 describes architecture of the processor and cache memory. We show the two models that we designed and implemented results in section 4 and 5. Section 6 summarizes our study.

## 2 Multithreaded Processor

In this section, we describe four representative multithreaded processors; CMP, Fine-Grain Multiprocessor, Coarse-Grain Multiprocessor and SMT processor. Comparing these processors, we explain why we focus on SMT technology. Figure 1 shows concept of each multithreaded processors that tells ALUs utilization in every cycle in the case of all processors with 4 resources.

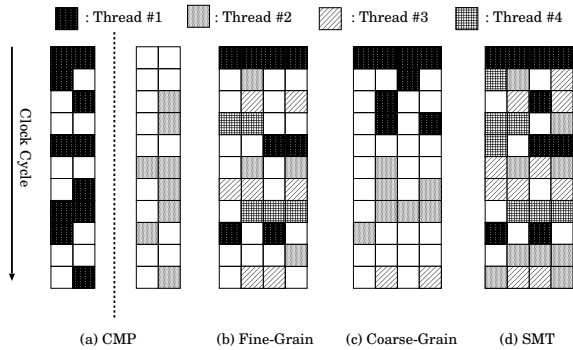


Figure 1: Resource usage of each multithreaded architecture

## 2.1 Chip Multiprocessor

A CMP has multiple processing elements (PEs) in a single chip to implement multithread processing. Each PEs has independent processing resources such as ALUs and register file. Figure 1 (a) shows an operation in the case of CMP with 2 PEs. ALUs are exclusively allocated in each PEs, and CMP cannot use both of PEs for a single thread. Furthermore, to increase number of executable threads in parallel, CMP has to add another PEs. Therefore each PEs needs some ingenuity to save its cost.

As the Commercial CMPs, POWER series [6] from IBM, Co., Ultra SPARC IV [7] from Sun Microsystems, Inc. and PA-8800 [8] from Hewlett-Packard Development Company are commonly known. In university study project, Hydra is famous [9].

## 2.2 Fine-Grain Multiprocessor

A Fine-Grain Multiprocessor switches executing thread by time-sharing. The thread switching occurs as frequently as once a few cycles or every cycle. Figure 1(b) shows Fine-Grain Multiprocessor with switching thread every cycle.

As examples of Fine-Grain Multiprocessor, we show document [10] and [11].

## 2.3 Coarse-Grain Multiprocessor

A Coarse-Grain Multiprocessor switches thread by not only time-sharing but also some trigger. Coarse-Grain Multiprocessor in figure 1 (c) adopts the facts that stall processor for a while as switching trigger. Both Fine-Grain method and Coarse-Grain method maintain thread context in their processors to switch thread in high speed which is different

with context switching of single-threaded processor.

Comparing between Fine-Grain method and Coarse-Grain method, Fine-Grain method compulsorily switches executing thread in static cycles so that performance of processor gains but thread-level performance narrows. On the other hand, Coarse-Grain method doesn't reduce main thread performance in the case that switching thread occurs when main thread stalls.

Course-Grain method is adopted on APRIL [12] of MIT and MAJC [13] from Sun Microsystems, Inc.

## 2.4 Simultaneous Multithreading

SMT processor [5] contains multiple program counters in a single chip to implement multithreading, but shares many resources such as ALUs and register file in multiple threads unlike CMP. SMT processor aims to use resources efficiently by the sharing. Figure 1 (d) shows SMT's resource usage in every cycle. SMT enables to increase number of executable threads in parallel with a smaller additional hardware than CMP does. SMT processor allows executing multiple threads simultaneously instead of switching executing thread in time-sharing as other methods; therefore SMT processor shows higher performance than Fine-Grain method's and Coarse-Grain method's [5].

There are two advantages of SMT processor: higher performance than Fine-Grain and Coarse-Grain methods and less hardware cost than CMP. Focusing on these advantage, our study adopts SMT processor as multithreaded processor on FPGA. In commercial product, Pentium4 HT [4] of Intel, Co. and POWER5 [6] from IBM, Co. adopts SMT processor.

## 3 SMT Processor and Memory Architecture

This section describes designed SMT processor and memory architecture.

### 3.1 SMT Architecture

Our original SMT processor which is implementable on FPGA is named SEMP (Simply Efficient Multithreaded Processor). SEMP complies with OChiMuS PE Instruction Set Architecture[14][15] which is advanced its development and study by us.

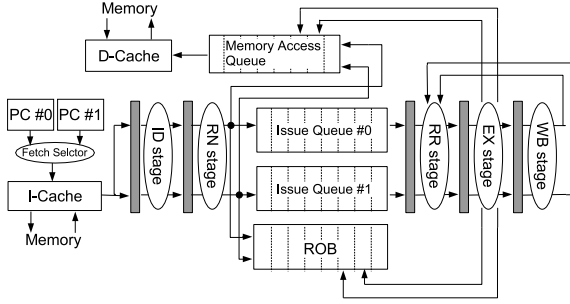


Figure 2: Structure of Simply Efficient Multi-threaded Processor

Table 1: Hardware resource of SEMP

Parameter	Value
Hardware Contexts	2
Issue Width	2
Execution Units	SimpleALU×2, ComplexALU×1, Load/Store Unit×1
Number of Physical Registers	92
ROB	24
Issue Queue	8
Memory Access Queue	8
Branch Predictor	PHT512entries, GHR2bit
Data Width	32bit

OChiMuS PE ISA, which is based on MIPS - II ISA, has thread control instructions to support multithreading. Using SEMP as a PE, we consider to develop multiple SMT PE on a single chip, named OChiMuS architecture in the future(See literature [14][15]).

Designed SMT processor, SEMP is shown in figure 2. Gray boxes in the figure are pipeline registers. There are two kinds of hardware resources in SMT processor: appropriate resources for each thread and shared resources among multiple threads. In the case of SEMP, program counter, reorder buffer, issue queue and branch prediction treat as appropriate resources, and rest of hardware resources, ALUs, register file and cache memory, are shared resources.

This processor mainly composes with those resources in table 1. SEMP is 2-way SMT processor based on 2-instruction-issued, out-of-order superscalar processor. As shown in figure 3, its number of pipeline has up to 10 stages.

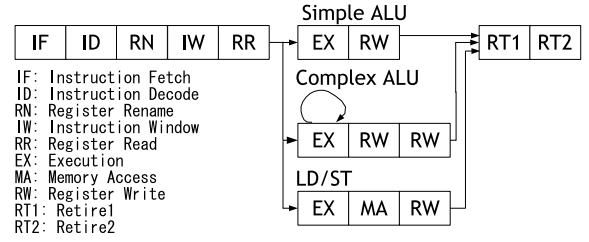


Figure 3: Pipeline stage of SEMP

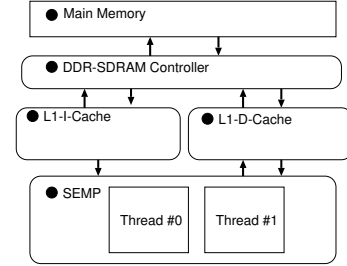


Figure 4: The overview of SEMP and cache memory

### 3.2 Memory Architecture

Our designed configuration of SEMP and cache memory is shown in table figure 4. SMT architecture shares cache memory among threads to efficiently use common data area.

Generally cache memory consists of Level-1 (L1) cache memory and Level-2 (L2) cache memory, but in some case, a chip doesn't have enough hardware resource to implement large size of storage media such as L2 cache memory. Further high integration in future will solve this problem, but at this time, as the FPGA device which is used in this study doesn't have enough hardware to implement L2 cache memory, we only implement L1 cache memory for cache memory.

We show the specification of designed cache memory in table 2. This model divides cache memory into instruction cache memory (I cache) and data cache memory (D cache). I cache adopts direct mapping, and D cache adopts 2-way set associative. Both of caches are 8 KB and cache block size are 16 Bytes. If cache hit occurs the processor gets data in a single cycle, otherwise memory spends up to 17 cycles to respond. When cache miss occurs, cache memory address to DDR-SDRAM controller. The cache-miss latency includes time lag generated by access to DDR-SDRAM.

We develop the SMT processor, the cache mem-

Table 2: Specification of designed cache memory

Parameter		Value
Capacity	L1-I-Cache	8KB
	L1-D-Cache	8KB
Way	L1-I-Cache	1
	L1-D-Cache	2
Block Size	L1-I-Cache	16B
	L1-D-Cache	16B
Latency	Hit	1 cycle
	Miss	17 cycle

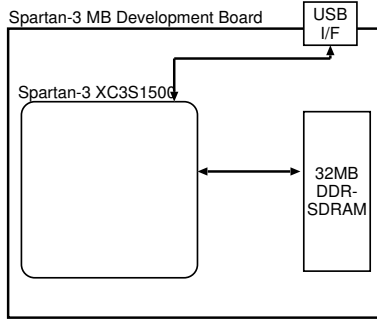


Figure 5: Target FPGA board, Spartan-3 MB Development Board

ory and DDR-SDRAM controller have just been explained above in Verilog-2000.

## 4 Implementable models of SMT processor on FPGA

First in this section, we describe FPGA board that is used in this study. After that we explain a model implements stand-alone SMT processor and another model contains SMT processor, cache memory and DDR-SDRAM controller.

### 4.1 Target FPGA board

In this study, we use Spartan-3 MB Development Board [17] from Memec, Inc. It adopts Spartan-3 XC3S1500 [16] of XILINX, Inc. for FPGA device. This FPGA has 13312 slices of hardware and 32 Block RAMs.

Figure 5 shows schematic diagram of FPGA board. This board is loaded with 32-MB DDR-SDRAM. The model implements cache memory and memory controller uses the DDR-SDRAM as main memory. And also, the development board has USB port for an external I/O. To load program into DDR-SDRAM, we connect a computer

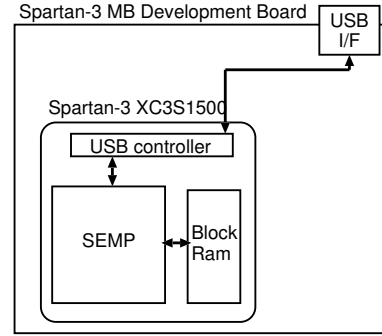


Figure 6: The model implements stand-alone SMT processor

to FPGA board via USB. There are other external I/Os, but we don't use rest of them in this paper..

### 4.2 The Model Implements Stand-Alone SMT processor

In Figure 6, we show the model implements stand-alone SMT processor.

In this case, there are two ways to form main memory; one is generating main memory with distributed RAM spending hardware cost, and the another one is generating main memory with Block RAM which is internal SRAM of FPGA. However, Using distributed RAM, main memory spends too much hardware slice to implement SMT processor, and SMT processor also spends a large size of hardware. That makes difficult to make logic synthesis, place and route more efficient, and wiring-delay would occur. On the other hand, using Block RAM, hardware slice isn't consumed by memory because XILINX's FPGA prepares Block RAM area separate from hardware slice. Therefore, using Block RAM as main memory, we achieve efficient use of hardware resources and keep high clock frequency.

However, Using Block RAM as main memory also has a problem. Block RAM is too small to store a large-size program because it is internal SRAM mounted on FPGA. Multithreaded program requires a larger size of memory area than single-threaded program does. In this model, SEMP only can execute a small size of program as just checking its operation.

For loading program, this model uses USB. On implementing SMT processor and USB controller on FPGA, a part of Block RAM is initialized to get program from USB interface such as BIOS loading. After loading done, the loaded program is now executed.

Table 3: Hardware costs for main stages of SEMP

Module	Slice (single)	Slice (SMT)	Slice (SMT-single)	Ratio (SMT/single)
IF_state	100	288	188	2.88
ROB	557	1128	571	2.03
EX_state	888	1112	224	1.25
MA_sate	1027	1115	88	1.09
Instrucation_Window	806	864	58	1.07
ID/RN_state	1273	2082	809	1.64
RR_state	1138	1155	17	1.01
<b>Total</b>	<b>6370</b>	<b>8366</b>	<b>1996</b>	<b>1.31</b>

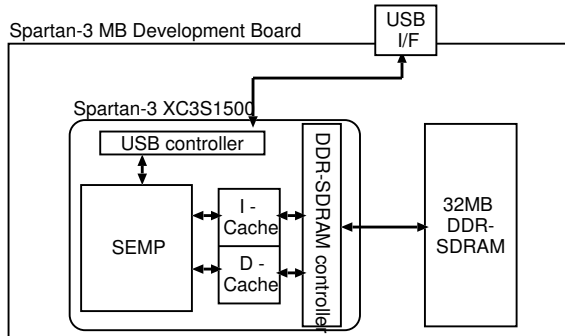


Figure 7: The model implements SMT processor with cache memory and DDR-SDRAM controller

### 4.3 The Model Implements SMT processor with Cache Memory and DDR-SDRAM

This model forms SMT processor with cache memory and DDR-SDRAM controller on FPGA as shown in figure 7. In this model, Block RAM is used for cache memory. We use DDR-SDRAM as main memory. Our FPGA board has 32-MB DDR-SDRAM; it is enough large to store multithreaded program. DDR-SDRAM controller receive requests from cache memory to access DDR-SDRAM. If I cache and D cache send requests in same time, the controller gives priority to I cache request and stalls D cache until I cache operation has done.

This model expands the range of use by replacing Block RAM to DDR-SDRAM for main memory. To initialize main memory, we use USB interface same as stand-alone SMT processor model.

## 5 Implementation Results

In this section, we evaluate designed models in hardware costs and clock frequency with XILINX

Table 4: Hardware cost of stand-alone SMT processor

Module	Silce
<b>SEMP</b>	<b>8366</b>
Memory Module	44
USB controller	133
<b>Total</b>	<b>8543</b>
<b>usage rate</b>	<b>64.18%</b>

ISE6.2.03i and Core Generator 6.2.03i. Table 3 compares our SMT processor, SEMP with single-thread processor in hardware cost. SEMP has large size of ID/RN stage and reorder buffer because these stages contain appropriate resources for each threads e.g. branch prediction unit and reorder-buffering queue. SEMP has a 1.3-fold larger scale than a single-threaded processor. For main memory, stand-alone SEMP utilizes 16 Block RAMs.

### 5.1 Result of stand-alone SMT processor

Table 4 reports the hardware cost for each elements used in stand-alone SMT processor. This model requires 8543 slices to implement. This number is equivalent with 64.18% of the FPGA device, xc3s1500. And the maximum operating frequency of stand-alone SEMP is 79.662MHz. However, this model has only 16-KB memory using 8 Block RAMs which is not enough size for operating multithreaded software. This configuration enables to examine how SMT processor works but not to operate large-size software. Usually, multithreaded software tends to be large size. We need larger size of main memory.

Table 5: Hardware cost of SMT processor with cache memory and DDR-SDRAM controller

Module	Silce
SEMP	8366
I - Cache	810
D - Cache	1533
DDR controller	350
USB controller	133
Total	11192
usage rate	84.07%

## 5.2 Result of SMT processor with cache memory and memory controller

The hardware cost of SEMP combined with cache memory and DDR-SDRAM controller is shown in table 5. Comparing with stand-alone model, this model obtains I cache and D cache as cache memory, and DDR-SDRAM controller to adopt DDR-SDRAM to main memory. Released Block RAMs by adoption of DDR-SDRAM are now used in cache memory; 4 Block RAMs for I cache and 8 for D cache. Although I cache obtains 8-KB storage area from 4 Block RAMs, D cache extracts 8-KB area from 8 RAMs because tag field for cache is implemented with distributed RAMs which consume hardware slice. In the case of 16-KB use for D cache, the increase in hardware usage defies implementing the model in the FPGA.

This model spends 11192 slices which are equivalent with 84.07

## 6 Conclusion

This paper dealt with SMT processor as high-efficient multithreaded processor. We proposed the model of SMT processor in the case of implementation into FPGA, and designed in hardware description language. As the result of implementation, both the model of stand-alone SMT processor and the model of SMT processor with cache memory and DDR-SDRAM controller are implementable, and we propose high-efficient use of FPGA structure utilizing Block RAM of XILINX FPGA.

We believe that SMT processor in FPGA will be active in multithreaded processor IP or embedded device core. For the future, we extend function of SEMP by adding floating point arithmetic unit, interrupt capabilities and so on.

## References

- [1] Xilinx Press Release #0412 : [http://www.xilinx.com/prs\\_rls/design\\_win/0412\\_marsrover.htm](http://www.xilinx.com/prs_rls/design_win/0412_marsrover.htm)
- [2] MicroBlaze : [http://www.xilinx.com/ipcenter/catalog/logicore/docs/microblaze\\_risc\\_32bit\\_proc\\_final.pdf](http://www.xilinx.com/ipcenter/catalog/logicore/docs/microblaze_risc_32bit_proc_final.pdf)
- [3] Nios 3.0 CPU : [http://www.altera.co.jp/literature/ds/ds\\_nios\\_cpu.pdf](http://www.altera.co.jp/literature/ds/ds_nios_cpu.pdf)
- [4] D. Marr, F. Binns, D. Hill, G. Hinton, D. Kofaty, J. Miller, M. Upton : "Hyper-Threading Technology Architecture and Microarchitecture" , *Intel Technology Journal*, Vol.6, pp.4-15, 2002.
- [5] D. Tullsen, S. Eggers, H. Levy : "Simultaneous Multithreading:Maximizing On-Chip Parallelism" , *Proceedings of 22rd Annual International Symposium on Computer Architecture*, pp.392-403, 1995.
- [6] R. Kalla, B. Sinharoy, J. Tendler : POWER5:IBM's Next Generation POWER Microprocessor, *HOTChips-15* (2003).
- [7] Sun Microsystems : Ultra SPARC IV Processor Architecture Overview (2004), [http://www.sun.com/processors/whitepapers/us4\\_whitepaper.pdf](http://www.sun.com/processors/whitepapers/us4_whitepaper.pdf)
- [8] Johnson, D. C. J. : HP's Mako Processor, *Microprocessor Forum* (2001), [http://ftp.parisc-linux.org/docs/whitepapers/mako\\_mpf\\_2001.pdf](http://ftp.parisc-linux.org/docs/whitepapers/mako_mpf_2001.pdf)
- [9] Hammond L., Hubbert B.A., Siu M., Prabhu M.K, Chen M. and Olukotun K. : The Stanford Hydra CMP, *IEEE MICRO* , Vol.20 , Issue 2, pp.71-84 (2000).
- [10] Robert A., David C., Daniel C., Brian K., Allan P. and Burton S.: The Tera Computer System, *Proceedings of the 1990 International Conference on Supercomputing (ICS1990)*, pp.1-6 (1990).
- [11] Farrens M. K. and Pleszkun A. R.: Strategies for Achieving Improved Processor Throughput, *Proceedings of the 18th Annual International Symposium on Computer Architecture (ISCA)*, pp.362-369 (1991).

- [12] Agarwal A., Lim B.-H., Kranz D. and Kubiawicz J.: APRIL: A Processor Architecture for Multiprocessing, *17th Annual International Symposium on Computer Architecture (ISCA)*, pp. 104-114 (1990).
- [13] Tremblay M., Chan J., Chaudhry S., Conigliaro A. W. and Tse S. S.: The MAJC Architecture: A Synthesis of Parallelism and Scalability, *IEEE Micro*, Vol.20, pp.12-25 (2000).
- [14] Y. Ogasawara, N. Kato, M. Yamato, M. Sato, K. Sasada, K. Uchikura, M. Namiki and H. Nakajo : "A New Model of Reconfigurable Cache for an SMT Processor and its FPGA Implementation", *Proceedings of the 2005 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA '05)*, Vol.II, pp.447-453, 2005.
- [15] K. Sasada, M. Sato, S. Kawahara, N. Kato, M. Yamato, H. Nakajo, M. Namiki : "Implementation and Evaluation of a Thread Library for Multithreaded Architecture" , *PDPTA '03*, pp609-615, 2003.
- [16] Spartan-3 Data Sheet (2004) :  
<http://www.xilinx.com/bvdocs/publications/ds099.pdf>.
- [17] Memec Spartan-3 FPGA-board :  
[http://www.memec.co.jp/html/xilinx/board/docs/sp3/Memec\\_3SMB\\_Users\\_Guide.v2\\_0.pdf](http://www.memec.co.jp/html/xilinx/board/docs/sp3/Memec_3SMB_Users_Guide.v2_0.pdf).