Design of SPARC V8 ISS Based on SystemC

Xue Yang, Yunkai Feng, and Lixin Yu

***Abstract*—Nowadays System on Chip (SoC) coming into use everywhere, SoC simulators are developing rapidly. Instruction Set Simulator (ISS) is one of the most popularly used tools for SoC exploitation. SoC designers can use it to test and verify their new designed processors, design and verify compilers, assistant debugging systems, and evaluate operating systems. A SPARC V8 ISS model base on SystemC is established in this paper, and it has been used to exploit application program, replant BenchMark and test and verify Golden Model. Test results proved the validity of the model function. In addition, the model has flexible structure with variety functions and strong practicability, so it can make great effect when design processors.**

***Index Terms*—ISS, SPARC, function model, SystemC.**

I. Introduction

Instruction set simulator is a model that can be used to simulate real processors. The model simulate the execution effect of each instruction running on the object processor, thus running of object program can be simulated, and it plays an important part in design, verify and application of processors. ISS can be divided into two kinds: explained ISS and compiled ISS. The explained ISS, for example SimpleScalar [1], explain and implement every instruction in object execution file on native computer, complete the process of fetch, decode and execute of every instruction, so as to simulate the function of object processor. It has the characteristics of easy to be designed and implemented, ease in replant and go to debugging mechanism. But the disadvantage is that the running speed is slow. When as, the compiled ISS, such as Static Compiled ISS [2] and Embra [3], translate the object binary system code into native machine binary system code. Running speed of the compiled ISS is 1 order of magnitudes faster than that of the explained ISS. On the contrary, the compiled ISS is hard to replant or debug. Thereby, the explained ISS is easier to be used as debugging software and testing tool for the exploitation of SoC. The model gives in this paper is an explained ISS.

Advantages of the ISS are listed as follows:

A. Cheapness Substitute of Real Hardware

Compare to the real hardware of the object machine, ISS is easier to get since it is a cheap software, so it is widely used as a cheapness substitute of expensive real hardware.

B. Sustain Hardware and Software Cooperated Development

The hardware and software cooperated development stress the feedback and parallel between hardware and software. Abuses of develop hardware and software separately have been overcame. Restrictions between hardware and software are corresponded so that the efficiency of the system is improved.

Manuscript received October 10, 2014; revised May 18, 2015.

Xue Yang , Yunkai Feng, and Lixin Yu are with Beijing Microelectronics Technology Institute, Beijing, China (e-mail: 15911106595@163.com).

C. Hardware Design and System Design Platform

Designers can use ISS to comprehend the dynamic behavior while the system is working. The dynamic information is quite important and useful for hardware architecture designers, instruction designers and system software developers.

D. Exploit and Debug Tools for Application Software Designers

Application software developers can exploit new software before the hardware implement, so the manufacture cycle can be shortened.

II. SystemC

SystemC is a language invented for system design. It is a hardware modeling platform presented by Open SystemC Initiative（OSCI), and is an expansion of C++. SystemC is the industry verify standard in system design at present, and is sustained by legion of world final EDA manufacturers, IP providers, semiconductor manufacturers and design companies. Compare to ordinary C++, the major augments in SystemC are a C class library that can describe and simulate hardware in different abstract levels and a light simulation core that independent of all hardware simulators. Presently, the extended basic hardware class in SystemC majors on the circuit characteristics such as module structure, time series, parallel behavior, response behavior and so on, as well as correlate characteristics of simulation monitor. It can be used to build hardware model and do simulation in gate level, RTL level, system level etc, and they sustain hardware and software cooperated development, can describe the complex system structure consist of both hardware and software. What's more, it can also depict the ports of hardware and software in C++. Thanks to all these features, it escapes designers from the complexity of design system level chips with plenty of languages, and reduces the design period [4].

III. SPARC V8 Processor Architecture

SPARC (Scalable Processor ARChitecture) is a system structure standard brought forward by SUN company in 1985. It is now one of the most popular worldwide microprocessor architecture. SPARC is an open architecture, so all organizations and individuals can exploit products based on SPARC architecture. Companies such as TOSHIBA, Fujitsu, Aeroflex, ESA and so on all have their own SPARC processors.

doi: 10.18178/ijiee.2016.6.1.596

SPARC microprocessors have a reduced instruction set and sustain data precision of 32/64 bits. The structure runs stable and is easy to expand. Besides, register window is a prominent characteristic of SPARC microprocessors. The technic can observably decrease the spending caused by process transfer, accordingly achieve high effect compile.

SPARC is an instruction set architecture (ISA) with 32-bit integer and 32-, 64-, and 128-bit IEEE Standard 754 floating-point as its principal data types. It defines general-purpose integer, floating-point, and special state/status registers and 72 basic instruction operations, all encoded in 32-bit wide instruction formats [5], as shown in Fig. 1.

Format 1（op=1）：CALL

op	disp30
31 29	0

Format2（op=0）：SETHI & Branches（Bicc，FBfcc，CBccc）

op	rd	op2	imm22
op	a cond	op2	disp22
31 29 28 24	21		0

Format3（op=2 or 3）：Remaining instructions

op	rd	op3	rs1	i=0	asi	rs2
op	rd	op3	rs1	i=1	simm13	
op	rd	op3	rs1	opf		rs2
31 29	24	18	13 12		4	0

Fig. 1. Summary of instruction formats.

Fig. 2 shows a common structure of SPARC V8. It has separate instruction and data caches, an interrupt controller, debug support unit with trace buffer, timers, UARTs, power-down function, watchdog, I/O port, flexible memory controller, ethernet MAC and PCI interfaces [6].

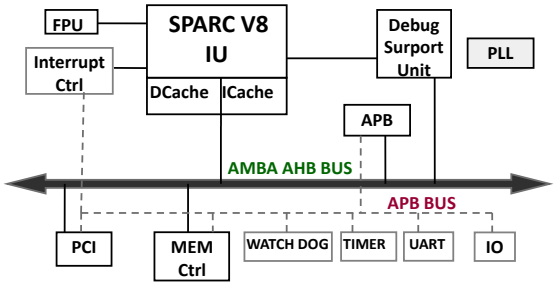

Fig. 2. Common structure of SPARC V8 processor.

IU is considered the most important unit in processors. IU is usually divided into numbers of pipeline stages. Fig. 3 gives the unit block diagram of an ordinary IU. Architecturally, an instruction is read from memory at the address given by the program counter (PC). It is then executed or not, depending on whether the previous instruction was an annulling branch (see below). An instruction may also generate a trap due to the detection of an exceptional condition, caused by the instruction itself (precise trap), a previous instruction (deferred trap), an external interrupt (interrupting trap), or an external reset request. If an instruction is executed, it may change program-visible processor and/or memory state.

IV. Realization of ISS

The ISS model of SPARC V8 processor is mainly consists of register form, memory form, instruction repertoire, and ELF file loading etc. Structure of the ISS is shown in Fig. 4.

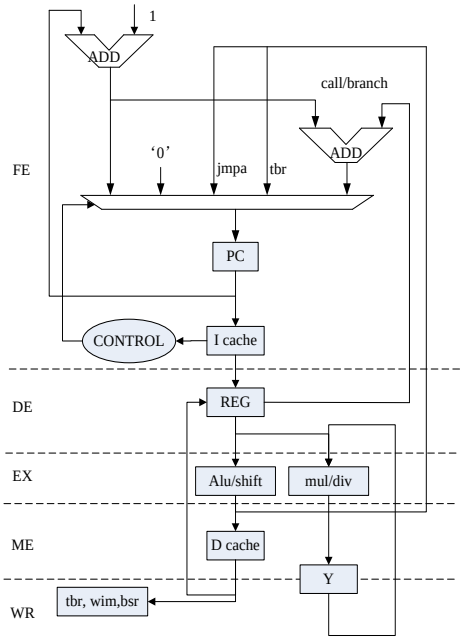

Fig. 3. LEON2 integer unit block diagram.

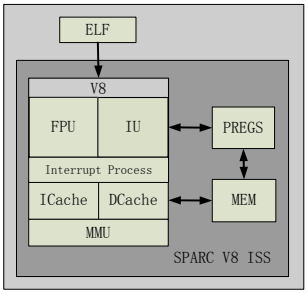

Fig. 4. Structure of SPARC V8 ISS.

1) The register form involves numbers, structures and using method of all registers such as general use registers, state/control registers and so on.
2) The memory form for instance kinds of controller registers, MEM and the like, are realized in the model. They are separated into different address part, which is ascertains by the router when accessing. Read-write functions of one-word, half-word and one-byte are implemented in MEM, and can be handle conveniently.
3) The instruction repertoire includes instruction functions, operation type, instruction code and so on. According to the define of instructions in SPARC V8 manual, a set of functions are compile to build the instruction repertoire model of the processor. One function corresponds to one instruction.
4) Load ELF file: the model load ELF file by compile load function. The files are load into MEM at the time the program is initialize.

Fig. 5 gives the progress how an instruction goes through ISS:

First of all, the ELF file is loaded into memory by the load operation. Secondly, memory gives out the instructions one by one. The instruction decoding process goes with the instruction classify in SPARC V8 manual. Decoded values are sent to the corresponding function to complete the given work. Results can be sent to registers or give out. Then

interrupt is dealt with. At last, PC is updated, and ISS is ready for the next instruction.

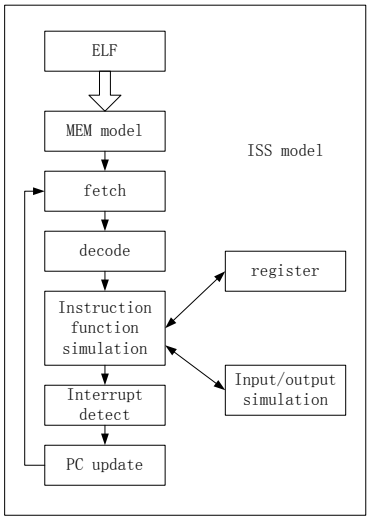

Fig. 5. Flow chart of SPARC V8 ISS.

V. Experimental Verification and Application

Programs been tested in this model include: all instruction tests, IEEE754_TEST, UCB_TEST, Mibench-basicmath, FFT, dhrystone, whetstone, coremark. The exploitation environment is Microsoft Visual Studio 2005 and the host computer configure is Intel i3-2100 3.1GHz, 2G EMS memory. The tool for intercross compile is sparc-elf-gcc.

Test results are listed below:

1) Overpass all Instruction tests and Functions of the ISS is Validated.

```
---- instruction test start --------
- add test ...          ok
- sub test ...          ok
 - mul test ...          ok
- div test ...          ok
- logical test ...       ok
- shift test ...        ok
- sethi test ...        ok
- ldst test ...         ok
- ticc test ...         ok
- save/restore test ... ok
- unimp test ...          ok
- wr/rd test ...      ok
- rett test ...         ok
- fbfcc test ...         ok
- Bicc test ...          ok
- FPop test ...          ok
---- instruction test completed --**
```

2) ISS can also list the instruction names, source registers, destination registers ect, which can be used to analysis the serial instructions.

```
or_reg      rs1=0    rs2=0    rd=16
sethi       rs1=0    rs2=0    rd=20
jmpl_imm      rs1=20   rs2=0    rd=0
nop         rs1=0    rs2=0    rd=0
wry_reg     rs1=0    rs2=0    rd=16
wry_reg     rs1=0    rs2=0    rd=17
nop         rs1=0    rs2=0    rd=0
```

3) ISS can output much information of the test program, such as numbers of instructions executed, simulation time, simulation speed, catch hit rate and so on.

```
SPARCV8 : ---------- Simulation
Started ----------
Dhrystone Benchmark
begin time: 0x000001fb
end time  : 0x000001e5

display user  time is:0x00000016
End_Time2 is:0x00000018
Test true time is:                0x00000000
Microseconds for one run through
Dhrystone: 0x3a102de0
Dhrystones per Second:
0x4d2d6534
MIPS = 0x47ca1d16
SPARCV8 : ---------- Simulation
Finished ----------
SystemC: simulation stopped by user.
SPARCV8 Simulation statistics:
    Number of instructions executed: 898845
    Simulation time : 0min 5sec
    Simulation speed: 179K instr/s

    Cache statistics:
    icache hit      : 897749
    icache miss     : 1096
    replace method  : RANDOM
    burst or not    : 1
    icache hit rate : 99.8781%
```

Works as Golden Model in UVM validate environment. Since the ISS has a flexible structure easy to be modified, it can evaluate how much the performance of the new structure has been promoted in a short design period. Presently, the 5-stage-pipelined model, dual-issue model and 5-stage-pipelined 4-issue model have been implemented (see Fig. 6).

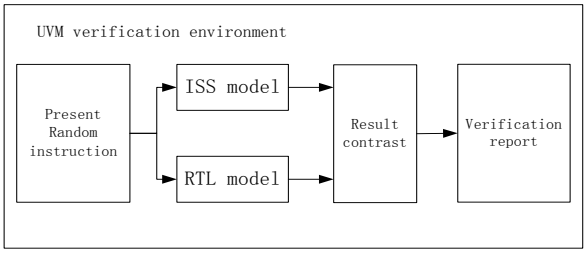

Fig. 6. UVM verification frame with ISS as Golden Model.

VI. Conclusion

A simulator based on SPARC V8 instruction is implemented in this paper, and plenty of tests validate and application is done. This model has clear hierarchy and well expansibility, reduce the difficulty of adding peripheral, and provide advantages for the investigation of SPARC architecture and its application.

References

[1] D. Burger and T. M. Austin, "The simple scalar tool set," version 2.0, *ACM SIGARCH Computer Architecture News*, vol. 25, no. 3, pp. 13-25, 1997.

[2] J. Zhu and D. Gajski, "A retargetable, ultra-fast, instruction set simulation," in *Proc. Design Automation and Test in Europe (DATE)*, 1999, Article no. 62.

[3] Emmett Witchel and Mendel Rosenblum, "Embra, fast and flexible machine simulation," *ACM SIGMETRICS Performance Evaluation Review*, vol. 24, no. 1, pp. 68-79, 1996.
[4] J. Bhasker, *A SystemC Primer*, Second Edition, copyright©2005 Star Galaxy Publishing.
[5] SPARC International Inc., "The SPARC architecture manual-version 8," 535 Middlefield Road, Suite 210 · Menlo Park, CA 94025, 415-321-8692.
[6] J. Gaisler, "The LEON2-2 processor user's manual XST edition," *Gaisler Research*, Version 1.0.09, August 2003.

Xue Yang was born in Pizhou city, Jiangsu province, China, in 1988. She graduate from China Aerospace Science and Technology Corporation, Fengtai District, Beijing, China and got her master degree of microelectronic and solid state electronics. She is a junior Engineer. She majors in design of microprocessor, familiar with SPARC V8 architecture, pipeline structure and multi-issue structure.

Yunkai Feng was born in Shandong province, China, in 1986 He got his master degree of microelectronic and solid state electronics. He is a junior Engineer. He majors in medical devices, familiar with C, SystemC, VHDL, Verilog.

Lixin Yu was born in Anhui province, in 1976. He got his doctor degree of microelectronic and solid state electronics, deputy chief engineer. He majors in design, simulation, manufacture and test of microprocessor, and he is also familiar with the development of microprocessors of different architectures and structures.