Design Methodology of a 200MHz superscalar microprocessor: SH-4

Toshihiro Hattori, Yusuke Nitta, Mitsuho Seki, Susumu Narita,
Kunio Uchiyama, Tsuyoshi Takahashi, Ryuichi Satomura

Hitachi, Ltd.
1-280 Higashi-Koigakubo, Kokubunji,
Tokyo 185-8601, JAPAN
+81-423-23-1111

hattori@crl.hitachi.co.jp

1. ABSTRACT

A new design methodology focusing on high speed operation and short design time is described for the SH-4 200MHz superscalar microprocessor. Random test generation, logic emulation, and formal verification are applied to logic verification for shortening design time. Delay budgeting, forward / back annotation, and clock design are key features for timing driven design.

1.1 Keywords

Microprocessor, design methodology, verification, timing

2. INTRODUCTION

Recently, LSI design is becoming more difficult and complicated, not only because of the increase in the number of transistors in LSI's and new design constraints required by deep submicron processes, but also because the designer must take into account many factors. Some targets of LSI design are listed below.

(1) High speed operation

(2) Minimizing chip fabrication cost (minimizing area)

(3) Low power dissipation (for portable use)

(4) Minimizing the design time (to shorten time to market)

(5) High reliability

Priority of these targets is different for each LSI design. It depends on the application and market ,etc..

To satisfy these difficult targets, the design methodology becomes more important for an LSI design project. The design methodology of each LSI design must be focused on the targets of the LSI. For example, in case of designing LSI's for portable equipment, a design methodology focusing on low power dissipation must be applied.

Hitachi has been releasing its original RISC architecture SuperH series microprocessors. And we have been developing the new SuperH microprocessor SH-4 which focuses on multi-media equipment for home use. The significant targets of SH-4 are (1) high speed operation and (4) short design time. Of course, other targets are also important for SH-4. This paper will present the design methodology of SH-4, especially focusing on 200MHz operation and short design time.

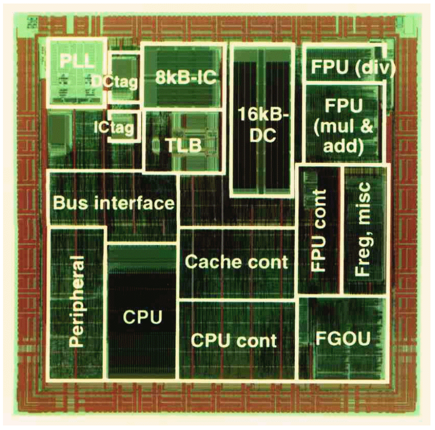

Fig. 1 Microphotograph of SH-4

3. Overview of the SH-4 microprocessor

SH-4 is a 200MHz CMOS 2-issue superscalar microprocessor. Fig. 1 shows a microphotograph of the chip. In this chip 3.2 M transistors are implemented in a die area of 7.6 mm x 7.6 mm. This chip is designed for low-cost graphic and embedded applications. Fig. 2 shows the specifications of SH-4, which runs at 360MIPS(Dhrystone) and 1.4GFLOPS(peak) performance.

SH-4 implements new inner product instructions and it makes possible excellent 3 dimensional computer graphics performance at a consumer price. According to a simple benchmark, SH-4's 3D CG geometry performance is 5.0M polygon/sec.

The process used is 0.25 µm CMOS, with Leff= 0.20 µm, and 5 metal layers. The metal pitch is 0.98 µm for layers 1, 2, and 3, and 1.96 mm for layers 4, and 5. The fourth and fifth layers are used for long (> 4 mm) wires and Vcc, GND because of their low resistivity.

Technology	0.25 µm CMOS, 5 Layer Metals
Voltage	1.8 V (I/O: 3.3 V)
Frequency	200 MHz (internal) / 100,66, etc. MHz (I/O)
Performance	360MIPS (Dhrystone), 1.4 GFLOPS (peak)
Cache	8/16 KB (Inst./Data) Direct-mapped
TLB	4/64-entry (Inst./Unified) Fully-associative
Interfaces	SRAM, DRAM, SDRAM, burst ROM, PCMCIA
Peripherals	DMAC, SCI, RTC, TMU,INTC,EMU

Fig. 2 SH4 Specifications

4. Overview of SH-4 Design Methodology

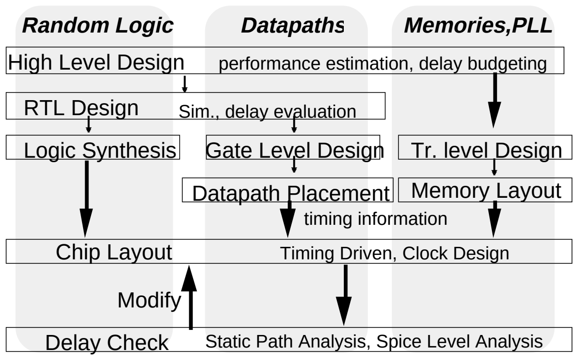

Fig. 3 Design Flow of SH-4

Fig. 3 shows the rough design flow of SH-4. SH-4 architecture simulator SH-4fun was implemented in the C-language. This simulator is used for performance estimation and generating expected behavior for test programs. SH-4 is mainly described in register transfer level Verilog language. First and main logic verification is done at this RTL description. A logic synthesis tool generates the gate level Verilog description. Timing verification is mainly based on the static path analysis of every path between FF's. Some analog circuits, for example memory array and PLL, are verified by a circuit simulator. Memory megacells are designed using a circuit simulator and manual layout tool as same as conventional LSI's. In the SH-4 design, only datapaths in FPU and CPU are performed as structured elements. Other datapaths are layouted as random logic, because timing constraints for these datapaths are not so severe and to layout together with other random logic can reduce the chip area. But, because the datapaths in FPU and CPU are critical for chip operation speed, these datapaths are semi-manually layouted. Their gate level logic is not synthesized from RTL description, but designed manually and modified by generating a new cell library. Semi-manual placement is applied for the purpose of data flow regularity. All random logic were placed at once preserving logic structure, and interblock and intra-block signals are routed at once, because of shortening design time and performing timing driven layout throughout the chip.

5. Logic Verification Methodology

Fig. 4 shows SH-4 logic verification flow. Because increasing transistor number and complicated architectures such as superscalar make logic verification difficult, a logic verification methodology is very important for shortening design time.

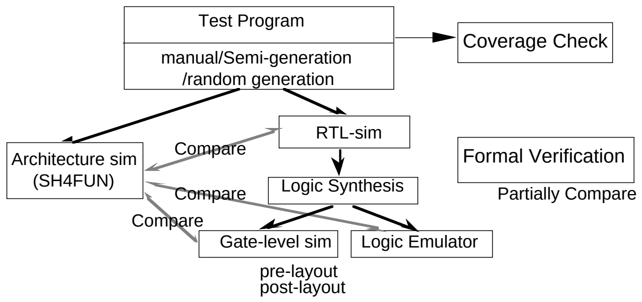

Fig. 4 Logic Verification Flow

5.1 Pseudo-board simulation model

Because preparing input 0/1 stimulus and expected output is very difficult for a large scale microprocessor, we built a pseudo-board in the simulation environment. The pseudo-board contains SH-4 and external memory, some of which data is programmed, and external circuits such as serial interface, and interrupt control description. In this environment, designers only make test programs in assembly language and the architecture simulator SH-4fun can generate expected results easily.

5.2 Random Test Generation

In logic simulation, quality of logic verification deeply depends on quality of the test programs. In our project, a test program design team separate from the logic design team is organized to make test programs in order to check logic functions. However, for the complicated microprocessor, it is impossible to make perfect test programs which can verify all functions of the LSI. Making many good test programs is an expensive and time consuming job. In addition to designer's well-considered test programs, we adapted semi-automatically generated test programs and random generated test programs. Semi-automatically generated test programs are generated by a script which varies the data value of test programs using a template program. The random test generator varies many factors of chip situation and the instructions themselves are randomly generated. But, by simple random test programs, it is hard to find logic bugs. We made many random test generators for specific purposes.

5.3 Logic Emulation

In order to verify a complicated microprocessor, the number of test programs and steps of each test program must be increased. Especially, because random generated test programs are easily made, the number of test sequences increases rapidly. However, to simulate the large scale microprocessor requires much CPU time. Fig. 5 shows the typical simulation speed using a compiled-type high-speed Verilog simulator on a high performance workstation. To overcome this situation, logic emulation is used. We use SimExpress[1] as the logic emulator. Conventional logic emulators have a problem in that the building the netlist inside the logic emulator is very difficult. Not only does netlist compilation and implementation into FPGA take 2days using 10 workstations, but also to fixing timing / skew trouble inside the logic emulator requires a long time. In some cases, the logic emulation environment using the final netlist was set up after first silicon had come out. SimExpress can build the logic emulation environment in 3~5 hours using one workstation from the final fixed netlist. Thus, we can use the logic emulator for not only system level final verification, but also for logic verification before tape out. In case of SH-4 verification, the emulation speed was limited to 100K cycle / second because we used two systems connecting by wires to emulate the large scale whole SH-4. But, practically emulation speed is limited by data transfer time from / to the emulator. Both a test program which requires 3 hours and another test program which requires 1 hour in logic simulation take only 20 seconds in logic emulation.

System	Compile Time	Execution Speed
Gate-level simulation	5Hr.	0.9 cycle/sec
RTL simulation	3Hr.	15 cycle/sec
Gate-level emulation	5Hr.	85K cycle/sec

Fig. 5 Logic Simulation/Emulation Performance

5.4 Formal Verification

Logic verification by simulation / emulation cannot be perfect verification because it is performed under limited test programs. If we must verify the equivalence between two logic descriptions, the formal verification technique is useful. In our project, formal verification is used in the case below.

(i) Verification of equivalence between RTL datapath description and gate-level datapath description which is designed manually.

(ii) Verification of equivalence between before and after logic modification, for example, clock tree generation by layout tool.

(iii) Verification of equivalence before and after minor logic change for performance such as buffer insertion to avoid minimum delay violation.

[1] SimExpress is a product of Mentor Graphics Corperation.

6. Timing Design

The most significant target of SH-4 design was the 200MHz high-speed operation. To achieve this target, we have taken care of the timing at every design phase. Fig. 6 shows the timing design flow of SH-4. We will describe some features below.

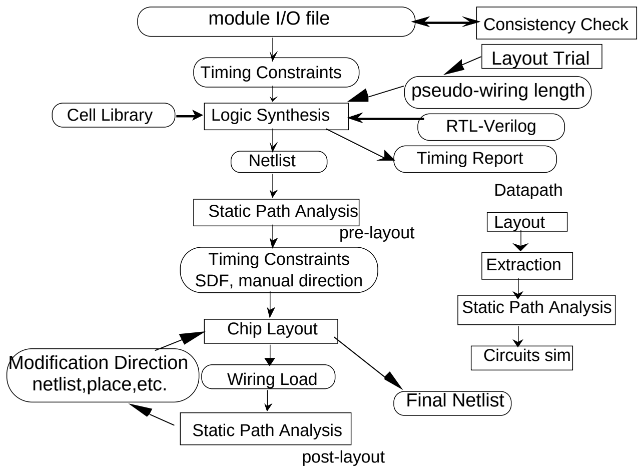

Fig. 6 Timing Design Flow

6.1 Delay Budget for each module

The key factor of timing design is the timing driven logic synthesis. But, logic synthesis can only be performed at the module level. In these cases, the most difficult is definition of timing constraints for inter-module delay. To perform good logic synthesis, accurate module i/o timing constraints are very important. In our project, the so-called module I/O file describes signal arrival time, target delay, and wiring load, which are checked for their consistency by the in house tool. This master file describes the delay budget of each module and makes chip level path delay satisfy the requirements.

6.2 Chip Level Static Path Analysis

In previous microprocessors, datapaths were layouted as structured elements. But this may allow an escape of a real critical path through a datapath from static path analysis. In our project, datapaths are layouted as white boxes for chip level static path analysis; all paths including CPU/FPU datapath can be listed up in the timing report.

6.3 Back Annotation & Forward Annotation

The timing-driven logic synthesis system requires estimated wiring load for the synthesizing net and wire load for the terminals of the synthesizing module. These constraints affect the synthesized netlist. We performed layout trials using preliminary netlists and evaluated back annotated wiring lengths for inter-module nets and estimate average wiring length for internal nets of the module. Results of the chip level static path analysis were forward annotated to the layout tool as timing constraints. Also the final results of net capacitance was back annotated to static path analysis and some modifications were added to the layout.

7. Clock Distribution Design

SH-4 has three clocks, internal (processor) clock (ICLK), bus-clock (BCLK), and peripheral-clock (PCLK). The frequency ratio of ICLK/BCLK and BCLK/PCLK are programmable; for example ICLK/BCLK can be 2,3,4,6, and 8. The reason why we use this three clock scheme is that the reduced bus- and peripheral-clock frequencies saves the maximum clock distribution power by 23% compared with a 200MHz single clock design. In order to perform at high speed, we must also reduce the clock skews between these different frequency clocks.

In order to realize this complicated clock distribution logic, logic design and physical design of clocks are performed at the layout design phase. At the logic design netlist, the clock net is expressed with only one gated clock buffer, which has imaginary large drivability, and drives hundreds of FF's in a logic module.

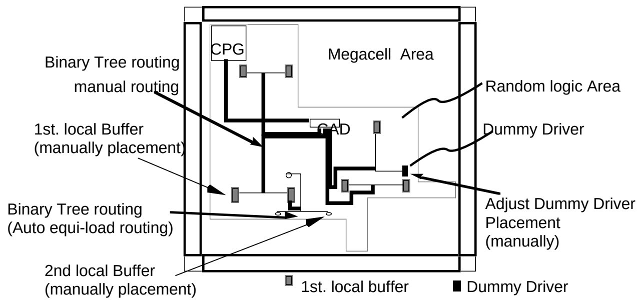

Fig. 8 Chip Level Clock Distribution

As we mentioned previously, minimizing clock skew and minimizing path delay must be pursued in order to design high speed LSI's. However, sometimes these two targets require different solutions. For example, in order to minimize clock skews, a model in which FF's are placed uniformly in the chip is adequate. But in order to minimize path delay, FF's which are at edges of critical paths must be placed near by. To minimize clock skew and minimize path delay simply, we separate the design phase and at each design phase only one target is focused upon.

(1)Perform top level clock distribution logic carefully, and pre-place chip level buffers.

(2)Place random cells only considering timing driven.

(3) Synthesize the clock tree and place clock buffers according to the placement of FF's.

(4)Manually modify the placement of clock buffers because clock tree synthesis and an automatic clock buffer placement create poor results near megacells.

(5)Route top level clock tree(from CPG to 1st local buffers) nets carefully by hand before signal routing because clock nets must avoid detours. Use Metal 4 and Metal 5 for some top level clock nets which drive large loads because of their low resistivity.

(6)Apply automatic zero skew routing for the clock nets from 1st local buffer to 2nd local buffers.

(7) Execute signal routing after clock routing. Direct signal nets not to run near some critical clock nets.

(8) Perform delay/skew analysis with wiring load because a skew value is very sensitive to adjacent wiring capacitance.

(9)Do final modification by adding extra segments of the net to increase wiring load and so on.

Using these careful design, we achieve less than 200pS clock skew for multi-phase FF's including FF's inside megacells according to the static path calculation. However, taking account of process scattering and power supply drop inside the chip ,etc., the worst case skew estimation for the racing problem is larger than 200pS. Measured clock skew from the sampling points in a fabricated chip is less than 100pS.

8. Conclusion

A design methodology for SH-4, which is a high performance superscalar microprocessor for home multi-media equipment, has been described. Pseudo-board simulation model, random test generation, logic emulation, and formal verification are applied in order to shorten design time. Combination of random test generation and logic emulation can verify the netlist very quickly. Formal verification for partial circuits can eliminate re-simulation. RTL delay estimation and delay budget for each synthesizing module help high quality timing driven logic synthesis. To achieve timing driven design, static timing analysis for whole chip and circuit simulation for detailed circuits are used to evaluate the timing. Back annotation from preliminary layout and forward annotation from logic level static path analysis are key features for reducing machine cycle of the chip. Clock distribution logic and layout is designed carefully by automatic tool and manual modification. Clock skews for multi-phase clock is designed to be less than 200pS and measured clock skew from sampling points in a fabricated chip is less than 100pS.

9. References:

[1] Arakawa, F., et al., "SH4 RISC Microprocessor for Multimedia," Hot CHIPS IX, Aug., 1997.

[2] Ishibashi, K., et al., "The Design of 300MIPS Microprocessor with a Full Associative TLB for Hand-held PC OS," Symp. VLSI Circ. Digest of Technical Papers, pp. 9-10, June, 1997.

[3] Nishii, O. et al., "A 200MHz 1.2W 1.4GFLOPS Microprocessor with Graphic Operation Unit, " Int. Solid-State Circ. Conf., Feb.,1998

[4] Osada, K., et al., "A 2ns Access, 285MHz, Two-Port Cache Macro using Double Global Bit-Line Pairs," ISSCC Digest of Technical Papers, pp. 402-403, 494, Feb., 1997.