VLDP Multipath Execution: Mechanism and Evaluation

Shengying Li,[††] Naoya Hattori,[†] Yuichiro Ajima,[††]
Shuichi Sakai[†] and Hidehiko Tanaka[†]

This paper studies and explores cost-effective execution mechanism for VLDP: a new microprocessor architecture, which performs multipath execution on a large number of execution units with distributed registers. Special attention is payed to current development on the interconnection network for distributed register communication. Trace-driven simulations are performed to quantitatively evaluate the execution mechanism.

1. Introduction

A lot of researches have been made in the area of superscalar, which focus on exploiting Instruction-Level Parallelism(ILP). An important point is whether there is enough parallelism to exploit. S.Jourdan [9] stated this problem that with infinite issue size and instruction window size of 256, instruction per cycle(IPC) could reach 10. Considering usual IPC from 1 to 2 in current even state-of-the-art microprocessor, there is still large space to extend the ILP by scaling the issue width and implementing very large instruction window.

However, the performance of processor is limited by the branch misprediction penalty. Once a branch prediction missed, it takes many cycles before the program recovers the right branch path. One approach, multipath speculative execution, helps to solve this problem.

Our Very Large Data Path(VLDP) architecture is a novel microprocessor combining wide instruction issue in large instruction window and multipath speculative execution together to heavily exploit the instruction level parallelism. We provide 16 on-chip execution units(EU) with register files distributed in them and make research on the method for fast communication among them. Each EU performs the wide issue out-of-order superscalar policy. Multipath speculative execution on the whole multi-EU implements the huge instruction window as large as around 200.

VLDP is constructed by control section [2], execution section [1] and memory section [3]. Former research on execution mechanism of VLDP is presented in detail by Y.Ajima [1]. This paper further describes the current development on register communication via interconnection network and quantitatively evaluates the execution mechanism.

The next section focuses on the execution mechanism to exploit the parallelism both in a single EU and among multiple EUs. We make an inquiry into to the issues of data structure, distributed registers and instruction window size. Section 3 debates the on-chip interconnection network. Moreover, related research that addresses speculative execution in different approaches is described in Section 4, and finally the summary can be seen in Section 5.

2. Execution mechanism

2.1 Introduction

VLDP performs a multipath execution model to overcome mis-speculation penalties by simultaneously processing instructions from both the taken and not-taken out-comes of a branch. Because other paths after additional branches are likely to be encountered before the first branch is resolved, speculative execution will lead more instructions in flight at the same time. That is, more additional functional units, registers, cache ports and so on are needed. Our VLDP compacts maximum 32 instructions into data structure unit , called instruction block(IB). The large number of functional units are clustered into execution units(EU), the register files which are usually centralized in current microprocessor are distributed into EUs and the load/store instructions are specially managed by memory section.

2.1.1 Data structure

Execution model of VLDP is shown in figure 1. Fetch and decode are done sequentially at the speed of 1 IB per cycle. Then the decoded IB is allocated to one free EU according to the arrangement of EU management unit. The IB will not retire from the EU until all instructions in it complete executing. Register communication among EUs and memory ac-

† Graduate School of Information Science and Technology, The University of Tokyo
†† Graduate School of Engineering, The University of Tokyo

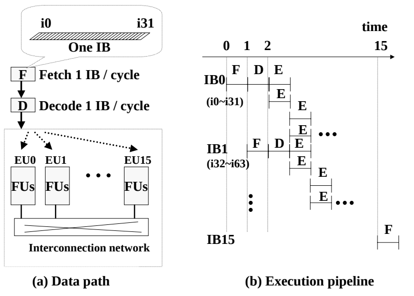

Fig. 1 Execution model of VLDP: (a) Data path; (b) Execution pipeline

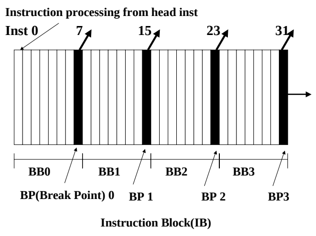

Fig. 2 Instruction block(IB) of VLDP, containing maximum 32 instructions in 4 basic blocks continuously at the same control flow. Branch instruction can only be in Break Point.

cesses are implemented via an interconnection network. Since multiple EUs execute IBs simultaneously, execution is performed in parallel.

To take advantage of instruction parallelism and improve the throughput of instructions per cycle, VLDP processes the instruction block(IB) as the basic unit for execution. The IB is constructed with instructions up to 4 continuous basic blocks on the same control flow statically produced by compiler, and contains opcode, operands, and data synchronous information for each instruction.

As shown in figure 2, an IB has fixed 32 instruction slots, which is divided into 4 Basic Block(BB) fields with 8 instruction slots in each BB. If it happens that there are more than 8 instructions in one basic block in original program, they are scattered into multiple BBs. The branch instruction can only appear at the end of the BB field, which is called *Break Point*. Each IB is always processed from the head instruction. Depending on the result of branch, the real control flow may divert from the path of the IB at a certain Break Point. Instructions after the break point then will be cancelled dynamically during execution. By fetching IBs in multipath [4] and processing multi-IB in parallel, VLDP executes instructions from multiple paths simultaneously.

2.1.2 Distributed register files

- Objective
  To support the multipath speculative execution, VLDP executes multiple IBs on EUs in parallel. Further, every EU performs a multi-issue superscalar execution and provides enough functional units to exploit the ILP of program. However, as described by S. Palacharla [10], large instruction issue size will cause comparably complexity and expensive register rename logic for shared register files. In order to avoid the high cost on it, VLDP distributes register files into separate EUs and manages registers by register synchronous maps in IB.
- Organization
  VLDP provides 64 logical registers with 64-bit, shared by both integer and floating point arithmetic processing. Physical registers are distributed in 16 EUs. In every EU, there are 8 register banks with 32 register entries, that is, 256 physical registers in each EU and overall 4096 physical registers.
- Register communication
  Instead of register rename logic, result of instruction in IB is written into fixed register entry of allocated register bank for the IB directly on the EU. So, VLDP only need to communicate for register reading, when an IB need values from other EUs. Although the distributed registers avoid the large cost of register rename logic for concentrated register file, it brings additional cost of register communication.

2.1.3 Instruction window

Instruction window size refers to all the instructions simultaneously held in the execution pipeline. Benefit from low cost of register access to distributed register files, VLDP introduces as large as 16 EUs to process maximum up to 512 instructions in parallel in order to support the multipath speculative execution. Because new IB can not enter an EU until old IB retires from it, the real instruction window size in execution is less than the maximum value, which is examined in simulation.

2.2 Evaluation

2.2.1 Simulation environment

- Simulator
  We developed cycle-level multi-EU superscalar simulator on the data structure of IB. This simulator achieves the most aggressive instruction issue policy inside of an EU: out-of-order issue using reservation station and forwarding mechanism. Further it per-

Table 1 Benchmark summary, including the number of IBs and instructions executed on the seven SpecInt 95 benchmarks, in the unit of million(M).

	go	m88ksim	compress	li	ijpeg	perl	vortex
IB	9.1M	9.6M	9.1M	18.3M	8.4M	12.7M	11.2M
Inst	165.0M	151.5M	174.0M	255.5M	211.0M	206.3M	210.3M

Table 2 Instruction window size of VLDP, referring to the instruction number simultaneously active in the execution pipeline.

go	m88ksim	compress	li	ijpeg	perl	vortex
198.4	201.6	192.0	139.2	304.0	188.8	148.8

forms the multipath execution across EUs: IBs in poly data paths can speculatively execute on multiple EUs simultaneously. In order to focus on investigation of the performance purely on execution mechanism, we assume:

 – ideal instruction-fetch mechanism: one IB per cycle;
 – perfect cache;
 – perfect branch predictor;
 – wide issue size of 32 in an EU;
 – 1 cycle for non-memory instruction execution and 2 cycles for memory instruction execution.

• Benchmarks
In this paper, we evaluate our scheme using seven applications in the SPECint95 benchmark summarized in Table 1.

2.2.2 Result and discussion

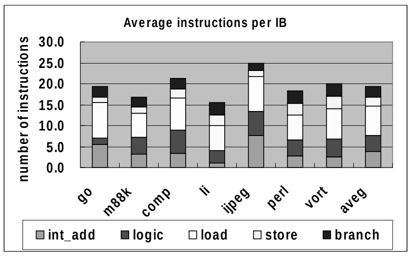

Fig. 3 Instructions per IB for benchmarks, showing the number of non-nop instruction per IB.

• Instruction block(IB)
Upon the definition of IB in VLDP, we examine its organization on benchmarks, summarized in figure 3. As it indicates, average non-nop instructions per IB is around 19, which is heavily limited by the return instruction.

• Instruction window size
Table 2 shows the investigation result on instruction window size. The upper bound of 512 for VLDP appears when 16 IBs with 32 instructions enter EUs. Although the

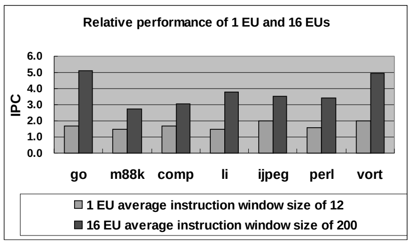

Fig. 4 Performance impact of instruction window size, showing instruction window size of around 200 speeds up 1.8-3.0 to that of around 12

EU can not accept new instructions until the IB on it retired or cancelled, the size of instruction window is still as large as around 200 in the simulations. Figure 4 verifies the performance(IPC) impact of the instruction window size. Compared to IPC on single EU with instruction window size around 12, VLDP speeds up from 1.8 to 3.0 on the benchmarks.

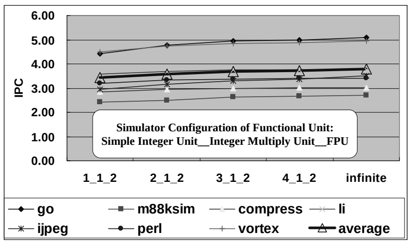

Fig. 5 Functional unit configuration

• Functional unit configuration
On the basis of data structure, IB, we investigated the performance of IPC for alternative fixed numbers of functional units in EUs, and compared with that for infinite number of functional units. To support floating point applications, we assume

two floating point units in our simulators. Figure 5 indicates that configuration of 3 simple integer units, 1 integer multiply unit and 2 floating point units in an EU is a comparably effective configuration for VLDP.

3. Interconnection network

3.1 Introduction

In VLDP, via the interconnection network on-chip, separate EU may communicate either through the shared memory, or through registers. Since the data transfered among EUs need not to leave the chip, communication is fast. The on-chip cache and shared memory allow quantities of simultaneous accesses [3]. VLDP also implements register communication among EUs, allowing an EU to read register values in other EUs. This section focus on the register communication among EUs.

Because results for instructions in an IB are written into local registers directly on the EU, VLDP only need transfer register value for read. To perform fast communication, we perform the register communication on the register-ready mechanism driven by sending-side EU. Once values in the registers are valid and interconnection network is free, they are transfered to the destination EUs. When data arrives from a remote register write, any operation waiting on the register is allowed to issue.

Register request information for an IB is produced by decode unit and broadcasted to all related EUs that hold the needed physical register. When the IB is allocated into an EU, through interconnection network, the requested register values are sent from other EUs, if ready. It is the time that may be a communication burst for an EU attempts to be accessed by multiple other EUs simultaneously. Here, such an EU is called a *hot EU* and the phenomenon is called *hot-spot contention*. Register communication at any other time is called *regular communication.*

3.2 Evaluation

- Simulation environment

  This study is made according to the assumptions outlined in section 2.2.1. Further, when allocating a new IB into free EU, the priority of EUs decrease as the EU number increases from 0 to 15. That is, EU 0 features the highest priority while EU 15 has the lowest one.
- Analysis of Register Communication

  Before conversion to the investigation on interconnection network, we first examine the register communication traffic among EUs.

  Figure 6-8 illustrate the frequency, traffic and EU relationship for distributed register communication. Focusing on one EU, **HS(d)** and **HS(s)** gives the information at the cycle when the EU is the destination or the source for *hot-spot contention* respectively; **R(d)** and **R(s)** shows information when the EU receives or sends as destination or source respectively in *regular communication*; **EU traf** includes all the communication information for an EU.

  The traffic for regular communication on an EU is around 1.5 registers per cycle with frequency of 4% among the simulation cycles. At the same time, it is clear that the traffic for a hot EU is not high, about 2.5 from 2 other EUs with frequency of 1% among the whole simulation time. Part of the reason is the instruction parallelism limited by the data dependence among the IBs in the same data path. Another reason is that 100% branch prediction accuracy in our current simulator lessens the communication among EUs. We will examine the impact of the hot-spot contention on performance in the latter section.

  In addition, simulation result verifies that there is no obvious difference in EUs for register communication.

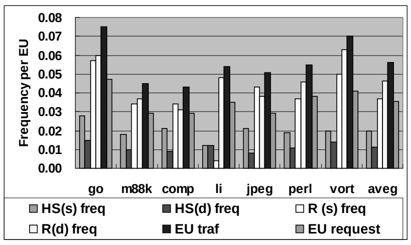

Fig. 6 Register communication frequency among EUs. Cycles for an EU to perform regular communication are 4 times as many as those for hot-spot contention.

- Requirement of register communication for data network

  In this section, we examine VLDP requirement to interconnection network on register communication. Figure 9 shows the heavy performance impact of communication latency with infinite bandwidth. IPC increases 30% when latency decreases from 4 cycles to 1 cycle. Figure 10 summarized the impact of communication band-

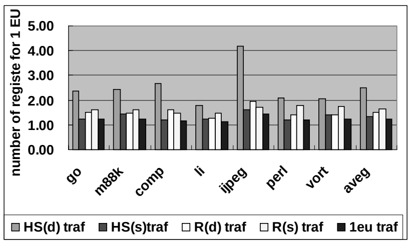

Fig. 7 Register communication traffic among EUs. A hot EU receives average 2.5 registers per cycle, which is larger than that of 1.5 per cycle for a regular EU.

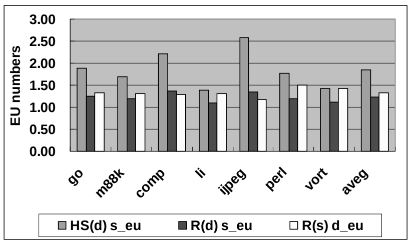

Fig. 8 Communication relationship among EUs. HS(d)s_eu shows that a hot EU as communication destination receives from average 2 EUs; R(d) s_eu or R(s) d_eu indicates that an EU in regular communication receives from or send to around 1 EU.

width with ideal latency zero on performance IPC. Here, Crossbar means the point to point crossbar. To examine the influence of the hot-spot contention, in the Crossbar_3_Rport, we added 2 receive ports especially for the hot EU. No obvious performance improvement can be seen while the bandwidth increases. Also, hot-spot contention has low impact on performance. So, we focus on small latency interconnection network in future research.

- Analysis of 3 typical data networks
  Considering the latency and hardware cost of different kind of the data network, we examine the following typical data networks.
  - Simple Bus
    Its advantage is low cost. However, latency for communication on bus is usually 3 cycles. Furthermore, bandwidth is limited to 1 messages per cycle.
  - Multistage Interconnection Network
    Cost of multistage interconnection network is more than bus but less than crossbar. By 2x2 crossbar, there are at least 4-cycle latency for register communication among 16 EUs. Further, message contention will cause high latency.
  - Crossbar
    16x16 Crossbar is the most expensive data network among the 3 data networks, but with lowest latency. So we choose the crossbar as our baseline performance for evaluation of interconnection network in VLDP.

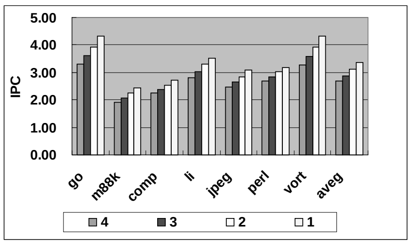

Fig. 9 Application IPC as a function of communication latency with infinite wide bandwidth.

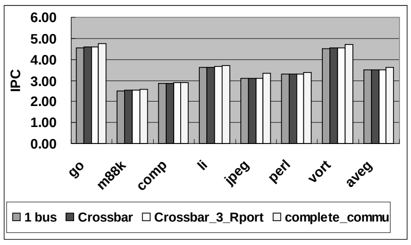

Fig. 10 Application IPC as a function of communication bandwidth with latency zero.

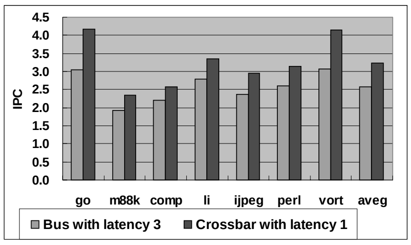

Fig. 11 Performance comparison with bus and crossbar. Comparing to crossbar with latency of 1, bus with latency of 3 decreases IPC of 26%.

Figure 11 demonstrates the performance difference among bus and crossbar. Here, the simulator did not implement pipeline for bus communication. With larger latency, performance of bus is 26% less than that of crossbar.

3.3 Future work

Upon the examination above, we consider to use multi-bus interconnection network with links shown in figure 12. The bus and link network combines the simple low cost bus and the low latency link together. As described in the section 3.2, the IB allocation mechanism in our simulator puts the nearest IBs in the same path into the close near two EUs. Since the register communication happens frequently between the nearest two IBs in the same path, communication among the close two EUs is expected to be high. So we add low latency links among the near EUs like illustrated in figure 12. Detailed research on it will be held in future and we intend to find a cost-effective interconnection network for VLDP.

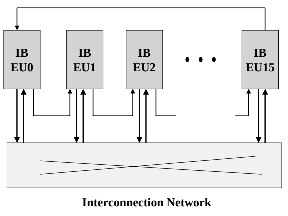

Fig. 12 Interconnection network with link among EUs.

4. Related research

Numerous researches have made on multipath speculative execution to reduce the misprediction cost in different approaches. Selective eager execution [5] executed both paths after branches in an extension of an aggressive superscalar, out-of-order architecture. Y-pipe [6] duplicated the early pipeline stages to eliminate the penalty of branch misprediction. TME [7] attempts to combines simultaneous multithreading(SMT [8]) with multipath execution.

5. Summary

This paper introduced the execution mechanism of Very Large Data Path(VLDP) architecture. We investigated the basic data structure of IB, the large instruction window size and functional unit configuration for VLDP. Furthermore, we present current development in on-chip interconnection network and analyzed requirement for distributed register communication. Upon the simulations on several SpecInt95 benchmarks, quantitative evaluation on performance of IPC shows that comparing to a single EU, the large instruction window of 16 EUs in VLDP speeds up to 1.8 to 3.0. Performance impacts on interconnection network of several basic data network models are also examined. Considering the heavy influence of latency and the hardware cost, we intend to extend our research on multi-bus with links among EUs in the future.

Acknowledgments

This work is supported in part by scientific research founding in Ministry of Education, Japan, (foundation research(B) NO.11480066) and Semiconductor Academic Research Center(STARC).

References

[1] 安島 雄一郎, 辻 秀典, 坂井 修一, 田中 英彦, 大規模データパス アーキテクチャの実行機構, 情報処理学会研究報告 2000-ARC-139, pp.55-60, 2000
[2] 辻 秀典, 安島 雄一郎, 坂井 修一, 田中 英彦, 大規模データパス アーキテクチャの提案, 情報処理学会研究報告 2000-ARC-139, pp.49-54, 2000
[3] 入江 英嗣, 安島 雄一郎, 辻 秀典, 坂井 修一, 田中 英彦, 大規模データパス アーキテクチャに適したロードストアユニット構成, 情報処理学会研究報告 2000-ARC-139, pp.43-48, 2000
[4] 塚本 泰通, 辻 秀典, 安島 雄一郎, 坂井 修一, 田中 英彦, 大規模データパス アーキテクチャにおける命令ブロックの構成の検討, 情報処理学会研究報告 2000-ARC-139, pp.61-66, 2000
[5] A. Klauser, A. Parthankar, D. Grunward, *Selective Eager Execution on the PolyPath Architecture*, 25th Intl. Symposium on Computer Architecture(ISCA), pages 122-131, 1998
[6] M. J. Knieser, C. A. Papachristou, *Y-Pipe: A Conditional Branching Scheme Without Pipeline Delays*, 25th Intl. Conf. on Microarchitecture, pages 125-128, 1992
[7] S. Wallace, B. Calder, and D. Tullsen, *Threaded Multiple Path Execution*, 25th Intl. Symposium on Computer Architecture, 1998
[8] D. M. Tullsen, S. J. Eggers and H. M. Levy, *Simultaneous Multithreading: Maximizing On-Chip Parallelism*, Proc. 22nd Annual International Symposium on Computer Architecture, pages 392-403, 1995
[9] S. Jourdan, P. Sainrat, D. Litaize, *Exploring Configuration of Functional Units in an Out-of-Order Superscalar Processor*, Proc. 22nd Annual Intl. Symposium on Computer Architecture, pages 206-218, 1997
[10] S. Palacharla, N. P. Jouppi, J. E. Smith, *Complexity-Effective Superscalar Processors* Proc. 22nd Annual International Symposium on Computer Architecture, pages 117-125, 1995