High-performance and Low-power Consumption Vector Processor for LTE Baseband LSI

● Yi Ge ● Mitsuru Tomono ● Makiko Ito ● Yoshio Hirose

Recently, the transmission rate for handheld devices has been increasing by Long Term Evolution (LTE), and baseband LSI has come to need a higher performance. In addition, handheld devices will use the second- and third-generation communication method, so a baseband LSI will need to handle multiple communication methods. Because implementing all communication circuits results in a large area, we have been developing Software Defined Radio (SDR), which switches each communication method with software. To implement SDR for handheld devices, a high-performance and low-power consumption digital signal processor (DSP) is needed. We have developed a DSP which inherits the architecture of vector supercomputers, and the architecture has advantages of a low power consumption and application developments. We have downsized the vector architecture for embedded systems. The peak performance is 12 giga operations per second (GOPS) at 250 MHz, and the power consumption is relatively low at 30 mW for 28 nm process technology on average. This paper presents the vector processor that we developed.

1. Introduction

Long Term Evolution (LTE) has recently been becoming the mainstream radio communication method for handheld devices. Baseband processing of LTE is known to be characterized by extremely large amounts of computation. In addition, LSIs for handheld devices require low power consumption and small footprints. Furthermore, handheld devices also use existing radio systems such as High Speed Packet Access Plus (HSPA+) and we have been working on the development of Software Defined Radio (SDR), which accommodates multiple radio communication methods by means of software. Making use of SDR allows circuits to be shared and this leads to cost reductions, and means specifications can be changed and functions added flexibly. For handheld devices, in particular, it is not possible to simply pursue programmability alone for reasons of power consumption, and a balance must be struck with power consumption and circuit area. At Fujitsu, we have developed an SDR baseband LSI with these taken into account.[1)]

The processing conducted in an LTE can be classified into five types: base station searcher (SEA), demodulator (DEM), decoder (DEC), coder (COD) and modulator (MOD) processes. For the baseband LSI development, we have roughly divided baseband processing into processes better served by a digital signal processor (DSP: a microprocessor designed especially for digital signal processing), which requires programmability, and those for which a hardware accelerator is suitable (e.g., turbo decoding of DEC), in order to avoid an unnecessary increase in circuit area or power consumption.

This paper presents the vector processor developed as the DSP.[2)] The base architecture uses a system of vector supercomputers to realize the processing capacity and low power consumption that satisfy the requirements for handheld devices. The present vector processor is intended for SEA, DEM and MOD processes.

In this paper, first, the background of the adoption of the vector architecture as the base architecture of the DSP is explained in the following section, and second, the details of the present vector processor is described. Finally, the vector system is discussed in comparison with other SIMD architectures.

2. Policy of base architecture

To assume software processing of radio signal processing involving large amounts of computing, an understanding must be gained of which part allows parallel computing. Program parallelism is generally classified into (a) data-level parallelism, (b) instruction-level parallelism and (c) thread-level parallelism, and they have an inclusion relation represented by (a) $\subset$ (b) $\subset$ (c).[3),4)] However, the amounts of hardware and processing overhead required for processing are increased as represented by (a) < (b) < (c) and it is better to process low-level parallelism with an apparatus suited to that level. For example, a multi-core processor capable of processing thread parallelism can handle wide parallelism but the ratio accounted for by the computing unit out of the entire circuit area is small, which means there is a low computational performance per unit area. This increases the processing overhead.

The present baseband LSI is given a multi-core architecture in which the respective process is assigned a DSP at the system level, because there is thread level parallelism between the five processes mentioned above. Meanwhile, each process is known to have a relatively large amount of data-level parallelism. For the parallelism, single instruction multiple data (SIMD) method is generally suitable for processing. The SIMD method, in which simply multiple computing units are provided in one control unit for processing multiple data with one instruction, can effectively enhance computational performance per unit area. For this reason, we have decided to use the SIMD method as a base approach to the DSP.

The SIMD method generally includes a fixed-length SIMD architecture such as Intel Streaming SIMD Extensions (SSE) and a vector architecture used for vector supercomputers. In a fixed-length SIMD architecture, SIMD-width data is processed by one instruction in one cycle and this ends the computation. Unlike this, in a vector architecture, vector-length data is automatically divided into SIMD-width data by the hardware sequencer and computation is executed over multiple cycles.

Benefits of a vector architecture include:

1) Allows programming without the need to be aware of the SIMD width, which is a hardware-specific parameter, and improves application portability.
2) Computation for an array length of more than the SIMD width can be specified with one instruction, which improves the instruction compression efficiency, and the number of instruction fetches from the instruction memory is reduced, leading to a reduction of power consumption.
3) The data parallelism extraction software platform, which has been developed for a long time in the vector supercomputer field, can be reused.

In addition, Fujitsu's accumulation of the past vector technology can be applied. Accordingly, a vector architecture has been adopted as the base architecture.

3. Present vector processor

This section outlines the developed vector processor. **Figure 1** shows the block architecture of the DSP developed. This vector processor is composed of a CPU unit and vector unit (VU). As the CPU core, LX3 processor of Cadence Design Systems is used.[5)]

3.1 Specifications

Table 1 shows the major specifications of the present vector processor. The four vector pipelines are composed of two multiplication pipelines and two load/store pipelines. ALU instructions can be issued to all of the four pipelines but multiplication and load/store instructions are limited to two pipelines respectively. The SIMD width is 8 (for 16-bit data instruction). The maximum vector length has been specified as 64 in view of the nature of the intended applications. The instruction-issuing performance is one instruction per cycle by in-order issuance. However, successive issuance is possible if there is no register conflict with the preceding and following instructions and, even if there is register conflict, successive issuance is possible if the latency of the preceding instruction is 1 on account of a forwarding mechanism. So the four execution pipelines can be put in full operation. The instruction latency is 1 for ALU computation, 2 for multiplication and 3 for load (including alignment). To reduce resource conflict, the data memory has been configured for 128-bit data width and it has four banks. The vector load/store instructions are equipped with powerful addressing features including the stride feature capable of accessing addresses at uniform intervals and indirect function capable of accessing an arbitrary address specified, which allows high-speed execution.

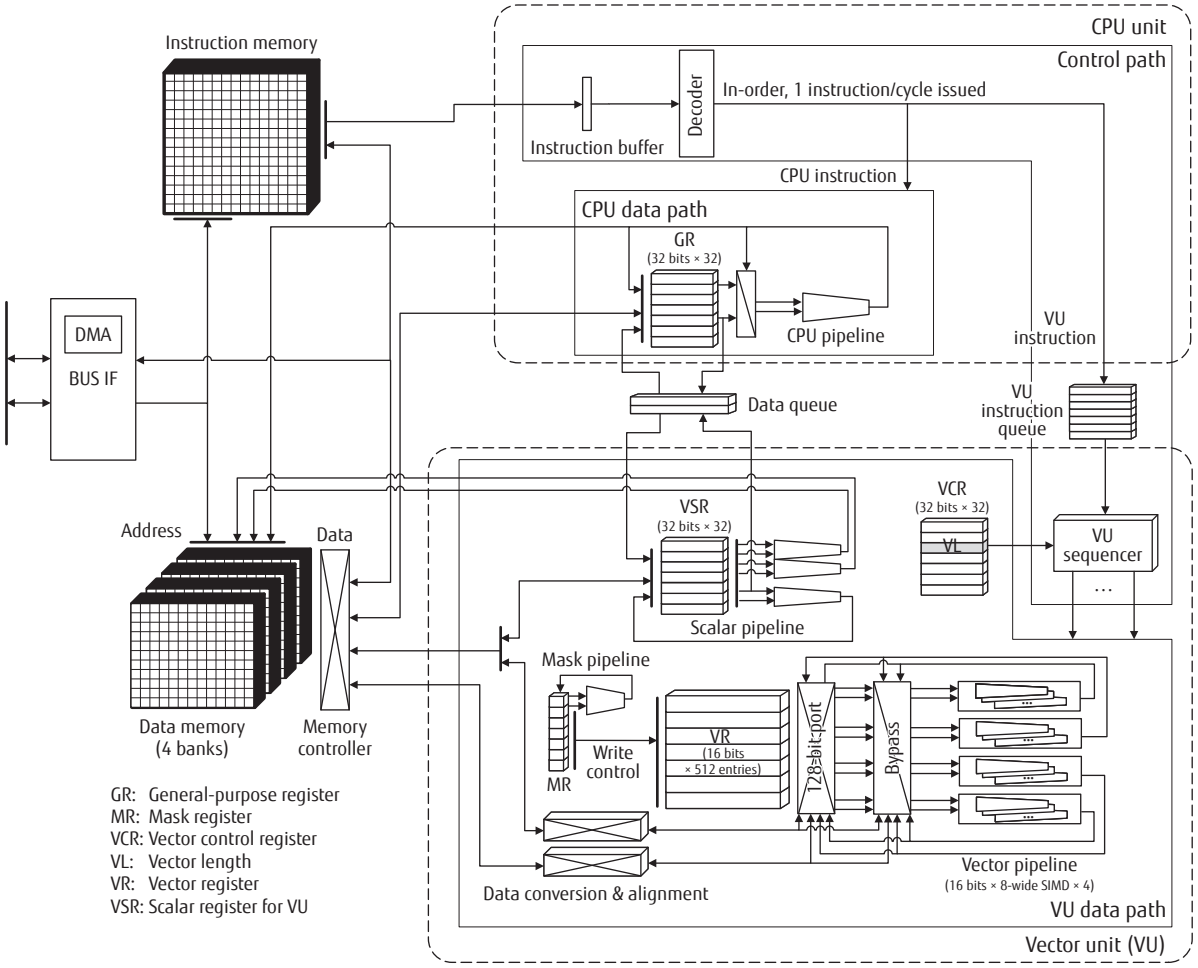

Figure 1
DSP block architecture.

3.2 Instruction set

Table 2 outlines the instruction set of the VU of the vector processor. The instruction set, which is composed of scalar instructions (not CPU instructions; 40 instructions) and vector instructions (113 instructions), contains a total of 153 instructions. Other than those listed, the vector instructions include several dedicated instructions for radio signal processing. The instruction length is 32 bits. The data word length handled by vector instructions can be 8, 16 or 32 bits. The default data word length has been specified to be 16 bits, in view of baseband processing. The processor is intended for radio processing and, for low power consumption, only integer computation is handled.

3.3 Microarchitecture

This subsection describes the flow of processing of the DSP. First, the program is stored in the instruction memory. Instructions are fetched by the CPU core only and stored in the instruction buffer. Then, the instructions are interpreted by the decoder of the CPU core. The program sequence contains CPU and VU instructions mixed together. CPU instructions are passed to the CPU data path and executed by the CPU. Branch instructions are only in the CPU and the control flow is controlled only by the CPU. VU instructions are passed to the VU sequencer via the VU instruction queue. Control of the VU is based on reception of instructions, after which they are processed independently of the CPU. However, the system has a memory barrier instruction, which can be used for synchronizing CPU and

Table 1
Major specifications.

Item	Specification
Operating frequency	250 MHz
Word length	8-bit, 16-bit (main), 32-bit integer
No. of pipelines	Vector load/store / ALU × 2 Vector multiplication / ALU × 2 Scalar × 1
No. of pipeline stages	5
SIMD width of 1 pipeline	8 (for 16-bit data), 4 (for 32-bit data)
Instruction issuing performance	1 instruction/cycle (in-order issuance)
Supported vector length	8 to 64
Vector register file	16 bits × 512 entries
Support for intra-loop branch	Mask registers provided
Data memory	Up to 512 KBytes (128 KBytes × 4 banks)
Peak computational performance	48 operations/cycle [12 GOPS (giga operations/second) at 250 MHz]
Peak load/store performance	256 bits/cycle
Power consumption (entire DSP including memory)	Up to 30 mW (28 nm process LSI)

Table 2
Instruction set.

Instruction supported by VU of vector processor		Data word length handled by instruction			
		8 bits	16 bits	32 bits	64 bits
Scalar instruction	Scalar load/store	○	○	○	○
	Scalar add	–	–	○	–
	Scalar shift	–	–	○	–
	Scalar logical	–	–	○	–
	Scalar move/immediate value set/cut	–	–	○	–
Vector instruction	Vector load/store	○	○	○	–
	Vector add	–	○	○	–
	Vector multiply/product-sum/inner product	–	○	–	–
	Vector shift/compare	–	○	○	–
	Vector logical	–	○	–	–
	Vector word length convert/bit operation	–	○	○	–
	Vector select	–	○	–	–
	Vector max/min value search	–	○	○	–
	Vector extract/sum	–	○	○	–
	Vector move/shuffle/merge	–	○	–	–
	Vector mask	–	–	–	–
	Vector barrier control	–	–	–	–

VU. The VU sequencer identifies any resource conflict or register conflict with already issued instructions in the VU and controls the issuance of the relevant VU instruction. If the instruction issued is a vector instruction, it is issued to the vector computation pipelines. The number of cycle repetitions of the relevant instruction is determined based on the vector length prespecified in the vector control registers (VCRs) and the SIMD

parallelism, and the instruction is repeated by that number in the vector pipelines. Mask registers (MRs) are provided to support intra-loop branch (function to conditionally select a computation result) and mask processing (write control) of the respective elements of a vector is possible with almost all vector instructions. For data transfer from the VU to the CPU, transfer instructions are used to move data from the vector registers (VRs) of the VU to the general-purpose registers (GRs) of the CPU.

3.4 Performance evaluation

Some core routines of radio signal processing have been used to compare the performance achieved by using the present vector unit with that of a standalone CPU (LX3). For the evaluation, a cycle-accurate simulator has been used. Vectorization is manual assembly programming. **Figure 2** shows the performance as compared with the CPU. The program for evaluation handles basic processing including inner product and maximum value search, etc., and the array sizes are 256 and 1024 (FFT: 2048 points). The figure shows that a larger improvement can be achieved with a higher array size. The performance improvement is up to 40 times as much as the standalone CPU. The desired processing performance is realized by significantly accelerating the core routine of the relevant process.

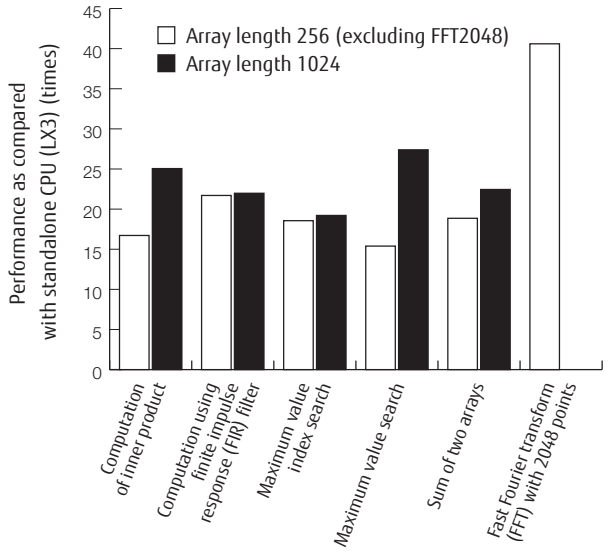

Figure 2
Performance comparison with CPU.

4. Discussion

Vector architecture systems include Fujitsu VPP Series and NEC SX Series (Earth Simulator)[6] and, as a vector unit separated as a co-processor, Fujitsu μVP.[7] The base architecture of our vector processor inherits these characteristics and has been adapted to baseband processing by limiting the data word and maximum vector length and reducing the size of the register file. Use of a vector architecture makes it easier to apply the existing platforms such as FORTRAN. It raises expectations as a promising architecture to be rediscovered in the future.

5. Conclusion

This paper has presented the vector processor developed by Fujitsu Laboratories. The base architecture inherits that of vector supercomputers. The peak computational performance is 12 giga operations per second (GOPS). The power consumption can be reduced to about 30 mW on average with a 28 nm process LSI and the performance desired for a baseband LSI supporting LTE for devices has been achieved. The present vector processor has been adopted for commercial baseband LSIs of Fujitsu.

References

1) T. Sasaki et al.: LTE-compliant Multi-Radio Access Technology (RAT) Baseband LSI. *Fujitsu Sci. Tech. J.*, Vol. 49, No. 2, pp. 245–249 (2013).
2) Y. Ge et al.: A vector coprocessor architecture for embedded systems. SoC Design Conference (ISOCC), 2011 International, pp. 195–198, 17–18 Nov. 2011.
3) C. Batten: Vector-thread architectures. Churchill College Graduate Student Seminar Series, University of Cambridge, Mar. 2004.
4) D. A. Patterson: Future of computer architecture. Berkeley EECS Annual Research Symposium 2006, Feb. 2006.
http://www.eecs.berkeley.edu/BEARS/presentations/06/Patterson.ppt
5) Cadence Design Systems.
http://www.tensilica.com/
6) T. Sato: The earth simulator: Roles and impacts Nuclear Physics B - Proceedings Supplements In Lattice 2003, Vol. 129–130, pp. 102–108, Mar. 2004.
7) M. Awaga et al.: The μVP 64-bit Vector coprocessor: A New Implementation of High-Performance Numerical Computation. *IEEE Micro*, Vol. 13, No. 5, pp. 24–36 (1993).

Yi Ge (Takeshi Katsu)
Fujitsu Laboratories Ltd.
Mr. Ge is currently engaged in development of DSPs for baseband SoCs for handheld devices.

Makiko Ito
Fujitsu Laboratories Ltd.
Ms. Ito is currently engaged in development of DSPs for baseband SoCs for handheld devices.

Mitsuru Tomono
Fujitsu Laboratories Ltd.
Mr. Tomono is currently engaged in development of DSPs for baseband SoCs for handheld devices.

Yoshio Hirose
Fujitsu Laboratories Ltd.
Mr. Hirose is currently engaged in development of DSPs for baseband SoCs for handheld devices.