International Journal of Engineering Research and Development
e-ISSN: 2278-067X, p-ISSN: 2278-800X, www.ijerd.com
Volume 9, Issue 5 (December 2013), PP. 33-37

A VLIW Vector Media Coprocessor with Cascaded SIMD ALUs

Prof.T.Chandrasekhar[1], J.S.Chakravarthi[asst. Professor2], Y.V.Sai Roja[3]
[1,2,3]*GIET Engg. College, Rajahmundry*

Abstract:- High-definition video applications, such as digital TV and digital video cameras, require high processing performance for high-quality visual images in addition to a complex video CODEC. Pre-/post processing to improve video quality is becoming much more important because requirements for Pre/post processing vary among applications and processing algorithms have not been stabilized. Therefore, a new processor architecture that has a highly parallel data path is needed. In this Paper, we introduce a VLIW vector media coprocessor, "vector coprocessor (VCP)," that includes three asymmetric execution pipelines with cascaded SIMD ALUs. To improve performance efficiency, we reduce the area ratio of the control circuit while increasing the ratio of the arithmetic circuit. The total gate count of VCP is 1268 kgates and its maximum operating frequency is 300 MHz at 90-nm CMOS process. Some of the processing kernels in an adaptive prefilter that is applied to preprocessing for video encoding are evaluated. In the case of the edgeness and the sum of absolute differences, the performance is 183 giga operations per second. VCP offers enough performance for HD video processing and good cost-performance while all processing pipeline units operate effectively.

Index Terms:- Single instruction stream, multiple data stream (SIMD), vector coprocessor (VCP), very long instruction word (VLIW).

I. INTRODUCTION

Now a days, high-definition video applications, such as digital TV and digital video cameras require high pro cessing performance for high-quality visual images in addition to a complex video CODEC. Pre/post processing to improve video quality is becoming much more important because re- quirements for pre-/post processing vary among applications and processing algorithms have not been stabilized. In this paper, we introduce a very long instruction word (VLIW) vector coprocessor, "vector coprocessor (VCP)," that has been customized to the computation requirements of image processing. The coprocessor architecture includes three asymmetric execution pipelines with cascaded SIMD ALUs to exploit the loop-level parallelism. The new architecture of VCP is a combination of cascaded SIMD ALUs and asymmetric parallel pipelines, which provide good cost-performance to enhance specialized data paths for lower-level image processing, such as preprocessing and post processing, at the expense of generality compared with conventional processors with SIMD instructions. VCP is designed to be a coprocessor for image processing of video CODECs and the width of SIMD ALUs is limited to that of macro blocks of CODECs. Therefore, we introduce a cascaded structure of SIMD ALUs to exploit high parallelism. To achieve high performance with small hardware size, we reduce the area ratio of the control circuit while increasing the ratio of the arithmetic circuit. On the other hand, for the embedded world (particularly for smart-cards), given the constraints of speed, power, size and security, special cryptographic accelerators have been deployed. Most of those Public Key (PK) crypto accelerators propose very elaborate arithmetic processors that work on long precision numbers of fixed lengths, resulting in complicated, bulky and inflexible architectures. Others have been trying to have a more general approach by enhancing the instruction set of general purpose scalar processors . However, none of those approaches have embraced a hardware-software co-design approach for data level parallel techniques to enhance cryptographic computation We begin this paper by performing an extensive study about how cryptography has been implemented on SIMD(Single Instruction Multiple Data) architectures. We show how a design-to-cost approach can be adopted by doing a quantitative analysis on the functional simulation of a modular multiplication operation. We finally summarize our results and compare our work to previous contributions in the field of cryptography's.

II. RELATED WORK

VLIW architectures have been studied to exploit high instruction-level parallelism. SIMD architectures have been used frequently to exploit high data-level parallelism. IMAPCAR operates 128 processor cores with VLIW and SIMD in parallel. The stream processors achieve massively parallel processing by single instruction, multiple data streams with single instruction set to the multiple operation lanes and multiple stream data set to each operation lane. The performance of the stream processors is 512 GOPS (Giga Operations Per Second) for 8-bit

Processing at 800 MHz frequency. SCALE has been in- traduced as an architecture based on this concept. Avispa has high generality to enhance SIMD ALUs for scalability with Ultra Long Instruction Word (ULIW).

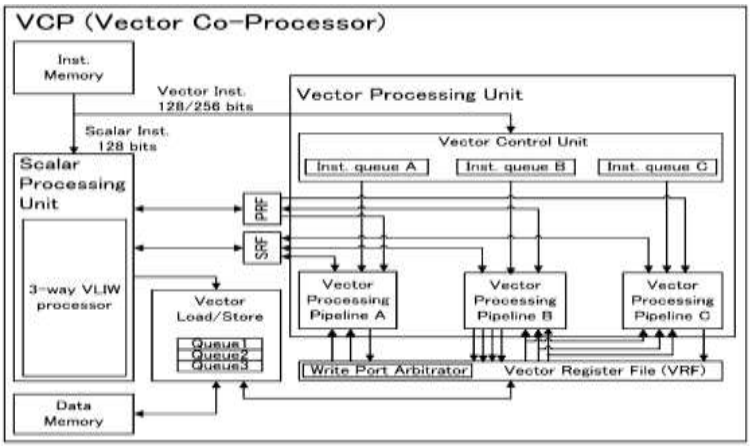

Figure.1: Architecture of VCP

Coarse-grained reconfigurable architectures have also been used. The reconfigurable architectures proposed include DRP, DAPDNA-IMX and REMARC .Automatic map- ping of applications to FPGA-based and other reconfigurable systems has also been investigated. ADRES is an example of an architecture that automatically maps applications onto coarse-grained reconfigurable arrays that are tightly coupled to the VLIW processors.

The proposed processor VCP can exploit high loop-level parallelism by vector processing in addition to VLIW and SIMD architectures. A feature of the architecture is that all processing pipeline units share a vector register file (VRF); this provides the performance improvement when the common data in VRF data is supplied to multiple pipeline units, or the VRF data is cooperatively updated by multiple pipeline units. This architecture concept is similar to those of VIRAM, SODA and DXP .However, using asymmetric vector pipelines with cascaded SIMD ALUs of VCP; we expect to achieve better cost-performance than other processors.

III. PROPOSED VECTOR ARCHITECTURE

The theory of vector processing and its application tomicro-processors is Vector Processor techniques have been widely used from supercomputers like the Cray machine [to Digital Signal Processing applications like in Intel's MMX or in embedded media architectures like VIRAM ,but never for cryptography. In our architecture we use a scalar MIPS to provide good scalar performance. To keep instruction decode simple, we delegate both vector and scalar instruction fetch and decode to the MIPS core. To suit the MIPS 'load-store' architecture and to avoid complex memory accesses, we chose a *Register-to-Register* vector architecture. With this approach we reduce memory-register transfers, which are also the privileged attack paths for side channel analysis. Details about the vector architecture implemented for the analysis done in this paper are given in .we need to highlight some of the vector processor's architectural details. The architecture of the vector register file is illustrated in Six architectural parameters influence the structure of our vector register file:

• m: The size of each element of the vector registers(m = 32).
• q: The number of vector registers.
• p: The number of elements, called *depth*, in each vector register.
• r: The number of lanes which correspond to the number of Vector Processing Units (VPUs). This notion is borrowed from .Ideally we would have $r = p$, allowing us to work on all p elements in parallel.

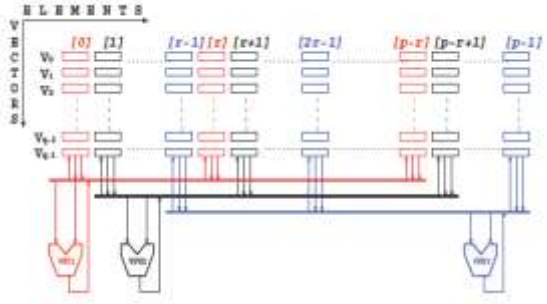

Figure.2: Vector Processing Unit

IV. DISTRIBUTION OF THE VECTOR REGISTERFILE ACROSS VECTOR PROCESSING UNITS (VPUS)

A vector instruction is meant to replace "software loop" where the data being operated on are independent from each other and where the calculation of each iteration of the loop is independent from the calculation of the 'adjacent' iterations. By looking at some of the instructions in Appendix A1, we can see that operations like VADDU do not obey this rule. For such instructions, we take advantage of the fact that the calculation on each element of the vector is only 'partially' independent from that of its neighbors. We hence define the GIVI (Genuinely Independent Vector

A.Coprocessor system and architecture

VCP consists of a local data memory (DMEM), an instruction memory (IMEM), a vector register file (VRF), a scalar register file (SRF), a predicate register file (PRF), a scalar processing unit, a vector processing unit, and a load/store unit. The scalar processing unit is a 16-bit 3-way VLIW processor. This unit executes 16-bit scalar arithmetic and logic instructions, vector and scalar load/store instructions, data transfer instruction via SRF, and branch/jump instructions and controls data transfer to the vector processing unit and program flow. The vector processing unit includes three asymmetric vector processing pipelines; each pipeline is identified by the index A, B, and C. The vector processing unit has 991 operators in total. There is an instruction queue for each vector processing pipeline. The instruction waits in this instruction queue until the execution condition is satisfied. The load/store unit has a queue with three entries to transfer data continuously without pipeline stall and to take advantage of the memory bandwidth. In addition to the linear transfer, the load/store unit supports rectangular transfer for block data.VRF has 64 vector registers that hold sixteen 8-bit data each. VRF has 6-read/3-write ports. SRF is a register file for communication between the scalar processing unit and the vector processing unit. It consists of 32 8-bit registers. The scalar processing unit can store constant values in SRF. The values are transferred into registers of the vector processing unit. Also, an output of the vector processing unit can be stored into a register in SRF. PRF is the predication register file. It consists of 64 registers that hold 16 1-bit data, each corresponding to 16 sub-words of SIMD data. PRF is used to store a condition flag used for the select operation of the vector processing unit. The scalar processing unit can store the 16-bit data in a register in PRF directly.

1) Vector Pipeline Processing: Fig. 2 shows the overall vector processing pipelines A, B, and C. The vector processing unit includes three asymmetric vector processing pipelines that consist of the following five stages of 16-parallel SIMD ALUs.

- **EX0:** data permutation such as byte alignment and unpack operations.
- **EX1:** first stage of ALU and shift operations.
- **EX2:** second stage of ALU and multiplication (only pipeline A) operations.
- **EX3:** accumulation stage.
- **WB:** write back to VRF with data reformatting.

Vector processing pipelines B and C have 3-parallel datapaths at the EX0, EX1, and EX2 stages. The accumulators are implemented in all the vector processing pipelines at the EX3 stage. Furthermore, the vector processing unit also includes a horizontal addition operator. The input of this operator is connected to all the accumulators of the vector processing pipelines and the results are stored into SRF. We assume that the input image data is 8-bit pixel and the input data of a vector processing unit is 8-bit SIMD for the first product instance. Because it is necessary to guarantee operation accuracy in the intermediate stages, operation bit width in- creases as the stage advances. For instance, in pipeline A, the result of the EX1 stage is signed 10 bits and the result of multiply in the EX2 stage is signed 17 bits, and the bit width of the accumulator is 21 bits.VRF has six read ports; one of them is used for vector load/ store, two of them are assigned to vector processing pipeline A, and the other three are assigned to vector processing pipeline B. No specific read ports are assigned to vector processing pipeline C. Instead it shares the three read ports of VRF with vector processing pipeline B in order to reduce the area of VRF. This port sharing technique yields a restriction, namely, vector LIW instruction C is executed simultaneously with vector LIW instruction B. For instance, it enables mapping of 2-D filter operations, vertical and horizontal directions, onto pipelines B and C simultaneously. To reduce the area, only pipeline A has multipliers. In- stead pipelines B and C support shift and add operators to realize constant multiplication that appears frequently in image processing. In the EX0 stage, pipelines B and C execute byte-shift operation on three SIMD input data independently and output three pairs of permutated SIMD data. Each pair of SIMD data is sup- plied to three parallel data paths.

V. EXPERIMENTAL RESULTS

A. Hardware Implementation

To evaluate the performance, we implemented VCP using HDL, synthesized it, and evaluated its frequency. Total gate count is 1268 kgates. The maximum frequency that is a result of STA (Static Timing Analysis) after P&R (Place and Route) is 300 MHz for 90-nm CMOS process. The peak performance of VCP is

estimated to be about 270 GOPS. We evaluate power consumption by gate-level simulation, which is synthesized using Power Compiler and Toshiba ASIC library (90 nm process, high-speed cell). The measured power consumption of the edge detect program is 173 mW at 300 MHz with Power Theater, which is the power consumption before special RTL optimization for power reduction. Fig. 16 shows the layout result of VCP and Table I summarizes the features of VCP's hardware implementation. Table II shows the gate counts of units in VCP.

B. Tool Chain and Evaluation Environment

We developed the compiler, the assembler, and the simulator for VCP. The tool chain of VCP. The VCP C compiler performs optimization to allocate the pipelines and the registers in VRF. Furthermore, the compiler analyzes data de- pendency and inserts a synchronization instruction to resolve data hazards between the vector LIW instructions statically. For the parallel execution in pipelines B and C and the pipeline chaining, programmers have to write program directives explicitly.. Furthermore, programmers have to write programs according to VCP C program style that consists of individual descriptions of each operator .The description violating the style invokes compile error. Then, programmers have to write in assembly language. We have developed the VCP Simulator for not only software development but also verification of RTL. Also, we have developed the FPGA evaluation system for VCP. Adaptive pre-filter processing operates on this system.

VI. PARALLEL IMPLEMENTATIONS OF CRYPTOGRAPHY

In the 'conventional' or non-embedded computing world, most of the research has concentrated around parallelizing the cryptographic operations in order to take advantage of the SIMD architecture originally developed for media applications: In the authors implement a long precision modular Multiplication on a Pentium4 using the SSE2 (*Streaming SIMD Extensions 2*) instructions. The authors execute four exponentiations in parallel, each exponentiation being implemented using a Redundant Representation of Montgomery's multiplication. The authors report that a 1024-bit modular multiplication takes 60μs, which roughly corresponds to 120000 clock cycles for a 2GHz Pentium4 processor.

• Crandall and Klivington illustrate in how the Velocity Engine of the PowerPC can be used to implement long precision multiplications for RSA. we can infer that a 1024-bitMultiplication takes about 3600 clock cycles with their approach. However, no figures were reported for a full modular multiplication.

• The AltiVec extension to the PowerPC was originally developed to target media applications. This vector extension is made of 32 128-bit vector registers. AltiVec also offers some superscalar capabilities since instructions belonging to different 'classes' can be executed in parallel. Galois Field arithmetics has been implemented on the AltiVec in . In the latter paper, the authors show how the Rijndael algorithm can be made to execute in 162 clock cycles on the AltiVec or, even better, in only 100 clock cycles if a bit-sliced approach is used. For embedded applications, studies around the use of SIMD architectures for cryptography are even more scarce:

Comparison With Conventional VLIW & SIMD Processor:

We compared the performance of VCP with that of our previous work, a 3-way 64-bit VLIW processor [11], which is one of the typical processor architectures with SIMD instructions and de- signed for image processing and image recognition. The VLIW processor has two 64-bit SIMD datapaths. One of the processor datapaths consists of only an 8-parallel SIMD ALU and an accumulator, whereas the other has an 8-parallel SIMD MAC (Multiply and Accumulator) as well. The area of the VLIW processor is 520 kgates including 8-KB instruction cache and 8-KB data cache. This gate count is used for comparison of architecture efficiency such as normalized performance by area.

Comparison With DSP Processors:

We compared the performance of VCP with that of DSP processors. Table V shows the performance.

VII. CONCLUSION

We introduced a VLIW vector media coprocessor, VCP, with asymmetric cascaded SIMD ALUs to exploit the loop-level parallelism. To achieve high performance with small hard- ware size, we reduce the area ratio of the control circuit while increasing the ratio of the arithmetic circuit. In particular, to execute instructions out-of-order with a small control unit, the compiler for static optimizations of synchronization is developed. To evaluate the performance of the proposed architecture, we measured the processing cycles for the calculation of edgeness and SAD by the simulator. we showed that high performance can be achieved through parallel computation of cryptography on a vector architecture. It is well known that instruction level parallelism is very expensive in terms of hardware and in particular very complex in terms of instruction decoding and scheduling. On the other hand, taking a vector approach is a relatively cheap way of achieving high performance parallelism as most of the logic goes into the data path and not in the control path.

REFERENCES

[1]. S. Kyo, S. Okazaki, and T. Arai, "An integrated memory array processor architecture for embedded image recognition systems," in Proc.Int. Symp. Computer Architecture (ISCA), 2005, pp. 134–145.

[2]. B. Khailany, T. Williams, J. Lin, E. Long, M. Rygh, D. Tovey, andW. Dally, "A 512 GOPS stream processor for signal, image and videoprocessing," in Proc. Int. Solid-State Circuits Conf. (ISSCC), 2007, pp.272–273.

[3]. B. Mei, S. Vernalde, D. Verkest, H. D. Man, and R. Lauwereins,"ADRES: An architecture with tightly coupled VLIW processor andcoarse-grained reconfigurable matrix," in Proc. Int. Conf. Field-Programmable Logic and Applications, 2003, pp. 61–70.

[4]. T. Sato, "A dual-core dynamically reconfigurable engine employs 955parallel processing elements," presented at the Microprocessor Forum,2007, unpublished.

[5]. S. Knowles, The SoC future is soft Dec. 2005, IEE Cambridge Pro-cessor Seminar.

[6]. R. Krashinsky, C. Batten, M. Hampton, S. Gerding, B. Pharris, J.Casper, and K. Asanovic, "The vector-thread architecture," in Proc.Int. Symp. Computer Architecture, 2004, pp. 52–63.

[7]. M. Baron, Applications Define Dsp Speed, Microprocessor Rep., Apr.2005, pp. 3–12.