Vector IRAM: ISA and Micro-architecture

Christoforos E. Kozyrakis

Computer Science Division
University of California, Berkeley

kozyraki@cs.berkeley.edu
http://iram.cs.berkeley.edu/
Outline

- Project motivation, goals and approach
- Vector IRAM ISA
- VIRAM-1 micro-architecture
- Project status
Project Motivation

- Processor-memory gap is growing exponentially
- Applications shifting from engineering/desktop to multimedia
  - importance of performance of media functions
  - importance of real-time predictable performance
- Embedded/ portable systems gain popularity
  - importance of energy consumption
  - importance system size
- Focus on processors for portable, multimedia systems
The Vector IRAM Approach

Vector processing
• multimedia ready
• predictable, high performance
• simple
• energy savings
• high code density
• well understood programming model

Embedded DRAM
• high memory bandwidth
• low memory latency
• energy savings
• system size benefits

Serial I/O
• Gbit/sec I/O bandwidth
• low pin count
• low power
Outline

• Project motivation and goals
• Vector IRAM ISA
  – Overview of VIRAM ISA extensions
  – Fixed-point and DSP support
  – Conditional and speculative execution
  – Memory model
• VIRAM-1 micro-architecture
• Project status
Vector Execution Model

**SCALAR**
(1 operation)

$$\text{add } r3, r1, r2$$

**VECTOR**
(N operations)

$$\text{add.vv } v3, v1, v2$$
Vector Architectural State

General Purpose Registers (32)

Flag Registers (32)

Virtual Processors ($vlr$)

Control Registers
Overview of V-IRAM ISA Extensions

### Scalar
MIPS-V scalar instruction set

- **alu op**
  - s.int
  - u.int
  - s.fp
  - d.fp

### Vector ALU

- **alu op**
  - 8
  - 16
  - 32
  - 64

- **.v**
  - .vv
  - .vs
  - .sv

### Vector Memory

- **load store**
  - s.int
  - u.int
  - 8
  - 16
  - 32
  - 64

### Vector Registers

- 32 x VL x 64b data
- 32 x 4VL x 32b data
- 32 x 8VL x 16b data

### Plus:
- flag, convert, fixed-point, and transfer operations

- **All ALU / memory operations under mask**
- Unit stride
- Constant stride
- Indexed

**C.E. Kozyrakis, IEEE Computer Elements Workshop, June 22, 1998**
Fixed-point and DSP support

• GOAL: Competitive DSP performance
• Many DSP features already provided
  – narrow data widths [provided]
  – high speed MACs [instruction chaining]
  – multiple LD/ST per cycle [multiple memory units]
  – auto increment / decrement [strided memory access]
  – zero overhead loops [vector instructions]
  – fixed floating convert [provided]
  – bit reverse addressing [use better FFT algorithm]
Fixed-point Multiply-Add Model

\[ \begin{align*}
\text{Mul} & : x \times y \\
\text{Add} & : z + a \\
\text{Round} & : \begin{cases}
\text{truncate} & \\
\text{round nearest even} & \\
\text{round nearest up} & \\
\text{jam} & \\
\end{cases} \\
\text{F} & : \begin{cases}
\text{signed saturate} & \\
\text{unsigned saturate} & \\
\text{shift by one} & \\
\end{cases}
\end{align*} \]
Fixed-point instructions

- Vector half-width integer multiply
- Vector fixed-point shift and add
- Vector saturate
- Vector saturating left arithmetic shift
Conditional (Predicated) Execution

• Almost every vector instruction is executed subject to one of two vector masks
• 15 GP flag register provided to buffer masks or operate on them
• 6 flag logical and 13 flag processing instructions (like population count, iota etc)
• 15 flag registers used for sticky exception bits for arithmetic/FP operations and speculative operations
Speculative Execution

- Vectorizing loops with conditional exit conditions
  - Need to speculate past loop exit
  - Need to temporarily suppress exceptions
- Speculation controlled by software
- Solution:
  - A duplicate set of arithmetic exception flag registers
  - A flag register reserved for load faults
  - Speculative loads and speculative arithmetic instructions write these duplicate exception bits
Speculative Execution (cont.)

- Perform loads and enough arithmetic to determine loop exit condition
  - Stores cannot be speculated!
- Generate mask to exclude iterations after loop exit (flag processor instruction)
- `VCOMMIT` instruction (under mask):
  - ORs speculative flags into real flags
  - Raises memory exceptions
Memory Model

- Relaxed consistency to simplify hardware: no guarantee about ordering of memory operations, even within the same VP
- Register interlocks provided on a per-element basis
- Vector memory barrier used for ordering between scalar unit and vector unit and between VPs
- Indexed memory operations do not specify ordering; separate ordered indexed store instruction
Outline

• Project motivation and goals
• Vector IRAM ISA
• VIRAM-1 micro-architecture
  – Overview of VIRAM-1 micro-architecture
  – Vector pipelines
  – Memory system architecture
• Project status
VIRAM-1 Block Diagram

[Diagram of VIRAM-1 block diagram with labeled components such as Instruction Cache, Flag Registers, VFFU, VAU, Vector Registers, VMU, I/O, and DRAM MACROs.]

C.E. Kozyrakis, IEEE Computer Elements Workshop, June 22, 1998
VIRAM-1 Features

- **Scalar unit**
  - 64-bit MIPS core with FP unit
  - 8KB I+D caches, write-through
  - cache invalidation interface

- **Vector unit**
  - maximum vector length 32
  - 64, 32, 16 bit data-types
  - 2 vector arithmetic units
  - 2 vector flag processing units
  - 4 pipelines per functional unit
  - 2 vector load/store units
  - 64 entry vector TLB, multi-ported
Vector Pipelines

- Multiple pipelines can increase performance OR
- Energy decrease by decreasing clock frequency and power supply
VIRAM-1 Memory System

- 16 to 32MB DRAM
- 16 independently addressed banks
- 8 2Mbit DRAM macros per bank with 256-bit synchronous interface
- Memory crossbar
  - interconnects scalar, vector unit and I/O to memory
  - 8 addresses per cycle
  - 12.8GB/sec maximum data bandwidth per direction
  - implemented using low-swing techniques
VIRAM-1 Floorplan
# VIRAM-1 Goals

<table>
<thead>
<tr>
<th>Category</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology</td>
<td>0.20 micron, 5 metal layers, embedded DRAM-logic process</td>
</tr>
<tr>
<td>Memory</td>
<td>16-32 MB</td>
</tr>
<tr>
<td>Die size</td>
<td>250-300 mm$^2$</td>
</tr>
<tr>
<td>Vector pipelines</td>
<td>4 64-bit (or 8 32-bit or 16 16-bit)</td>
</tr>
<tr>
<td>Clock Frequency</td>
<td>200MHz scalar, 200MHz vector, 100MHz DRAM</td>
</tr>
<tr>
<td>Serial I/O</td>
<td>4 lines @ 1 Gbit/s</td>
</tr>
<tr>
<td>Power</td>
<td>2 W @ 1.5 volt logic</td>
</tr>
<tr>
<td>Performance</td>
<td>1.6 GFLOPS$<em>{64}$ − 6.4 GOPS$</em>{16}$</td>
</tr>
</tbody>
</table>

First microprocessor above 0.25B transistors?
Scaling Down VIRAM-1

- Scaled-down version automatically generated from the original
- 8 MB in 4 banks
- Vector unit with single pipeline per functional unit => same control
- die: 80 mm$^2$
- transistors: 70M
- power: 0.5 Watts
- performance: 0.4 GFLOPS$^{64}$
  1.6 GOPS$^{16}$
Project Status

- ISA extensions frozen
- Micro-architecture still under development but design has started
- Developing simulation infrastructure
- Designed 2 test-chips for circuit evaluation
  - serial I/O @ 1Gbit/s
  - embedded DRAM and on-chip crossbar
- Expected VIRAM-1 tape-out: early 2000
Acknowledgments

• Thanks for advice/support: DARPA, California MICRO, ARM, Hitachi, IBM, Intel, LG Semicon, Microsoft, Mitsubishi, Neomagic, Samsung, SGI/Cray, Sun Microsystems