



The VIS Instruction Set

Lawrence Spracklen

VIS Engineering

Processor & Network Products

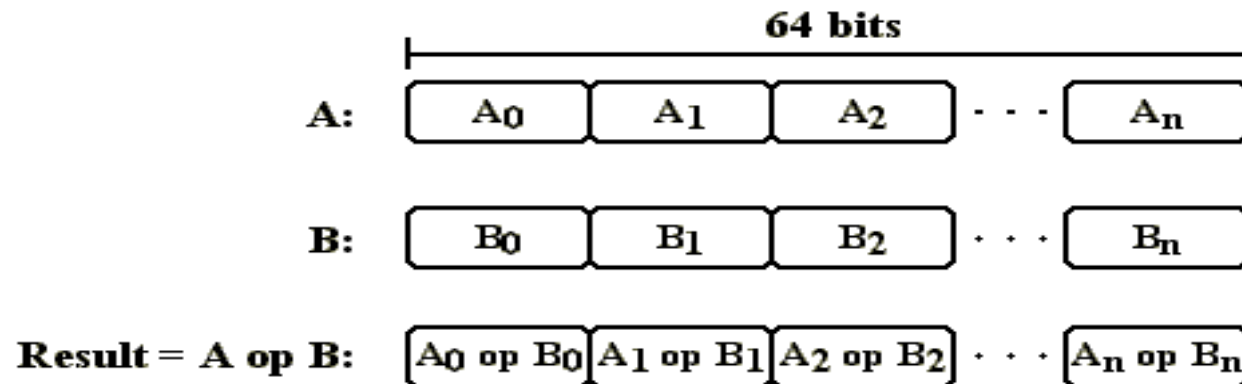


Overview

- What is the VIS instruction set?
- VIS benefits
- Using VIS

What is VIS?

- SIMD (Single Instruction Multiple Data)
 - Registers, datapaths and ALUs now 64-bits
 - Variables frequently only 8, 16, and 32-bits
 - Pack multiple variables into each register
 - Process these 'packed' variables in parallel



- Achievable using standard integer instructions

What is VIS?

- Using standard integer instructions is slow
 - Need to correct for field interactions
- Sun introduced specialized hardware in 1995
- The VIS instruction set is Sun's SIMD instruction set
 - A set of "RISC-style" SIMD instructions
 - An extension to the SPARC V9 instruction set

VIS 1.0

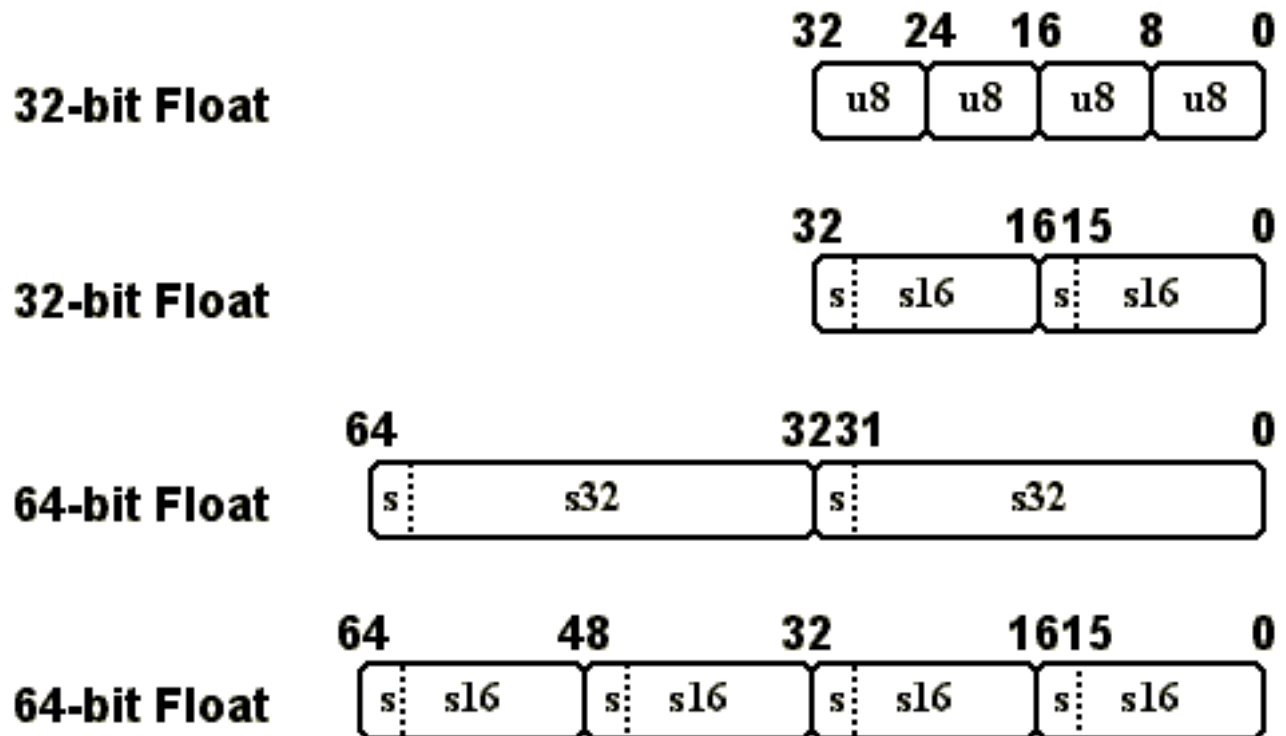
- Introduced with the UltraSPARC I
- Comprised of over 80 instructions
 - SIMD arithmetic & logical
 - SIMD compare
 - PDIST - rapid SAD computations for video codecs
 - SIMD format conversion (8<->16-bits etc.)
 - Data misalignment handling
 - Memory access (8-bit, 16-bit, masked 64-bit & 64-byte loads/stores)
 - Array instructions

VIS 1.0

- Operates on integer/fixed point data
- Provides both 2 and 4-way SIMD operations
- Utilizes the FP pipelines
 - Maximizes instruction level parallelism
 - Maximizes # of useable registers
- VIS facilitates
 - 2 VIS instructions can be dispatched each cycle
 - VIS instructions are fully pipelined
 - VIS instructions have short latencies

VIS 1.0

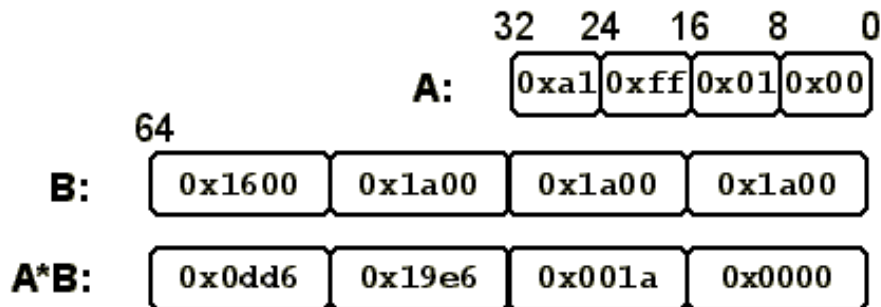
- VIS packed data types
 - VIS handles 2 and 4-way partitioned 32-bit and 64-bit variables



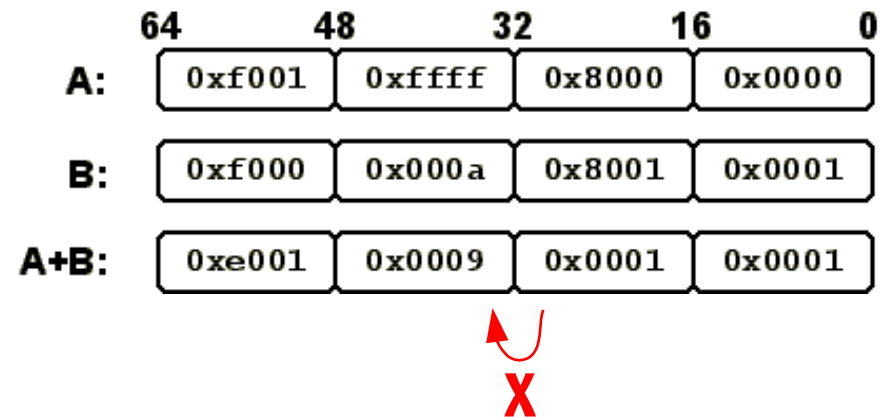
VIS 1.0

VIS Arithmetic instructions

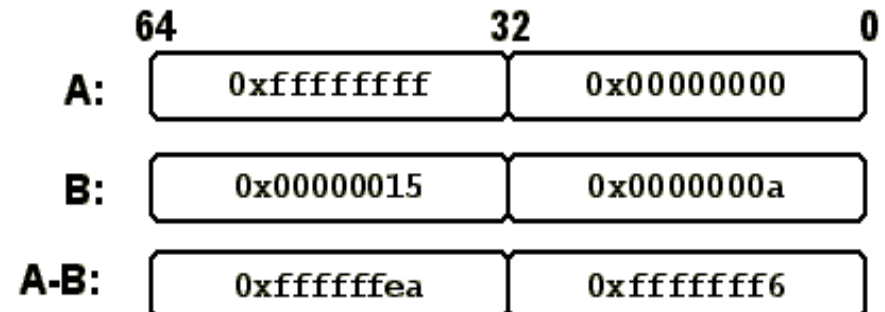
VIS 8x16-bit Multiplication



VIS 16-bit Addition



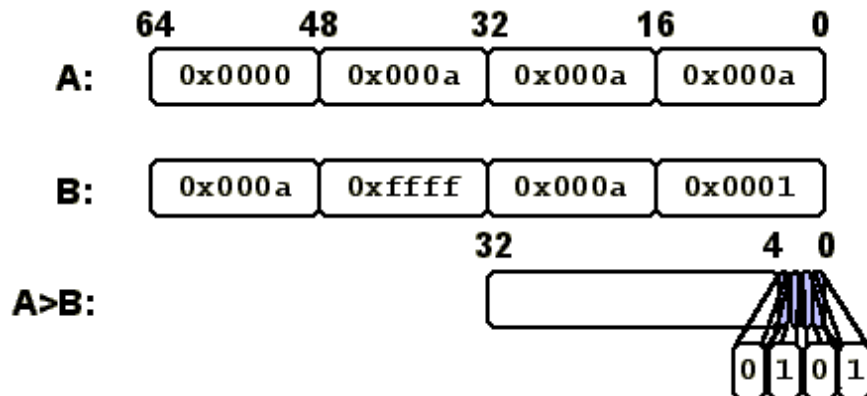
VIS 32-bit Subtraction



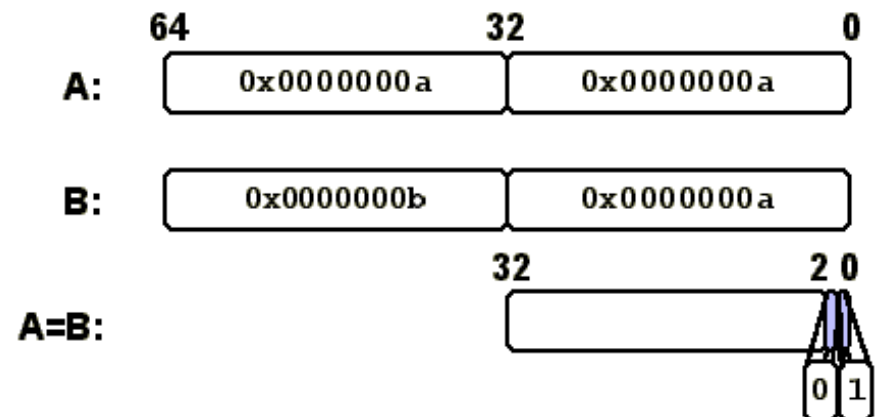
VIS 1.0

- VIS compare instructions
 - Results returned to the integer registers
 - Significant advantage over other SIMD ISEs

VIS 16-bit compare

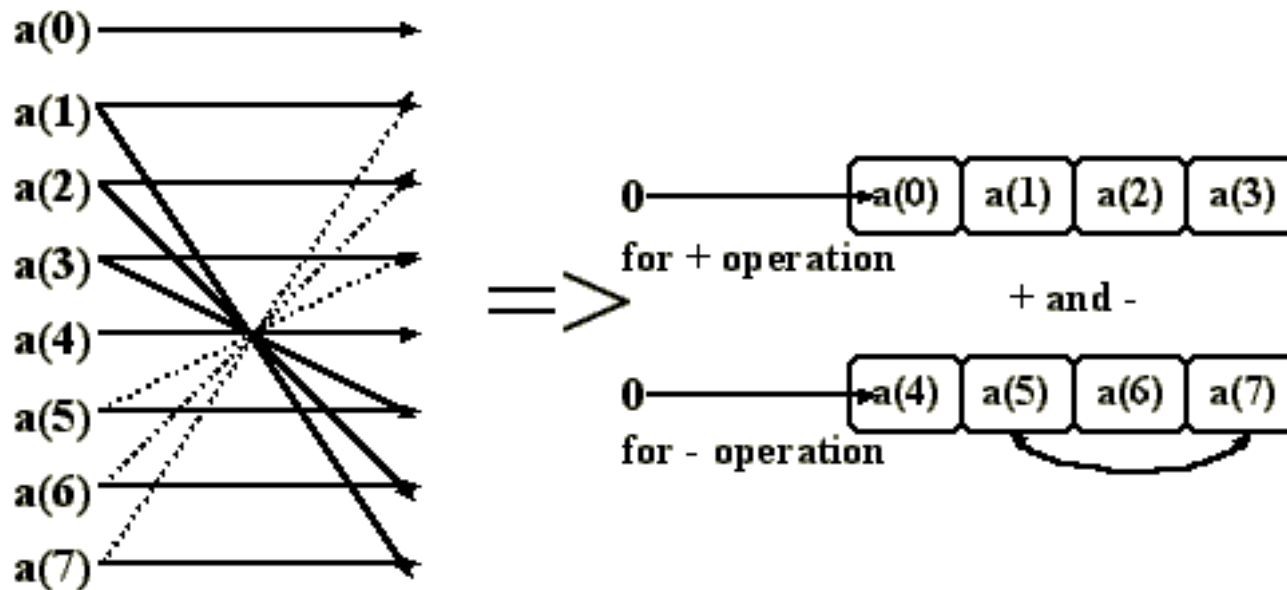


VIS 32-bit compare



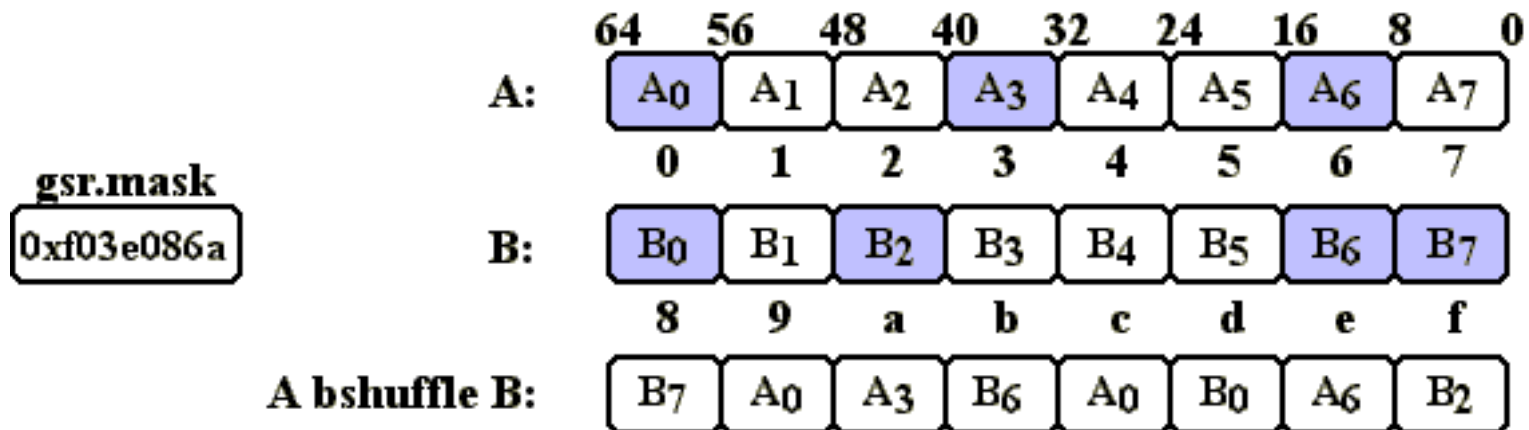
VIS 2.0

- Problem:
 - Ordering of packed data elements is frequently incompatible with SIMD processing



VIS 2.0

- VIS 2.0 provides instructions for enhanced data shuffling
 - Allows VIS to efficiently handle incompatible data organizations
- Introduced on the UltraSPARC III



Competitive Analysis

- VIS was the 1st
- Now there are many
 - Intel: MMX and SSE for x86
 - Motorola: AltiVec
 - HP: MAX
 - Many DSPs also have SIMD e.g. TigerSHARC
- IBM's Power-4 doesn't!

| Features | VIS | SSE | AltiVec |
|------------------------|------------|----------|----------|
| Variable size | 64-bits | 128-bits | 128-bits |
| SIMD FP | in VIS 3.0 | yes | yes |
| # of registers | 32 | 8 | 32 |
| # of pipelines | 2 | 1 | 1 |
| Instructions Pipelined | yes | sort-of | yes |

VIS Performance

- VIS provides performance gains of up to 5X

| Algorithm | Performance gain, % | Algorithm | Performance gain, % |
|-------------------|---------------------|-------------------|---------------------|
| Fletcher checksum | 600 | 90° image rotate | 500 |
| IDCT | 300 | Sequential search | 400 |
| MPEG motion est. | 500 | Image blending | 900 |
| Vector dot prod. | 200 | Ray casting | 143 |

- Allows UltraSPARC to perform 10 arithmetic operations/cycle
 - 2 4-way VIS operations and 2 integer instructions
- VIS/Integer hybrids are possible

VIS Applications

- Originally targeted at multimedia applications
 - MPEG-2 decoder 100%
 - JPEG decoder 126%
- Also benefits a much wider set of applications
 - Bioinformatics
 - DSP
 - Cryptography
 - 3D Visualization
 - Networking
 - Telecom
 - Graphics
 - Imaging
- VIS 3.0 will continue to expand this range of applications

Using VIS

- SIMD opportunities can't be readily identified by compilers
 - Efforts frustrated by a myriad of potential data type, data organization, and data alignment issues
 - Only simple loops are currently automatically optimized
- Developer needs to expose and exploit available parallelism
 - Sun provides support to allow VIS to be leveraged from C/C++
- Sun simplifies development process by providing VIS optimized libraries for common functions

Coding VIS

- Sun provides inline macros to allow VIS to be utilized from C/C++
 - Provided in the VIS Software Developers Kit (VSDK)
 - Just assert -xvis during compilation when using the latest SunONE compilers

Standard (SISD)

```
int
searchArray(short *key, short *dataArray, int size)
{
    int i;

    for (i = 0; i < size; i++)
        if (key != dataArray[i]) break;

    if (i >= size)
        return (-1);
    else
        return(i);
}
```

VIS (using inline macros)

```
int offset[16] = {0,3,2,2,1,1,1,1,0,0,0,0,0,0,0,0};

int
searchArray(short *key, short *dataArray, int size)
{
    int i, sr0;
    vis_d64 *dataArray64, key64;

    dataArray64 = (vis_d64 *) dataArray;

    key64 = vis_bshuffle(vis_ld_ul6(key), vis_fzero());

    for (i = 0; i < size; i+=4)
        if ((sr0 = vis_fcmpeql6(key64, dataArray64[i]) != 0) break;

    i = i + offset[sr0];
    if (i >= size)
        return (-1);
    else
        return(i + offset[sr0]);
}
```

mediaLib™

- mediaLib is Sun's VIS library
- Makes using VIS fast and easy
- Callable from C, C++ and Java
- Contains over 3000 functions
- VIS enhanced
- UltraSPARC tuned
- FREE - download from sun.com
 - Starting with Solaris 10, will be bundled with Solaris

mediaLib™

- Current mediaLib coverage

| <i>mediaLib Components</i> | | | | | |
|------------------------------|----------------------------------|---|-----------------------------|-------------------|-----------------------------|
| <i>Algebra</i> | <i>Graphics</i> | <i>Imaging</i> | <i>Signal</i> | <i>Video</i> | <i>Volume</i> |
| Matrix operations | Draw & fill circles and ellipses | Basic arithmetic: add, ave., blend..... | Basic DSP: FFT, FIR, IIR... | DCT/IDCT | Maximum intensity searching |
| Vector operations | Draw & fill polygons | convolution & auto-correlation | ADPCM | Motion estimation | Ray casting functions |
| Data manipulation operations | Draw & fill lines & arcs | Max, min, thresholding | Windowing | Color conversion | |
| | | Rotate & zoom | Convolution Correlation | Interpolation | |

- Search/sort component to debut soon

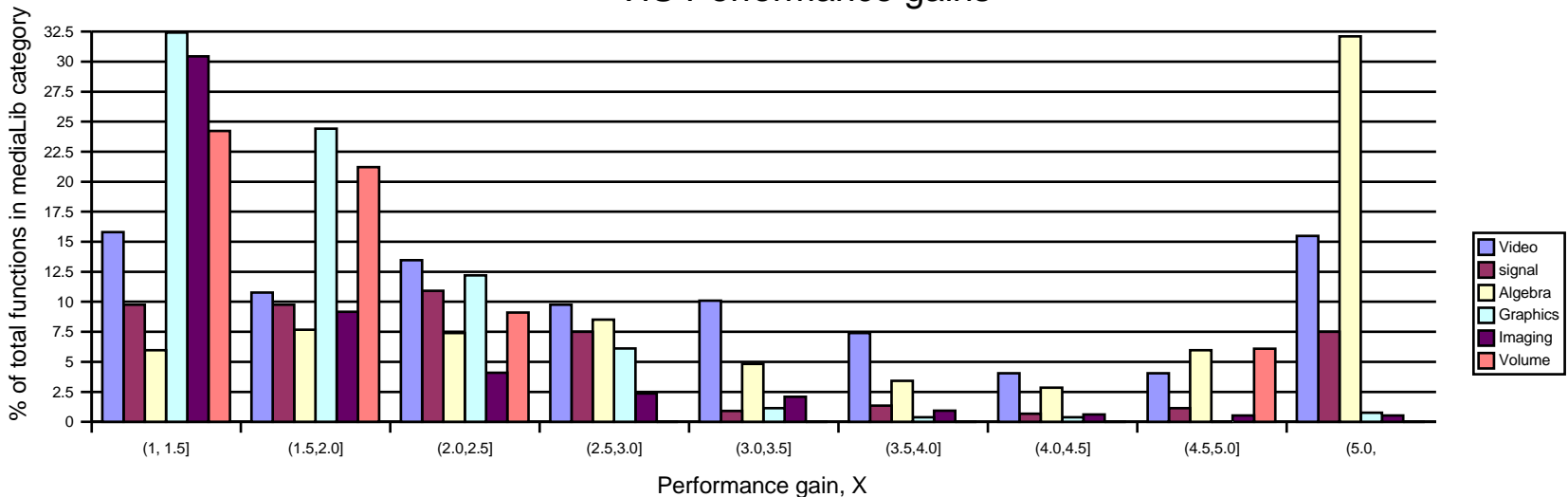
VIS & Java

- Sun provides Java wrappers to allow mediaLib functions to be leveraged from Java
- On UltraSPARC platforms, many Java libraries are accelerated using VIS
 - Java 2D
 - JAI (Java Advanced Imaging)
 - JMF (Java Media Framework)
- Java Native Interface (JNI) allows any VIS optimized code to be leveraged from Java
 - Useful for accelerating small portions of time-critical code

Conclusions

- VIS/mediaLib provide significant performance gains
- VIS is applicable to a wide variety of applications
- VIS is still expanding
- Sun provides the tools to make leveraging VIS easy

VIS Performance gains



Useful links

- VIS Whitepaper
http://www.sun.com/processors/whitepapers/vis_wp_external.pdf
- VSDK & mediaLib
<http://www.sun.com/sparc/vis/>
- VIS - a developer's guide
To be completed



Lawrence Spracklen

lawrence.spracklen@sun.com

