

CSSE3001
CSSE7000

Lecture 9

Extended Instruction Sets

School of Information Technology and Electrical Engineering
The University of Queensland

Previously...

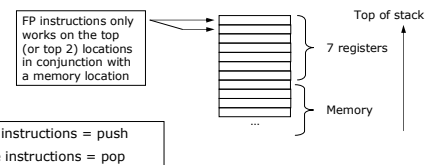
- Recall from previous lectures,
 - Familiar with the MIPS datapath, and its 5-stage pipeline.
 - Discussed Intel & SPARC microprocessor architectures.
 - Primary instruction sets for these architectures.
- Improvement in manufacturing, yield, number & size of transistors
- New applications demand
 - Extending the instruction sets

Today

- In depth look at the evolution of the Math Co-processor for Intel x86.
- VIS instruction Set for UltraSPARC.
- Example application.
- Discussion on general trend, success, support and use of extended instructions.

Intel 8087 Co-processor

- Intel 8087 FP coprocessor was announced in 1980.
 - Extended the 8086 processor with 60 floating point instructions.
- Unlike the 8086, it is based on a "stack" architecture.
 - It is also a physically separate chip located on a different part of the PCB.

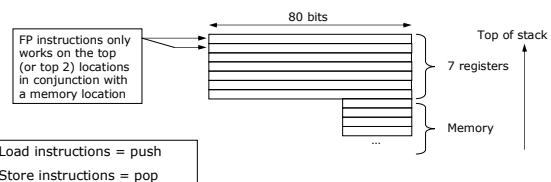


80287, 80387, 80487 coprocessors

- What are the reasons for having a separate chip?
 - Economics
 - Software Requirements
 - Technology
- 80486DX processor has integrated the co-processor on the same die
 - 486SX & DX saga & Intel marketing strategies.

IA-32 Floating Point Architecture (Part 1)

- Up to Pentium & PII MMX model.
 - The Internal 80-bit format (double extended precision) is not supported by programming languages.



Ch 3.7 P. 218-220

IA-32 Floating Point Instruction Classes

- Load/Store = Push/Pop
 - convert 32-bit or 64-bit FP to the internal size automatically.
 - convert integers to FP (and vice versa) automatically.
- Arithmetic: + - x ÷ √ | |
- Comparisons
- Transcendental: sin cos log e^x

Fig 3.21 & 3.22

IA-32 Floating Point: Before MMX era

- One operation per clock cycle
- One FP number per operation (ie. 1 source & 1 destination)
- Based on the same stack architecture as original co-processor

IA-32 Floating Point: MMX in Pentium & PII

- SIMD format on multiple short data elements.
 - Additional 57 instructions
- Still based on the stack architecture
- No other hardware vendor in the market has used this FP architecture
- MMX support is quite rare and performance is below average

IA-32 Floating Point: SSE in PIII

- SIMD format on multiple data elements.
 - Additional 70 instructions
- Still based on the stack architecture
 - Added 8 separate SSE registers (128-bits wide)
 - Cache Prefetch Instructions
 - Streaming Store direct to memory (bypass cache)

IA-32 Floating Point: SSE2 in P4

- The SSE registers are now "normal" floating point registers like the ones in MIPS.
 - The stack is still part of the processor, but hardly anyone uses it.
- Additional 144 instructions (mostly overlap with MMX & SSE, but changed for the "new" SSE registers).
- Floating point performance is now on-par with other processors.

Extending the instruction set: SSE3, AMD64 & EM64T

- AMD64 (AMD)
 - Increase to 16 SSE registers
 - All register size is now 64 bits
 - Hence address space is 64 bits
- EM64T (Intel)
 - 1 extra instruction to AMD64
- SSE3
 - Extra 13 instructions
 - Graphics operations on array of structures
 - Video Encoding
 - FP Conversion
 - Thread Synchronisation

SPARC Floating Point: Before UltraSPARC

- SPARC design is based on RISC architecture from the beginning.
- Naturally, the Integer Unit and the Floating Point Unit have similar arrangements.
- So compiler can use the FP hardware just as easily as the integer hardware.
- Hence, SPARC was commonly used for intensive computations, simulations and modelling applications.
 - More often than x86 processors at the time.

UltraSPARC Processors

- Move to 64-bit architecture, while it is binary compatible with 32-bit code.
- Extra floating point registers
 - Total to 32 FP registers.
- Extended instruction set to handle images, audio, video, multimedia
 - VIS (Visual Instruction Set) – SIMD

VIS

- Format conversions
- Arithmetic operations
- Logical operations
- Address handling for misaligned data
- Array instructions
- Memory access instructions
- Pixel distance calculations

SUN VIS Sample Applications

- Imaging
 - Resampling
 - Thresholding
 - Blending
- Graphics
 - Texture Mapping
- Audio
 - FIR
- Video
 - Motion Vector Estimations

What is the common theme?

- Operate on many pixels simultaneously;
- Memory accesses can be aligned and partitioned;
- Minimal data dependency;
- Operations (eg. Arithmetic or logic) are uniform and “preset” for all algorithms in this area.
 - Eg. MAC
- Advantages on libraries and example, and rely on “in-line” substitutions to interface hardware.

vis_fpadd32() and vis_fpsub32()

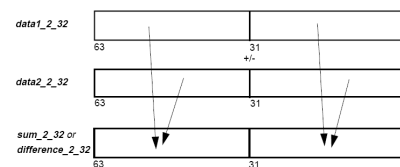


Figure 4-8 vis_fpadd32() and vis_fpsub32() operation

Intel's Streaming SIMD Extension Instruction Set

- Arithmetic
- Square Root
- Approximation
- Min/Max
- Moving data, masking
- Compare & Logical
- Conversion

Intel SSE on FP

- "Packed" (or vector) and scalar variants.
- 4 single precision FP numbers in a 128bit register.
- Lookup table for reciprocal & square root approximations, in addition to "normal" ones.
- Shuffle instruction – copy 2 singles from 2 registers or 4 singles from 1 register.
- Unpack – get either the higher (or lower) 2 singles from 2 registers and put into 1 register.

Intel SSE on Integer

- 64 bit MMX register instead of 128 bit register

SSE on Memory

- Streaming store (write direct to memory)
 - Why? – do not "pollute" cache.
- Masked bytes to memory
- Prefetch control
 - To L1 only, or to L2 only, or to both.
- Store Fence – blocks later stores (only) to maintain ordering.

Discussion

- Compiler support issues
 - Library
- Knowledge of hardware
 - Almost writing in assembly
 - Portability
- Specialisation
 - How do you use VIS/SSE for other applications?
 - Limited parallelisation – limited by register size.
 - Limited in control – program flow is not part of the extended instruction set. This does NOT promote concurrent processing.
 - Vendor specific.

Summary

- Evolution of FP hardware
- Intel MMX, SSE sets
- Sun VIS
- More involved than the standard set
- Limited purpose rather than "general purpose"