Worldwide Competitiveness

Microprocessors vs. DSPs:
Fundamentals and Distinctions

Workshop Outline

- Definitions
- DSP Algorithms Shape DSPs
- Comparing DSPs and GPPs
- Comparing Performance
- When to Use Which
- Conclusions
Microprocessors vs. DSPs:
Fundamentals and Distinctions

Definitions

Microprocessors—General-Purpose Processors (GPPs)
- CPUs for PCs and workstations
  - E.g., Intel Pentium III
- 32-bit GPPs for embedded applications
  - E.g., ARM ARM7

Digital Signal Processors (DSPs)
- Microprocessors specialized for signal processing applications

Basic DSP/GPP
- Architectures targeting extremely cost sensitive markets, often older architectures

High Performance DSP/GPP
- Architectures that use advanced techniques to improve parallelism, performance
  - Usually have higher clock rates

DSP Algorithms Shape DSPs

How Signal Processing is Different From Other Tasks

- Very computationally demanding
- Requires attention to numeric fidelity
- High memory bandwidth requirements
- Streaming data—and lots of it
- Predictable data access patterns
- Execution-time locality
- Math-centric
- Real-time constraints
- Standards: algorithms, interfaces
### DSP Algorithms Shape DSPs

<table>
<thead>
<tr>
<th>Computational demands</th>
<th>→ Multiple parallel execution units, hardware acceleration of common DSP functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numeric fidelity</td>
<td>→ Accumulator registers, guard bits, saturation hardware</td>
</tr>
<tr>
<td>High memory bandwidth</td>
<td>→ Harvard architecture, support for parallel moves</td>
</tr>
<tr>
<td>Predictable data access patterns</td>
<td>→ Specialized addressing modes, e.g., modulo, bit-reversed</td>
</tr>
</tbody>
</table>

© 2004 Berkeley Design Technology, Inc.

---

### DSP Algorithms Shape DSPs

<table>
<thead>
<tr>
<th>Execution-time locality</th>
<th>→ Hardware looping, streamlined interrupt handling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Math-centricity</td>
<td>→ Single-cycle multiplier(s) or MAC unit(s), MAC instruction</td>
</tr>
<tr>
<td>Streaming data</td>
<td>→ No data cache; DMA</td>
</tr>
<tr>
<td>Real-time constraints</td>
<td>→ Few dynamic features, on-chip RAM instead of cache</td>
</tr>
<tr>
<td>Standards</td>
<td>→ 16-bit data types; rounding, saturation modes</td>
</tr>
</tbody>
</table>

© 2004 Berkeley Design Technology, Inc.
## Comparing DSPs and GPPs

### Architecture Type

#### Basic DSP and GPP
- **Single-issue**
  - **DSP**
    - Compound instructions perform multiple operations, e.g., multiply + load + modify address register
  - **GPP**
    - RISC instructions perform single operation, e.g., add, load, or store

#### High-Performance DSP and GPP
- **Superscalar or VLIW**
  - **DSPs typically VLIW**
    - Up to 8 instructions/cycle
    - E.g., TMS320C64xx, SC140, TigerSHARC
  - **GPPs typically superscalar**
    - Up to 4 instructions/cycle
    - E.g., PowerPC 74xx
  - Both classes usually include SIMD instructions
  - Both classes may include dynamic features, but more common on GPPs

---

## Comparing DSPs and GPPs

### Trade-Offs: Superscalar vs. VLIW

**Superscalar (High-performance GPPs, mostly)**
- Increased hardware complexity
  - Silicon area, power consumption
- Dynamic behavior
  - Complex performance model, timing variability
- Increased performance with binary compatibility
- Decreased software complexity (programmer/compiler)

**VLIW (High-performance DSPs, mostly)**
- Decreased hardware complexity
- No dynamic behavior
- Binary compatibility difficult
- Increased software complexity
## Comparing DSPs and GPPs

### Data Path

#### Basic DSP
- Dedicated hardware performs all key arithmetic operations in 1 cycle
- Hardware support for managing numeric fidelity:
  - Shifters
  - Guard bits
  - Saturation
  - Rounding modes

#### Basic GPP
- Multiplies often take >1 cycle
- Multi-bit shifts often take >1 cycle
- Saturation, rounding typically take multiple cycles

### High-Performance DSP
- Up to 6 arithmetic units
  - Extensive SIMD support in some cases
- Some specialized arithmetic units
  - E.g., MAC unit, Viterbi unit
- Limited bit-manipulation capabilities
  - But good support for block floating-point

### High-Performance GPP
- 1-3 arithmetic units
  - Extensive SIMD support in many cases
- General-purpose arithmetic units
  - E.g., integer unit, floating-point unit
- May have superior bit-manipulation capabilities
  - But limited support for block floating-point
Comparing DSPs and GPPs

SIMD Features

<table>
<thead>
<tr>
<th>Basic DSP and GPP</th>
<th>High-Performance DSP and GPP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very limited SIMD features in basic DSP</td>
<td></td>
</tr>
<tr>
<td>E.g., dual add, subtract of 16-bit fixed-point data</td>
<td></td>
</tr>
<tr>
<td>No SIMD support in basic GPP</td>
<td>Limited to extensive SIMD features in high-end DSPs</td>
</tr>
<tr>
<td>E.g., TigerSHARC</td>
<td>4 x 32-bit float</td>
</tr>
<tr>
<td>4 x 32-bit integer</td>
<td>8 x 16-bit integer</td>
</tr>
<tr>
<td>16 x 8-bit integer</td>
<td></td>
</tr>
<tr>
<td>Extensive SIMD features in high-end GPPs</td>
<td></td>
</tr>
<tr>
<td>E.g., PowerPC 74xx</td>
<td>4 x 32-bit float</td>
</tr>
<tr>
<td>4 x 32-bit integer</td>
<td>8 x 16-bit integer</td>
</tr>
<tr>
<td>16 x 8-bit integer</td>
<td></td>
</tr>
</tbody>
</table>

SIMD: Challenges

Each instruction performs lots of work
- Data parallelism
Algorithms, data organization must be amenable to data-parallel processing
- May require programmer creativity, alternative algorithms
- Data-reorganization penalties can be significant
Compilers generally don’t use SIMD capabilities
Most effective on algorithms that process large blocks of data
## Comparing DSPs and GPPs

### Instruction Set

<table>
<thead>
<tr>
<th><strong>Basic DSP</strong></th>
<th><strong>Basic GPP</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Specialized, complex instructions</td>
<td>General-purpose instructions</td>
</tr>
<tr>
<td>Multiple operations per instruction</td>
<td>Typically only one operation per instruction</td>
</tr>
<tr>
<td>Poor orthogonality</td>
<td>Good orthogonality</td>
</tr>
</tbody>
</table>

**Example Instructions**
- `mac x0,y0,a x:(r0)+,x0 y:(r4)+,y0`
- `mpy r2,r3,r4`
- `add r4,r5,r5`
- `mov (r0),r2`
- `mov (r1),r3`
- `inc r0`
- `inc r1`

### High-Performance DSP

**VLIW:**
- Simple to moderately-complex instructions
- Moderate orthogonality

**Superscalar and enhanced conventional:**
- Complex instructions
- Poor to good orthogonality

### High-Performance GPP

**Baseline:**
- Simple instructions
- Moderate to excellent orthogonality

**With SIMD extensions:**
- Moderately complex instructions
- Moderate to excellent orthogonality
## Comparing DSPs and GPPs

### Memory Architecture

#### Basic DSP
- Harvard architecture
- 2-4 memory accesses per cycle
- No caches; on-chip SRAM

#### Basic GPP
- Von Neumann architecture
- Typically 1 access per cycle
- Typically use cache(s)

#### High-Performance DSP
- Harvard architecture
- Per cycle accesses:
  - 1-8 instructions
  - ~two 16- to 64-bit data words
- Usually no caches

#### High-Performance GPP
- Harvard architecture
- Per cycle accesses:
  - 1-4 instructions
  - ~two 32- to 64-bit or one 128-bit data word
- Usually use caches
Comparing DSPs and GPPs

Caches: Challenges

Caches work by lowering average access time
- They are effective at doing this in many applications
- But access times vary significantly

Some applications are sensitive to maximum access time (not average)
- E.g., many “hard-real-time” signal processing applications

Signal processing application access patterns tend to be predictable
- Thus, DMA may be preferable to a cache
- Some recent caches provide pre-fetching capability

Comparing DSPs and GPPs

Addressing

Basic and High Performance DSP
- Dedicated address-generation units
- Specialized addressing modes
  - Autoincrement
  - Modulo (circular)
  - Bit-reversed (for FFT)

Basic and High Performance GPP
- Often, no separate address-generation units
- General-purpose addressing modes

© 2004 Berkeley Design Technology, Inc.
## Comparing DSPs and GPPs

### Program Control

<table>
<thead>
<tr>
<th>Basic DSP</th>
<th>Basic GPP and High Performance GPP and DSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware looping</td>
<td>Software loops only</td>
</tr>
<tr>
<td>Interrupts disabled during certain operations</td>
<td>Interrupts rarely disabled</td>
</tr>
<tr>
<td>Limited or no register shadowing</td>
<td>Register shadowing common in GPPs</td>
</tr>
</tbody>
</table>

Latency typically a few cycles  
May support fast interrupts

### Dynamic Features

Dynamic features are used heavily in high-end GPPs to boost performance  
- Superscalar execution  
- Caches  
- Branch prediction  
- Data-dependent instruction execution times

These features are occasionally used in DSPs too  
These features complicate software development for real-time DSP applications  
- Ensuring real-time behavior  
- Optimizing code

© 2004 Berkeley Design Technology, Inc.
Comparing DSPs and GPPs

Dynamic Features

**Basic GPPs and DSPs**

- GPPs:
  - Dynamic caches common

- DSPs:
  - Rarely have dynamic features
  - Small "loop buffer" instruction cache exception

**High-Performance GPPs and DSPs**

- GPPs: Moderate to extensive use of dynamic features
  - Dynamic caches standard
  - Superscalar execution, branch prediction common

- DSPs: Generally avoid dynamic features
  - Dynamic cache is most common dynamic feature
  - Superscalar execution, branch prediction rare

Comparing DSPs and GPPs

Branch Prediction: Strengths and Weaknesses

In many applications, branch prediction is very accurate
- This includes signal processing applications, where most branches are part of for-next loops

Complex branch prediction algorithms introduce timing uncertainty
- It can be difficult to predict whether the prediction will be correct at any given instant

© 2004 Berkeley Design Technology, Inc.
### Comparing DSPs and GPPs

#### On-Chip Integration

<table>
<thead>
<tr>
<th>Basic DSP</th>
<th>Basic GPP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relatively narrow range of on-chip peripherals and I/O interfaces</td>
<td>Wide range of on-chip peripherals and I/O interfaces</td>
</tr>
<tr>
<td>DSP-oriented on-chip integration features</td>
<td>Not DSP-oriented</td>
</tr>
<tr>
<td>• E.g., “autobuffered” synchronous serial port...</td>
<td>• E.g., asynchronous serial port...</td>
</tr>
</tbody>
</table>

Typical complement:
- 2+ buffered synchronous serial ports, six-channel DMA controller, I²C port, 3+ timers, 8 bit I/O pins

### Comparing DSPs and GPPs

#### On-Chip Integration

<table>
<thead>
<tr>
<th>High-Performance DSP</th>
<th>High-Performance GPP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moderate on-chip integration</td>
<td>High-end Embedded:</td>
</tr>
<tr>
<td>Somewhat DSP-oriented</td>
<td>Moderate to extensive on-chip integration</td>
</tr>
<tr>
<td>Generally not tailored to an application</td>
<td>May be oriented to a specific application</td>
</tr>
</tbody>
</table>

**PC CPU:**
- Very little integration
  - L1, L2 caches, bus interfaces, MMU

Significant support/interface hardware required
# Comparing DSPs and GPPs

## Compatibility and Availability

### Basic DSP
- Mostly proprietary architectures
  - I.e., one architecture, one vendor
- Limited (at best) compatibility between successive generations
- Not typically available as licensable core

### Basic GPP
- Many shared architectures
  - I.e., one architecture, several (to many) vendors
- Often binary compatibility between successive generations
- Often available as licensable core
  - E.g., ARM, MIPS

### High-Performance DSP
- Mostly proprietary architectures
  - Exception: SC1xxx
- Limited compatibility between successive generations
  - TMS320C62/64 exception
- Usually not available as licensable cores
  - Exception: SC1xxx, Ceva-X

### High-Performance GPP
- Mostly shared architectures
  - PowerPC, MIPS, ARM, x86
- Usually binary compatibility between successive generations
- Sometimes available as licensable core
  - E.g., ARM, MIPS, SH-5
Comparing DSPs and GPPs

### Development Support

<table>
<thead>
<tr>
<th></th>
<th>DSPs</th>
<th>GPPs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tools</strong></td>
<td>Primitive to moderately sophisticated</td>
<td>Primitive to very sophisticated</td>
</tr>
<tr>
<td><strong>DSP-specific tool support</strong></td>
<td>Good to excellent E.g., cycle-accurate simulators, DSP C extensions</td>
<td>Poor but improving E.g., general lack of cycle-accurate simulators</td>
</tr>
<tr>
<td><strong>3rd-party DSP software support</strong></td>
<td>Poor to excellent</td>
<td>Limited but growing</td>
</tr>
<tr>
<td><strong>Non-DSP 3rd-party software support</strong></td>
<td>Poor Few to moderate RTOS options</td>
<td>Extensive Few to extensive RTOS options</td>
</tr>
<tr>
<td><strong>Links w/other high-level tools</strong></td>
<td>E.g., MATLAB</td>
<td>E.g., GUI builders</td>
</tr>
</tbody>
</table>

Comparing Performance

When evaluating processors for signal processing, application-specific, product-specific considerations dominate

- Relative performance can vary dramatically depending on the benchmark

Vendor performance claims should be viewed skeptically

- “MIPS” = ...
- Benchmarks are a sharp tool

Performance is more than speed

- Cost/perf, energy efficiency, memory use...
Comparing Performance
Basic DSP/GPP FIR Filter (Lower is Better)

- 'C54x (160 MHz)
- ARM7 (simple, 100 MHz)
- ARM7 (optimized, 100 MHz)
- Basic DSPs (75-160 MHz)
- Basic GPPs (33-200 MHz)

Comparing Performance
High Performance DSP/GPP FIR Filter (Lower is Better)

- 'C67x (225 MHz)
- Pentium III (1330 MHz) (L1 cache preloaded)
- Pentium III (1330 MHz) (L1 cache cleared)
- High-end DSPs (225-1000 MHz)
- High-end GPPs (300-2500 MHz)

Includes fixed-point devices
Microprocessors vs. DSPs:
Fundamentals and Distinctions

When to Use Which

**DSP**

- Heavy signal processing requirements
- Limited control processing
- The DSP is incumbent
- Software compatibility between generations not required
- Multi-vendor architecture not desired
- DSP has better integration for application

**GPP**

- Modest signal processing requirements
- Extensive control processing
- Especially if code density and portability are important
- The GPP is incumbent
- Software compatibility between generations required
- Multi-vendor architecture desired
- GPP has better integration for application

Challenges in Using GPPs for Signal Processing Tasks

- Not enough DSP horsepower
  - Usually an issue only for very basic GPPs or very demanding applications
- Limited memory bandwidth
  - Again, mostly an issue for basic GPPs
- Lack of execution-time predictability
- High cost, power consumption
  - True of PC CPU class GPPs
- Few DSP-oriented development tools
  - E.g., lack of cycle-accurate simulators
- Few DSP-oriented software libraries
- Limited on-chip integration in some cases
When to Use Which

Challenges in Using DSPs for Non-Signal-Processing Tasks

Limited data-type agility
  - Focus on 16-bit fixed-point
Momentum of popular GPP architectures
Generally inferior tools (except for DSP-oriented features)
Inferior third-party support for non-DSP tasks
  - E.g., RTOSs
Proprietary architectures

Conclusions

Take-Away Points

Since GPPs and DSPs often have comparable performance, other factors become prominent:
  - Energy efficiency
  - Integration
  - Compatibility, availability
    - Multi-vendor architectures
    - Licensable cores
  - Tools
    - DSP-oriented
    - Other-oriented
  - Software, availability
Conclusions
Will DSP-Capable GPPs Render DSPs Obsolete?

No, but they will pose increasingly strong competition
• In PCs, PDAs, and other CPU-centric devices, CPUs will handle DSP tasks
• In embedded apps, increasingly competent hybrid processors will challenge DSPs

Software infrastructure is key
• DSPs have the advantage for DSP tasks
• GPPs have the advantage for other tasks

For DSPs, the competitive field has become much larger
• Differentiating criteria are changing

For More Information...
www.BDTI.com

Free Information
• Processor benchmarking
  • BDTImark2000™ scores for dozens of processors
• Pocket Guide to Processors for DSP
  • Basic stats on 40+ processors
• White papers/presentation slides on
  • DSP software optimization
  • Processor architectures and performance
• Article reprints on DSP-oriented processors and applications
  • EE Times
  • IEEE Spectrum
  • IEEE Computer and others
• comp.dsp FAQ