Alternatives to DSPs: What and Why?

Presentation Goals

Why consider alternatives?
What types of alternatives are relevant?
Which companies are developing these?
What are the major distinguishing characteristics, advantages, and disadvantages of each type of alternative?
Alternatives to DSPs: What and Why?

DSP Application Needs

Diverse Requirements in Many Dimensions

- Algorithms: type, complexity
  - From 10’s to 10’s of thousands of ops/bit
- Data rates: ~10 orders of magnitude!
- Data types: 1-D, 2-D, precision, range
- User/channel capacity
- Cost, energy, size envelope
- Flexibility
  - Multiple, evolving standards
  - Market windows, product life cycles

Application Needs

Key Considerations

- Throughput, latency
- Energy efficiency
- System cost
  - Chip cost
  - Memory use
  - Size and integration
- Development cost and risk
  - Tools and support
  - Compatibility
  - Installed base
  - Roadmap
  - Shared vs. proprietary architecture

In varying combinations, with diverse algorithms

© 2003 Berkeley Design Technology, Inc.
Alternatives to DSPs: What and Why?

Why Consider Alternatives?

Convergence
- DSP-intensive products increasingly include complex non-DSP functionality

Processing throughput, density
- E.g., 3G wireless computation demands outstripping DSP processor advances

Development
- DSP processor software development infrastructure (e.g., compilers) suffers from significant limitations

Cost
- Desire for integration drives SoC adoption

Energy efficiency

Flexibility

Flexibility vs. Suitability

General-Purpose: GUI, OS, etc.

Customizable Core

App-Specific DSP

DSP

GPP

Cable Modem

ASIC

ASSP

DSL

802.11b

MP3

VoIP

GSM

© 2003 Berkeley Design Technology, Inc.
Alternatives to DSPs: What and Why?

Key Alternatives

GPPs/DSP-enhanced GPPs
Media processors
Massively parallel processors
ASICs
• Licensable cores
• Customizable cores
• Platform-based design
ASSPs
Reconfigurable architectures
• FPGAs
• Reconfigurable processors

GPPs and Hybrids

Attacking from the Top and Bottom

Today, many general-purpose processors have strong DSP capabilities
• High-performance GPPs with DSP enhancements
  • E.g., Pentium 4, PowerPC 7xxx
• Embedded GPPs with and without DSP enhancements
  • E.g., SH3-DSP, XScale
System designers often must choose between a GPP and a DSP
• Many products contain both a DSP and a GPP; eliminating one can reduce cost
• Many GPPs are adding DSP-oriented features
Example: Embedded RISC CPU

PXA250
- 400 MHz, 32-bit RISC with minor DSP extensions
- BDTImark2000™ score: 930
- MPEG-4 decode (simple profile, CIF, 30 fps): 200–240 MHz
- 16-bit SIMD, 32-bit data types benefit media apps
- Predicated instruction execution good for control
- Good development tool support; optimized DSP software components available (e.g., Intel IPP)
- Price $37.30, qty 10k

Embedded RISC CPUs

Strengths and Weaknesses

- Can have strong DSP performance
- Dynamic features complicate programming
  - Complicates optimization & ensuring real-time behavior
- Sometimes, convoluted programming model
- Good tools, generally lack DSP support
- 32-bit GPPs better targets for non-DSP tasks
  - E.g., TCP/IP network stacks
- Multi-vendor architectures more common
- Very good 3rd-party non-DSP software component support
- Compatibility more common
- High integration parts increasingly common

© 2003 Berkeley Design Technology, Inc.
Alternatives to DSPs: What and Why?

Example: PC CPU

VIA Technologies C3
- 1 GHz x86 compatible
- Moderate power consumption, cost
- SSE support for media applications, supports fixed-, floating-point types
- Access to massive x86 3rd-party software, tools base
- Familiar to software, hardware developers
- MPEG-2 decode (D1 @ 30 fps) uses 5-15% of CPU when coupled with VIA Apollo CLE266 chipset
- Price $39, qty 10k (C3 only)

PC CPUs (GPPs)

Strengths and Weaknesses

- High-performance GPPs can implement demanding DSP tasks
  - May be as fast or faster than DSPs...
  - ...but cost & power consumption may be higher
  - Dynamic features complicate optimization, real-time design
- Generally weak on integration
- Sometimes, convoluted programming model
- 32-bit GPPs better targets for non-DSP tasks
  - E.g., TCP/IP network stacks
- Many options for OS, 3rd party application software
- Development tools mature, powerful
  - But typically lack DSP-oriented features
- Compatibility more common
**Are Processors Efficient?**

*The Monarchial Model of Computing*

Steps for performing one basic operation:
- Fetch instruction from memory
- Decode instruction
- Compute address
- Fetch data
  - (Off-chip memory \(\rightarrow\) L2, update cache state)
  - (L2 \(\rightarrow\) L1, update cache state)
  - L1 \(\rightarrow\) registers
  - Registers \(\rightarrow\) arithmetic unit
- **Perform desired operation**
- Write result
  - All of the above in reverse order!!
- Update data pointers
- Update program counter

**ASICs**

Chips designed for a specific end product or group of end products
Designed by the system developer
“ASIC” does not imply an architecture
- Traditionally DSP ASICs have used hard-wired logic with varied architectures
  - Sometimes with proprietary processor cores
- Increasing licensed IP content:
  - Processor cores, accelerators
  - On-chip peripherals, I/O interfaces
  - Buses
- Plus dedicated, custom logic
Alternatives to DSPs: What and Why?

ASICs

Strengths and Weaknesses

† Offers the ultimate in tailored hardware
   † Speed, energy efficiency, cost/performance, ...
   † Integration to match the product requirements

↓ Large development costs and risks vs. off-the-shelf hardware; mask-making costs increasing
   ↓ Iteration is costly and time consuming

↓ Lengthy development cycles
   ↓ Hardware/software integration and whole-chip verification are particularly challenging
   ↓ Hardware/software partitioning typically must be done early

↓ Inflexibility
   ↓ Long, costly development precludes frequent design changes

↓ Complex, costly, unreliable tools

† Vast architectural options

Licensable Cores

At the intersection of DSPs, ASICs, ASSPs

Licensable cores have attained critical mass
Growing importance of SoCs
Growing cost of in-house processor architectures
Expanding core options
   · Including customizable architectures
Licensable cores change the competitive landscape
   · E.g., {LSI Logic + IBM + Broadcom} vs. TI
The recent wave of new core vendors is now receding
**Customizable Cores**

Certain features selectable by the chip designer (e.g., a 2nd MAC unit, cache)
Data path, other features often customizable
Synthesizable HDL description generated
Software tools automatically customized
A good fit for DSP applications:
- Key algorithms known and amenable to acceleration
- Computationally demanding, cost-sensitive, and/or energy-sensitive

Examples: ARC, Tensilica, Improv

---

**Example Profile**

*Group III Fax Modem*

- Viterbi: 46%
- Equalizer: 14%
- Sample Processing: 16%
- Other: 24%

% time spent in each module
Alternatives to DSPs: What and Why?

Example: Tensilica Xtensa

Customizable Cores

Strengths and Weaknesses

↑ DSP application characteristics mean that customization can yield huge gains
  ↑ Speed, energy efficiency, cost/performance,...
  ↑ Requires a very large investment
    ↑ Must design own chip
  ↑ Tools immature
    • Additional layer of complexity in tools
  ↑ Unproven technology
  ↑ Uncertain company/technology roadmaps
↑ Can use any foundry
Alternatives to DSPs: What and Why?

ASSPs
Application-Specific Standard Products

Off-the-shelf, fixed-function chips specialized for an application
Similar to an ASIC in design
- Many architecture possibilities
- May contain one or more processor cores
  - Which may or may not be user-programmable
- May be a SoC with memory, peripherals, special I/O, etc...
- ...or a building-block, like a video decoder
Similar to off-the-shelf processors in business model

Example ASSP: Micronas MDE9500

- High-integration digital TV receiver
- Analog decode, DVB decryption, decode
- On-chip MPEG-2 video decoder
- Interfaces to other DTV components, VCRs, HDD
- Multi-layer software architecture
- Price not disclosed
Alternatives to DSPs: What and Why?

ASSPs

*Strengths and Weaknesses*

- Often very well matched to the application
  - SoCs with extensive integration
  - Architecture tuned for the application
- Ease of use
  - Reduce system development costs
  - Reduce required implementation expertise
- Often inflexible
- Limited differentiation opportunities for system designer
- Usually single-source
- Roadmap often unclear

FPGAs

*Field-Programmable Gate Arrays*

An amorphous “sea” of reconfigurable logic with reconfigurable interconnect
- Possibly interspersed with fixed-logic resources, e.g., processors, multipliers
Potential for very high parallelism

Historically used for prototyping and “glue logic,” but becoming more sophisticated
- DSP-oriented architecture features
- DSP-oriented tools and design libraries
  - Viterbi, Turbo, and Reed-Solomon coders and decoders, FIR filters, FFTs,...

Key DSP players: Altera and Xilinx
**Example: Altera Stratix**

Up to 28 hard-wired “DSP blocks”
- 8x9-bit, 4x18-bit, 1x36-bit multiply operations
- Optional pipelining, accumulation, etc.

Three sizes of hard-wired memory blocks

---

**Altera Stratix**

*High-end, DSP-enhanced FPGAs*

**IP blocks**
- Filters, FFTs, Viterbi decoders, ...
- Nios processor
- Third-party IP, e.g., DMA controllers

**DSP tools**
- Parameterized IP block generators
- Simulink-to-FPGA link
- C+Simulink-to-FPGA design flow

Most family members available now
Prices begin at $170 (1 ku)
## Alternatives to DSPs: What and Why?

### BDTI Communications Benchmark™

<table>
<thead>
<tr>
<th></th>
<th>Motorola MSC8101 (300 MHz)</th>
<th>Altera Stratix 1S20-6 (Preliminary)</th>
<th>Altera Stratix 1S80-6 (Preliminary)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channels</td>
<td>&lt;&lt;1</td>
<td>~10</td>
<td>~50</td>
</tr>
<tr>
<td>Cost (1 ku)</td>
<td>$120</td>
<td>$325</td>
<td>$3,480</td>
</tr>
<tr>
<td>Cost per channel</td>
<td>~$500</td>
<td>~$10</td>
<td>~$50</td>
</tr>
</tbody>
</table>

From BDTI’s report, *FPGAs for DSP.*

### FPGAs

**Strengths and Weaknesses**

- **Massive performance gains on some algorithms**
- **Architectural flexibility can yield efficiency**
  - Adjust data widths throughout algorithm
  - Parallelism where you need it; distributed storage
- **Re-use hardware for diverse tasks**
- **Slow time-to-market compared to DSPs**
- **Cumbersome design flow that’s unfamiliar to most DSP engineers**
  - Suitability for single-channel, low-power, cost-sensitive DSP applications unclear
Alternatives to DSPs: What and Why?

Summary of Alternatives

<table>
<thead>
<tr>
<th></th>
<th>DSPs</th>
<th>GPPs</th>
<th>FPGAs</th>
<th>Custom Cores</th>
<th>ASICs</th>
<th>ASSPs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Design Effort</strong></td>
<td>B</td>
<td>A</td>
<td>D</td>
<td>C</td>
<td>E</td>
<td>A+</td>
</tr>
<tr>
<td><strong>Design Flexibility</strong></td>
<td>E</td>
<td>E</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>E</td>
</tr>
<tr>
<td><strong>Run-time Flexibility</strong></td>
<td>C</td>
<td>B</td>
<td>A</td>
<td>C</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td><strong>Top Speed</strong></td>
<td>D</td>
<td>E</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td><strong>Energy Efficiency</strong></td>
<td>C</td>
<td>D</td>
<td>C</td>
<td>B</td>
<td>A</td>
<td>A</td>
</tr>
</tbody>
</table>

A = Best, E = Worst

Conclusions

DSPs face growing competition from many directions
- GPPs, FPGAs, licensable cores...
Software—not hardware—is often the key
- Performance advantage for DSPs over GPPs and FPGAs is diminishing
- As application complexity increases, development costs and effort shift to software
- Cutting-edge compilers and other tools are critical
There is no ideal processor
- The best processor depends on the application
- Heterogeneous solutions will become more common