Application Needs

Diverse Requirements

- 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
- Speed
- 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

Algorithm Kernel Benchmarks
- BDTI's benchmarks are based on DSP algorithm kernels
  - DSP algorithm kernels are the most computationally intensive portions of DSP applications
- Example algorithm kernels include FFTs, IIR filters, and Viterbi decoders
- Application-relevant algorithm kernels are strong predictors of overall performance

IDCT 39%
Denorm 11%
Window 25%
Other 25%
Benchmark Results
Example: TI TMS320C5409 (160 MHz)

1
Energy Use

1
Memory Use

1
10 ku Price

DSP Software Development
Increasingly Important

- Not like other kinds of SW development. Why?
  - Resource-hungry, complex algorithms
  - Severe cost limitations
  - Numeric fidelity
  - Hard real-time constraints
- Optimization is essential
- Often, specialized and/or complex processor architectures
- Testing challenges
DSP Software Development

Key Considerations

- The bare essentials:
  - Assembler, linker
  - Instruction set simulator
  - Scan-based emulator
  - Code generation, i.e., C compiler
  - Debugging tools
  - Profiling tools

- Increasingly important:
  - Software libraries
  - Real-time operating systems

Cores vs. Chips

- Synthesizable cores
  - Map into chosen fabrication process
  - Speed, power, and size vary
  - Choice of peripherals, etc.
  - Requires extensive hardware development effort

- Off-the-shelf chips
  - Highly optimized for speed, energy efficiency, and/or cost (depends on chip)
  - Limited performance, integration options
  - Tools, 3rd-party support often more mature
Classes of Processors for DSP

- Conventional DSPs
- Enhanced conventional DSPs
- VLIW-based DSPs
- Superscalar DSPs
- DSP-enhanced MCUs, CPUs

Modern Conventional DSPs

- Circa ~1986-1996
- Fixed-point: mostly 16-bit
  - Some 20-, 24-bit
- Floating-point: 32-bit
- 1 instruction/cycle
- 1 MAC/cycle
- On-chip SRAM, serial ports, host port, timers, DMA, ...
- Typically 75-160 MIPS
Case Study: TMS320C54xx
A Conventional DSP

- 16-bit fixed-point DSP
- Issues one 16-bit instruction/cycle
- Modified Harvard memory architecture
- Peripherals typical of conventional DSPs
  - 2-3 synchronous serial ports, parallel port
  - Bit I/O, timer
  - DMA
- Cheap (100 MHz '5402 is ~$5 qty 10K)
- Low power (60 mW @ 1.8 V, 100 MHz)

TMS320C54xx
Data Path
TMS320C54xx

Memory

- Prog/Data ROM
- Prog/Data SARAM
- Prog/Data DARAM

Data Buses
- (2 x 16 bits read, 1 x 16 bits write)

Address Buses
- (4 x 16 bits)

Data Path
- MAC
- ALU
- Shifter

Addr./Data Registers

Addr. Units (2)

Data (16-bit)

Addr. (16-bit to 23-bit)

TMS320C54xx

Strengths and Weaknesses

- Good memory and energy efficiency
- Decent speed
- Good cost-execution
- Useful peripherals...
- ...but limited integration
- Good DSP tools
- Poor support for GPP tasks
- Compatible with ‘C55xx
- Quality, quantity of 3rd-party support is staggering

www.BDTI.com
Conventional DSPs

**Strengths and Weaknesses**

↑ Cheap and fairly memory efficient
↑ Good speed and energy use…
   ↓ …but not fast enough for demanding apps
↓ Limited integration
↑ Good DSP tools and 3rd-party support
↑ Huge installed base (in some cases)
↓ Uncertain roadmaps…
   ↑ …but sometimes compatible with next-generation DSPs
↓ Poor support for non-DSP tasks

Enhanced Conventional DSPs

- Additional execution units
- Complex, compound instructions
- Mixed-width instructions
- Hardware accelerators or execution units for key DSP functions (Viterbi,…)
- Expanded buses
- SIMD operations
- Even more SRAM, on-chip peripherals, I/O interfaces
**SIMD**

**Single Instruction, Multiple Data**

- Splits words into smaller chunks for parallel operations
- Some SIMD processors support multiple data widths (16-bit, 8-bit,...)

---

**SIMD Characteristics**

- Each instruction performs lots of work
  - Algorithms, data organization must be amenable to data-parallel processing
  - Most effective on algorithms that process large blocks of data
- Loss of generality
  - Typically 4-8 elements per loop iteration
- High program memory usage
  - Rearranging data for SIMD processing
  - Merging partial results
- Drawbacks amplified if loops are unrolled for speed
TI TMS320C55xx

- Based on ‘C54xx, but:
  - Two instructions/cycle
  - Two MAC units
- Complex, compound instructions
  - Assembly source code compatible with ‘C54xx
  - Mixed-width instructions: 8- to 48-bit
- Targets 3G handsets, portable audio players, etc.
- Sampling at 200 MHz @ 1.5 V, ~130 mW
- $35 quantity 10K

TMS320C55xx

- Prog/Data ROM
- Prog/Data SARAM
- Prog/Data DARAM
- Instr. Cache
- Instr. Flow Unit
- Prog. Flow Unit

Memory

Instruction Bus (1 x 32 bits)
Data Buses (3 x 16 bits for read, 2 x 16 bits for write)
Address Buses (6 x 24 bits)

Data Path
MAC MAC
ALU
Shifter

Addr. Gen.
Addr./Data Registers
Addr. Units (3)

Instr. Buff.

Same as 54xx
New on 55xx
Enhanced on 55xx

Speed
Energy Efficiency
C55xx
C54xx
C62xx
TMS320C55xx
Strengths and Weaknesses

↑ Good performance on key metrics (speed, power, cost-execution)
↑ Compatible with ‘C54xx
↓ Incompatible with ‘C6xxx
↑ Ample 3rd-party support
↑ Mature tools
↑ A “safe” choice
↓ Convoluted architecture
↓ Poor compiler target
↑ OMAP (‘C55xx + ARM7)

Enhanced Conventional DSPs
Strengths and Weaknesses

↑ Significant improvements in speed, energy use, and memory use…
  ↓ …but still not fast enough for the most demanding apps
↑ Still fairly inexpensive
↑ Better integration
  ↓ …but usually not licensable
↓ Poor support for non-DSP tasks
↑ Good DSP tools and 3rd-party support
↑ Look and feel of earlier generations (and sometimes compatibility)
Multi-Issue Architectures

RISC-Based Approach

- Execute multiple instructions/cycle
  - More parallelism
- Use simple, regular instruction sets
  - Simpler decoding, faster execution
    - Faster clock
  - Better compiler target
- More parallelism, higher clocks $\rightarrow$ faster processors
- Better compiler targets $\rightarrow$ simplified software development

Multi-Issue Approaches

VLIW vs. Superscalar
Our Terminology

- **VLIW**: compile-time scheduling
  - Traditionally used positional instructions, e.g., Philips TriMedia
  - Newer processors have flexible grouping, e.g., TI 'C6xxx, StarCore SC100

- **Superscalar**: run-time scheduling
  - e.g., Intel Pentium III, LSI Logic ZSP400

- Instruction Parallelism vs. Data Parallelism
  - VLIW or superscalar can be combined with SIMD

VLIW-Based DSPs

- Speed-focused
- Independent execution units
- Simple, RISC-like instructions
- Regular, orthogonal instruction sets
- Large, uniform register sets

- VLIW DSPs sometimes feature
  - Deep pipelines, latencies
  - Predicated execution
TI TMS320C64xx
The 'C62xx Gets Serious Enhancements

- 8-issue architecture
  - Dual 16-bit multiplies in each multiplier
  - 8-bit operations for image/video processing
  - Application-specific instructions
- 600 MHz clock speed, but…
  - 11-stage pipeline with long latencies
  - Dynamic caches
- The only DSP family with compatible fixed- and floating-point versions

TI TMS320C64xx
VLIW DSP Processor

2 independent data paths, 8 execution units

L=40-bit ALU
S=32-bit ALU, 40-bit Shifter
M=Multiplier
D=32-bit Add/Sub for Address Generation

On-Chip Program Memory

Dispatch Unit

32x8=256 bits (8 instructions)

On-Chip Data Memory
### TMS320C64xx

**Strengths**
- Very fast, particularly on imaging and SIMD-friendly algorithms
- Compatible with ‘C6xxx family
- Builds on mature ‘C62xx tools
- High level of integration
- Caches reduce execution-time predictability

**Weaknesses**
- Expensive; high memory use
- Deep, complex pipeline
- Tough challenge for programmer, compiler

### VLIW-Based DSPs

**Strengths**
- Increased performance
- Better compiler targets
- Potentially easier to program
- Potentially scalable

**Weaknesses**
- Parallelism must be identified, exploited by programmer or tools
- Often, high program memory use and bandwidth requirements
- Often, higher power consumption
Processors with DSP Capabilities: Which is Best?

Superscalar DSPs

- Resemble high-end CPUs
- Run-time instruction scheduling
  - Possibly other dynamic features, e.g., branch prediction, caches
- Lots of parallelism
- Simple, RISC-like instructions
- Regular, orthogonal instruction sets
- Examples: LSI Logic ZSP400, Lexra LX5280, 3DSP SP-5

LSI Logic ZSP400
A 4-Way Superscalar DSP Core

- 16-bit, fixed-point DSP
- 16-bit RISC-like instructions
  - Up to four dynamically scheduled instructions per cycle
  - Small instruction and data buffers
- Two MAC units, two ALU/shifter units
  - Limited SIMD support
  - MACs can be combined for 32-bit operations
  - ALUs also function as AGUs, shifters
- Licensable synthesizable core; also used by LSI Logic in chips
- LSI402ZX shipping at 200 MHz in 0.18 µm
LSI Logic ZSP400

Strengths and Weaknesses

- Good performance on key metrics (speed, memory, price)
- Chips have poor energy efficiency
  - Core has better energy efficiency
- Poor tool support for dynamic behavior
- Good 32-bit support
- Growing acceptance
- Roadmap to high performance

Superscalar DSPs

Strengths and Weaknesses

- Many of the same advantages, disadvantages as VLIW-based DSPs
- But unlike VLIW,
  - Programmer (or code generation tool) isn't required to schedule instructions
  - Peak performance may be elusive without careful scheduling, though
  - Dynamic behavior complicates DSP software development
  - Ensuring real-time behavior
  - Optimizing code
DSP-Enhanced GPPs, Hybrids

- Nearly all vendors of GPPs (both embedded processors and CPUs) now offer DSP-enhanced versions because
  - Processor workloads shifting to DSP
  - DSPs and GPPs often found together (e.g., in cell phones)
  - Integration is imperative

A Spectrum of DSP Enhancements

Add a separate DSP
- MCore + StarCore

Minor changes to ISA
- R4650
- ColdFire

No change
- PowerPC 604e

Architectural renovation
- DSP-like
  - SH-DSP
  - ARM9E
- SIMD
  - MMX, SSE
  - AltiVec

Coprocessor
- FILU-200
- MPC8xxx

Totally new design
- TriCore
- Hyperstone
Intel's MMX, SSE, and SSE2

- **MMX**
  - Fixed-point: 8x8, 4x16, 2x32, and 1x64
  - Non-orthogonal instruction set
- **SSE and SSE2**
  - Floating-point: 1x32, 4x32, 1x64, and 2x64
  - Eight new 128-bit registers
  - Additional integer MMX operations
  - Relatively orthogonal instruction set
  - No MAC instruction

Intel's MMX, SSE, and SSE2

Strengths and Weaknesses

- **P4 probably faster than any floating-point DSP currently available**
- **Good memory efficiency**
- **High cost, energy use**
- **Dynamic features**
  - Kill execution-time predictability
  - Complex instruction-pairing rules hamper optimization
- **Mature tools**
- **Poor support for MMX and SSE**
- **Little integration**

(PIII, floating-point)
ARM ARM9E
The ARM9 Gets DSP Extensions

- Faster, wider multiplier hardware
  - 32 x 16 replaces 32 x 8 of ARM9
  - Adds 16 x 16 → 32 and 16 x 32 → 32 with single-cycle throughput
  - Retains 32 x 32 → 64

- Improved support for 16-bit data
  - New multiply instructions treat 32-bit registers as two 16-bit values
  - ALU instructions can access register halves via “free” shifts

- No DSP-oriented addressing
- 200 MHz in 0.18 µm
  - Fabricated by LSI Logic

ARM ARM9E
Strengths and Weaknesses

↑ Good memory efficiency
↑ Decent speed
↓ Poor energy efficiency
↑ Compatible w/ other ARM cores
  - ARM V6 adds SIMD operations
↑ Simple architecture
↓ No DSP addressing, parallel moves, or hardware loops
↑ Extensive 3rd-party support
↑ Synthesizable

(ARM946E-S, 0.18µm)
GPPs and Hybrids
Strengths and Weaknesses

↑ DSP performance can be as strong as DSP processors
↓ Often weak on integration
   ♦ Particularly high-performance CPUs
↑ General-purpose tools, infrastructure strong
↓ DSP-oriented tools, infrastructure may be weak
↑ Widely known, large installed base
↑ Compatibility (in many cases) with previous generations

↓ Higher energy use
↓ Often higher cost (mostly high-end CPUs)
↓ Dynamic features can complicate real-time operation (especially in high-end CPUs)
   ↓ Complicates ensuring real-time behavior
   ↓ Complicates software optimization
↓ Sometimes, convoluted programming model
↑ 32-bit GPPs are often easier software targets for many non-DSP tasks (e.g., network stacks)
Alternatives

- **DSP processors**
  - Many new types
- **DSP-enhanced GPPs**
  - DSP-oriented features now mainstream
- **Media processors**
- **ASSPs**
- **ASICs**
- **Customizable processors**
- **Reconfigurable processors**
- **FPGAs**

Conclusions

**Comparing Performance**

- Performance is more than speed
  - Cost/performance, energy efficiency, memory use,…
- Performance is hard to measure
  - Use appropriate benchmarks
- Consider all the options
  - Increasing performance overlap between dissimilar architectures
  - Alternatives increasingly viable
- Application requirements and processor performance are both moving targets
Software development
  ■ Tools, especially compilers
  ■ Packaged application modules
  ■ GPP-like general software support
  ■ Compatibility increasingly important

Integration, system-on-chip design
  ■ Increasing application content in chips, in chip-vendor-supplied software
  ■ Customizability

Conclusions
Comparing System Costs

Conclusions
Comparing Development Costs and Risks

• Compare processors the way you’d compare cars
  ■ Not exclusively on their top speed, price
  ■ Suitability for the task at hand
  ■ “Cost of ownership”
  ■ Time to market, ease of use,…

• Compatibility, installed base increasingly important

• SoC designs introduce new costs, risks
  ■ Processors available as both a core and a chip have a real advantage