# Current Directions for DSP Processors

February 23, 1998

Berkeley Design Technology, Inc.

2107 Dwight Way, Second Floor
Berkeley, CA 94704
+1 (510) 665-1600
http://www.bdti.com

# Some Hot Topics

- What's happening in architectures for DSP?
- Measuring DSP performance
- Integration
- Tools
- Conclusions

# New Architectures for DSP

- Enhanced conventional DSPs:
  - Lucent DSP16000, Motorola DSP56300
- VLIW:
  - TI TMS320C6200, Philips TM1000
- General-purpose processors:
  - 80x86 with MMX, PowerPC 604e
  - ARM, SH-DSP, TriCore

# Baseline: "Conventional DSPs"

- Common attributes:
  - 16- to 24-bit fixed-point (fractional) arithmetic
  - 16-, 24-, 32-bit instructions
  - One instruction per cycle ("single issue")
  - Complex, "compound" instructions encoding many operations
  - Highly constrained, non-orthogonal architectures

# Baseline: "Conventional DSPs"

- Common attributes (cont.):
  - Dedicated addressing hardware w/ specialized addressing modes
  - Multiple-access on-chip memory architecture
  - Dedicated hardware for loops and other execution control
  - Specialized on-chip peripherals and I/O interfaces
  - Low cost, low power, low memory usage

# Enhanced Conventional DSPs

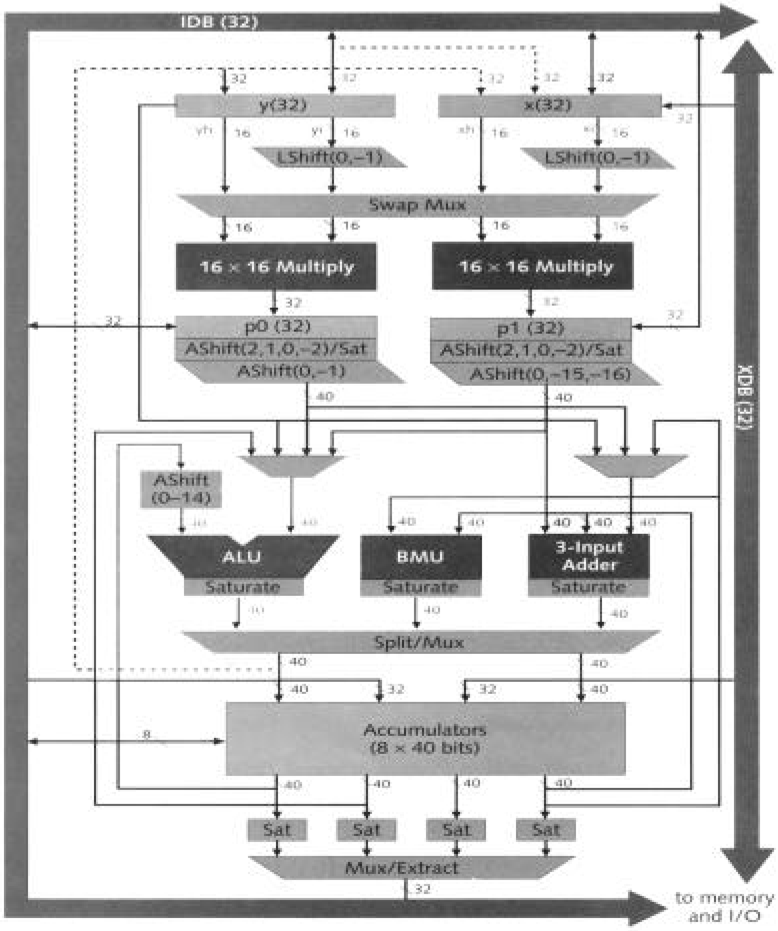
- More parallelism
  - e.g., 2nd multiplier, adder
  - limited SIMD operations
- Highly specialized hardware in core
  - e.g., application-oriented data path operations
- Co-processors
  - Viterbi decoding, FIR filtering, etc.

*Examples: Lucent DSP16000, Motorola DSP56300*

# Enhanced Conventional DSPs

- Advantages:
  - Allows incremental performance increases while maintaining competitive cost, power, code density
  - Compatibility is possible; similarity is likely
- Disadvantages:
  - Increasingly complex, hard-to-program architectures
  - Poor compiler targets
  - How much farther can we get with this approach?

# DSP16000 Data Path



# VLIW (Very Long Instruction Word)

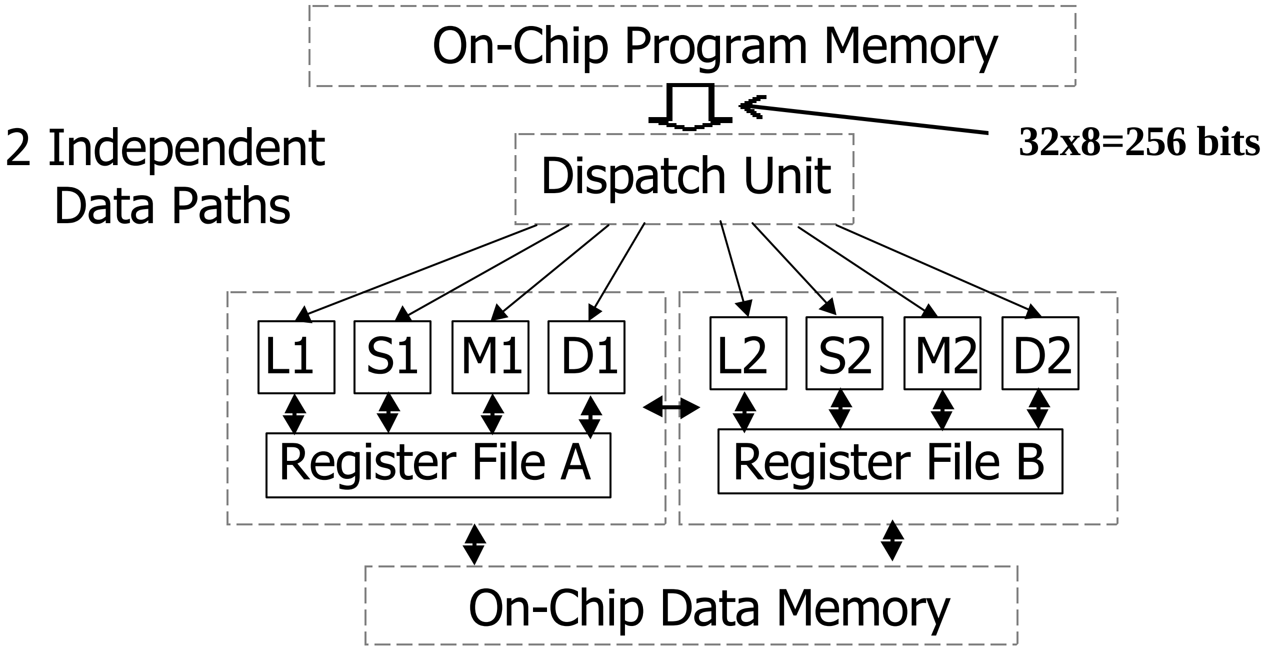
Current VLIW architectures in DSP applications:

- TI TMS320C6xxx, Philips TM1000

Characteristics:

- Multiple independent operations per cycle, packed into single large "instruction" or "packet"
- More regular, orthogonal, RISC-like operations
- Large, uniform register sets

# Example VLIW Data Path ('C6x)



© 1998 Berkeley Design Technology, Inc.

# VLIW Architectures

- Advantages:
  - Increased performance
  - More regular architectures
    (potentially easier to program, better compiler targets)
  - Scalable (?)

# VLIW Architectures

- Disadvantages:
  - New kinds of programmer/compiler complexity
  - Code size bloat
    (High program memory bandwidth requirements)
  - High power consumption

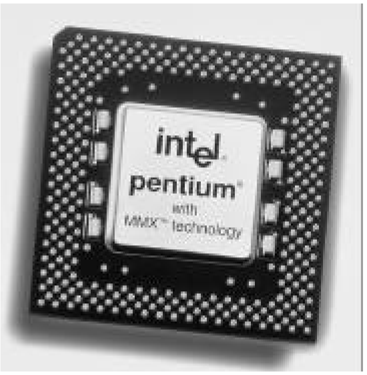
© 1998 Berkeley Design Technology, Inc.

# The GPP Threat

High-performance general-purpose processors for PCs and workstations are increasingly suitable for DSP tasks. Examples:

- MMX Pentium
- PowerPC 604e

*Why?*



# General-Purpose Processors

- Very high clock rates (200-500 MHz)
- Superscalar (multi-issue)
- Single-cycle multiplication, arithmetic ops
- Good memory bandwidth
- Branch prediction
- In some cases, SIMD operations

# High Performance GPPs for DSP

- Advantages:
  - Strong DSP Performance
  - Already present in PCs
  - Strong tools support for the major processors
  - Cost-performance can rival floating-point DSPs

# High Performance GPPs for DSP

- Disadvantages:
  - Lack of execution-time predictability (may cause problems in real-time applications)
  - Difficulty in developing optimized DSP code
  - Limited DSP-oriented tools support
  - High power consumption
  - Cost-performance doesn t approach that of fixed-point DSPs

# GPP Optimization Challenge

Vector addition on PowerPC 604e:

```
@vec_add_loop:
  lfsu fpTemp1, 4(rAAddr)     # Load A data
  lfsu fpTemp2, 4(rBAddr)     # Load B data
  fadds fpSum,fpTemp1,fpTemp2 # Perform add
  stfsu fpSum, 4(rCAddr)      # Store sum
bdnz @vec_add_loop            # loop
```

Q: How many instruction cycles per iteration?

# Embedded GPPs

GPPs for embedded applications are beginning to address DSP needs:

- Hitachi SH-DSP, ARM Piccolo, Siemens TriCore

Various approaches:

- Integrate fixed-point DSP data path & related resources with an existing uC core (SH-DSP)
- Add a DSP co-processor to existing uC core (Piccolo)
- Create an all-new hybrid architecture (TriCore)

# Embedded GPPs for DSP

- Advantages:
  - Respectable DSP performance
  - Cost-performance can rival that of fixed-point DSPs
  - Already present in many embedded applications--upgrade path
  - Many potential benefits of using one processor vs two: size, cost, etc.

# Embedded GPPs for DSP

- Disadvantages:
  - Compromise architectures betray their compromises:

    Programming complexity

    Performance penalties
  - Starting with limited DSP infrastructure

# Measuring DSP Performance

~~MIPS~~ ~~MOPS~~ ~~MFLOPS~~ ~~BOPS~~

The problem:

- Need accurate, quick comparisons of processors' DSP performance (speed, energy consumption, etc)
- Increasing diversity of architectures
- Simple metrics (MIPS, MOPS) are useless
- High performance requires hand-coded assembly
- Complete applications are impractical as benchmarks

# Measuring DSP Performance

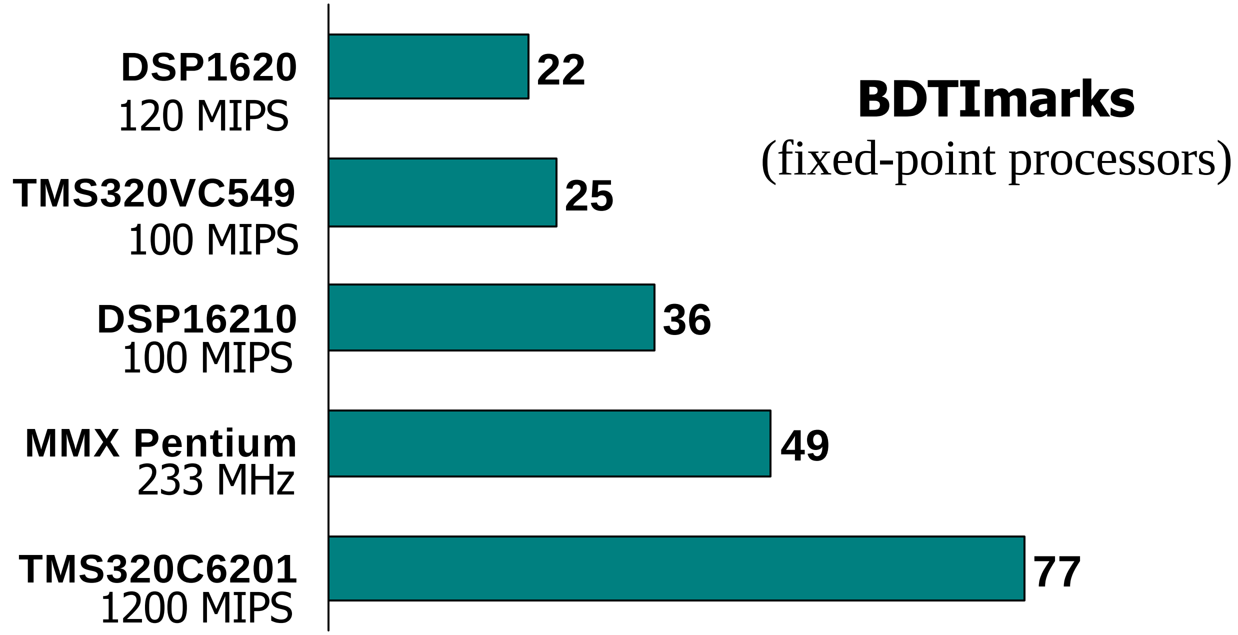
A solution:

An approach based on algorithm kernel benchmarks that are:

- Derived from important DSP applications
- Implemented in a consistent fashion
- Carefully optimized for each processor
- Verified by an independent third party

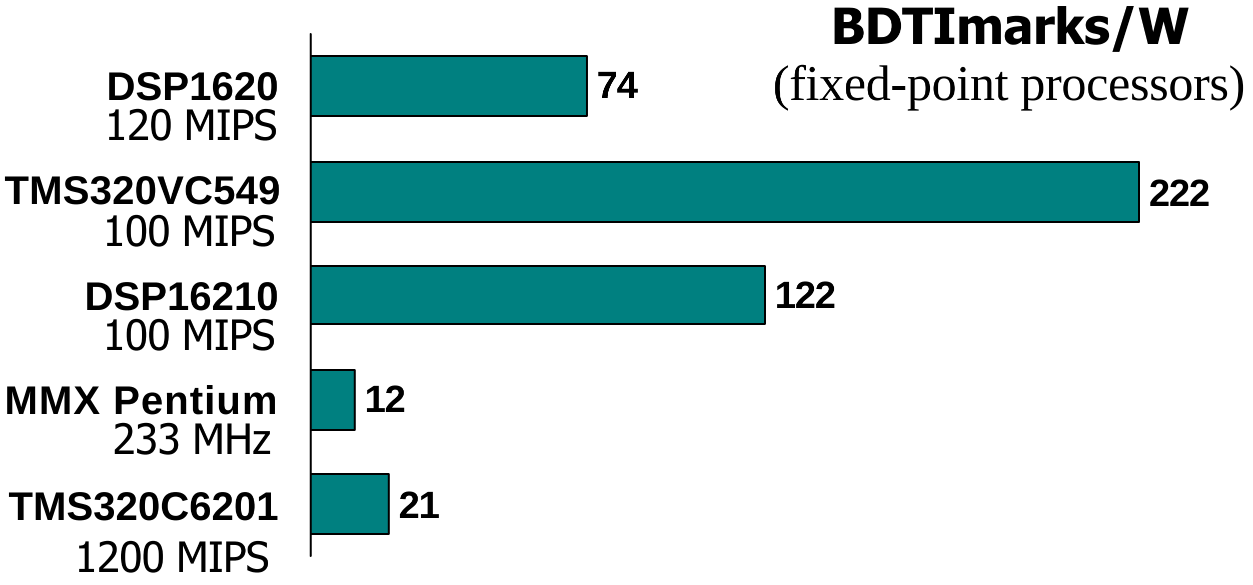
is practical and yields meaningful results.

# Example Results: Speed

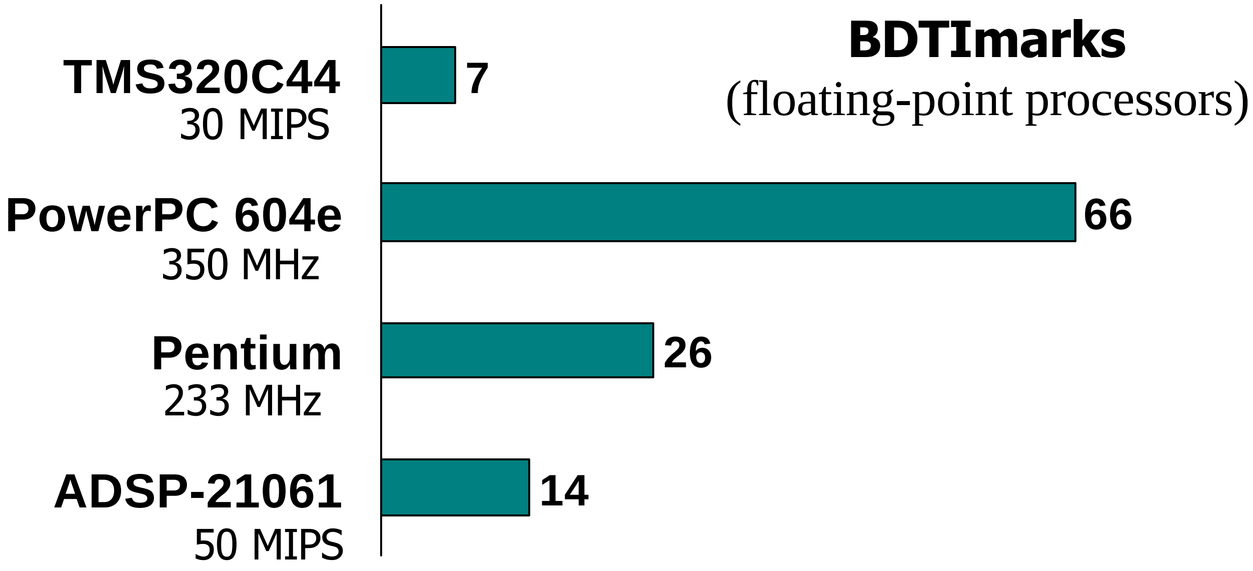


**BDTImarks**
(fixed-point processors)

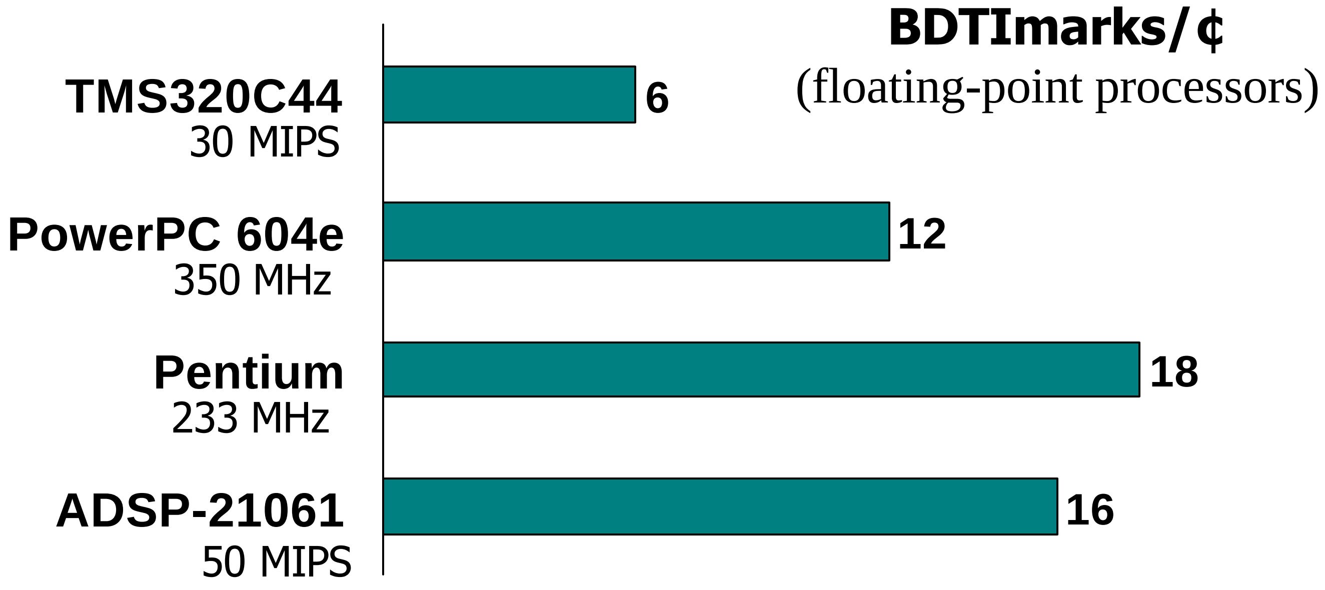
# Example Results: Energy Efficiency



# Example Results: Speed

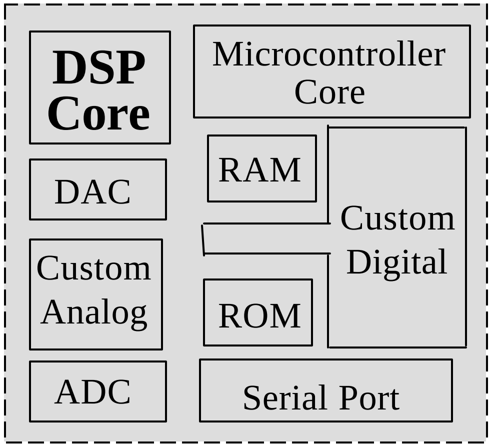


# Example Results: Cost-Performance



# Integration

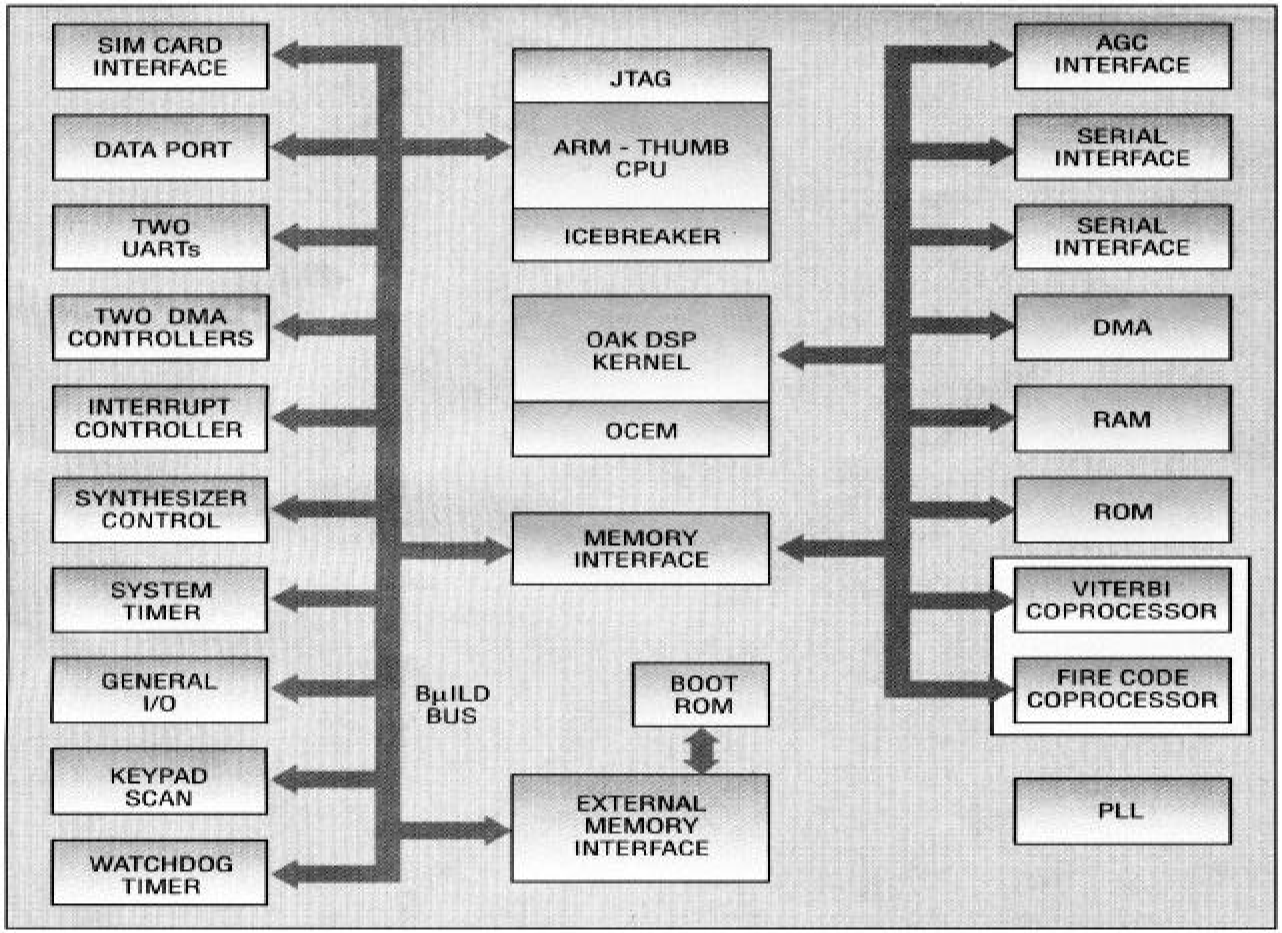
DSP processors have become building blocks for chips, rather than boards:



# Integration

High-volume DSP users want highly specialized chips for their embedded apps. Some options:

- Specialized, highly integrated off-the-shelf processor variants
- Application-specific processors and ASSPs from processor vendors and third parties
- Foundry-captive cores
- Licensable cores


SIM CARD INTERFACE
DATA PORT
TWO UARTs
TWO DMA CONTROLLERS
INTERRUPT CONTROLLER
SYNTHESIZER CONTROL
SYSTEM TIMER
GENERAL I/O
KEYPAD SCAN
WATCHDOG TIMER
JTAG
ARM - THUMB CPU
ICEBREAKER
OAK DSP KERNEL
OCEM
MEMORY INTERFACE
BμILD BUS
BOOT ROM
EXTERNAL MEMORY INTERFACE
AGC INTERFACE
SERIAL INTERFACE
SERIAL INTERFACE
DMA
RAM
ROM
VITERBI COPROCESSOR
FIRE CODE COPROCESSOR
PLL

# Tools

Shifts in tools for DSP-based development:

- On-chip debug
  - JTAG-compatible debug ports almost universal
  - Increasing sophistication of debugging resources
  - Emerging support for real-time debugging
- C Compilers
  - Compilers are becoming more credible
  - but maximum performance still requires assembly

# Tools

- Instruction-set simulators:
  - Robust cycle-accurate instruction-set simulators are needed early to facilitate software development for new processors
  - Performance, accuracy are challenges
- Third-party vendors:
  - Several major tool vendors with DSP C compiler expertise have withdrawn (Intermetrics, Tartan)
  - A new wave is emerging, with focus on integrated environments (Wind River, Allant)

# Conclusions

- Architectures
  - Increased diversification, specialization in architecture, integration, and market strategies
  - GPPs will increasingly tackle DSP tasks
  - DSP and GPP family trees will mingle
- Performance measurement
  - Getting harder, not easier
  - Application specific
  - Independent results are increasingly important

# Conclusions

- Integration
  - Cores and the ability to quickly generate custom processor-based devices will be most important
  - Tools and other infrastructure are critical for this
- Tools
  - May become more important than architectures
  - An area of challenge and opportunity
  - Ease of development of <u>efficient</u> code is key

# For More Information...

- These slides will be available at BDTI's web site:

  **http://www.bdti.com**

- *DSP Processor Fundamentals* (BDTI, 1996), a textbook on DSP processors

- *The BDTImark: A Measure of DSP Execution Speed* (BDTI, 1997), a white paper describing the methodology used to develop the BDTImark

# Join BDTI--We're Hiring!

BDTI is currently recruiting for the following positions:

- DSP Software Engineers
- DSP Engineers/Analysts

*Both full-time and part-time positions are available.*

## Why work for BDTI?

- Cool new technology: You will get to work with the hottest new processors and tools, often before they are publicly announced
- Diverse assignments: You will have the opportunity to use DSP technology for a wide range of applications, from audio to communications to
- High-caliber colleagues: We have a reputation for employing some of the sharpest engineers in the DSP industry

BDTi