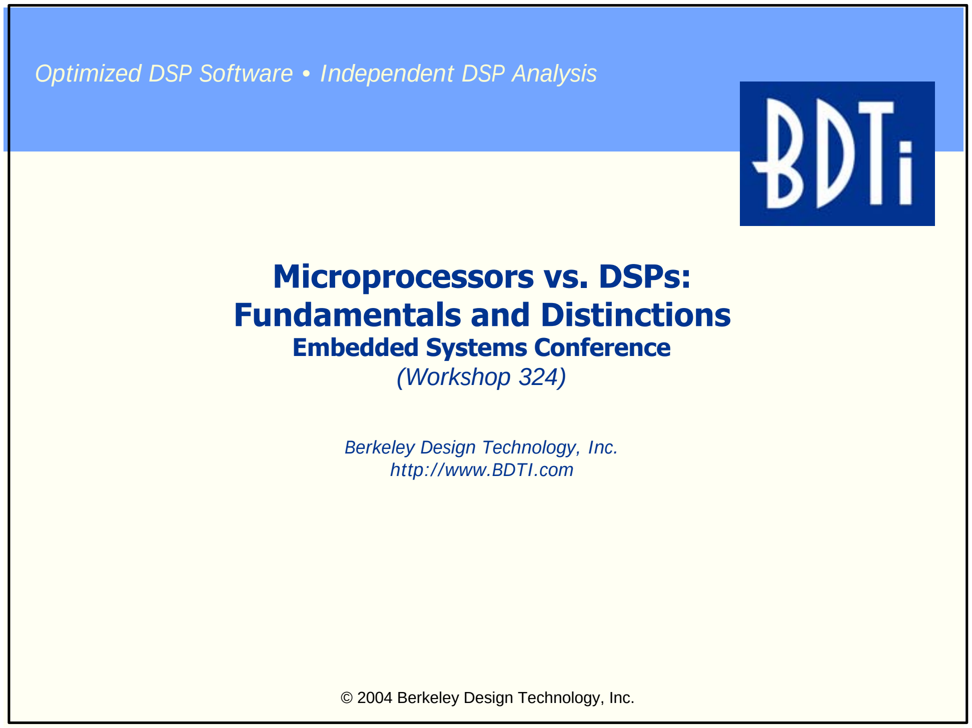


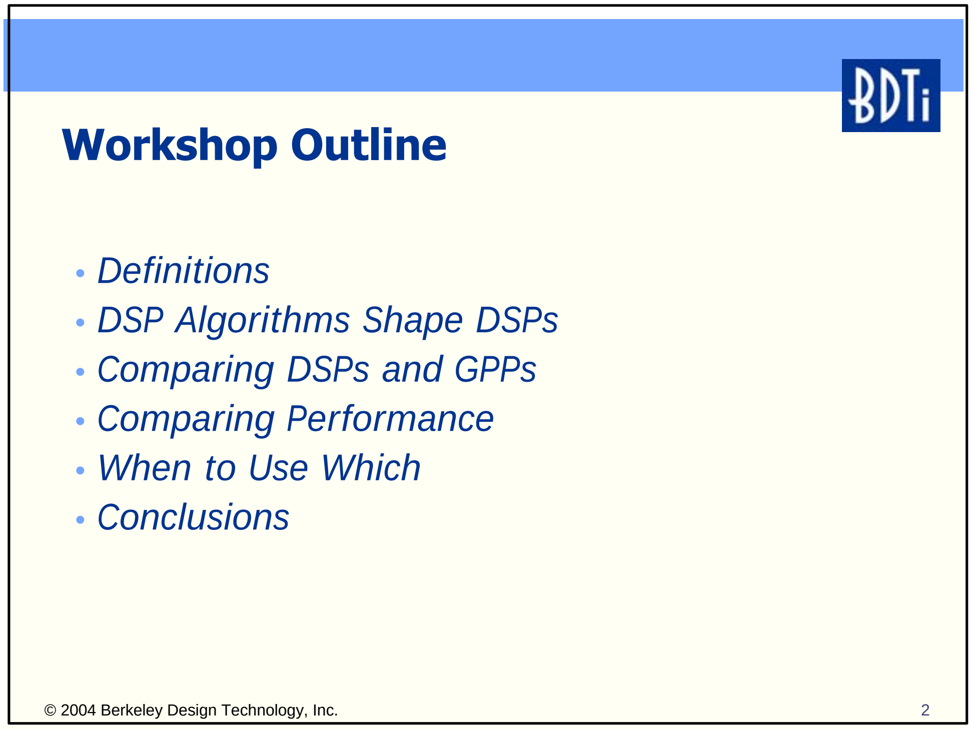
Optimized DSP Software • Independent DSP Analysis

Microprocessors vs. DSPs: Fundamentals and Distinctions

Embedded Systems Conference

(Workshop 324)

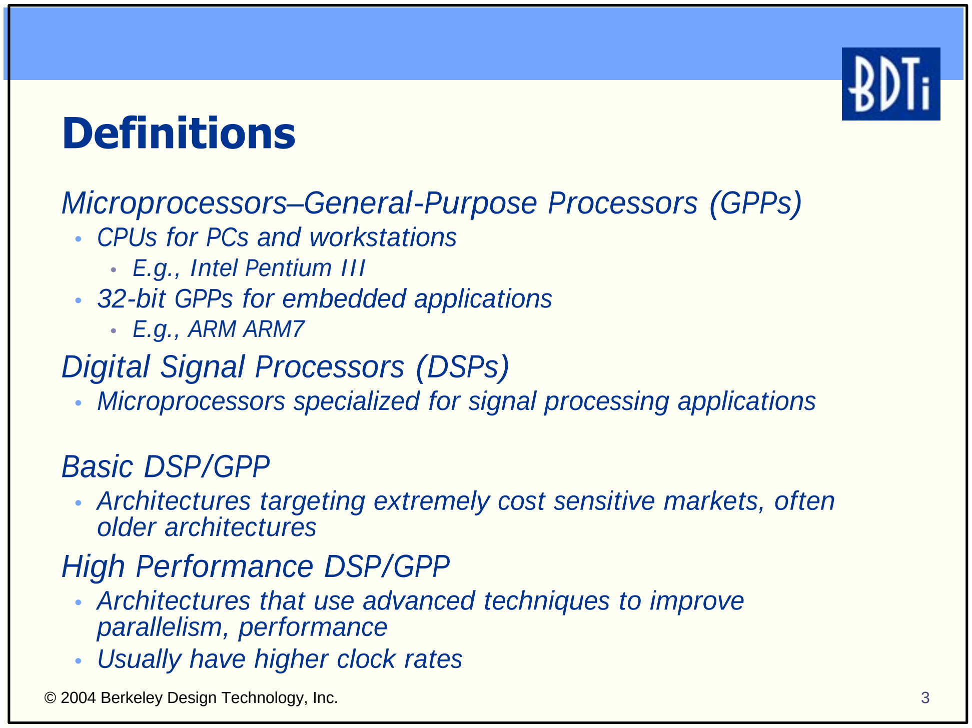
Berkeley Design Technology, Inc.
http://www.BDTI.com



Workshop Outline

- *Definitions*
- *DSP Algorithms Shape DSPs*
- *Comparing DSPs and GPPs*
- *Comparing Performance*
- *When to Use Which*
- *Conclusions*

2



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)

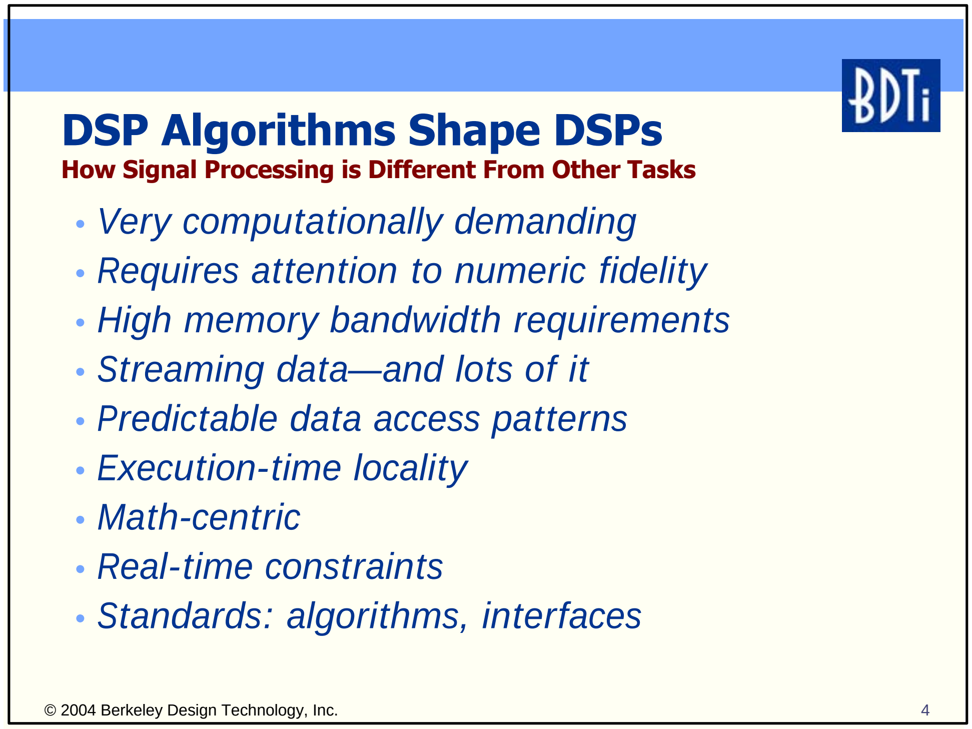
- *Microprocessors specialized for signal processing applications*

Basic DSP/GPP

- *Architectures targeting extremely cost sensitive markets, often older architectures*

High Performance DSP/GPP

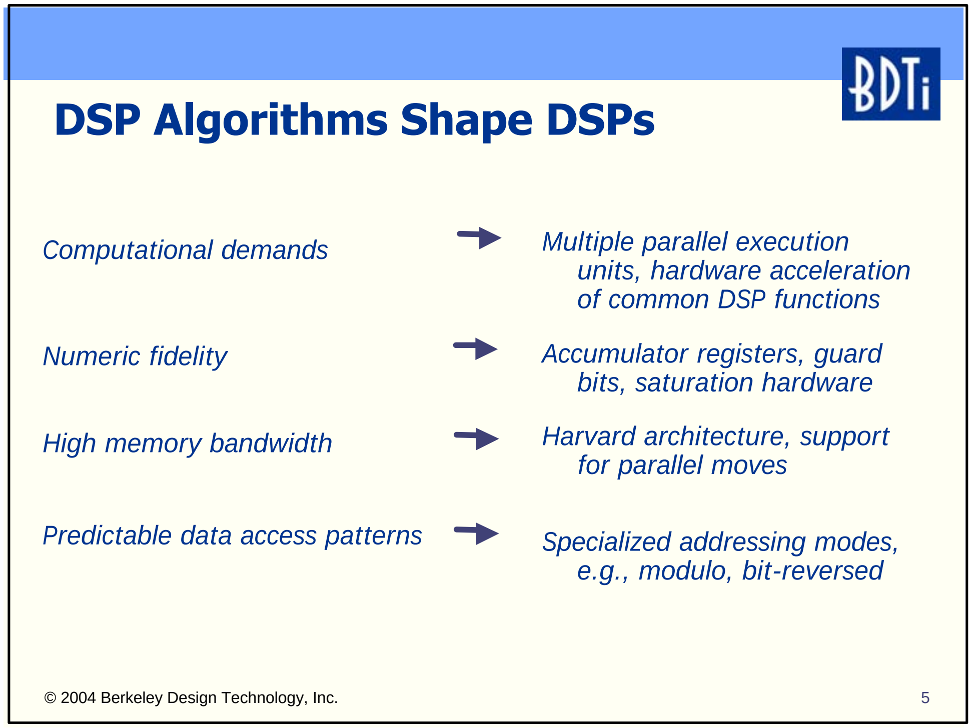
- *Architectures that use advanced techniques to improve parallelism, performance*
- *Usually have higher clock rates*



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*

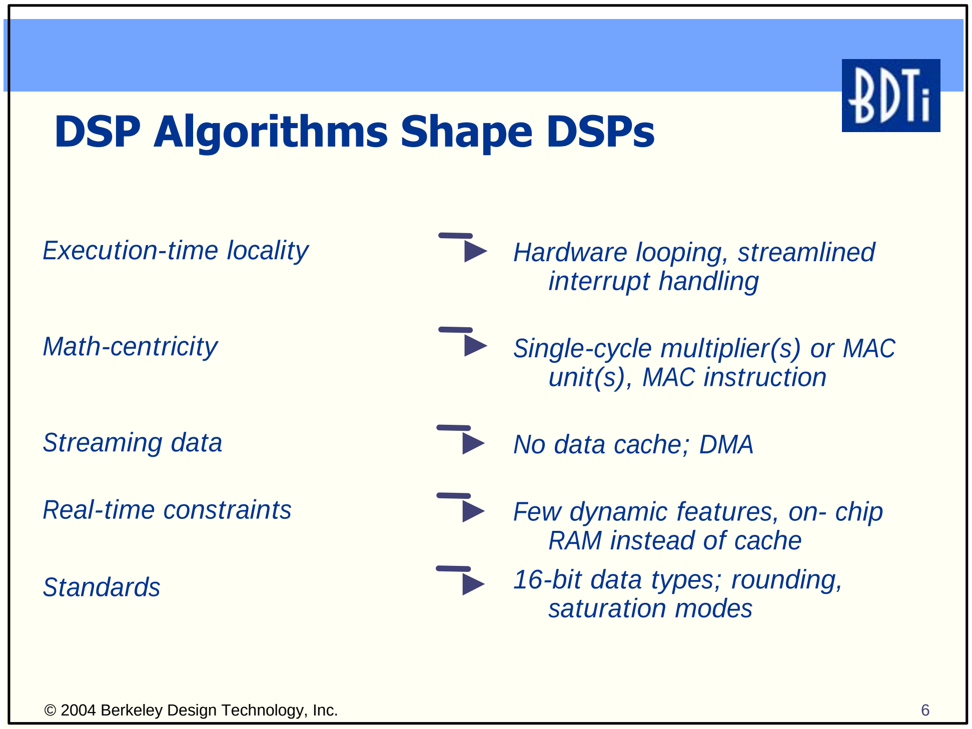


DSP Algorithms Shape DSPs

BDTi

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

© 2004 Berkeley Design Technology, Inc. 5

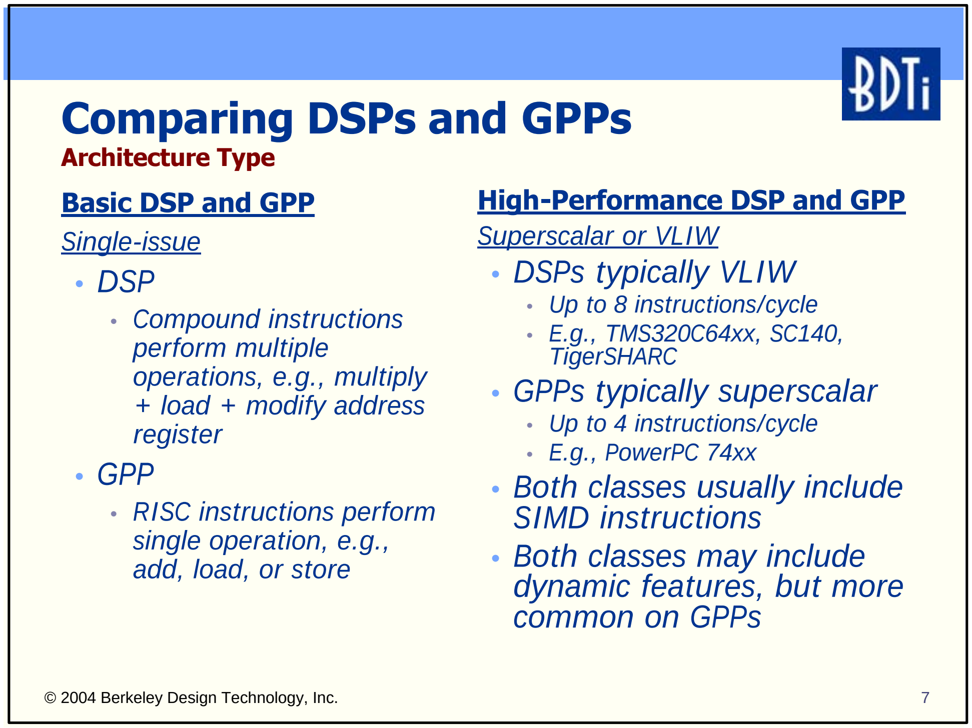


DSP Algorithms Shape DSPs

BDTi

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

© 2004 Berkeley Design Technology, Inc. 6



Comparing DSPs and GPPs

Architecture Type

Basic DSP and GPP

Single-issue

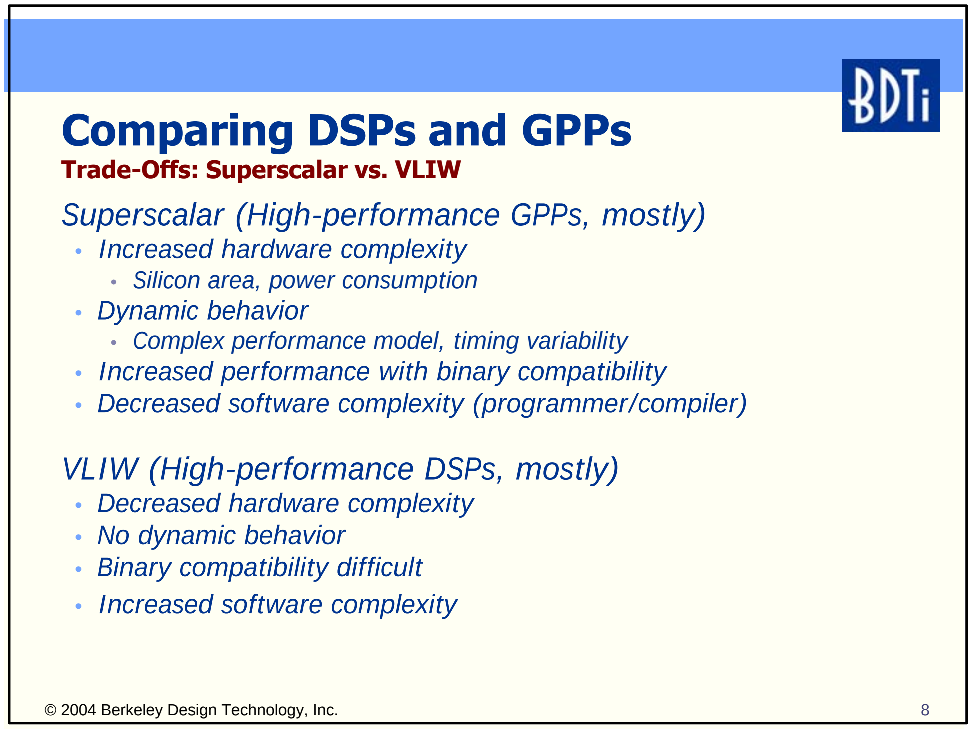
- *DSP*
  - *Compound instructions perform multiple operations, e.g., multiply + load + modify address register*
- *GPP*
  - *RISC instructions perform single operation, e.g., add, load, or store*

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*

© 2004 Berkeley Design Technology, Inc. 7



Comparing DSPs and GPPs

Trade-Offs: Superscalar vs. VLIW

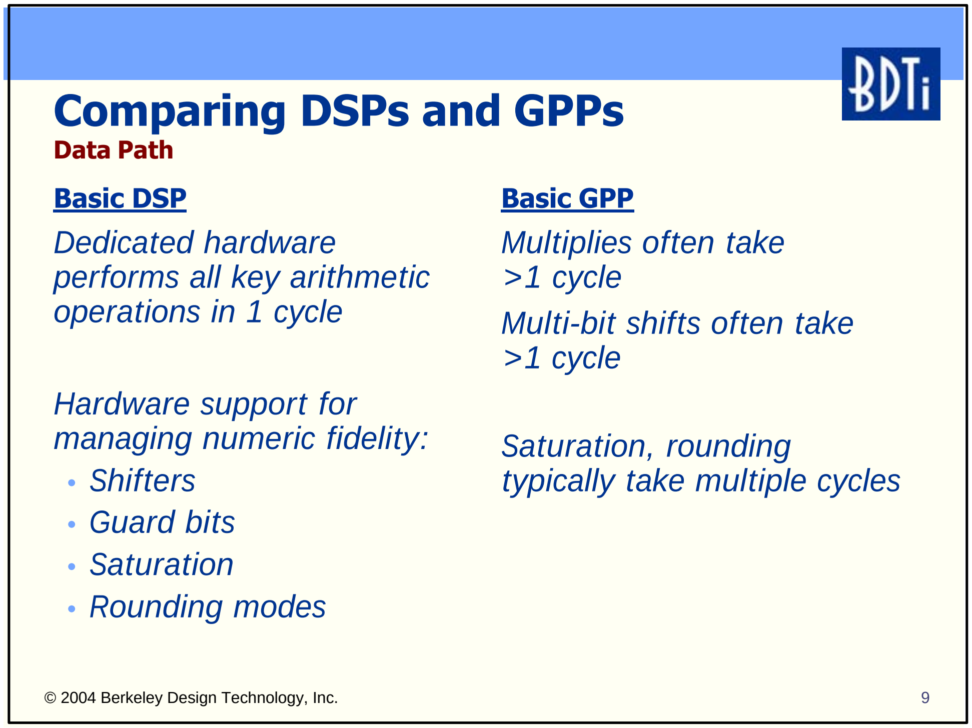
Superscalar (High-performance GPPs, mostly)

- *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*

© 2004 Berkeley Design Technology, Inc. 8



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*

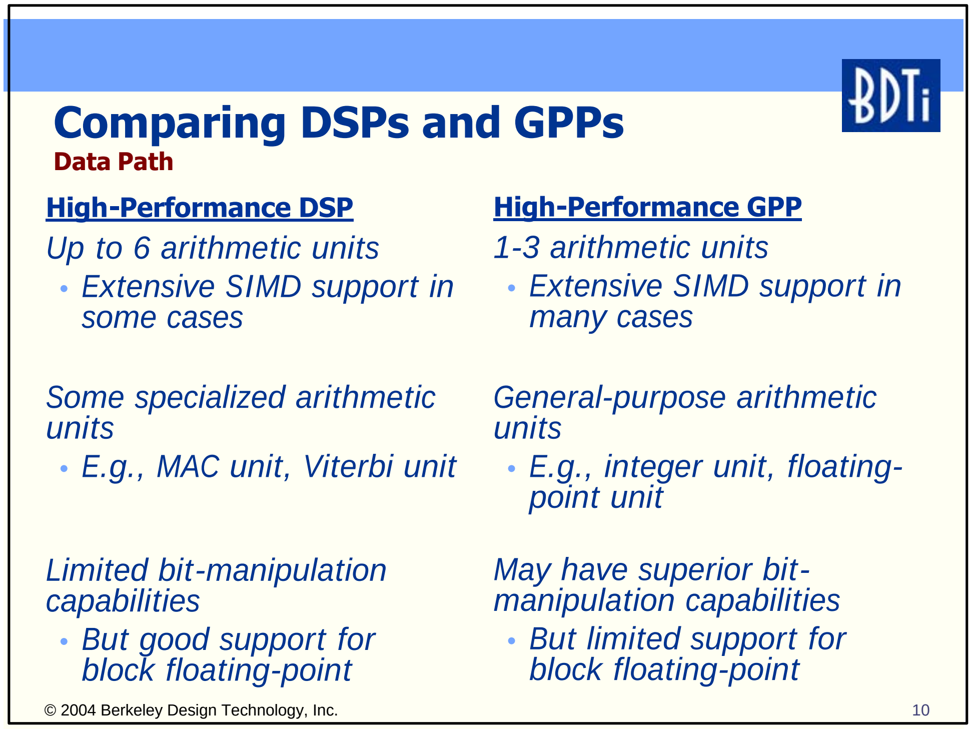
Basic GPP

Multiplies often take >1 cycle

Multi-bit shifts often take >1 cycle

Saturation, rounding typically take multiple cycles

9



Comparing DSPs and GPPs

Data Path

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*

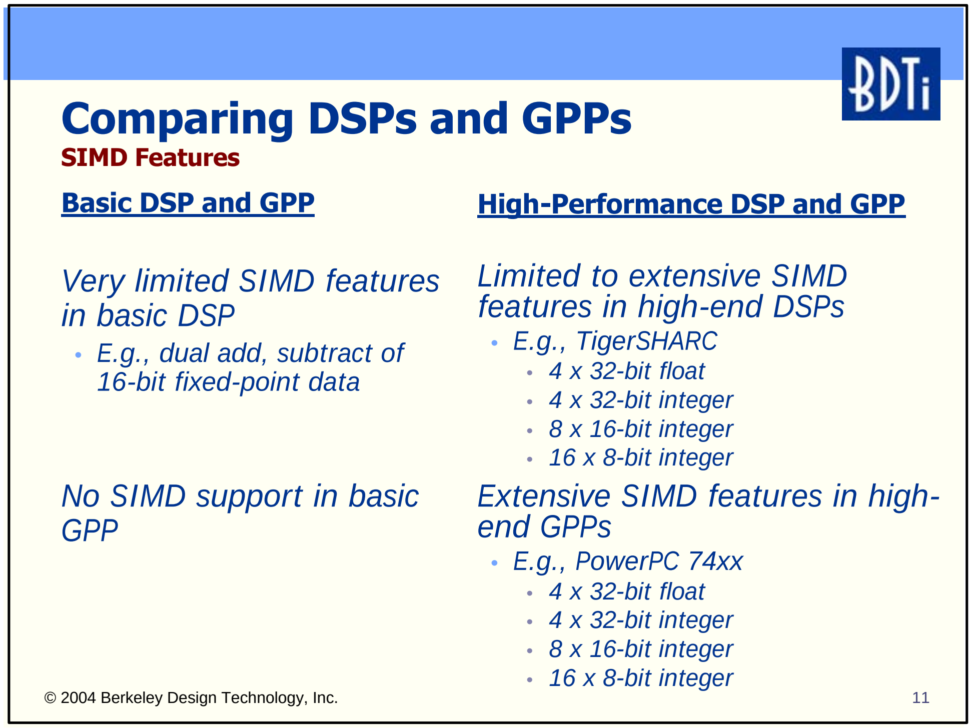
General-purpose arithmetic units

- *E.g., integer unit, floating-point unit*

May have superior bit-manipulation capabilities

- *But limited support for block floating-point*

10



Comparing DSPs and GPPs

SIMD Features

Basic DSP and GPP

Very limited SIMD features in basic DSP

- *E.g., dual add, subtract of 16-bit fixed-point data*

No SIMD support in basic GPP

High-Performance DSP and GPP

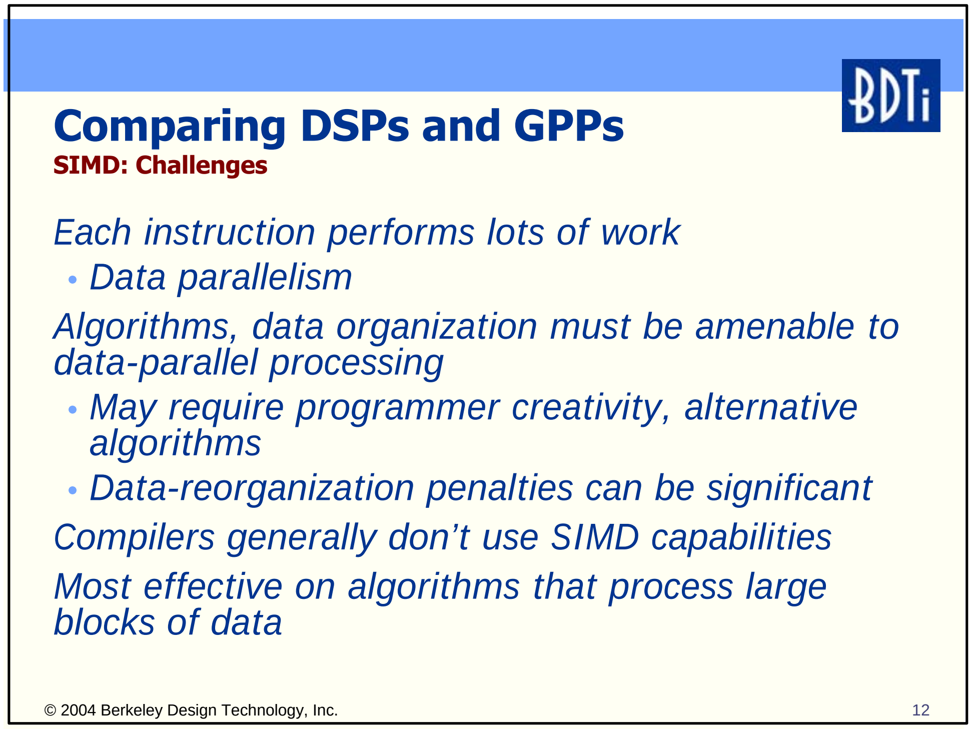
Limited to extensive SIMD features in high-end DSPs

- *E.g., TigerSHARC*
  - *4 x 32-bit float*
  - *4 x 32-bit integer*
  - *8 x 16-bit integer*
  - *16 x 8-bit integer*

Extensive SIMD features in high-end GPPs

- *E.g., PowerPC 74xx*
  - *4 x 32-bit float*
  - *4 x 32-bit integer*
  - *8 x 16-bit integer*
  - *16 x 8-bit integer*

© 2004 Berkeley Design Technology, Inc. 11



Comparing DSPs and GPPs

SIMD: Challenges

Each instruction performs lots of work

- *Data parallelism*

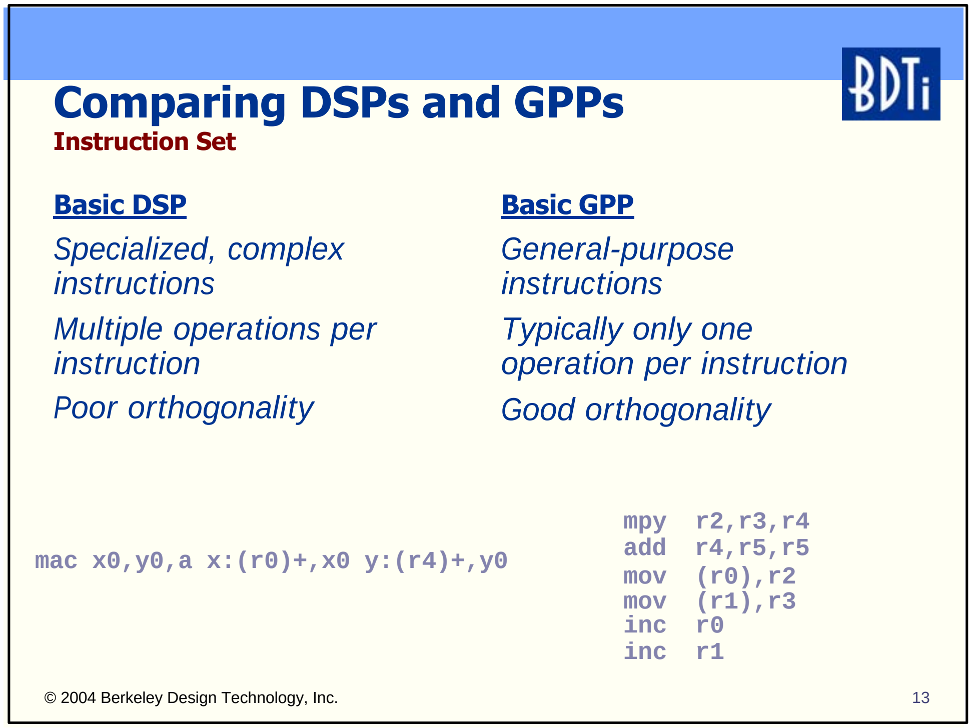
Algorithms, data organization must be amenable to data-parallel processing

- *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

© 2004 Berkeley Design Technology, Inc. 12



Comparing DSPs and GPPs

Instruction Set

Basic DSP

Specialized, complex instructions

Multiple operations per instruction

Poor orthogonality

```
mac x0,y0,a x:(r0)+,x0 y:(r4)+,y0
```

Basic GPP

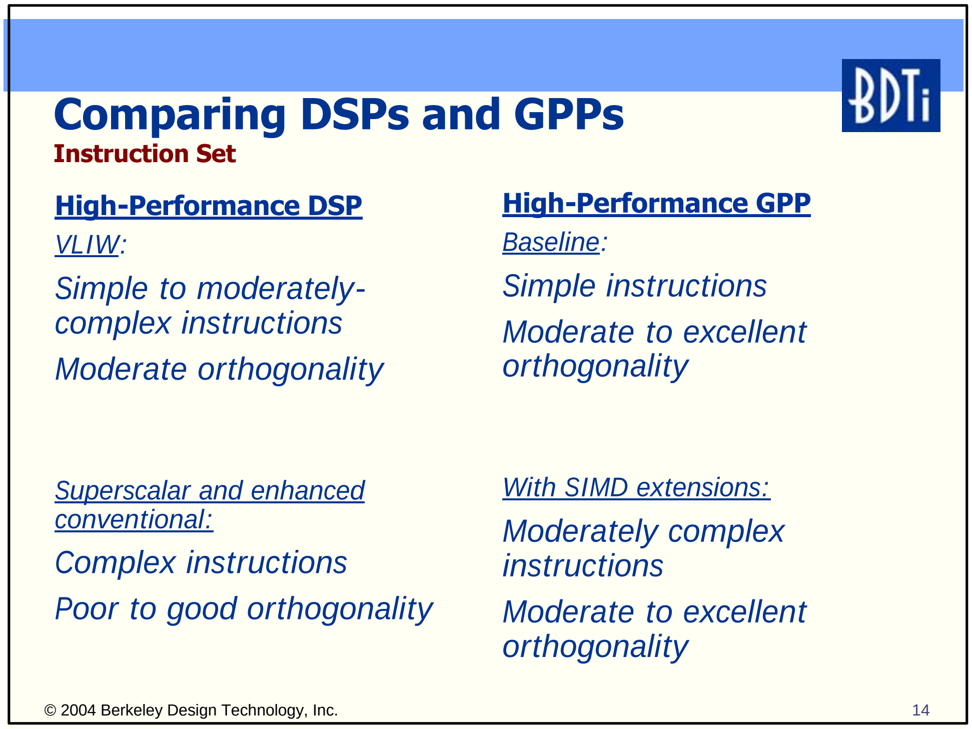
General-purpose instructions

Typically only one operation per instruction

Good orthogonality

```
mpy  r2,r3,r4
add  r4,r5,r5
mov  (r0),r2
mov  (r1),r3
inc  r0
inc  r1
```

13



Comparing DSPs and GPPs

Instruction Set

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

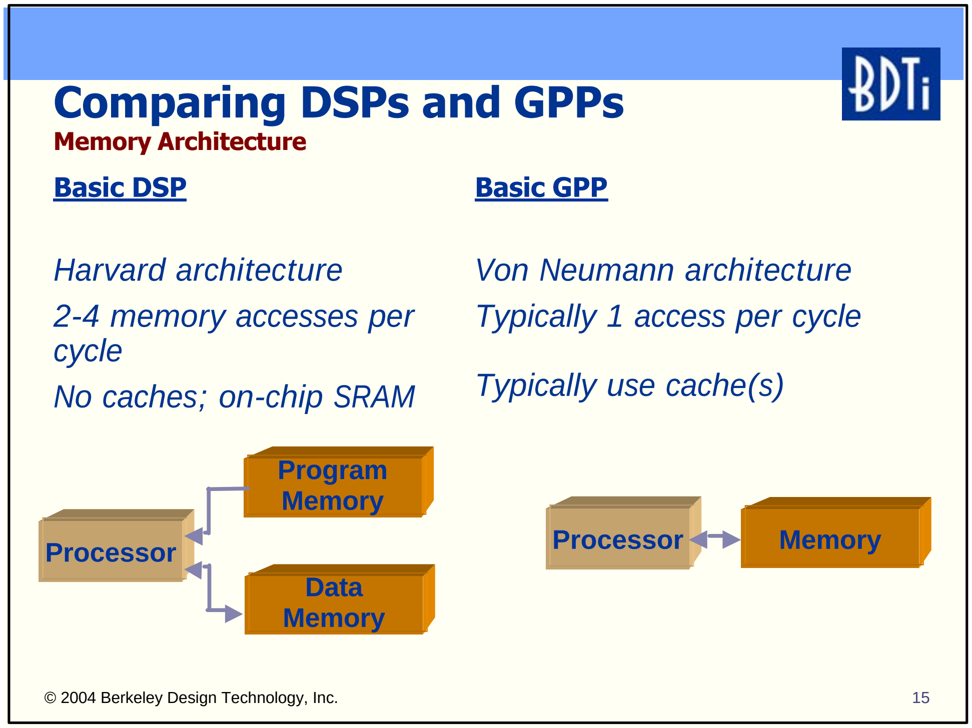
Moderate to excellent orthogonality

With SIMD extensions:

Moderately complex instructions

Moderate to excellent orthogonality

14



Comparing DSPs and GPPs

Memory Architecture

Basic DSP

Harvard architecture

2-4 memory accesses per cycle

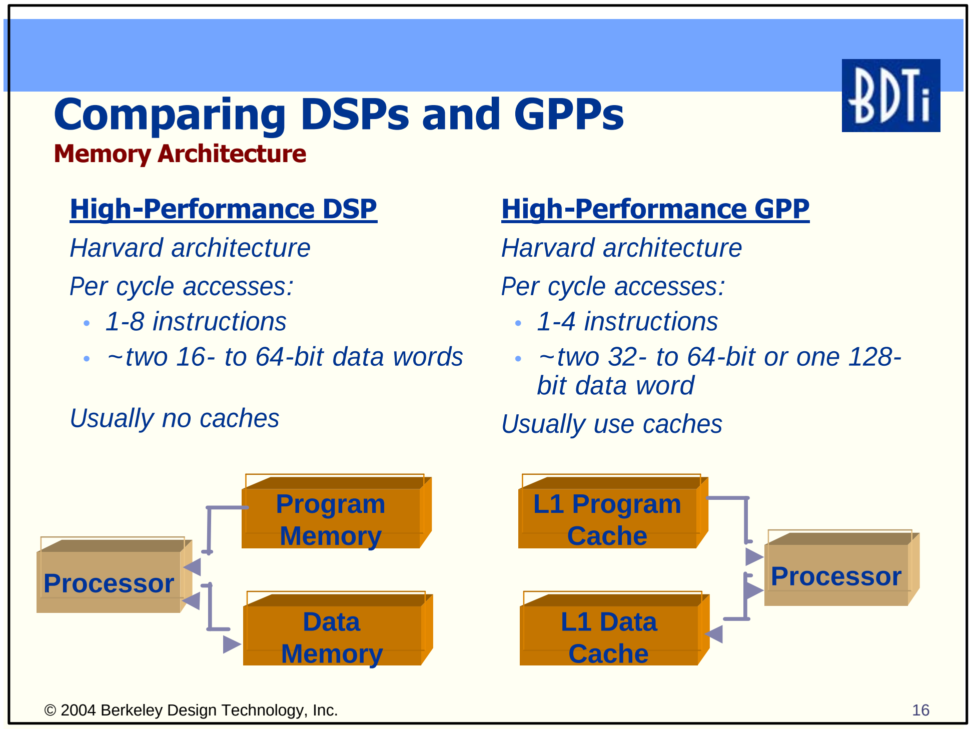
No caches; on-chip SRAM

Basic GPP

Von Neumann architecture

Typically 1 access per cycle

Typically use cache(s)



Comparing DSPs and GPPs

Memory Architecture

High-Performance DSP

Harvard architecture

Per cycle accesses:

- *1-8 instructions*
- *~two 16- to 64-bit data words*

Usually no caches

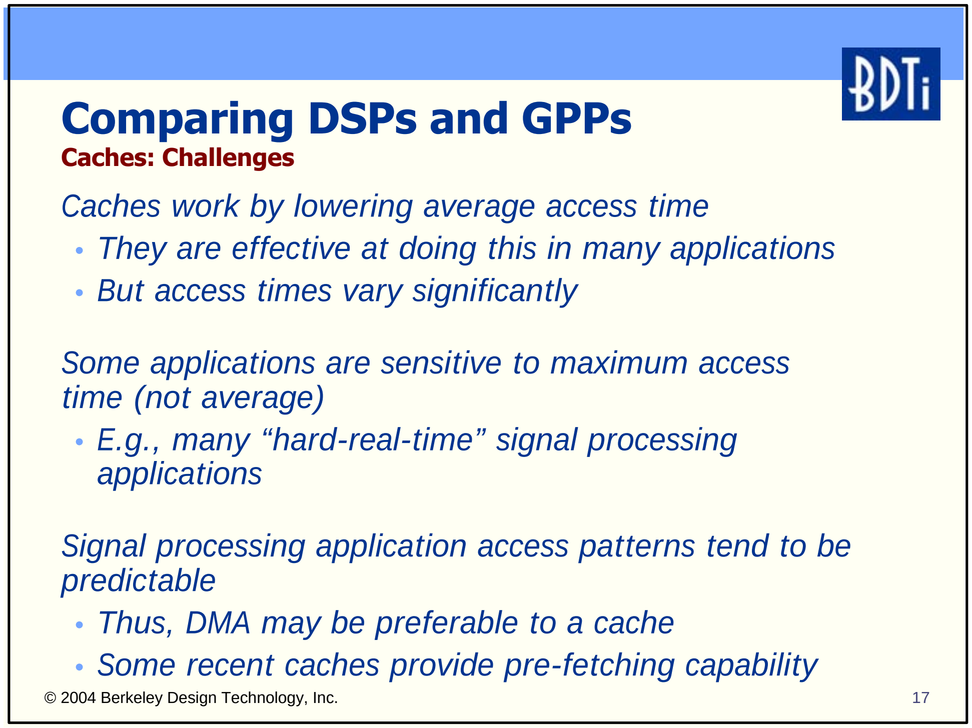
High-Performance GPP

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*

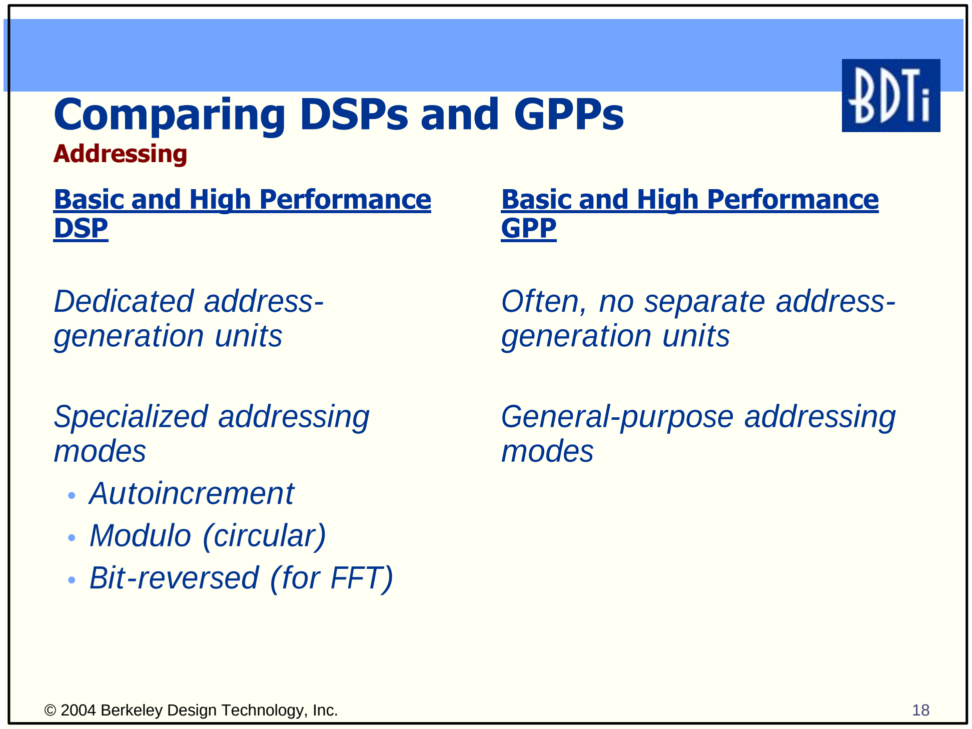
Some applications are sensitive to maximum access time (not average)

- *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*

17



Comparing DSPs and GPPs

Addressing

Basic and High Performance DSP

Dedicated address-generation units

Specialized addressing modes

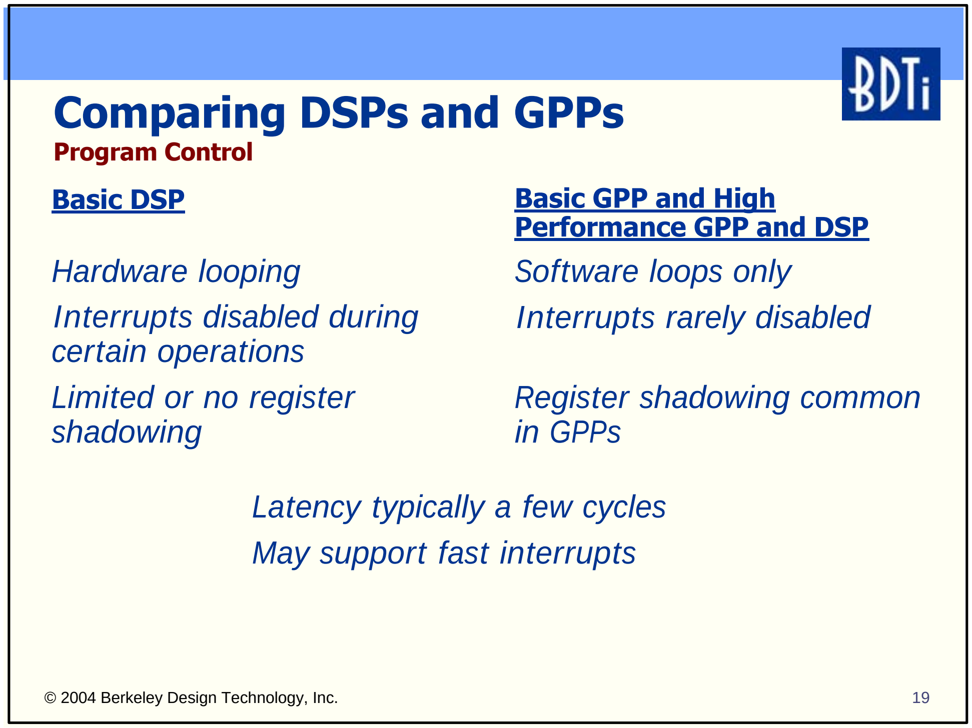
- *Autoincrement*
- *Modulo (circular)*
- *Bit-reversed (for FFT)*

Basic and High Performance GPP

Often, no separate address-generation units

General-purpose addressing modes

18



Comparing DSPs and GPPs

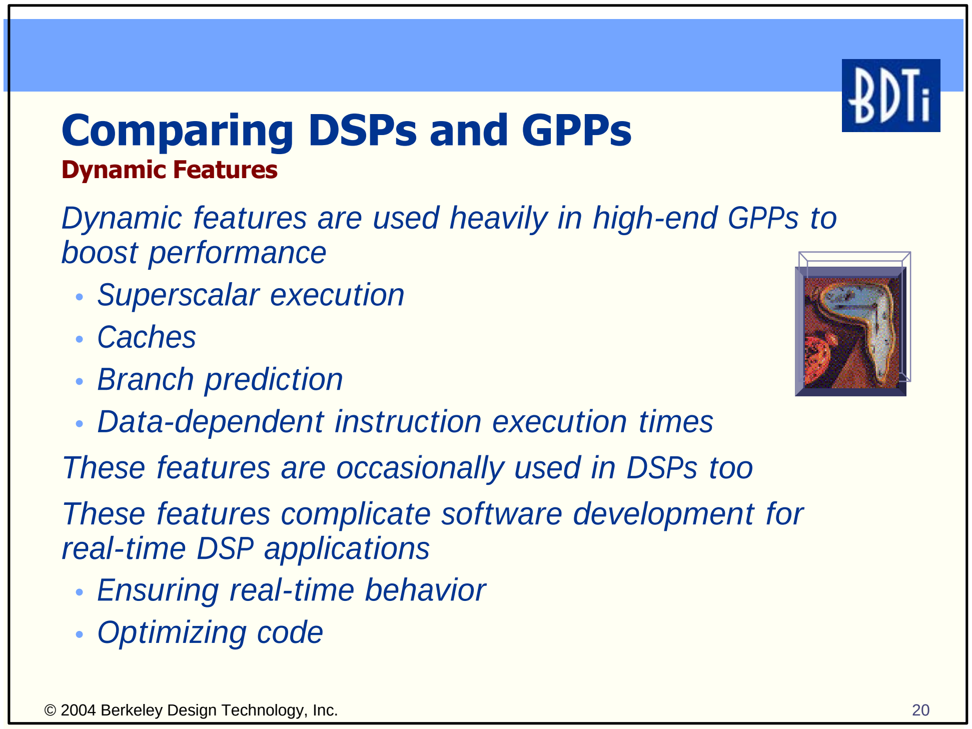
Program Control

Basic DSP	Basic GPP and High Performance GPP and DSP
Hardware looping	*Software loops only*
Interrupts disabled during certain operations	*Interrupts rarely disabled*
Limited or no register shadowing	*Register shadowing common in GPPs*

Latency typically a few cycles

May support fast interrupts

19



Comparing DSPs and GPPs

Dynamic Features

Dynamic features are used heavily in high-end GPPs to boost performance

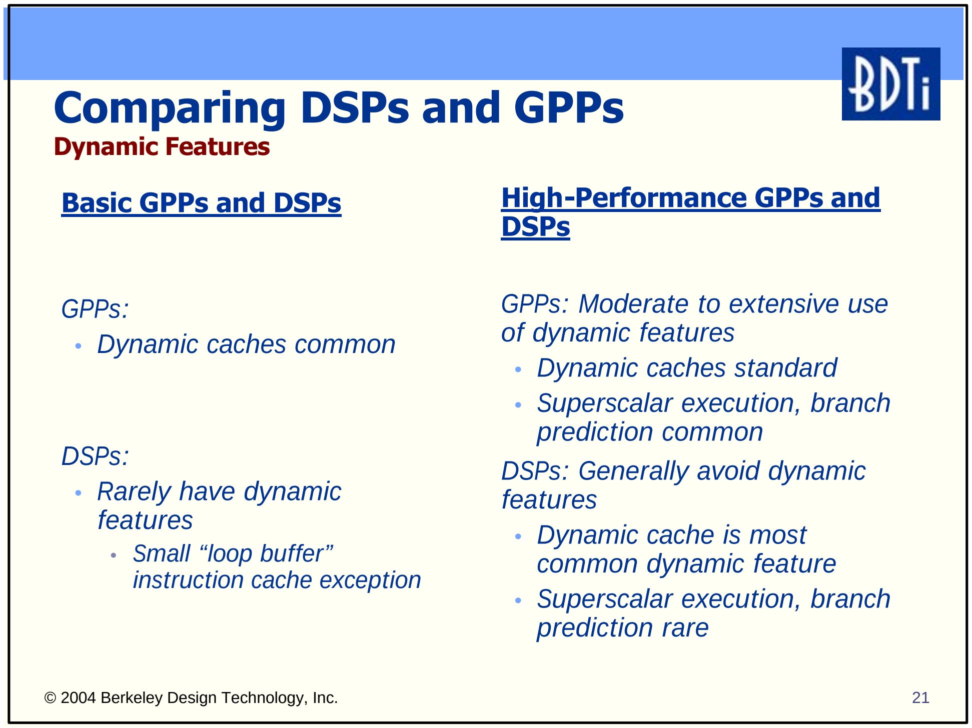
- *Superscalar execution*
- *Caches*
- *Branch prediction*
- *Data-dependent instruction execution times*

These features are occasionally used in DSPs too

These features complicate software development for real-time DSP applications

- *Ensuring real-time behavior*
- *Optimizing code*

20



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*

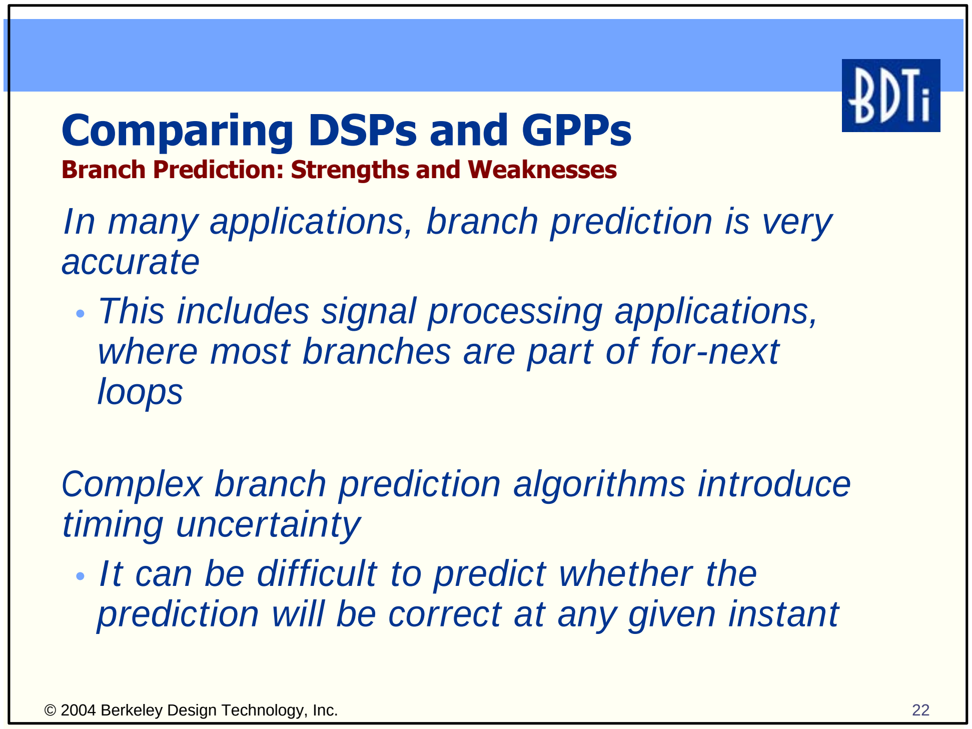
High-Performance GPPs and DSPs

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

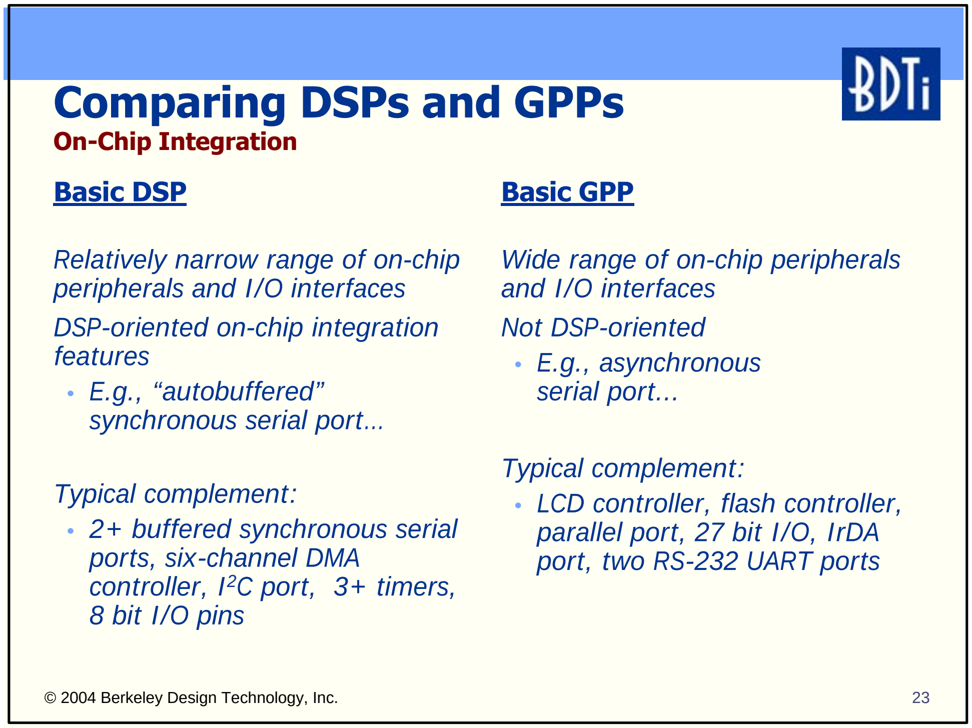
Branch Prediction: Strengths and Weaknesses

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*



Comparing DSPs and GPPs

On-Chip Integration

Basic DSP

Relatively narrow range of on-chip peripherals and I/O interfaces

DSP-oriented on-chip integration features

- *E.g., "autobuffered" synchronous serial port…*

Typical complement:

- *2+ buffered synchronous serial ports, six-channel DMA controller, I^2C port, 3+ timers, 8 bit I/O pins*

Basic GPP

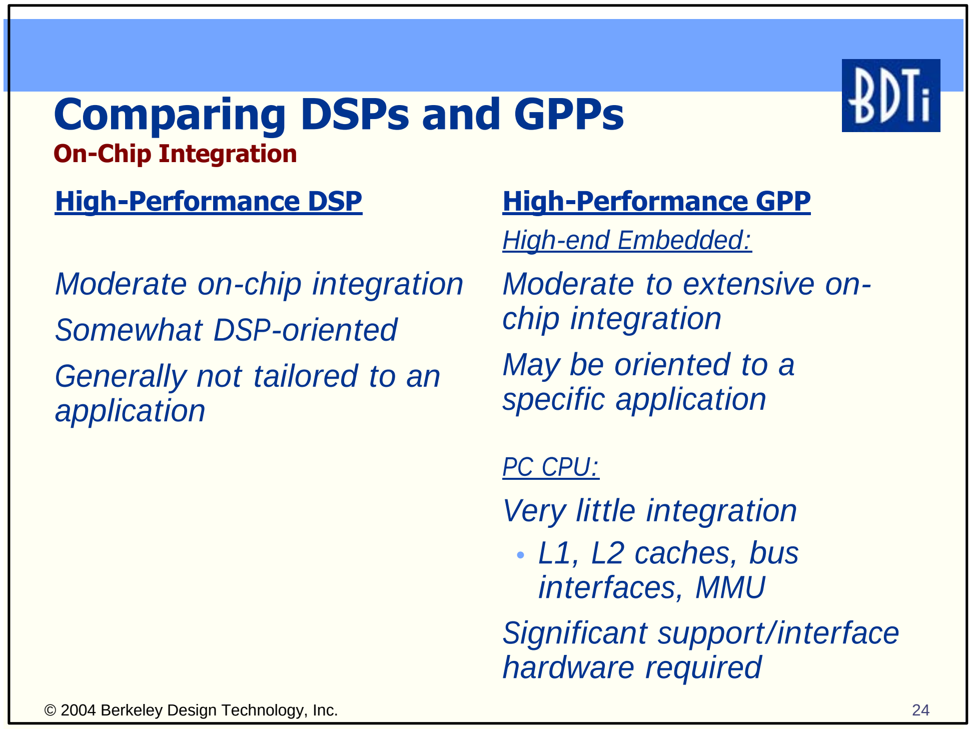
Wide range of on-chip peripherals and I/O interfaces

Not DSP-oriented

- *E.g., asynchronous serial port…*

Typical complement:

- *LCD controller, flash controller, parallel port, 27 bit I/O, IrDA port, two RS-232 UART ports*



Comparing DSPs and GPPs

On-Chip Integration

High-Performance DSP

Moderate on-chip integration

Somewhat DSP-oriented

Generally not tailored to an application

High-Performance GPP

High-end Embedded:

Moderate to extensive on-chip integration

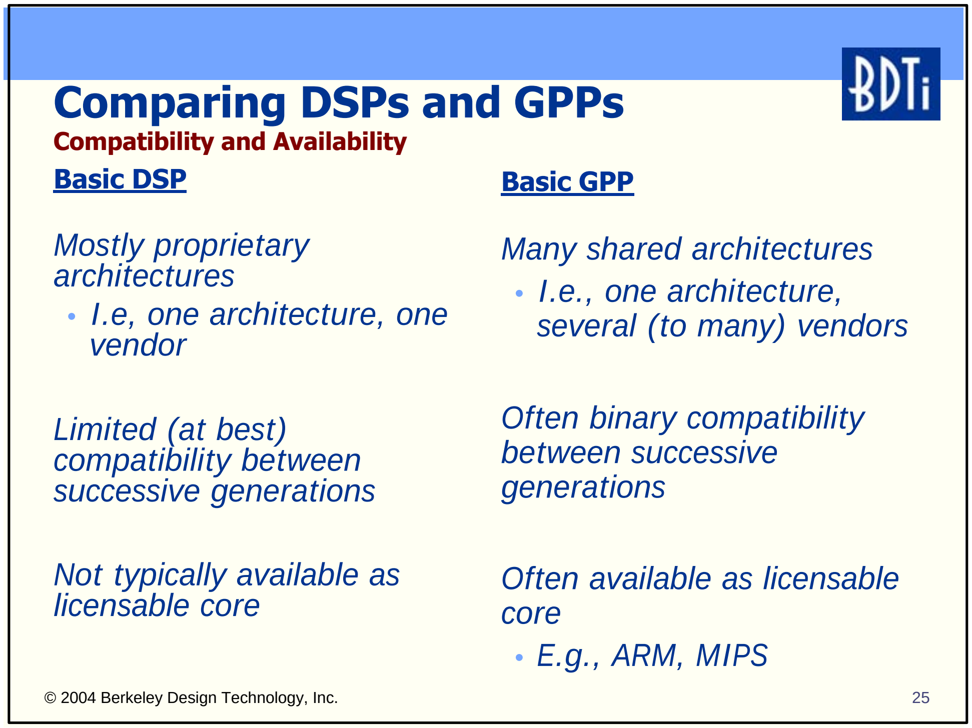
May be oriented to a specific application

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

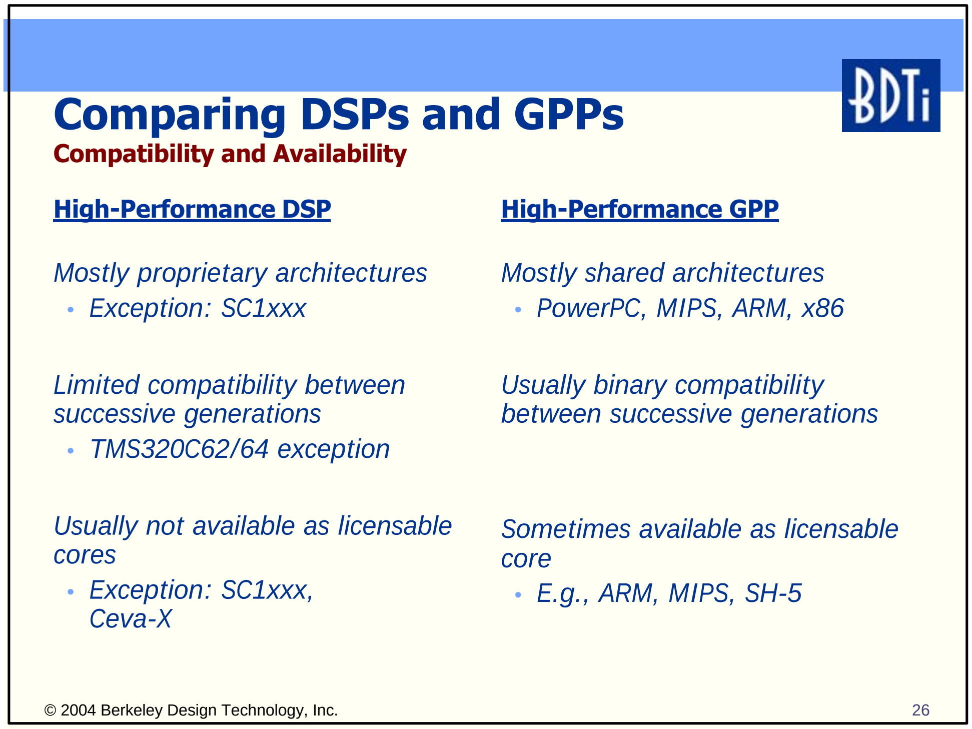
- *I.e., one architecture, several (to many) vendors*

Often binary compatibility between successive generations

Often available as licensable core

- *E.g., ARM, MIPS*

25



Comparing DSPs and GPPs

Compatibility and Availability

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

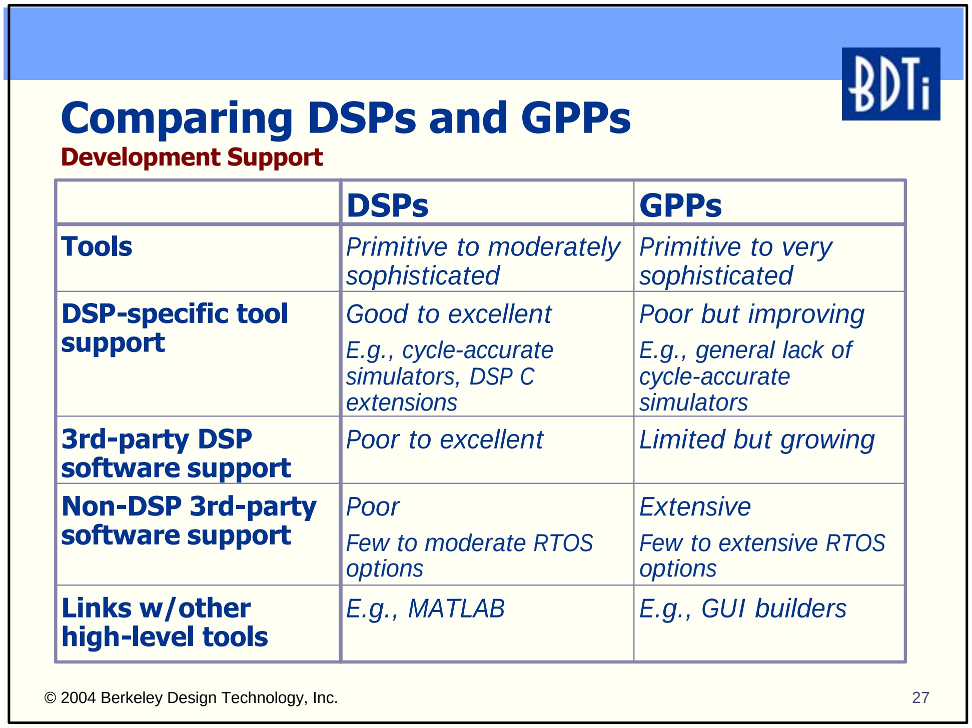
- *PowerPC, MIPS, ARM, x86*

Usually binary compatibility between successive generations

Sometimes available as licensable core

- *E.g., ARM, MIPS, SH-5*

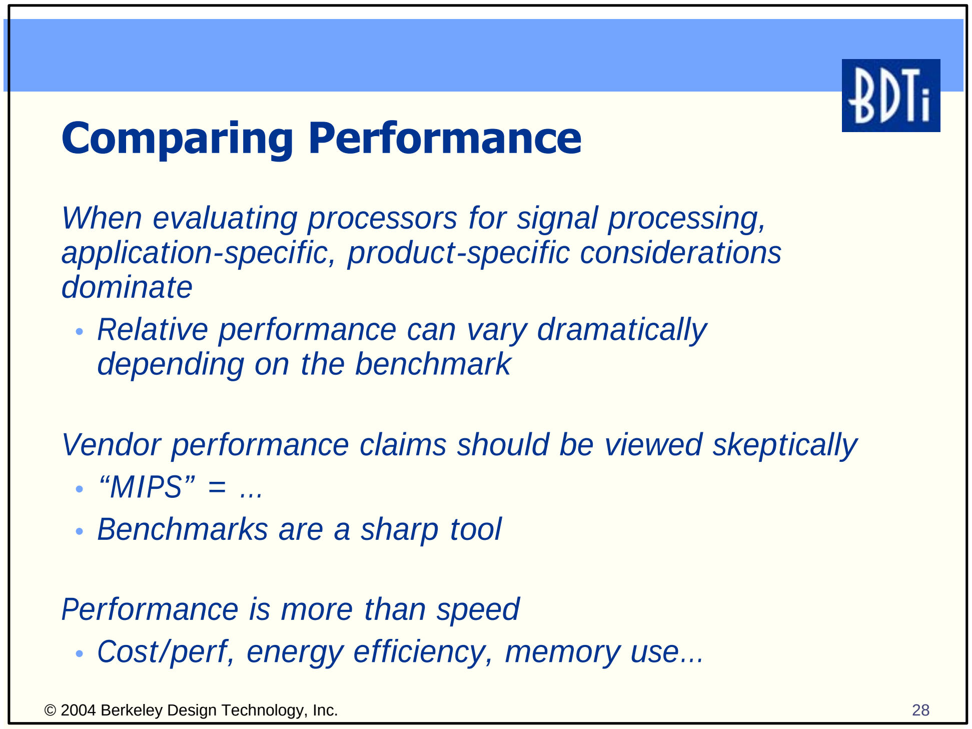
26



Comparing DSPs and GPPs

Development Support

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



Comparing Performance

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

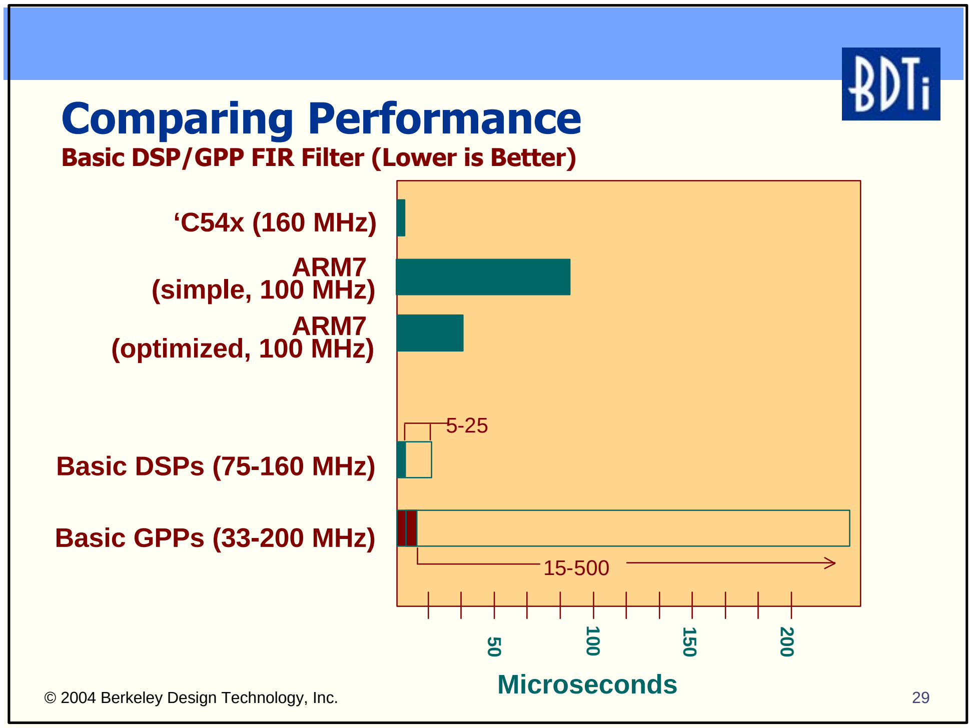
- *Relative performance can vary dramatically depending on the benchmark*

Vendor performance claims should be viewed skeptically

- *"MIPS" = ...*
- *Benchmarks are a sharp tool*

Performance is more than speed

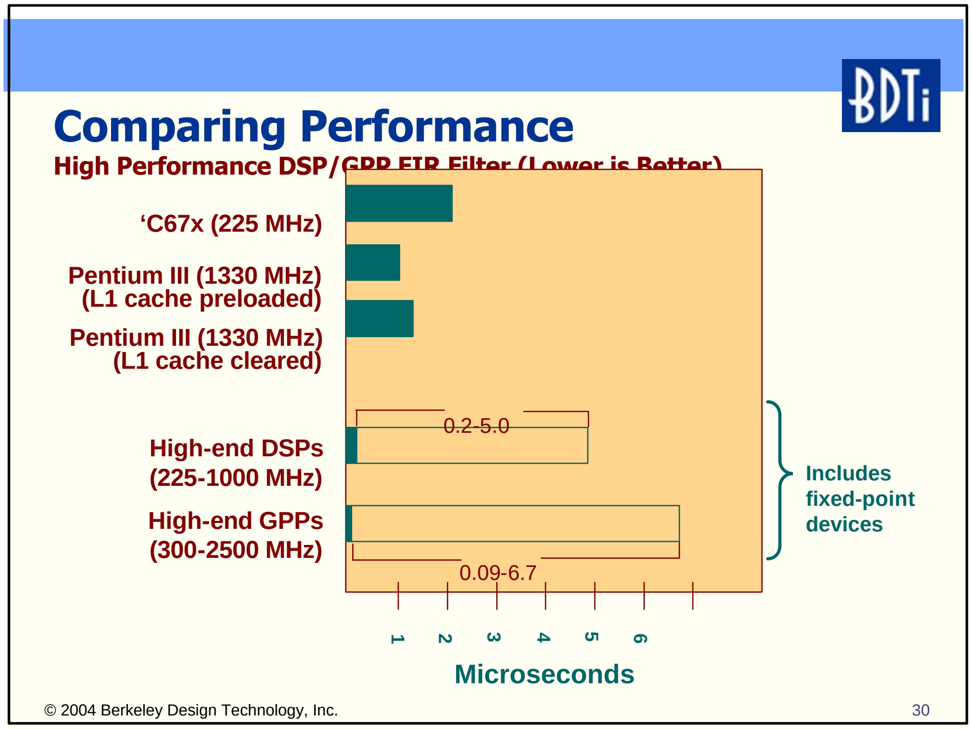
- *Cost/perf, energy efficiency, memory use...*



Comparing Performance

Basic DSP/GPP FIR Filter (Lower is Better)

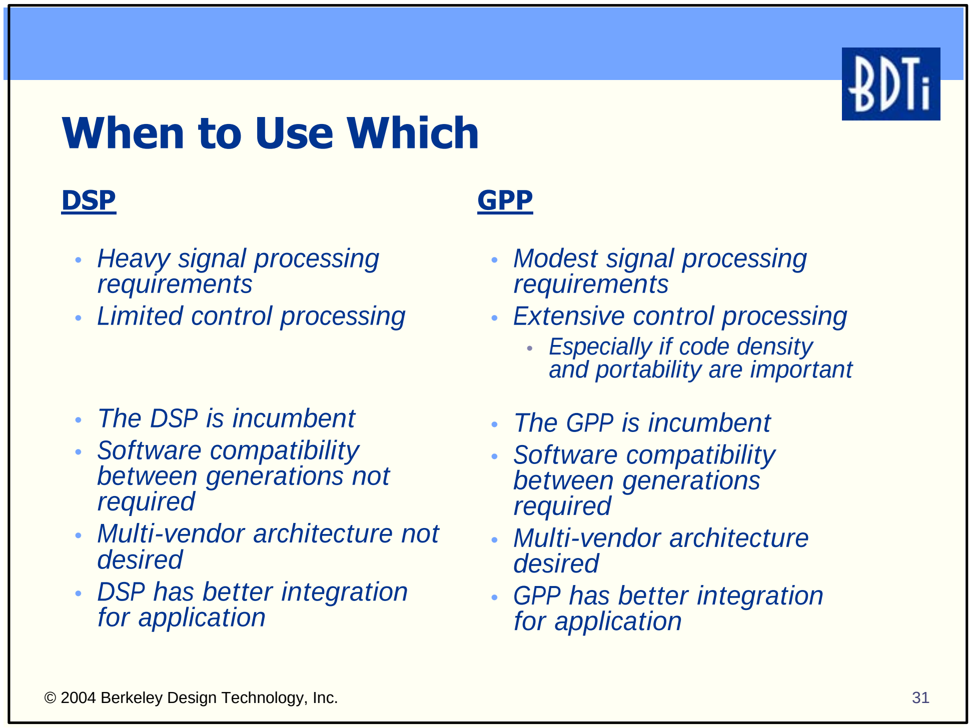
© 2004 Berkeley Design Technology, Inc. 29



Comparing Performance

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

© 2004 Berkeley Design Technology, Inc. 30



When to Use Which

DSP

- Heavy signal processing requirements
- Limited control processing

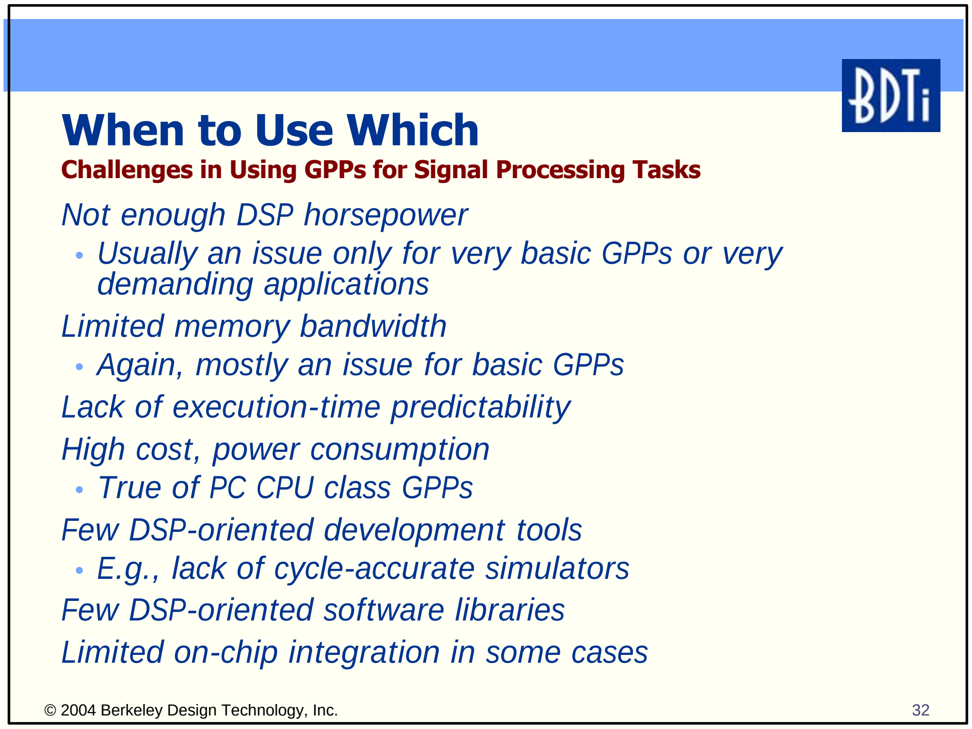
- The DSP is incumbent
- Software compatibility between generations not required
- Multi-vendor architecture not desired
- DSP has better integration for application

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

31



When to Use Which

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

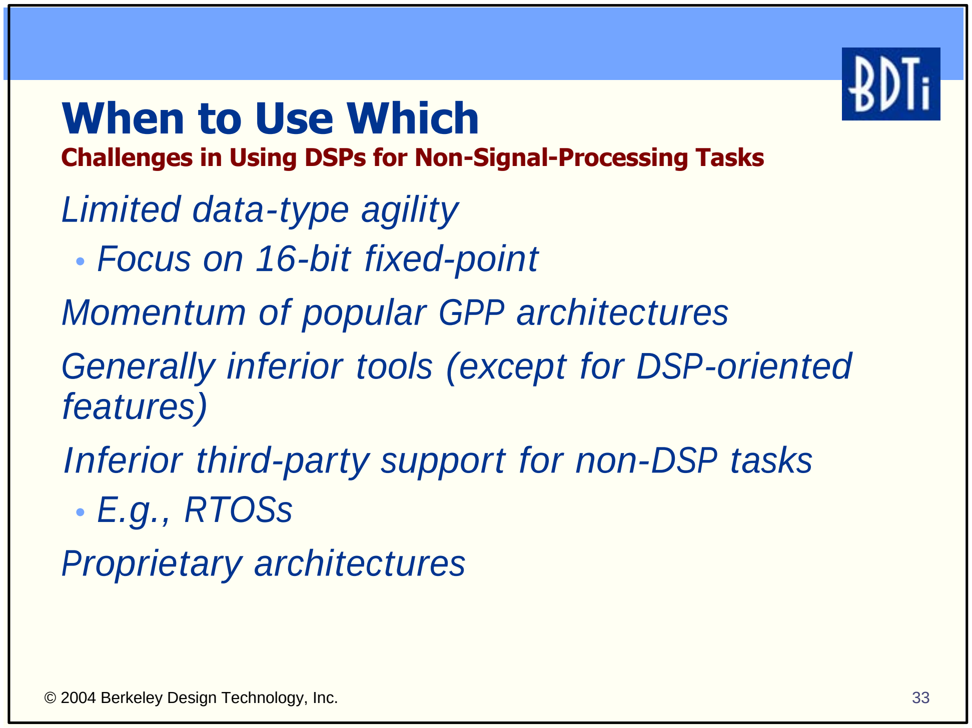
Few DSP-oriented development tools

- E.g., lack of cycle-accurate simulators

Few DSP-oriented software libraries

Limited on-chip integration in some cases

32



When to Use Which

Challenges in Using DSPs for Non-Signal-Processing Tasks

Limited data-type agility

- *Focus on 16-bit fixed-point*

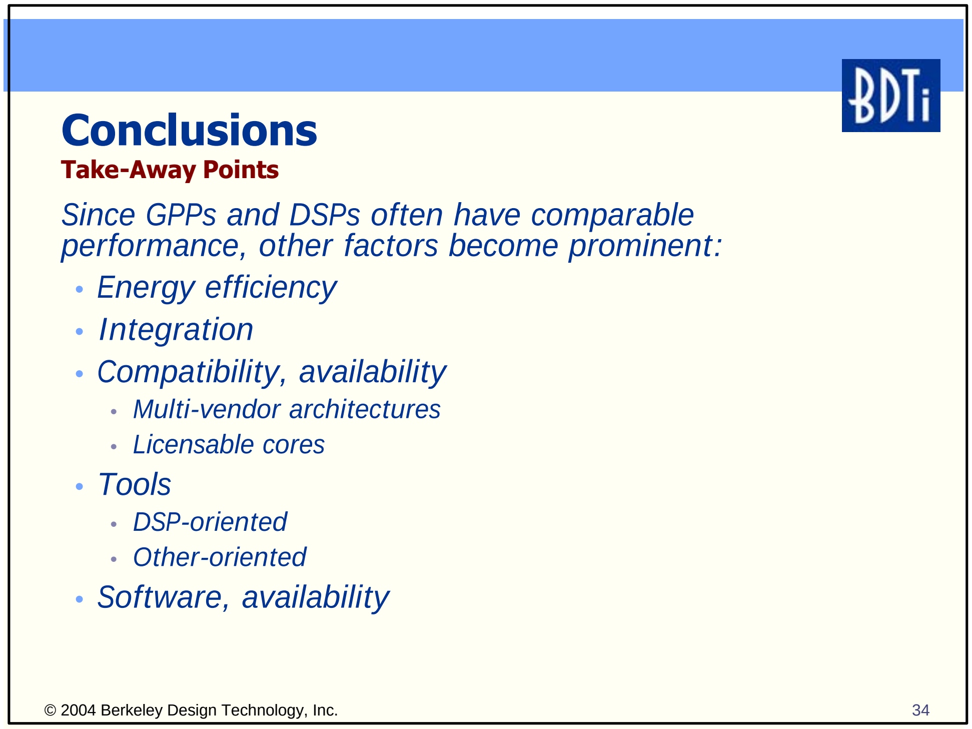
Momentum of popular GPP architectures

Generally inferior tools (except for DSP-oriented features)

Inferior third-party support for non-DSP tasks

- *E.g., RTOSs*

Proprietary architectures

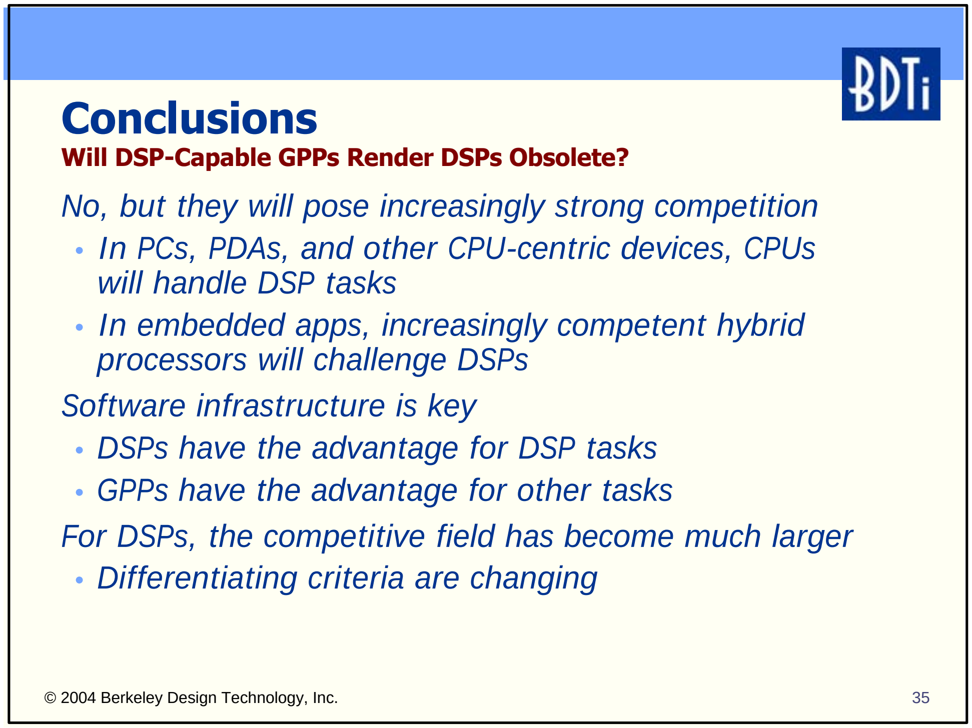


Conclusions

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

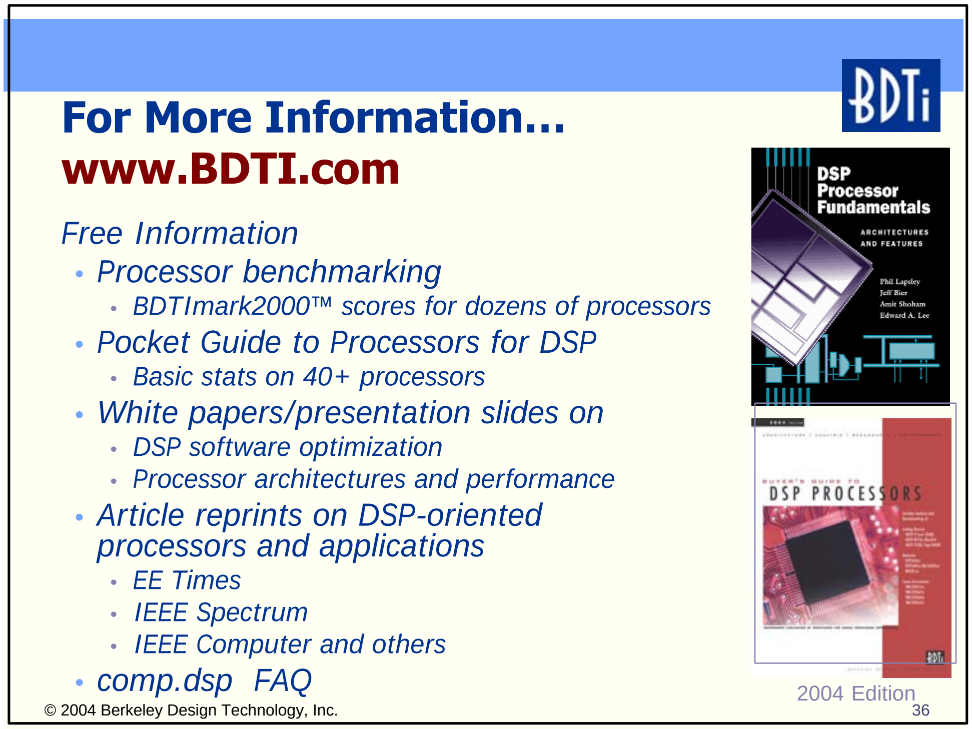
- *In PCs, PDAs, and other CPU-centric devices, CPUs will handle DSP tasks*
- *In embedded apps, increasingly competent hybrid processors will challenge DSPs*

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*

DSP Processor Fundamentals
Architectures and Features
Phil Lapsley
Jeff Bier
Amit Shoham
Edward A. Lee
Buyer's Guide to DSP Processors

2004 Edition