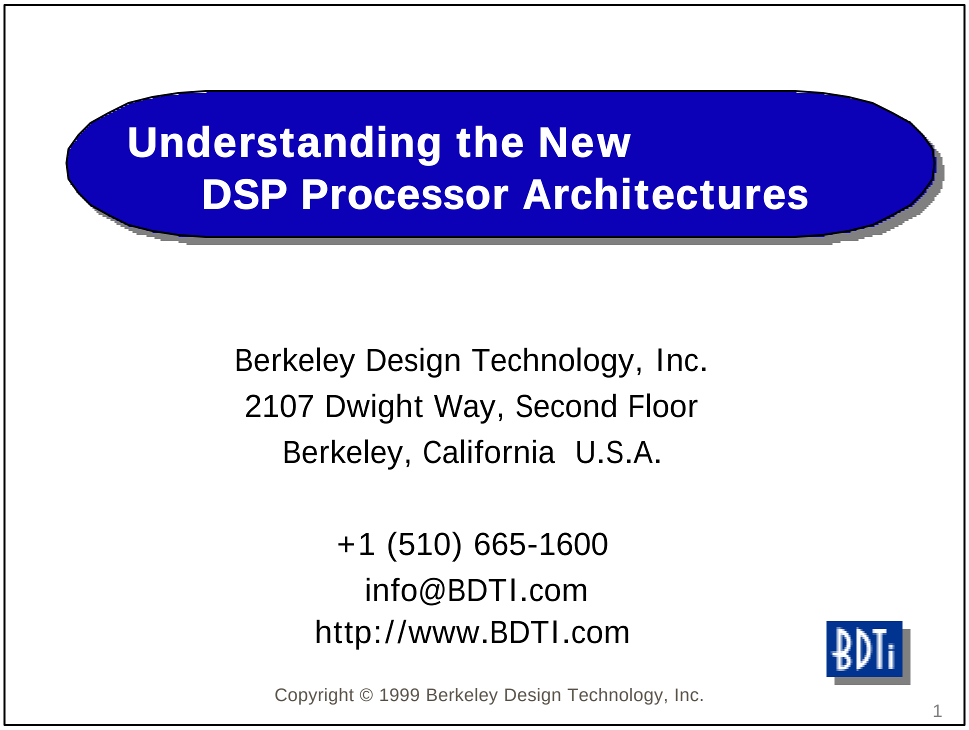
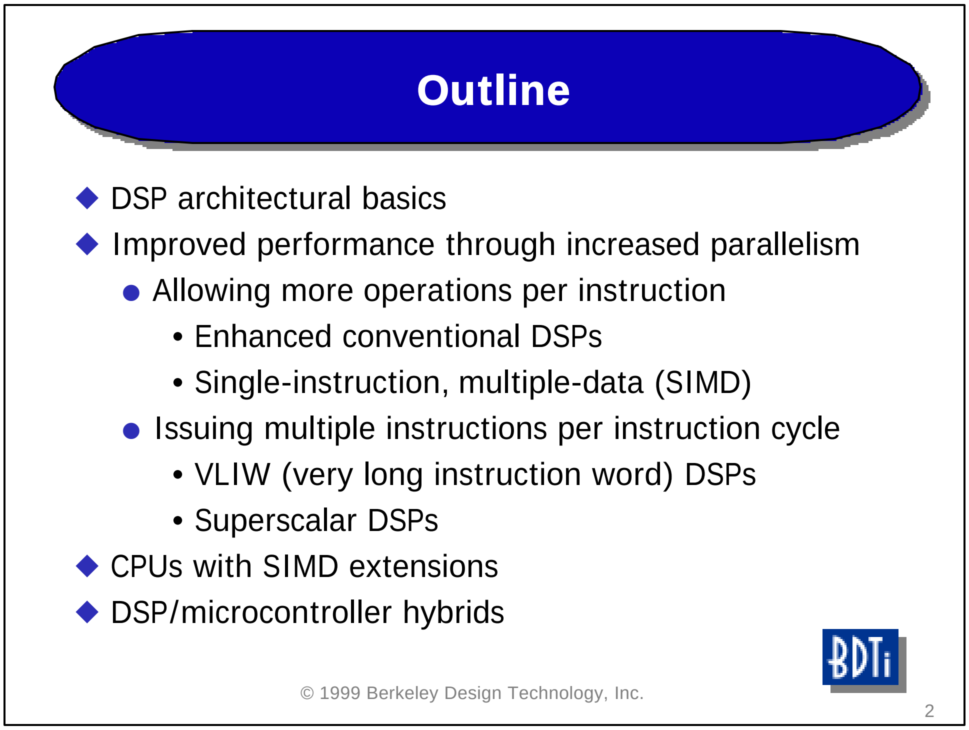


Understanding the New DSP Processor Architectures

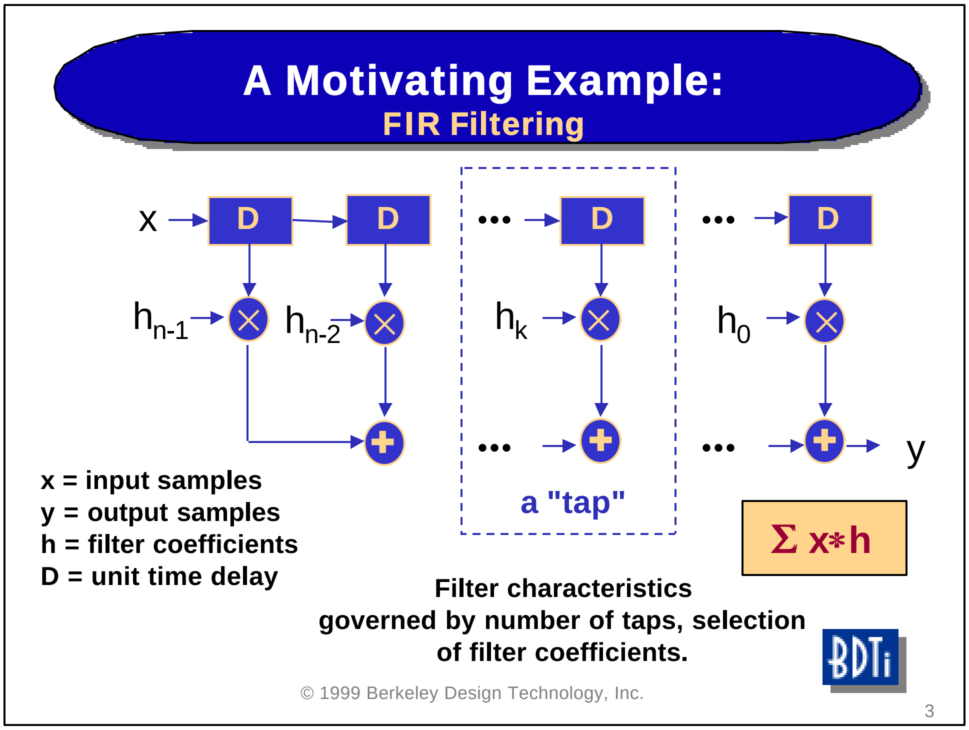
Berkeley Design Technology, Inc.
2107 Dwight Way, Second Floor
Berkeley, California U.S.A.

+1 (510) 665-1600
info@BDTI.com
http://www.BDTI.com



Outline

- DSP architectural basics
- Improved performance through increased parallelism
  - Allowing more operations per instruction
    - Enhanced conventional DSPs
    - Single-instruction, multiple-data (SIMD)
  - Issuing multiple instructions per instruction cycle
    - VLIW (very long instruction word) DSPs
    - Superscalar DSPs
- CPUs with SIMD extensions
- DSP/microcontroller hybrids



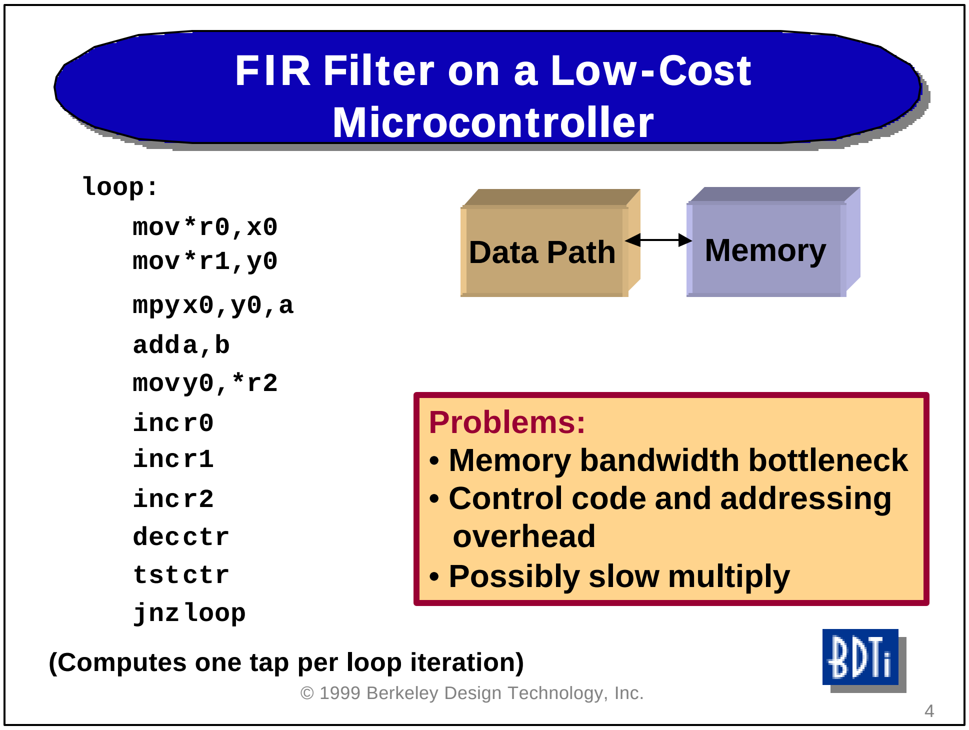
A Motivating Example:
FIR Filtering

x = input samples
y = output samples
h = filter coefficients
D = unit time delay

Filter characteristics governed by number of taps, selection of filter coefficients.

© 1999 Berkeley Design Technology, Inc.

3



FIR Filter on a Low-Cost Microcontroller

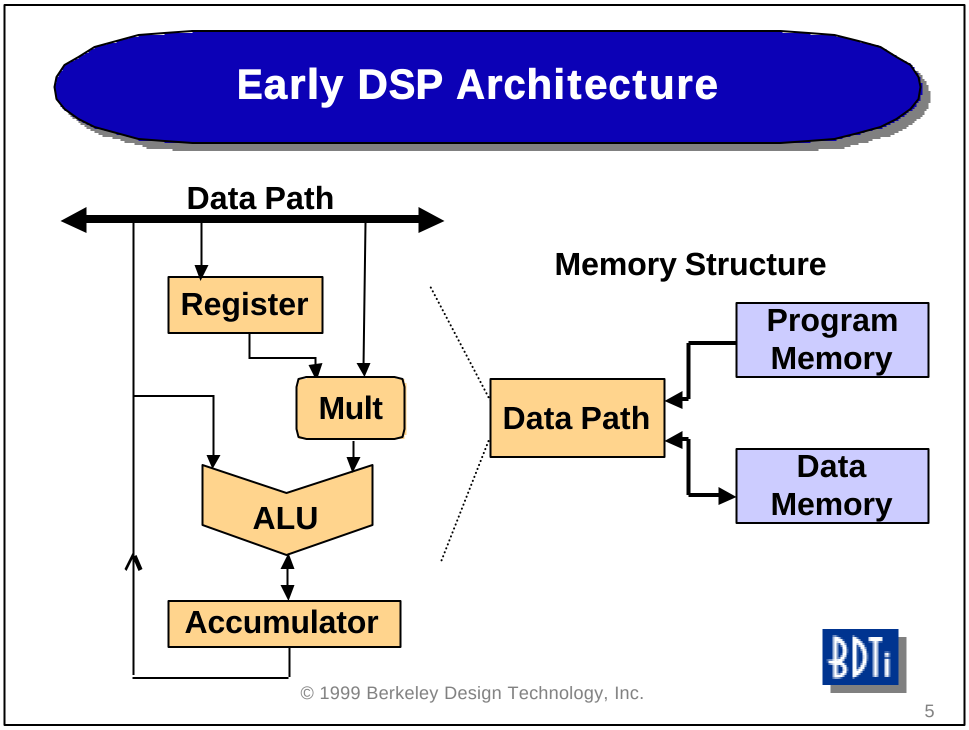
```
loop:
    mov*r0,x0
    mov*r1,y0
    mpyx0,y0,a
    adda,b
    movy0,*r2
    incr0
    incr1
    incr2
    decctr
    tstctr
    jnz loop
```

Problems:
- Memory bandwidth bottleneck
- Control code and addressing overhead
- Possibly slow multiply

(Computes one tap per loop iteration)

© 1999 Berkeley Design Technology, Inc.

4



Early DSP Architecture

Data Path

Register

Mult

ALU

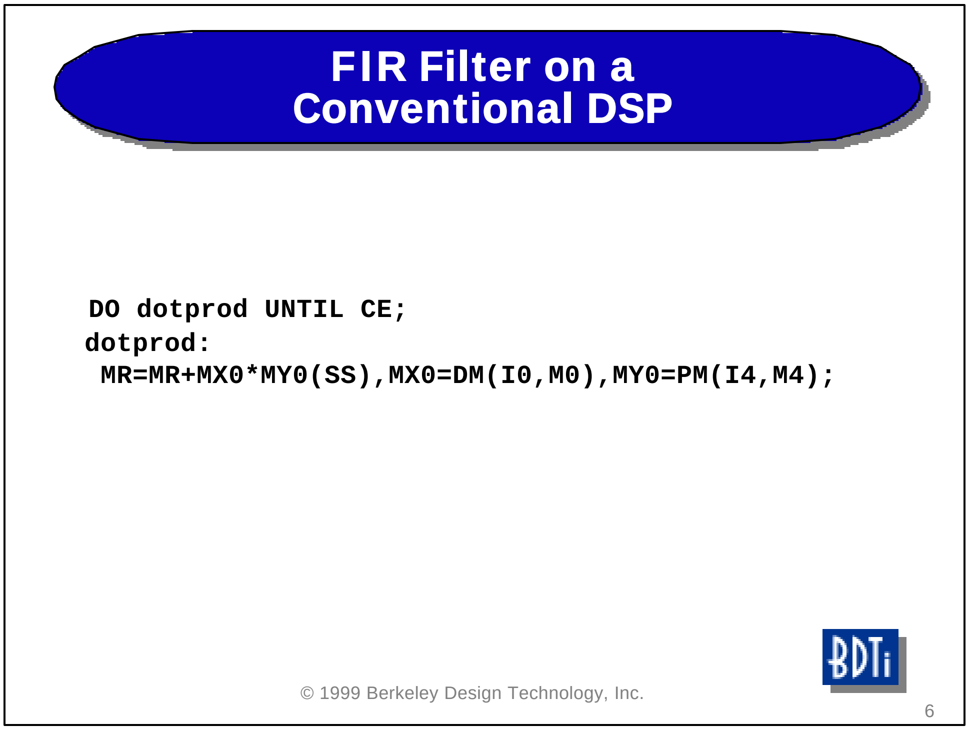
Accumulator

Memory Structure

Data Path

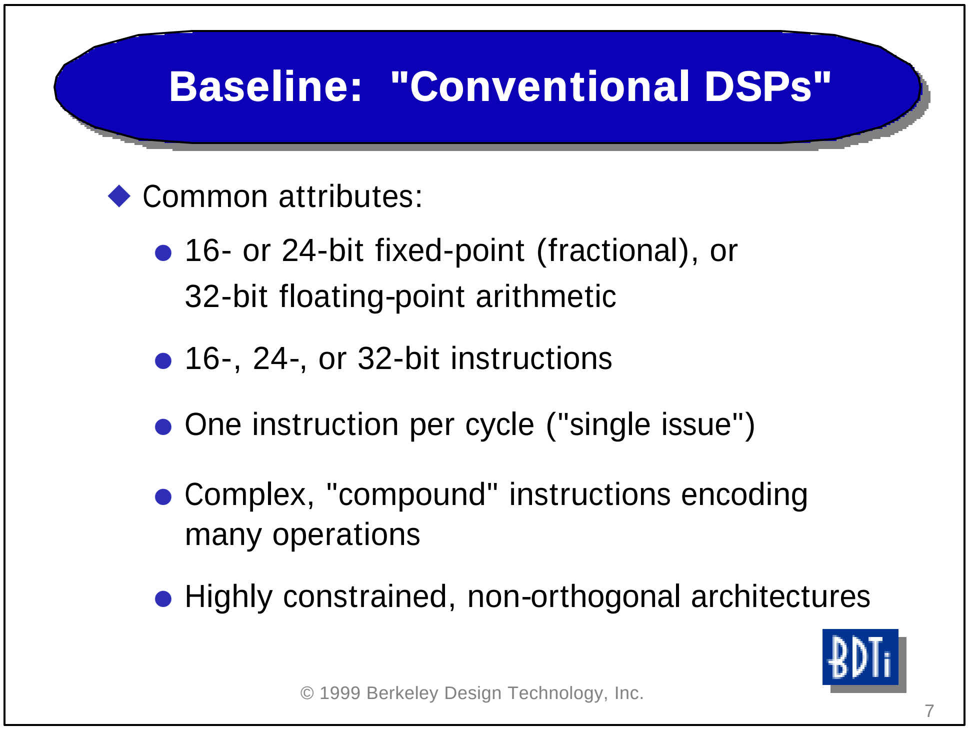
Program Memory

Data Memory



FIR Filter on a Conventional DSP

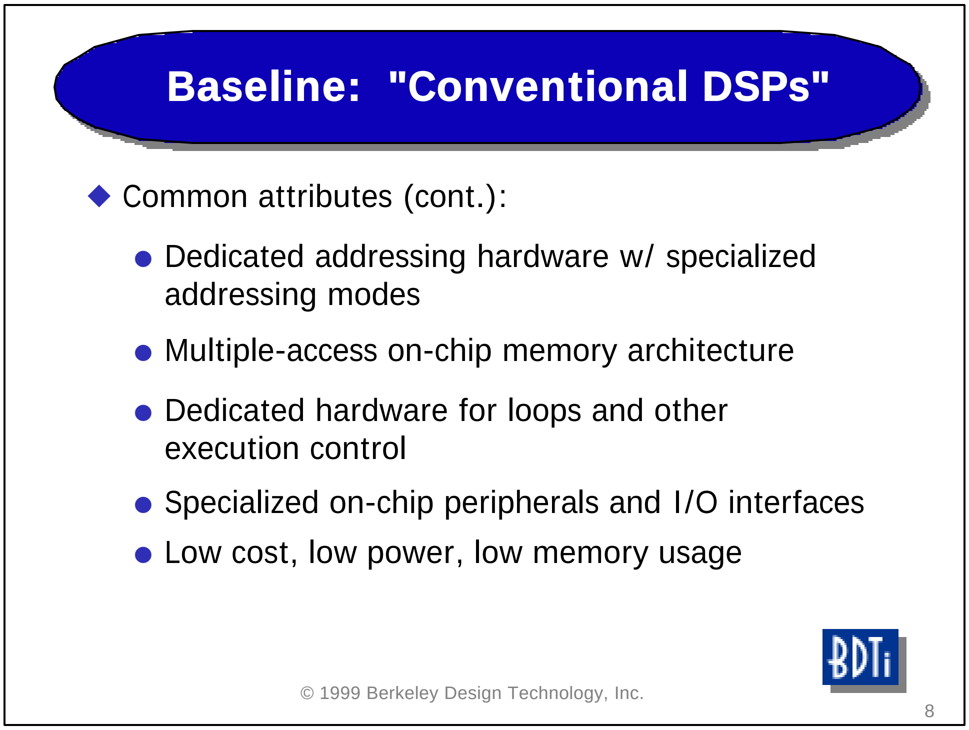
```
 DO dotprod UNTIL CE;
dotprod:
 MR=MR+MX0*MY0(SS),MX0=DM(I0,M0),MY0=PM(I4,M4);
```



Baseline: "Conventional DSPs"

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

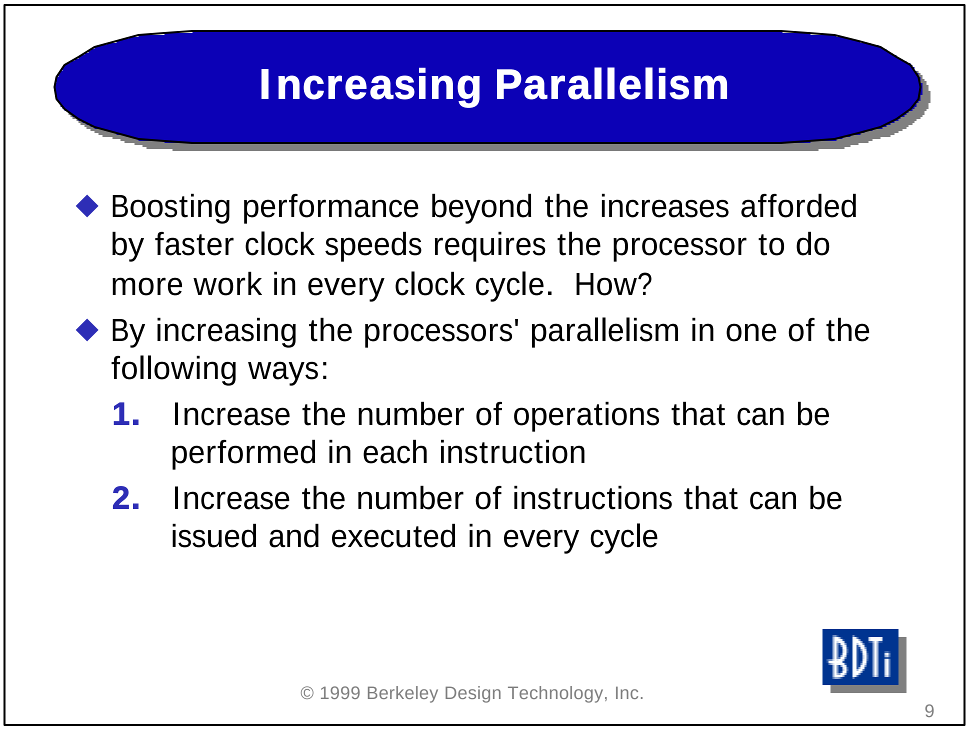
7



Baseline: "Conventional DSPs"

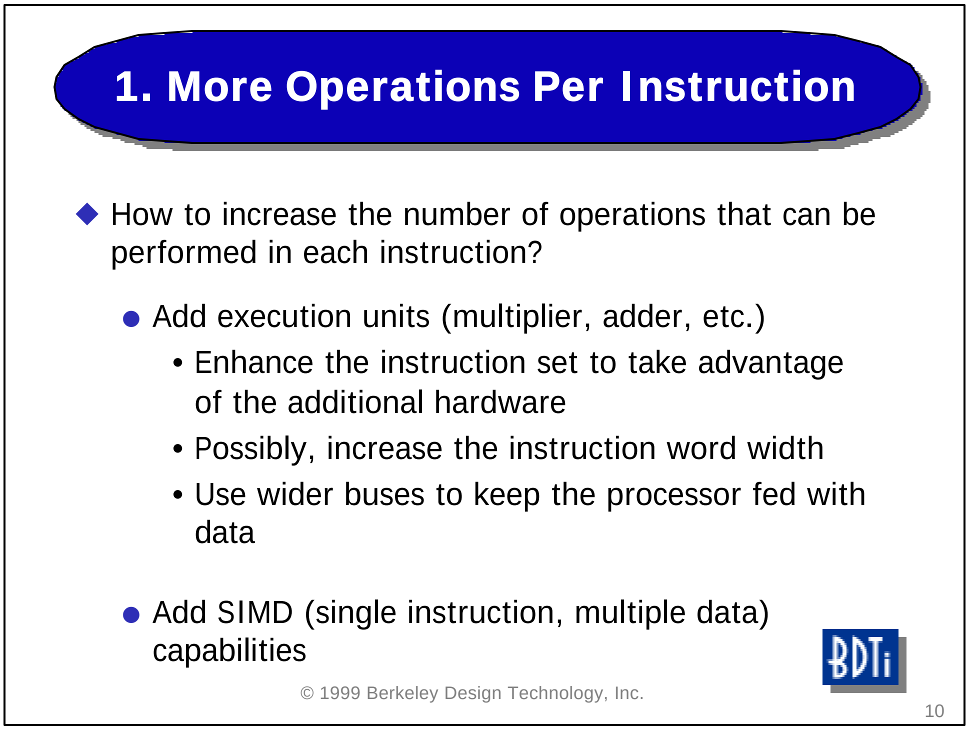
- Common attributes (cont.):
  - Dedicated addressing hardware w/ specialized addressing modes
  - Multiple-access on-chip memory architecture
  - Dedicated hardware for loops and other execution control
  - Specialized on-chip peripherals and I/O interfaces
  - Low cost, low power, low memory usage

8



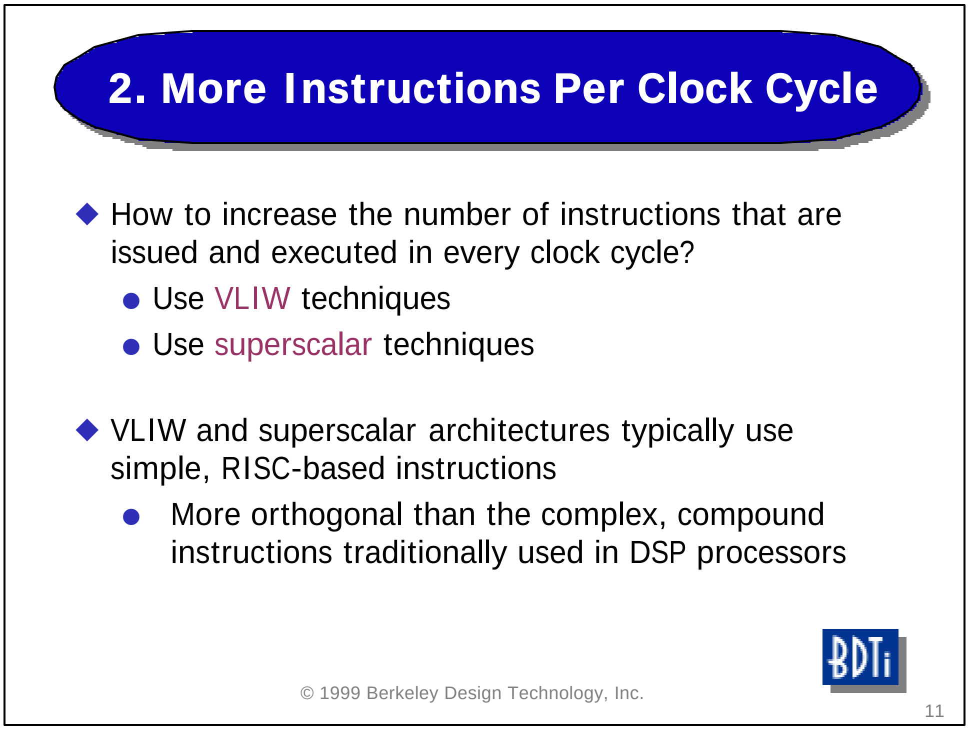
Increasing Parallelism

- Boosting performance beyond the increases afforded by faster clock speeds requires the processor to do more work in every clock cycle. How?
- By increasing the processors' parallelism in one of the following ways:
  1. Increase the number of operations that can be performed in each instruction
  2. Increase the number of instructions that can be issued and executed in every cycle



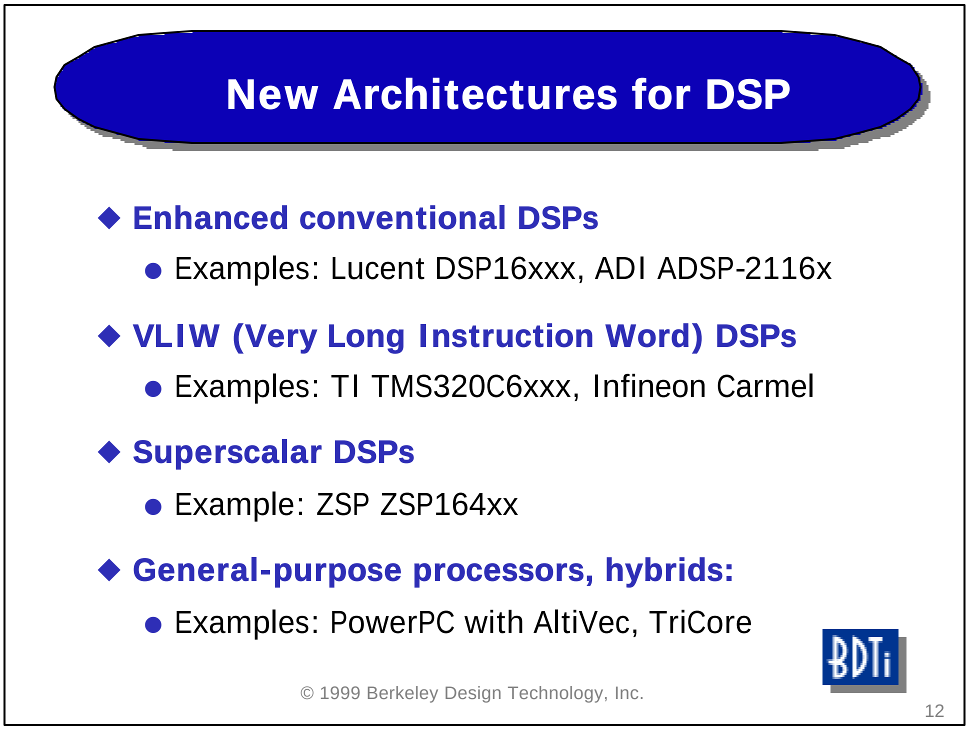
1. More Operations Per Instruction

- How to increase the number of operations that can be performed in each instruction?
  - Add execution units (multiplier, adder, etc.)
    - Enhance the instruction set to take advantage of the additional hardware
    - Possibly, increase the instruction word width
    - Use wider buses to keep the processor fed with data
  - Add SIMD (single instruction, multiple data) capabilities



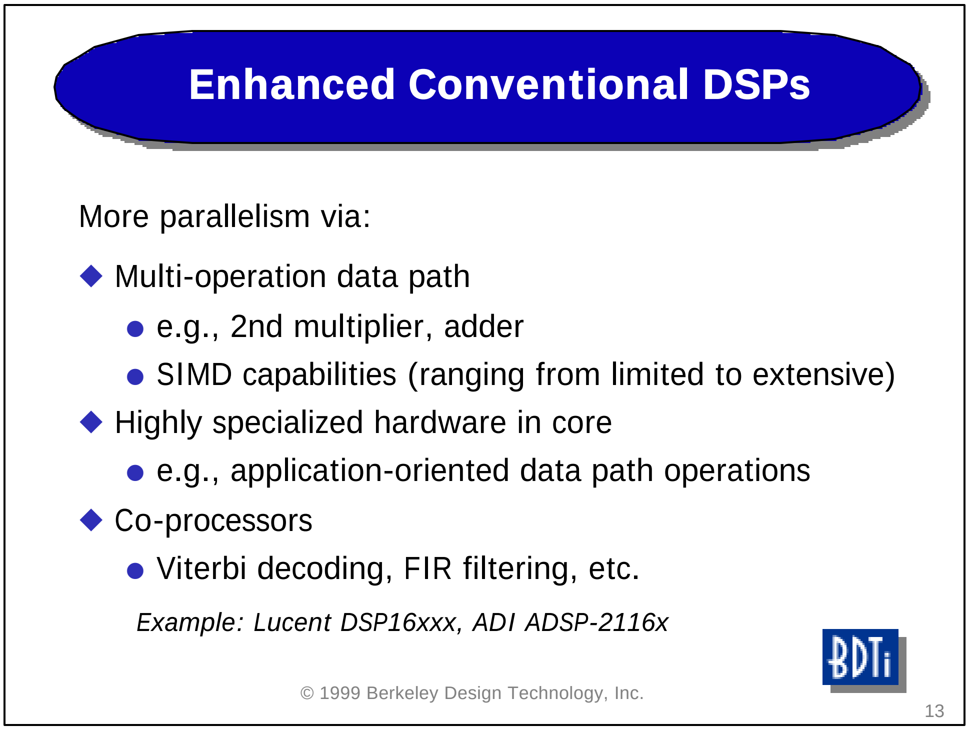
2. More Instructions Per Clock Cycle

- How to increase the number of instructions that are issued and executed in every clock cycle?
  - Use VLIW techniques
  - Use superscalar techniques
- VLIW and superscalar architectures typically use simple, RISC-based instructions
  - More orthogonal than the complex, compound instructions traditionally used in DSP processors



New Architectures for DSP

- **Enhanced conventional DSPs**
  - Examples: Lucent DSP16xxx, ADI ADSP-2116x
- **VLIW (Very Long Instruction Word) DSPs**
  - Examples: TI TMS320C6xxx, Infineon Carmel
- **Superscalar DSPs**
  - Example: ZSP ZSP164xx
- **General-purpose processors, hybrids:**
  - Examples: PowerPC with AltiVec, TriCore

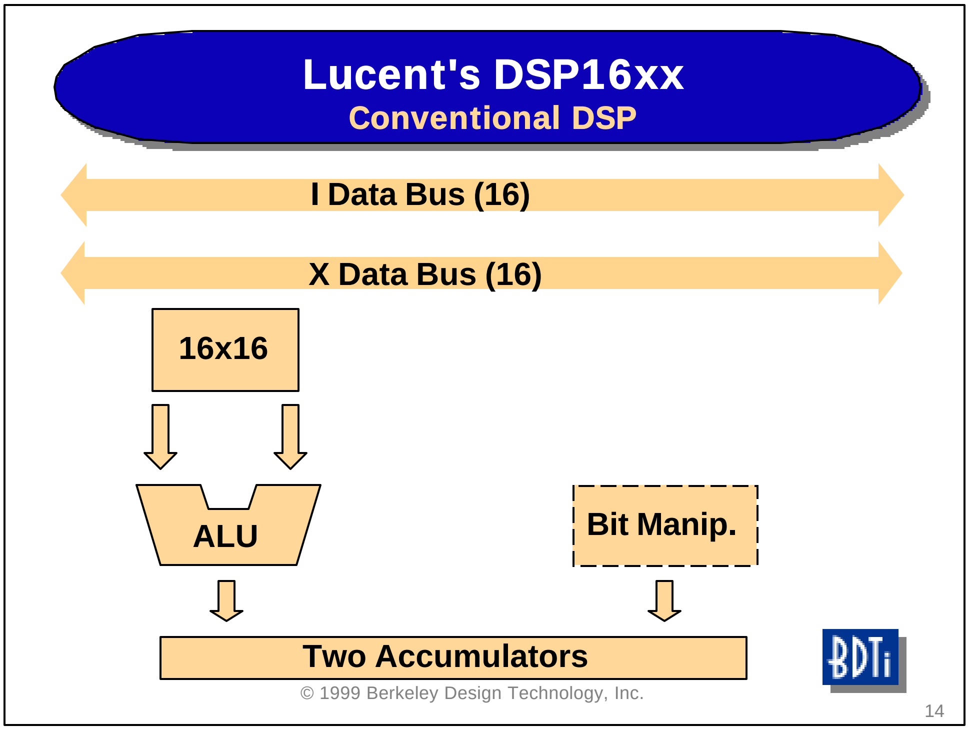


Enhanced Conventional DSPs

More parallelism via:

- Multi-operation data path
  - e.g., 2nd multiplier, adder
  - SIMD capabilities (ranging from limited to extensive)
- Highly specialized hardware in core
  - e.g., application-oriented data path operations
- Co-processors
  - Viterbi decoding, FIR filtering, etc.

Example: Lucent DSP16xxx, ADI ADSP-2116x



Lucent's DSP16xx

Conventional DSP

I Data Bus (16)

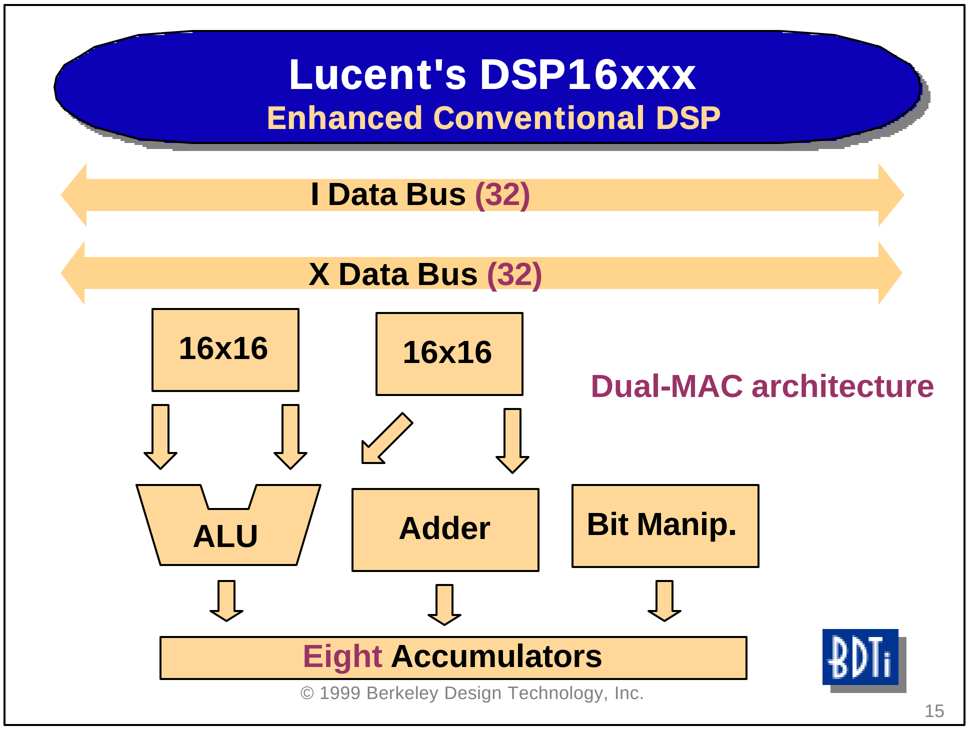
X Data Bus (16)

16x16

ALU

Bit Manip.

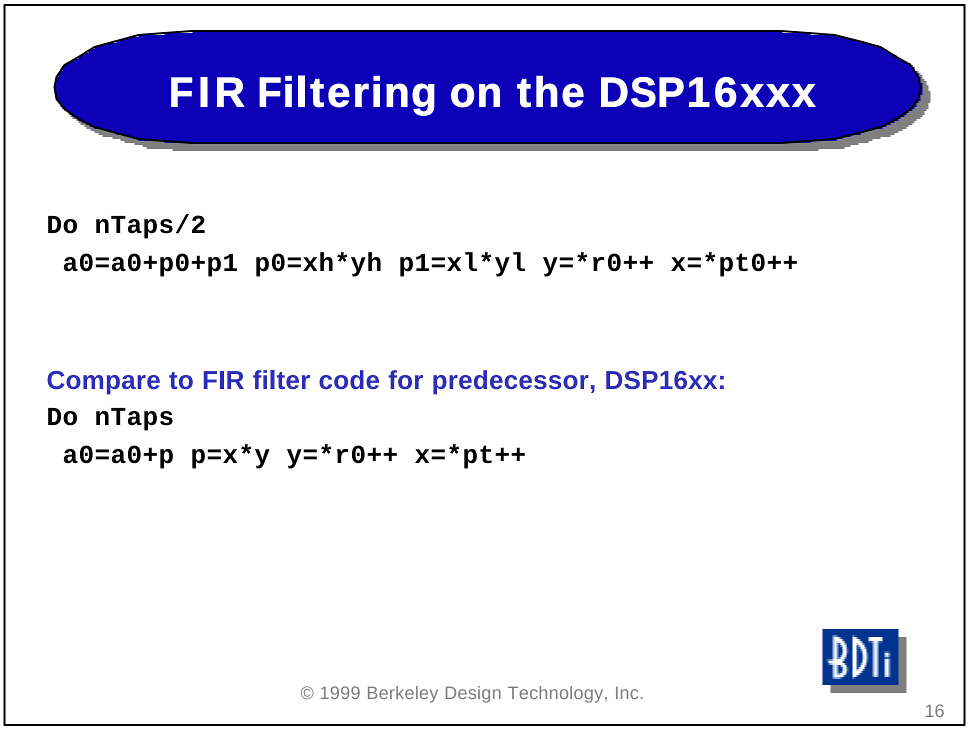
Two Accumulators



Lucent's DSP16xxx

Enhanced Conventional DSP

15



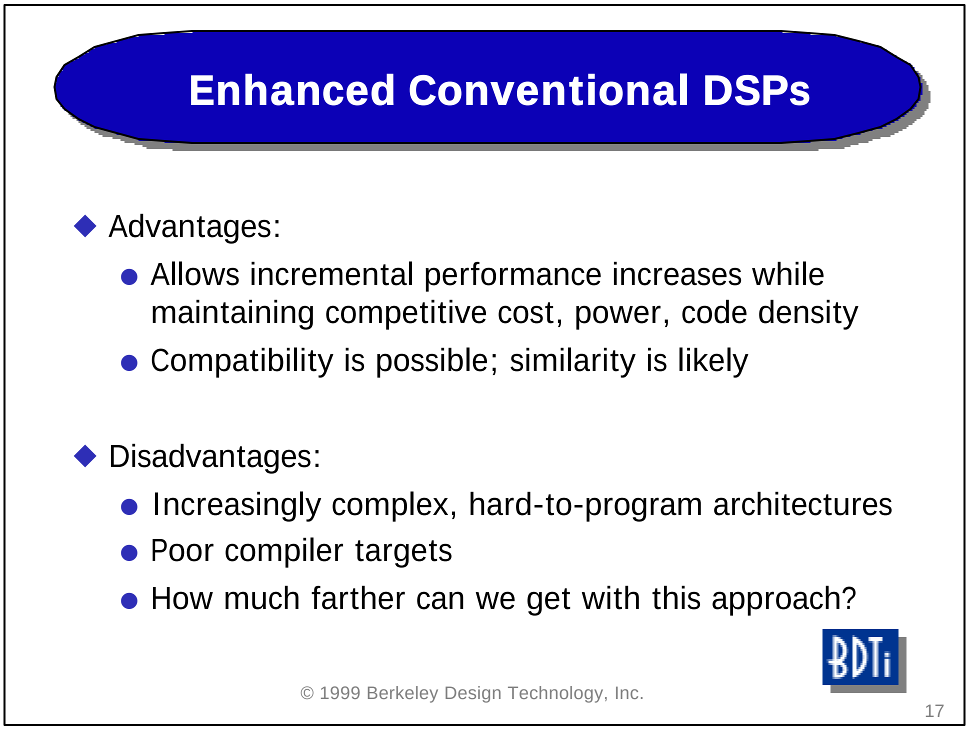
FIR Filtering on the DSP16xxx

```
Do nTaps/2
 a0=a0+p0+p1 p0=xh*yh p1=xl*yl y=*r0++ x=*pt0++
```

Compare to FIR filter code for predecessor, DSP16xx:

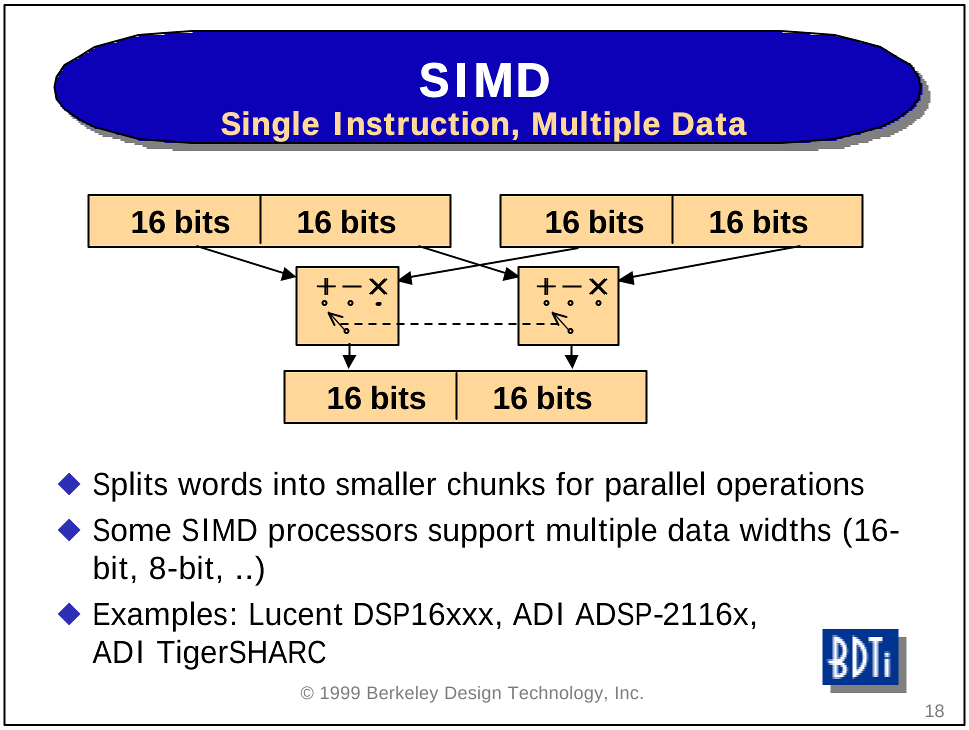
```
Do nTaps
 a0=a0+p y=x*y y=*r0++ x=*pt++
```

16



Enhanced Conventional DSPs

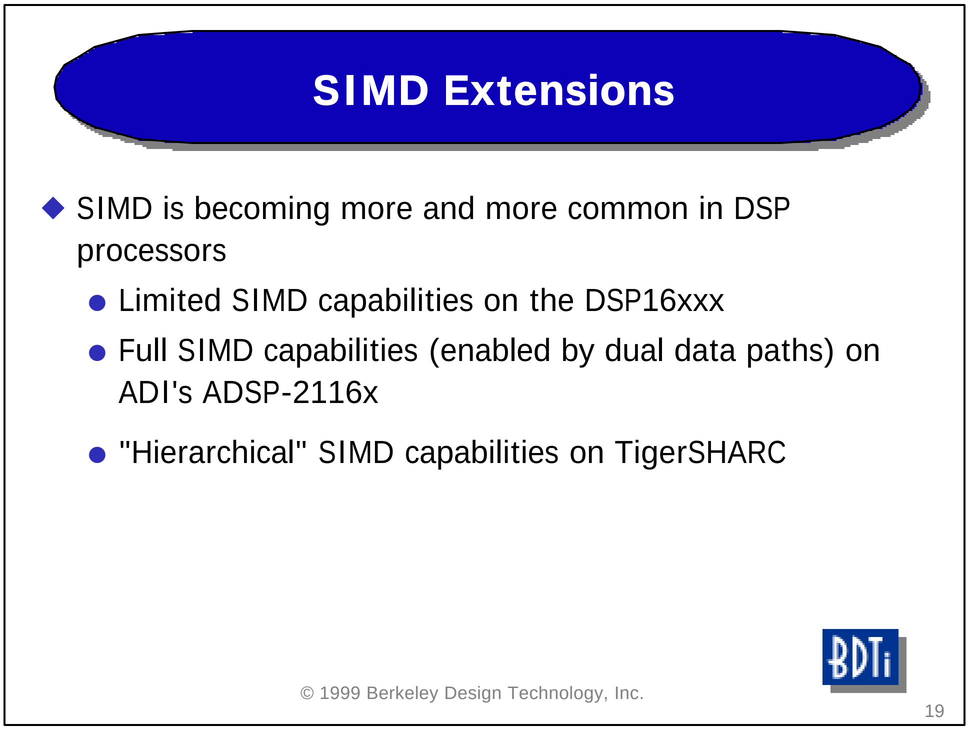
- Advantages:
  - Allows incremental performance increases while maintaining competitive cost, power, code density
  - Compatibility is possible; similarity is likely
- Disadvantages:
  - Increasingly complex, hard-to-program architectures
  - Poor compiler targets
  - How much farther can we get with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



SIMD

Single Instruction, Multiple Data

- Splits words into smaller chunks for parallel operations
- Some SIMD processors support multiple data widths (16-bit, 8-bit, ..)
- Examples: Lucent DSP16xxx, ADI ADSP-2116x, ADI TigerSHARC

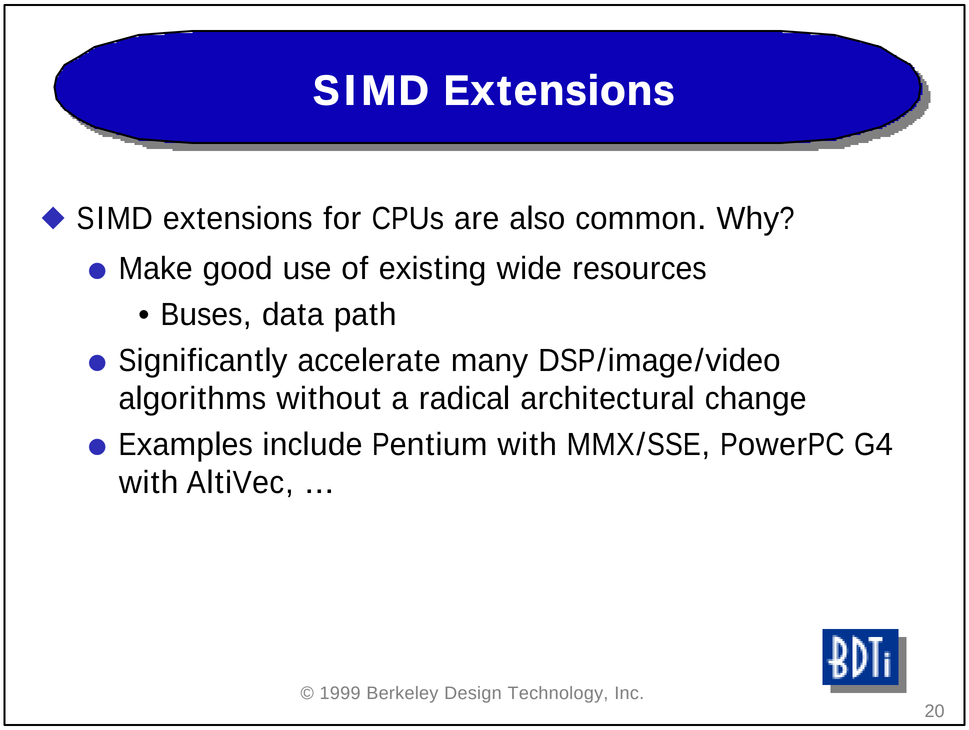


SIMD Extensions

- SIMD is becoming more and more common in DSP processors
  - Limited SIMD capabilities on the DSP16xxx
  - Full SIMD capabilities (enabled by dual data paths) on ADI's ADSP-2116x
  - "Hierarchical" SIMD capabilities on TigerSHARC

© 1999 Berkeley Design Technology, Inc.

19

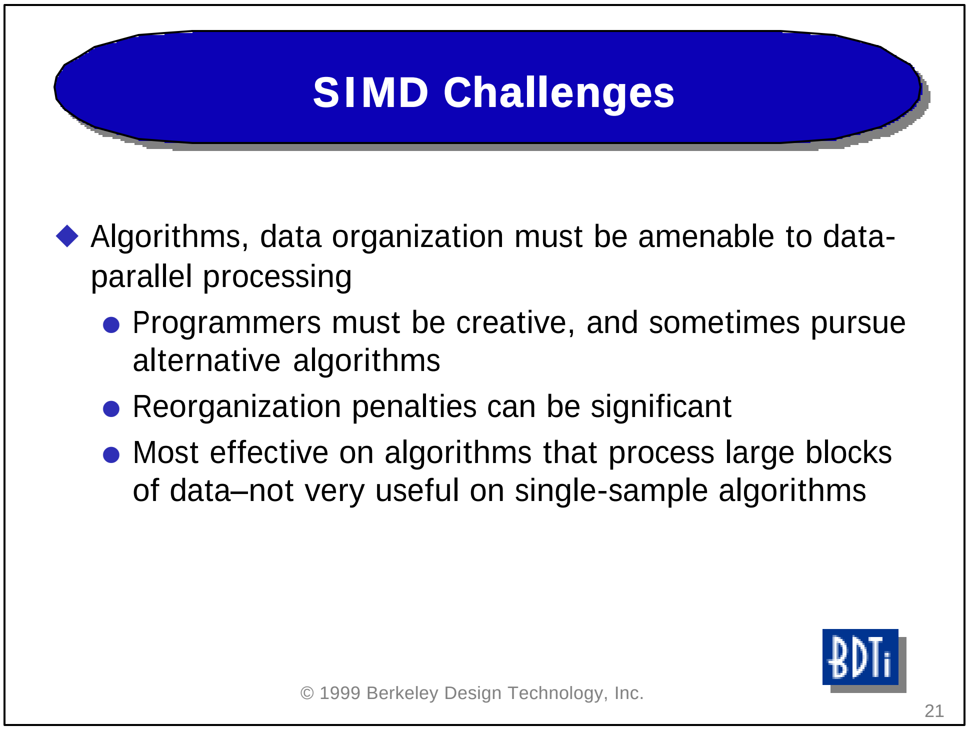


SIMD Extensions

- SIMD extensions for CPUs are also common. Why?
  - Make good use of existing wide resources
    - Buses, data path
  - Significantly accelerate many DSP/image/video algorithms without a radical architectural change
  - Examples include Pentium with MMX/SSE, PowerPC G4 with AltiVec, ...

© 1999 Berkeley Design Technology, Inc.

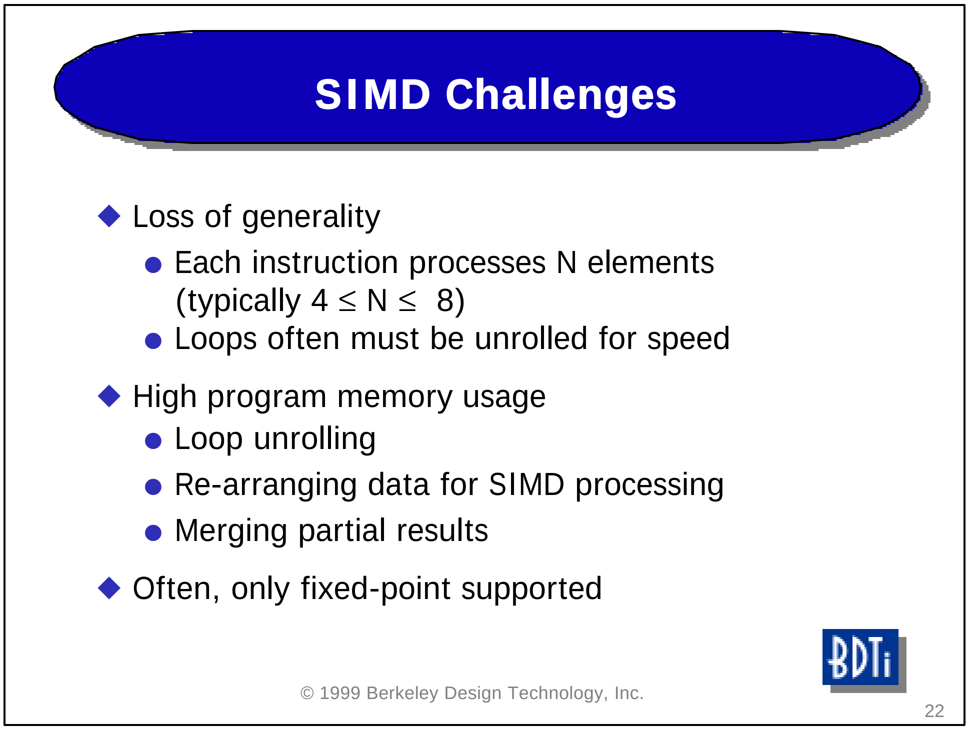
20



SIMD Challenges

- Algorithms, data organization must be amenable to data-parallel processing
  - Programmers must be creative, and sometimes pursue alternative algorithms
  - Reorganization penalties can be significant
  - Most effective on algorithms that process large blocks of data–not very useful on single-sample algorithms

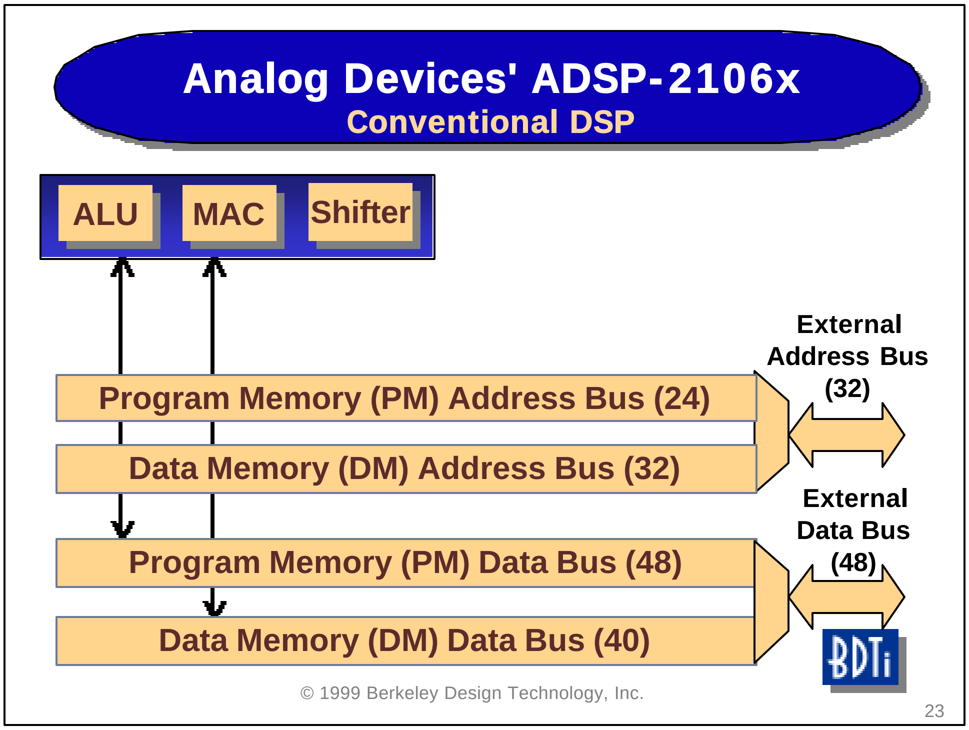
21



SIMD Challenges

- Loss of generality
  - Each instruction processes N elements (typically 4 ≤ N ≤ 8)
  - Loops often must be unrolled for speed
- High program memory usage
  - Loop unrolling
  - Re-arranging data for SIMD processing
  - Merging partial results
- Often, only fixed-point supported

22



Analog Devices' ADSP-2106x

Conventional DSP

ALU

MAC

Shifter

Program Memory (PM) Address Bus (24)

Data Memory (DM) Address Bus (32)

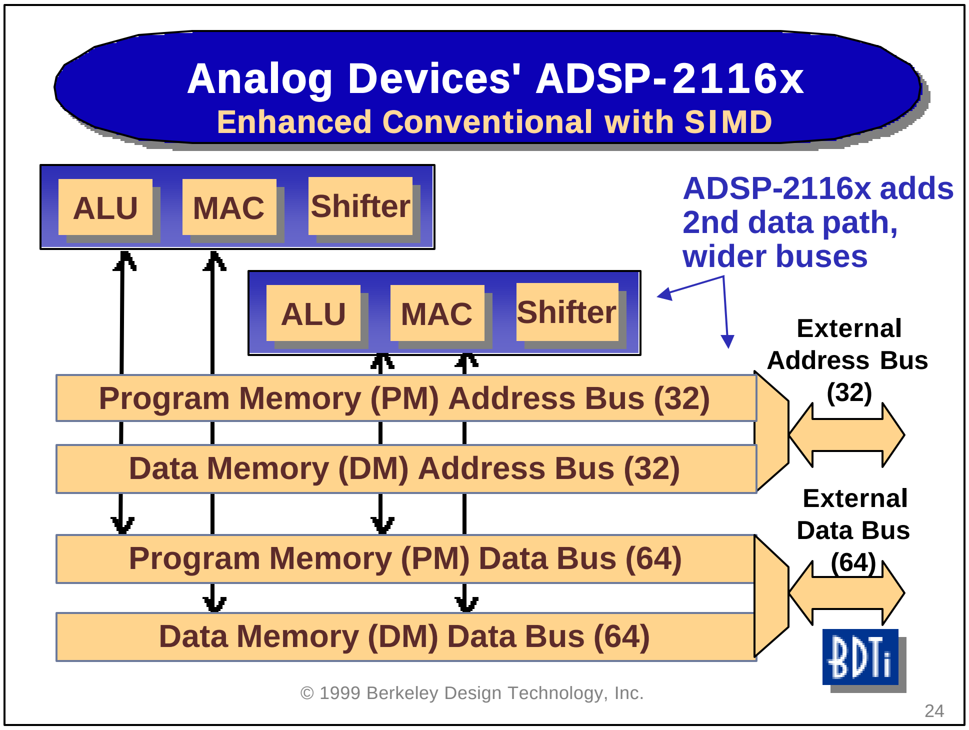
Program Memory (PM) Data Bus (48)

Data Memory (DM) Data Bus (40)

External Address Bus (32)

External Data Bus (48)

BDTi



Analog Devices' ADSP-2116x

Enhanced Conventional with SIMD

ALU

MAC

Shifter

ADSP-2116x adds 2nd data path, wider buses

ALU

MAC

Shifter

Program Memory (PM) Address Bus (32)

Data Memory (DM) Address Bus (32)

Program Memory (PM) Data Bus (64)

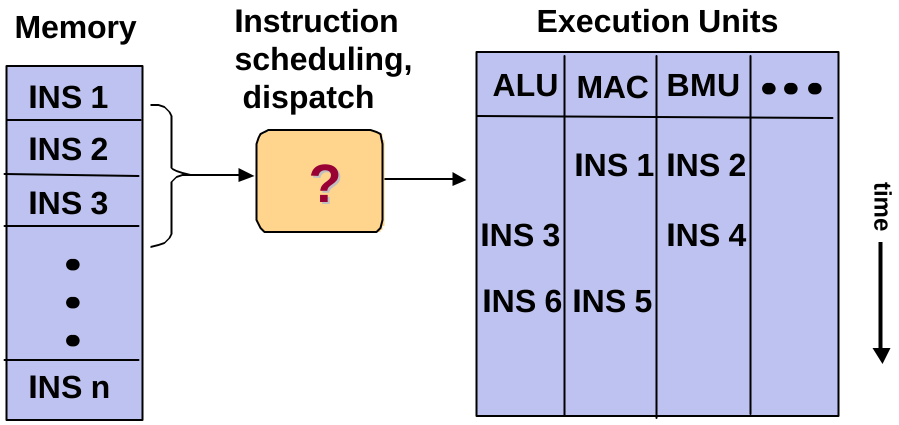
Data Memory (DM) Data Bus (64)

External Address Bus (32)

External Data Bus (64)

BDTi

Superscalar vs VLIW



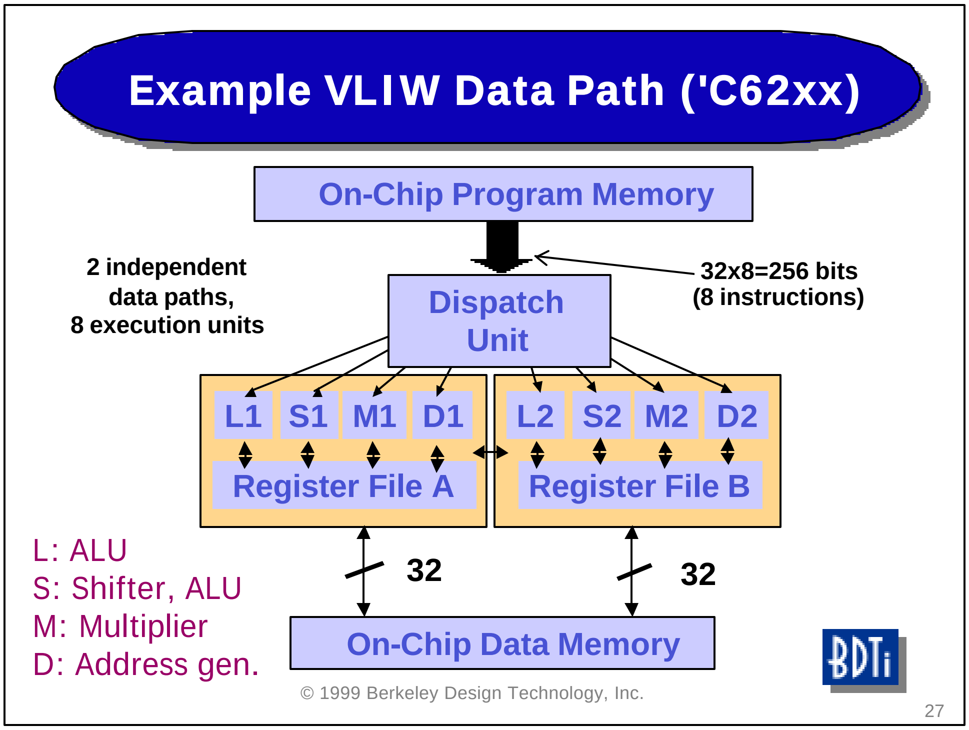
VLIW (Very Long Instruction Word)

Examples of current & upcoming VLIW-based architectures for DSP applications:

- TI TMS320C6xxx, Infineon Carmel, ADI TigerSHARC, StarCore SC140

Characteristics:

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



Example VLIW Data Path ('C62xx)

On-Chip Program Memory

2 independent data paths, 8 execution units

32x8=256 bits (8 instructions)

Dispatch Unit

L1 S1 M1 D1

Register File A

L2 S2 M2 D2

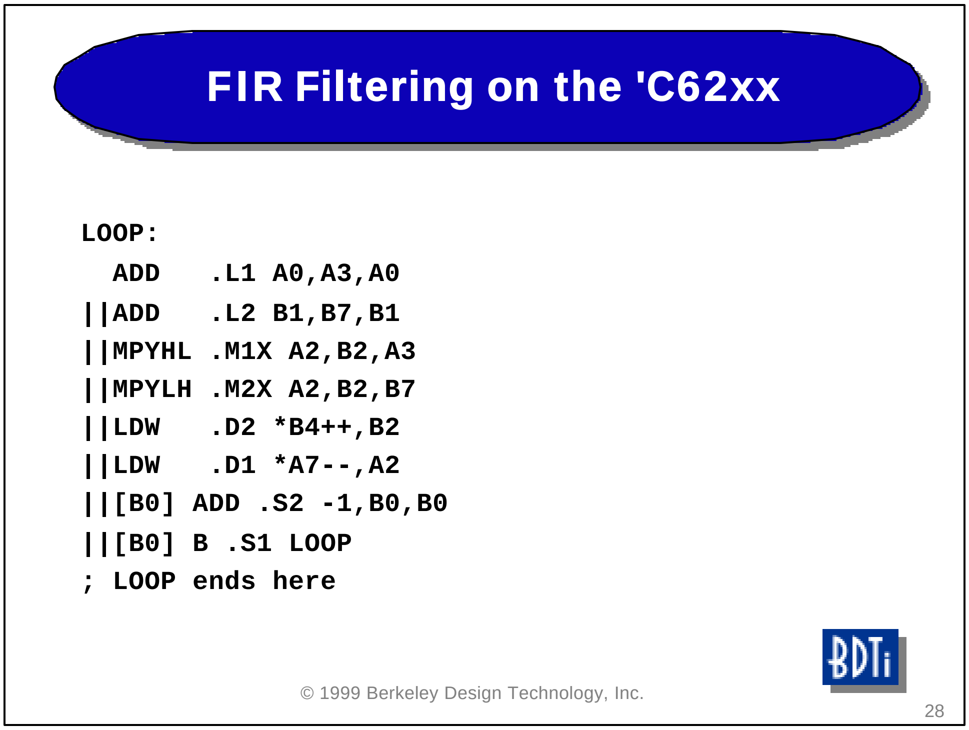
Register File B

32

32

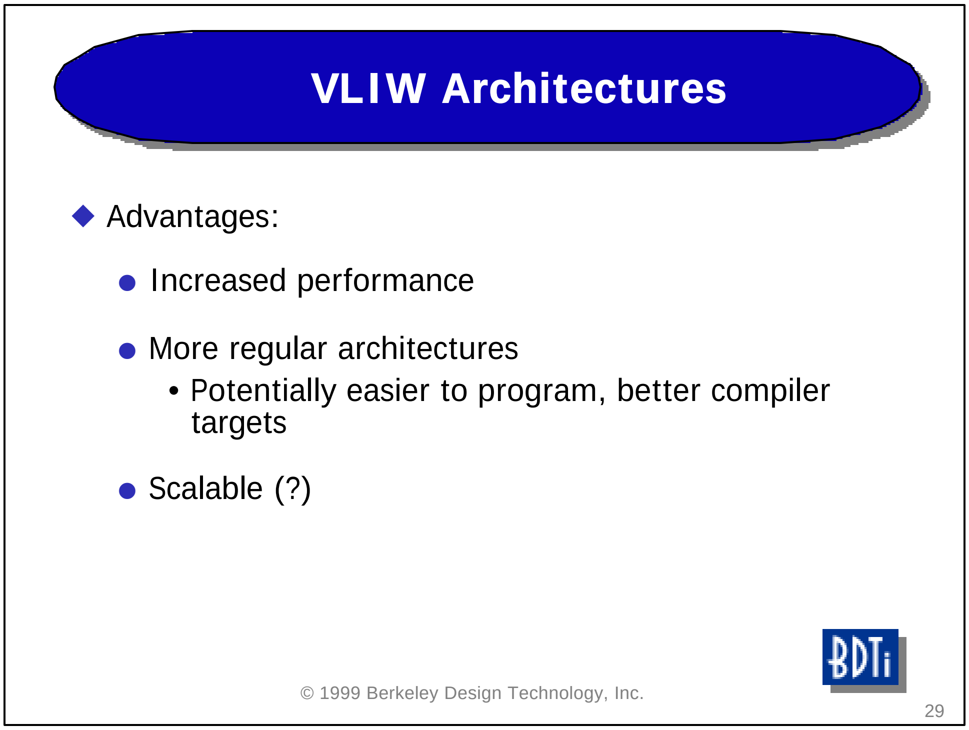
On-Chip Data Memory

L: ALU
S: Shifter, ALU
M: Multiplier
D: Address gen.



FIR Filtering on the 'C62xx

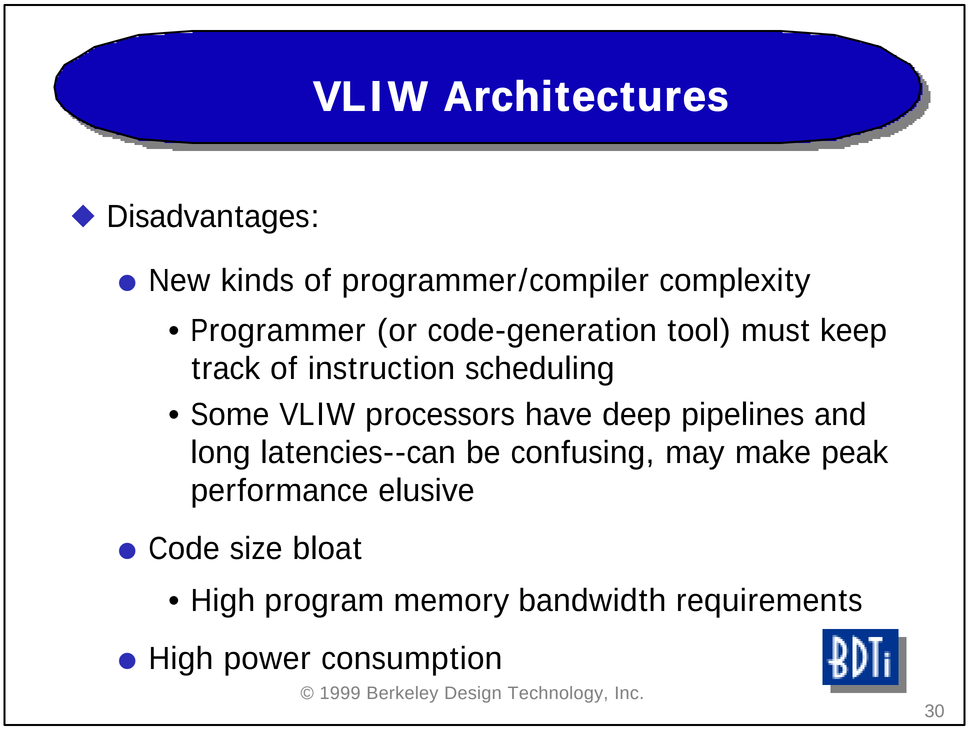
```
LOOP:
  ADD   .L1 A0,A3,A0
||ADD   .L2 B1,B7,B1
||MPYHL .M1X A2,B2,A3
||MPYLH .M2X A2,B2,B7
||LDW   .D2 *B4++,B2
||LDW   .D1 *A7--,A2
||[B0] ADD .S2 -1,B0,B0
||[B0] B .S1 LOOP
; LOOP ends here
```



VLIW Architectures

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

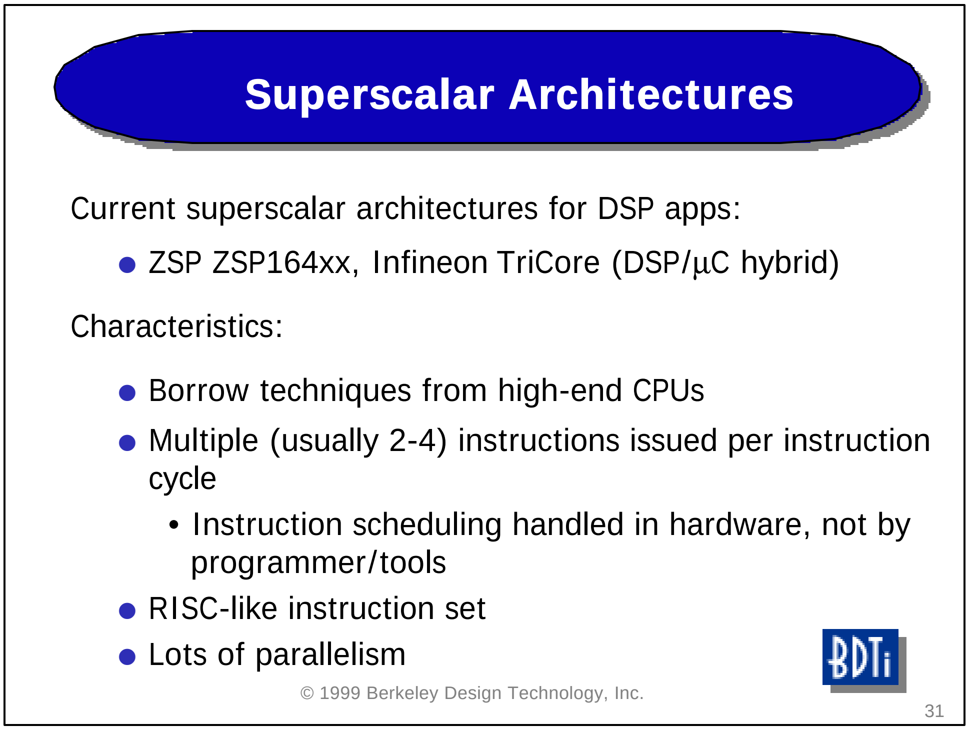
29



VLIW Architectures

- Disadvantages:
  - New kinds of programmer/compiler complexity
    - Programmer (or code-generation tool) must keep track of instruction scheduling
    - Some VLIW processors have deep pipelines and long latencies--can be confusing, may make peak performance elusive
  - Code size bloat
    - High program memory bandwidth requirements
  - High power consumption

30



Superscalar Architectures

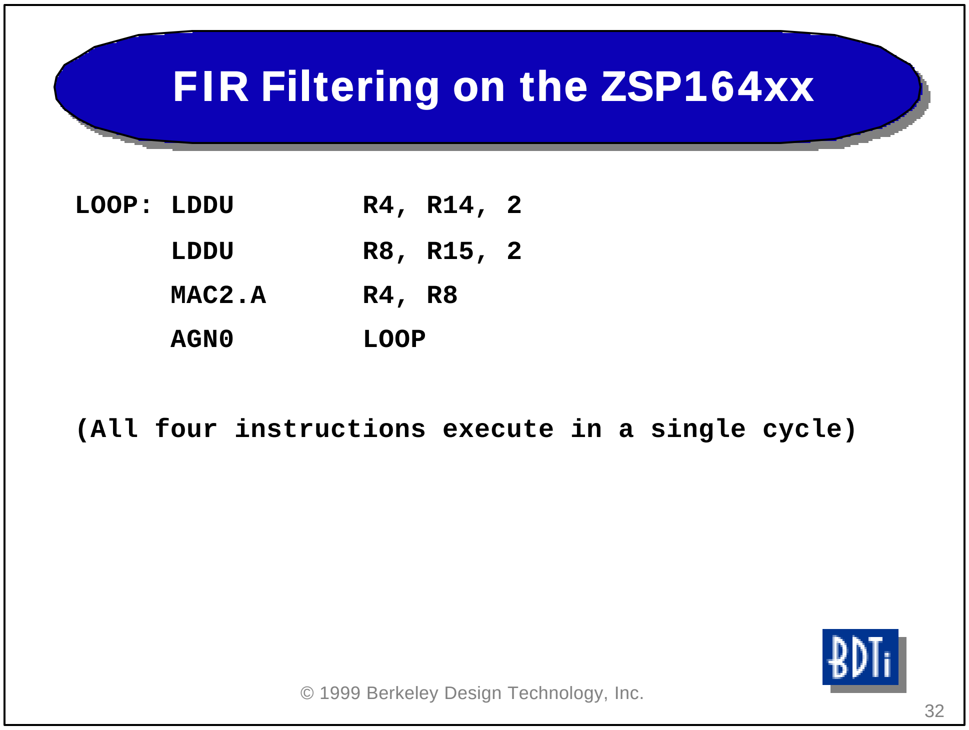
Current superscalar architectures for DSP apps:

- ZSP ZSP164xx, Infineon TriCore (DSP/µC hybrid)

Characteristics:

- Borrow techniques from high-end CPUs
- Multiple (usually 2-4) instructions issued per instruction cycle
  - Instruction scheduling handled in hardware, not by programmer/tools
- RISC-like instruction set
- Lots of parallelism

31

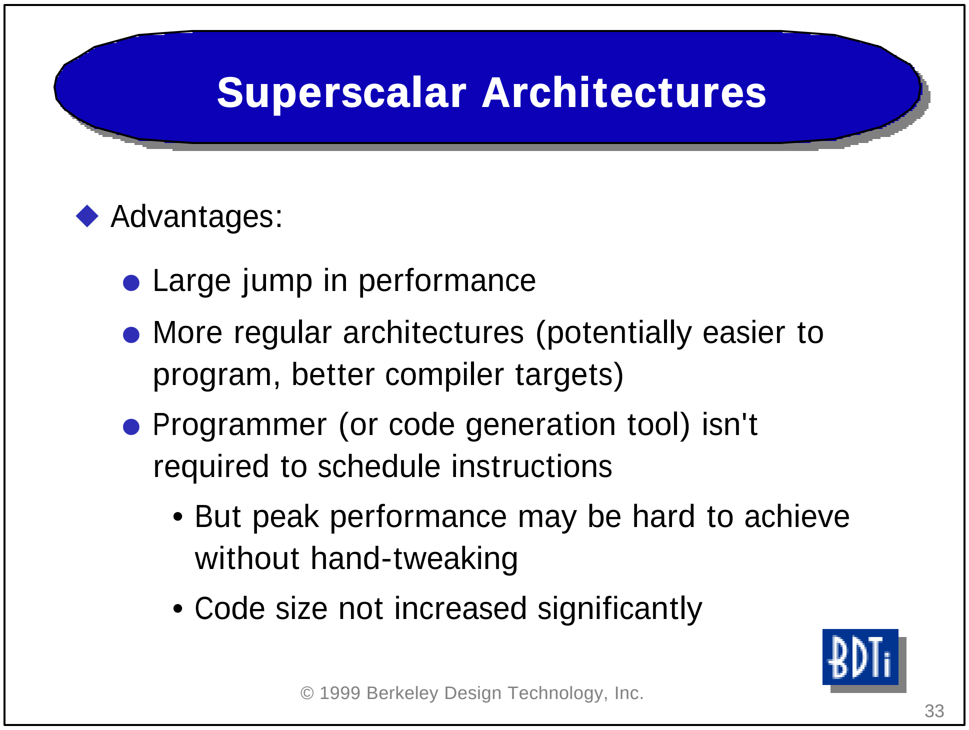


FIR Filtering on the ZSP164xx

```
LOOP: LDDU        R4, R14, 2
      LDDU        R8, R15, 2
      MAC2.A      R4, R8
      AGN0        LOOP
```

(All four instructions execute in a single cycle)

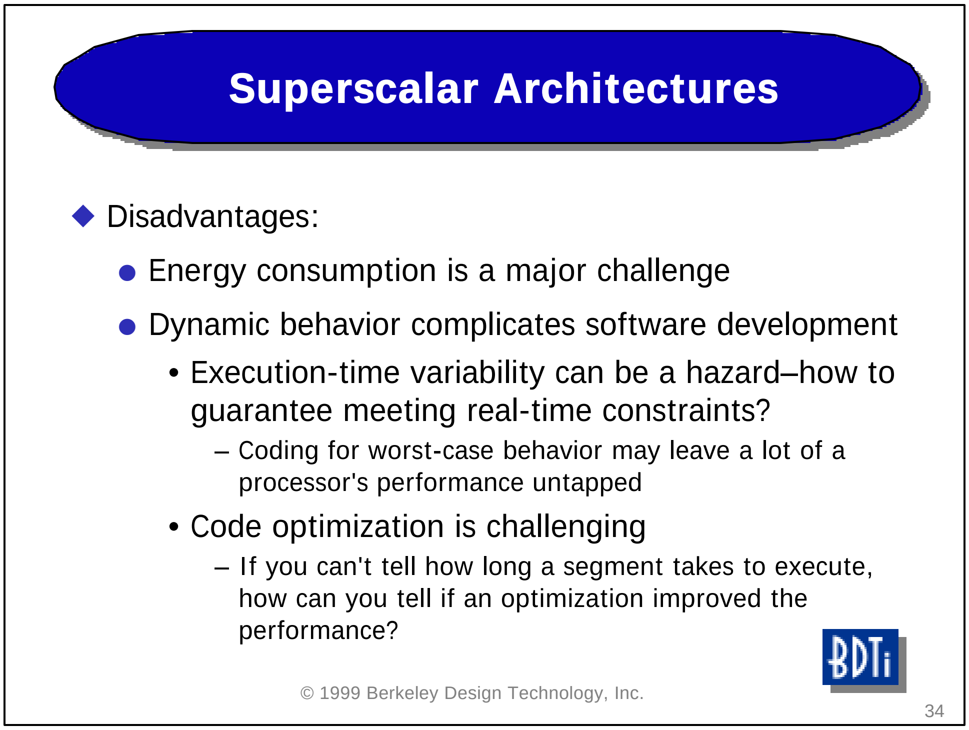
32



Superscalar Architectures

- Advantages:
  - Large jump in performance
  - More regular architectures (potentially easier to program, better compiler targets)
  - Programmer (or code generation tool) isn't required to schedule instructions
    - But peak performance may be hard to achieve without hand-tweaking
    - Code size not increased significantly

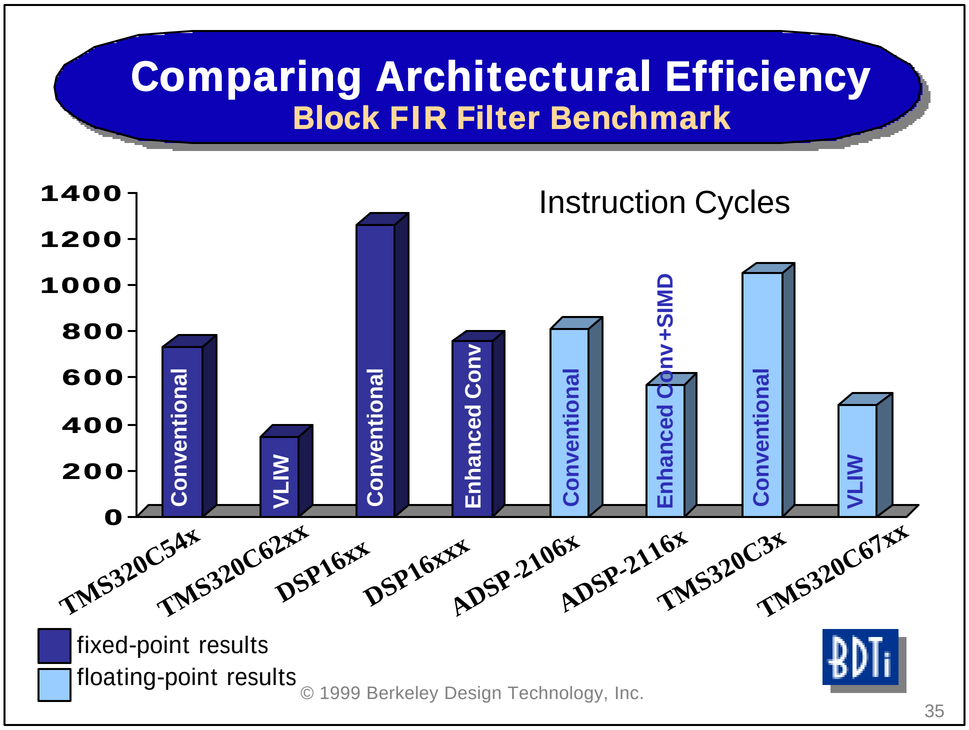
33



Superscalar Architectures

- Disadvantages:
  - Energy consumption is a major challenge
  - Dynamic behavior complicates software development
    - Execution-time variability can be a hazard–how to guarantee meeting real-time constraints?
      - Coding for worst-case behavior may leave a lot of a processor's performance untapped
    - Code optimization is challenging
      - If you can't tell how long a segment takes to execute, how can you tell if an optimization improved the performance?

34

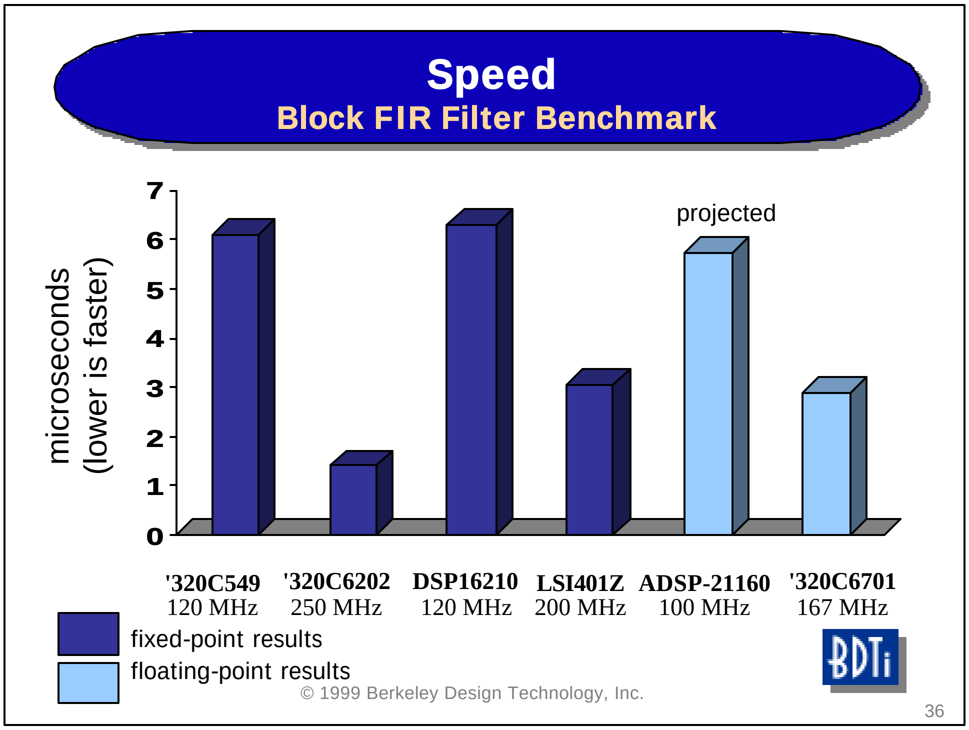


Comparing Architectural Efficiency

Block FIR Filter Benchmark

- ▨ fixed-point results
- □ floating-point results

35

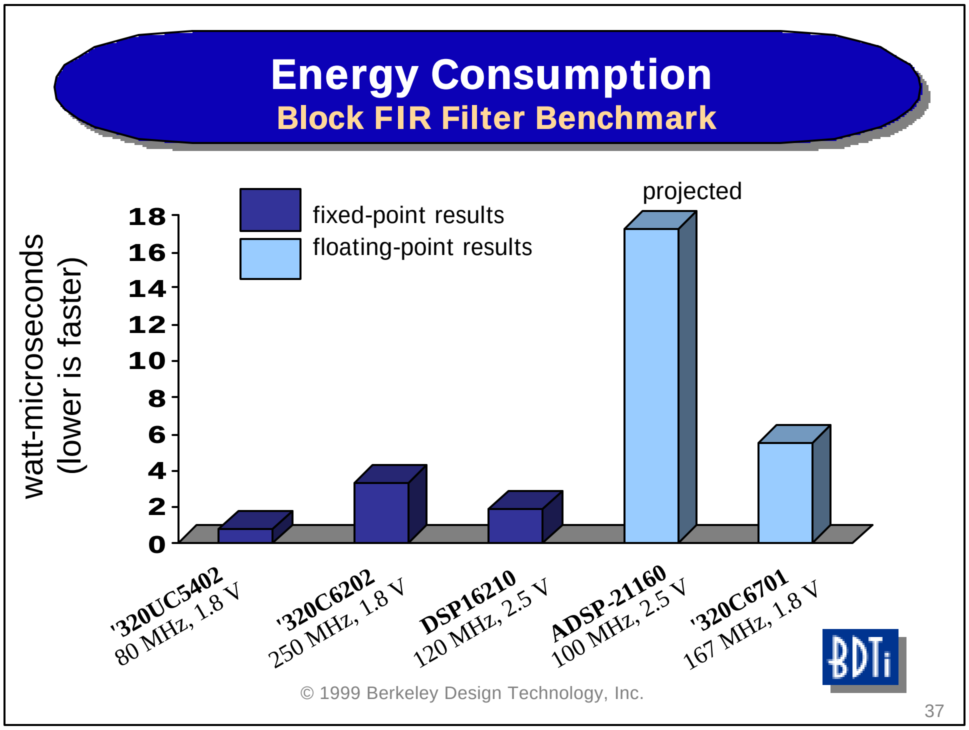


Speed

Block FIR Filter Benchmark

- ▨ fixed-point results
- □ floating-point results

36

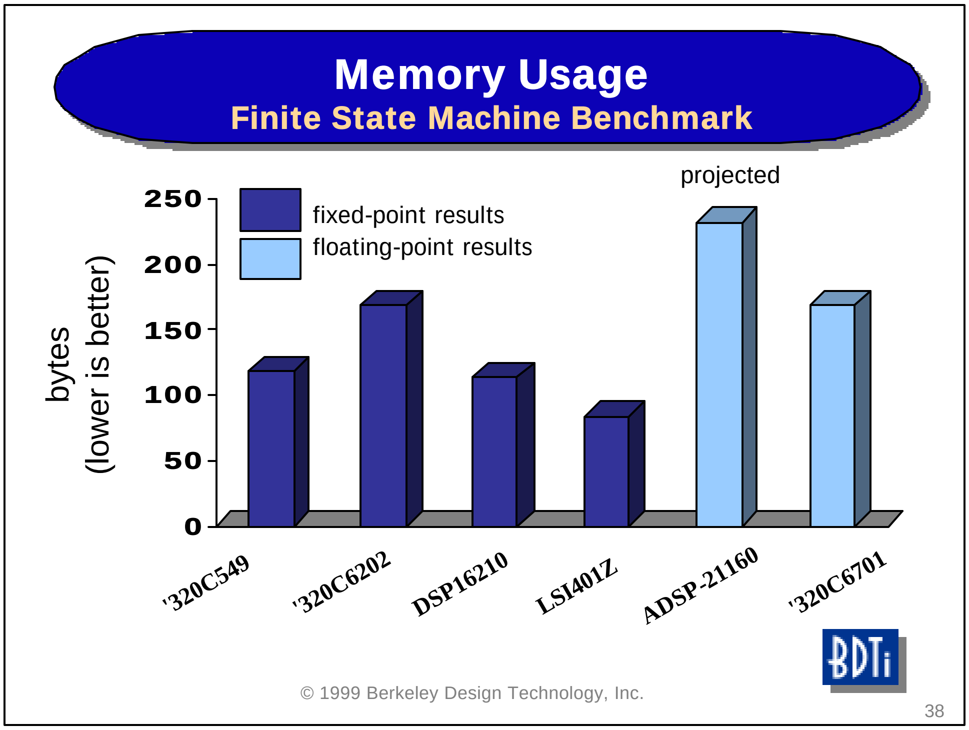


Energy Consumption

Block FIR Filter Benchmark

© 1999 Berkeley Design Technology, Inc.

37

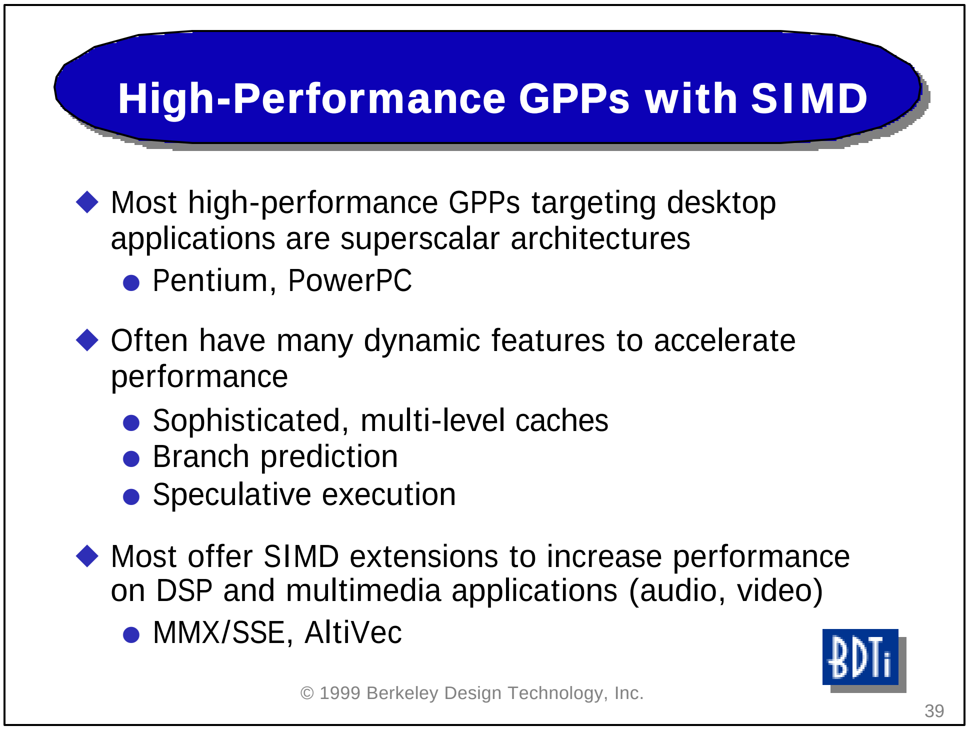


Memory Usage

Finite State Machine Benchmark

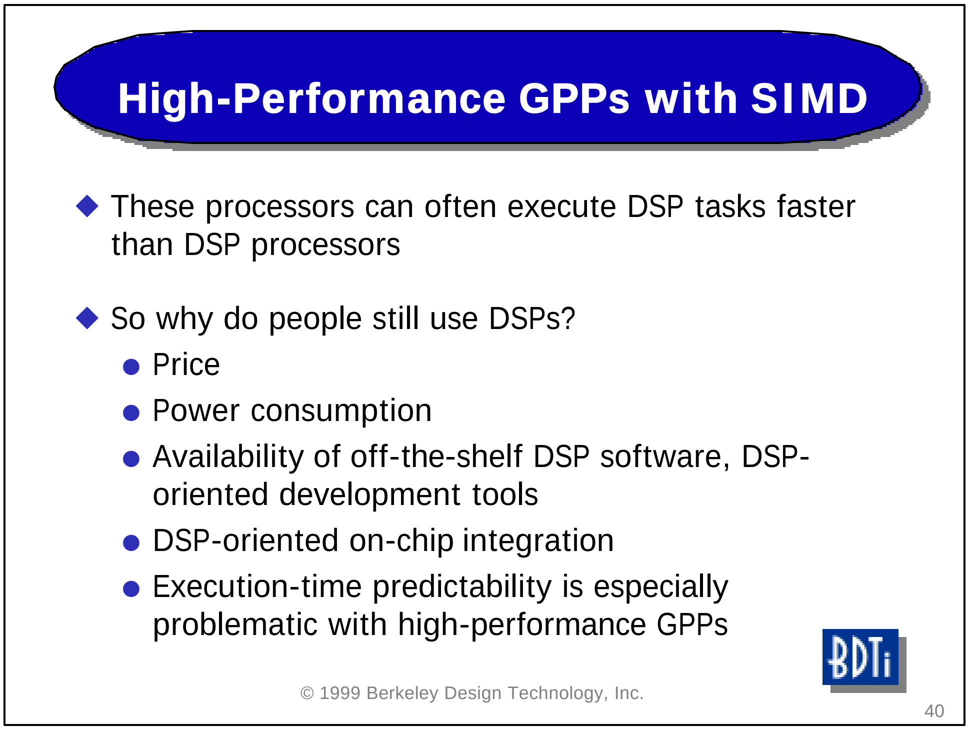
© 1999 Berkeley Design Technology, Inc.

38



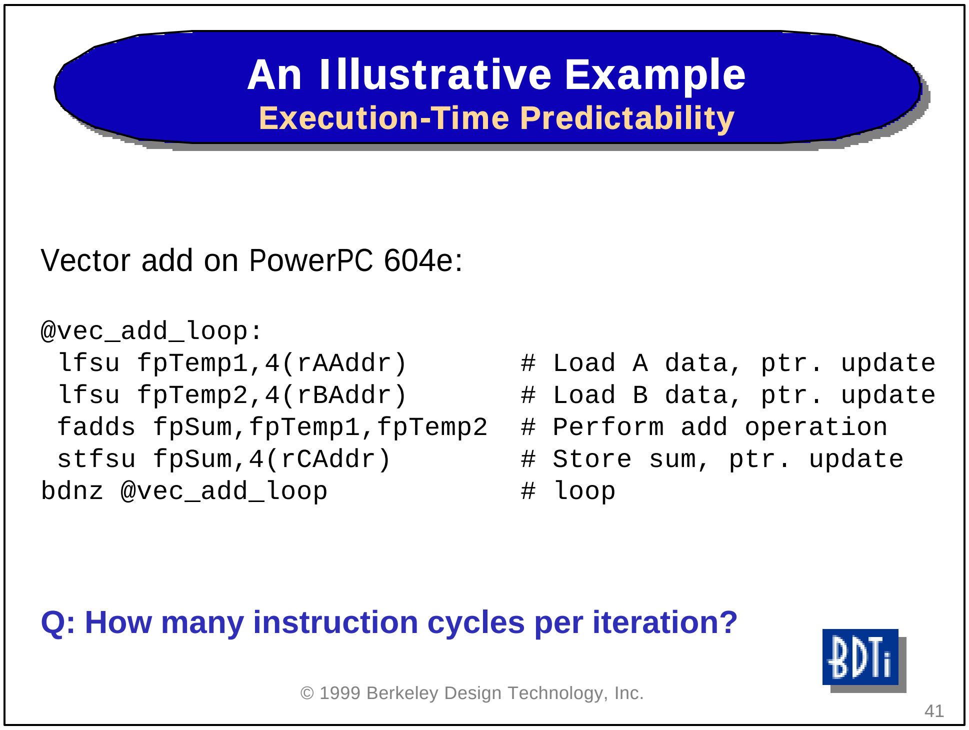
High-Performance GPPs with SIMD

- Most high-performance GPPs targeting desktop applications are superscalar architectures
  - Pentium, PowerPC
- Often have many dynamic features to accelerate performance
  - Sophisticated, multi-level caches
  - Branch prediction
  - Speculative execution
- Most offer SIMD extensions to increase performance on DSP and multimedia applications (audio, video)
  - MMX/SSE, AltiVec



High-Performance GPPs with SIMD

- These processors can often execute DSP tasks faster than DSP processors
- So why do people still use DSPs?
  - Price
  - Power consumption
  - Availability of off-the-shelf DSP software, DSP-oriented development tools
  - DSP-oriented on-chip integration
  - Execution-time predictability is especially problematic with high-performance GPPs

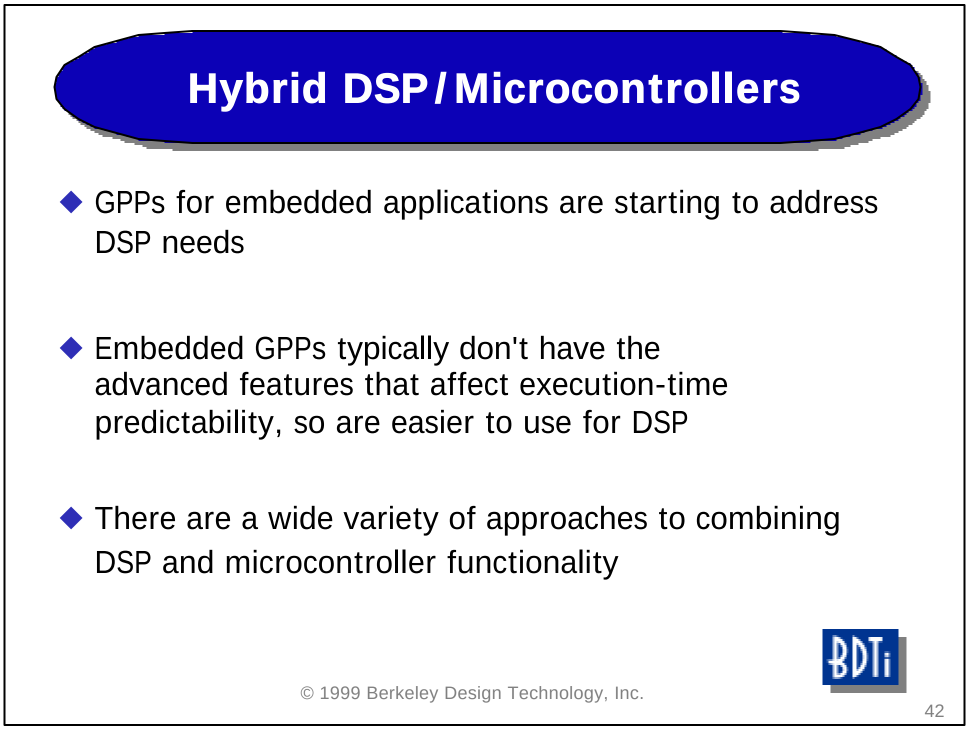


An Illustrative Example
Execution-Time Predictability

Vector add on PowerPC 604e:

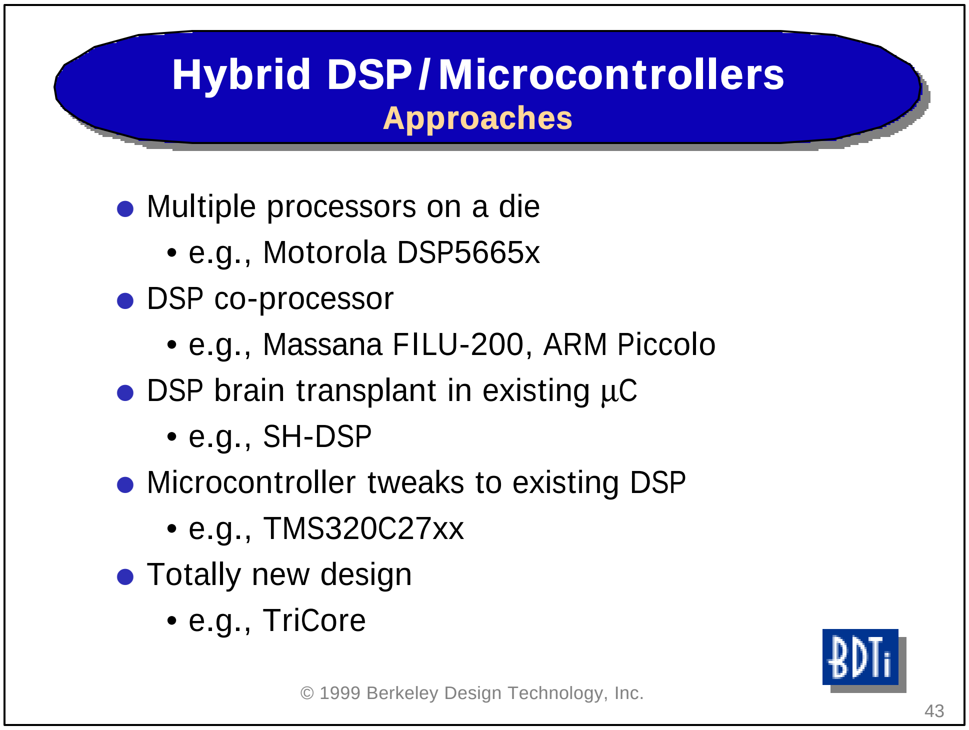
```
@vec_add_loop:
 lfsu fpTemp1,4(rAAddr)       # Load A data, ptr. update
 lfsu fpTemp2,4(rBAddr)       # Load B data, ptr. update
 fadds fpSum,fpTemp1,fpTemp2  # Perform add operation
 stfsu fpSum,4(rCAddr)        # Store sum, ptr. update
bdnz @vec_add_loop            # loop
```

Q: How many instruction cycles per iteration?



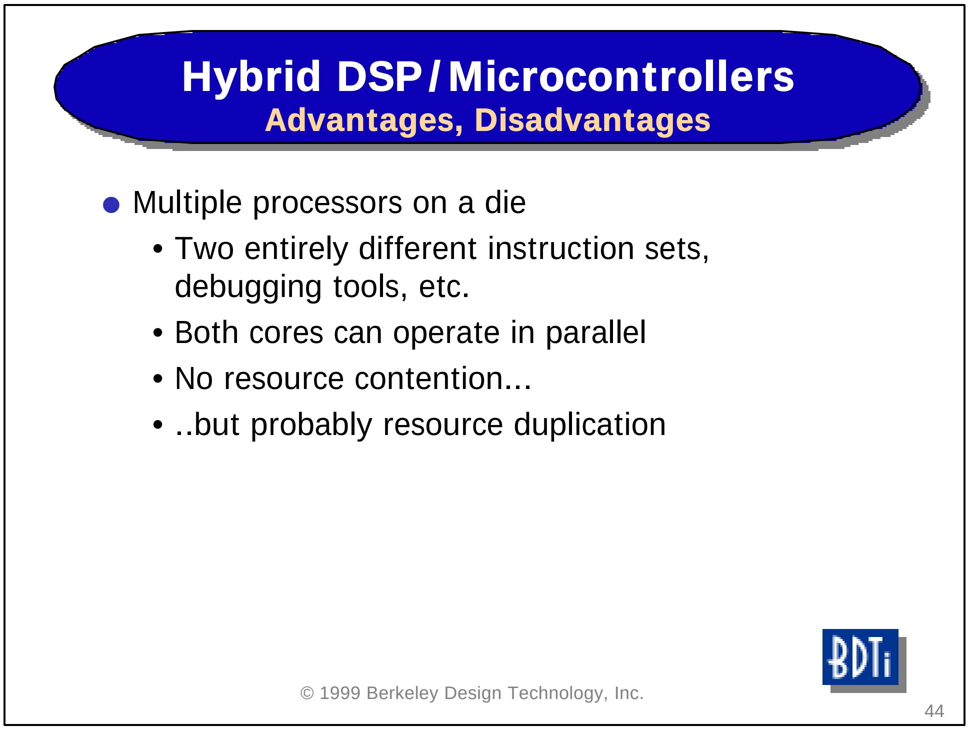
Hybrid DSP / Microcontrollers

- GPPs for embedded applications are starting to address DSP needs
- Embedded GPPs typically don't have the advanced features that affect execution-time predictability, so are easier to use for DSP
- There are a wide variety of approaches to combining DSP and microcontroller functionality



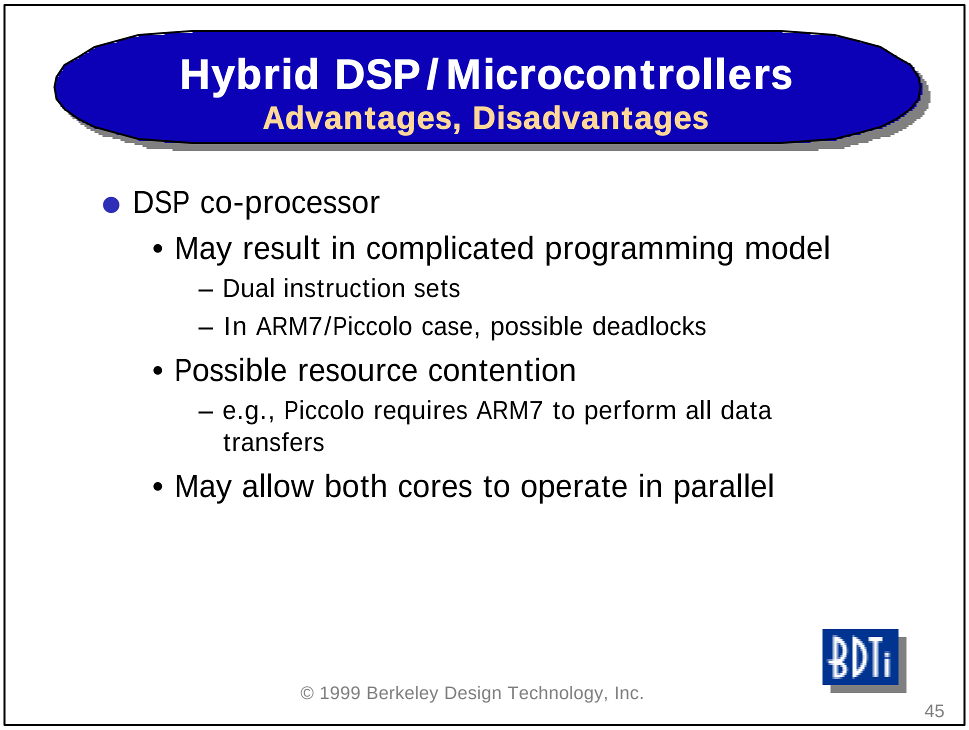
Hybrid DSP / Microcontrollers
Approaches

- Multiple processors on a die
  - e.g., Motorola DSP5665x
- DSP co-processor
  - e.g., Massana FILU-200, ARM Piccolo
- DSP brain transplant in existing µC
  - e.g., SH-DSP
- Microcontroller tweaks to existing DSP
  - e.g., TMS320C27xx
- Totally new design
  - e.g., TriCore



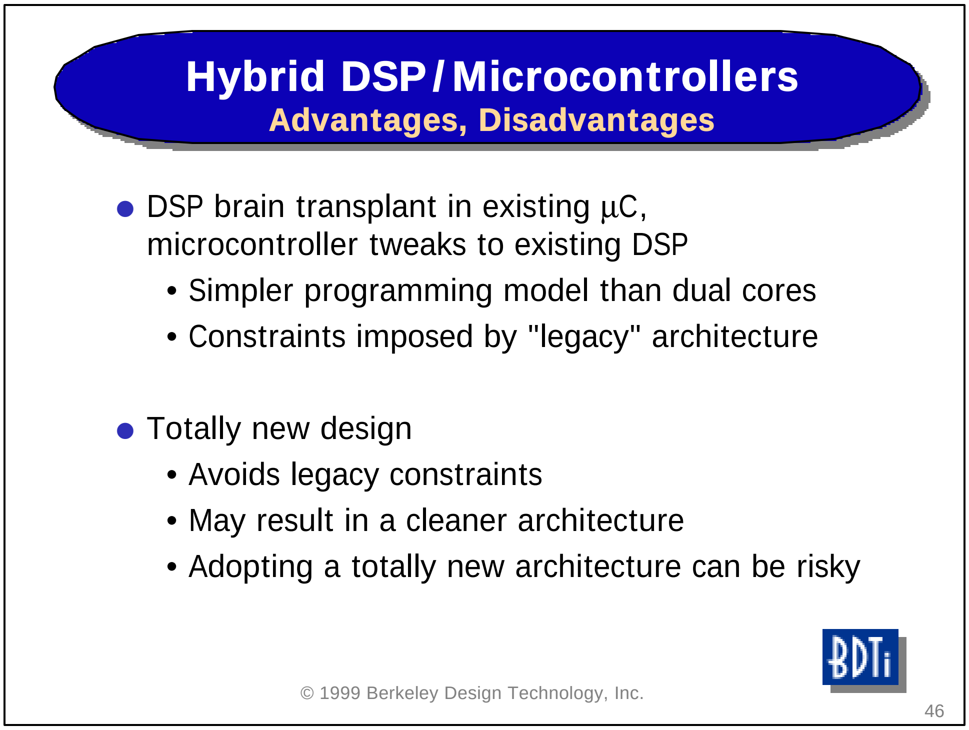
Hybrid DSP / Microcontrollers
Advantages, Disadvantages

- Multiple processors on a die
  - Two entirely different instruction sets, debugging tools, etc.
  - Both cores can operate in parallel
  - No resource contention...
  - ..but probably resource duplication



Hybrid DSP / Microcontrollers
Advantages, Disadvantages

- DSP co-processor
  - May result in complicated programming model
    - Dual instruction sets
    - In ARM7/Piccolo case, possible deadlocks
  - Possible resource contention
    - e.g., Piccolo requires ARM7 to perform all data transfers
  - May allow both cores to operate in parallel



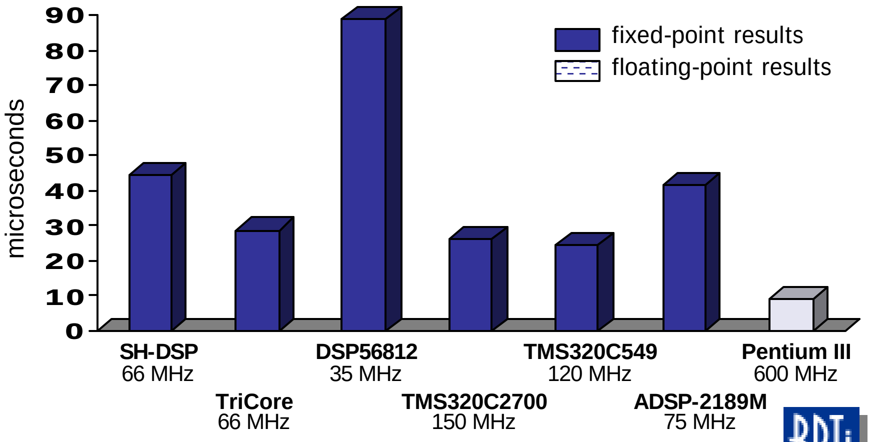
Hybrid DSP / Microcontrollers
Advantages, Disadvantages

- DSP brain transplant in existing µC, microcontroller tweaks to existing DSP
  - Simpler programming model than dual cores
  - Constraints imposed by "legacy" architecture
- Totally new design
  - Avoids legacy constraints
  - May result in a cleaner architecture
  - Adopting a totally new architecture can be risky

Execution Times

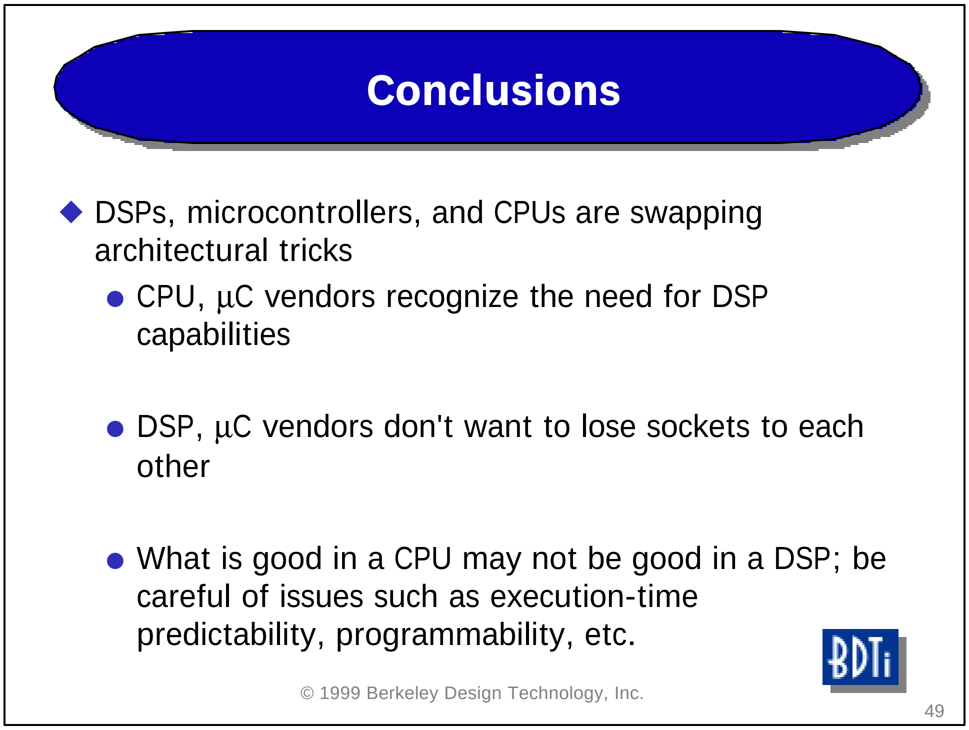
Complex Block FIR Filter Benchmark

(lower is better)



Conclusions

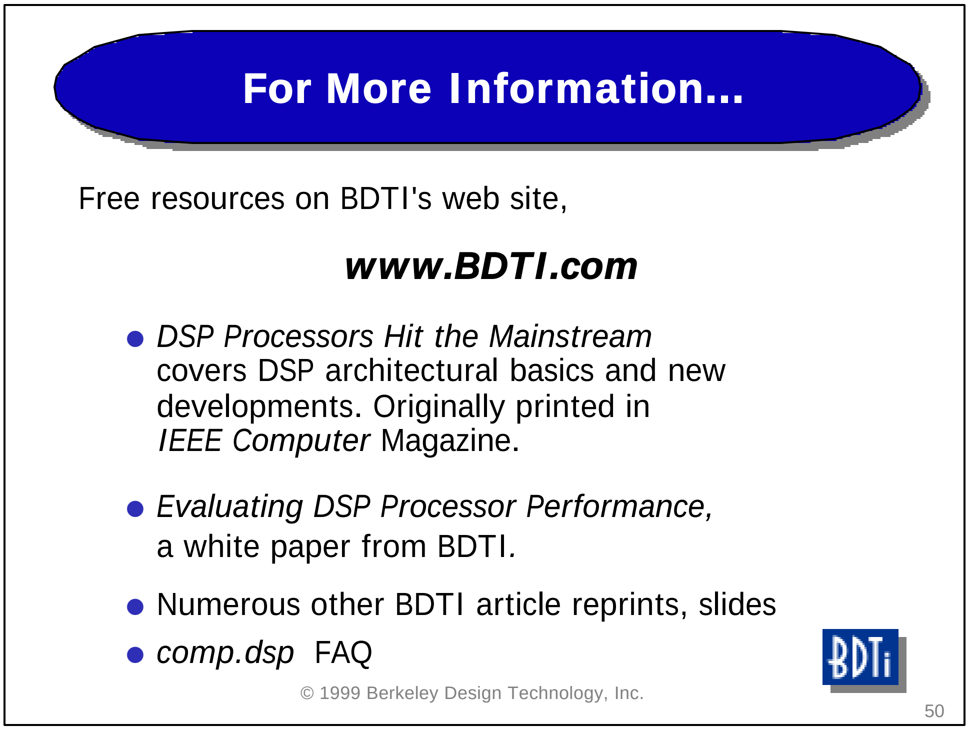
- The variety, performance range of processors for DSP is exploding
  - Better selection, flexibility,...
  - ...but harder to choose the "best" processor
  - Architectures are interesting, but other factors may be more important



Conclusions

- DSPs, microcontrollers, and CPUs are swapping architectural tricks
  - CPU, µC vendors recognize the need for DSP capabilities
  - DSP, µC vendors don't want to lose sockets to each other
  - What is good in a CPU may not be good in a DSP; be careful of issues such as execution-time predictability, programmability, etc.

49



For More Information...

Free resources on BDTI's web site,

www.BDTI.com

- *DSP Processors Hit the Mainstream* covers DSP architectural basics and new developments. Originally printed in *IEEE Computer* Magazine.
- *Evaluating DSP Processor Performance,* a white paper from BDTI*.*
- Numerous other BDTI article reprints, slides
- *comp.dsp* FAQ

50