

---

# **Effectiveness of the MAX-2 Multimedia Extensions for PA-RISC 2.0 Processors**

**Ruby Lee**

**Hewlett-Packard Company**

**HotChips IX**

**Stanford, CA, August 24-26, 1997**



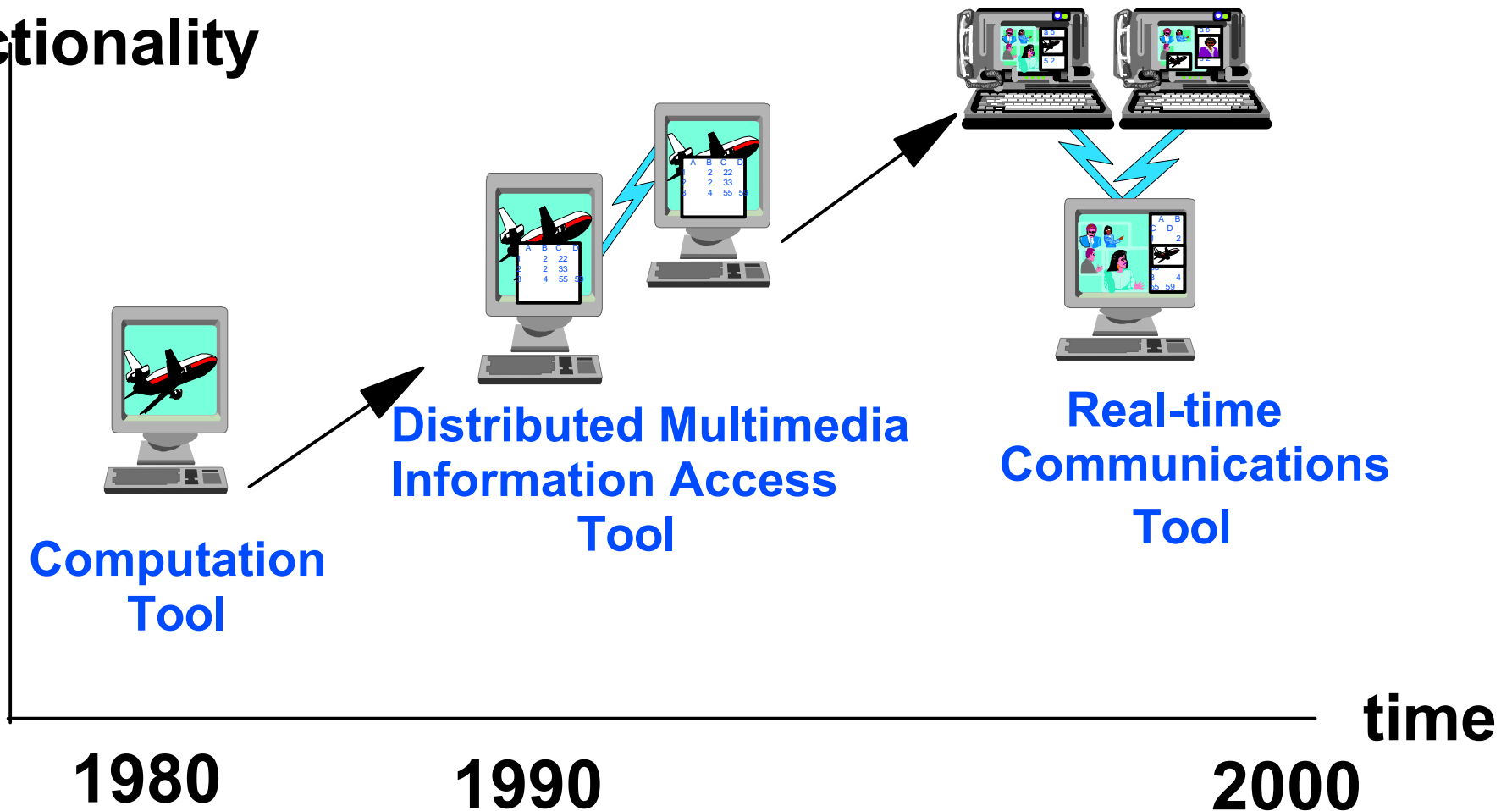
# Outline

---

- Introduction
- PA-RISC MAX-2 features and examples
  - Mix
  - Permute
  - Multiply with Shift&Add
  - Conditionals with Saturation Arith (e.g., Absolute Values)
- Performance Comparison
  - with / without MAX-2

# General-Purpose Workloads will include Increasing Amounts of Media Processing

**MM  
functionality**

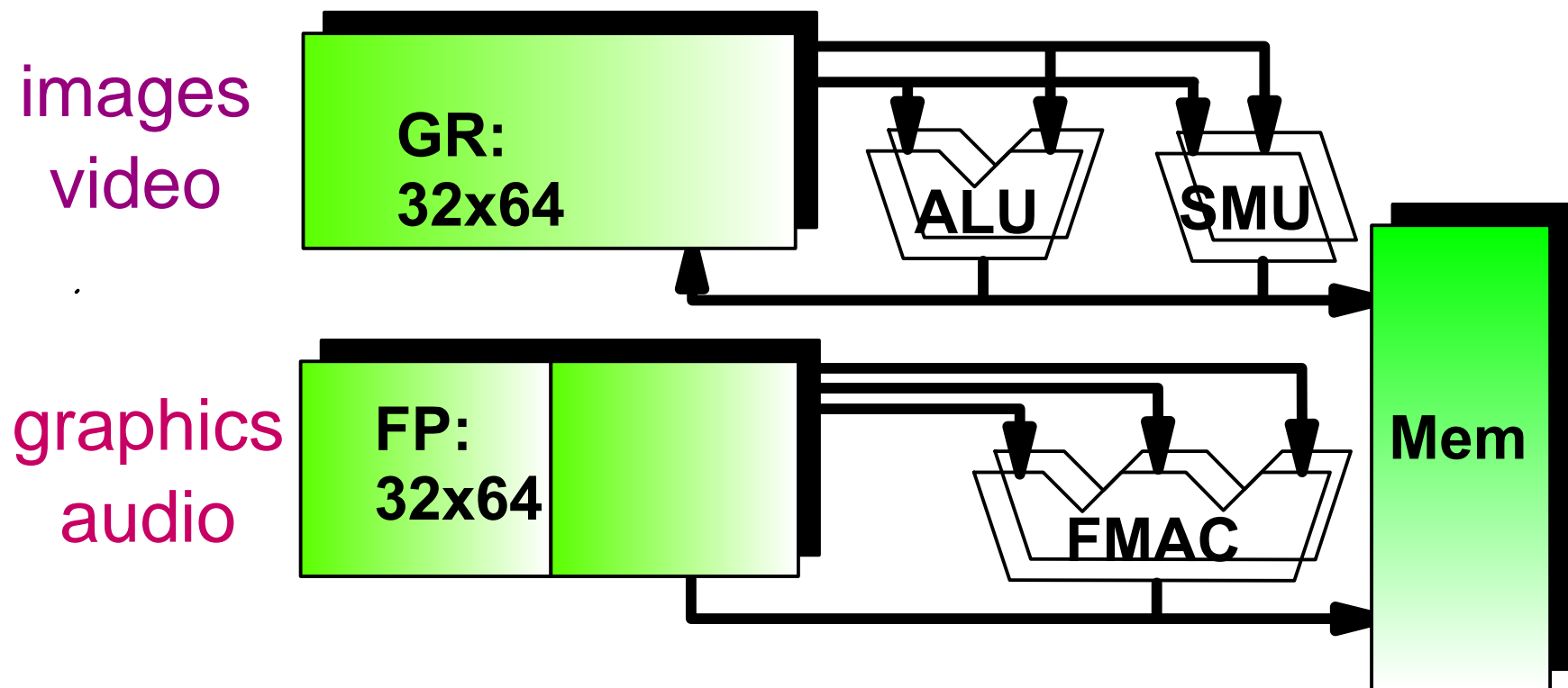


# Multimedia Extensions for General-Purpose Processors

---

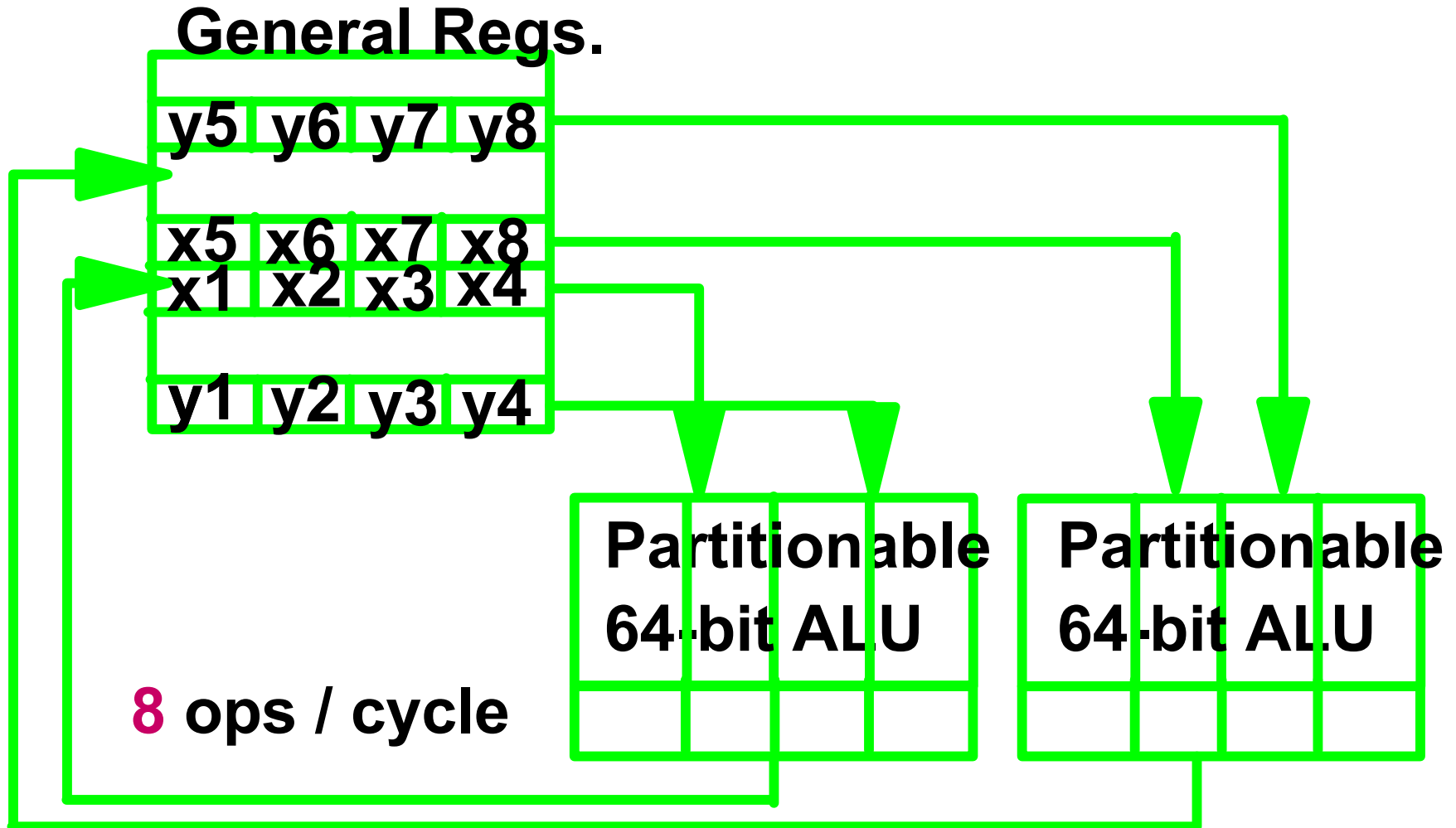
- MAX-1 for HP PA-RISC (product Jan '94)
- VIS for Sun Sparc (H2 '95)
- MAX-2 for HP PA-RISC (product Mar '96)
- MMX for Intel x86 (chips Jan '97)
- MDMX for SGI MIPS-V (tbd)
- MVI for DEC Alpha (tbd)

# Ideally, different media streams map onto both the integer and floating-point datapaths of microprocessors



PA-RISC 2.0 Processor Datapath

# Subword Parallelism in a General-Purpose Processor with Multimedia Extensions

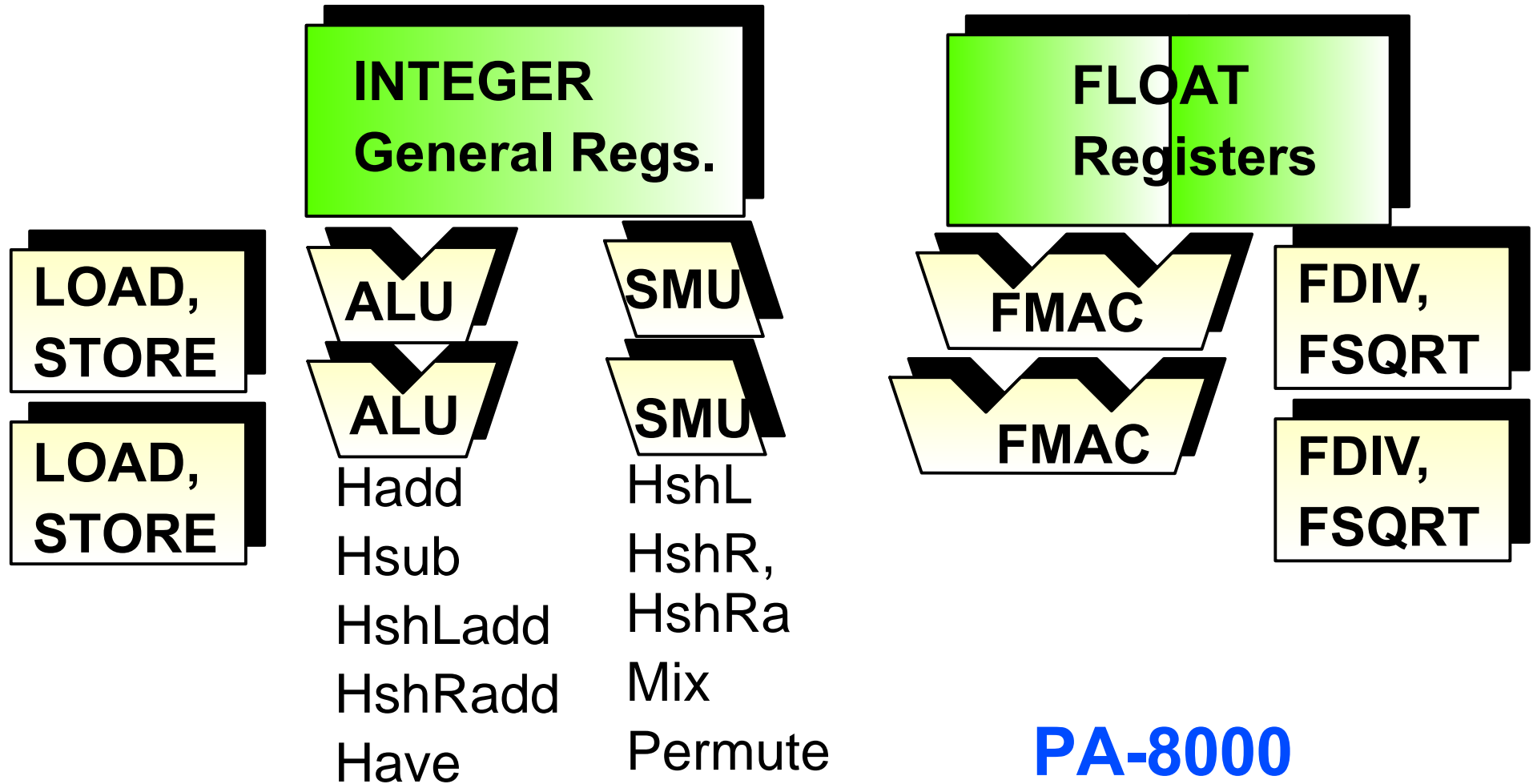


# Subword Parallel MAX-2 Instructions in PA-RISC 2.0

---

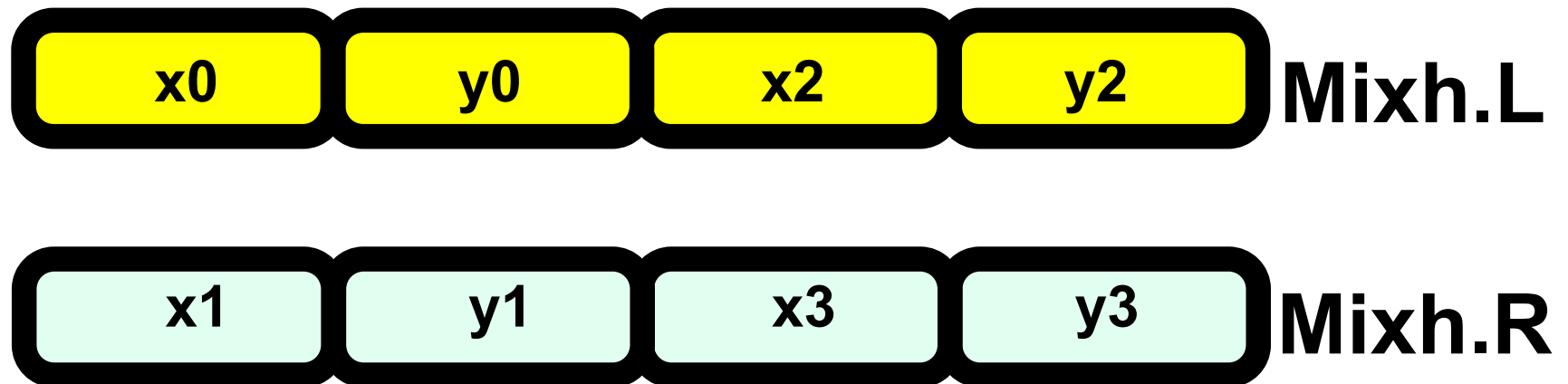
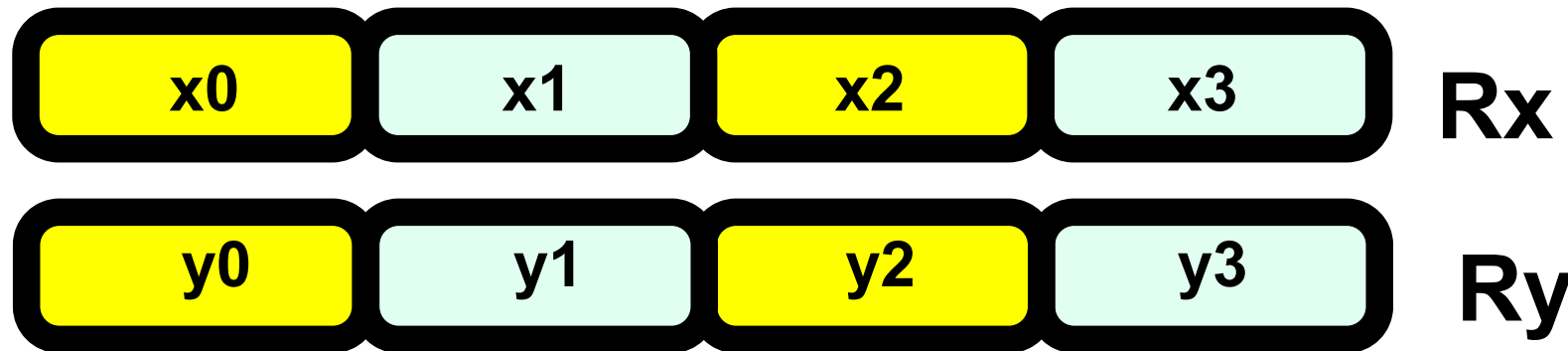
- Parallel Add (modulo or saturation)
- Parallel Subtract (modulo or saturation)
- Parallel Shift Right (1,2 or 3 bits) and Add
- Parallel Shift Left (1,2 or 3 bits) and Add
- Parallel Average
- Parallel Shift Right (n bits)
- Parallel Shift Left (n bits)
- Mix
- Permute

# MAX-2 Leverages Existing Processing Resources



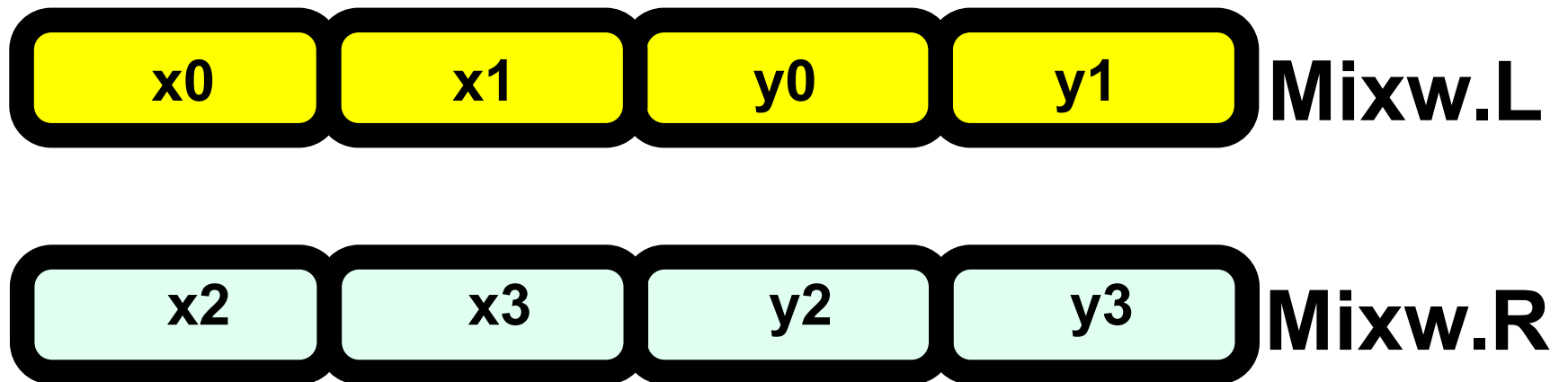
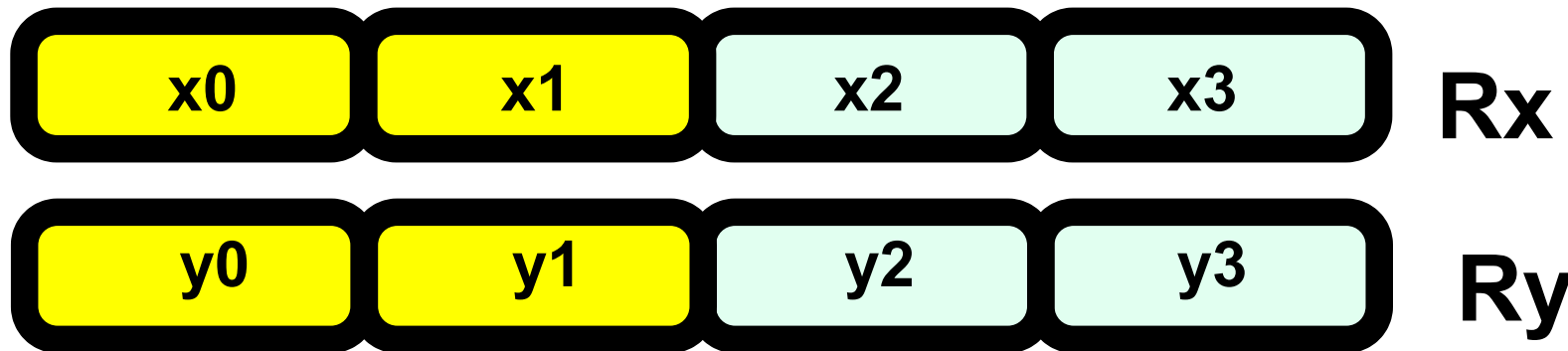
# MIX interleaves even (or odd) Subwords from 2 source registers

---



# Mix of larger (32-bit) subwords

---



# 4x4 Matrix Transpose in 2 steps with 8 MIX instructions



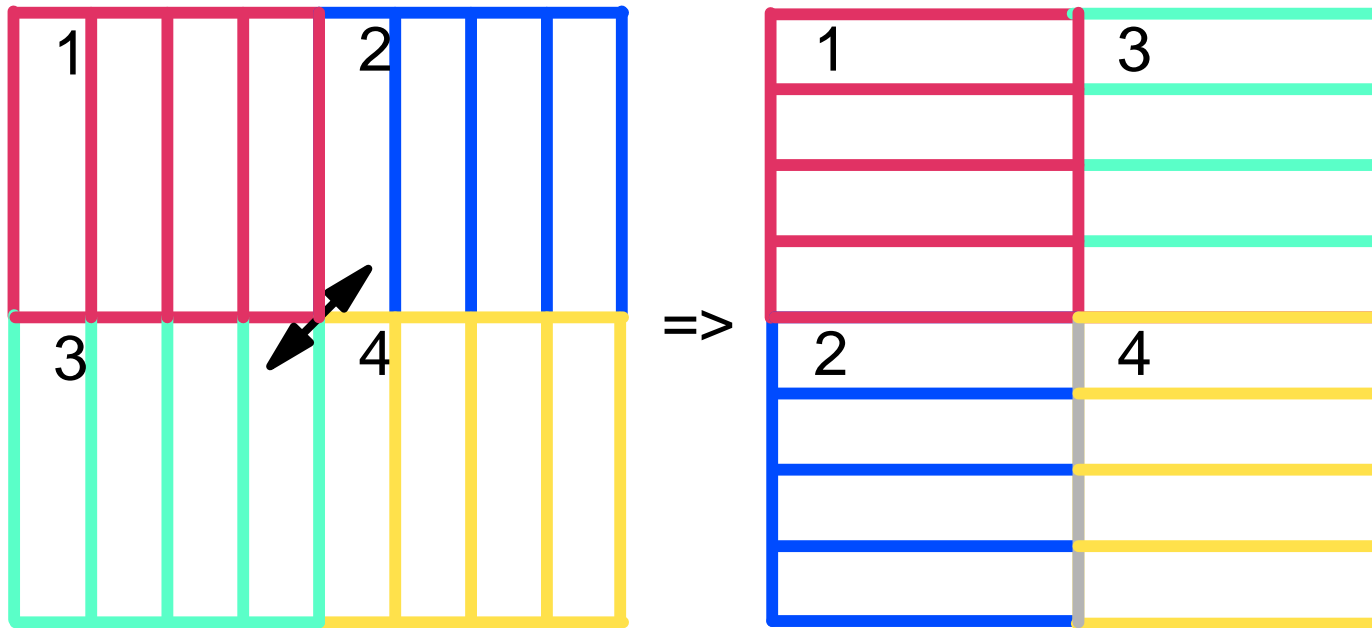
mixh,l	r1,r2,t1
mixh,r	r1,r2,t2
mixh,l	r3,r4,r1
mixh,r	r3,r4,r2

mixw,r	t1,r1,r3
mixw,l	t1,r1,r1
mixw,r	t2,r2,r4
mixw,t	t2,r2,r2

In general, takes  $(\log n)$  steps, and  $(n \log n)$  instructions

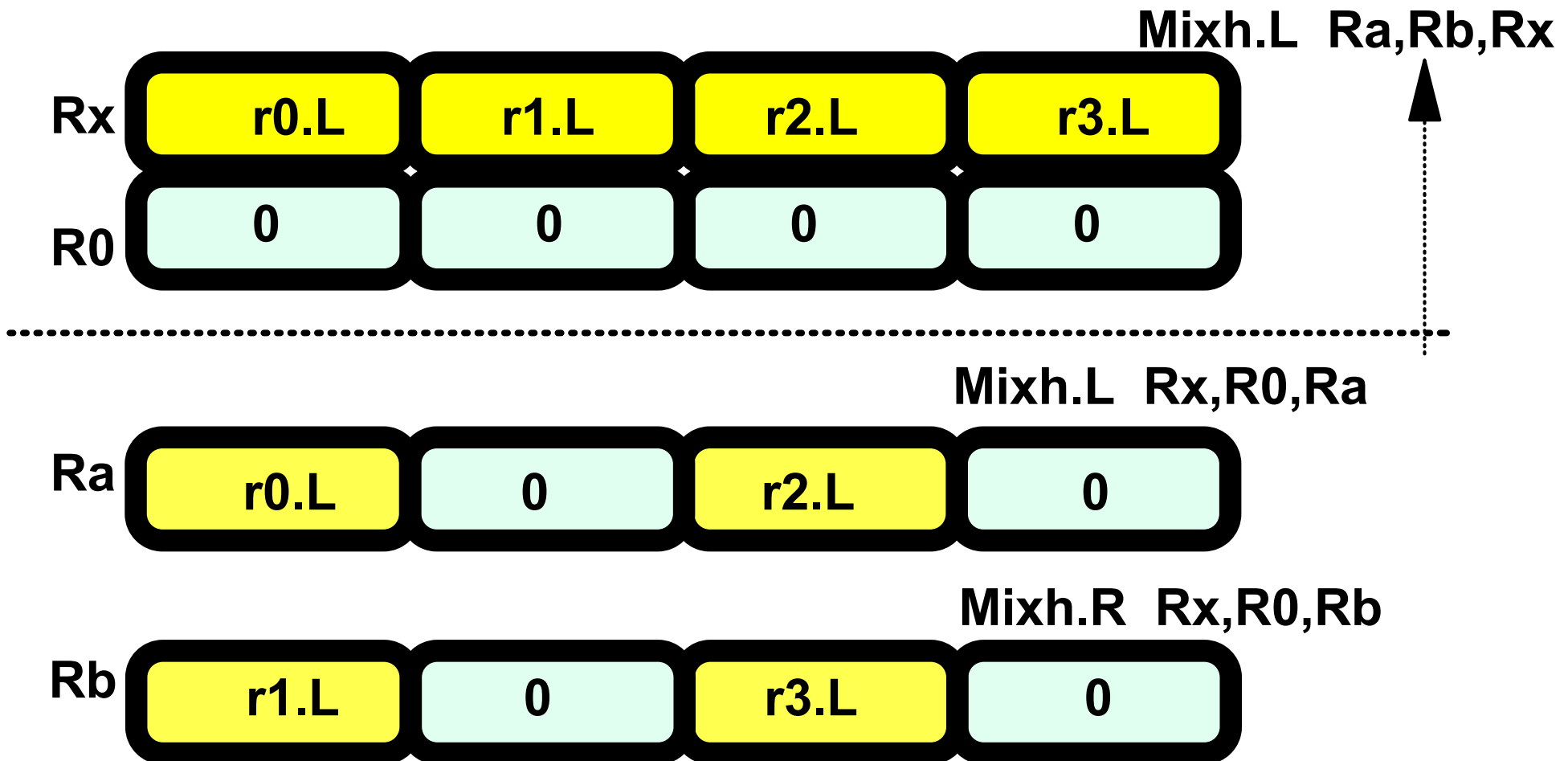
# 8x8 Matrix Transpose is just four 4x4 Matrix Transposes

---



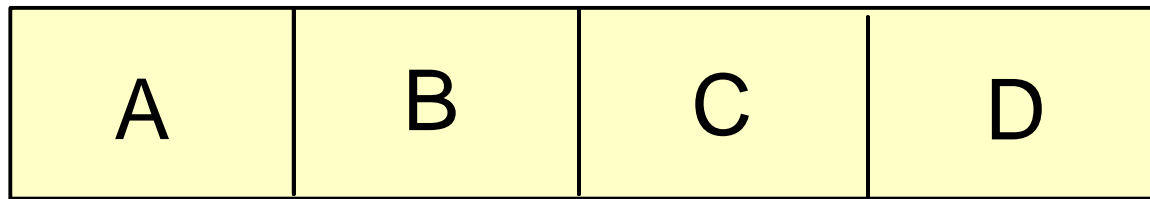
No extra cycles needed  
to swap sub-matrices 2 and 3

# Expanding (and then contracting) Subwords using MIX

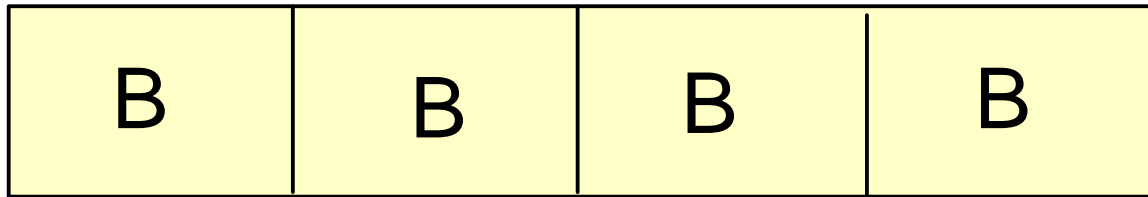


# Permute allows any rearrangement of subwords from 1 source register

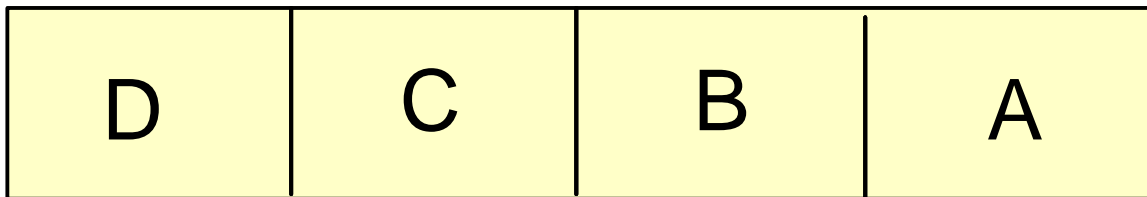
---



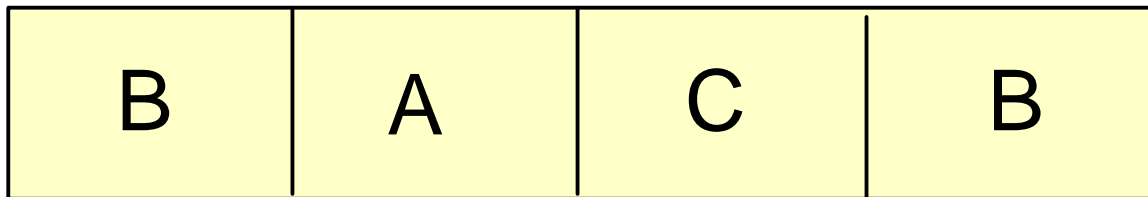
Input Register



Replicate scalar  
vector-scalar ops.



Reverse  
bi-endian data



Arbitrary permutation

# Example: 8x8 IDCT Algorithm

---

- for  $i = 1$  step 4 to 8 do
  - 1-dimensional IDCT on columns
- 8x8 matrix transpose
- 
- for  $i = 1$  step 4 to 8 do
  - 1-dimensional IDCT on rows
- 8x8 matrix transpose

# Parallel Multiplication with Parallel Shift Right and Add Instructions

---

Express Constant as binary fraction:

$$1.4142 = 1.01101010$$



$$t = 1.01 * x$$

$$t = x + x \gg 2$$

hshradd x,2,x,t

$$s = 1.0101 * x$$

$$s = x + t \gg 2$$

hshradd t,2,x,s

$$t = 1.0110101 * x$$

$$t = t + s \gg 3$$

hshradd s,3,t,t

# High Performance of Parallel Shift&Add for Multiplications

---

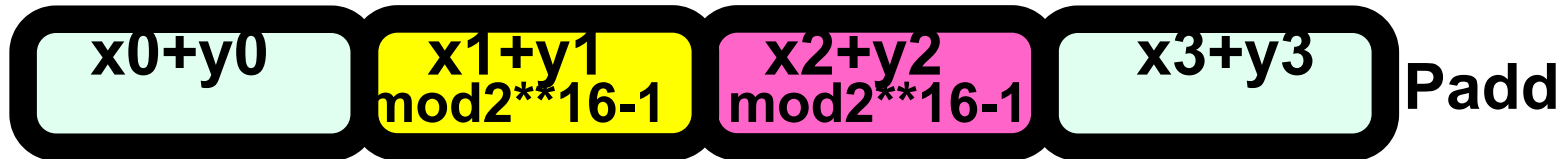
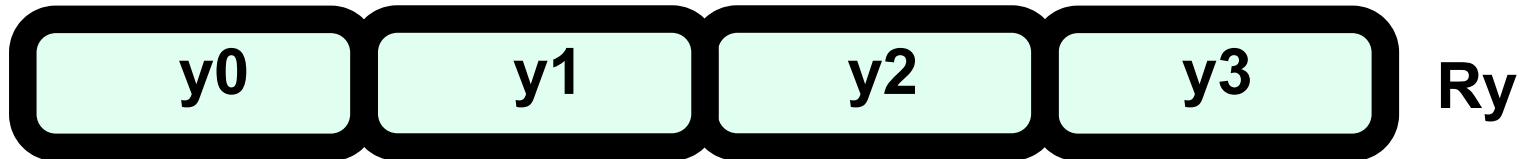
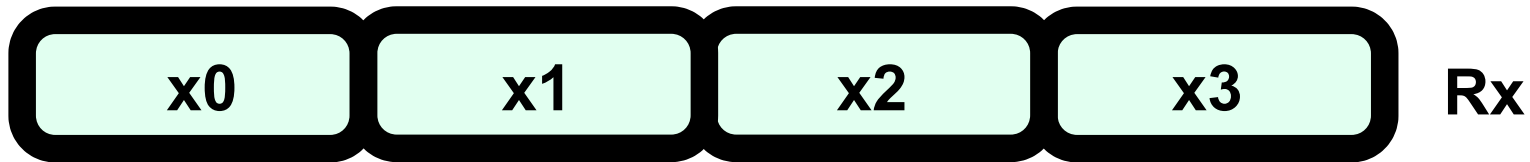
## 16-bit Multiplies per Cycle

	2	3	4
	ShAdd's	ShAdd's	ShAdd's
1 adder	2	1.33	1
2 adders	4	2.67	2

# Parallel Subword Add

with modulo arithmetic, signed saturation and unsigned saturation

---



Negative Overflow

Positive Overflow

# Sum of Absolute Differences using Saturation Arithmetic

---

An example:

r1 

85	20	100	5
----	----	-----	---

 r2 

45	211	0	33
----	-----	---	----

hsub,us r1, r2, r3

r3 

40	0	100	0
----	---	-----	---

hsub,us r2, r1, r4

r4 

0	191	0	28
---	-----	---	----

hadd r3, r4, r5

r5 

40	191	100	28
----	-----	-----	----

differences or zero

absolute values

By using 2 accumulator registers in actual SAD code,  
4 pixels are retired every 2 cycles

# Performance Comparison Goals

---

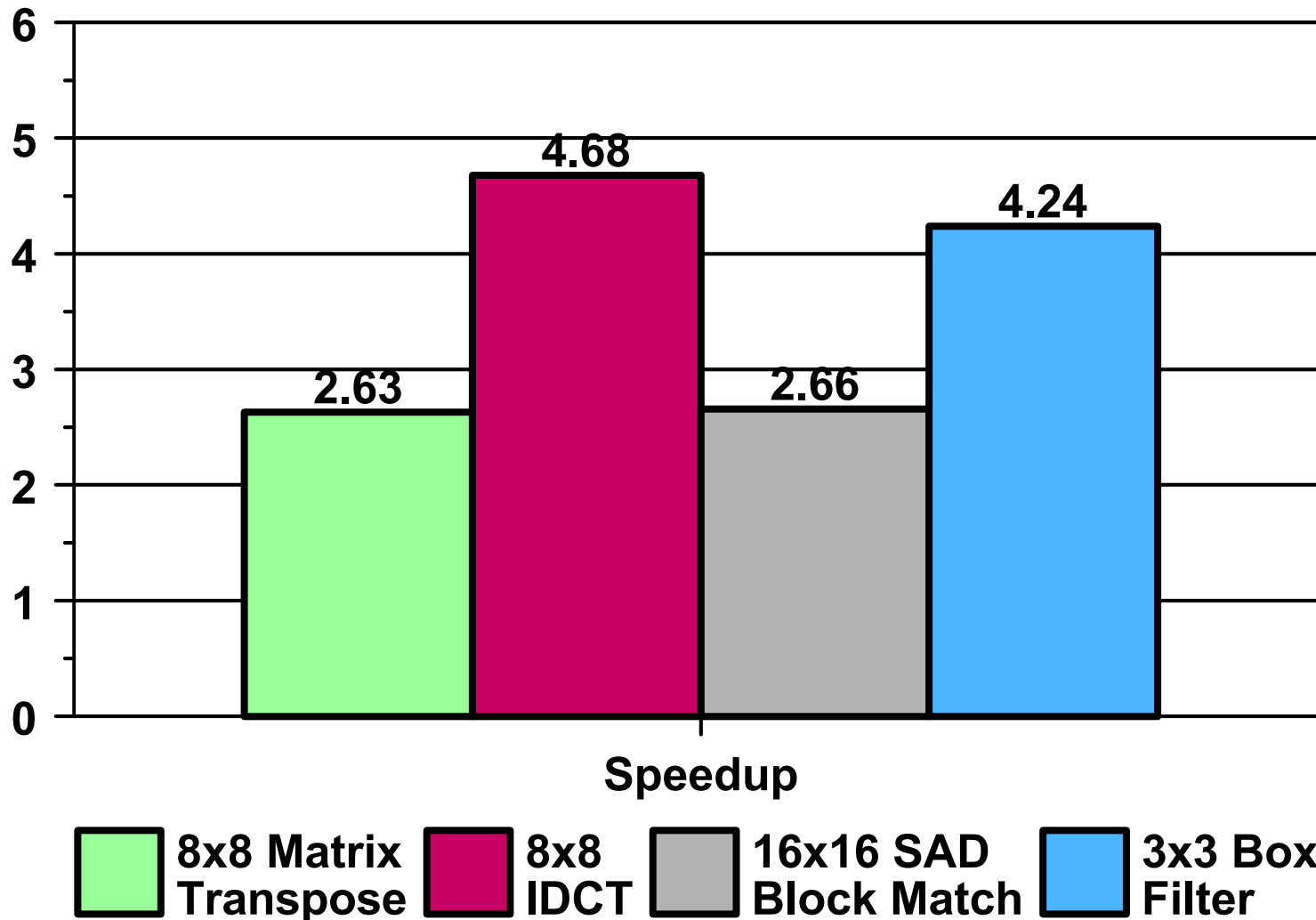
- Apples-to-apples comparison of PA-RISC code, with and without MAX-2
- Code aggressively optimized in both cases
  - even without MAX-2, PA-RISC has many features useful for media processing
- Select performance critical loops, e.g.,
  - Matrix Transpose
  - 2-D IDCT
  - Block Match (SAD)
  - Box Filter

# Other PA-RISC features useful for Media Processing

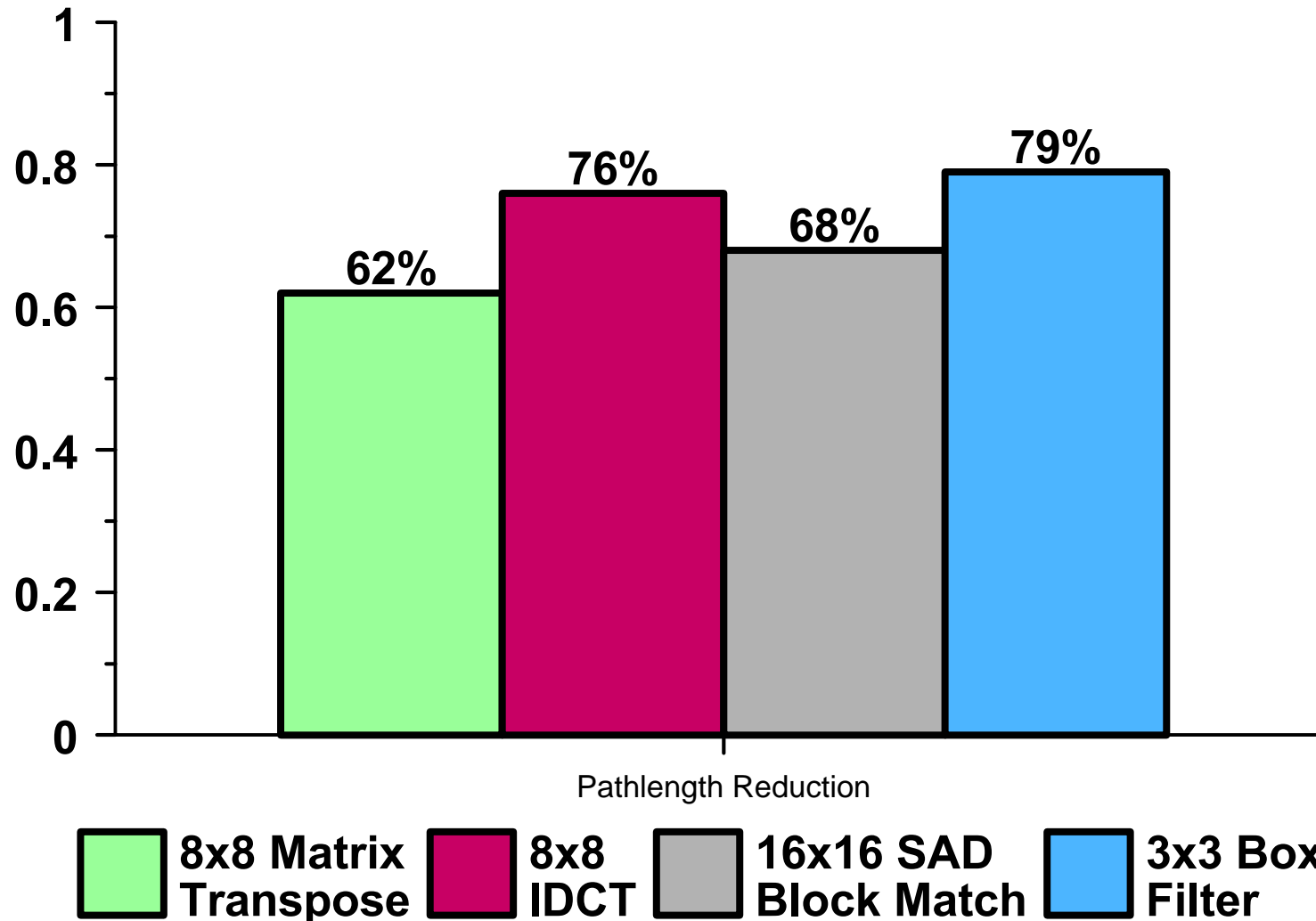
---

- Shift Right Pair
- Extract bit-field
- Deposit bit-field
- Logical instructions (And, Andcm, Or, Xor)
- Floating Point Multiply-Accumulate
- Multiple floating point condition codes
- Cache line prefetch (read and write)
- Spatial locality cache hint
- Arithmetic Nullification

# Execution Time Speedup (PA8000 with MAX-2 vs. no MAX2)



# % Pathlength Reduction (PA8000 with MAX-2 vs. no MAX2)



# Cycles Per Pixel (without and with MAX-2)

---

	8x8 Matrix Transpose	8x8 IDCT	16x16 SAD	3x3 Box Filter
without MAX-2	0.66	11.18	1.66	11.86
with MAX-2	0.25	2.29	0.63	2.80

# Register Usage (without and with MAX-2)

---

	8x8 Matrix Transpose	8x8 IDCT	16x16 SAD	3x3 Box Filter
without MAX-2	22	20	12	18
with MAX-2	18	21	14	15

# Conclusions

---

- Programmable Media Processing is here
- MAX-2 has high-perf, low-cost primitives
  - data re-arrangement: Mix and Permute
  - multiply: Parallel Shift&Add, Parallel Shift (and FMAC)
  - in-line conditionals: Parallel Add/Sub. with saturation arith
  - combined operations: Parallel Average
- Super-linear speedup achievable in loops
- Ubiquity of Subword Parallel instructions in microprocessors assured
  - HLL, compiler and/or pre-processor support are next steps
- Further ISA innovations likely