

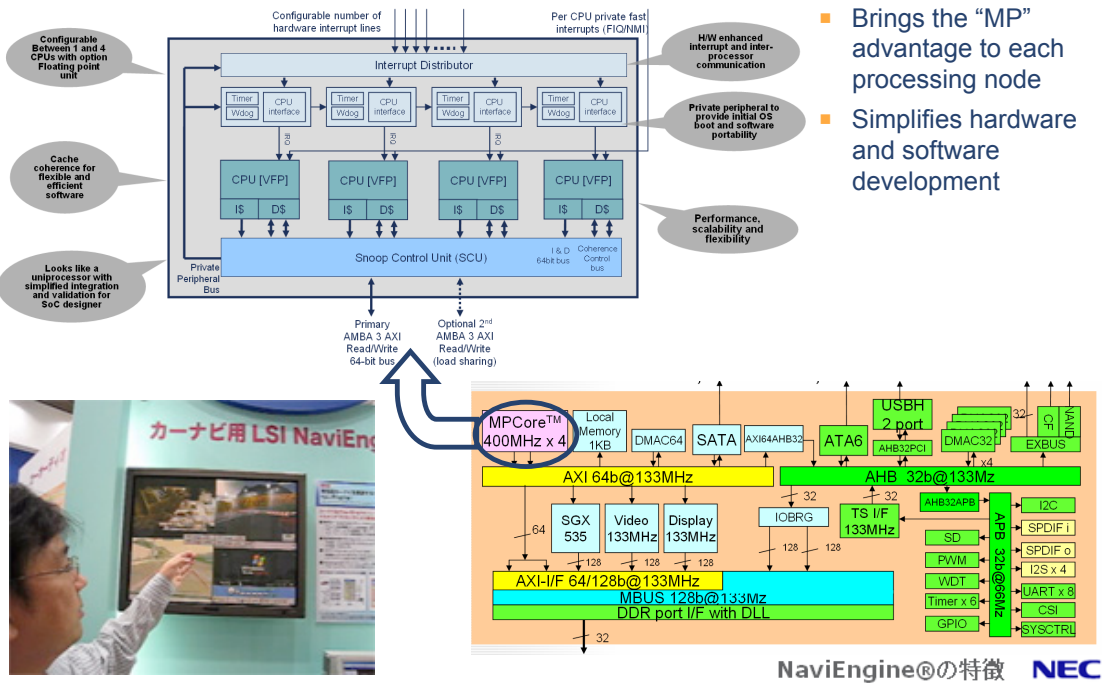
Myths of Multicore

Ian Rickards
CPU Product Manager
ARM Inc, San Diego

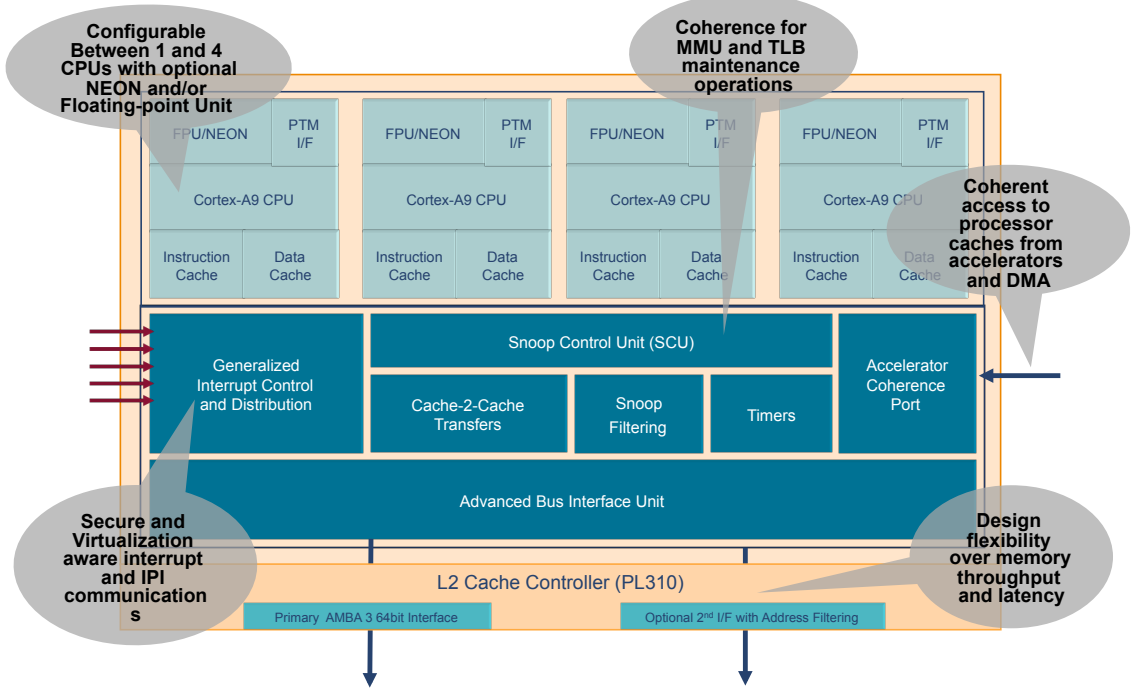
EDP Seminar Monterrey
April 16th 2008

Myth #1: Multicores are Tomorrow's Technology

First ARM MPCore ASSP available



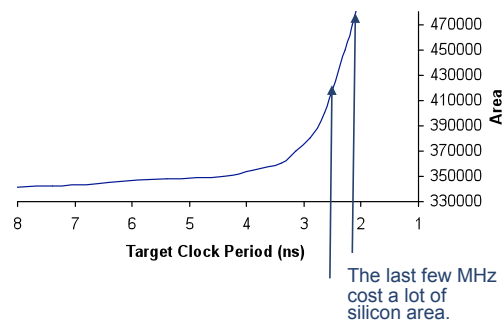
2nd Generation: Cortex-A9 MPCore



Myth #2: Aren't two cores twice the size?

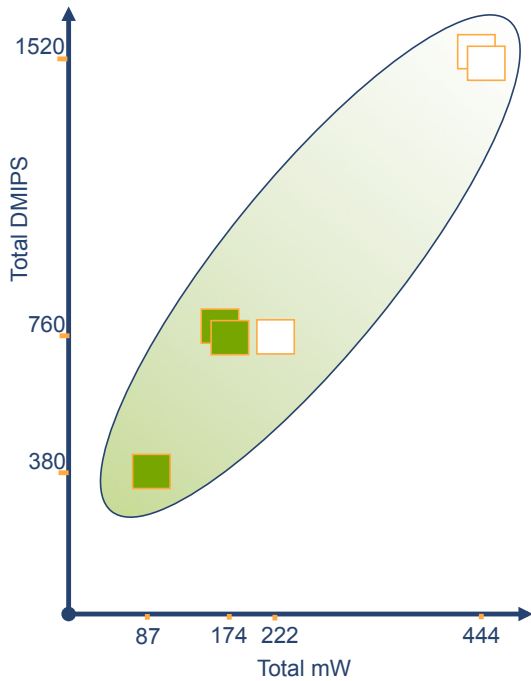
Multicore for Implementation Efficiencies

Performance/ Area Tradeoffs



- Increasing MHz increases power and area exponentially
 - Synthesis area increases significantly for the last few MHz
 - High MHz requires high-speed libraries/process and require more dynamic/leakage power
 - Higher voltages for higher MHz
 - Complicates SoC design and extends time to market

Multicore: Physical Implementation



Multiprocessing and library choice together provide huge implementation flexibility

	Area opt	Speed opt
CPUs	1	1
Std cells	Metro	Adv-HS
Freq/MHz	320	620
Area with cache/mm2	1.46	2.54

ARM11 MPCore				
CPU #	1	1	2	2
Priority	Power	Speed	Power	Speed
Library	Metro	Adv-HS	Metro	Adv-HS
DMIPS	380	760	760	1520
MHz	304	608	304	608
mW/MHz	0.23	0.32	0.46	0.64
Static mW	17	27	34	55
Total mW	87	222	174	444

TSMC 90G, Leakage at 85°C

THE ARCHITECTURE FOR THE DIGITAL WORLD®



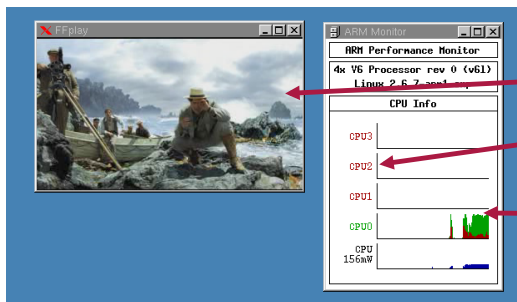
Myth #3: Multiple cores use more power

THE ARCHITECTURE FOR THE DIGITAL WORLD®

8

ARM®

Controlling Power Consumption

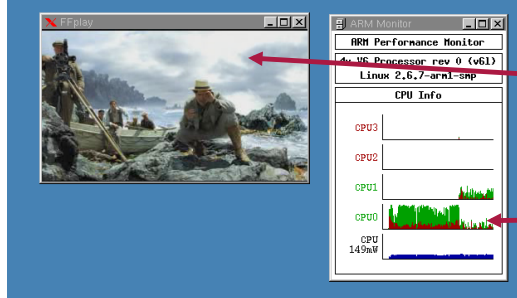


Single CPU

For a given workload requirement

Unused processors are 'turned off'

Single CPU @ 260MHz,
Testchip consuming ~160mW



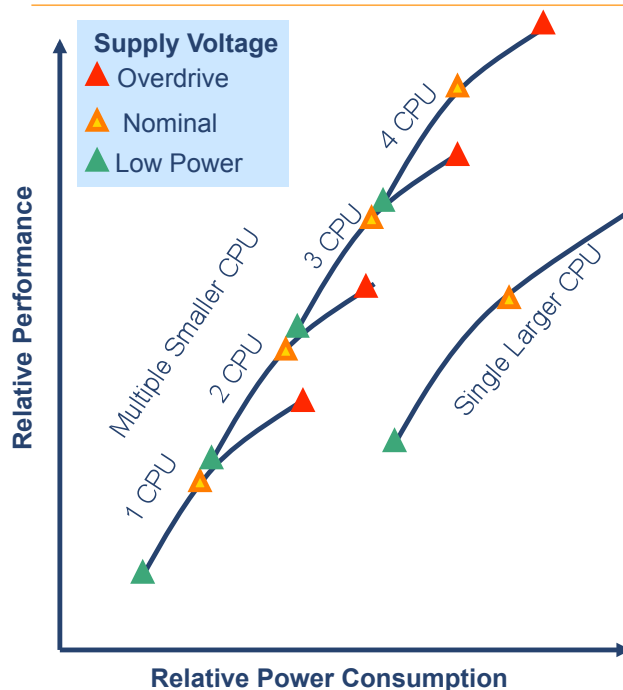
Dual CPU (same MHz, same voltage)

Same workload level leaves headroom on CPUs

Alternatively, up to 50% energy saving potential with voltage and frequency scaling

Multiprocessing offers more performance at lower MHz

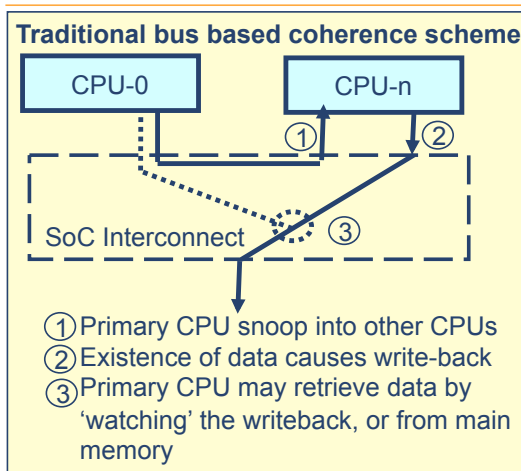
Performance and Power Scalability



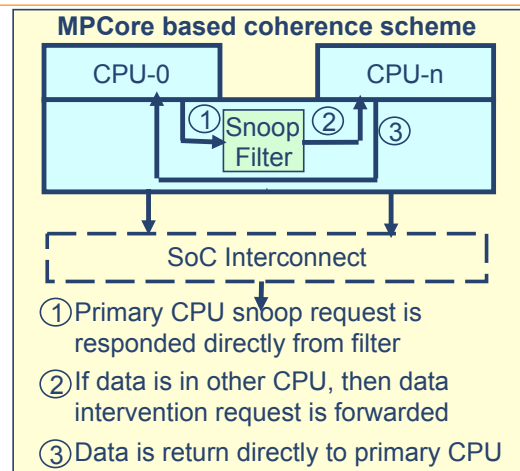
- Voltage scaling affects peak performance
- Multicore can provide scalability over performance and power
- Larger single core processors typically provide less scalability over both power and performance
- Larger processors typically are less power efficient at providing a given performance

Myth #4: Multicores have More Overhead

MPCore: Reduction in bus contention



- Sharing of data causes increased loading on system bus due to 'unnecessary' cache write-backs

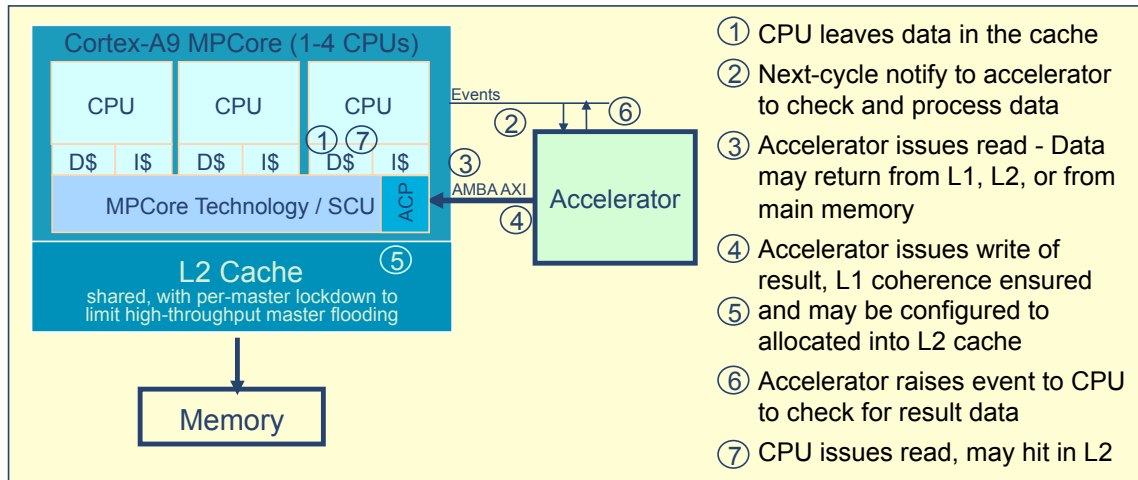


- ARM SCU provides additional bandwidth within processor core to shield system bus from any additional loads due to sharing of data

Enhanced Accelerator SoC Integration

ARM MPCore: Accelerator Coherence Port (ACP)

- Sharing benefits of the ARM MPCore optimized coherency design
- Accelerators gain access to CPU cache hierarchy, increasing system performance and reducing overall power
- Uses AMBA® 3 AXI™ technology for compatibility with standard un-cached peripherals and accelerators



ACP - Access to Shared Caches

- Example: CRC engine for TCP packet forwarding on 64 byte packet
 - Using typical system latency, ignoring common processing overhead
 - Assumed writes are fully buffered

Algorithm Stage	Approximate Cycle Counts	
	Traditional shared memory with mailbox communication	ACP attached accelerator with synchronous event
Packet received and processed by CPU	0	0
Flush cache to make data visible to accelerator	20	0
Accelerator notified of data availability	> 4 [write to mailbox GPIO]	1 [Send Event]
Accelerator Reads data	120 [read data from off-chip]	10 [read from L1/L2]
Accelerator Write data (assuming buffered)	8	8
Processor reads data	120	12 [from L2]
Total latency overhead	~272 cycles	~31 cycles

ACP solution is appropriate for cycle-offload accelerators executing in 100's of cycles with cache resident workloads. For example in low latency situations required by audio echo cancellation

Myth #5: Multicores are More Work

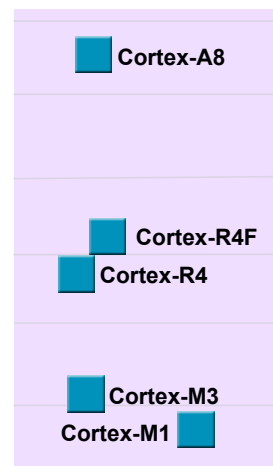
(for the software engineers)

ARM Cortex Family of Processors

Bringing the benefits of architectural innovation across the spectrum

Cortex[™]
Intelligent Processors by ARM®

- ARM Cortex-**A** Series:
 - Applications processors for complex OS and user applications
- ARM Cortex-**R** Series:
 - Embedded processors for real-time systems
- ARM Cortex-**M** Series:
 - Deeply embedded processors optimized for microcontroller and low-power applications



Cortex: ARMv7 Architecture (A profile)

■ Thumb-2: Power Efficient Integer Execution

- 30% smaller when starting from ARM code
- 30% faster when starting from Thumb code

■ TrustZone: Trusted Secure Environment


- Device integrity, Digital Rights Management, Electronic payment, etc
- Wide industry support



■ Jazelle-RCT: Run Time Compilation Target

- Efficient target for Java, Microsoft .NET MSIL, Perl, Python etc
- Optional DBX Java byte code acceleration
- Early Adopters include Sun Microsystems, Aplix and Esmertec

■ NEON: Multimedia and Signal Processing Architecture

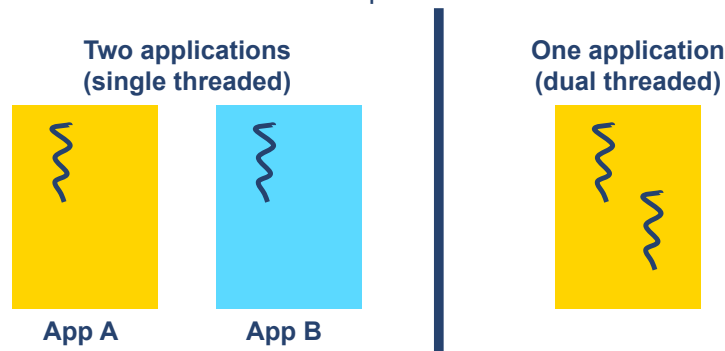
- Significant performance uplift from ARMv6 SIMD
- Supports both Integer and Floating Point SIMD 
- Accelerated software development with



compiler, |

What is a ‘thread’

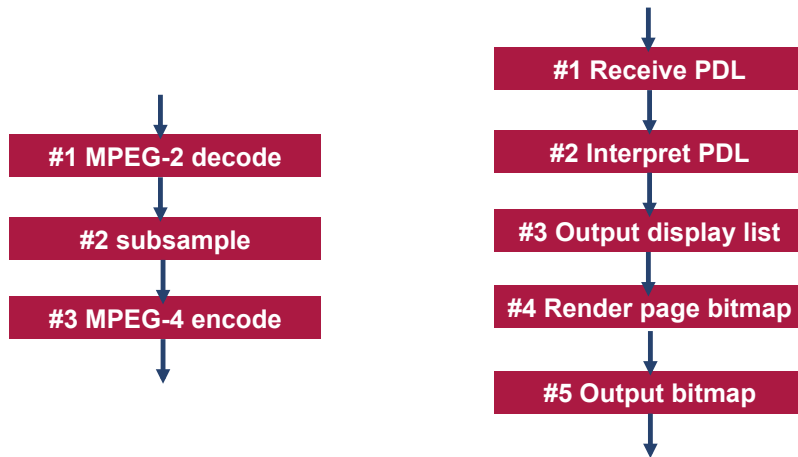
- Term “thread” applies to both MP and MT systems – means ‘thread of execution’
- Key to parallelism is finding independent operations
 - SMP OS will naturally run separate processes on different cores
 - SMP OS processes and device drivers runs on any core
 - Application can be “threaded” if required



- Both examples above make full use of dual core system

Threading 1: Task decomposition

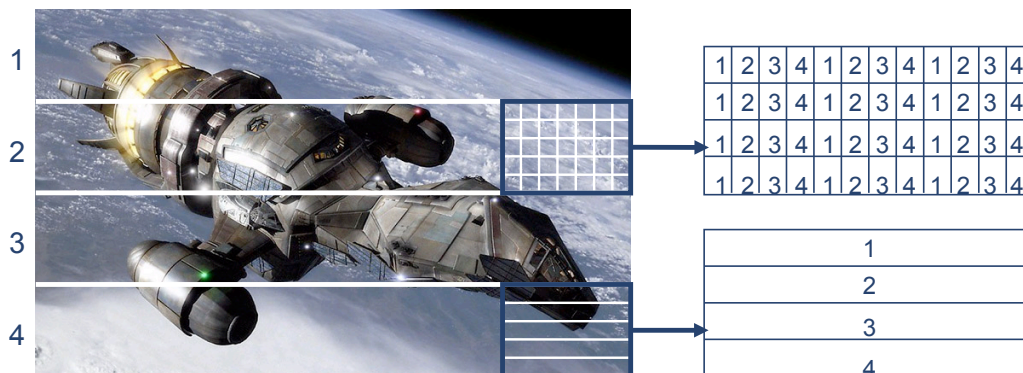
- Application is a pipeline, with each stage in a separate thread
- E.g. video recompress for PMP, or laser printer



- Shared data and semaphores used to pass data around

Threading 2: Data decomposition

- Subdividing a data processing operation into several threads executing in parallel on smaller chunks of data
 - Block by block – 1 block per thread
 - Line by line - 1 line per thread
 - Section by section – 1/4 of picture per thread



ARM MPCore technology dispels the Myths

- #1: Multicore is here now for embedded
- #2: Two cores are not necessarily twice as big as one
- #3: Multiple cores can use less power
- #4: Multicores can have low overheads
- #5: Multicore software is not difficult

