



The Engine of SOC Design

## Embedded Boot Camp: AMP vs. SMP

Grant Martin and Steve Leibson, Tensilica

Electronic Design Processes 2008: Monterey, 17-18 April



### Outline

- SMP
- AMP
- AMP and SMP
- Programming Models
- Example of programming MP for Video
- Conclusions

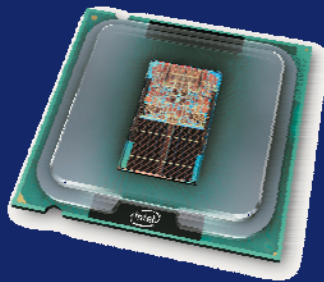
## tensilica SMP: Symmetric MultiProcessing

- A collection of homogeneous cores
- Common view of system resources
- Share a coherent memory space
- CPUs communicate via the large large coherent memory space on multi-core die or bus
- It's what everyone knows and is talking about  
Intel, AMD, SUN, IBM.....

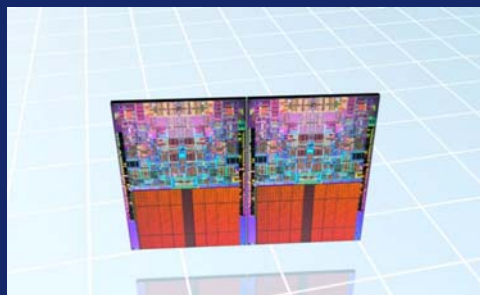
3

© 2008 - Tensilica, Inc

## tensilica Intel SMP Multicore



Intel Core 2 Duo Processor

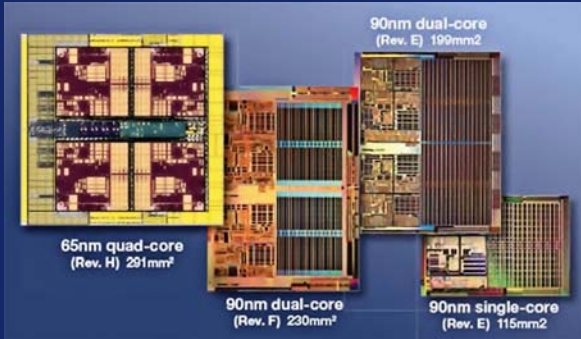
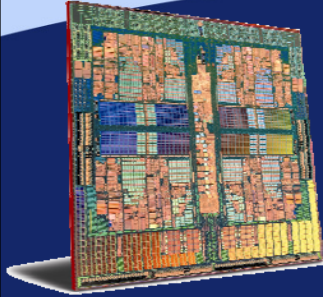


Intel Quad-Core

4

© 2008 - Tensilica, Inc

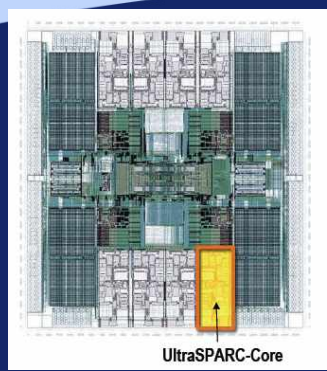
**tensilica** AMD SMP Multicore



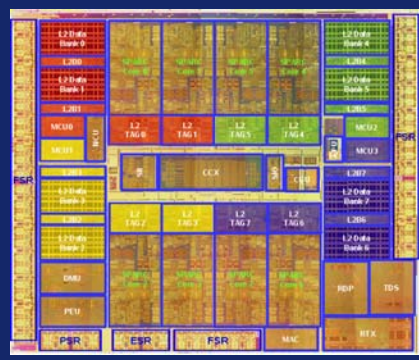
5

© 2008 Tensilica, Inc.

**tensilica** SUN SMP Multicore



UltraSPARC-Core



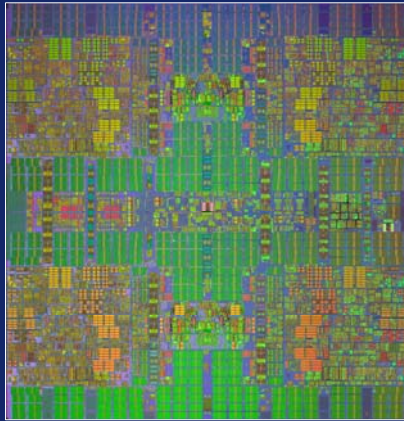
Sun Ultrasparc T1  
Eight CPUs  
Four threads/CPU

Sun Ultrasparc T2  
Eight CPUs  
Eight threads/CPU

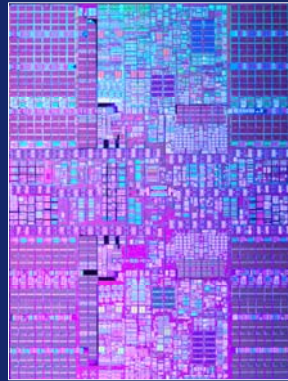
6

© 2008 Tensilica, Inc.

## tensilica IBM SMP Multicore



IBM Z10 processor (four CPUs)



IBM Power6 processor (two CPUs)

7

© 2008 - Tensilica, Inc

## tensilica Why SMP?

- **What is SMP good for?**
  - General Purpose Applications
  - Applications not known at design time
  - Applications that may suspend (e.g. due to memory access) and need to restart on any core
    - Hence large coherent memory
  - Multithreaded applications
  - Large Server applications
    - Web serving by Google a classic
    - Large scientific applications
    - “HPC – High Performance Computing”
- **What is SMP *not* good for?**
  - Not efficient on specific known tasks – performance, area, energy
  - Especially data-intensive ones:
    - audio, video, signal processing
- **Why do we have SMP chips?**
  - Easy for HW designers to step and repeat

8

© 2008 - Tensilica, Inc

**tensilica** AMP: Asymmetric MultiProcessing

- **A collection of heterogeneous, differentiated CPUs**
  - May be hand designed
  - May be generated as ASIPs
- **Many distributed system resources**
  - e.g. local memories
- **Processors communicate via:**
  - Large coherent bus memory
  - Shared local memories
  - HW FIFOs, other direct connections
- **It is what everyone is *not* talking about**
  - Yet there are lots of examples, especially in portable and consumer appliances
  - Cell phones, games, imaging, ...

9

© 2008 - Tensilica, Inc

**tensilica** AMP example  
Epson's REALOID Printer SOCs

- Heterogeneous, asymmetric, 7-CPU design with very little RTL-designed HW
- 90nm process technology, 288 MHz clock rate, >7M gate-count complexity, Less than 2.5W power



Epson PM-T990



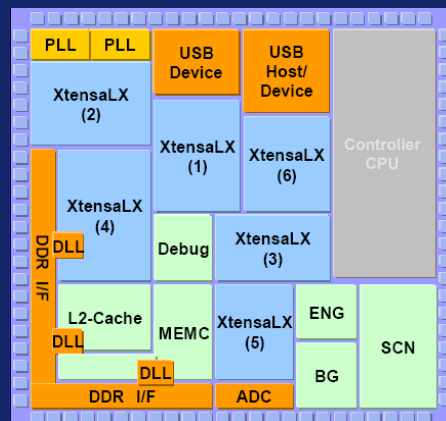
Epson PM-A970



Epson PM-D870  
Epson Stylus  
Photo R380



Epson PM-A920



10

© 2008 - Tensilica, Inc

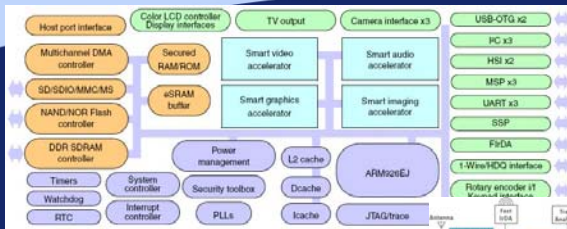
**tensilica** Mediaworks Video/Audio SoC (5 CPUs)



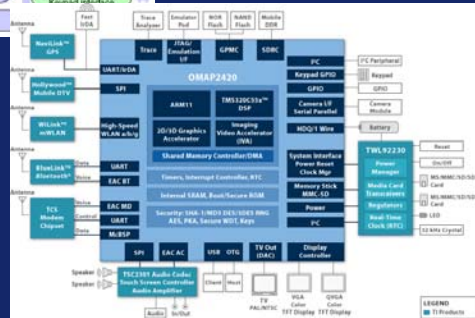
11

© 2008 - Tensilica, Inc.

**tensilica** Media application processors




ST Nomadik STn8815



TI OMAP 2420

12

© 2008 - Tensilica, Inc.




## Why AMP?

- **Why do we have AMP?**  
Real world messier than SMP
- **What is AMP good for?**  
Known data-intensive apps...  
Maximising efficiency for every task  
Audio, video
- **What is AMP *not* good for?**  
Not a pool of general computing resources  
Not efficient if tasks come after the product is designed
- **Why do we have AMP chips?**  
Most economical way to deliver MP to specific tasks  
Performance, energy, area envelope is profoundly better than SMP

13

© 2008 - Tensilica, Inc

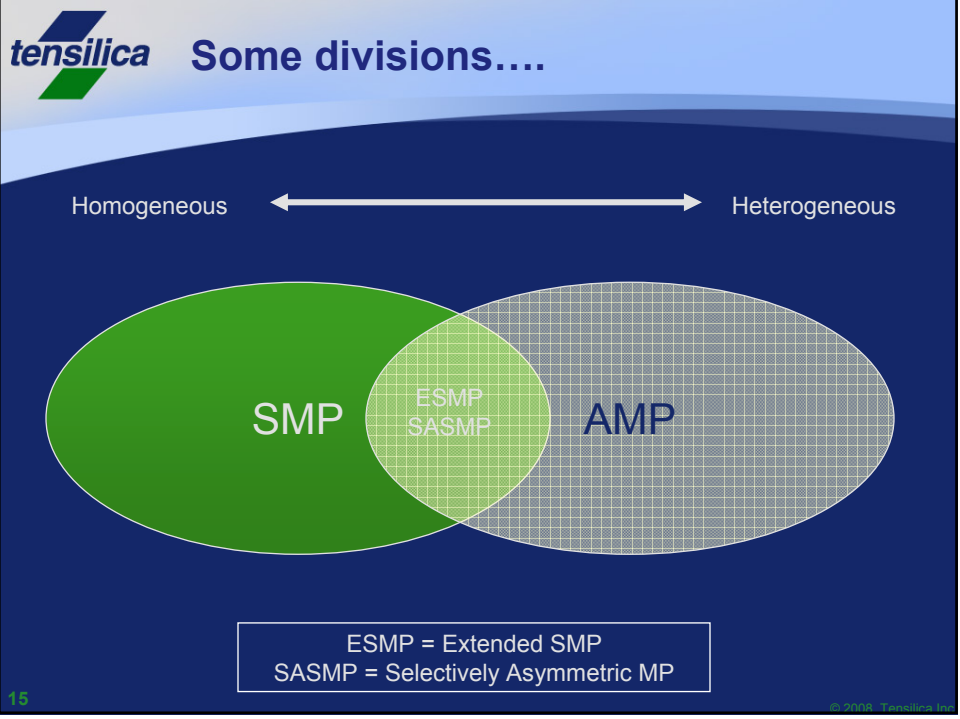


## The real world is messier....

- **Than a simple AMP-SMP division into two parts**  
There are more things in heaven and earth, Horatio,  
Than are dreamt of in your philosophy.  
  
\* Hamlet, scene v  
  
Gallia est omnis divisa in partes tres.  
Julius Caesar

14

© 2008 - Tensilica, Inc



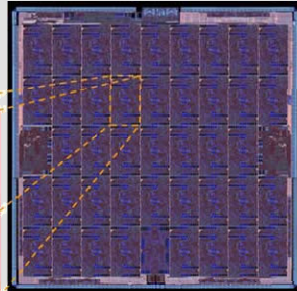
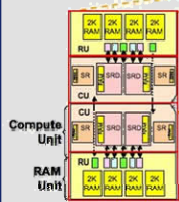
- 
- tensilica** ESMP: Extended SMP
- All CPUs are homogeneous
  - Yet they are not entirely “general purpose”
  - All have the same set of ISA extensions and configuration parameters
    - Oriented to specific classes of applications
    - For example, all may be media processors
    - Or all may be networking processors
- 16 © 2008 - Tensilica, Inc



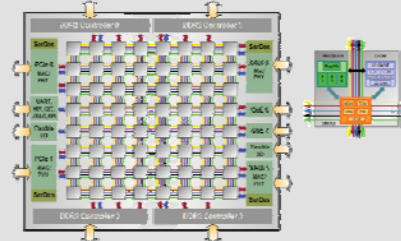


## Other MP architectures: Step and repeat tiles

### Ambric



### Tilera



17

© 2008 - Tensilica, Inc.



## Selectively Asymmetric Symmetric Multi Processing (SASMP)

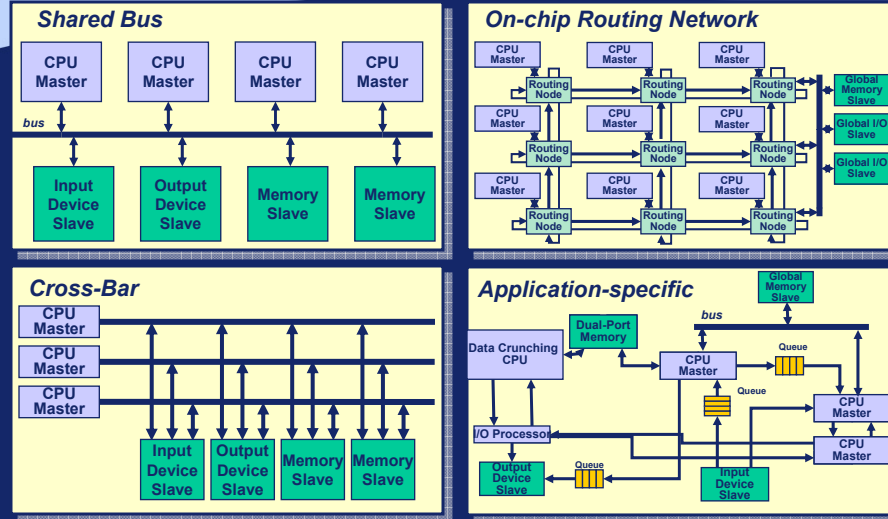
- **Most CPUs are homogeneous SMP, but**  
Some CPUs are selected to be oriented to specific applications
- **WHY?**  
All the benefits of SMP for general purpose processing pool  
The benefits of AMP tailoring for known specific data-intensive tasks that need maximum efficiency  
eg audio, video, .....
- **Where use it?**  
Portable devices for media++  
Run some general tasks as well as known media tasks
- **Needs evolution of SMP OS (eg SMP Linux) to support AMP tasks**  
By defining affinities of tasks to known CPUs  
Or Asymmetric extensions to SMP Linux: e.g. ARTIS (U. de Lille)

18

© 2008 - Tensilica, Inc.



## Don't forget the on-chip communications architecture



19

© 2008 Tensilica, Inc



## Programming Models

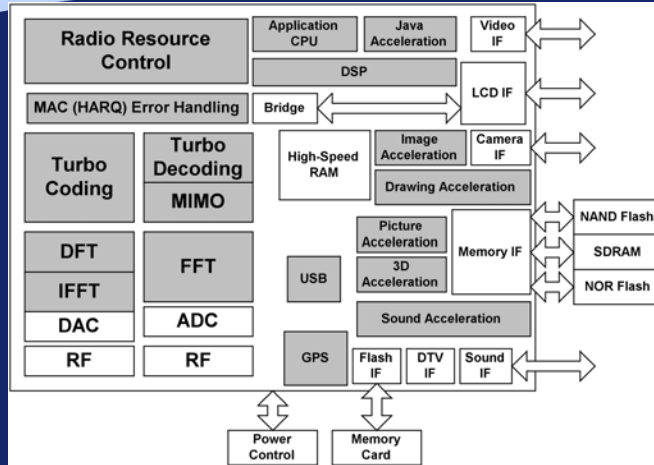
- Conveniently Concurrent:  
Compositional
- Unembarrassingly Parallel  
Datawise
- Ad-hoc

20

© 2008 Tensilica, Inc



# Convenient Concurrency: Composition



Super 3G Mobile Phone Block Diagram

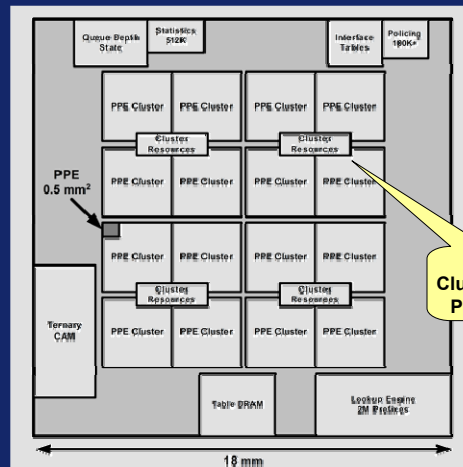
21

© 2008 Tensilica, Inc.



# Don't be embarrassed by this.....

Cisco's Silicon Packet Processor  
 192 Xtensa processor cores per  
 chip – 0.5 sq-mm each, 18M  
 gates, 0.13 micron  
 All data moved via intelligent DMA  
 channels without a common bus  
 (not SMP)



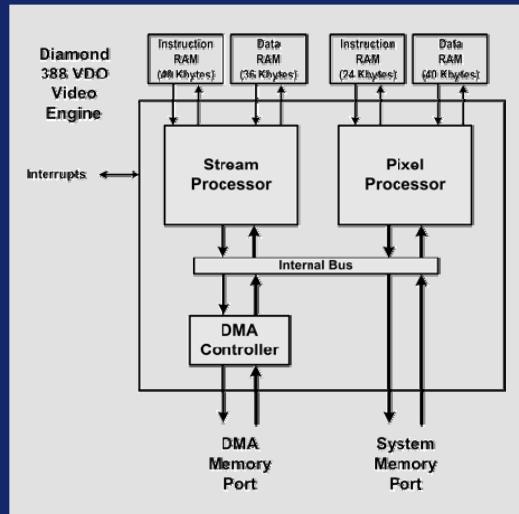
16 PPE  
 Clusters of 12  
 PPEs each

22

Cisco CRS-1 Terabit Router

© 2008 Tensilica, Inc.

**tensilica** Ad-hoc example: Video programming



23

© 2008 - Tensilica, Inc

**tensilica** An ad-hoc Video programming model

- Start with a new video codec
- Start with a Single Core model of a 2-core heterogeneous AMP Video Core:
  - The Single Core model incorporates all the instruction extensions of both cores
- Port/design reference application code on the single core model
- Profile application to find hot-spots on cycle-accurate ISS model
- Speed up hot-spots with relevant using instruction extensions
  - Be careful that 'chunks' of processing use only extensions from one core
  - Simple methodologies to check this
- Split application into 2 sets of files, one per core, emulating DMA data transfer with memcpy
- Port application to multiple cores on ESL model of multicore system, and program DMA
- Performance data on RTL/Emulation model of multicore system
  - Proven on several Video codecs

24

© 2008 - Tensilica, Inc



## New Programming Problems Caused by Old Single-CPU Assumptions

- Latent concurrency issues
- Explicitly re-entrant code
- Priorities no longer assure mutex
- Masking one CPU's interrupts no longer locks
- New possibilities for race conditions
- Inter-processor deadlocks
- Single-CPU crashes can hang entire system
- Silent parallelization issues through parallel APIs
- Bad timing assumptions on task completion
- Weak memory consistency among CPUs

25

© 2008 - Tensilica, Inc.



## Parallel Debugging Issues

- Where will we get programmers that can think in parallel and understand multiprogram behavior?
- Problem repeatability in the face of weak memory consistency models, task-completion timing jitter, and loose inter-CPU timing
- Debugging dissimilar CPUs in parallel
- Coordinating traces among CPUs (where's the master clock?)

26

© 2008 - Tensilica, Inc.



## Conclusions

- **AMP, SMP, ESMP, and SASMP hardware is relatively easy to construct on SOCs**
  - SMP OSeS understood, perhaps not well
  - AMP architectures use multiple OSeS, ~ 1 per CPU
  - Optimizing MP system architectures is hard
- **Programming MP systems is relatively easy**
  - Divide and conquer reduces problem complexity
  - Not likely to be a pushbutton conversion from single-CPU code
  - Harder problems to solve
  - Think about solutions differently (more pipelining, etc.)
  - Debugging MP system code is hard