The Engine of SOC Design

Customized Processors, Customized Interconnects

A Requirement for Next Generation Embedded Systems

David Goodwin, Chief Engineer

ASP-DAC 2007

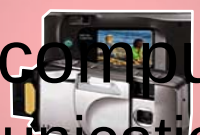
"Interesting" Embedded Systems

- Demanding product requirements
  - Differentiation
  - Cost
  - Power
  - Convergence
- Demanding computation requirements
  - High communication bandwidth
  - High computation bandwidth
- Demanding market requirements
  - Short time-to-market
  - Highly competitive
  - Huge volumes

Portable Devices, Wireless

Digital TV

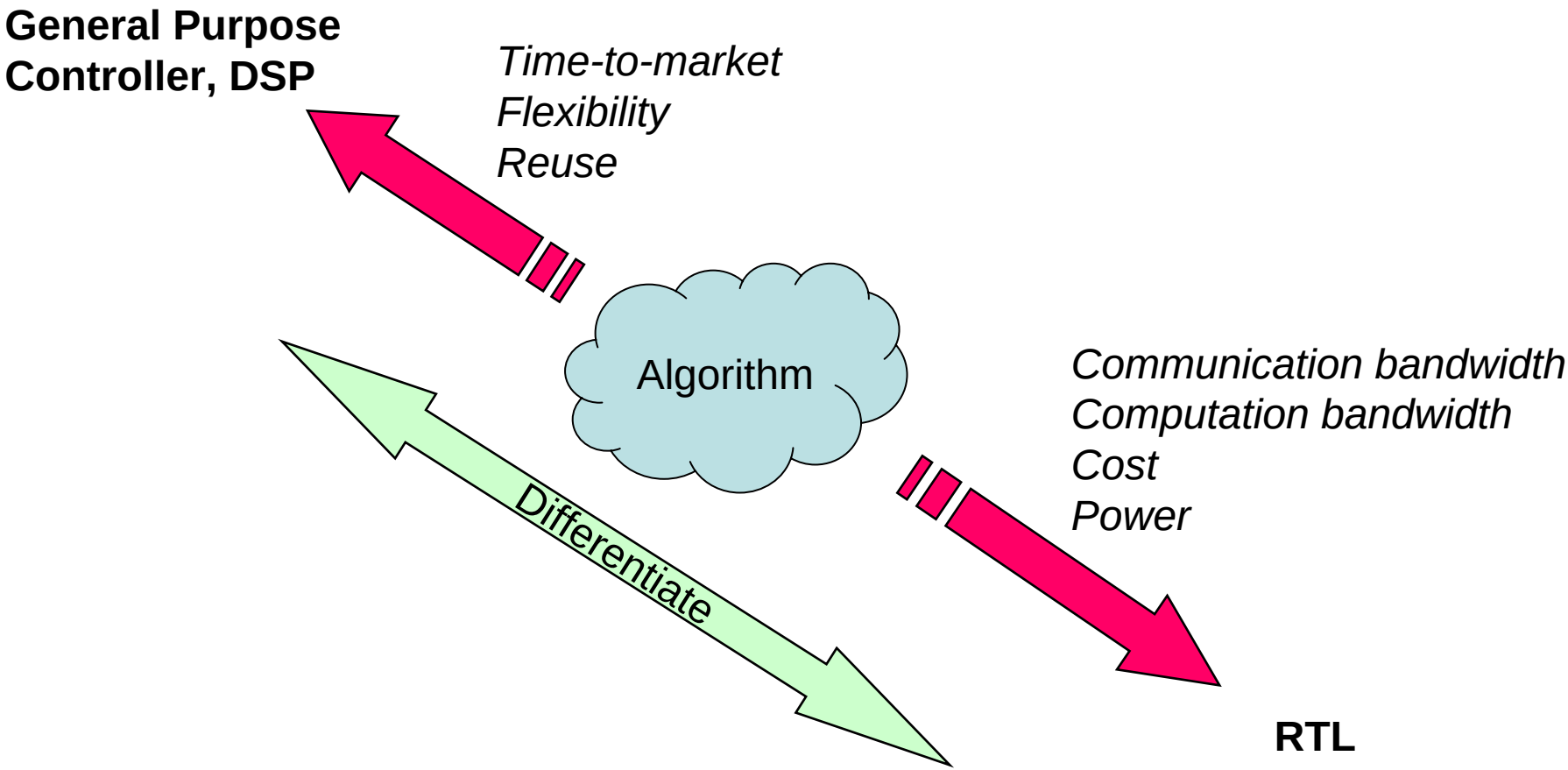
Video, Imaging



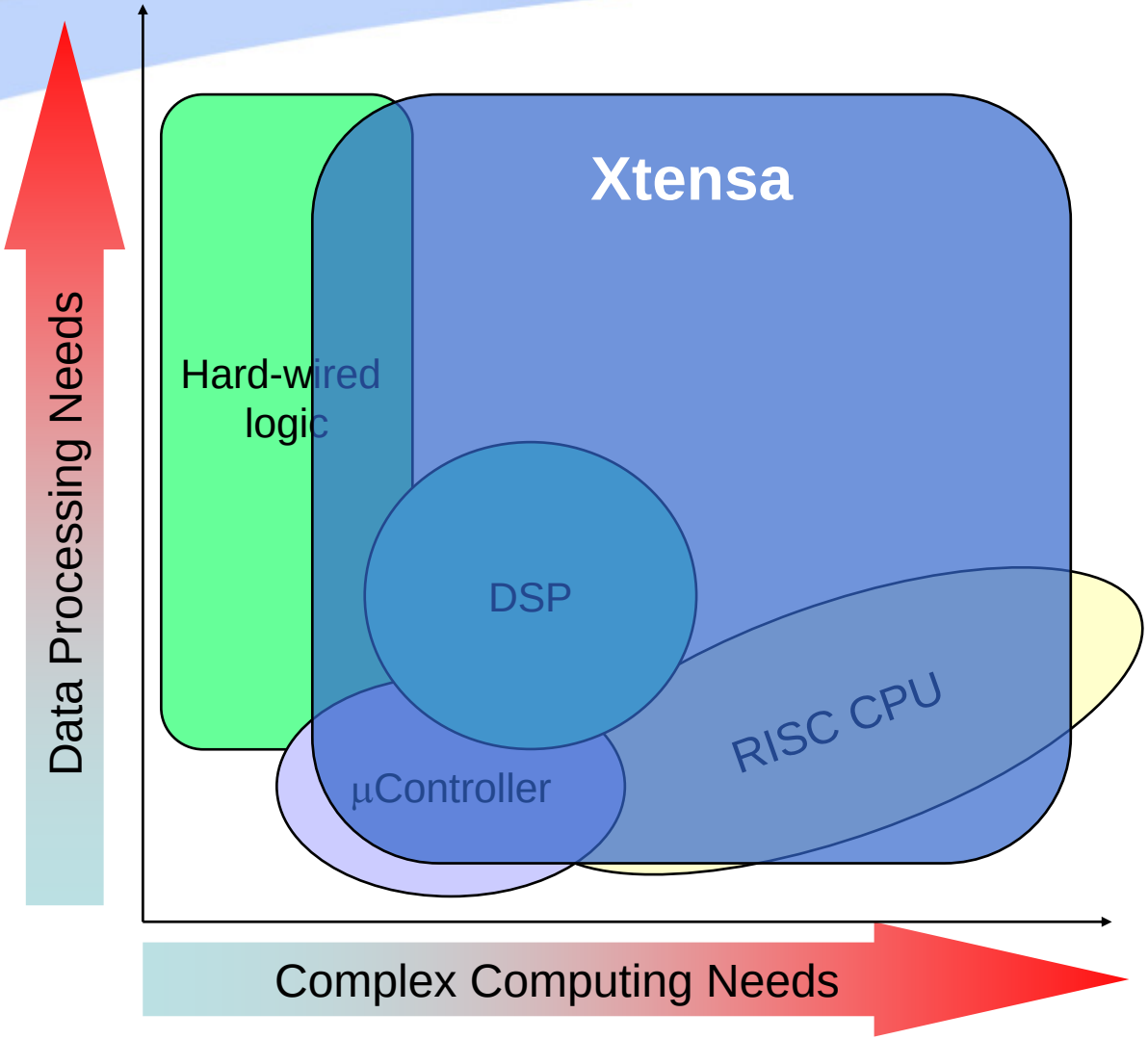
Networking, Storage

Peripherals

Satisfying the Requirements



Xtensa: Covering the Breadth of SOC Processor and RTL Demands



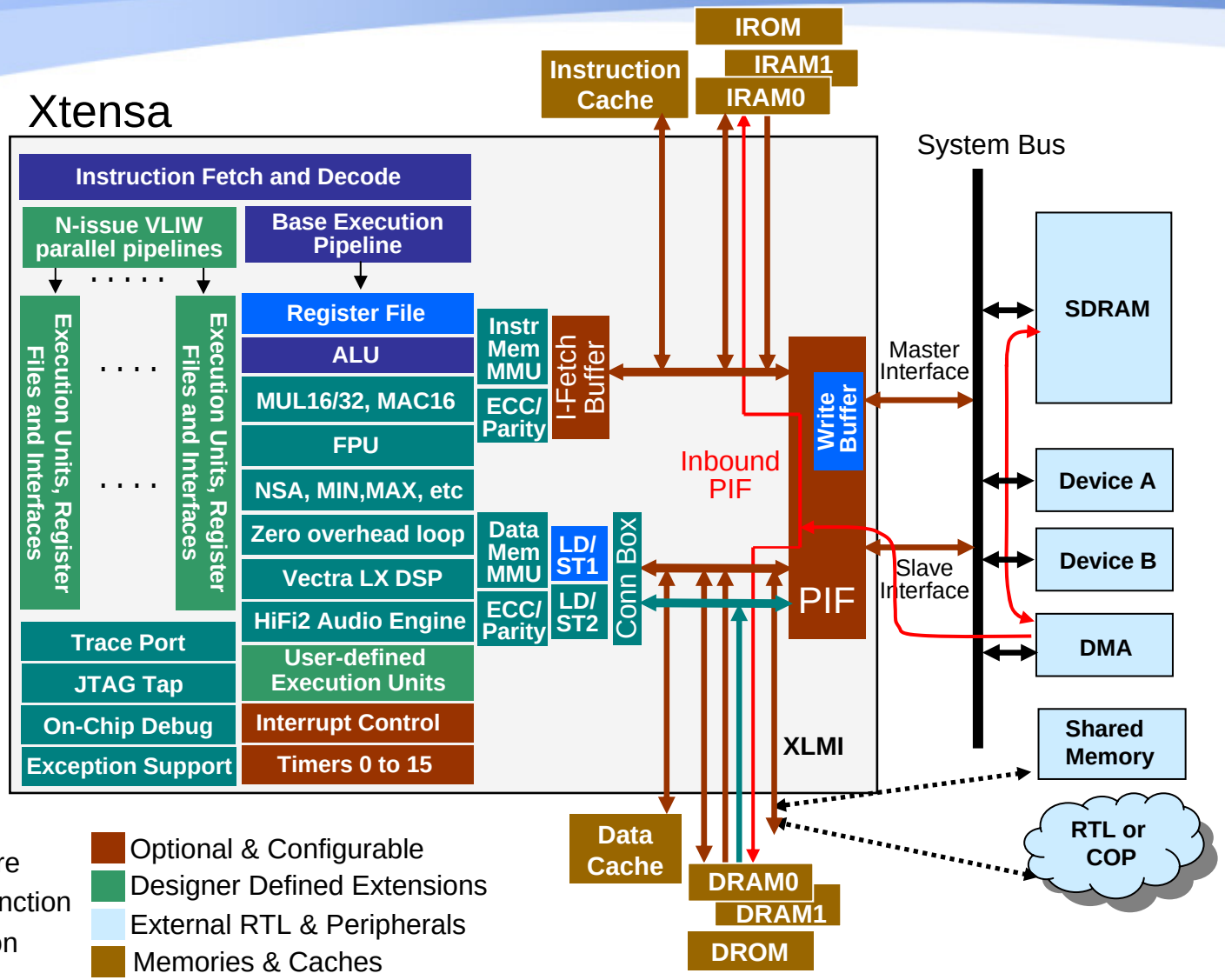
- Tensilica's Xtensa provides the benefits of general-purpose and application-specific processors and software
- Xtensa provides communication and computation bandwidth of RTL

How does Xtensa do this?

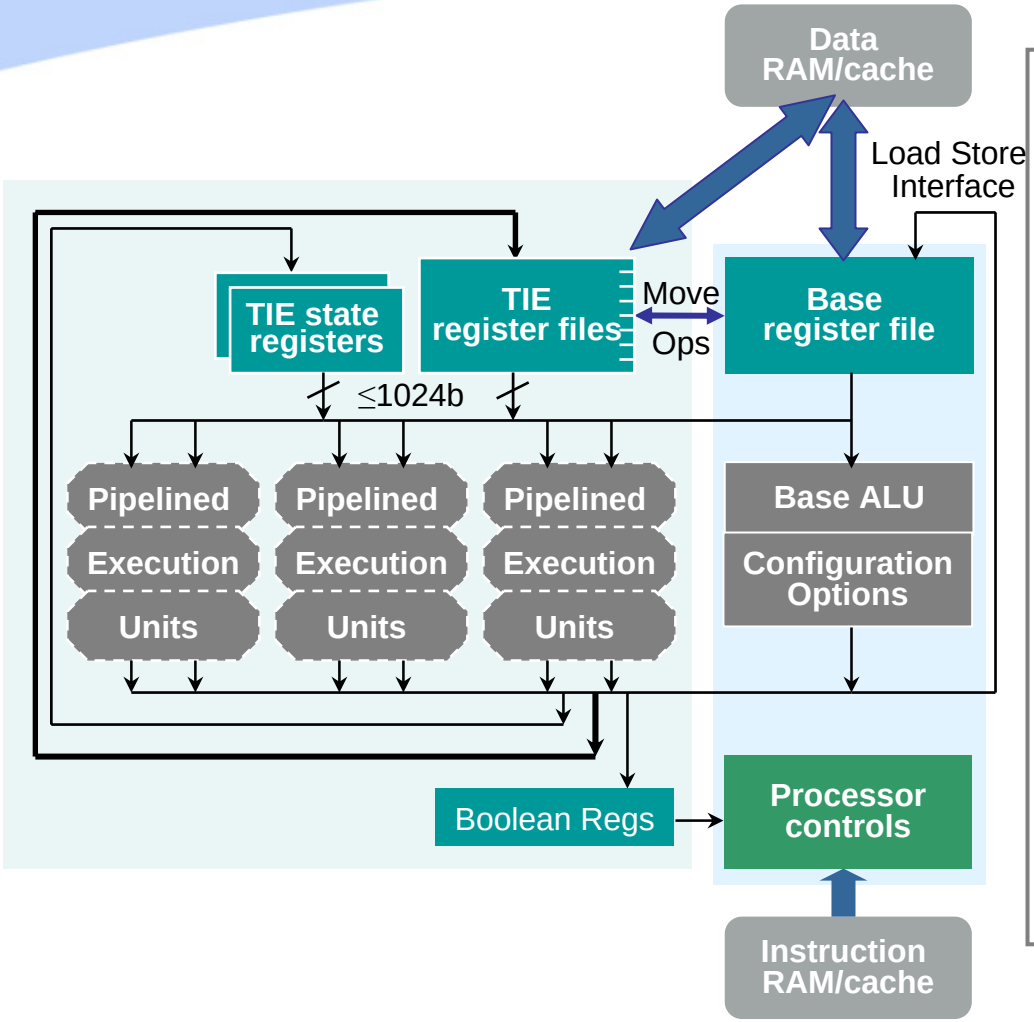
Configurable – satisfy traditional processor requirements in application-specific way to minimize cost and power

Extensible – custom register files, operations, functional units, and pipelines allow RTL-like computation bandwidth.

Configurable and Extensible



Extensibility with Tensilica Instruction Extensions (TIE)



- Start with configured Xtensa core
- Add custom register files and state registers
  - Add custom data types with automatic C/C++ compiler support
- Add custom load/store instructions
  - 1 or 2 load/store units
  - Each up to 128-bit wide
- Add custom function units
  - Multi-cycle
  - SIMD
  - Up to 64 source, destination registers
- Create multi-issue VLIW datapath

Building a Customized Processor

Complete Hardware Design

Source pre-verified RTL, EDA scripts, test suite

Processor Configuration

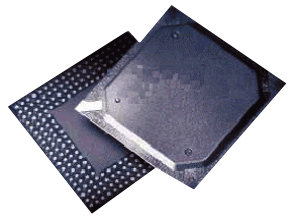
1. Select from menu
2. Explicit instruction description (TIE)
3. Automatic instruction discovery (XPRES)

Xtensa
Processor Generator

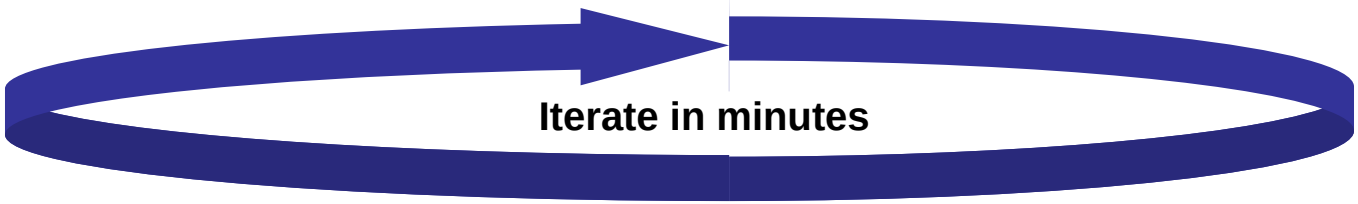
Processor
Extensions

Customized Software Tools

C/C++ compiler
Debuggers, Simulators
RTOSes

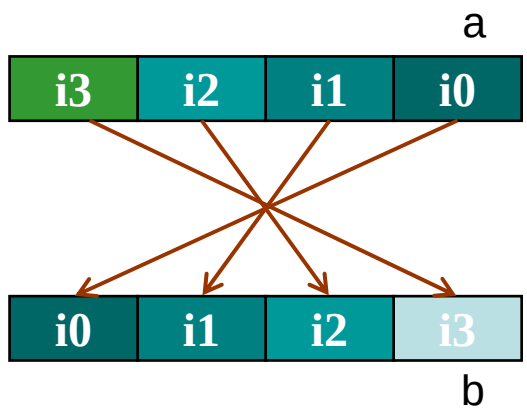


Use standard ASIC/COT design techniques and libraries for *any* IC fabrication process



TIE: Simple Example

```
operation BYTESWAP {out AR a, in AR b} {}
{
  assign a = { b[7:0],b[15:8],b[23:16],b[31:24] };
}
```



The ***operation*** statement describes an entire new instruction, including:

1. Operation name
2. Operation arguments
3. Functional behavior [written in a Verilog subset]

From this single statement, Tensilica tools generate processor hardware, simulation and software development tool support for the new operation.

BYTESWAP Usage

```
// C language syntax
#include <xtensa/tie/TIEFilename.h>
int a, b;
. . .
a = BYTESWAP(b);
. . .
```


```
// Assembly language usage of instruction
byteswap a2, a3
```

- TIE Compiler updates all software tools (assembler, compiler, debugger, simulator, etc.) to recognize BYTESWAP operation
- Use intrinsics in C to use TIE operation
  - TIE also provides a way to have C/C++ compiler automatically recognize operations
- C compiler does scheduling and register allocation

TIE: VLIW Example

- 3 line TIE description generates VLIW processor
  - Many variations possible
- 2 slots of Xtensa base ISA operations
  - Could also include custom operations
- Xtensa C/C++ compiler automatically discovers parallelism in C code and bundles operations into VLIW instructions

TIE for 2-Issue VLIW

```
format f64 64 { slot0, slot1 }

slot_opcodes slot0 { L32I, L32R, L16SI, L16UI, L8UI, S32I, S16I,
    S8I, MUL16S, MUL16U, MULL   ABS, MAX, MAXU, MIN, MINU,
    NEG,  EXTUI, SEXT, AND, OR, XOR, ADD, ADDI, ADDMI,
    ADDX2, ADDX4, ADDX8,    SUB, SUBX2, SUBX4, SUBX8, SLL,
    SLLI, SRA, SRAI, SRC, SRL, SRLI, J, JX, BNEZ, BEQZ, BNE,
    BEQ, BLT, BGE, BGEZ,  BLTZ, BGEU, BLTU, MOVI, MOV.N }

slot_opcodes slot1 { ABS, NEG, EXTUI, SEXT, ADD, ADDI, ADDMI,
    ADDX2, ADDX4, ADDX8, SUB, SUBX2, SUBX4, SUBX8, SLL,
    SLLI, SRA, SRAI, SRC, SRL, SRLI, MOVI, MOV.N }
```

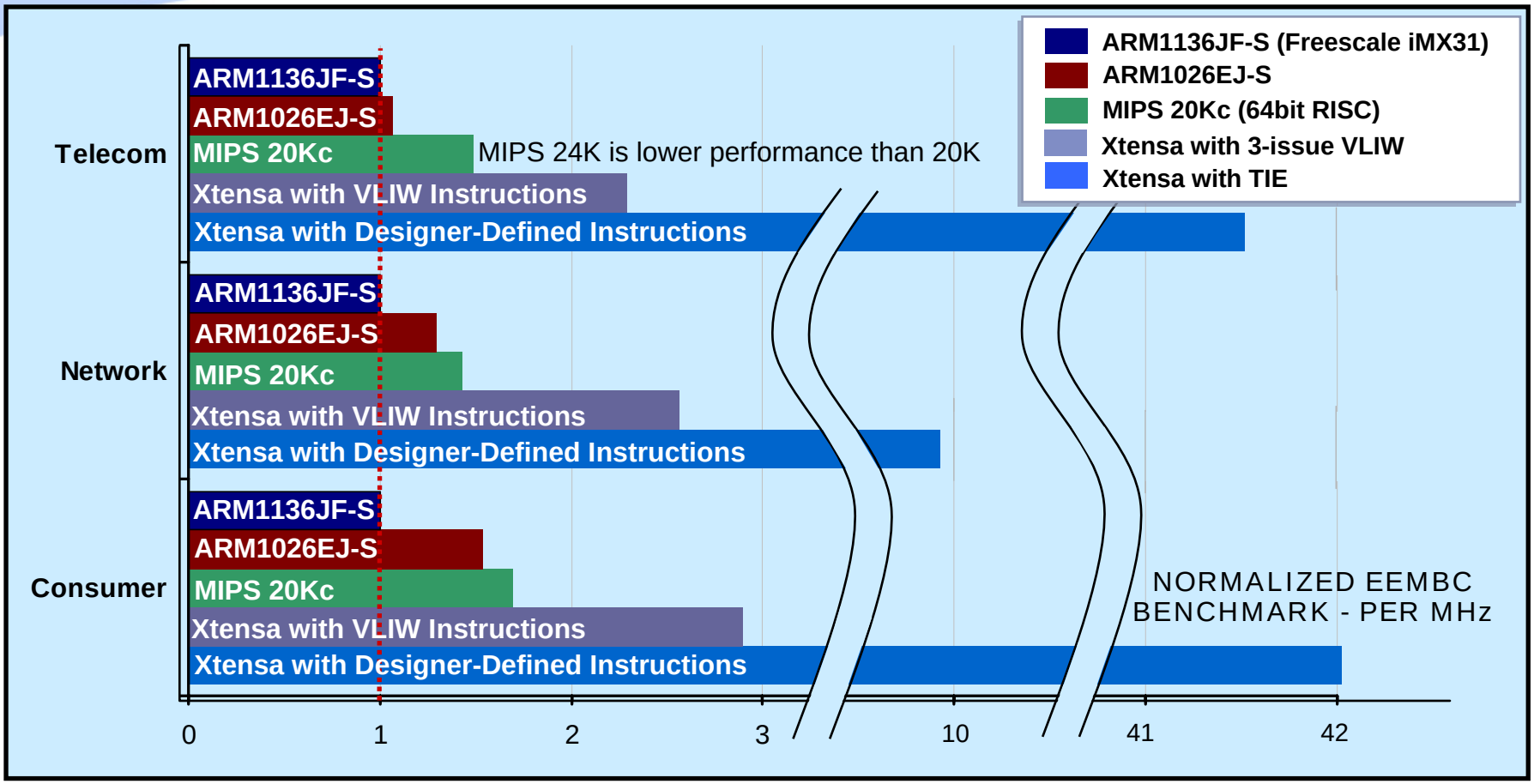
Results

- **46% average performance increase compared to base-case (scalar) Xtensa on EEMBC Consumer Suite**
- **12,500 gates more than base Xtensa**
  - **Approx 40K total gates**
  - **Includes 6-port base register file**
- **325 MHz in 0.13 TSMC LV (worst case)**
- **Power: 0.16 mW/Mhz**

Xtensa: A Superior Base Architecture

And Extending the Processor Extends the Benefits

EEMBC Benchmark Results

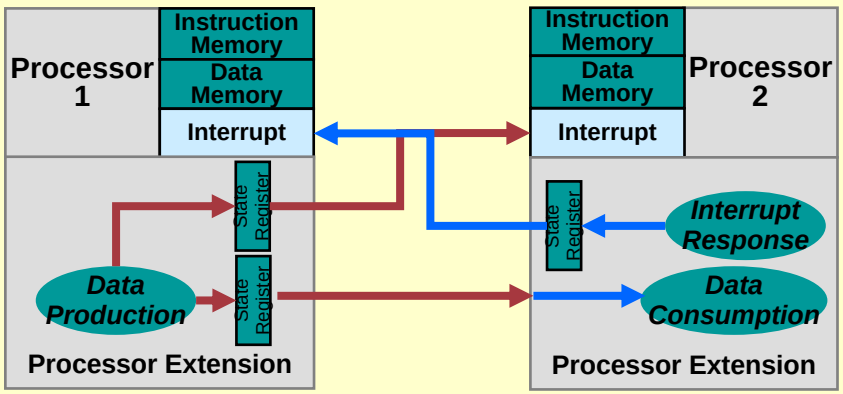


All scores are Simulations of Licensable cores.
All scores are EEMBC/ECL Certified. All scores "out of the box" except Xtensa with TIE (optimized)
Per-MHz certified benchmark scores normalized to ARM = unit score of 1 for suitability in graphing.
Competitive Data as of June 2006. Source: www.eembc.org

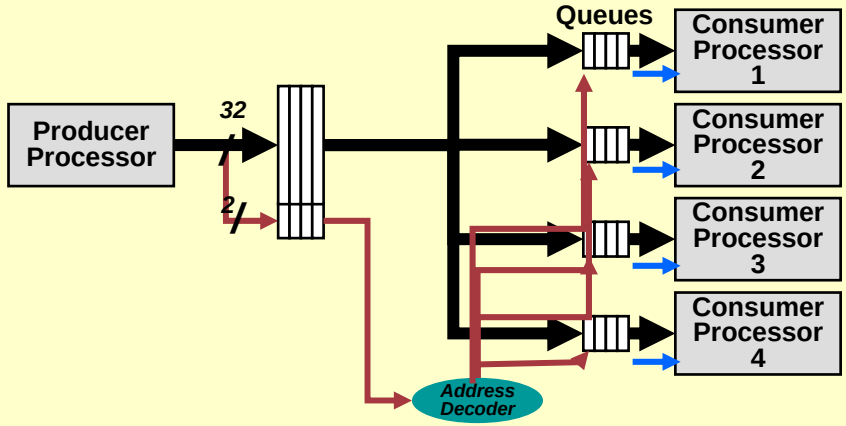
Certified by eCL

Customized Interconnects

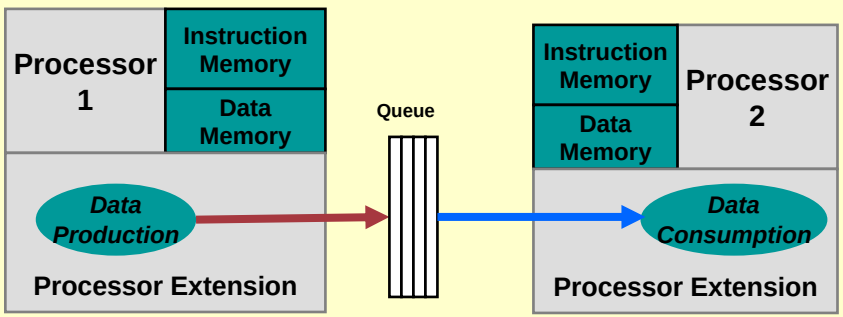
Processor 1 drives data with interrupt handshake



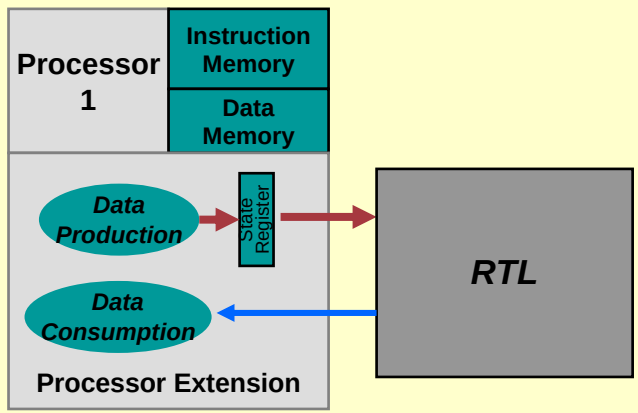
Hardware queues with addressing



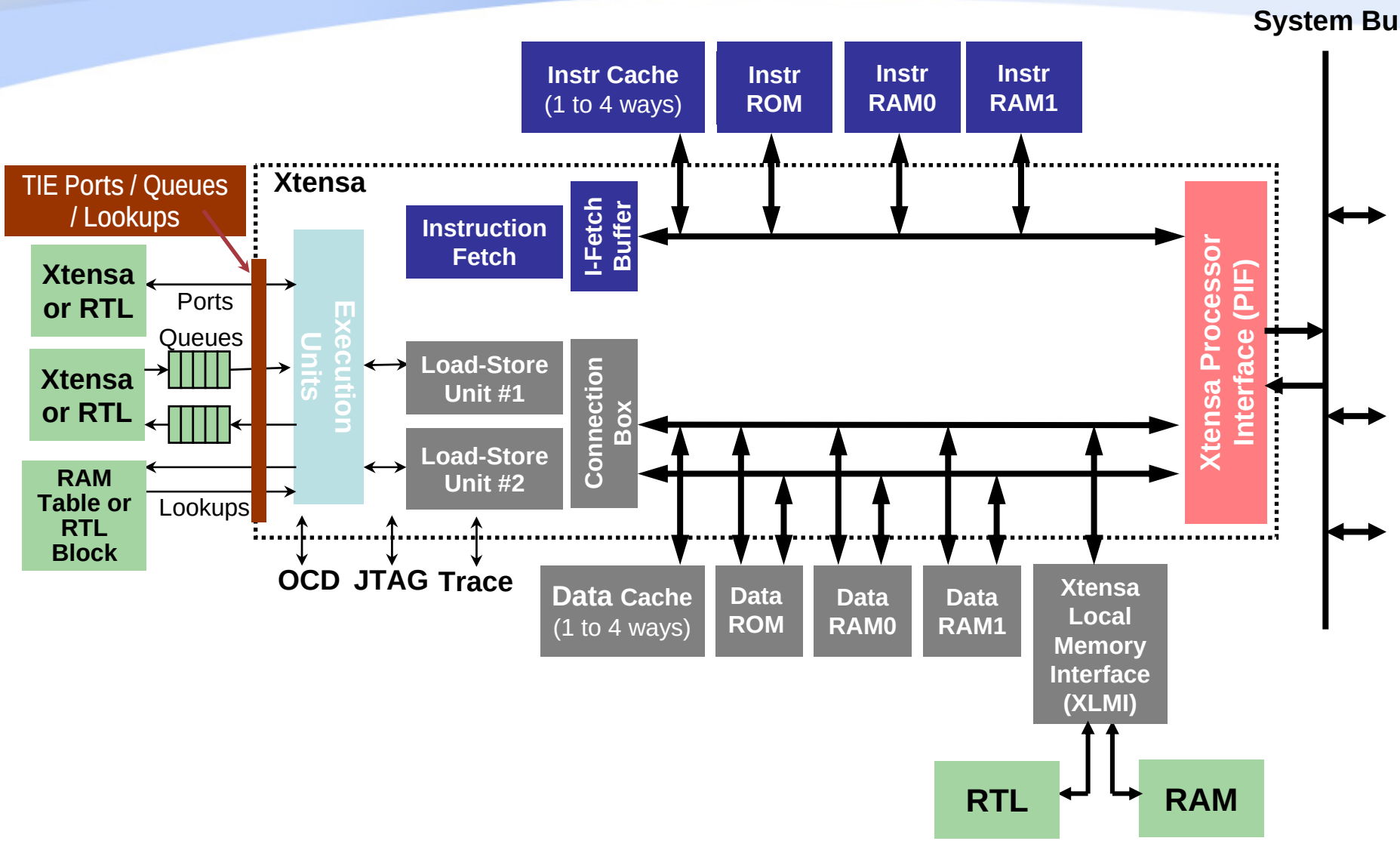
Processor 1 drives data into processor 2 via queue



Direct interface with hardwired logic functions

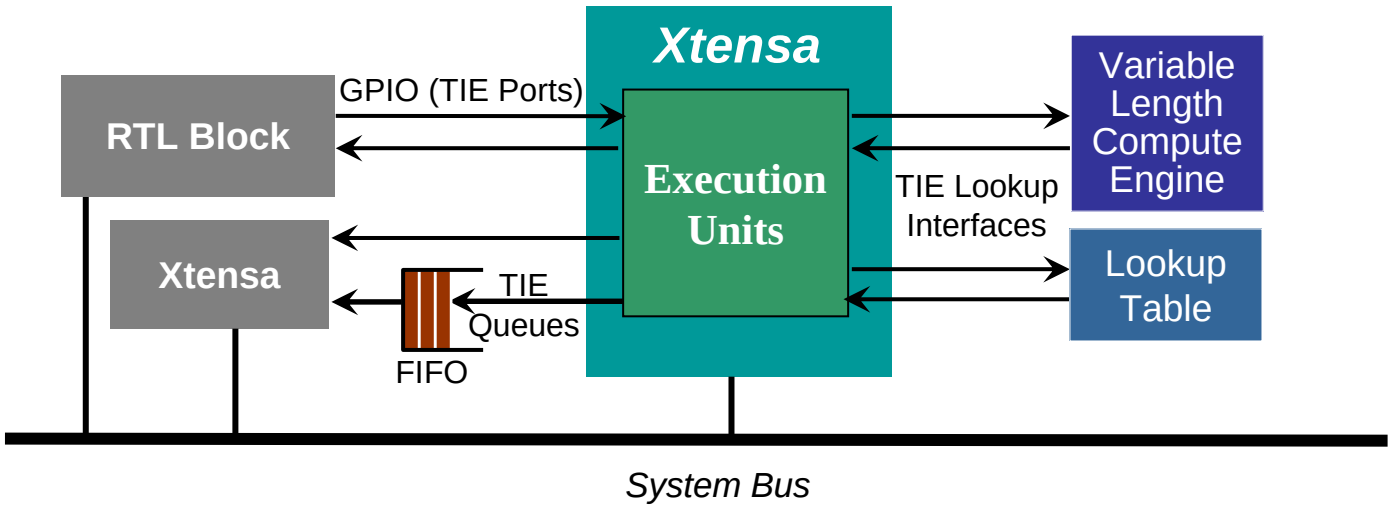


Xtensa Interfaces & Memory Sub-System

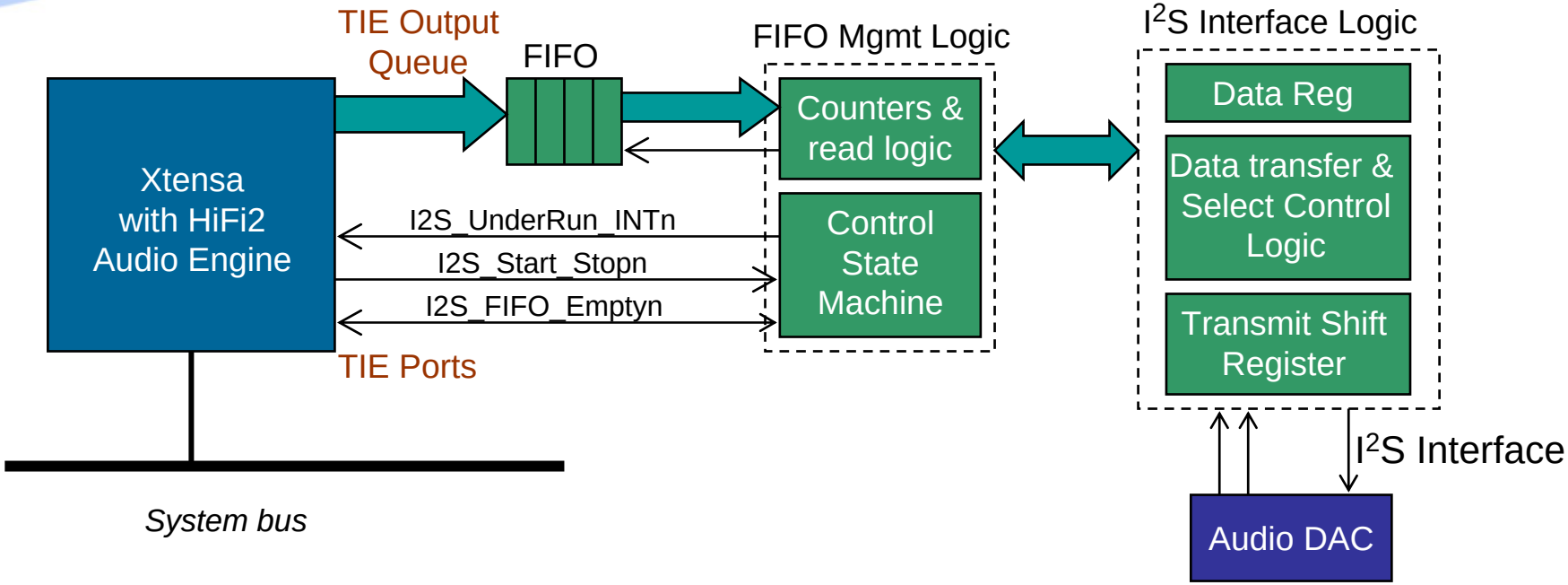


Unlimited *Unconventional* I/O:
Xtensa Port, Queue, & Lookup Interfaces

- Connect wires, queues, RAM tables directly to processor data path
  - –Enables RTL replacement with a Xtensa processor without changing the interface to the rest of the RTL blocks
- Read/write to I/O ports, queues, lookups & operate on data in same cycle
  - –No load or store instructions required to directly access data
- Synchronization is built-in
  - –Automatic processor stall on empty input queue or full output queue



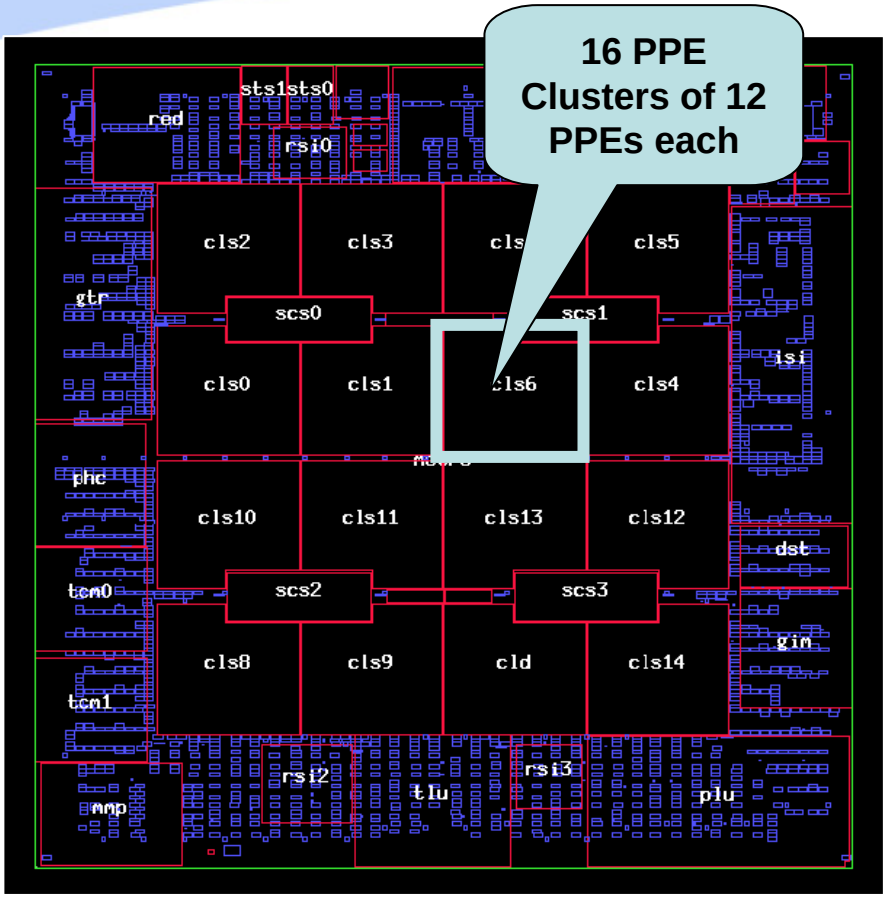
An example of interfacing to external RTL logic using Ports and Queues

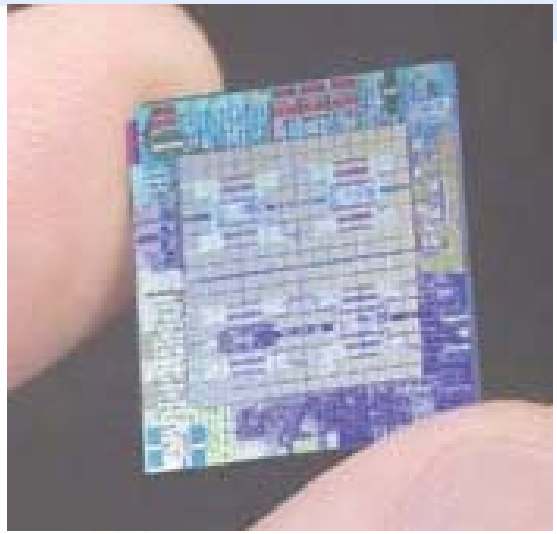


- Use a TIE Queue to send the decoded audio stream to the DAC via the I²S Interface
- Use TIE Ports to exchange control signals with I²S interface logic
- Interface directly with other RTL blocks through ports and queues
  - No need to go through processor system interface

Example: Cisco Multi-Processor NPU

188 Xtensa per chip in *Cisco CRS-1 Terabit Router*

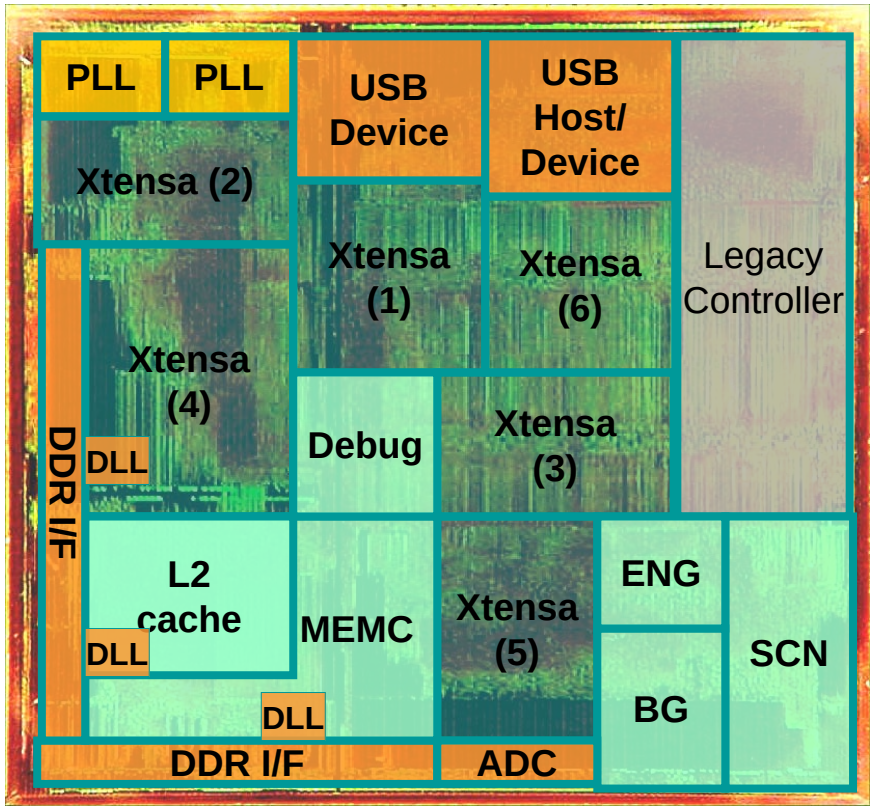




188 Xtensa network processing cores per Silicon Packet Processor. Up to 400,000 processors per system

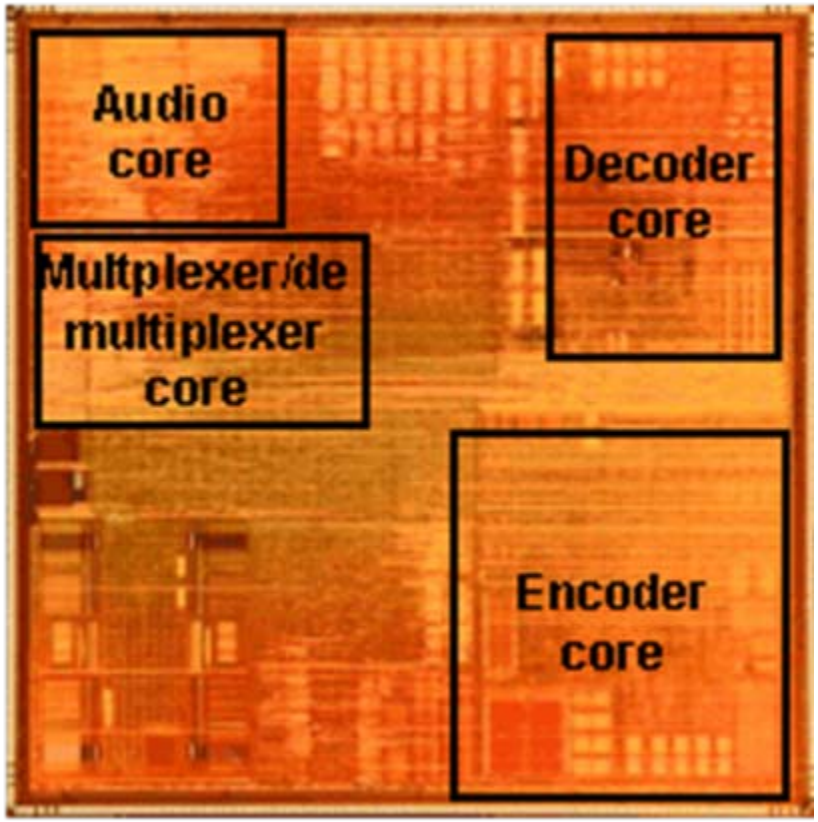
Example: EPSON's REALOID Printer SOCs

Heterogeneous, asymmetric, 6 Xtensa core design



For more details, see the EPSON presentation from the 2006 Nikkei Electronics Processor Symposium / Multi-Core Expo Japan

Performance + Programmability



First High-Definition Camcorder

JVC GR-HD1

Conclusion

- Xtensa processor covers breadth of SOC system requirements
- Customized processors match application computation, size, and power requirements
- Customized interconnects match system communication requirements