

# Xtensa: A Configurable and Extensible Processor

By Suvudhean Dhirakaosal

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

1

## What is Xtensa?

- A configurable and extensible processor.
  - System designers can optimize Xtensa for embedded applications by 1) sizing and selecting features 2) and adding new instructions.
  - Provides easy customization for both hardware and software.
  - Process is simple, fast, and robust.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

2

## Why Xtensa?

- Previously, processor designs are costly and fixed solutions.
- It was not possible to modify these cores for particular application.
- Xtensa lets the system designer select and size only the features required for a given application.
- The designer may define new system-specific instructions if preexisting features don't provide the required functionality.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

3

## Why Xtensa? ...continued

- Xtensa fits easily into the standard ASIC design flow.
- Fully synthesizable, therefore designers can use the popular physical-design tools during the place and route process.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

4

## Processor Development

- Recent research focuses on automatic instruction set design, or reconfigurable/retargetable processors.
- These groups implement automatic instruction set design by systematically analyzing a benchmark program to derive an entirely new instruction set for a given micro architecture.
- Tensilica focuses on how to generate a high-performance and low-power implementation of a given micro architecture with application-specific extensions.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

5

- Reconfigurable processors couple a general purpose computer engine with a lot of hardware programmable logic. (Or entirely hardware programmable logic in extreme case.)
- Compared to non-configurable processor, the reconfigurable processor can be an order of magnitude slower.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

6

- A compromise by Razdan and Smith is to use a custom designed high performance processor with small amounts of hardware-programmable logic.
- Compiler generated information is used to dynamically reconfigure the hardware programmable logic.
- However, differences in operational frequency of the programmable and non programmable hardware requires the system to be simple or deeply pipelined.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

7

- Tensilica processor generator adds application specific functionality at the time the hardware is designed, eliminating need for programmable logic.
- However, this also prevents the designer from modifying the extensions for different applications.
- Another approach included adding coprocessors for application specific functionality, but this increased communication overhead.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

8

- Yet another approach is the modification of processor at the RTL level.
- But this solution is fixed, and any modifications to the extension in the future would require the modification of the RTL level again. (Tedious.)

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

9

- Tensilica uses high-level language, TIE (Tensilica Instruction Extension) to express processor extensions.
- TIE can add new functionality to RLT description and automatically extend software tools. (Allows C/C++)
- No communication overhead since extensions are integral parts of the processor.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

10

## Synthesizable Processors

- Processors used to be custom designed.
  - Sophisticated circuit structures.
  - Efficient (can implement TLA buffers, specialize RAM.)
  - High frequency (700~1000MHz).
- However, requires a lot of development time, and most cases not efficiently designed.
- Not suited for embedded systems:
  - Use different CAD tools than the rest of the system.
  - Hard to modify to better match the application.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

11

- Arrival of Synthesizable Processors.
- Although cannot match raw frequency of custom designed processors, configurability and extensibility more than compensate for difference in maximum operating frequency.
- Easier to integrate into large ASICs. Matches design flow. Quickly manufactured.
- Enables configuration and extension by designer.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

12

## Overview of Xtensa

- Xtensa ISA (Instruction Set Architecture) enables configurability, minimizes code size, reduces power dissipation, and maximizes performance.
- Base ISA defines approximately 80 instructions (superset of traditional 32-bit RISC instruction sets).
- Achieves smaller code size with the use of denser encoding and register window.
  - Compiler use smaller instructions for most common operations.
  - Register window eliminates register saves and restores at entry and exit of subroutines.
- 24-bit and 16-bit instruction formats.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

13

Table 1. Comparison of code size for Xtensa and MIPS.

Benchmark	Description	Xtensa (Kbytes)	MIPS (Kbytes)
Jpeg-6	Image compression and decompression	107	214
Li	SPEC benchmark	30	71
Soft modem	Modern codec	763	1,290
WinWorks	Real-time operating system	339	607

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

14

## Hardware Implementation



Figure 2. The Xtensa pipeline.

- First Xtensa implementation with traditional RISC five stage pipeline.
- Processor accesses instruction cache and tags in first half of I stage. Computes cache hit/miss signal in 2<sup>nd</sup> half.
- Instruction is decoded and register file is accessed in R stage.
- Machine computes effective address for loads and stores and executes ALU instructions on E stage and also determines if conditional branch is taken.
- For loads, the processor accesses data cache in the first half of the M stage and computes the cache hit/miss signal in 2<sup>nd</sup> half.
- Register file is updated in W stage.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

15

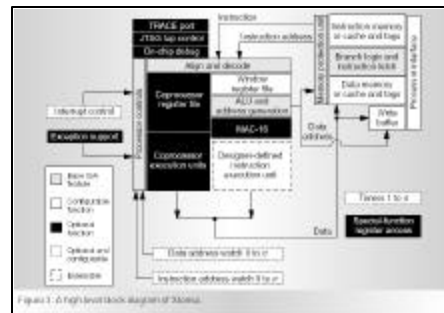


Figure 3. A high-level block diagram of Xtensa.

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

16

Table 2. Examples of Xtensa configuration parameters.

Parameter	Legal values
Instruction cache size	1, 2, 4, 8, and 16-Kbytes
Data cache size	1, 2, 4, 8, and 16-Kbytes
Data ROM size	1, 2, 4, 8, and 16-Kbytes
Size of windowed register file	32, 64 registers
External bus width	32, 64, 128 bits
Number of interrupts	0-32
Interrupt levels	0-6
Timers	0-3
Memory order	Big-endian, little-endian

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

17

Table 3. Xtensa speed, power, and area (0.25 micron, worst-case conditions).

Optimization	Core area (mm <sup>2</sup> )	Gates (Kbytes)	Power (mW/MHz)	Speed (MHz)
Area	1.1	25	0.8	100
Speed	1.5	35	1.0	140

25/2002

CS 269 Hardware and Software Engineering of Embedded Systems

18



- TIE compiler automatically extends software tools. (Adds new instructions as intrinsics to the C and C++ compiler.)
- The semantics of the new instructions are also translated into native C implementation, allowing the designer to verify the functionality of the instructions.

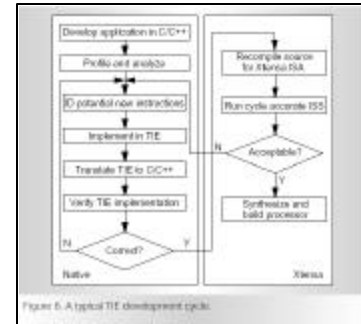


Figure 6. A typical TIE development cycle.

### Advantages of TIE

- New hardware is seamlessly integrated into the pipeline. (No coprocessor communication overhead.)
- Easier to verify due to faster simulation times (four or five orders of magnitude faster than RTL).
- Hardware and software are configured together automatically.

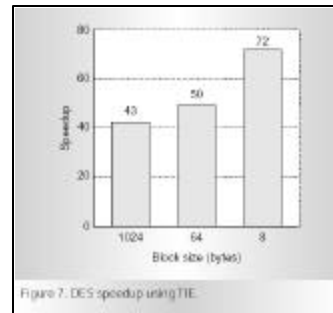


Figure 7. DES speedup using TIE.

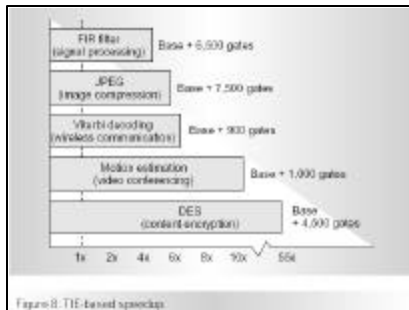


Figure 8. TIE-based speedup.