

Xtensa LX4 Customizable DPU

High Performance with Flexible I/Os and Wide Data Fetches

Product Brief

FEATURES

- Highly efficient, small, low-power 32-bit base architecture
- Configurable over a wide range of pre-verified options including 10 different DSP options
- Extend with designer-defined application-specific instructions, execution units and register files
- Virtually unlimited I/O bandwidth with designer-defined FIFO, GPIO and Lookup interfaces
- Selectable 5- or 7-stage pipeline depth
- Local memories configurable up to 8MB with option for parity or ECC
- Up to 128b wide instructions and up to two 512b wide load/stores and Hardware Prefetch Unit
- Complete matching development tool chain automatically generated for each core

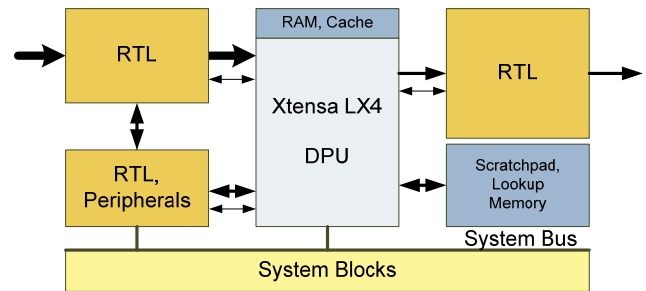
BENEFITS

- Implement hardware for complex dataplane processing dramatically faster than with pure RTL methods
- High bandwidth data flow through processor with flexible I/O interfaces that bypass the system bus
- Quickly and easily scale hardware architecture with task-customized processors
- Lower verification effort with pre-verified correct-by-construction RTL generation
- Post-silicon programmability
- Accurate high-speed processor and system simulation models automatically created for software development
- Higher code density from efficient 16/24-bit ISA leads to memory savings
- Mature, highly-optimizing C/C++ compiler means you can work at the 'C' level for most applications

> 100 Giga-MACs/Second Performance

The Xtensa LX4 processor can reach speeds of 1 GHz on 45nm GS process technology with an area of 0.044 mm² and power of 0.015 mW/MHz.

The Xtensa LX4-based ConnX BBE64-128 DSP can sustain 128 MACs per cycle. Targeted for use in LTE/LTE-Advanced, this equates to over 100 Giga-MACs per second in 28 nm high-performance process technology.



Xtensa LX4: Flexible direct connections allow RTL-like throughput

Processors for the SOC Dataplane

In today's complex SOC designs, processors can be found in many places throughout the chip to add programmability for added flexibility. While most processors do a good job with control functions, they often fail at the complex dataplane processing tasks. That's why designers often turn to RTL blocks for the complex "heavy lifting" SOC tasks. The problem with those RTL blocks is that they take too long to design, take even longer to verify, and are not programmable.

What's needed are processors that can be customized for the task at hand with just the functions, registers and datapath required. In order to provide enough data bandwidth to and from other system blocks, the processor must provide direct connectivity with arbitrary widths and predictable latency without using the system bus.

Xtensa LX4—The Basic Building Block for SOC Design

Tensilica's Xtensa LX4 DPU (dataplane processing unit) was designed from the start to be a basic building block in SOC designs. It is ideal for handling complex compute-intensive DSP applications where an RTL implementation may be the only other option. Xtensa LX4 DPUs are configurable and extensible to meet application requirements exactly.

Configurable: You are offered a menu of pre-verified checkbox and drop-down options ranging from memory size and width to complex DSP functions.

Extensible: You can use our TIE (Tensilica Instruction Extension) methodology, based on the Verilog language, to implement the datapath elements in the processor pipeline. The control FSM (finite state machine) can be implemented as software running on the processor. Just specify the functional behavior of the new datapath and the RTL is automatically generated, along with the full matching software toolchain and models.





Feature Overview

Backwards-compatible ISA since 1999

- Fundamentally architected for extensibility
- Base instruction set of 80 instructions
- All optional blocks are still available
- Any differentiating designer-defined instructions written in 1999 can be re-used today

Optional pre-defined execution units

- 32-bit multiplier and/or 16-bit multiplier and MAC
- Single-precision floating point unit
- Double-precision floating point acceleration
- 3-way 64-bit VLIW (VLIW3)
- Pre-defined 32-bit GPIO and FIFO-like Queue interfaces

Optional execution units for additional licensing:

- ConnX Vectra LX DSP engine
- ConnX Vectra VMB for baseband acceleration
- ConnX D2 DSP engine
- ConnX BBE16 Baseband engines
- HiFi 2 and HiFi EP Audio engines

Differentiate with designer-defined instructions

- Make your specific algorithm run even more efficiently by adding the instructions it needs
- Development tools automatically adapt for full support

Natural connectivity with RTL blocks

- Multiple custom width I/O Ports for peripheral control and monitoring
- Multiple custom width Queue interfaces to FIFOs for data streaming into and out of the processor
- Co-simulation with RTL down to the pin level in SystemC

Highly configurable interfaces

- Optional processor interface (PIF) to system bus, choice of 32-, 64-, or 128-bit width with in-bound slave DMA option
- Up to 128b wide instructions and up to two 512b wide load/stores and Hardware Prefetch Unit
- Write buffer: selectable from 1-32 entries
- Optional second data Load/Store unit
- Optional AMBA AXI and AHB-Lite bridges with synchronous or asynchronous clocking
- Choice of 1-, 2- or 4-way cache and/or local memories
- Up to 32 interrupts

Multi-core design style support

- Multi-core system creation, modeling and SystemC co-

simulation out-of-the-box, fully supported within the Xtensa Xplorer IDE

- Homogenous and heterogeneous subsystems supported
- Inter-core on-chip debug with break-in/out control
- Optional 16-bit processor ID
- Conditional store option and synchronization library provide shared memory semaphore operations and the “release consistency model” of memory access ordering

Complete hardware implementation and verification flow support

- Automatic generation of RTL and tailored EDA scripts for leading-edge process technologies, including physical synthesis and 3D extraction tools
- Auto-insertion of fine-grained clock gating for low power
- Hardware emulation support including automated FPGA netlist generation
- Comprehensive diagnostic test bench
- Formal verification support for designer-defined instructions

High-speed, high-accuracy system simulation models automatically created

- High-speed instruction-accurate simulator for software development
- Pipeline-modeling, cycle-accurate Xtensa instruction set simulator
- Xtensa SystemC (XTSC) transaction-level modeling (TLM) support, including out-of-the-box multi-core simulation
- Hardware co-simulation with RTL in SystemC with Tensilica's pin-level XTSC

Integrated design environment

- Create, simulate, debug and profile whole designs in one tool—Xtensa Xplorer is a high productivity IDE
- Ninth generation software development tools target each processor. The advanced Xtensa C/C++ compiler (XCC) includes optimizations for base, optional and designer-defined instructions
- New Vectorization Assistant directs the programmer to areas of the application that can benefit most from modifications to enable better Vectorization
- Multi-core subsystem design and simulation support
- Custom data display formatting for easy debug of vector and fixed-point data types as well as bit-mapped status and control

Robust operating system support

- Use Mentor Graphics Nucleus+, Express Logic's ThreadX, Micrium's uC/OS-II or the Linux operating systems



Efficient Base Architecture

The Xtensa LX4 32-bit architecture features a compact instruction set optimized for embedded designs. The base architecture has a 32-bit ALU, up to 64 general-purpose physical registers, six special purpose registers, and 80 base instructions, including improved 16- and 24-bit (rather than 32-bit) RISC instruction encoding. Key features include:

- A wide range of configurable options to ensure you get just the logic you need to meet your functional and performance requirements
- Modelessly intermix standard 16- and 24- as well as custom 32-, 64- or 128-bit FLIX (VLIW) instructions for lowest code and performance overhead.
- Selectable 5-or-7-stage pipeline to match memory
- Virtually unlimited I/O bandwidth with optional Queue (FIFO), Port (GPIO) and Lookup interfaces for data transfers that don't require system bus bandwidth
- One or two 32/64/128/256/512-bit wide Load/Store units
- Local memories configurable up to 8MB with optional parity or ECC
- Optional hardware prefetch reduces memory latencies
- Automated fine-grained clock gating throughout processor for ultra-low power solutions
- Can be multi-issue VLIW architecture for parallel instruction execution with FLIX™

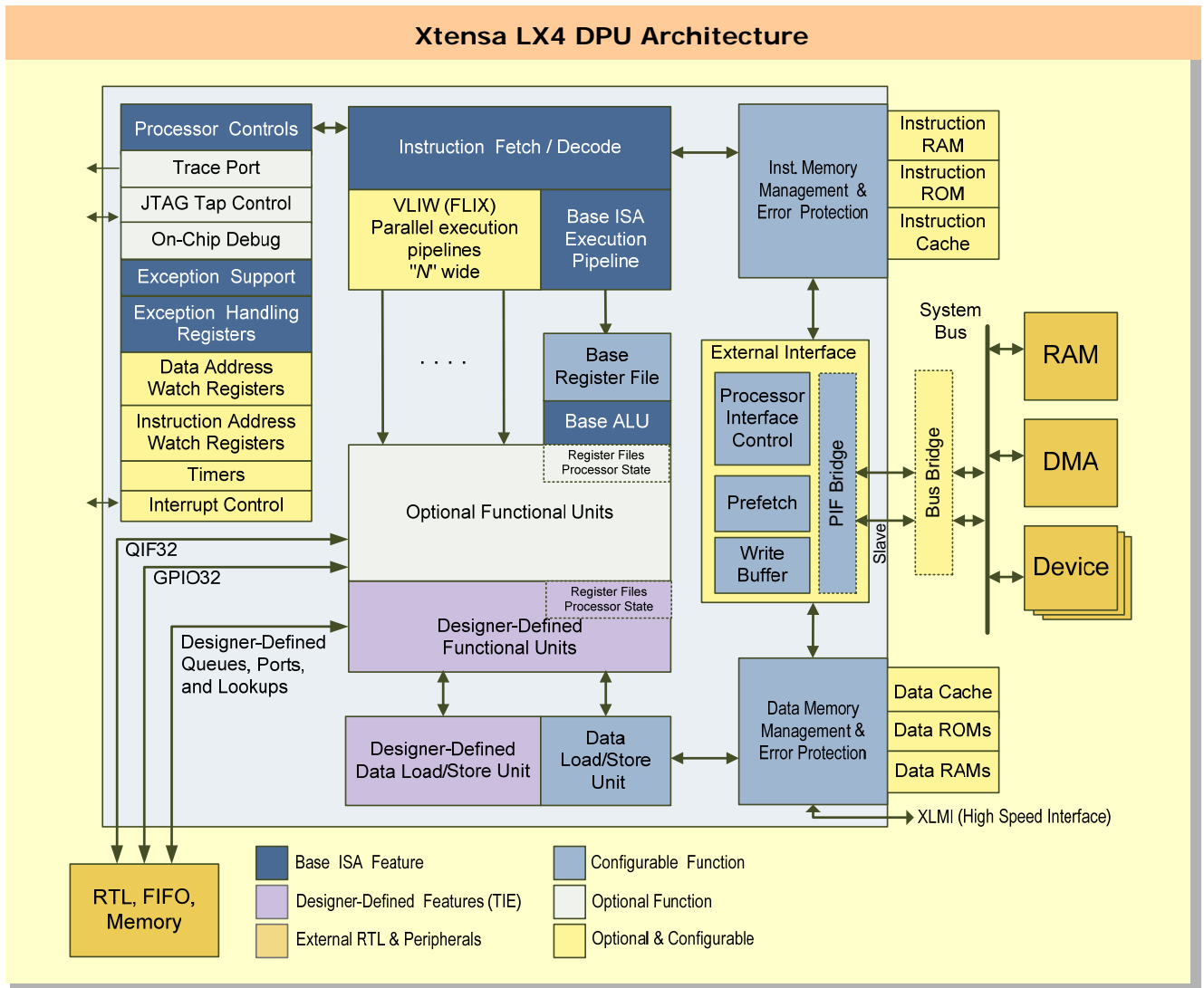


Figure 1. Xtensa LX4 DPU showing standard, optional, and designer-defined blocks





Base Instruction Set Compatibility

Configurability of a Tensilica processor core never compromises the underlying base Xtensa instruction set architecture (ISA), thereby ensuring availability of a robust ecosystem of third party application software and development tools. All configurable, extensible Xtensa processors are always compatible with major operating systems, debug probes, and ICE solutions. For each processor, the automatically generated complete software development toolchain includes an advanced Integrated Development Environment (IDE) based on the ECLIPSE framework, a world-class C/C++ compiler, a cycle-accurate SystemC-compatible instruction set simulator, and the full industry standard GNU Toolchain.

Tensilica uses an ISA that has been backwards compatible since its introduction in 1999. It uses a base instruction set of 80 instructions and was fundamentally architected for extensibility. Designers can run application code written back in 1999 and it will run on the Xtensa LX4 processor today. Any differentiating designer-defined instructions from earlier designs can be re-used today.

Smaller Code Size

The Xtensa LX4 DPU can modelessly intermix 24-bit and 16-bit instructions, leading to 25-50% better code density and, therefore, smaller memories than mixed 32- and 16-bit architectures. Since memories typically dominate SOC area, this code density advantage translates into significant SOC area savings.

Powerful Base ISA

The Xtensa ISA includes powerful compare-and-branch instructions and zero-overhead loops, which allow the compiler to generate tight, optimized loops. It also provides bit manipulations including funnel shifts and field-extract operations that are critical for applications such as networking that process the fields in packet headers and perform rule-based checks.

Extensible ISA

One of the fundamental technology innovations in the Xtensa processor is the ability to easily and seamlessly add new instructions into the processor's datapath. The associated C data types, software tool chain support and the EDA scripts required to synthesize the processor are all generated automatically, just as if they had been there from the start. The specification of this new datapath and associated instructions and C data types is written in the Tensilica Instruction Extension (TIE) language, which is explained in more detail in a later section.

Highly Configurable Functionality

Select from click-box options to add functionality to your processor and evaluate performance improvements in a matter of minutes. Basic interface options include:

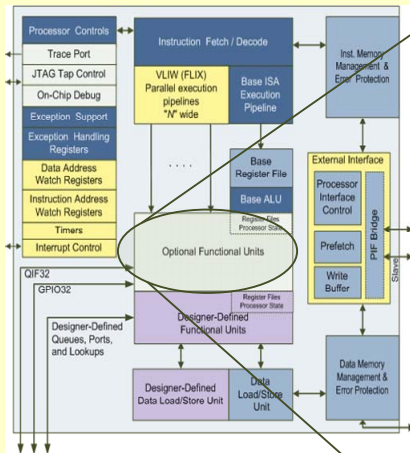
- Processor interface (PIF)
 - Width: 32/64/128-bit
 - Optional "no PIF" configuration
- Optional AMBA AXI and AHB-Lite bridges with synchronous or asynchronous clocking
- Inbound DMA option
- XLMI high-speed local interface
- Big-Endian/Little-Endian byte ordering
- Choice of 1 or 2 general-purpose load/store units, each 32-, 64-, 128-, 256- or 512-bits wide
- On-chip debug port (IEEE 1149.1 compliant)
- Trace port signals
- Up to 32 interrupts with up to 7 levels of priority plus a separate Non-Maskable Interrupt level
- Write buffer: selectable from 1 to 32 entries
- Multiple custom width GPIO ports for direct control and monitoring of peripherals
- Multiple custom width queue interfaces for streaming data into and out of the processor via FIFOs
- Single 16-bit MAC (multiply accumulator)
- 16- or 32-bit multipliers
- Low-area integer divider
- Single precision floating point unit
- Double precision floating point accelerator
- Optional AMBA AXI and AHB-Lite bridges with synchronous or asynchronous clocking
- Designer-defined Queues, Ports, and Lookups
- Optional 16-bit processor ID
- Support of 32-, 64 -or 128-bit wide FLIX instructions

Memory subsystem options include:

- 7-stage pipeline option
- Local data and instruction caches
 - Up to 4-way set associative
 - Up to 32 KB
 - Write-back and write-through cache write policy
- Memory management options
- Region protection
- Region protection with translation
- Memory Management Unit (MMU) with Translation Look Aside Buffers (TLBs), includes no-execute bit security support
- MMU for the Linux operating system
- Up to 6 local memory banks can be connected for instruction and data accesses (up to 12 in total). Memory banks may be local ROM, RAM or cache ways.
- Hardware prefetch for reducing long memory latencies
- Local data and instruction memories up to 8 MBytes
- Optional parity or ECC for all local memories



Xtensa LX4 Major Configuration Options



Options

Register Files
Processor State

MAC 16 DSP

MUL 16/32

Integer Divider

Single Precision
Floating Point (FP)

Double Precision
FP Acceleration

32-bit GPIO
(GPIO32)

32-bit Queue
Interface (QIF32)

Choose Pre-verified
functionality.

Click-box options and side-
by-side profiling allow easy
"what-if" assessments

VLIW (FLIX) Options

HiFi 2 or HiFi EP Audio Engine *

ConnX D2 DSP Engine *

ConnX Vectra LX DSP Engine (1, 2 Load Stores) *

ConnX BBE16/64 DSPs *

VectraVMB (DSP Acceleration Instructions) *

FLIX3 (3-way FLIX configuration)

* Available for licensing in addition to Xtensa LX4

Figure 2. Tensilica offers the widest range of configurable functional units for the Xtensa LX4 DPU

Configurable Options

Tensilica offers pre-verified options that you can just add to your designs when they are needed, so you don't have to spend the time designing these yourself. Tensilica's configurable options include:

- Single 16-bit MAC (multiply accumulator)
- 16- or 32-bit multipliers
- Single precision floating point
- Double precision floating point acceleration
- A pair of 32-bit direct GPIO interfaces (GPIO32)
- A pair of 32-bit FIFO-like queue direct interfaces (QIF32)
- 3-way 64-bit VLIW (VLIW3)

In addition to these options, Tensilica has developed several pre-configured major functional blocks that are licensed in addition to Xtensa LX4:

HiFi 2 Audio engine—the industry's most popular 24-bit audio subsystem with a library of over 80 audio, voice and sound enhancement software packages.

HiFi EP Audio DSP—a superset of the HiFi 2 Audio DSP with advanced optimizations for DTS Master Audio, improved voice pre- & post-processing, and improved cache memory subsystem.

ConnX BBE16 DSP—for use in LTE baseband processors in cellular radios and multi-standard broadcast receivers.

ConnX BBE64 DSP—the most powerful DSP we offer with up to 128 MACs for use in LTE Advanced baseband processors.

ConnX D2 DSP engine—for 16-bit communications DSP functions. The ConnX D2 DSP engine delivers outstanding performance from 'C' code.

ConnX Vectra DSP engine—for higher performance 16-bit communications DSP functions. The ConnX Vectra DSP engine uses 64-bit instruction words containing three issue slots for ALU, multiply-accumulate, and load/store operations.

ConnX Vectra VMB—for baseband acceleration. Vectra VMB (Viterbi 8x20-bit Multiply and Bit unpacking) includes instructions targeted at FIR, IIR and filtering operations. These instructions also include bit stream unpacking and Viterbi trellis decode operations. Available only when the ConnX Vectra DSP engine is also selected.



Extensibility Unlocks the True Power of Xtensa LX4

Most embedded processors offer fixed hardware functionality with options for memory size, cache size, and bus interface. Performance is proportional to the clock speed. Beyond that, application code optimization effort or a move to the next processor in the roadmap is required. Tensilica offers something different—the opportunity to optimize the processor itself using Tensilica’s TIE language.

Tensilica’s TIE language is used to describe new instructions, registers and execution units that are then automatically added to the Xtensa LX4 processor. TIE is a Verilog-like language used to describe desired instruction mnemonics, operands, encoding, and execution semantics. The TIE files are inputs to the Xtensa Processor Generator. The Generator automatically builds the processor and the complete software tool chain that incorporates all configuration options and new TIE instructions. The base instruction set remains for maximum compatibility with third party development tools and operating systems.

The TIE language unlocks the true power of Xtensa LX4. It allows designers to get orders of magnitude performance increases for their applications and create differentiation.

Flexibility to Add Just What You Need

Just as the designer can choose from a set of predefined functional options to improve processor performance, the designer can now create instructions that can speed up standard or proprietary algorithms. Using the tools provided, application hot spots can be identified and additional logic created to process

these hot spots more efficiently, without the need to increase the clock frequency or re-write the software.

Differentiate—Make a Processor That’s Uniquely Your Own

When processors have fixed hardware functionality and your competitors are using the same or similar processors, then differentiation is often limited to the algorithm implementation itself. Fixed processors are good at general-purpose computing, but not so good at any specific algorithm. Tensilica gives you the opportunity to differentiate at the hardware level and implement algorithms more efficiently by designing hardware that will accelerate your particular algorithm. This means that your design will be almost impossible to copy, as only your hardware will reach the performance required on the same software implementation.

Xtensa LX4 Designer-Defined Functional Units and Interfaces

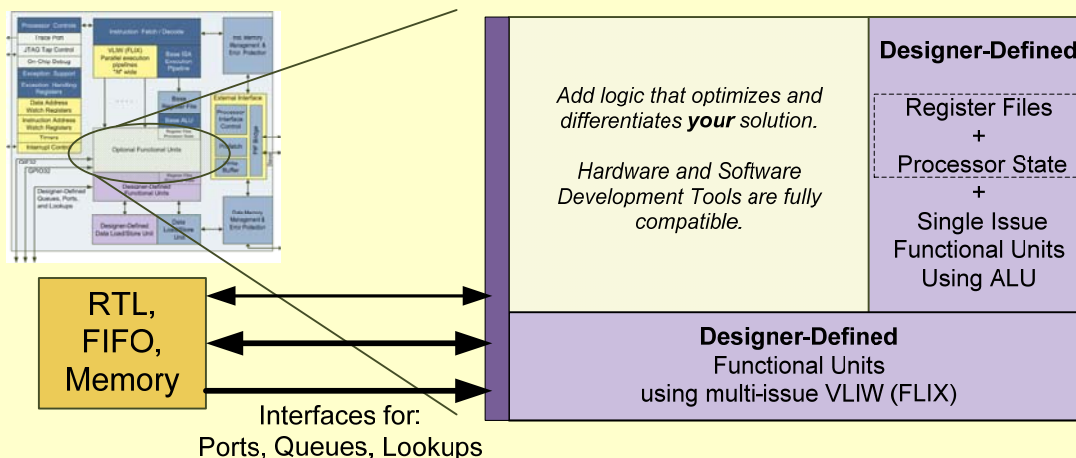


Figure 3. Xtensa LX4 offers a proven method of adding designer-defined functional units and interfaces to the Xtensa LX4 DPU



FLIX for Parallel Execution (VLIW)

Many of the major pre-configured functional blocks take advantage of Tensilica's FLIX capabilities.

The FLIX architecture makes the Xtensa LX4 into a VLIW processor that executes 2 to 30 parallel execution units when needed. Wide 32/64/128-bit FLIX instruction formats are seamlessly intermixed with the base Xtensa 16/24-bit instructions so there is no mode switch penalty when using FLIX.

With FLIX, the Xtensa LX4 processor can deliver the ultra-high performance characteristics of a specialty ultra-wide instruction word processor without the negative code size implications typically found in such VLIW or ULIW processors. In fact, Xtensa LX4 processors with FLIX can often deliver higher performance and smaller code size at the same time. This performance comes with very little overhead—adding only 2,000 gates to the size of

the processor for instruction decode and control.

The Xtensa C/C++ Compiler (XCC) automatically extracts parallelism from source code and bundles multiple operations into FLIX instructions. In this way, a 3-issue Xtensa LX4 processor running at 300 MHz can deliver performance up to the equivalent of a 900 MHz processor. Additionally, the compiler can bundle the branch and load/store instructions in parallel with compute instructions to gain a performance boost over straight-line code.

Designers can go beyond the capabilities in the major pre-configured functional blocks and use the FLIX capabilities in their own design by using Tensilica's TIE language to specify exactly what's needed.

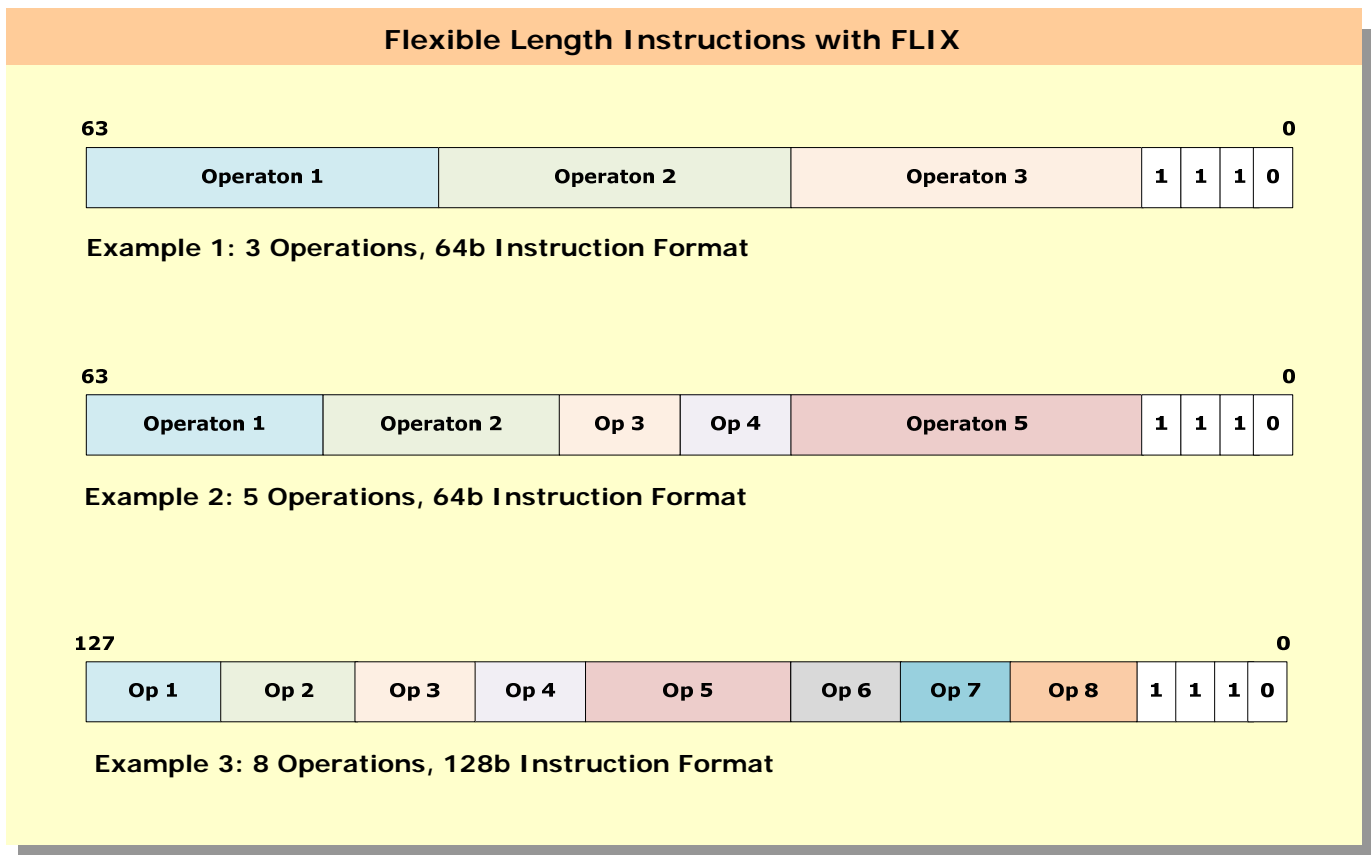


Figure 4. Designers can use FLIX to create VLIW instructions up to 128 bits wide to execute 2 to 30 parallel execution

Designer-Defined I/O Bypasses the System Bus for Maximum Data Throughput

The Xtensa LX4 processor brings another fundamental breakthrough in embedded processor designs—the ability to define direct data interfaces into and out of the processor for maximum data throughput. This ability is a key reason that Tensilica's Xtensa LX4 is ideal for the SOC dataplane.

Tensilica provides three direct interface capabilities:

TIE Ports: These provide direct connection to other logic within an SOC or to other Xtensa processors. These are created with simple one-line declarations in a TIE file.

TIE Queues: These function like FIFO interfaces. TIE input Queues function with a familiar pop/empty/data interface to external logic while TIE output Queues present a similar push/full/data interface. All interactions with the Xtensa LX4 processor pipeline are automatically implemented by the Xtensa Processor Generator.

TIE Lookups enable designers to connect RAMs or external devices to Xtensa LX4 processors. These external memories or

devices can be accessed directly from the processor's datapath without using load/store instructions. These interfaces are useful for connecting table lookup RAMs, for example in networking applications, or for connecting long latency hardware computation units.

Port connections can be up to 1024 wires wide, allowing wide data types to be transferred easily without the need for multiple load/store operations. As many as one million signals (1000 1024-bit-wide ports) can be used. While this is an outrageous number, far exceeding the performance demands of real systems today, this clearly demonstrates that the conventional I/O bottlenecks inherent in a processor-based solution do not apply to Xtensa processors.

While Ports are ideal to quickly convey control and status information, Queues provide a high-speed mechanism to transfer streaming data with buffering. Input queues and output queues operate to the programmer's viewpoint like traditional processor registers - without the bandwidth limitations of local and system memory accesses.

TIE Port & Queue Wizard

The Xplorer IDE provides a wizard for quickly generating Ports and Queues without the need to write any TIE.

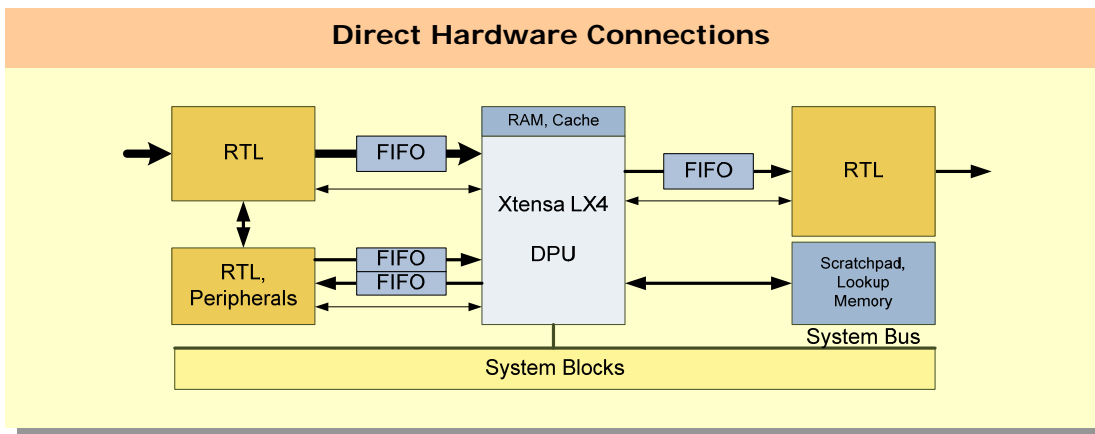


Figure 5. Example of direct FIFO and Port connections using TIE Queues and TIE Ports

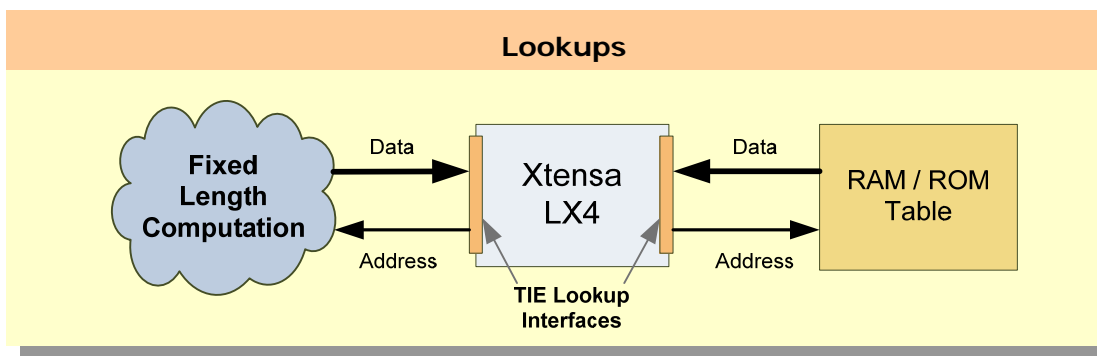


Figure 6. Example of TIE Lookups showing connections to memory and logic





Xtensa LX4 as an RTL Companion

RTL verification has become the most resource and time consuming aspect of SOC design. The Xtensa LX4 processor offers unique advantages as an addition to your own hardwired RTL blocks in complex SOCs. The Xtensa LX4 can connect directly to your RTL with dedicated high bandwidth data and control interfaces.

Bandwidth of Hard-Wired Logic and Performance without Hand-Coded State Machines

The Xtensa LX4 processor can achieve virtually the same levels of inter-block I/O bandwidth and intra-block computational parallelism as hard-wired logic designed with traditional RTL design methodologies. How? By using a combination of TIE Ports and Queues, parallel FLIX execution units, and some TIE instructions.

Unlike RTL-based designs, Xtensa LX4 processors are pre-verified, and do not require hard-wired implementation of complex state machines. Instead of state machines, the complex data-paths added to Xtensa LX4 cores are sequenced and controlled by the processor's instruction stream. That means the "control logic" is fully programmable and can be debugged using software development methodologies – reducing verification time and risk for the entire SOC.

Lower Verification Effort and Time

Designing hardwired RTL blocks has become more about verification than about design. Design teams typically spend twice the number of resources and person months on verification than on design. Design changes made late in the project cycle are often limited by the verification effort.

Typically, 90% of the RTL block's area lies in the datapath and only 10% in the control logic, yet most (perhaps 90%) of the bugs are found in the control logic. The ability to extend the Xtensa LX4 processor using TIE enables designers to create datapaths inside the Xtensa processor without the need to generate and verify the associated control logic. Instead the control logic is expressed in software as instructions that execute on the processor.

It is easier to verify TIE extensions made to the Xtensa LX4 processor than it is to verify an equivalent RTL data path, since only the input-output relationship and functional behavior of the operations specified in TIE have to be verified. The TIE Compiler and Xtensa Processor Generator take care of converting the TIE specification into data path elements in the processor pipeline and implementing the control, decode, and bypass logic in the processor control units.

Reuse of the Same Hardware for Multiple Tasks

Complex SOCs consist of millions of gates of logic and are designed to perform multiple tasks. Often these multiple tasks do not need to be performed at the same time. This provides an opportunity for multiple tasks to share the same hardware units. Processors are particularly amenable to enabling this type of sharing.

Designers can specify a datapath in TIE that consists of a set of execution units that can be used by multiple tasks and then use the programmability of the processor to determine which tasks are executed. For example, an audio engine can be designed to implement a range of audio codecs, such as MP3, AC-3, WMA, etc.

Flexibility to Fix and Upgrade Algorithms Post Silicon

Using an Xtensa LX4 processor to implement an algorithm lets the designer fix, enhance and tweak the algorithm close to and after the SOC has taped out. In particular, post-silicon bugs now have a chance of being worked around.

Algorithms that are a subject for continuous research, such as half-toning in printers and image and video post processing, are ideal candidates for implementation in an Xtensa LX4 processor.

When new and updated standards emerge, such as for audio and video processing, they will likely run on your existing hardware because it is programmable.

Co-Simulation at the RTL Pin Level

Connect directly to your RTL wires using pin-level XTSC SystemC model interfaces without the need to purchase additional EDA vendor tools. This enhancement to transaction-level XTSC models allows designers to interchange SystemC and RTL blocks for co-simulation. This works with all of the major EDA vendor simulation tools.



Extending the Life of an Existing RTL Design

You can easily add functionality to an existing design, or upgrade parts of it for the latest standard, with limited development effort by using Xtensa LX4 DPUs.

A Conventional Processor SOC with RTL

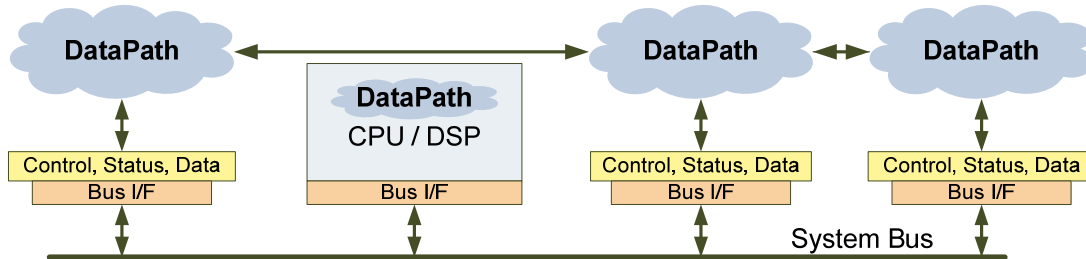


Figure 7. With any other 32-bit processor core, all communication is through the system bus, which must have the available data bandwidth and must keep bus latency manageable.

Add New Functionality with Xtensa LX4 DPUs

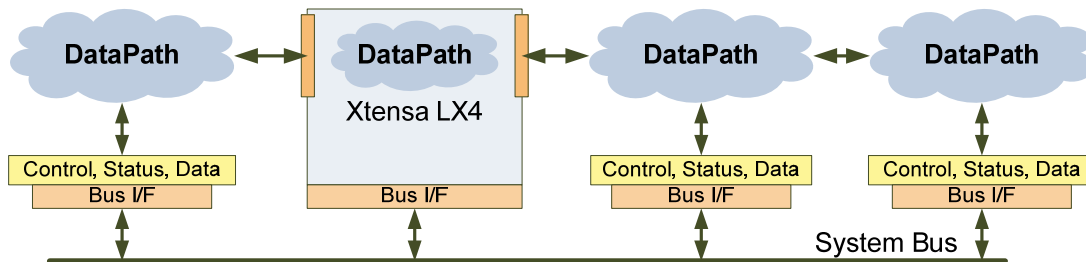


Figure 8. With Xtensa LX4, data can be kept off the system bus by using direct connectivity to RTL through Ports and Queues. These provide almost unlimited bandwidth with precise latencies.

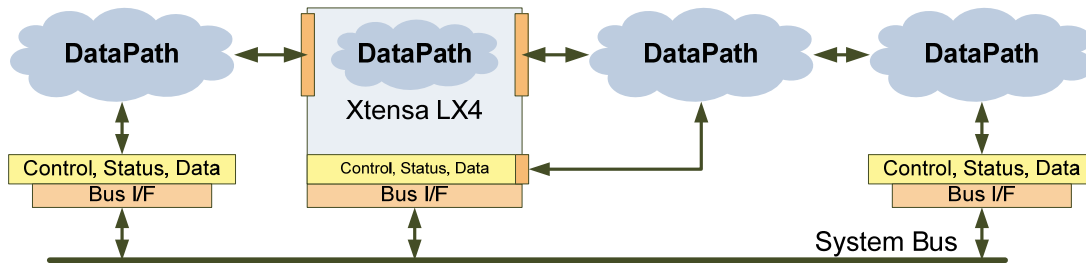


Figure 9. When extending the functionality of existing RTL blocks, the control logic parts can be brought into the Xtensa LX4 to make the FSM easier to debug and verify.

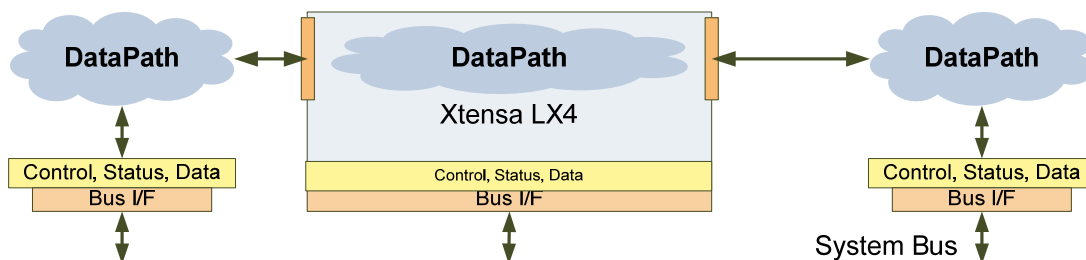


Figure 10. When larger changes to the datapath are needed, then the whole RTL function can be brought into the processor.

Rapid Design Development, Simulation, Debug and Profiling

The Xtensa Xplorer™ integrated design environment (IDE) serves as the graphical user interface (GUI) for the entire design experience. From Xtensa Xplorer, designers with existing application software can profile their application, identify hot spots, decide on configuration options, add new instructions and execution units to optimize performance, and then generate a new processor—all within a matter of hours. No other IP provider puts such flexibility directly into the hands of the designer with a tool that integrates software development, processor optimization, and multiple processor SOC architecture in one IDE.

Hardware designers now have new options for implementing

algorithms. Interfaces can be added to the processor to offer direct, deterministic connectivity to SOC logic. With the customizable Port and Queue interfaces, designers can stream data into or out of the processor. This direct connectivity with the rest of the SOC offers great control and predictable bandwidth. The simple C programs needed to control the Xtensa processor can be written and debugged within the Xtensa Xplorer IDE.

The Xtensa Processor Generator creates a complete hardware design with matching software tools, including a mature, world-class compiler, a cycle-accurate SystemC-compatible instruction set simulator (ISS) and the full industry standard GNU Toolchain.

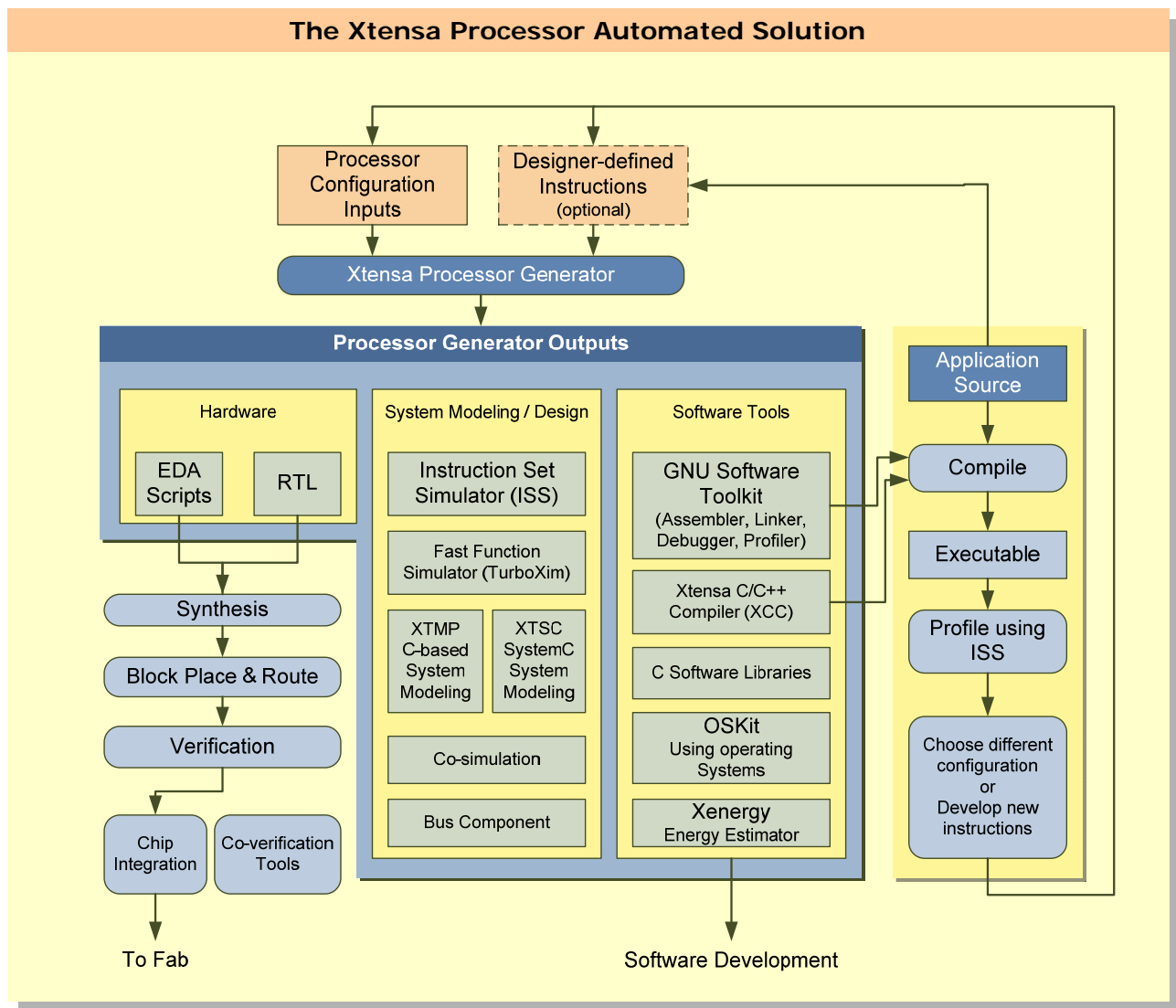


Figure 5. Tensilica's proven methodology automates the creation of customized processors and matching software tools

Hardware Development

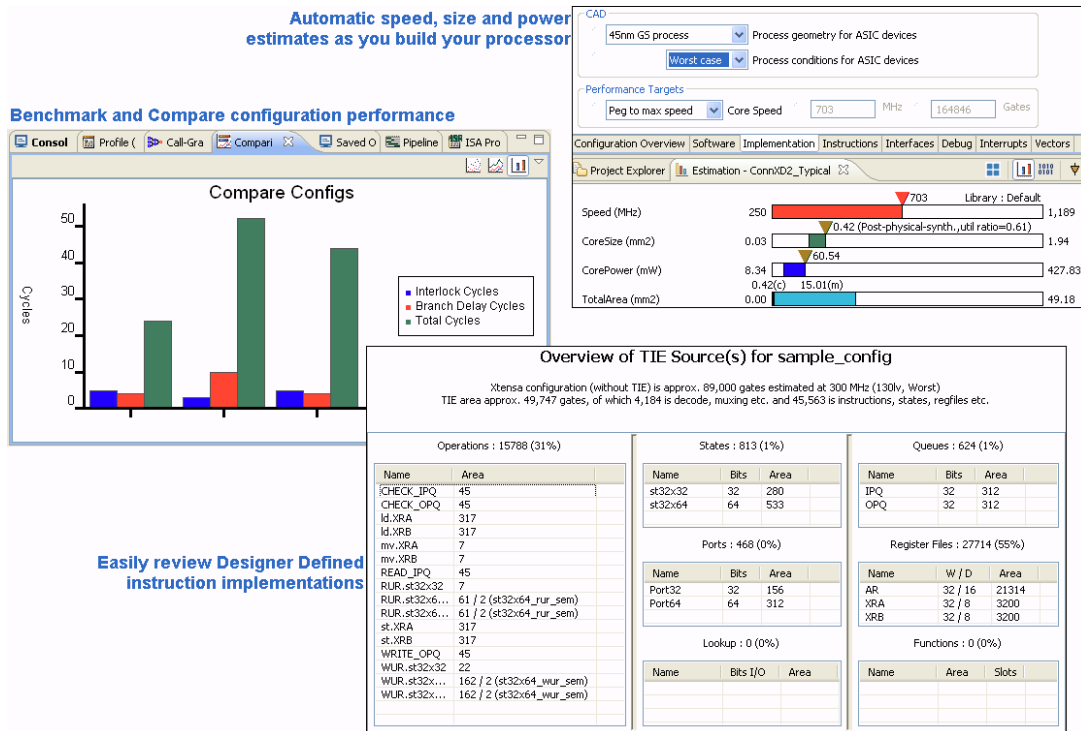


Figure 6. Xtensa Xplorer can display valuable information including performance comparisons, instruction sizes, and processor size, area and power

For Hardware Development

Hardware designers can profile, compare and save many different processor configurations. Use the ISS to simulate a single processor or, for multiple processor subsystems, choose Tensilica's XTensa Modeling Protocol (XTMP) or Tensilica's XTensa SystemC (XTSC) modeling tools.

Xtensa Xplorer serves as the gateway to the Xtensa Processor Generator. Once a processor configuration is finalized, the Xtensa Processor Generator creates the automatically verified Xtensa processor to match all of the configuration options and extensions you have defined, in about an hour. The full software tool chain is also created that matches all processor modifications made. See the Processor Developer's Toolkit product brief for more information.

Complete Hardware Implementation and Verification Flow Support

- Automatic generation of RTL and tailored EDA scripts for leading edge process technologies, including physical synthesis and 3D extraction tools
- Auto-insertion of fine-grained clock gating delivers ultra-low power

- Hardware emulation support including automated FPGA netlist generation
- Comprehensive diagnostic test bench
- Format verification support for designer-defined functions
- Pipeline-modeling, cycle-by-cycle accurate Xtensa instruction set simulator (ISS)
- System modeling capabilities with optional XTMP and XTSC simulation environments
- Multiple-processor on-chip debug capable with break-in/out control
- Hardware co-simulation in System C with Tensilica's pin-level XTSC connectivity to RTL
- XTSC transaction-level modeling support, including out-of-the-box multi-core co-simulation
- Xenergy™ energy estimation tool to optimize hardware and software for power



Software Development

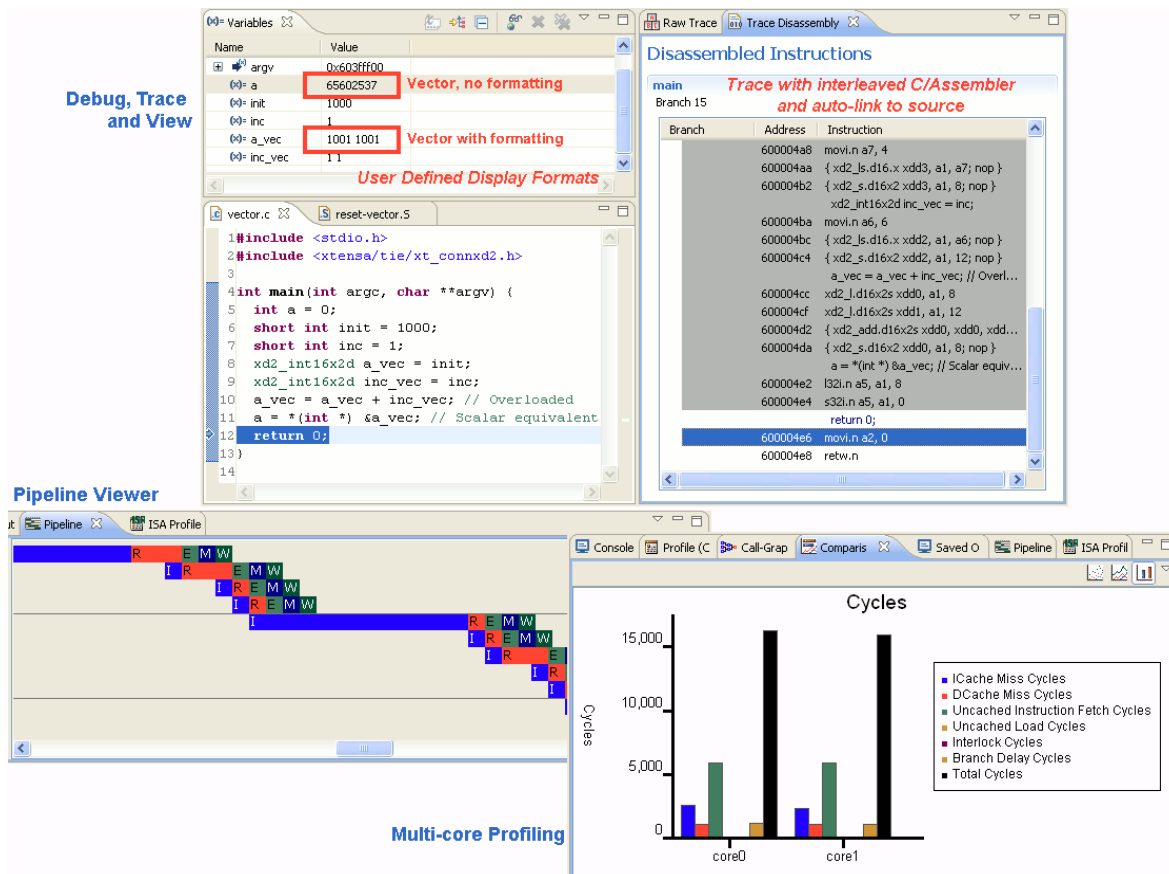


Figure 7. Xtensa Xplorer shows debug/trace, profiling of pipeline utilization, and a cycle comparison for a multiple core simulation

For Software Development

The Xtensa Software Developer's Toolkit (SDK) provides a comprehensive collection of code generation and analysis tools that speed the software application development process. Tensilica's Eclipse-based Xtensa Xplorer GUI serves as the cockpit for the entire development experience and also provides powerful visualization tools to aid application optimization.

The entire Xtensa software development tool chain, along with simulation models, RTOS ports, optimized C-libraries, etc., are automatically generated by the Xtensa Processor Generator. This also ensures that all the software tools – such as the compiler, linker, assembler, debugger, and instruction set simulator – always match and are tuned exactly to any custom processor hardware.

Complete Software Development Tools

- Pipeline-modeled, cycle-accurate ISS
- High-speed (40-80x) instruction accurate TurboXim™ simulator speeds software development
- XTMP and XTSC for multiple processor simulation and modeling
- Debug offers full GUI and command line support for single and multiple processor designs
- Profiling views processor pipeline utilization as well as time in functions across multiple processors. Allows "what if" comparisons
- Xenergy energy estimation tool helps the designer tune the software for energy consumption
- Vectorization Assistant discovers and locates code that could not be vectorized along with an explanation that can help the programmer modify the code so that it can be vectorized.
- Support for major operating systems including Mentor Graphics' Nucleus Plus, Express Logic's ThreadX, Micrium's μC/OS-II, Sophia Systems' μITRON, and open-source Linux.
- Mature, highly optimizing C/C++ compiler (XCC) that rivals hand-coded assembly applications on other processors
- GNU-based Assembler and Linker





Ideal for Applications Where Low Power is Critical

Power often is the key issue in an SOC design. Tensilica employs many techniques to reduce power consumption, both built in to the base hardware and into the configuration options, allowing more control over system and memory resources. Tensilica processors consistently consume less power than other licensable embedded CPUs at equivalent gate counts.

Tensilica automates the insertion of fine-grained clock gating for every functional element, including those defined by the designer. This automation gives the Xtensa LX4 DPU a significant advantage over RTL design where manual, error-prone post-layout tuning of clock circuits is often required.

Accessing local memories is one of the highest power consuming activities. Tensilica has designed the Xtensa LX4 processor to eliminate any unnecessary local memory interface activation if that memory is not directly addressed by the processor.

Tensilica automates the implementation of these energy saving techniques in the Xtensa Processor Generator.

The designer can configure the external data bus width and internal local memory data widths independently. This allows system-level power optimizations depending on whether the processor is constrained by external or internal instruction and data access.

Designers use Tensilica's Xenergy™ energy estimation tool to evaluate energy-related tradeoffs in the design process. The Xenergy tool can be used to optimize TIE hardware instructions and to fine tune the software application for the lowest energy usage.

Specifications

Configuration	Post-Route Area (mm ²)	Clock Rate (MHz)	Power Dissipation (mW/MHz)
Smallest* —Synopsys library, TSMC 40LP, low-power flow	0.024	60	0.012
Smallest* —Synopsys library TSMC 40LP, high-speed flow	0.044	670	0.018
Smallest* —Synopsys library, TSMC 45GS, low-power flow	0.024	62	0.009
Smallest* —Synopsys library, TSMC 45GS, high-speed flow	0.044	1032	0.014
570T** —Synopsys library, TSMC 40LP, low-power flow	0.163	57	0.046
570T** —Synopsys library TSMC 40LP, high-speed flow	0.295	493	0.093
570T** —Synopsys library TSMC 45GS, low-power flow	0.158	58	0.034
570T** —Synopsys library TSMC 45GS, high-speed flow	0.283	780	0.066

*Smallest—smallest configuration used by customers with only local instruction and data RAM interfaces and full clock gating.

**570T—similar to Tensilica's Diamond Standard 570T.

Tensilica, Inc.

3255-6 Scott Blvd., Santa Clara, CA 95054-3013, USA
Tel: 1-408-986-8000 Fax: 408-986-8919 Website: www.tensilica.com

©August 2011, Tensilica and Xtensa are registered trademarks of Tensilica, Inc.
The Tensilica logo, Xenergy, Xplorer and TurboXim are trademarks of Tensilica, Inc. All other trademarks are the property of their respective owners.

