Selecting Your Processor or DSP IP—A Checklist

When looking for a processor or DSP, designers must make careful choices and evaluate tradeoffs to find the most optimal solution for their designs. This white paper provides a list of questions that will help you make the best processor and DSP IP choices so you get exactly what your team needs to develop successful SoC designs on time, within budget, and with minimal hassle.

Contents

Introduction

While SoC designs can consist of many IP blocks, some of the most complex are processors and DSPs. With their associated software-development tools, simulation models, and EDA flow scripts, these processor IP blocks can influence the entire SoC design. Designers must carefully evaluate the options to find the DSP or processor best suited for their individual needs. This white paper provides a list of questions to ask yourself, your team, and processor and DSP IP providers during your selection process.

In addition to evaluating general-purpose processors and DSPs, consider using customizable processors, such as the Cadence® Tensilica® Xtensa® processors that can scale from tiny microcontrollers to DSPs. You can optimize the Xtensa processor to run your application more efficiently. All development tools are created automatically to handle all your optimizations. Your new processor is correct-by-construction.

Processor or DSP IP Performance Checklist

Processor and DSP IP have a wide performance range. Intuitively, there is a big performance difference between a simple 8-bit controller and a large 32-bit RISC processor, or between a simple, general-purpose, 16-bit DSP and an advanced DSP with special hardware execution units designed for specific applications such as audio and video. Before you can select a processor or DSP for a specific task or set of tasks on your SoC, you must list all the tasks you want it to handle so that you have an idea of the performance you need from each.

Many experienced SoC designers have an outdated notion of what processors and DSPs can and cannot do. These assumptions are based on years of experience with conventional fixed instruction-set architecture (ISA) processors and DSPs. Almost universally, their performance expectations are too low with respect to customizable processor IP because the performance of customizable processors and DSPs can be increased almost without limit for specific on-chip tasks while increasing energy and area efficiency. If you are not familiar with the greatly expanded performance capability of customizable processor IP, ask a vendor for a quick seminar. You will want to take full advantage of the processor so you do not underutilize your silicon.

Which processor tasks are routine and not performance limited, and which will stretch the limits of available technology?

Many on-chip housekeeping tasks and other simple tasks are routinely handled in firmware on a general-purpose processor that acts as the SoC's CPU. The heavy lifting—the big data-processing tasks—is often handled by dedicated hardware because of the substantial processing load. Audio, video, imaging, and network-packet processing are all examples of dataplane processing, where the heavy lifting takes place on every modern SoC. While designers assume that processors and DSPs cannot handle these heavier processing tasks on advanced SoCs—that only dedicated hardware can manage the problem—the limits are changing.. Let your processor IP vendor help you understand what the real limits are today. Processors that can do the heavy lifting in the dataplane can give you greater system performance with reduced overall power and energy consumption.

How much I/O bandwidth do you need?

Processor throughput is usually limited by its bus characteristics. A bus is a fixed-width, unidirectional resource, so it is a limited resource with a maximum bandwidth determined by clock frequency. You cannot increase the I/O bandwidth of a fixed-ISA processor without increasing its clock frequency (along with its power and energy consumption). DSPs have similar limitations.

However, not all processors have these limits. You can increase the I/O bandwidth of the processor by two or three orders of magnitude by adding more firmware-controlled ports (GPIOs), FIFO queues, or memory interfaces to the processor, avoiding the bus in favor of direct, wide connections similar to and compatible with the types of connections used in RTL blocks. This can revolutionize your thinking about where and when to use processors, as the I/O can be as flexible as an RTL block.

What happens when you exceed a processor's capacity?

When you select a processor or DSP, you marry into the processor's entire family including the hardware architecture, the software-development tools, the simulation models, and the EDA scripts that ease the processor through your EDA flow. What happens when the processing burden on a selected processor exceeds its processing abilities? Can you increase the processor's operating clock frequency? If you do, will that choice force you to use a more advanced IC process node to reach the higher required operating frequency? Will it cause the SoC to consume more operating power and energy? Will the increased power dissipation require a more expensive IC package? A heat spreader? A heat sink? A cooling fan? Liquid cooling? All of these consequences trigger chip-, board-, and system-level cost increases.

There are alternatives to boosting clock frequency. Some fixed-ISA processors and DSPs are members of code-compatible families with more powerful family members that you can use to increase performance without increasing clock rate. The Xtensa customizable processors are members of an infinite family—you can incrementally add new, more efficient instructions, add registers and register files, change the width of part of the processing pipeline to more efficiently handle native data types, and add direct Wide I/O ports to boost bandwidth. A few simple processor optimizations can often deliver all the added performance and energy efficiency needed while still retaining the same base instruction set, development tools, and design flow.

Software-Development Tools Checklist

What software-development tools does your team need?

At a minimum, a software-development team needs a compiler, assembler, debugger, instruction-set simulator, and profiler—closely matched to the processor—to develop robust code within today's tight product schedules.

For example, compilers must deeply understand the underlying architecture of the specific processor you are using to perform efficient code optimizations. Note that debuggers and instruction set simulators for fixed-ISA processors perform operations defined for the processor, but cannot understand the operations of any custom hardware coprocessors your team adds.

Compilers and assemblers are not all equally adept at code optimization. Some perform far more optimizations than others, including:

- Constant folding and propagation
- Common sub-expression elimination
- Peephole optimizations
- Local register allocation
- Inlining
- Dead-code elimination
- Partial redundancy elimination
- Strength reduction
- Global register allocation
- Instruction scheduling
- Loop unrolling
- Heuristic-based inlining of static functions
- Loop transformations based on dependence analysis
- Vectorization
- Inter-procedural analysis and optimization

Many of these optimizations can only be performed for processors with certain hardware features—for example, SIMD execution units in the case of vectorization.

When you evaluate software development tools, ensure they have an intimate understanding of all your hardware, including any underlying processors, coprocessors, and RTL accelerators, to achieve superior optimization.

What operating system does your processor need to run? Do you even need an operating system?

Nearly all processors used as on-chip CPUs run an operating system to help them manage the various house-keeping tasks assigned to them. There are many popular operating systems including various flavors of Linux and several traditional real-time operating systems such as Express Logic's ThreadX, Mentor Graphics' Nucleus, FreeRTOS, Micrium's µC/OS, and many more.

However, many deeply embedded processors and DSPs do not run complete operating systems. Instead, they may run OS kernels or no OS at all in favor of a simple custom scheduler.

Your team will probably not need more than one operating system, so you do not need a processor that will run as many operating systems as possible. You need a processor that will run the operating systems that your team needs now and can imagine needing in the future.

Technical Support and Training Checklist

What forms of technical support are offered and when is technical help available?

Ideally, you want speakers of your native language who are experts in SoC design and are available 24/7/52. Are you willing to pay for that level of support? Development teams on a limited budget will need other forms of more affordable support better suited to their needs and budget, such as email support with guaranteed response times, knowledge-base access, and FAQs.

What kind of training is available?

Processor and DSP IP is expensive, so it is a good idea to get the right team members trained on the architecture and tools you choose so that you can get your money's worth. Therefore, you should understand the training alternatives available (books, live classes, online training) while you are evaluating processors and their associated development tools.

Consider the Xtensa Processor

Whether you focus on one product or create thousands, requirements change over time. Those changes often include reducing cost, increasing performance, and/or increasing energy efficiency. Using Xtensa processors, you can meet these new requirements with less cost and risk than other approaches.

Scalability

The Xtensa processor offers one processor architecture and one set of tools that scales from tiny microcontrollers and digital signal controllers to high-performance real-time controllers and DSPs.

Using a higher performance or more functional processor does not require learning a new set of software development tools or a new hardware flow, because all Xtensa processors use the same flow.

Customize and innovate

Xtensa processors are designed to be optimized so you can make the processor run your application more efficiently. Choose to add instructions to the base ISA from pre-defined options or create your own using the Tensilica Instruction Extension (TIE) language. You can add elements to include deep data pipelines, parallel execution units, task-specific state registers, wide data buses to local and global memories, and direct connection I/Os. You get this customization with an automated flow that requires no extra processor verification and keeps the development tools updated with every change. Our automated process guarantees your new processor is correct-by-construction.

Keep programmability in your designs even when a conventional fixed-ISA processor needs to be offloaded because the instruction set or I/O is unsuitable or I/O limited. Using Xtensa processors, you can build the "offload" with more application-specific instructions and I/Os in a fraction of the time that it would take to design and verify in RTL. You can also avoid any interfacing delays that can be incurred when communicating with this functionality in an external logic block—new instructions can use the general and/or custom registers sized to the data types involved for instant availability.

Optimizing the processor with new instructions and I/Os creates a unique design, making your particular application code run more efficiently. And with the benefit of additional security: only reveal them to your own programmers or your customer's programmers as needed.

All the development tools are created automatically to handle anything that is added. Full C/C++ support is provided with all the debug and profiling that you expect (and need) for professional development with our familiar Eclipse-based IDE.

Additional Information

To get more information on the unique abilities and features of Cadence Tensilica Xtensa processors, see ip.cadence.com.

cādence®

Cadence Design Systems enables global electronic design innovation and plays an essential role in the creation of today's electronics. Customers use Cadence software, hardware, IP, and expertise to design and verify today's mobile, cloud and connectivity applications. www.cadence.com