

Processor Ports and Queues:

Easily overcome I/O-bandwidth obstacles in your next ASIC or SOC design

Computational horsepower is always a concern for ASIC and SOC designers. I/O bandwidth is yet another. Configurable processor cores allow ASIC and SOC designers to add internal registers and function units that boost computational throughput, often by a factor of 10 to 100. However, increased computational performance places even more demand on the processor's I/O speed, to bring operands into the core and to ship results out. Conventional bus-centric design for on-chip I/O simply cannot handle the resulting increased traffic. Direct port I/O and FIFO queue interfaces can quickly ease the traffic load on overused buses, which greatly simplifies the design of complex chips.

Most processors, configurable or not, rely on one or a few buses to move data into and out of the processor core. These buses are increasingly inadequate for high-throughput applications such as video compression/decompression or high-speed networking. A new, configurable feature called “ports and queues,” based on tried-and-true system technology, provides the ASIC and SOC design team with as much bandwidth as any system can possibly use, essentially unlimited I/O bandwidth.

Figure 1 portrays a simple processor-based system. This single-bus approach to processor-based design dates back to the introduction of the first microprocessor in 1971. It hasn't changed much after more than 30 years despite radical improvement in processor performance. Simply said, the processor loads incoming data from input devices over the bus and sends results to output devices over the same bus.

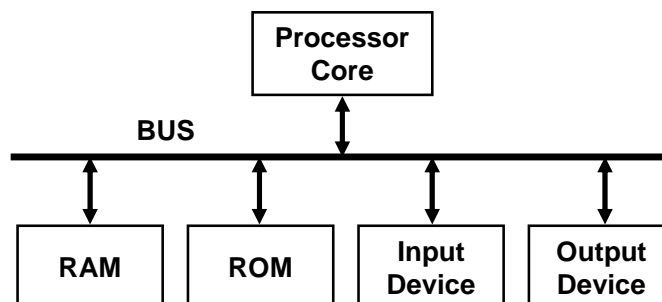


Figure 1: A Simple processor-based system

For discrete integrated circuits and printed circuit boards, the adoption of bus-based I/O saved package pins and circuit-board traces. However, for ASIC and SOC designs, the bus-centric design approach leaves a lot of IC technology's fundamental performance untouched. The reason is simple. In Figure 1, the two memory devices and the two peripherals share the processor's bus. Each of these four devices receives some fraction of the processor's bus bandwidth and the system's overall throughput suffers. As the number of blocks connected to the bus increases, the I/O congestion increases as well.

To counter this problem, most contemporary embedded processor cores have separate local-memory buses to remove instruction and local data traffic from the bus. The resulting systems look something like the one shown in Figure 2.

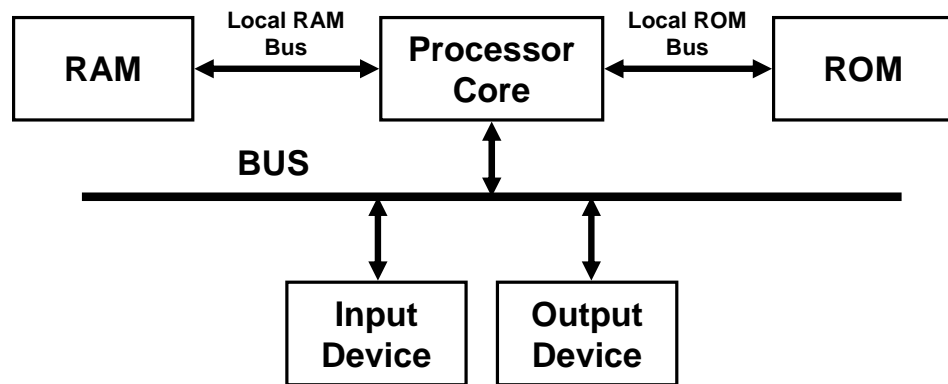


Figure 2: Processor-based system with local memory buses

Figure 2 shows a processor core with three buses: one for local data RAM, one for local instruction ROM, and one for global system memory and I/O devices. (Note: Figure 2 does not show local cache memories or their private cache buses, but most RISC processor cores have such local caches to improve overall performance.) This subsystem design clearly has more potential I/O throughput than the one shown in Figure 1 because the processor has three buses instead of one. All buses can potentially be active simultaneously, so the potential I/O bandwidth for the system shown in Figure 2 could be three times that of the simple system shown in Figure 1. The instruction data stream and the storage and retrieval of local data don't subtract from the bus bandwidth available to peripheral I/O devices, which use a different processor bus. However, the input and output devices shown in Figure 2 must still share a bus, which can limit performance in applications with truly high performance requirements where both the input and output devices have the ability to consume all of the available bus bandwidth. The problem becomes even worse if there are many I/O devices, as illustrated in Figure 3.

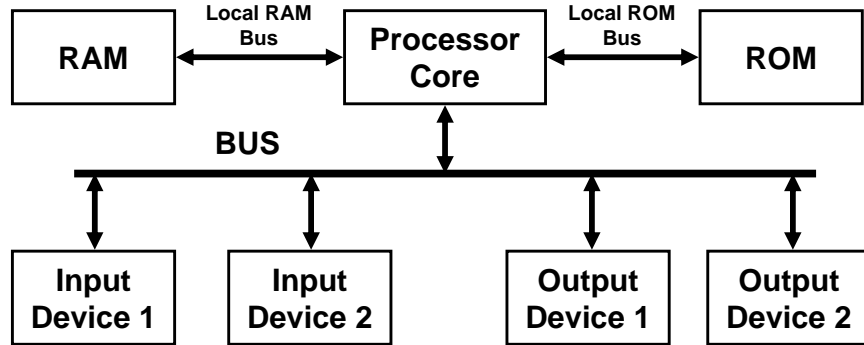


Figure 3: Processor-based system with multiple I/O devices

At this point, SOC designers that require even more I/O throughput will usually abandon the use of a processor and develop a custom RTL accelerator block to perform the required processing. A flow-through design like the one shown in Figure 4 boosts I/O performance as well.

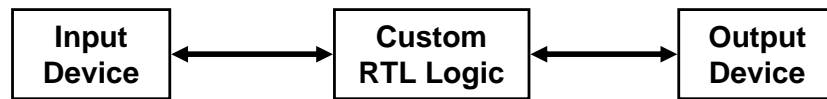


Figure 4: Solving the I/O bandwidth problem with custom RTL logic.

The system design depicted in Figure 4 provides the highest possible I/O bandwidth by giving the input and output devices separate connections to the custom RTL accelerator block. As long as the custom-designed logic block processes the incoming data at the full input rate, the system operates at its theoretical limit.

It's relatively easy to add an extra bus to a configurable processor to enhance the system shown in Figure 2 by providing separate buses for the I/O devices. Such a system design might look like the one shown in Figure 5. The system design shown in Figure 5 allows a processor-based design to deliver as much throughput as the system based on custom RTL shown in Figure 4.

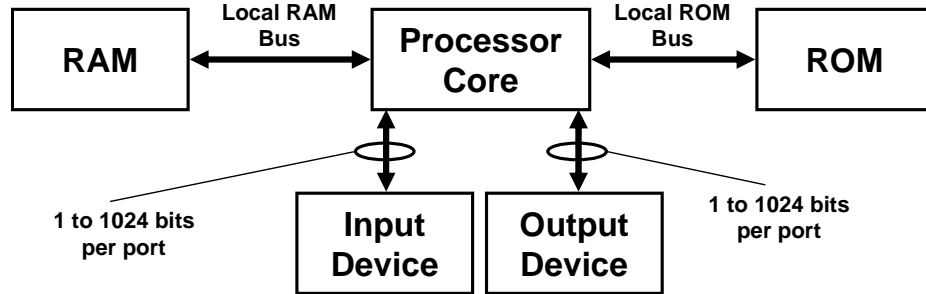


Figure 5: Processor-based system with separate input and output device buses

For example, the ports and queues features of Tensilica’s Xtensa processors allows ASIC and SOC designers to add multiple I/O buses to a processor core as shown in Figure 5. Designers can add as many as 1024 ports to the processor core and each port can have as many as 1024 pins, which boosts I/O bandwidth more than 1000x relative to a few conventional 32- or 64-bit processor buses. In addition, FLIX technology, another feature introduced with the Xtensa LX processor core, allows designers to add separate, parallel execution units to handle multiple simultaneous computational tasks. Using both of these features, the resulting configured processor might look something like the block diagram shown in Figure 6, which uses six ports to connect three pairs of input/output devices to three custom execution units in the processor.

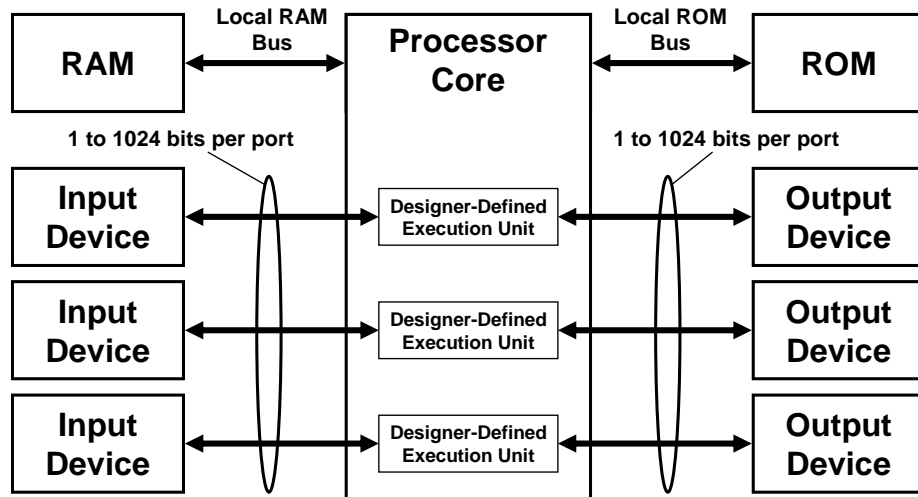


Figure 6: Processor core with multiple I/O ports and execution units

Note that the same sort of ports can be used to attach an existing block of custom RTL accelerator logic to a processor without reducing the I/O bandwidth available to other devices. Figure 7 shows how such an accelerator might be connected to a configurable processor.

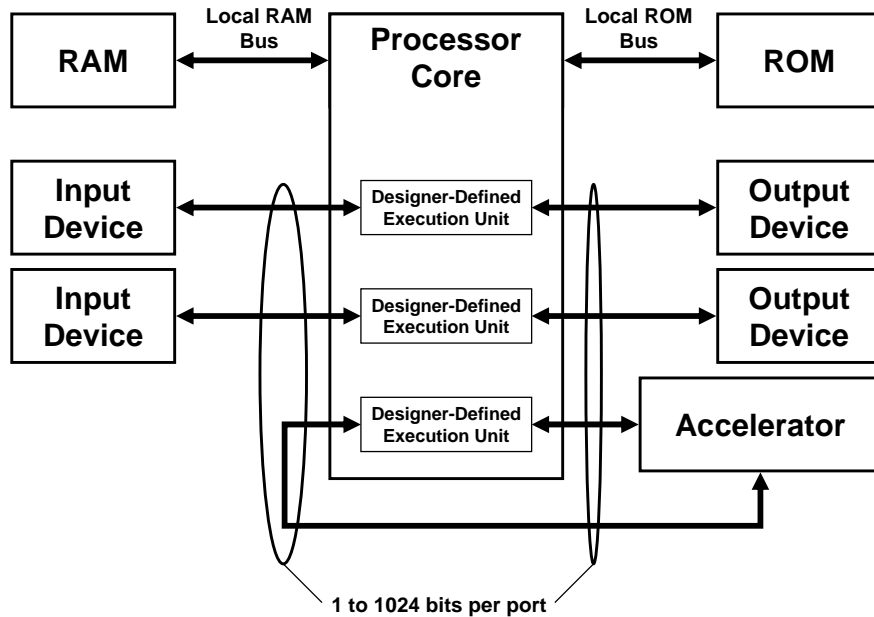


Figure 7: Using TIE ports to attach an existing acceleration unit to a processor core

So why do this? Why not just design a new accelerator block using RTL? After all, that's what ASIC and SOC designers have done for at least the last 15 years. The answer to this question goes right to the heart of a processor-centric SOC-design philosophy. Design teams are already choking on the amount of new custom logic required by today's nanometer ASICs and SOCs. The verification time required to check newly designed RTL logic blocks now consumes more than 70% of the design team's time and effort. Consequently, custom logic design has become very expensive. The hard reality is that the use of custom RTL for large portions of a chip's design has become unaffordable. Design teams must turn to more automated methods because of the evolving economics of nanometer chip design.

Conclusion

A processor-centric ASIC or SOC design approach offers a solution to the rising costs of ASIC and SOC design by allowing application tasks to be defined in firmware that runs on processors that can be made fast enough, through configurable architectural extensions, to equal the application performance of custom RTL accelerator blocks. Evolving to a more processor-centric design methodology moves much of the design definition from HDLs (hardware-description languages including Verilog and VHDL) to HLLs (high-level languages including C and C++), which offers several big advantages:

1. Many applications are already defined by C programs that serve as the reference definitions for the application. Audio, video, and image compression/decompression standards are good examples of applications that are defined by C programs, not simple text descriptions.
2. There are many more HLL programmers in the electronics industry than there are HDL programmers, so it's actually easier to assemble a chip-design team if the emphasis is on firmware-based, processor-centric design.
3. The use of multiple processor cores (all based on a common, extensible architecture) for a chip's design creates a design environment with a consistent set of software-development tools that work across many on-chip task blocks. This common set of development tools means that a developer's software skills are more mobile and can be used for more than one of the on-chip blocks. Put another way, developer's skills are more portable when using processor-centric design.
4. Finally and, perhaps most importantly, the operation of processor-centric ASIC and SOC designs can be changed after the SOC is manufactured without re-spinning the SOC (as long as the processors' code is stored in an EPROM and downloaded into the processors' RAM during boot time).

This last aspect provides many benefits to the chip-design team:

1. It reduces the risk involved in ASIC and SOC design by reducing the need for silicon re-spins. Many bugs can be fixed and many new features can be added with firmware changes alone.
2. It allows one chip to be used in several end products within a product family, simply by changing the SOC's firmware.
3. It allows a design team to alter a product through firmware after product release, to either add features or to conform to changes that are triggered by changing industry standards or the subsequent introduction of competing products.
4. It allows the design team to use the same SOC for two or more product generations (mid-life kickers).



Note: If you would like help optimizing the computation and the I/O in your next ASIC or SOC design, contact Tensilica for a consultation.

US Sales Offices:

Santa Clara, CA office:
3255-6 Scott Blvd.
Santa Clara, CA 95054
Tel: 408-986-8000
Fax: 408-986-8919

San Diego, CA office:
1902 Wright Place, Suite 200
Carlsbad, CA 92008
Tel: 760-918-5654
Fax: 760-918-5505

Boston, MA office:
25 Mall Road, Suite 300
Burlington, MA 01803
Tel: 781-238-6702 x8352
Fax: 781-820-7128

International Sales Offices:

Yokohama office (Japan):
Xte Shin-Yokohama Building 2F
3-12-4, Shin-Yokohama, Kohoku-ku,
Yokohama
222-0033, Japan
Tel: 045-477-3373 (+81-45-477-3373)
Fax: 045-477-3375 (+81-45-477-3375)

UK office (Europe HQ):
Asmec Centre
Eagle House
The Ring
Bracknell
Berkshire
RG12 1HB
Tel : +44 1344 38 20 41
Fax : +44 1344 30 31 92

Israel:
Amos Technologies
Moshe Stein
moshe@amost.co.il

Beijing office (China HQ):
Room 1109, B Building, Bo Tai Guo Ji,
122th Building of Nan Hu Dong Yuan, Wang Jing,
Chao Yang District, Beijing, PRC
Postcode: 100102
Tel: (86)-10-84714323
Fax: (86)-10-84724103

Taiwan office:
7F-6, No. 16, JiHe Road, ShihLin Dist,
Taipei 111, Taiwan ROC
Tel: 886-2-2772-2269
Fax: 886-2-66104328

Seoul, Korea office:
27th FL., Korea World Trade Center,
159-1, Samsung-dong, Kangnam-gu,
Seoul 135-729, Korea
Tel: 82-2-6007-2745
Fax: 82-2-6007-2746