What If You Could Design Tomorrow's System Today?

Neal Wingen
NXP Semiconductors, USA

***Abstract*—This paper highlights a series of proven concepts aimed at facilitating the design of next generation systems. Practical system design examples are examined and provide insight on how to cope with today's complex design challenges.**

1 INTRODUCTION

In today's economy, consumers are always searching for the "next experience". Consumers have grown to expect a steady stream of new and constantly improving products like cell phones, televisions, automobiles, computers, and other electronic gadgets. To meet this demand, the semiconductor design community must be able to release next generation systems with a regular "heart beat".

There are several techniques and methodologies that have been employed to aid in this effort. A hierarchical design methodology known as "Islands of Synchronicity" (IoS) in tandem with advanced System-on-Chip (SoC) infrastructures can be used to break down large, complex systems into independent, locally synchronous islands. The Islands of Power methodology takes this one step further and accounts for the advanced power management required in today's systems. Platform based design becomes a reality when these methodologies and the infrastructures are fully leveraged and the right chip-to-chip interconnect is available. Finally, the latest design automation tooling can speed the integration and verification of these systems.

2 ISLANDS OF SYNCHRONICITY

2.1 Synchronous Design Limitations

As SoC designs grow bigger, the globally synchronous design grows slower. In a globally synchronous environment, the maximum operating frequency of on-chip interconnect is limited by the absolute delay between the source and destination. Interconnect delays are increasing with each new geometry node and wire delay now dominates logic delay for long traces. Buffer insertion to balance clock trees, to resolve edge rates and to avoid cross-talk problems can further limit the operating speed of the interconnect.

Another ominous trend for synchronous design regards clock distribution. As SoC designs grow bigger and contain more clock domains, clock tree balancing becomes very difficult to achieve at any operating point, and is further complicated by variations in process, voltage and temperature across the die and intentional variations in voltage for power management. Globally synchronous designs with deep clock buffer trees can suffer from large instantaneous power surges and Electro Magnetic Interference (EMI) coincident with the clock switching activity.

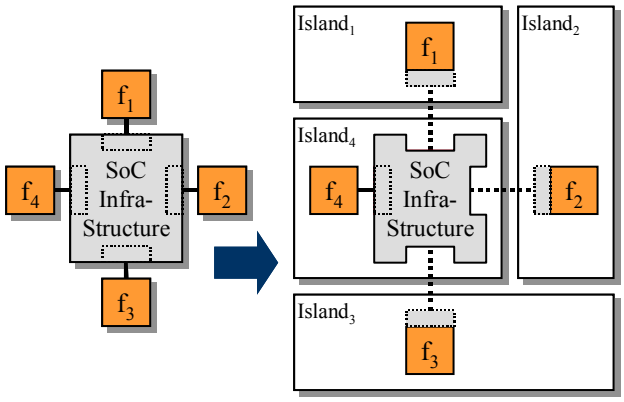
This combination of interconnect performance limitations and global clocking problems requires new solutions that scale with technology and alleviate timing closure issues.

2.2 Hierarchical Design Trend

The IoS methodology provides unique solutions for the hierarchical design process. The move to a hierarchical approach is inevitable for most large SoC designs. Designs are just too large and complex to be completed flat. Rather than push EDA tools and compute resources beyond their capabilities, it makes sense to break designs into smaller, manageable pieces. Different design teams in different locations can then implement the design, each with its own area of expertise, in a concurrent engineering environment.

2.3 SoC Infrastructure "Seams"

The key to hierarchical design is efficient partitioning and re-assembly. The SoC infrastructure must have "seams" to guide the partitioning of the design and identify possible clock and voltage domain boundaries. In Figure 1, the seams embedded in the infrastructure can be split to allow the design to be partitioned into "islands". Each seam straddles two islands and provides asynchronous or source synchronous communication between them.



a) "Seams" in SoC Infrastructure. b) "Seams" split to make islands.

Figure 1. Communication Centric System-on-Chip with "Seams"

The non-synchronous inter-island communication keeps the timing of the islands independent of the top-level design. It also removes the need for global synchronous clocking and allows clock trees to be implemented locally within islands, making them easier to balance with fewer levels of buffering. In fact, clock balancing between islands is not needed. The

978-3-9810801-2-4/DATE07 © 2007 EDAA

overall result is a SoC that is globally asynchronous, but locally synchronous within the islands.

2.4 Seam Classification

There are several varieties of seams based on the possible relationships two islands or domains can have with one another. A "phase seam" is needed between two islands that operate at the same frequency and voltage, but different clock phase. When two islands operate at different frequencies, a "frequency seam" is needed to enable the communication at different rates. A "voltage seam" provides isolation logic necessary when two islands communicate but one or both sides are capable of turning off. A combination "voltage-frequency" seam is needed for communication between islands with varying voltages and frequencies, which is common in designs that utilize Dynamic Voltage Frequency Scaling (DVFS).

2.5 Islands of Power Methodology

New power management techniques like DVFS require further advancements beyond IoS. Islands of Power (IoP) is an evolving methodology to enable multiple variable supply voltage (MVSV) designs. IoP is a comprehensive collection and alignment of many disciplines, including power aware: library components, memories, power infrastructure IP, SoC interconnect, compute engines, EDA tooling and flows, board components and software.

In DVFS, the major power consumers in the SoC operate as close to their optimum efficiency curve with respect to voltage and frequency (Figure 2a). By operating as close to the failure point as possible, the power consumption is as small as possible. In a closed loop mode of operation, hardware automatically finds the optimum voltage given any desired frequency (Figure 2b).

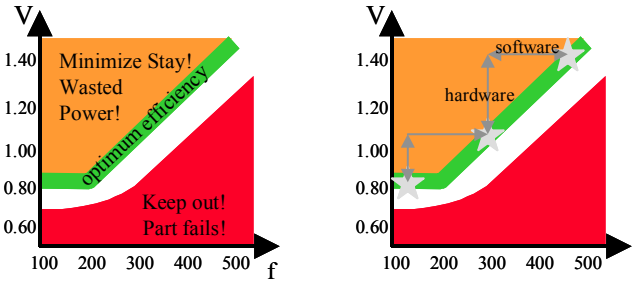


Figure 2. Dynamic Voltage Frequency Scaling Applied to a Hypothetical Major Power Consumer.

To raise performance, one must wait for the hardware to find the new voltage before we increase the frequency via software. Conversely, to lower performance, one can lower the frequency immediately while the hardware finds the lowest operating voltage automatically. The optimum efficiency curve can change with process and temperature and closed loop DVFS hardware can account for this fluctuation and can still find the optimum point for any die at any operating condition.

An open loop mode of operation utilizes a look up table of frequencies and voltage stored in software. In the open loop model, the performance must be chosen for worst case parts and does not adjust for processing and temperature fluctuations.

Since DVFS controls the frequency and voltage components of the power equation, both dynamic and static power can be reduced.

2.6 Voltage Seams

Voltage seams account for when one side of the seam turns off voltage. When this happens, the un-driven input signals need to be clamped and held to their reset states. Signals traversing from low to high voltage domain also need level shifters when the voltage difference is great enough. Isolation logic is inserted in receiving domain. In Figure 3, the source island can turn off so the destination island utilizes a combination level shift-clamp cells to provide the necessary isolation.

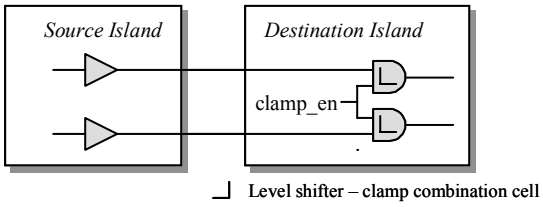


Figure 3. Voltage only seam.

2.7 Asynchronous Based Voltage Frequency Seams

An asynchronous link uses a two-phase or four-phase handshake mechanism to pass data safely between islands. Figure 5 shows a typical implementation with a request (REQ) and acknowledge (ACK) signal pair performing the handshake. The handshake signals are synchronized to the clock domain they travel to.

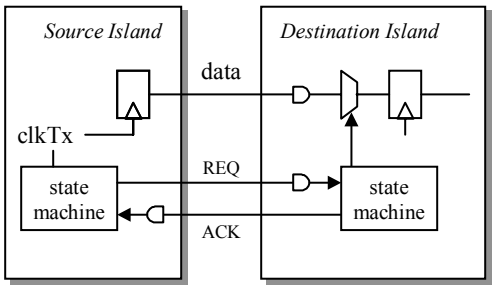


Figure 5. Asynchronous Seam.

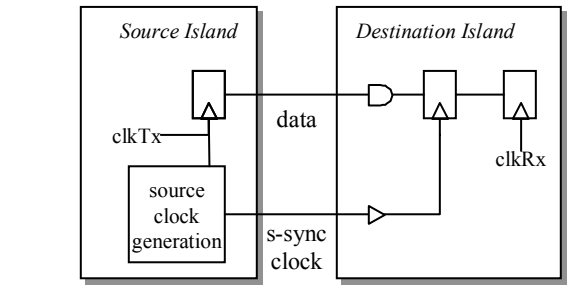
The 4-phase handshake seams are naturally voltage tolerant, as the handshake protocol always returns to a known state. A 2-phase handshake seam is toggle based and more difficult to predict. When voltage domains turn off and on, a false handshake trigger could occur. To guard against this, both sides of a 2-phase handshake need to be held in reset when either side powers down.

2.8 *Source Synchronous Based Voltage Frequency Seams*

A source synchronous link (Figure 4a) transfers the clock with the data payload from the source island to the destination island. To limit power consumption, the clock only needs to be sent when data is present. Tokens or sideband signals can handle flow control. Typically, storage or data buffering is best kept on the destination side of the seam. Data storage size can be tuned to match cache line size, for instance, and can minimize impact of the latency associated with the link.

Source synchronous links can operate at very high speeds across long distances. The speed of the link is limited by clock and the data payload skew alignment (Figure 4b), not propagation delays like a synchronous connection. Source synchronous links can be easy to partition if the source and destination sides are delivered in separate modules.

Source synchronous links may also need provisions to reset both sides of the link whenever one side powers down to maintain correct pointer and token values.



a) Source Synchronous Seam Structure

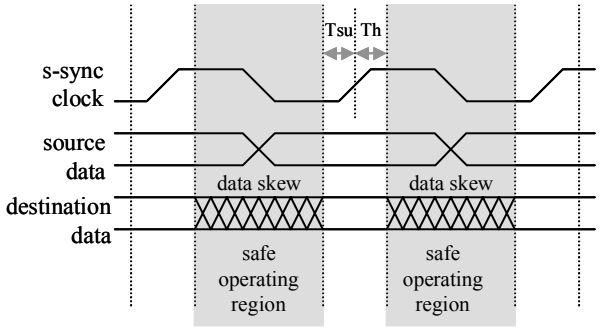


Figure 4. Source Synchronous Seam.

2.9 *IoS - IoP Design Considerations*

2.9.1 *Partitioning Criteria*

The chip architect must understand the performance requirements, floor plan, connectivity and use case information to create effective partitions. There is currently no automated way to determine the optimum partitioning for a large SoC.

Island sizes will vary based on the maximum operating frequency. A smaller island will typically run faster than a larger island. The island size should also be suitable for EDA tools to produce short run-times.

Performance or latency requirements will also determine partition placement. Partition seams have latency associated with them. Communication that is latency critical should be self-contained in an island and not travel across a seam to another island.

Voltage domains also influence the partitioning strategy. Major power consumers may be controlled using DVFS techniques and will require hierarchy levels to contain the specific voltage domains. Use cases may require voltage domains to turn off in certain situations to meet power budget requirements. Power supply unit capabilities and internal or external voltage switches will further determine hierarchy and may force voltages to be combined or separated.

Partitioning also needs to consider the physical floor plan. Pin intensive blocks will naturally prefer to be close to the pads and not combined with a large embedded block. The design should be partitioned in a way to limit the top-level wires or the connections on the island boundaries. Major power consumers should have dedicated power supplies and should be placed as close as possible to the voltage source. IR-drop also needs to be tightly limited otherwise the voltage margin needed for effective DVFS operation is reduced.

2.9.2 *Performance*

The source synchronous and asynchronous links exhibit more latency cycles than the synchronous equivalents. But caches can do an excellent job of hiding these latencies. The latency impact is only exposed when a cache miss occurs. In a real-life example that shows the benefit of caching, a VLIW DSP running an H.264 video decode benchmark with a source synchronous link towards the main memory controller had virtually identical performance when compared to a direct synchronous connection. Of course, the source synchronous implementation did not require clock balancing and was easier to timing close. It is very difficult to theorize cache hits and misses without running an application, but reasonable cache hit rates would suggest only a modest impact if an embedded application processor had source synchronous links to the memory network. Another consideration is that asynchronous design can run faster than synchronous design, especially for large devices as simplified timing closure yields higher frequencies. Asynchronous design can actually be higher overall performance than synchronous equivalent in some cases.

The added delay needed for the isolation logic to separate voltage domains is absorbed well in a voltage seam since the added propagation delay is of no consequence. This would not be the case for a synchronous design.

2.9.3 *Area Impact*

It is difficult to say categorically that an IoS methodology increases chip area. In some cases, asynchronous versions of networks can be smaller in area and net count when compared to a synchronous counterpart. Source synchronous links do have an area overhead associated with them; however, for large SoC designs, the added area is typically negligible.

3 PLATFORM BASED DESIGN

In most cases, the next generation product is not a revolutionary new architecture, but merely an extension or improvement of the current product. Therefore, to make the next generation product, new functions need to be added to the

existing product. In platform-based design, a platform is built of the generation N product and this platform is used to create as many derivatives (generation N+1) as possible.

3.1 History

Making derivatives from a platform is not a new idea. The automotive and personal computer industries are good examples of successful platform based design. An automotive platform can be applied to several different derivative car styles and models. Automotive designers can concentrate on the differentiating features of the automobile that return the maximum value and not the underlying infrastructures. The automotive industry has proven that platform based design reduces engineering costs, speeds up development time and makes it possible to offer a greater variety of new models. Platform Based Design has never flourished within the semiconductor industry; however, new industry chip-to-chip interconnect standards can drive the concept into the mainstream.

3.2 Hardware Platforms

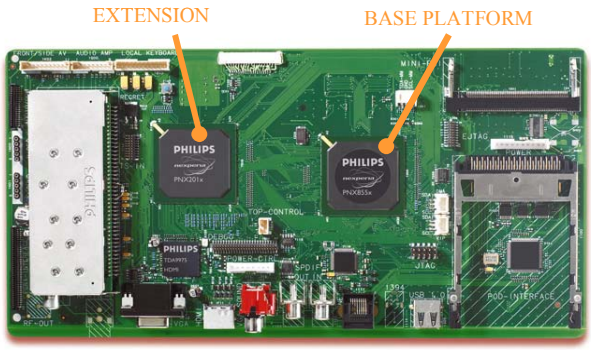


Figure 6. Nexperia TV810 Digital Television Platform.

Nexperia™ is the NXP brand for a unique group of platforms and products that streamline development of next-generation, connected multimedia appliances. From highly integrated, programmable system-on-chip (SoC) and companion ICs to reference designs, system software, and development tools, flexible Nexperia solutions help manufacturers meet demands for new products. Figure 6 shows one example of how the Nexperia platform concept has been applied to digital television sets. Manufacturers use the base platform to build their basic sets. The platform is extended to add features necessary for their higher end television sets. The extension, in this case, is a companion IC; however, an FGPA is also a common choice. A high bandwidth, low pin count chip-to-chip interconnect enables the two IC's to communicate with each other.

3.3 Chip-to-Chip Interconnect

The chip-to-chip interconnect used in Figure 6 is from a family of NXP proprietary links called Transaction Transport Tunnels. These tunnel mechanisms serve as a bridge to make the extended logic appear as if it is actually on the original base platform or reference design.

In Figure 7, IC1 represents the generation N product and includes a tunnel to extend functionality in the future. The tunnel contains several layers of abstraction, including transaction, transport and data link layers and just appears as an extra port on the internal bus network. By connecting IC2 to IC1 via the tunnel, the next generation product, generation N+1, is created.

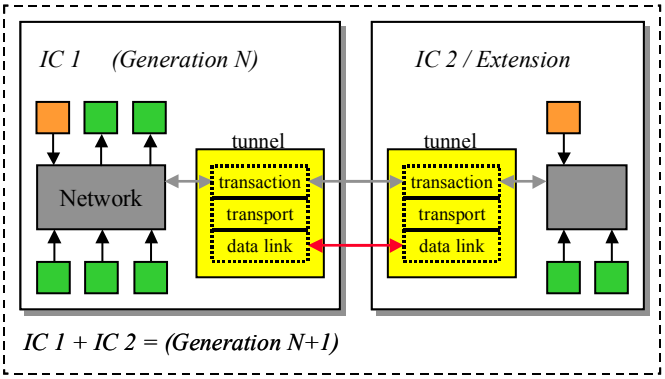


Figure 7. Platform Extension with Chip-to-Chip Interconnect.

In the transaction layer, native bus transactions, such as AXI, APB and AHB, are passed implicitly between the IC's. From the IC perspective, only these native bus transaction passing between the IC's are visible.

The transport layer converts the bus transactions into packet formats needed to pass data, address, general-purpose signaling, credits and error correction information. The general purpose signaling is useful for passing power state information, interrupts or any other relatively low frequency signaling back and forth. These general-purpose signals can be transported in a dedicated packet, which can be inserted anywhere in the packet stream in order to keep the latency reasonable.

The physical communication link between the two IC's occurs in the data link layer. The link information can be specially encoded and decoded for DC balancing, to increase noise immunity and minimize power. Low swing I/O are also a popular choice to improve speed and further reduce power. In some cases, a PHY device can be used to boost performance and further minimize pin count. By making this layer source-synchronous, both sides of the link are free of synchronous frequency and phase relationship constraints and the link must only balance skew between the clock and data payload.

The tunnel has no software view and is transparent from the application perspective. The IC's only see the native transaction protocol, not the special transport protocol or data link layer. Therefore, the tunnel logic can simply be removed from the final product without any impact. Of course, the tunnel mechanisms can be left in, as well, to serve as a means for possible product differentiation in the future.

The tunnel data width can be configured or tuned to meet the performance and bandwidth requirements for the application. Since tunnels are packet based, fewer pins are necessary, especially when compared to a traditional, on-chip

bus being brought off-chip. By using source synchronous techniques, the tunnels can maintain or exceed the full speed of the internal bus network, despite traveling off-chip. The internal, synchronous bus networks are not suited for use off-chip; resulting in a slower performance and higher pin cost.

Since the base platform and the extension logic can operate at full speed, real-time prototyping can be utilized. By operating at the target system speed, real-time system problems can be exposed and many orders of magnitude more patterns can be exercised than traditional simulation. In addition, full speed, real-time operation enables system capabilities to be fully demonstrated. Engineers are more efficient at debugging video applications when they can view video streams and hear audio streams real time. Also, the actual silicon running at-speed can be more accurate than a model.

3.4 Tunnels

The AXI Tunnel (also called TAXI, for Tunnel-AXI) was created by NXP as a way to extend an on-chip AXI bus or interconnect to a second, off-chip AXI interconnect. Because information from all five channels of AXI must be sent through a single connection, bandwidth through the link is limited. For example, with an 8-bit interface operating at 150MHz, the peak bandwidth is about 200 MB/s (100MB/s in each direction). With a 48-bit interface, the bandwidth can be six times higher, but at a cost of nearly six times more pins. The AXI tunnel gate count is 16K-20K gates, depending on the configuration.

The CTL12 Tunnel utilizes an NXP proprietary transaction layer. The transport layer is comprised of 12-bit symbols while the data link layer utilizes 6b8b decoding. This tunnel can achieve a relatively high bandwidth (1066MB/s) with a modest amount of pins (39).

In the future, several industry standard serial interconnects may become excellent tunnel candidates. The Mobile Industry Processor Interface (MIPI) Alliance has defined the UniPro 1.0 interface as a mobile terminal device standard and an 8-bit interface is capable of 250MB/s. Inter-Chip USB extends the scope of USB 2.0 to include the use of USB within embedded systems. A full speed IC-USB link can accommodate 1.5MB/s given a few pins, while the high-speed version will handle 60MB/s. PCI Express can deliver an impressive 500MB/s through a 4-bit interface.

In general, the industry standards are very robust and have a larger footprint or gate count than the proprietary interfaces. Nevertheless, as these interfaces become commodities and silicon integration capabilities continue their expansion, new opportunities such as Platform Based Design, become more feasible.

3.5 Extension Examples

The extension logic provides additional functionality; either not considered at the time of the original IC design or that is optional and well suited for a separate IC. For prototyping purposes, a new IC could be prototyped using an existing IC together with an FGPA. Also, the companion IC could be in a different technology than the base platform.

3.6 Architecture Considerations

When designing the platform, it is important to make sure the available bandwidth of the tunnel can accommodate all the possible extensions. In a processor-centric platform, the CPU would typically require a majority of the available bandwidth through the tunnel device. A product centric platform, where a majority of the functionality already exists in the platform, may only require a small additional bandwidth through the tunnel. If the extension will process a data stream, the tunnel must be able to accommodate the data bandwidth with the appropriate margin above and beyond.

4 SoC Integration and Verification

4.1 Nx-Builder

Today's SoC designs are a myriad of reusable IP components including peripherals, infrastructures, compute engines, subsystems, and even entire IC's. As a result, SoC integration and verification have become monumental tasks and the next frontier for the EDA industry. To further facilitate this effort, the Spirit Consortium [1] has established a set of IP and tool integration standards with the IP-XACT specification [2] as the first embodiment. Automation and standards have the potential to reduce the many months of effort required for SoC integration and verification to weeks, directly impacting time to market. Nx-Builder is an NXP proprietary SoC design environment built on top of Platform Express from Mentor Graphics and uses a simple "enter, build and verify" approach to system architecture development.

4.1.1 Design Entry

Design entry is performed with a high-level schematic editor or a table based entry mechanism. IP is added to the design with a drag-and-drop approach. Generally, an IP block is selected from the library window and dropped onto the bus it is to connect to. If the IP is configurable, the user will be prompted to provide any necessary parameters. Each IP block has an "electronic data-sheet" in eXtensible Markup Language (XML) [3] form that allows the IP to evolve with the design.

There are a number of automation mechanisms in place to simplify and speed entry. A large design will contain a daunting number of configuration parameters; however, as many parameters as possible are automatically derived from the system context leaving minimal user input. The tool is also capable of automatically adding interface IP (bus bridges, adapters) and infrastructure IP (clocks, resets, interrupts, interconnect networks) during the course of design entry. In addition, the tool maintains an automated interaction with the IP repository to track the abstraction of the IP and flag any possible errors.

4.1.2 Design Build

Once the design has been entered and defined, the underlying design files can be generated and compiled. The front-end views (RTL, models, documentation etc) are extracted from the IP repository given the configuration parameters defined in the design entry phase. A standardized

file structure helps assemble the individual IP directories in to a cohesive database. In addition, batch mode flow scripts are created for synthesis, simulation and design-for-test insertion. Simple "make" files control the generation and compilation activities, including provisions to choose a specific hardware model type or source.

4.1.3 Design Verification

Software is automatically created to exercise the system hardware. Each piece of IP includes a software module that tests that particular IP when instantiated in a system. Special software module "stubs" exercise external interfaces via software, too. All of the software modules are combined and compiled for the target system to make a self-checking executable ready for hardware-software co-simulation or real-time verification in an FGPA environment. Different software packages corresponding to various use cases can also be created. Software to stress performance, quality of service properties and power management use cases will become available in the future.

The generated software can verify the integrity of the IP interconnect to the system and give confidence that the design has been connected correctly. Also, the SoC infrastructure (busses, bridges, register space, memories) can be stressed and verified. Modules can be exercised simultaneously, to verify interoperability of the system.

Other design flow scripts for synthesis, static timing analysis, formal equivalence checking and Design-for-Test insertion can be executed in this stage as well.

4.2 Methodology Reuse

Nx-Builder is created based upon designers' experience and standards. It essentially encapsulates the best engineering knowledge and makes this available to the entire user community, providing a generic framework that enables different SoC designers to implement their systems in a consistent, flexible and easy-to-use way.

4.3 Architecture Reuse

Nx-Builder also raises the level of design reuse, beyond the traditional IP level. Integration reuse determines how the IP interconnections are made; including the connections to the SoC infrastructures and how to constrain these interconnect nets. The delivery of simulation and synthesis scripts specific to the architecture specified enforces methodology reuse. The SoC infrastructure is automatically generated based on the system context providing a means for infrastructure reuse. As architectures become IP, architecture reuse and software reuse matures.

4.4 Virtual Platforms

Nx-Builder provides a flexible way to define systems or "virtual" platforms and makes it easy to build and evaluate platform and derivative architectures, including software, without the manufacturing commitment. Virtual platforms are cheaper than hardware platforms and can be put on the workstation of every software developer. Once the initial design is available in Nx-Builder, SoC designs or derivatives can be developed and verified in a minimum amount of time.

5 CONCLUSION

A platform device has been developed that utilizes all of the concepts included herein (Figure 8).

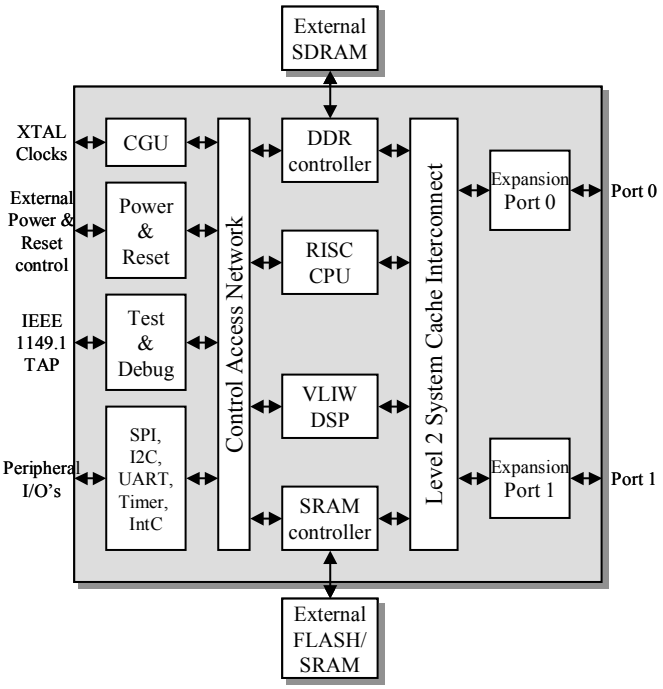


Figure 8. Energizer II Platform Chip.

The SoC consists of 11 islands. The 3 major power consumers (RISC CPU, VLIW DSP and L2 System Cache) are controlled using DVFS. High bandwidth expansion ports enable the platform to be extended, with graphics or cellular modem subsystems for example. Nx-Builder also played a role in the integration and verification effort. This platform chip is 42mm2 in a 65nm process.

ACKNOWLEDGMENT

Tim Pontius, Greg Ehmann, George Spatz, Marino Strik, Ramon Baas, Marc Heijligers

REFERENCES

[1] SPIRIT Consortium, http://www.spiritconsortium.com
[2] SPIRIT Schema Working Group, "IP-XACT User Guide", Version 1.2, July 2006
[3] eXtensible Markup Language [Online]. Available at http://www.w3schools.com/xml/xml_syntax.asp