

**ULTRA-LOW POWER? THINK MULTI-ASIP SOC!**

**Gert Goossens, Johan Van Praet, Dirk Lanneer, Werner Geurts**  
**Target Compiler Technologies**

**Leuven, Belgium**

ABSTRACT

This paper reviews architectural strategies for the design of ultra-low power systems-on-chip. Drawing from experience in power-critical markets like hearing instruments, we contend that multi-processor SoC (MPSoC) architectures must be heterogeneous and based on application-specific processors (ASIPs), in order to meet the ultra-low power requirements of next-generation telecom and multi-media systems. Low-power optimisations for MPSoC architectures, and a retargetable tool suite for power-efficient ASIP design are introduced.

I. INTRODUCTION

In the coming years, multi-processor systems-on-chip (MPSoCs) are expected to become the platform of choice for new feature-rich devices in high-volume markets like wireless telecom and portable multi-media. With increasing chip densities and an ever growing quest for both more functionality and longer battery autonomy, low energy consumption becomes of the essence in the design of these systems.

Table 1 summarises some key elements of the International Technology Roadmap for Semiconductors (ITRS) for the next decade, relating to power dissipation. While system and chip complexity will increase dramatically, the allowable power budget is expected to increase only moderately for desktop and power-managed high-performance designs, and to even stay constant for battery-powered hand-held designs. The reduction of technology geometries will not automatically result in significant power savings because of the growing importance of static (or leakage) power. Therefore, new architectural methodologies are indispensable to meet the power challenges of SoCs.

**Table 1 - ITRS roadmap for power supply and allowable dissipation [1]**

Year of production	2007	2010	2013	2016
Technology node	65	45	32	22
Power supply voltage – V <sub>dd</sub> (Volts)				
- High-performance	1.1	1.0	0.9	0.8
- Low-power	0.8	0.7	0.6	0.5
Allowable maximum power (Watts)				
- Desktop	189	198	198	198
- Power-managed high-performance	104	119	137	151
- Battery-powered hand-held	3.0	3.0	3.0	3.0

In this paper, we focus on heterogeneous MPSoC architectures, as the preferred platform to meet the ultra-low power requirements of next-generation telecom and multi-media systems. Rather than replicating identical processor cores and let them communicate via a general-purpose on-chip network, we believe that each processor core as well as the communication network must be optimised for the application. This provides for an optimal balance of architectural resources, supporting both task-level, data-level, and instruction-level parallelism, resulting in lowest energy consumption. MPSoCs based on application-specific instruction-set processors (ASIPs) thus allow to push the power envelope to the next level.

This paper also shows how multi-ASIP SoCs become a reality thanks to the use of a power-aware retargetable tool suite. Such a tool suite supports architectural exploration and profiling for ASIPs, automatic RTL generation, and software development support in terms of optimising C compilation, instruction-set simulation and debugging.

In the deep sub-micron era, low-power design requires a holistic approach. In other words, one must simultaneously optimise for power in all phases of the design. MPSoC design using ASIPs effectively addresses power optimisations in each of the following design phases: system architecture design, processor architecture design, and logic design.

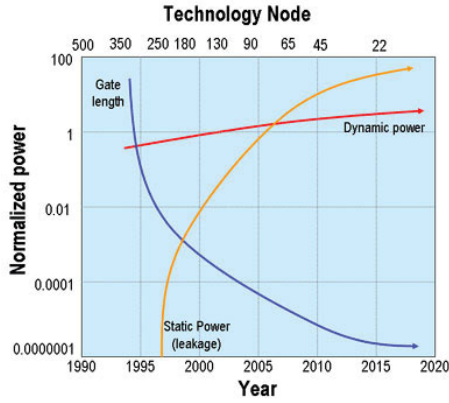
This paper is organised as follows. Section II reviews the main sources of power dissipation in digital circuits. Sections III, IV, and V then describe how these elements influence the design decisions in each of the three design phases of the holistic optimisation approach. A concrete design example is discussed Section VI. Conclusions are drawn in Section VII.

II. SOURCES OF POWER DISSIPATION

Whereas early approaches for low-power design primarily focussed on dynamic power [2], Figure 1 illustrates that from the 65 nm technology node onwards, static power dissipated by a circuit will likely surpass its dynamic power. Below, a brief review is presented of the main sources of both dynamic and static power.

*A. Dynamic power*

The dynamic power dissipation of a circuit is largely due to the capacitance effect, as expressed by the following equation [4]:



**Figure 1 - Evolution of dynamic versus static power dissipation [3]**

$$P_{dyn} = C \times (Act \times f_{clock}) \times V_{dd}^2 \quad (1)$$

with  $C$  the total capacitance,  $Act$  a switching activity factor,  $f_{clock}$  the clock frequency, and  $V_{dd}$  the supply voltage. This suggests the following remedies to reduce dynamic power dissipation:

- *Supply voltage reduction:* Due to the quadratic dependency in equation (1), important power savings are obtained by reducing  $V_{dd}$ . Low-power design therefore relies on the use of low-voltage technologies. Likewise, dynamic voltage scaling techniques can be applied.
- *Concurrency exploitation:* The concurrent execution of multiple operations will save power. Its effect is that the same application can be completed in fewer clock cycles, which allows clocking the system at a lower frequency. As shown in equation (1), this results in a linear reduction of dynamic power.

In addition, note that a reduction of the supply voltage as discussed in the first item above slows down the logic, and therefore may also necessitate a reduction of the clock frequency, which is to be compensated by more concurrent execution.

Three types of concurrency (or parallelism) are generally distinguished:

- Task-level parallelism, i.e. the parallel execution of multiple threads of operation.
  - Data-level parallelism, i.e. the same instruction stream is simultaneously applied to multiple data samples. This is also referred to as “single instruction, multiple data” (SIMD) computing, or “vector processing”.
  - Instruction-level parallelism, i.e. the parallel execution of multiple operations in a single instruction.
- *Switching activity reduction:* Several techniques can be applied to avoid unnecessary switching in logic circuitry. This reduces the activity factor in equation (1).

To reduce switching related to functional units, the following techniques are applicable: clock gating (i.e.

turning of the clock for a part of the circuit when it is not used) and operand isolation (i.e. keeping a logic function’s inputs constant at times when the function’s output is not used).

Furthermore, it is important to avoid switching on long, high-capacitance communication paths, by keeping computations localised as much as possible. This principle is referred to as “locality of reference”.

### B. Static power

The static power dissipation of a circuit can be calculated as the product of static (or leakage) current and the supply voltage [4][5]:

$$P_{stat} = I_{stat} \times V_{dd} \quad (2)$$

$I_{stat}$  is a negative exponential function of the threshold voltage  $V_t$ , and proportional to the number of logic gates  $N_{gates}$  and to the average device length  $W_{dev}$ . Therefore, equation (2) can be further elaborated as:

$$P_{stat} \sim (a^{-V_t} \times N_{gates} \times W_{dev}) \times V_{dd} \quad (3)$$

This suggests the following remedies to reduce static power dissipation:

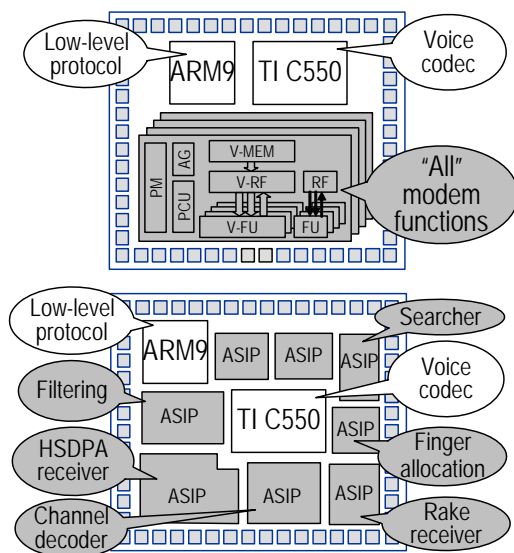
- *Synthesis with multi-threshold libraries:* Threshold voltages decrease with the introduction of new technology nodes. Semiconductor and library vendors are developing libraries in which multiple instances of logic cells are provided with different threshold voltages. Logic synthesis tools can then use faster low- $V_t$  cells in the critical timing path, and slower but more power-efficient high- $V_t$  cells off the critical path [6].
- *Power gating:* Sub-circuits can be connected to the  $V_{dd}$  or ground rail via a power gating transistor. When the sub-circuit is not used, this allows cutting off the power supply [7].
- *Minimal logic:* Keeping the number of gates  $N_{gates}$  as low as possible contributes to static power reduction. In terms of processor cores, this calls for the use of application-specific, rather than general-purpose, architectures.

In the subsequent sections, we will discuss how the above remedies for reducing dynamic and static power dissipation impact the design decisions in each of the MPSoC design phases (i.e. system architecture design, processor architecture design, and logic design) and how a multi-ASIP approach can make the difference.

## III. SYSTEM ARCHITECTURE DESIGN

While MPSoCs are becoming the platform of choice for next-generation wireless and multi-media systems, two broad classes of MPSoC architectures can be distinguished (see Figure 2):

- *Homogeneous MPSoCs, using general-purpose processors:* A single general-purpose processor is replicated a number of times. The system functionality must be partitioned and assigned to the available general-purpose processor instances.



**Figure 2 - Conceptual diagram of homogeneous (top) and heterogeneous (bottom) MPSoC, for 3G wireless terminal**

Several architectures have been announced that belong to this class. In some cases, the general-purpose processor is a very-long instruction word (VLIW) or single-instruction multiple-data (SIMD) machine [8][9][10]. In other cases a 2-dimensional array processor [11] or a large array of basic microprocessors [12][13] is used.

- *Heterogeneous MPSoCs, using ASIPs:* Multiple ASIPs are combined, each optimised for a single or a small set of system functions.

Such multi-ASIP architectures can be based on configurable processor templates offered by intellectual property vendors [14][15], or generated with a retargetable tool suite [16][19].

While homogeneous MPSoCs are intended to implement a wide range of different algorithms, heterogeneous MPSoCs are inherently better suited to meet the challenges of ultra-low power design. The prime reason is that each ASIP is optimised to implement one or a few complete system functions. Communication between ASIPs is thus minimised (locality of reference). At the same time, this results in well balanced task-level concurrency. In contrast, homogeneous MPSoCs more likely exhibit a computational mismatch between system functions and processor, resulting in either under-utilised processors or system functions having to be spread over multiple processors. The latter may result in a control and communications overhead.

Furthermore, in a heterogeneous MPSoC, power gating can be applied straightforwardly at the level of individual ASIPs, under control of system-level status information.

#### IV. PROCESSOR ARCHITECTURE DESIGN

The performance and power dissipation of individual processor cores in an MPSoC can be optimised by tailoring the processor architecture to the functions that it

must implement, i.e. by designing ASIPs. This section elaborates on ASIP architectural optimisations that are beneficial to reduce power.

- *Concurrency:* In an ASIP architecture, both *instruction-level* and *data-level parallelism* can be exploited. Typically this can be achieved at a smaller cost than in a general-purpose architecture.

Clearly, the number of parallel functional units in the ASIP can be tailored to the target applications' needs.

General-purpose processors often employ very-long instruction words (VLIW) to achieve instruction-level parallelism. This may result in more parallel combinations of operations being offered than can be exploited by the target application. In contrast, ASIPs can employ instruction encoding, to encode only those parallel combinations that are useful for the target applications. This results in compact instruction words and reduced power spent in program memory access.

- *Reducing memory access:* Data and program memory accesses often account for a large percentage of power dissipation, due to capacitance effects of data and address busses and to the energy needed for loading the word lines in memory.

ASIP architectures can be designed such as to reduce memory accesses:

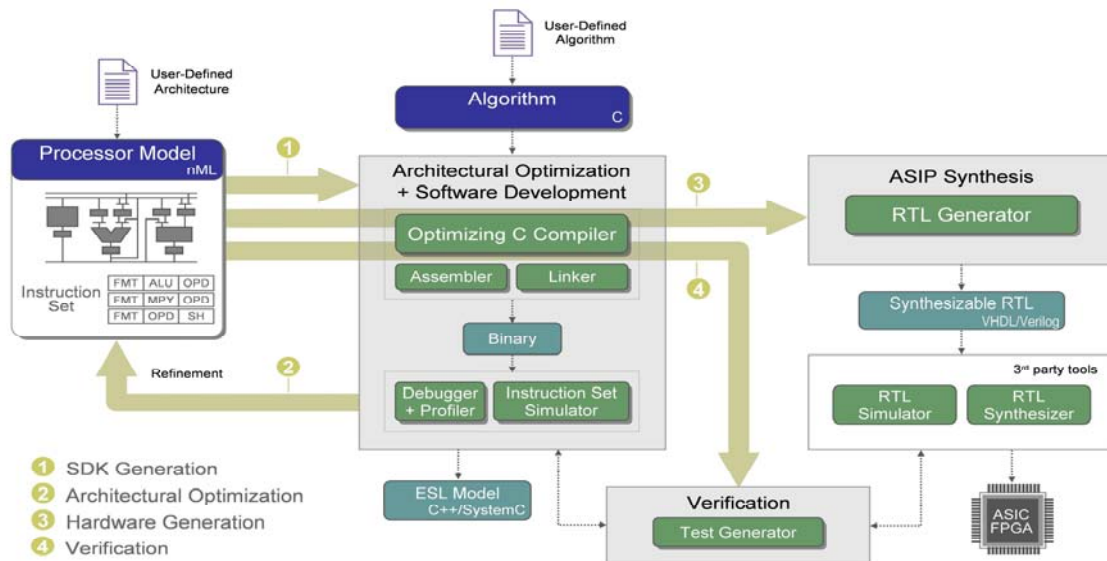
- By designing a distributed register architecture with special-purpose registers that are local to functional units, locality of reference is maintained and data storage in memories can be reduced effectively.
- An application-specific memory hierarchy can be designed, including optimised data and instruction caches, as well as loop buffers.
- As mentioned above, an application-specific encoded instruction set reduces program memory access.

- *Architectural specialisation:* Specialisation clearly leads to minimisation of logic and thus power. This applies both to data types, functional units, and instructions.

An ASIP data path can contain specialised functional units that execute critical portions of the target application in only a small number of instruction cycles. In contrast, general-purpose processors may have to resort to an all-software implementation resulting in a large number of instruction cycles for the same functionality.

The ASIP's data path can be designed such that the logic used in the different stages of the instruction pipeline is well balanced.

While configurable processors available from intellectual property vendors [14][15] cover part of the architectural freedom offered by ASIPs, true ultra-low power design requires a methodology that is not hindered by the architectural restrictions of a configurable processor template. This is the purpose of a retargetable tool suite for ASIP design.



**Figure 3 - IP Designer (Chess/Checkers) tool flow**

Figure 3 depicts the “IP Designer” (Chess/Checkers) retargetable tool suite available from Target Compiler Technologies [16]. The user of IP Designer can easily explore alternative instruction-set architectures in nML, a high-level processor description language that captures a programmer’s view of the architecture [17]. The nML description is an input to a retargetable software development kit (SDK): after loading the nML description, the SDK is instantaneously targeted to the user-defined architecture.

The SDK consists of an optimising C compiler, assembler, linker, and instruction-set simulator with debugging and profiling capabilities. The C compiler employs patented graph-based optimisation techniques to produce highly efficient machine code, even for irregular ASIP architectures [18].

nML and the retargetable SDK are the essential tools for ASIP architectural exploration. Thanks to the SDK’s profiling capabilities, the user can quickly determine potential architectural improvements, modify the nML description accordingly, and accurately verify their impact by recompiling the application. Target Compiler Technologies offers several baseline ASIP architectures described in nML that can serve as a starting point for architectural optimisation.

nML is versatile enough to capture all essential architectural optimisations for low power discussed in Section IV. Therefore, the tools enable (power) optimisations well beyond the capabilities of configurable processor templates.

Once the user has converged on an ASIP architecture, the optimised nML model can be transformed into a synthesisable RTL model, using IP Designer’s RTL generator. Excellent interoperability is provided with all important third-party RTL synthesis and simulation tools. The RTL generator supports various logic optimisations for low power, as further elaborated in Section V.

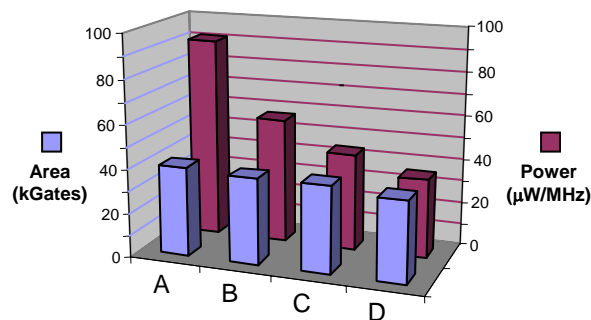
Thanks to this smooth path to hardware, accurate power simulations can be made very quickly for any ASIP architecture coded in nML.

A test program generator is available to build assembly-level test programs for verifying the ASIP hardware integrated in an SoC.

## V. LOGIC DESIGN

To implement power-efficient ASIP architectures, the logic design level must be given proper attention as well.

Figure 4 illustrates the impact of four different RTL optimisations on the power dissipation of an ASIP: selective application of clock gating, operand isolation for functional units, operand isolation for multiplexers, and latching of register addresses in the instruction decoder.



**Figure 4 - Impact of RTL power optimizations, illustrated for an audio ASIP**

All above optimisations are available in the RTL generation path of the IP Designer tool suite. The figure shows measurement data, based on gate level simulations of the resulting circuits. The ASIP in these experiments is an audio DSP implemented in 90 nm CMOS at a clock of 200 MHz. The example shows that combination of all

four optimisations results in a reduction of power dissipation by approximately 60%. Remarkable, the circuit area (gate count) remains almost constant.

## VI. MPSOC DESIGN EXAMPLE

In recent years, multi-ASIP SoCs designed with Target's IP Designer retargetable tool suite have been successfully introduced in a wide range of products. This includes power-sensitive and power-critical systems, such as portable audio and multimedia devices, mobile handsets, wireless sensor nodes, and hearing instruments.

As an example illustrating the effectiveness of this approach, consider the Voyager platform of Sound Design Technologies (formerly Gennum Corporation). The Voyager chip is used for beamforming audio processing in multi-microphone hearing instruments and Bluetooth headsets. It contains multiple ASIPs designed with the IP Designer tool suite.

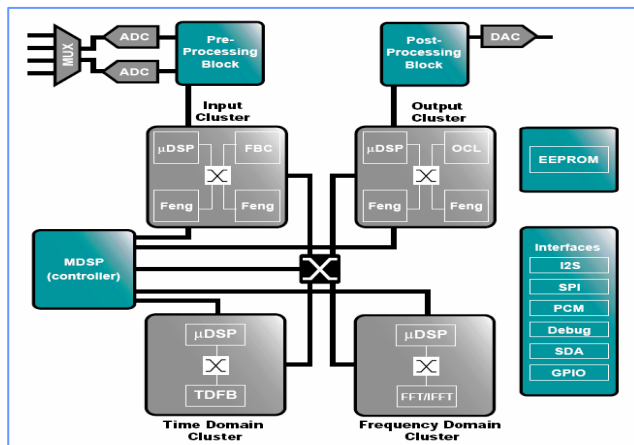


Figure 5 - Block diagram of Voyager MPSoC

Figure 5 shows a block diagram of the Voyager MPSoC. It contains a microprocessor ASIP (MDSP), four instances of an audio signal processing ASIP ( $\mu$ DSP), and four hardwired filter accelerators (Feng). This composition illustrates the abundant use of task-level parallelism. In addition, the platform relies on architectural specialisation. For example, the  $\mu$ DSP ASIP and the filter engines use 20-bit precision, which offers exactly the dynamics range required by the target applications. The  $\mu$ DSP employs instruction-level parallelism, using a dual multiply-accumulate structure with parallel dual memory accesses.

Thanks to the available parallelism and architectural specialisation, the Voyager platform can achieve a computational performance of 42 MIPS, while requiring a clock frequency of only 2 MHz (in 130 nm CMOS) and a power consumption of 1 mW, i.e. only 40  $\mu$ W per multiply-accumulate operation.

## VII. CONCLUSIONS

This paper discussed the use of heterogeneous multi-ASIP SoCs to meet the deep-submicron power dissipation challenges of next-generation electronic systems. Thanks to an optimal balance of task-level, data-level, and instruction-level parallelism, dynamic power can be

reduced. Additionally, by reducing the amount of logic through architectural specialisation, and by applying techniques like power gating, static power can be reduced.

A key element of a multi-ASIP SoC design methodology is the use of a retargetable tool suite for ASIP design and optimisation. Such a tool suite enables energy-efficient optimisations beyond the restrictions of configurable processor templates. At the same time, it guarantees the availability of an efficient software development toolkit for each of the ASIPs. Last but not least, it offers a fast and efficient path to hardware, which includes RTL level power optimisations, and enables fast and accurate power measurements.

## REFERENCES

- [1] "International Technology Roadmap for Semiconductors, 2006 Update, Overview and Working Group Summaries", Sematech, 2006, www.itrs.net.
- [2] A. P. Chandrakasan, S. Sheng, R. W. Brodersen, "Low-power CMOS digital design", IEEE J. Solid State Circuits, Vol. 27, No. 4, April 1992.
- [3] N.S. Kim et al., "Leakage current: Moore's law meets static power", Computer, Vol. 36, No. 12, IEEE Computer Society, December 2003.
- [4] J.M. Rabaey, M. Pedram, "Low power design methodologies", Kluwer Academic Publishers, 1996.
- [5] "Leakage power consumption", Berkeley Advanced Chip Performance Calculator, www.eecs.umich.edu/~dennis/bacpac.
- [6] B. Pangrle, S. Kapoor, "Managing leakage power at 90nm and below", EEdesign.com, May 11, 2004.
- [7] J. Frenkil, S. Venkatraman, "Power gating design automation", in: D. Chinnery, K. Keutzer, "Closing the power gap between ASIC and custom", Springer, 2007.
- [8] C.H. Van Berkel et al., "Vector processing as an enabler for software-defined radio in handsets from 3G+WLAN onwards", Proc. 2004 Software Defined Radio Technical Conf., November 2004.
- [9] J. Glossner et al., "The Sandbridge Sandblaster convergence platform", white paper, www.sandbridgetech.com.
- [10] R. Leupers et al., "SHAPES: a tiled scalable software hardware architecture platform for embedded systems", Proc. 4th Int. Conf. on Hardware/Software Codesign and System Synthesis (CODES+ISSS), 2006.
- [11] B. De Sutter et al., "Hardware and a tool chain for ADRES", in: "Reconfigurable Computing: Architectures and Applications", Springer, 2006.
- [12] K. Amiri, "CT3600 Multi-processor DSP Family: A Closer Look at the Architecture and Target Applications", TechOnLine Seminar, July 12, 2005.
- [13] G. Panesar, "One die. 300 processors. Real systems", MultiCore Expo, 2006.
- [14] "Configurable processors: what, why, how", whitepaper, www.tensilica.com.
- [15] "Customizing a soft microprocessor core", whitepaper, www.arc.com.
- [16] G. Goossens et al., "Design of ASIPs in multi-processor SoCs using the Chess/Checkers retargetable tool suite", Proc. Intl. Symp. on SoC, 2006.
- [17] J. Van Praet et al., "nML: a structural processor modelling language for retargetable compilation and ASIP design", in: P. Mishra, N. Dutt: "Processor description languages - Applications and methodologies", Morgan Kaufmann, 2008.
- [18] J. Van Praet et al., "Processor modelling and code selection for retargetable compilation," ACM Tr. Design Autom. of Electronic Systems, Vol. 6, no. 3, pp. 277-307, July 2001.
- [19] F. Fielder, A. Nohl, A. Hoffman, "A methodology and tooling enabling application-specific processor design", Proc. GSPX, 2004.
- [20] "Voyager TD Open Platform DSP System for Ultra Low Power Audio Processing", Sound Design Technologies, www.sounddesigntechnologies.com/products/pdf/37601DOC.pdf.