Université catholique de Louvain
Ecole Polytechnique de Louvain
ICTEAM Institute

Towards the design of a pseudo-synchronous multi-processor system-on-chip for processing-intensive ultra-low-power applications

François Botman

Thesis submitted in partial fulfillment
of the requirements for the degree of
Docteur en sciences appliquées

Dissertation committee:
Prof. Jean-Didier Legat (UCLouvain, Belgium), *Advisor*,
Prof. David Bol (UCLouvain, Belgium), *Advisor*,
Prof. Denis Flandre (UCLouvain, Belgium), *President*,
Prof. François-Xavier Standaert (UCLouvain, Belgium),
Prof. Marc Belleville (CEA-LETI, France),
Prof. Andreas Burg (EPFL, Switzerland).

September 2014

CONTENTS

ACKNOWLEDGMENTS

A thesis is not only the culmination of years of research resulting in an advancement of the state-of-the-art for a given field; it is also a personal journey leading to the mastery of the chosen field and the attainment of the scientific culture and rigour that such an undertaking requires. Such a journey is not a lone voyage, for it is the meetings and interactions with extraordinary people that enrich it and make it of relevance. Looking back along the road I have travelled, I see how much I have learnt and am truly grateful to those who helped, stimulated, challenged and encouraged me along the way.

In particular, I would like to thank my advisor, Pr. Jean-Didier Legat, for both encouraging me to undertake this challenge, for giving me incredible resources and freedom with which to pursue it, and for his unwavering confidence and support throughout.

I would also like to thank Pr. David Bol, who initiated the chip design projects I had the opportunity of working on that gave me invaluable experience in the tape-out process and technological aspects, and whose support, insight, and challenges have forced me to become, I hope, a better scientist.

I am also grateful to Pr. Kaushik Roy, with whom I worked at Purdue during the autumn of 2012. I have often had cause to profoundly respect his wisdom and insight, and working together truly widened my horizons. The time I spent at Purdue will forever remain a cherished memory.

I would also like to thank the distinguished members of my jury, Pr. Marc Belleville, Pr. Andreas Burg, Pr. Denis Flandre, and Pr. François-Xavier Standaert for their encouragement, valuable feedback, and all their work associated with this thesis.

I am also grateful to the CETIC, Walloon region of Belgium and the European fund for regional developments for funding this work.

Lastly, I would like to thank my family, friends and colleagues for the encouragements, the horizon-broadening discussions, and for all the fantastic moments we shared. Angelo Kuti, Julien De Vos, Guerric de Streel, Sébastien Bernard, and François Stas, the office would not have been the same without you. Adeline Decuyper, Emilie Renard, Maguy Trefois, Nicolas Boumal and Romain Hollanders, who have allowed me to glimpse at the mathematical beauty of the world, and Pierre-Antoine Haddad, Aline Emplit, Cédric Verleysen, Thomas Wallewyns and so many others for all the amazing times. And thank you to the administrative team, Viviane Sauvage, Anne Adant, Isabelle Dargent and Brigitte Dupont for their valuable help.

François.

ABSTRACT

In the context of the Internet-of-Things (IoT), the number of Wireless Sensor Nodes (WSNs) is set to explode. Limited in their autonomy by the energy they can store or harvest, these nodes must ideally consume as little power as possible in order to maximize their lifespans. Since the sensor and wireless link are the largest energy drains, a good system-level optimization strategy is to include a processor to filter the data and intelligently manage the node, keeping it in sleep as long as practicable.

The design of such a processor is the research topic of this thesis, with a focus on performance and low power consumption since the overhead of such a processor in the system must be small. In particular, the following questions were asked;

- Can a processor provide sufficient computing performance to carry out high-level tasks and signal processing, specifically the on-board analysis of sensed image data, whilst operating within the constraints of an energy-harvesting system?
- Beyond traditional low-power design techniques, are there alternative circuit design strategies that could yield increased processing speed in low-power systems?
- How can processors be made to operate efficiently in spite of the varying workloads typical of such devices?

This thesis aims to answer these questions through three distinct contributions. Firstly, a custom-designed MIPS-1 instruction set compliant 5-stage pipelined 32-bit CPU was designed and, along with the top-level assembly of the associated System-on-Chip (SoC), taped-out. It is able to achieve over 50MHz operation at 0.37V, and is notable for its use of a dedicated multiply and divide (MAD) unit that operates in parallel with the main execution branch, and its introduction of a software-controlled variable-width pipeline throughout. This enables the processor to handle either complex 32-bit data, or operate in SIMD mode on 2×16-bit or 4×8-bit data. Interaction with an 8-bit sensor peripheral thus provides significant performance improvements. Overall, the processor achieves 7.7μW/MHz operation at 50MHz, with a 1.55μW sleep mode, thereby improving on state-of-the-art designs.

With this chip it became clear that good design practices and careful attention to low-power issues could result in a very economical processor. However, additional improvements are required in order to achieve performance beyond that and the benefits provided by technology scaling. This thesis therefore introduces pseudo-synchronicity, an orthogonal circuit-level optimization technique

that allows the acceleration of combinatorial circuits beyond the performance generally achievable by conventional synthesis and timing closure, by exploiting the data-dependent delay variations inherent in such circuits. Through the automatic insertion of transition detectors within the target circuit, the progress of operations underway can be monitored and prematurely completed, thereby increasing the operation speed from the worst towards the average case. In addition, a synthesis flow is proposed to increase the proportion of fast paths, thereby increasing the technique's impact. Applied automatically to a series of benchmark circuits, synthesis results show it to achieve an average performance increase of 29% over conventional synthesis, which must ensure deterministic timing closure. In spite of a power 21% overhead caused by the addition of the hardware required, this increased speed allows shorter wake periods and is thus able to lower the circuit's overall active energy by 14%.

Lastly, this thesis observed traditional workloads for WSNs and noted that in addition to bursts of high performance, for which the traditional sleep-wake model of processors was well suited, they also often needed continuous operation for monitoring tasks, for which current designs are badly suited. A dual-processor system is therefore presented that is able to operate efficiently in either high-performance burst or continuous monitoring modes. Software running onboard can transparently switch to a workload-optimized core during execution in order to best match the application and tasks being processed. In doing so it can either provide a lower-power performance for time-indifferent tasks, and provide flexible continuous monitoring capabilities during idle periods between processing bursts, or a DSP-capable core for data intensive tasks.

ACRONYMS

AES	Advanced Encryption Standard
ADC	Analog to Digital Converter
AOP	Always-On Peripheral
AOR	Always-On Register
AFS	Adaptive Frequency Scaling
AVS	Adaptive Voltage Scaling
BAN	Body-Area Network
BL	BootLoader
BPS	Bits Per Second
CAD	Computer-Aided Design tools
CIS	CMOS Imaging System
CISC	Complex Instruction Set Computer
CMOS	Complementary Metal-Oxide Semiconductor
COTS	Commercial, Off-The-Shelf
CPF	Cadence Power Format
CPR	Critical-Path Replica
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CT	Clock Tree
D$	Data Cache
D2D	Die-to-Die variations
DIBL	Drain-Induced Barrier Lowering
DMEM	Data MEMory
DMA	Direct Memory Access
DSP	Digital Signal Processing
DVFS	Dynamic Voltage and Frequency Scaling
EOV	Energy-Optimal Voltage
FD-SOI	Fully Depleted Silicon-On-Insulator

FF	Flip-Flop
FFT	Fast Fourier Transform
FIFO	First-In, First-Out (a memory buffer interface system)
FIR	Finite Impulse Response filter
GPIO	General-Purpose Input Output
HDL	Hardware Description Language, eg: Verilog, VHDL
HDR	High Dynamic Range
I$	Instruction Cache
IC	Integrated Circuit
IMEM	Instruction MEMory, as as PMEM
IoT	Internet-of-Things
IRQ	Interrupt (request)
JTAG	Joint Test Action Group (a debugging and programming interface)
LER	Line Edge Roughness
LDO	Low Drop-Out (regulator)
LS	Level Shifter
LZR	Line Width Roughness
MAD	Multiply and Divide (Unit)
MC	Monte-Carlo simulation
MEP	Minimum Energy Point
MEMS	Micro Electro-Mechanical Systems
MMMC	Multi-Mode, Multi-Corner
MOS	Metal-Oxide Semiconductor
MOSFET	Metal-Oxide Semiconductor Field Effect Transistor
PCB	Printed Circuit Board
PMEM	Program MEMory, same as IMEM
PMU	Power Management Unit
POR	Power On Reset
PVT	Process, Voltage and Temperature
PWM	Pulse Width Modulation
RC	Resistor-Capacitor
RDF	Random Dopant Fluctuations
RF	Radio-Frequency

RFID	Radio-Frequency IDentification
RISC	Reduced Instruction Set Computer
RO	Ring Oscillator
ROM	Read-Only memory
RTL	Register Transfer Level
SDC	Synopsys Design Contraints
SDP	Structured DataPath
SFR	Special Function Registers
SIMD	Single-Instruction, Multiple-Data
SoC	System-on-Chip
SOI	Silicon-On-Insulator
SPI	Serial Peripheral Interface (a communication protocol)
SSTA	Statistical Static Timing Analysis
TD	Transition Detector
TDC	Time-to-Digital Converter
TSV	Through-Silicon Vias
UART	Universal Asynchronous Receiver Transmitter (a communication protocol)
ULP	Ultra-Low-Power
ULV	Ultra-Low-Voltage
VLIW	Very Long Instruction Word
WDT	Watchdog Timer
WID	WIthin-Die variations
WSN	Wireless Sensor Node

INTRODUCTION

Fifty years ago, mankind witnessed something incredible: the landing of a man on the moon marked one of the greatest achievements of science and technology. Incredible perseverance, the will to solve all problems one by one was key to making the voyage possible. Among the many technological innovations of the Apollo program, perhaps one of the most significant was the Guidance and Navigation computer due to its small size, being one of the very earliest computers to be built using integrated circuits. Fifty years on, computers are both far more powerful and more integrated, with even the simplest pocket calculator able to outperform the Apollo computer, a fitting testimony to the pioneering spirit and engineering determination that inspired the original.

Enabling this incredible evolution of computers is the scaling of features capable of being integrated onto a silicon chip in a trend known as Moore's Law [1], allowing ever more functionality for ever-decreasing cost and size. In recent years, however, there has also been a diversification in end-use of the chips produced, moving away from high-power, high-performance devices such as desktop computers and servers to distributed, miniature, low-power wireless sensor networks. This trend is set to become increasingly important with the advent of a vision known as the Internet-of-Things (IoT), where data is to be accumulated from more sources and interact more intensely to enable new high-level, intelligent applications.

Enabling the IoT, and indeed vital for its operation, is the underlying array of Wireless Sensor Nodes (WSNs).

I.1 STRUCTURE OF A WIRELESS SENSOR NODE

A wireless sensor node is an embedded system that sends sensor data to a control node over a wireless link, or receives instructions to perform some sort of actuation. For the IoT vision, where these WSNs are to be deployed ubiquitously and in large numbers, these must be small, standalone, and easy to maintain.

Figure I.1 shows the general structure of a typical WSN. Being size-constrained, the on-chip component integration is generally maximized, using external components only where necessary. The power source is dictated by the environment into which the sensor node is placed, but will often be of limited magnitude; in order to maximize autonomy and limit maintenance, such as the replacement of batteries, the system must be designed to function for as long as possible off a battery, or even directly employ energy available in the environment through energy harvesting methods [2]. It is therefore the role of the on-board Power Management Unit (PMU) to maximise the power that can be obtained

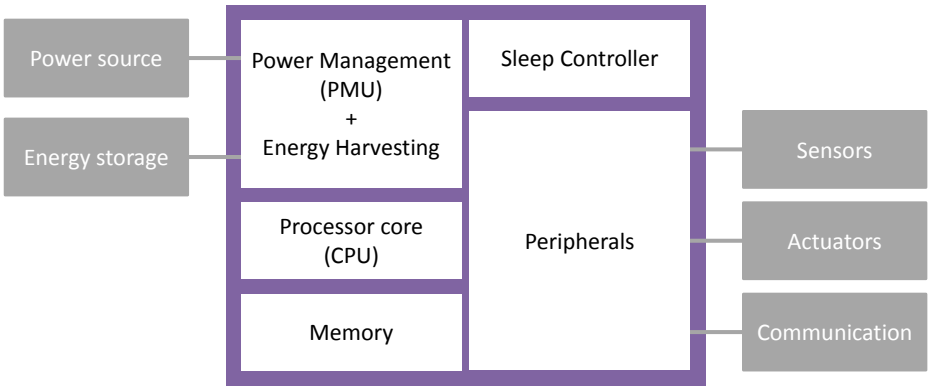


Fig. I.1.: Typical architecture of a Wireless Sensor Node, showing the on-chip components (blue), and external interfaces (grey).

from the source, and manage any external reserves available, traditionally a small battery or even simply a large capacitor.

Also integrated externally are the sensors and actuators, and a communication interface such as a radio link to interact with nearby nodes and report back to a controller. These represent the useful payload of the sensor node, gathering and sending the raw data, but also represent the most power-intensive resources. Indeed, without pre-processing, it is very difficult to reduce the operating power down to manageable levels.

Several research directions are therefore possible to make these devices a reality; either the energy harvesting aspect needs to be improved in order to provide sufficient energy to power everything for the device lifespan, or the sensors and radios must be optimized to consume less. While these optimizations are significant, most WSNs also employ a system-level optimization, whereby the data is pre-processed on-chip before transmission. By adding a processor core, the node itself will have a power overhead, but in doing so the power-hungry components can be intelligently managed and on-board digital signal processing (DSP) can be used to minimize their active time, thereby bringing a significant power reduction to the overall system.

Having on-board processing capabilities also enables high-level functionality, such as the ability to manage a complex wireless network and prevent conflicts with other nodes, and features such as data security. It can also enable entropy coding, compressed sensing [3, 4], feature extraction and classification [5], or event detection, further reducing the amount of data that needs to be transmitted.

However, for this to be optimal, maximising a processor's signal processing and control functionalities with respect to power consumption becomes a priority. Optimizing the processor core for this case is the focus of this work.

I.2 THE IOT VISION

It may be asked why any of this is necessary, what benefits such a technological investment can bring. After all, we already live in a technologically advanced society that is feeling the pressures of information overload, and adding more data sources seems redundant if not foolhardy. The IoT, more than simply an expansion of existing technology, is a paradigm shift in the way such technology is used. Even in the early days of computing, researchers at the famed Xerox PARC realised that users of technology were being forced to interact with it through very limited channels, such as screens and keyboards, in contrast to how they interacted with everyday objects [6]. If anything, this has become even more relevant in the modern day. Instead of interacting with technology through limited screens and keyboards, the IoT aims to interact with the human environment directly. The aim of the IoT is therefore to bring technology into the human environment, rather than forcing humans into the technological environment.

Another facet of the IoT is the strange realization that in fact more data is needed to provide meaningful information in large-scale systems. Whereas previously the main limitation was computational capabilities of dealing with the data, there exists today the ability and the necessity of dealing with more data in order to refine the underlying models and improve analysis and predictions. Through more advanced sensing techniques or vast arrays of sensors, the IoT vision aims to improve human lives by collecting and cross-correlating more data sources to form a more reliable and informative outcome.

This vision is also relevant at the individual level, where a myriad of sensors and networks seamlessly and transparently gather, analyse and transmit information to and from the user. Be they a short-range Body-Area Network (BAN), allowing different sensors embedded directly into a user's body to interact and communicate with the controllers in the home [7]. Or the home automation systems currently being considered, which must centralize information from a myriad of sources, such as RFID tags [8] in closets and sensors on the doors and windows, and compile them into relevant facts. Or a city-scale network that coordinates traffic, or even large-area networks [9] composed of randomly-positioned (air-dropped, for example) sensors for applications such as detecting forest fires before they are too large to contain. By cross-referencing all this information, by allowing devices to sense and act on the human environment, the IoT paradigm is both empowering and liberating.

I.3 PURPOSE OF THIS THESIS

While other works [10, 11, 12] focused on ultra-low power hardware designs describe system-level optimizations that can be brought to the node, this thesis concentrates on the processor-level implementation and optimization techniques. Set in the IoT vision, it aims to improve the overall system power consumption of

typical WSNs through the design and optimization of micro-controllers enabling high-level signal processing and complex power management strategies.

Specifically, this thesis aims to answer the following questions;

- Can a processor provide sufficient computing performance to carry out high-level tasks and signal processing, specifically the on-board analysis of sensed image data, whilst operating within the constraints of an energy-harvesting system?

It is clear that a system running off harvested energy will have very little power at its disposal, often making it difficult to send raw data directly over the wireless link. Digital signal pre-processing and on-board decision-making can in theory considerably lower the required power, but the question remains as to whether such processing is possible under the stringent power constraints set by the power source. To determine the feasibility of such a system, an advanced 32-bit processor was designed and taped-out, employing state-of-the-art technological implementation techniques and capable of processing 32-bit data with an optional SIMD mode, in order to achieve significant energy savings over regular designs.

- Beyond traditional low-power design techniques, are there alternative circuit design strategies that could yield increased processing speed in low-power systems?

It is a well-known fact that low power can be achieved through reduction of the processing speed and capabilities. Beyond the techniques that can enable sufficient processing to achieve the previous objective, there remains the question of whether there are any alternative circuit design strategies that could achieve increased processing speed specifically in the context of low-power systems. Pseudo-synchronicity, an orthogonal circuit-level optimization technique is therefore proposed.

- How can processors be made to operate efficiently in spite of the varying workloads typical of such devices?

Much of the work required from a sensor node is data accumulation, a relatively low-power and time-insensitive task, interspersed with high-performance bursts of processing and computation, which are significantly different design targets for such systems. The question is thus whether a processor can be designed to manage both modes effectively, without undue inefficiency in either. A dual-processor system is therefore presented that is able to operate efficiently in either high-performance burst or continuous monitoring modes.

I.4 THESIS OUTLINE

In order to answer the questions outlined above, the remainder of this thesis is organised as follows.

Chapter 1

Chapter 1 presents a general introduction to the techniques used for low-power design, and the state of the art processors currently available in research. Rather than aiming towards high performance, this class of micro-controllers focuses on low power consumption in order to fit within the constraints of energy-harvesting systems. The underlying technology is presented, as well as features of the 65-nm CMOS manufacturing node used to implement later designs, along with the challenges and pit-falls that low-voltage brings. This chapter also presents the design metrics that will be used to evaluate the low-power core developed as part of this thesis.

Chapter 2

Chapter 2 presents the work done for the design and implementation of the Bellevue processor, a custom 50MHz 32-bit micro-controller running at 0.37V built on a 65-nm LP/GP CMOS process. It features a low-power 1.55μW sleep mode, and a variable-width SIMD pipeline and multiply/divide unit, achieving 7.7μW/MHz overall. The proposed micro-controller is designed to operate under the same stringent power constraints required by harvested energy operation, but with a significantly increased processing capability with respect to the state-of-the-art designs, enabling the processing of captured image data directly within the WSN. By allowing greater data processing abilities for the same low power, the WSN is able to analyse the sampled data on-chip for less energy than required for offloading to a remote node via a wireless link, thereby providing more features and better real-time reaction to events. For example, a typical image sharpening task running on-board consumes 5.4μJ and allows real-time data processing. Offloading via a wireless link would consume more, and not be able to offer the same throughput capabilities nor realtime analysis.

Chapter 3

Chapter 3 details a circuit optimization design strategy that enables operation acceleration within a combinatorial circuit beyond that generally achievable by conventional synthesis methods, by exploiting the data-dependent delay variations inherent in such circuits. Through the automatic insertion of signal transition detectors to monitor the progress of operations underway, calculations can be prematurely completed, thereby increasing the operation speed from the worst towards the average case. In addition, a synthesis flow is proposed to increase the proportion of fast paths, thereby increasing the technique's impact. Applied automatically to a series of benchmark circuits, synthesis results show it to achieve an average performance increase of 29% over conventional synthesis, which must ensure deterministic timing closure. In spite of a 21% power overhead caused by the addition of the hardware required, this increased speed allows shorter wake periods and is thus able to lower the circuit's overall active energy by 14%.

Chapter 4

Whereas traditional WSNs are optimized either for processing-intensive or monitoring applications, Chapter 4 presents a low-power, dual-core processor system that can function efficiently in both modes. Software running onboard can transparently switch to a workload-optimized core during execution in order to best match the application and tasks being processed. In doing so it can either provide a lower-power performance for time-indifferent tasks, and provide flexible continuous monitoring capabilities during idle periods between processing bursts, or a DSP-capable core for data intensive tasks.

Conclusion

In the conclusion, the main contributions of this thesis are summarized, and perspectives for future research and developments are outlined.

AUTHOR'S PUBLICATION LIST

Journal papers

JP1. D. Bol, J. De Vos, C. Hocquet, F. Botman, F. Durvaux, S. Boyd, D. Flandre, and J-D. Legat, "SleepWalker: A 25-MHz 0.4-V Sub-mm^2 7-μW/MHz Microcontroller in 65-nm LP/GP CMOS for Low-Carbon Wireless Sensor Nodes," in *IEEE J. Solid-State Circuits*, vol. 48, no. 1, pp. 20-32, 2013.

JP2. F. Botman, D. Bol, J-D. Legat, and K. Roy, "Data-Dependent Operation Speed-Up Through Automatically Inserted Signal Transition Detectors for Ultralow Voltage Logic Circuits," in *IEEE Trans. Very Large Scale Integr. Syst.*, 2014.

JP3. F. Botman, J-D. Legat, and D. Bol, "Ianus: a workload-optimized ULV dual-processor system for monitoring and data-processing applications in the Internet-of-Things," in *IEEE Trans. Very Large Scale Integr. Syst.*, 2014, *submitted.*

Invited tutorials and keynotes

ITK1. D. Bol, J. De Vos, F. Botman, G. de Streel, S. Bernard, D. Flandre, and J-D. Legat, "Green SoCs for a Sustainable Internet-of-Things", in *Proc. Workshop Faible Tension Faible Consommation (FTFC)*, 4 p., 2013.

Related conference papers

CP1. D. Bol, C. Hocquet, J. De Vos, F. Durvaux, F. Botman, D. Flandre, and J-D. Legat, "Design Techniques for Reliable Timing Closure in Ultra-Low-Voltage Logic SoCs," in *Proc. IEEE Subthreshold Microelectronics Conference*, 2011.

CP2. C. Hocquet, F. Botman, D. Bol, and J-D. Legat, "A near-threshold instruction cache with zero miss overhead time for dual- Vdd microcontrollers," in *Proc. IEEE Subthreshold Microelectronics Conference*, 2011.

CP3. F. Botman, D. Bol, C. Hocquet, and J-D. Legat, "Exploring the Opportunity of Operating a COTS FPGA at 0.5V," in *Proc. IEEE Subthreshold Microelectronics Conference*, 2011.

CP4. D. Bol, J. De Vos, C. Hocquet, F. Botman, F. Durvaux, S. Boyd, D. Flandre, J-D. Legat, "A 25MHz 7μW/MHz ultra-low-voltage microcontroller

SoC in 65nm LP/GP CMOS for low-carbon wireless sensor nodes", in *IEEE International Solid-State Circuits Conference*, pp.490-492, 2012.

CP5. F. Botman, D. Bol, and J-D. Legat, "Data-dependent operation speed-up through automatically-inserted signal transition detectors for ultra-low voltage logic circuits," in *Proc. IEEE Subthreshold Microelectronics Conference*, 2012.

CP6. F. Botman, J. De Vos, S. Bernard, J-D. Legat, D. Bol, "Bellevue: a 50MHz Variable-Width SIMD 32bit Microcontroller at 0.37V for Processing-Intensive Wireless Sensor Nodes", in *IEEE International Symposium on Circuits and Systems*, 2014.

CP7. D. Bol, G. de Streel, F. Botman, A. Kuti Lusala and N. Couniot, "A 65-nm 17-pJ/frame.pixel 0.5-V DPS CMOS Image Sensor for Ultra-Low-Power SoCs achieving 40-dB Dynamic Range," in *IEEE Symposium on VLSI Circuits*, 2014.

"640K ought to be enough for anybody"

Widely, though incorrectly, attributed to ***Bill Gates, 1981.***

CHAPTER 1

FUNDAMENTALS OF LOW-POWER PROCESSOR DESIGN

This chapter presents a general introduction to the techniques used for low-power circuit design, with a special focus on the state-of-the-art low-power processors. Whereas traditionally the design of a processor system is geared towards high-performance, these have energy efficiency as their primary focus, and aim to be employed in wireless sensor nodes running off harvested energy. Features of the underlying technology is presented, specifically those of the 65-nm CMOS manufacturing node which is primarily used by this work. This chapter also gives an overview of the design metrics and problems encountered when designing such low-power systems, giving an insight into the background setting of later chapters.

Contents

Associated author publications

JP1. D. Bol, J. De Vos, C. Hocquet, F. Botman, F. Durvaux, S. Boyd, D. Flandre, and J-D. Legat, "SleepWalker: A 25-MHz 0.4-V Sub-mm^2 7-μW/MHz Microcontroller in 65-nm LP/GP CMOS for Low-Carbon Wireless Sensor Nodes," in *IEEE J. Solid-State Circuits*, vol. 48, no. 1, pp. 20-32, 2013.

CP2. C. Hocquet, F. Botman, D. Bol, and J-D. Legat, "A near-threshold instruction cache with zero miss overhead time for dual- Vdd microcontrollers," in *Proc. IEEE Subthreshold Microelectronics Conference*, 2011.

CP4. D. Bol, J. De Vos, C. Hocquet, F. Botman, F. Durvaux, S. Boyd, D. Flandre, J-D. Legat, "A 25MHz 7μW/MHz ultra-low-voltage microcontroller SoC in 65nm LP/GP CMOS for low-carbon wireless sensor nodes", in *IEEE International Solid-State Circuits Conference*, pp.490-492, 2012.

In the IoT, nodes must have powerful processors in order to provide data processing capabilities, efficiently manage the wireless network, for sensing and actuation procedures, and high-level tasks. However, the power consumption of the processor core itself is of concern.

Processors come in all shapes and forms, but as a general rule the simpler the instruction set, the lower the logic power consumption but also the longer the task will take to complete, and possible overhead from increased memory access. Although the choice of architecture is ultimately dependent on the target application, a number of design-level optimization strategies can be employed across the board.

1.1 CIRCUIT DESIGN FOR ULTRA-LOW POWER

It is well known that the total power of a circuit can be modelled as $P_{tot} = P_{leak} + P_{switch}$. P_{leak} is the power lost through transistor leakage and factors independent of activity. The switching power P_{switch} is that which results from operation. The latter is largely due to the switching of the input capacitors of each transistor the driving gate is electrically connected to, temporary short-circuits that occur in CMOS gate designs, signal propagation along metal connections and associated capacitance and cross-talk, and other logic transition-related factors. These are proportional to the switching activity and frequency, and specifically, in the case of losses due to the input capacitors, can be modelled as $P_{switch\_C_L} = \alpha_F \times C_L \times V_{DD}^2 \times f$, where α_F is the switching activity, C_L the load capacitance, V_{DD} the supply voltage and f the operating frequency of the overall circuit.

Since in nominal conditions the switching power dominates, efforts to reduce the power consumption of a circuit will therefore attempt to minimize each component of it.

Activity reduction

In an attempt to reduce the activity factor of circuit sections where switching is not required at certain instants, the clock can be gated [13] or data lines can be isolated through operand isolation [14] to limit redundant switching. Clock gating prevents a clock signal from activating a set of registers and cause these to switch, and operand isolation is the same principle applied to data propagation. These techniques, combined, are very effective and are widely supported by commercial design tools, which are able to automatically identify groups of cells that can benefit from gating in spite of the hardware overhead, and insert gating cells accordingly.

Another obvious strategy is to duty-cycle the processor and power gate it when it is no longer actively needed (see Fig. 1.1). In order to achieve this, a sleep controller peripheral can shut down the power to the processor, and control the wake cycle based on internal or external inputs. This can, for example, be combined with an RF wake-up receiver [15] to respond to wireless events, or simply wake up periodically based on a timer or on critical sensor events.

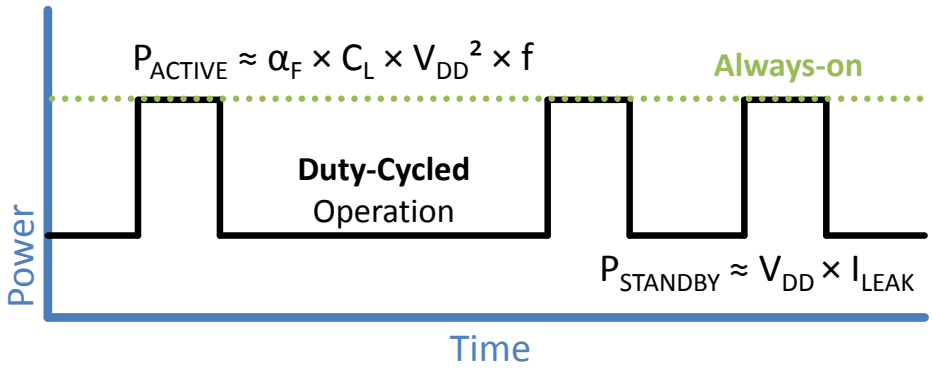


Fig. 1.1.: Typical system power consumption displaying active and standby power, and demonstrating duty-cycled operation.

Voltage scaling

Since the switching power P_{switch} is quadratically dependant on the supply voltage, another way to lower the power significantly is to reduce the operating voltage of the circuit. In fact, the voltage can actually be reduced to below the transistor threshold voltage, where so-called sub-threshold CMOS designs have been shown to operate at down to 36mV [16]. However, this is not always a wise optimization strategy [17].

The main problem with voltage scaling is that whereas $P_{switch} \propto V_{DD}^2$ decreases quadratically with the supply voltage, it also increases the logic delay [18, 19]. So by reducing V_{DD}, the performance achievable is also affected [20].

Similarly, although the leakage power also scales, it does so less fast thereby causing its relative contribution to become more significant at low voltage and operating frequency [21]. This results in a trade-off; on the one hand at higher voltages, high performance can be achieved at the cost of high switching-dominated energy per cycle. On the other, at low voltages, the leakage becomes increasingly dominant. Optimising for the lowest overall energy per cycle yields an energy optimal voltage (EOV) [21] at which it is best to operate, usually at ultra-low voltages (ULV).

Conversely, it is possible to look at this from a performance perspective and find for a given operating frequency the optimal supply voltage that will enable it. This is the minimum energy point (MEP) [22, 23], where for a given technology choice and operating conditions the energy per operation is at a minimum.

It is also possible to use reduced voltage swing in localised signals, as is quite frequent in SRAM memory designs. By reducing the voltage swing the resulting power consumption of the driver and sink gates will be much lower and, crucially, the required voltage level will be achieved much faster, enabling a performance increase. However, particularly in the case of ULV designs, this will further reduce the voltage margins and thus potentially degrade the stability of these signals.

Overall, voltage reduction is an effective way of decreasing the power consumption of a circuit, albeit at the cost of performance. However, by combining voltage scaling to ULV with acceleration techniques it is possible to maintain performance for the targeted class of devices while achieving low power overall.

Technology scaling

As per Gordon Moore's self-fulfilling prophecy almost 40 years ago [1], the IC device count has been increasing phenomenally through Dennard scaling [24], thereby reducing gate capacitances C_L and greatly improving the switching power at each technological node. Specifically, industry-guided scaling theory typically aims to achieve a 30% reduction in gate delay and dimensions [25, 26] at each new node.

However, because of a shorter channel and subthreshold current there results an intrinsic leakage increase, which in itself becomes an important optimization aspect [27], especially at low frequencies where leakage integrated over time becomes significant with respect to the switching power.

In order to compensate for this, it is possible to modify the underlying transistors' threshold voltage V_T, which has a localised impact on the performance and leakage of the underlying transistors. The 65-nm CMOS technology, for example, features several flavours, core oxide thickness GP and LP, and doping levels HVT, SVT and LVT [1].

In a typical micro-controller context, the different underlying blocks feature differing activities, and so it is possible to differentiate their physical implementation using these flavours in order to maintain performance and minimum energy at the design point [28, 29, 30].

For example, in SleepWalker [31] there are two main activity regions. The main processor core features comparatively high activity and a clock frequency of 25MHz, so the designers chose to implement this using GP SVT MOSFETs which best fit the speed requirements at ULV (0.4V). Similarly, the peripherals and low-frequency components display much lower activity, and so are implemented using LP HVT MOSFETs at nominal 1V supply voltage. By taking advantage of technology scaling and the inherent power benefits associated with it, and combining this with a suitable choice of transistor variant and dual supply voltage, the operational requirements of the system can be well matched while maintaining a power advantage [28].

Memories

Memories are required in a micro-controller system for storing both data and instructions, enabling the processor to perform a given function. Particularly in the case of the IoT, a large memory capacity is required for data logging, more advanced signal analysis, and complex communication stack. While external (off-

[1]Capabilities of the ST Microelectronics foundry, see `http://cmp.imag.fr/aboutus/slides/slides2007/04_KT_ST.pdf`

chip) memories can be used, their access is generally very costly in terms of power, latency and system (PCB) manufacturing price.

While the research community has been intensively researching ULV SRAM memories [32, 33], some designs use foundry-optimized SRAM bit arrays in a nominal power domain in an attempt to maximize density. Their access, however, is therefore costly in terms of power due to the level-shifting necessary and the switching in the nominal power domain. However, whereas historically memories have been considerably slower than the processor itself, the voltage differential now enables both to have similar performance. An accelerating cache is therefore no longer necessary, but if implemented in the ULV region using low-power registers can be used instead to reduce the access frequency to these memory blocks and thereby save power [34]. The use of a cache in ULV designs therefore no longer serves the purpose of reducing access times, but to minimize the nominal-power domain power overhead.

SleepWalker [31, 35] takes this one step further through the design of a ULV cache that features a zero-miss delay penalty, meaning that overall memory access time from the perspective of the processor core remains the same, whether the requested data is cached or not.

The question of cache size is a design consideration, as is the choice to use memory banking, but varies significantly on a per-application basis. Dedicated simulations are needed to assess the impact of the various arrangements on the overall power consumption [36], taking into account the possible increased access time. And while having multiple memory banks may lower the access power costs through lower internal capacitances, the decoding overhead and overall increased leakage must also be considered.

Lastly, there is the question of volatility. Since non-volatile memory is typically difficult to embed into the system while maintaining a reasonable power budget, micro-controllers in wireless sensor nodes usually require their program to be given to them on activation. This can be done manually, through dedicated circuitry and an external flash memory, or through a dedicated read-only memory (ROM) that is programmed at design time. The node will then typically fetch more complete programs from the network [37].

Hardware acceleration

Through a careful mixture of technology scaling, voltage scaling, and transistor flavor choice, the processor is able to rapidly perform whatever operation is required of it. In fact, given the high active power, it is generally best to perform the task as fast as possible, then return to a power-gated sleep state. In this way, tasks can be duty-cycled based on usage, and the best power option obtained.

With this in mind, the advantage of having dedicated hardware to accelerate frequent tasks and return the processor to its idle state as fast as possible becomes significant. A processor may benefit from a dedicated hardware multiplier if the application requires frequent multiplications. Similarly, since cryptography will be a necessity in the IoT, a dedicated module may prove to be a necessity. Also, in applications such as biomedical devices, digital signal processing (DSP)

hardware such as FFTs and FIRs can be beneficial [38]. The same is true for many other types of applications, for instance those based on machine-learning, or in baseband telecommunication systems.

As a result, the design of the micro-controller system must not only take into account features of each composing block, but also how these blocks will interact with each other in the context of the targeted application. Fundamentally, the choice of processor core and features will be gleaned from a hardware/software partitioning analysis, where potential gains can easily be identified and a balance struck between the increased power consumption, and the longer sleep time afforded to the system by its inclusion.

Parallel execution

Although reducing the supply voltage of the processor core is an effective way of lowering the power consumption, it also negatively impacts the processing speed possible under those conditions. Technology choice can to some extent mitigate this, but in some cases this is not sufficient. Similarly, adding pipeline stages can increase throughput, but thereby also reduces the path lengths and introduces overhead from an increased number of registers.

For high-performance, desktop-class processors hitting the power density barrier, the solution was to migrate from single to multi-thread computation, and dispatching operations across multiple cores. This can also be done in ULV systems, with the added advantage that unused cores can be power gated when not in active use. In this way, and through careful design of the algorithms running on the system, tasks can be executed in parallel, and increase the effective processing speed whilst maintaining an excellent energy profile.

Similarly, whereas traditional processors sometimes implement single instruction multiple data (SIMD) instructions in order to increase performance, the same features can be used in ULV systems [39], with a view to utilize resources more efficiently and, ultimately, consume less energy by completing tasks faster.

CPU architecture choice

There are many types of processor cores, each with their own special features and unique aspects that are useful in the context of ultra-low power systems. The MIPS architecture, for instance, has a large number of registers available for use, thereby limiting memory access, and a branch delay slot that in most implementations removes the need for a branch predictor. The SPARC architecture on the other hand uses register windows to optimize procedure calls, making interrupt handling and operating system support particularly efficient. The ARM architecture implements the Thumb instruction set, which makes a subset of operations available in reduced instruction width, thereby significantly reducing memory access costs and improving code density and thus requiring a smaller instruction memory. PIC and AVR are low-power 8-bit micro-controller families designed for simple, embedded applications requiring little processing power. And as a balance between these simple 8-bit systems and 32-bit architectures, TI's MSP430 processor claims a low-power advantage through its analog

features and fine-grained power management system. There are also some more exotic architectures [40, 41, 42, 43].

There are also attempts to automatically generate a processor instruction set that best matches an application [44]. The main advantage of processors over dedicated ASIC circuits is their flexibility, allowing changes to the software without major circuit re-design. However, such automatically-optimised instruction sets often tend to be overly specific to the target application, and therefore reduces the resulting flexibility. Also, due to the integrated nature of the instruction set in the design of the processor, such systems will either not allow many modifications, or result in a sub-optimal processor design. In many ways, therefore, fixed instruction sets have the advantage for ULV, but selecting one is a difficult choice.

As a general rule, however, the simpler a processor's instruction set, the less power the core will consume, but the greater the number of cycles required for a task to execute.

In ULV systems, the leakage is significant but can mostly be limited through power gating during sleep, so the key factors in designing ULV systems is limiting active-state switching activity and reducing overall processing time. This makes the choice of architecture highly dependent on the target application. For instance the choice of a 16-bit architecture over a 32-bit architecture is inspired, if the data to process is of that length and the memory range to access is small. On the other hand, a 32-bit or even 64-bit architecture may be necessary in order to seamlessly interface with servers and more complex control nodes in the IoT. A smaller datapath width will induce less switching when accessing the various memories, but this effect may be offset by more complex computing behind the scenes necessary to handle large numbers, or a greater number of requests. Similarly, is a 32-bit instruction set truly necessary, and does a reduced decoding complexity offset the higher instruction memory access costs? There is no definitive answer, since the application requirements are so different, but in each case the processor designer must keep these issues in mind when determining the best solution to use.

Algorithm optimization

Although not strictly related to microprocessor optimization, the way in which it is used can have a significant impact on the overall power performance. Algorithms are implemented to efficiently implement a given task, but often with little thought about the power implications it may have on a system. For instance it may require access to both the memory blocks and a hardware accelerator simultaneously, whereas distributing these in time will have a positive impact on the peak current and, therefore, the power converter requirements and overall performance. There is thus a need to adequately study the scheduling aspects of the applications running on the system [45].

This is typically a task that is very hard to efficiently perform at the hardware level, since there is no information available at that level as to the long-term usage and actual requirements of the application. This is therefore best done at

the software level, but work has been done to implement power-aware compilation [46] that will perform this analysis automatically.

Similarly, data representation should be adequately considered. On the one hand a more efficient memory storage of the data will result in fewer requests to access it, and also conserve valuable memory space. And features such as gray encoding can limit switching in the busses, which will ultimately lead to decreased active energy utilization.

Summary

This section has briefly touched on the techniques used to decrease microcontroller power consumption, from a design perspective. Technology choice and supply voltage are crucial enabling factors, and the use of advanced nodes brings significant benefits. Execution speed is an important aspect since, combined with duty-cycling, it will allow the processor to remain inactive for longer periods and thereby achieve a power advantage. This execution speed can be improved through a variety of ways, such as hardware accelerators and operation parallelization. Memory access, and data movement more generally, can be reduced through localized caches and CPU architecture choice, but in the final analysis a system-wide power reduction can only stem from a concerted effort across all hardware blocks and co-design of the associated software.

1.2 ULV CIRCUIT IMPLEMENTATION CHALLENGES

Designing a circuit to function at ULV brings associated power benefits, as seen in the previous section. There remain, however, challenges from a physical implementation perspective, due to the lower voltage margins and inherent variability associated with the manufacturing process for such an advanced technology.

This section details some of the problems that are frequently encountered, and techniques used to mitigate their effects.

PVT variation compensation

One of the main concerns are inter-die process, voltage and temperature (PVT) variations [47, 48, 49]. As part of the manufacturing procedure, there exists a process variation from die to die (D2D) that causes the underlying circuits to be implemented slightly differently. In addition, the differences in voltage and temperature where the devices will be used also causes uncertainty at design time.

In order to model these variations, chips are conventionally designed using a multi-mode, multi-corner (MMMC) analysis. The slow (SS die and low temperature at ULV) corner represents the worst-case performance the device is designed to cope with, and the fast (FF die and high temperature at ULV) corner represents the best case. In addition, some designs also consider the impacts of transistor ageing and electron-migration that affect the circuit over the long-term. Such an MMMC analysis is widely supported by the CAD tools, and

enable them to obtain not only timing closure, but also resolve setup and hold violations.

Conventionally, timing and voltage margins are inserted in order to achieve guaranteed functionality throughout the range of supported conditions. These, however, add an overhead to the system by not allowing the voltage to be lowered to the optimum, and represents wasted performance.

An alternative technique often employed is to compensate for these effects adaptively, observing the actual conditions and ensuring that the circuit operates within design specifications. One proposed approach is through the use of an adaptive voltage system (AVS) [50] that adapts the supply voltage V_{CC} to ensure appropriate performance. Such AVS units [51, 52] observe the on-chip conditions, for example through the use of a calibrated ring-oscillator (RO) in the target domain [11, 31, 51] in order to tune the voltage output optimally, with a configurable safety margin.

Conversely, this can also be done by adapting the operating frequency of the target block [53] using an adaptive frequency scaling (AFS) system. Along with AVS units, these are specific implementations of the general field of Dynamic Voltage and Frequency Scaling (DVFS) systems, which in addition to being able to adapt the operating conditions on-chip, do so adaptively according to what is actually encountered by the system during use.

The compensation condition detection mechanism could also potentially be calibrated by sensing the error rate encountered within the circuit's critical paths and ensuring optimal operation. For resolution-loss architectures [54] this can be done by determining the acceptable error rate, or, if errors cannot be tolerated, by using error-correction techniques such as Razor [55] and ensuring minimum correction delay. In this way, PVT variations can be compensated and, crucially, allows the target block to be designed for only typical (TT die at room temperature) operating conditions, thereby significantly reducing the resulting margins [31].

Random Dopant Fluctuations

Another effect that results from the manufacturing process is random dopant fluctuations (RDF) that occur intra-die (WID). In advanced technologies, the device size is so small that it becomes particularly sensitive to the dopant chemistry, which is very difficult to control reliably. Similar effects [56, 57] include Line Edge Roughness (LER) and Line Width Roughness (LWR) [58], which result from the sub-wavelength lithography process, and wider-ranging gradient effects. As a result, in spite of designing a transistor with well-known and well-characterized V_{th} properties, the resulting device might suffer from unexpected deviations from the reference model. It has even been proposed [59] to exploit this as a unique and unreproducible encryption key embedded within chip.

RDFs, by their very nature, cannot easily be compensated for by the AVS systems described previously, due to their localised nature. As a result, margins must be taken, or fault-tolerant techniques must be used [55]. There also exists statistical analysis methodologies that aim to predict these effects [60] and allow

the designer to compensate for them, but these are highly resource-intensive and therefore tend to take prohibitively long to perform on a complex circuit such as a processor.

Alternatively, if the critical paths are long, their impact tends to be averaged out [61, 51, 62], which is generally the case for processor systems with reasonable pipeline depths typical of ultra-low power micro-controllers. This means that smaller margins can be used, resulting in an acceptable compromise.

Cell library optimization

Because of the noise and variability sensitivity of devices in the subthreshold regime, standard cells designed for nominal voltage and re-characterized for ULV may perform poorly, or have yield issues. Similarly, these cells may be ideally suited to low-power operation. The construction of an appropriate cell library is thus an important aspect when designing for ULV.

The first problem is a stability issue. Because of the reduced voltage margins inherent in ULV systems, it is possible that noise or transistor variations cause signal loss through the mis-detection of a logic level. Many works have therefore produced custom cell libraries, using Monte-Carlo (MC) to simulate the impact of noise on the reliability [63, 64]. Also, although transistor stacking improves the subthreshold leakage current issue [65], it also tends to induce instability at ULV due to the reduced voltages seen by individual transistors. As a result, transistor stacking is generally limited to 3 [63, 31].

In addition, the transistor gate length can be increased (in 65-nm this would lead to an increase to ~80nm), which lowers the subthreshold leakage current, albeit at the cost of area and performance. In SleepWalker [31], this was found to only increase the area by less than 5%, but allowed the MEP to be reduced by ~15% through an improved subthreshold voltage swing and drain-induced barrier lowering (DIBL) effect reduction [66, 67].

In a similar spirit, it is also possible to re-characterise the nominal voltage foundry-provided cell library for ULV, and select only the appropriate cells, rather than having to provide a full-custom implementation of the entire library, which constitutes a considerable amount of work. It is also possible to limit the initial selection to low-driver strength cells, in order to prevent the excessive use of high-strength cells in a futile attempt to achieve stringent timing requirements during CAD synthesis [68]. Where these are specifically required, in order to drive high fan-out nets or the clock tree, they can be inserted through dedicated, supervised commands during the flow.

For this reason, the optimization of the standard cell library for the target system is often of great importance, since its misuse can easily incur manufacturing issues, yield problems, or decreased stability. Whereas custom-designed cell libraries can be designed for ULV use, careful selection of cells and re-characterisation of a nominal voltage library is also possible. There have also been attempts to replace cells in-circuit; [69, 63] used a post-layout Monte-Carlo analysis in order to detect and correct hold-time violations and determine de-

lay variability in short paths. Low-voltage cell operation is then analysed, with failing cells replaced as necessary.

Clock tree design

A traditional clock tree (CT) at nominal voltage is implemented using a cascade of clock tree buffer and gating elements from a single root element to the leaf cells that require them. The challenges in routing the clock tree are the low-skew requirement at the endpoints, and also (because of the inherently high switching activity) the power consumption.

These challenges are even greater at ULV, since because of the variability and performance differential, the potential endpoint skew between branches of the clock tree can be significant. Typical clock trees are also relatively short, causing their variations not to be averaged out as is the case in the datapath [62].

A potential solution is to employ a common buffering stage for the entire clock tree [70]. This enables all the endpoints to be affected by the same variability, and the signal propagation issues which suffer less from this issue are easily handled by the CAD tools.

For low-power aspects, elements such as clock gating cells can be inserted to reduce unnecessary switching. In [31], these were implemented using upsized LVT MOSFETs for cells within the timing path, instead of the SVT variants employed in the rest of the logic, in order to reduce incurred delays, for a fairly modest power overhead, creating a multi-V_T clock tree.

Another method would be to employ statistical timing analysis (SSTA) in order to better simulate the variability effects and plan the CT accordingly, but this is currently prohibitively computationally intensive for large designs and so is not usually possible for complex SoCs.

Glitch reduction

Another unexpected effect of reducing the supply voltage and increased variability is the appearance of glitches [71] that are hard to model at the system-level.

Contrary to in a ring oscillator, where typically only one propagation wave exists, combinatorial blocks receive many data sources, and propagate data to a number of outputs. With added variations, signals may no longer arrive in the predicted order, thereby increasing a gate's actual activity factor α_F and with it the dynamic power consumption.

While this remains a concern, requiring in the future better modelling of this effect by the design tools, designers can limit its impact through the addition of glitch-masking logic on critical nets, which charge large capacitors such as memory arrays.

Summary

This section has briefly touched on the implementation difficulties of physically implementing ULV systems. The main issue is the unpredictable nature of transistor variability, which can induce effects that are not usually taken into consideration by traditional models. SSTA methods are able to accurately model these

effects, and are progressively being integrated directly into the CAD tool flows, but remain very computationally intensive.

These effects are set to become ever more significant as the underlying technology continues to scale, and while voltage and timing margins can offer protection against them, these significantly and negatively impact the performance achievable in ULV designs. However, through the use of adaptive on-chip power converters and frequency generators, careful selection of library cells, and proper clock tree planning, these effects can be overcome for the time being without having to fall back on more aggressive circuit analysis techniques.

1.3 MICRO-CONTROLLER DESIGN METRICS

In order to achieve this thesis' goal of optimizing a micro-controller for low-power operation, it is important to define metrics in order to evaluate the impact of the various techniques employed. In general, there are two distinct aspects of processors that need optimizing in the context of WSNs; the power, and the speed.

The power aspect is fairly straightforward, in that since the useful payload of a WSN is the sensor and data transmission to a control node, the micro-controller must strive to contribute as little power as possible and even attempt to reduce the overall system power consumption through data processing techniques. This translates to minimizing P_{tot}, the total power consumption of the micro-controller system.

It is also necessary to consider the speed at which an operation is able to execute. For this, the latency and throughput of the processor system will be considered, and thus also the operating frequency, which all have to be maximized.

Since one of the optimization techniques outlined in Section 1.1 requires good duty-cycled performance, another criteria is the inactive power P_{sleep} consumed by the processor when it is inactive. Ideally, this should be as close to zero as possible, since this represents wasted power. Similarly, it is necessary to consider the sleep and wake times, the delay required in order to transition to and from sleep and active states. This phase is problematic from a power consumption viewpoint, since whereas the processor is still not achieving productive operation, it is generally ramping up its power consumption. These times therefore also need to be optimized.

A standard micro-controller power metric is the power consumed normalized to the frequency of operation, expressed in micro-Watts per mega-Hertz (μW/MHz). This is equivalent to the power per instruction and power per operation. Assuming processor systems have comparable latencies and code lengths required to implement a specific task, this metric allows the comparison between different systems and optimization techniques. However this alone is insufficient in this case, since this would favour a system that consumes nothing but does not necessarily do much either. What the system is capable of doing during each

cycle must therefore be considered, as well as the time required to perform a given task. The former is given by a list of features of the instruction set, while the latter is given by the operating frequency.

One unbiased metric for gaining an insight into the performance of a processor is simply the energy required to perform a given task, E_{task}. Minimising this will ensure that the average performance of the system is adequate, but also that it can efficiently execute tasks requested of it by the application and minimise the time required to do so. This will allow features such as parallel data processing, which consume more per operation but allow the much faster execution of a task. One danger with such a metric is a tendency to over-optimise for a given task, which is why all features of the processor are discussed from the viewpoint of a common need for these types of applications in the context of the IoT, and the benchmark programs chosen are representative of tasks that could actually be encountered in real-life applications.

The optimization objective of this thesis is, therefore, to reduce the energy per task of the resulting system, under the constraints of 10-100MHz operation and with an average energy within the capabilities of the power management system. The further objective is to ensure that the resulting system is more effective than the raw transmission of data with offline processing, to achieve the same end result.

1.4 A BRIEF BACKGROUND ON ENERGY HARVESTING SYSTEMS

Since the aim of this thesis is to design a processor system capable of operating on harvested energy, it is necessary to briefly present the different options available. The aim of an energy harvesting, or energy scavenging, system is to obtain power for the node through ambient means, so that the device does not have to be powered externally and can thus, in theory, operate continuously for the duration of the device lifetime. This means energy harvesting seeks to capture wasted ambient power and put it to good use.

Energy harvesting can be used to either power the node directly, or through the use of an on-board power management unit (PMU) that stores energy in a battery or capacitor for later use. A combination is the favoured strategy, powering the node and using the remainder to recharge an energy storage unit, since this allows the system to be relatively resilient against temporary power interruptions. In particular, many nodes only have volatile memory storage (see previous sections), that, aside from a probable loss of sensor data, require a high-power reboot and efficient external management of the reconfiguration and internal software.

The main sources of energy are solar, wind, mechanical, vibrational, thermal, and radio (RF) [2]. Although wind power is highly effective, it requires large structures in order to appropriately harness it. While this may suit a number of applications of the IoT, the focus of this thesis is on small-scale, sub-cm^3

('smart dust') nodes which therefore also imposes a size constraint on the energy harvester.

Solar, using a photovoltaic (PV) cell, provides the largest source. The outdoor power density from the sun ranges from $100 - 1000\text{W/m}^2$, while indoors this is generally less than 10W/m^2 [72]. This of course varies widely with the location, time of day, and other environmental conditions. Given that current PV cells have a low efficiency of only $1 - 5\%$ under artificial lighting conditions [73], the expected indoor power generation is of the order of $10 - 500\mu\text{J/cm}^2$ [74, 72, 2]. It must of course be noted that this is a very volatile power source, due to its dependence on environmental conditions and the absence of light at night for instance, which further requires a battery to be charged to ensure continuous operation. Furthermore the voltage can vary significantly and must therefore be regulated appropriately.

The piezoelectric harvesting of mechanical energy is able to achieve around $330\mu\text{W/cm}^3$, vibrational transducers $116\mu\text{W/cm}^3$, and thermoelectric heat converters $40\mu\text{W/cm}^3$ [2]. Radio energy harvesting is a bit different in that the receiver antenna has to be finely tuned to the anticipated frequency, requiring it to be highly targeted towards a specific application. Also, since radio by its very nature radiates, the transmitter has to be installed within a few metres for it to be effective. However, in the case of active RFID tags, that are powered by a read request from a control device, this has been shown to work very effectively within the constraints of that system.

Considering the available power that can be harvested from a miniature node, it can be concluded that energy harvesting will be able to give around $10-500\mu\text{W}$ to the underlying micro-controller. The aim is thus to design a system that can achieve operation under this power constraint, with appropriate duty-cycling.

1.5 MICRO-CONTROLLERS FOR ULTRA-LOW POWER SYSTEMS

When the world was mainly dominated by desktop-class processors, most of those released by the largest commercial players targeted high performance, with energy efficiency being only a secondary consideration. But with the advent of mobile computing, energy efficiency is becoming increasingly important. ARM and TI are both important players, recognized for their low-power systems, with others working on competitive offerings, such as INTEL's Quark product line specifically targeting the IoT.

However, most relevant to the IoT vision are the research micro-controllers, offering the lowest power consumption. These come in many shapes and sizes, with some achieving under 2pJ per instruction. A recent intra-ocular pressure monitor [75] for example includes a simple 8-bit processor operating in the sub-threshold region at minimum-energy point. It implements an 10T SRAM memory, and is supplied 0.45V for 100kHz operation (see Table 1.1).

Similarly, [69] presents a low-power digital micro-controller used to calibrate the subthreshold bias of the System-on-Chip's (SoC) analog components and

Table 1.1.: Micro-controller system comparison, showing the impact of architecture, supply voltage and technology.

Work	Hanson, JSSC [75]	Zhai, VLSI [76]	Zwerg, ISSCC [77]	Kwong, JSSC [63]	Bol, JSSC [31]	Ickes, ESS-CIRC [78]	Luetkemeier, ISSCC [39]	Bartling, ISSCC [79]
CPU	8bit	8bit	16bit	16bit	16bit	32bit	32bit	32bit
Technology	180nm	130nm	130nm	65nm	65nm	65nm	65nm	130nm
Memory	0.5kB	0.25kB	1kB + 32B I$ + 16kB FRAM	16kB + 8B I$	18kB + 64B I$	8kB IM + 8kB DM + 128B I$ + 128B D$	16kB IM + 16kB DM + 0.6kB Subvt	10kB ROM + 8kB SRAM + 64kB FRAM
Frequency [Hz]	100k	833k	24M	434k	25M	1.65M	4.2M	8M
V_{DD} **[V]**	0.5	0.36	2.0 - 3.6	0.5	0.4 CPU, 1.0 MEM	0.6	0.5 CPU, 1.0 MEM	1.5
Active power [μW/MHz]	2.6	2.6	164 (FeRAM), 130 (SRAM)	27.3	7	~11	~18	225 (FRAM) 112 (SRAM)
Standby power [μW]	0.035		<6 @ 85°C	<1 @ 25°C, ~ 7 @ 85°C	1.5 @ 25°C, 17 @ 85°C			0
Die size [mm^2]	~1	~0.09	4.4	1.62	0.42	~1	~1	4.4

for signal processing. The core is based on a modified PIC16C5X architecture, and was designed using Monte-Carlo analysis of library cells for subthreshold robustness. A similar, but higher performance micro-controller is presented in [76]. These 8-bit systems [69, 75, 76] target low frequency operation, at which point the MEP is achieved when implemented in larger technologies (0.18 and 0.13μm respectively) that offer the best leakage options. Since their computational load is low, and their sensors limited to 8 significant bits, their 8-bit processor architecture is ideally suited.

Should more performance be required, a 16-bit architecture may be necessary. The commercial variant of TI's MSP430 micro-controller [77] achieves good performance, but also embeds a large, 16kB non-volatile FeRAM which makes it of greater practical value, but also increases the system's power consumption. When running from the FeRAM, it consumes 164μW/MHz, whereas only 130μW/MHz using only the on-board SRAM memory. Being a 16-bit processor, it obviously consumes more power than the 8-bit alternatives described previously, but it is also able to accomplish more advanced processing tasks. Additionally, it can operate at up to 24MHz, orders of magnitude more than the former.

Research variations of this processor, such as [63] and SleepWalker [31] employ a more advanced technology node (65nm), and in the case of SleepWalker more aggressive techniques to maximize operating speed, allowing it to achieve 25MHz operation with a 0.4V supply and a power consumption below that of the commercial variant. These both employ an on-chip DC/DC converter in order to achieve best conversion efficiency.

Moving on to larger systems, [39, 78] present micro-controllers based on 32-bit processor cores. Both have a low-power cache in the ULV domain, and exploit technology features in an attempt to minimize power consumption. The architecture presented in [39] is also SIMD-compliant, though only one SIMD way was implanted on the demonstration chip. Although once again the power consumption is increased with respect to 16-bit architectures, the capabilities of such systems enable far more advanced computing, providing significant benefits to the target application. Once again, through faster computing they are able to complete tasks faster and with more resolution, thereby allowing longer sleep times and more sophisticated data analysis and compression.

It is also interesting to mention the icyflex family of 32/64-bit micro-controllers developed by the CSEM, which are distributed in HDL form to be used in a variety of third-party systems. The icyflex1 has been characterised in the 180-nm node in [80]. Due to its availability in HDL form, the core can be tuned to specific applications by removing unnecessary features, and is configurable at runtime specifically with regards to addressing mode. It is also capable of optionally executing two instructions in parallel, but maintains a 32-bit instruction word size to limit fetch power overhead, in contrast to traditional VLIW architectures. Performance results show it to achieve a maximum clock frequency of 50MHz at 1.8V, and an average 120μW/MHz at 1.0V. The icflex2 targets lower-power control applications, by sacrificing DSP features and limiting the operations possible. Unfortunately there exists as yet no publicly-available

published paper documenting the performance of these systems under the conditions expected of WSNs.

Lastly, [79] presents a 32bit ARM Cortex M0-based micro-controller that implements non-volatile logic, along with a large FeRAM. Although non-volatile flip-flops present manufacturing challenges, they allow the system to exhibit zero standby power characteristics. Additionally, the processor state is restored when power is returned, thereby avoiding the necessity of a lengthy bootload process.

Another important aspect of nodes for the IoT is the level of integration. In many applications, especially for bio-monitoring, 'silicon dust'-sized systems are desirable. Although the more advanced processors in Table 1.1 demonstrate high system integration, they often require proportionally large external components in order to operate autonomously, such as solar panels, batteries or crystals.

There has been a lot of recent progress in reducing the size of the overall system through in-package integration, combining multiple different designs implemented on varying technologies, and through the use of miniature thin-film batteries. [81] is one such highly integrated system, implementing a cubic-millimeter scale device combining solar panels, sensors, processing and energy storage. Another highly-integrated system that combines up to 64 ARM cores in-package is presented in [82], which although not itself targeted to the IoT, shows the potential for system-in-package integration.

There are clear benefits stemming from a high degree of integration. Being in close proximity, transmission losses can be minimized, and co-integration gives designers a greater choice of technology, selecting the most suitable for each task, rather than for the most demanding. And for many applications in the IoT, systems of such reduced size will be key enablers.

The systems presented in Table 1.1 clearly show the power impact of using more powerful architectures. However, it should be noticed that through the use of more advanced technology nodes, reduced supply voltages, ULV caches and an increased system integration, they are each able to perform within a similar power range. While slower processors clearly benefit from older, lower leakage technology, high performance can be achieved on the newer nodes through careful design and system partitioning. Through co-integration, the benefits and specificities of each can easily be exploited, making for a better, more efficient overall system.

1.6 NEXT STEPS

This chapter briefly outlined the various optimization strategies used for low-power design, the implementation challenges faced at ULV, and has presented a number of state-of-the-art processor designs that fall within the application context of WSNs.

Using these techniques, it is possible to reduce the operating power significantly, yet still achieve reasonable performance. SleepWalker [31], for example, yielded a 16-bit micro-controller capable of 7μW/MHz active operation, while

operating at 25MHz at 0.4V. This falls well within the expected capabilities of an associated energy harvesting system.

Building on the state-of-the-art, the following chapters of this thesis will in turn explore how to move beyond what current micro-processors achieve, and aim for the same low-power system but with increased computational capabilities. Beyond that, alternative strategies will be used to lower the power consumption of wireless sensor nodes.

CHAPTER 2

BELLEVUE: A LOW-POWER MICRO-CONTROLLER FOR PROCESSING-INTENSIVE APPLICATIONS

In the IoT, minimizing the power requirement of each node is a practical necessity for harvested energy operation, which requires that both the power necessary for the sensor itself and that required to transmit the sensed data be reduced. Due to the high impact of the latter, a wise system-level optimization strategy must allocate more computing power to pre-process the sensed data in order to reduce the wireless communication time and thereby the overall system power. While the state-of-the-art WSN designs reviewed in the previous chapter were capable of low-power operation, the focus of this chapter is to increase the achieved performance within the constraints of an harvested-energy system, and targets video monitoring applications that require greater DSP capabilities. This chapter therefore presents Bellevue. This is a 32bit 50MHz MIPS1-compliant high-performance CPU running at 0.37V, which enables online data analysis through good processing capabilities, a low-power 1.55μW sleep mode, and a variable-width SIMD pipeline and multiply/divide unit, achieving 7.7$\mu W/MHz$ overall.

Contents

Associated author publications

ITK1. D. Bol, J. De Vos, F. Botman, G. de Streel, S. Bernard, D. Flandre, and J-D. Legat, "Green SoCs for a Sustainable Internet-of-Things", in *Proc. Workshop Faible Tension Faible Consommation (FTFC)*, 4 p., 2013.

CP6. F. Botman, J. De Vos, S. Bernard, J-D. Legat, D. Bol, "Bellevue: a 50MHz Variable-Width SIMD 32bit Microcontroller at 0.37V for Processing-Intensive Wireless Sensor Nodes", in *IEEE International Symposium on Circuits and Systems*, 2014.

CP7. D. Bol, G. de Streel, F. Botman, A. Kuti Lusala and N. Couniot, "A 65-nm 17-pJ/frame.pixel 0.5-V DPS CMOS Image Sensor for Ultra-Low-Power SoCs achieving 40-dB Dynamic Range," in *IEEE Symposium on VLSI Circuits*, 2014.

Chapter 1 described state-of-the-art low-power processors [78, 39, 83, 31] capable of operating under the constraints of energy harvesting, but these were limited in their computational ability through low operating speeds or restrictive architectures. With clock frequencies below 1MHz or with 16-bit architectures, these micro-controllers lend themselves well to WSN control applications, but not to DSP or processing-intensive tasks. This chapter therefore focuses on the design of a low-power processor targeting DSP tasks, while remaining sufficiently efficient to run off harvested energy.

In order to demonstrate this, the custom 32-bit Bellevue micro-controller was designed and taped-out. At the heart of the 65-nm CMOS SunPixer SoC featuring an energy-harvesting power management unit (PMU) with external supercapacitor storage, an on-chip low-power CMOS image sensor (CIS), Bellevue features greater DSP capabilities required [84] for image-processing applications.

2.1 BELLEVUE SYSTEM ARCHITECTURE

Figure 2.1 shows an overview of the SunPixer SoC architecture. There are three main functional blocks; the PMU, responsible for harvesting power and managing an external energy cache, an on-chip CIS, which is the useful sensor payload of the node, and Bellevue, the CPU block in charge of data storage and processing.

The 65-nm CMOS design is partitioned in manner similar to [31]; the always-on peripherals and memories are powered using a nominal 1V supply that is best suited to their switching activity, low-standby power, and always-on requirement, and are therefore implemented using the highest available V_t (thick-oxide LP with high-V_T). Similarly, the two 32-KByte memory blocks use foundry 6T SRAM macros in order to achieve the required stability and density. Meanwhile the CPU uses a 0.37V supply generated by the on-board PMU/AVS units and is implemented in thin-oxide GP cells optimized for low power. The CIS is supplied 0.5V, also generated on-chip by an on-chip LDO.

Since the PMU, AVS, LDO and CIS are external to Bellevue, their implementation details are left to their associated publications [86, 87, 88], and are henceforth considered as peripherals with which the processor can interact. Since CIS is the useful sensor payload of the chip, but consumes only around 0.3μJ/frame [88], and that of the rest of the system is much greater, the optimisation to minimise the consumption of the processor is a relevant effort. The remainder of this chapter is thus structured as follows; Section 2.2 presents the design of the Bellevue processor core. Section 2.3 describes its physical implementation, with simulation results outlined in Section 2.4. Lastly, conclusions are drawn in Section 2.5.

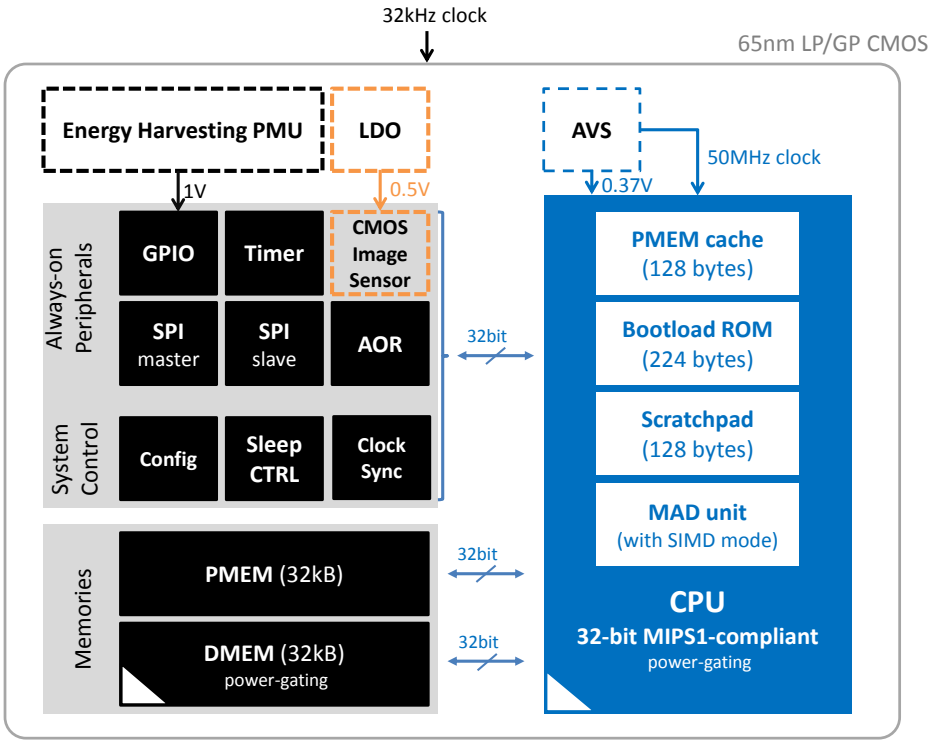


Fig. 2.1.: Bellevue system architecture, colour-coded by power domain. A 32kHz input clock is the only external requirement. Both the CPU and DMEM can be power-gated in sleep mode (switchable domain). The PMU, AVS, LDO and CIS (dotted outline) are external to Bellevue.

2.2 PROCESSOR DESIGN

At the heart of the SunPixer SoC is the Bellevue micro-controller, a custom-designed processor for high performance and low power. In order to achieve the stated goal of increased performance with respect to the state-of-the-art for an equivalent power usage, three main directions were followed; the clock speed was increased, the architecture augmented to 32bits, and additional performance-enhancing techniques were installed.

2.2.1 Choice of architecture

Increasing the clock frequency beyond 24-25MHz [77, 31] is not simple, given that such speeds are already pushing the envelope for transistor switching times at such low voltages. This therefore requires, amongst other factors, that the critical path lengths be considerably shorter in order to achieve a doubling of

the operating frequency. In Bellevue, a 5-stage pipeline structure was therefore adopted with a view to both increase throughput through shorter and faster register-to-register paths within the core, as well as simplifying the overall structure to avoid complex timing loops.

At the same time, a 32-bit architecture was deemed necessary in order to perform all the operations required of it without undue overhead. This choice impacts not only the core, but also the peripherals and memories, the latter needing to be of much greater capacity.

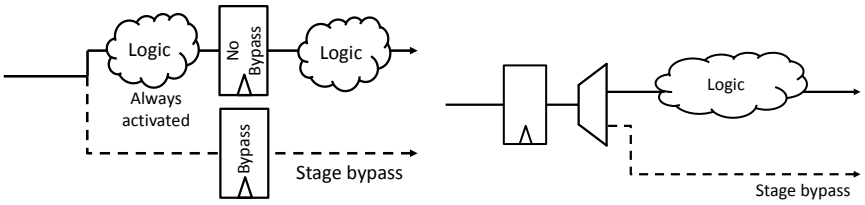
The MIPS architecture was eventually chosen as being a good compromise between the different requirements. On the positive side, the instruction set being a RISC is limited and able to yield a coherent implementation of considerable elegance, and the presence of a branch delay slot eliminates the need for a branch prediction mechanism, without significant impact on the program flow. The presence of 31 usable internal registers reduces the need to access data stored in memory, in a way acting as an inline data cache towards the latter, thereby reducing the switching activity and dynamic power consumption of the memories. On the less positive side, the core lacks a compressed instruction set representation, like the Thumb mode in ARM processors, that would allow the program memory to be used more densely, and also has less advanced compiler support than the latter. Overall, the MIPS architecture was found to be most promising, and was thus chosen for Bellevue.

The Bellevue processor therefore implements the MIPS-1 instruction set, though not necessarily in a cycle-accurate manner due to the custom nature of the implementation. Only instructions that are still covered under copyright protection are not implemented: `swl`, `swr`, `lwl`, `lwr`. Avoiding these instructions was also done in order to ensure a cohesive memory access module implementation that could be shared by all other instructions and internal requirements, thereby significantly streamlining the resulting design.

2.2.2 Switching isolation and glitch reduction

Designing a processor core for low-power is a change in perspective when compared to more traditional, performance-driven design. Whereas the latter is a constant search for shortcuts and acceleration techniques, low-power design involves unification, an attempt to process all instructions using unified modules in order to streamline both the instruction execution process and the underlying hardware itself. In a way, therefore, it is a search for elegance within the design in order to reduce the amount of hardware used, while still maintaining the required functionality.

In the Bellevue core, all instructions follow an identical route, only passing data where necessary to the downstream stages. Resources are reused, for example the main ALU is also used to calculate PC-relative offsets when accessing memory and similar operations. This allows the hardware to be largely streamlined.



(a) Timing driven register placement.

(b) Low power, switching optimised register placement.

Fig. 2.2.: Possible register placements at the head of a pipeline stage, depending on optimization strategies. Timing requirements and the need for perfectly-balanced pipeline stages often result in designs such as (a), whereas switching and low power concerns favour (b).

One of the advantages of such a structure is that the registers are placed outside the functional core of the pipeline stage modules. This strategic register placement reduces the number of switching registers to barely more than the data width from a possible overhead caused by duplication in the case of timing-driven register placement (Fig. 2.2). Such duplication is traditionally the result of attempting to equalise the lengths of each pipeline stage, in an effort to maximize performance. While this works very well for high-performance designs, it does introduce hardware duplication and unnecessary switching in the case of stage bypass (Fig. 2.2a). Although in Bellevue the stages are relatively well balanced, a switching-optimized design (Fig. 2.2b) was consistently favoured for low-power power reasons.

Similarly, this placement enables the isolation of switching activity to modules where such switching is required for functionality, and prevents propagation of redundant switching and glitches into modules that have no use for it. This, for example, prevents unnecessary propagation onto high-capacitance nets and allows a better bounding of glitches to a single stage.

Since register duplication occurs in roughly 20% of registers in the processor core due to tight timing requirements before or after a complex pipeline stage (generally equivalent to roughly one in five stages), with these registers being inserted within the first 25% of the logic path, then a conservative estimate of the energy gained in this way through proper register placement gains over 5% for every cycle, almost 0.24μW/MHz overall through a reduction in the amount of activated but redundant logic. The registers themselves, being controlled by mutually exclusive bypass and non-bypass signals, only contribute towards the leakage energy.

2.2.3 Variable-width SIMD pipeline

Since the primary function of the CPU, besides WSN control, is to interface with the 8bit CIS peripheral, the use of a 32bit architecture may seem wasted

in this case, since 24-bits potentially remain unused. For this reason, a pseudo-SIMD variable-width pipeline is implemented throughout, giving the processor the capability of transparently operating on 32bit data, on two independent 16bit data simultaneously, or on four 8bit data, performing the same operation on each. Depending on the mode, data contained in memory, peripherals, or internal registers is either interpreted as a single 32bit data word, two independent 16bit half-words or four bytes, thereby maintaining an identical bus architecture in all cases. All instructions are available in all modes, and the underlying hardware is the same in each case.

Since some MIPS-1 instructions are able to generate an overflow exception, these are generated whenever any one of the processed data would induce an overflow. Mode switching is done under CPU control and takes only a single cycle, allowing flexible processing options depending on the data and application requirements.

By being able to handle 4 data bytes in parallel, the processor is able to access all pixels from the CIS in a quarter of the time normally necessary, thereby using only a quarter of the energy before returning to sleep (see Fig. 2.10).

2.2.4 Multiply-Divide (MAD) Unit

In order to achieve the performance requirements imposed by the application, a hardware multiply-and-divide (MAD) unit is implemented. It operates in parallel to the existing pipeline where traditional operations are executed, the latter only stalling when a read request is made to data that is still being calculated in the MAD unit.

Consistent with the variable-width SIMD pipeline functionality of the rest of the CPU, the MAD is also SIMD-capable, and can function in 32bit, 16bit or 8bit mode using only hardware resources common to all modes (Fig. 2.3). By breaking up a data word into its component bytes, the hardware is able to incrementally obtain the desired result over a varying number of cycles. For multiplication, this is the result of the multiplicative expansion of the following: let $A = \{A_1, A_2, A_3, A_4\}$ be the component bytes of the word A, and $B = \{B_1, B_2, B_3, B_4\}$ those of word B respectively, then (ignoring shifts) the expansion can trivially be written as;

$$A \times B = \tag{2.1}$$

$$(A_1, A_2, A_3, A_4) \times (B_1, B_2, B_3, B_4) = \tag{2.2}$$

$$(A_1 \times B_1) + (A_2 \times B_2) + (A_3 \times B_3) + (A_4 \times B_4) + \tag{2.3}$$

$$(A_1 \times B_2) + (A_2 \times B_1) + (A_3 \times B_4) + (A_4 \times B_3) + \tag{2.4}$$

$$(A_1 \times B_3) + (A_2 \times B_4) + (A_3 \times B_1) + (A_4 \times B_2) + \tag{2.5}$$

$$(A_1 \times B_4) + (A_2 \times B_3) + (A_3 \times B_2) + (A_4 \times B_1) \tag{2.6}$$

The first partial calculation performed (Eqn. 2.3) corresponds to a multiplication in 8bit mode, and the MAD unit completes at this time when in 8bit

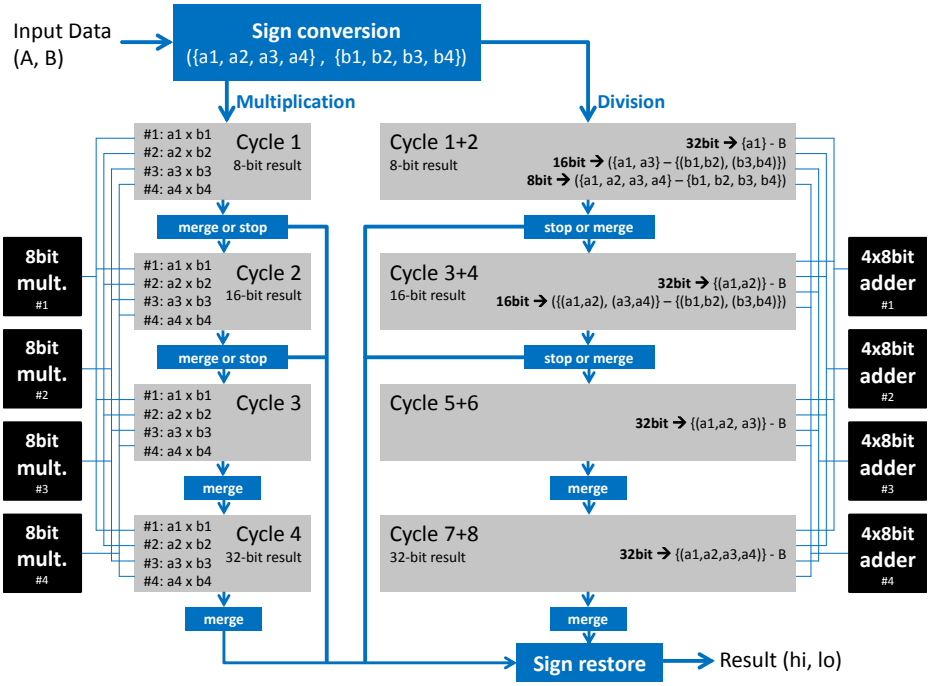


Fig. 2.3.: Architecture of the MAD unit, a fully SIMD-capable multiplication/-division unit for Bellevue.

mode. After the next iteration (Eqn. 2.4), the partial calculation corresponds to a 16bit operation result. After a further two iterations, the 32bit result is obtained (Eqn. 2.6). In this way, using only four 8×8 multipliers and four adders, the calculation can be performed for all requested SIMD modes over a given number of cycles.

Division is performed using a similar expansion, using chained adders. Because of the relative complexity of the division operation in comparison to multiplication, twice as many cycles are required for the former, but only requires the adders used for the multiplication. In order to meet the tight timing constraints without excessive overhead, each MAD cycle is multi-cycled $4\times$, thereby enabling 32-bit multiplication and division in 16 and 32 CPU cycles respectively, 8 and 16 cycles to handle two 16-bit data, and 4 and 8 for four 8-bit data. This MAD unit, combined with the variable-width pipeline architecture, provides a $70\times$ performance improvement over software alternatives in the best case, as shown in Section 2.4.

2.2.5 Dedicated compiler

Although Bellevue implements the MIPS-1 instruction set, through a design choice it does not support the optional instructions `swl`, `swr`, `lwl`, and `lwr` (see Section 2.2.1). Although the MIPS processor itself is well supported by compilers such as `GNU GCC`, a reliable open-source compiler with ports to numerous architectures, these compilers require the use of these instructions in order to function correctly.

In simple control applications, these instructions are rarely used and can trivially be avoided through 'alignment' directives in the source program code. But in more complex applications, such as zip compression, the algorithm accesses memory in such a way as to make the use of these instructions all but inevitable without significant development effort.

For this reason, as part of this work, a custom port of the `GCC` compiler was developed. The modifications prohibit the use of these instructions by preventing unaligned word memory access. This means that word access must have addresses in multiples of 4 bytes. All memory remains accessible through aligned word, half-word or byte access, only unaligned words are no longer permitted in order to match the limitations of the underlying system. This means that the compiler will automatically favour normal instructions to access aligned memory, and when unaligned access is absolutely required by the application the compiler will instead issue two instructions to replace the missing functionality.

While this does increase the amount of generated code in the rare cases where such memory access is required, it does not happen often and amounts to a negligible fraction of the total program size. However, it does allow the memory access hardware module to be considerably simpler and react within a single cycle, and better match the way all other instructions are handled. This reduces by half the amount of hardware required to implement the memory pipeline stage, and potentially avoids a single-cycle stall that would result from most implementations, thereby reducing the total amount of power required for the CPU, all for a negligible increase in program size.

2.2.6 Summary of features

The Bellevue CPU, at the heart of the SunPixer SoC, is designed to run from a 0.37V supply at 50MHz. The custom-designed MIPS-1 instruction set compliant 5-stage pipelined 32-bit CPU minimizes active energy by reducing unnecessary switching through strategic register placement within the pipelined framework, as well as through industry-standard clock and data gating techniques.

Bellevue features several AOPs including two SPIs with 8-byte FIFOs, used to bootload and send data to either an external memory or radio, a timer, GPIOs, and an always-on-register (AOR). Two 32kB memories are used for program (PMEM) and data (DMEM). In order to mitigate the energy overhead to the 1V SRAM [78], register-based caches are implemented in the ULV domain, using upsized gate-length (L_G=80nm) transistors that minimize leakage. For the PMEM,

a 128-byte direct-mapped zero miss latency cache architecture is used, with an inline write-back policy. For the DMEM, due to the random access requirement, a 128-byte 'stratchpad' memory is implemented instead, allowing full CPU control over the cached data. All caches can be disabled at runtime. Furthermore, the DMEM can itself be power-gated during sleep, sacrificing state-retention for lower power.

The timer and both SPIs can be configured to generate an internal interrupt (IRQ), waking the core when necessary. The processor sleep/wake process is managed by a custom-designed sleep controller, granting enough time to power up the domain, stabilize the supply and clock, and manage the isolation cell control for inter-domain communication. This takes a nominal two 32kHz cycles. A single 50MHz cycle is needed to sleep, a process which is initiated under processor control. It also centralises IRQ requests, and forwards them to the CPU when the latter is able to accept them. A bootloader, capable of receiving program code from either master or slave SPI, and chip configuration module, designed to selectively enable chip features, were also added.

Through optimised design, and the implementation of an SIMD-capable pipeline and MAD unit, the Bellevue core is able to achieve greater performance at low voltage than the competing designs discussed previously.

2.3 PHYSICAL IMPLEMENTATION

The SunPixer SoC was implemented in 65nm CMOS using a LP/GP process mix that matches the underlying transistor technology to the anticipated switching activity, as per the state of the art designs. The die micrograph of the final result is shown in Fig. 2.4.

The main challenges of ULV design in small-nanometer CMOS technologies are noise sensitivity and hold time violations [28]. Since an on-board AVS unit largely protects the underlying design from environmental and D2D variations, the main challenge here are WID variations. This section details both the design choices made to compensate for these problems, as well as some unexpected challenges that arose during the tape-out phase.

2.3.1 SoC implementation flow

The chip tape-out flow was done using the Cadence design tools, with in particular Cadence RTL compiler for the synthesis and Cadence Encounter for the place and route. A CPF-based multi-mode multi-corner (MMMC) flow was implemented, in order to give all tools a common insight into the different power domains: the default 1V domain containing the memories, AOP, sleep controller, bootloader, AVS digital logic, and chip controls; the 0.5V domain containing the CIS and digital control logic; and the 0.37V domain containing the CPU itself. The memories and all level-shifters and isolation cells were additionally given a dedicated power domain, in order to facilitate the layout phase. One of the advantages of such a flow is the automatic addition and connection of the level

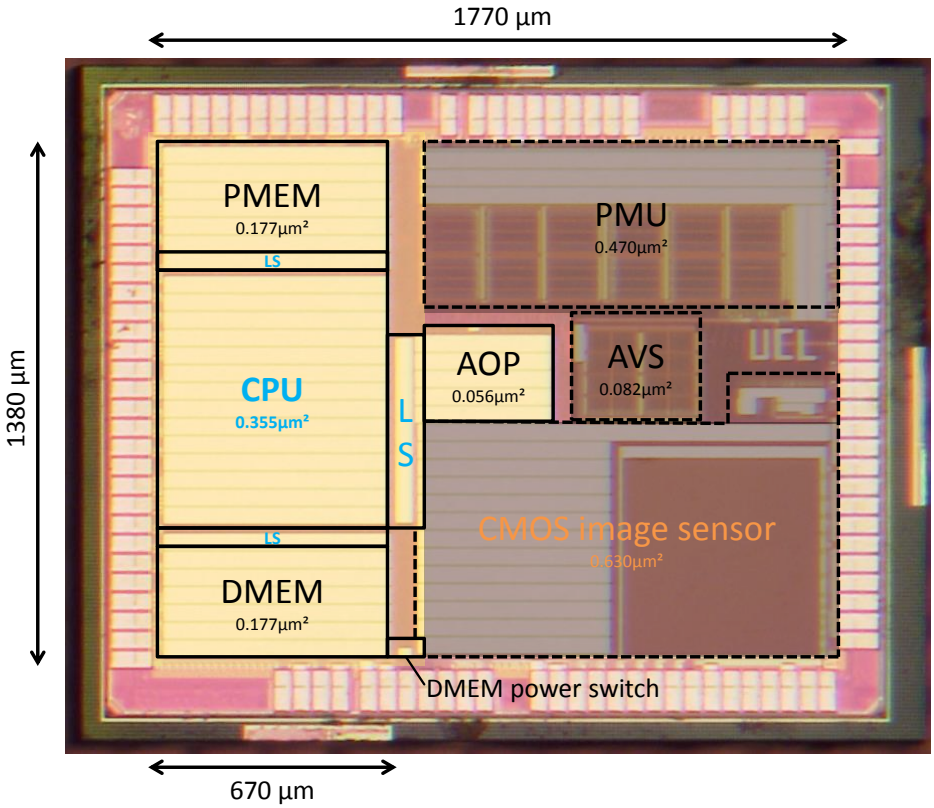


Fig. 2.4.: Die micrograph of the SunPixer SoC, physically implemented as part of this thesis, showing the functional regions and 1V (black), 0.5V (orange) and 0.37V (blue) voltage domains. The active regions are covered by AP and M7 metal layers for light shielding. The highlighted sections section represents Bellevue.

shifters and isolation cells between domains. The addition of the IO cells was done programmatically through custom scripts, in order to generate the correct external connections at the appropriate physical locations, based on a simple specification file, in order to quickly iterate over a number of floorplan variations.

As per state-of-the-art design recommendations for compensating PVT variations, an AVS unit [51, 89] was used to generate the 0.37V ULV domain power supply and the 50MHz clock. This enables the ULV domain to be modelled using only a typical TT corner (25°C, 0.37V) which the AVS strives to maintain. The efficiency of the chosen converter is 72% (from 1V to 0.37V). In this way, the cross-corner sign-off of the ULV block was considerably easier. All other blocks were modelled using all corners. The cell library used originated from [31], and

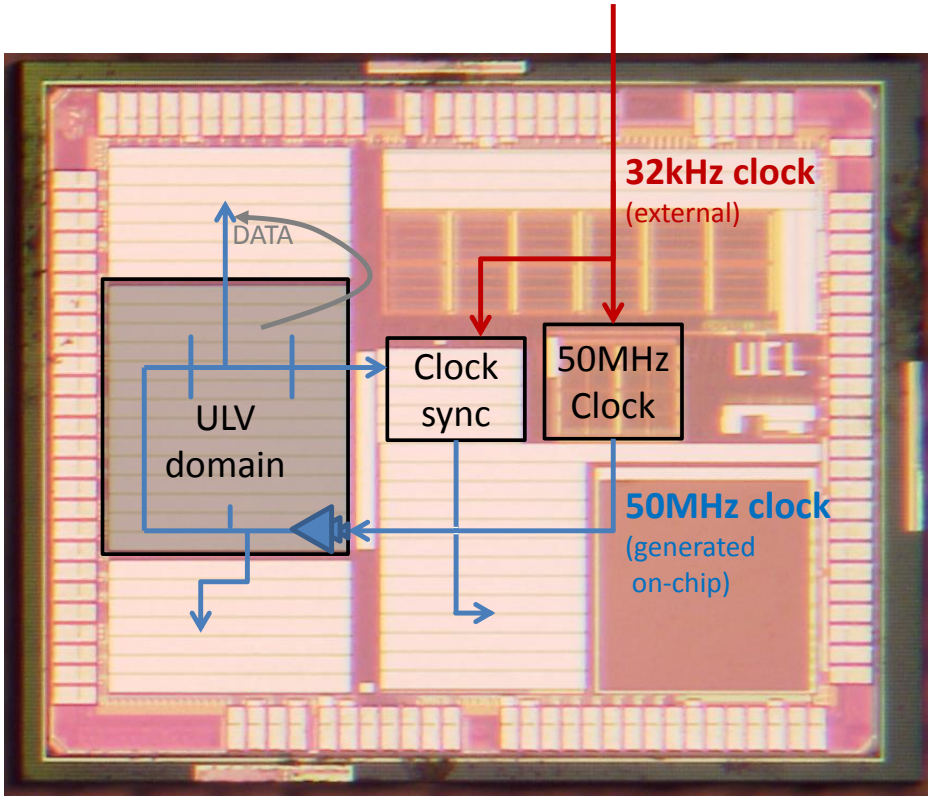


Fig. 2.5.: Clock path within Bellevue. 50MHz Clock generation is performed by the AVS unit (using an external 32kHz signal as reference), which then loops through the ULV domain. It is then re-synchronised with the 32kHz signal before reaching the different peripherals and memories.

contains 268 low drive-strength 3-stack maximum cells based on the foundry's nominal 1V cell library re-characterized for 0.37V and with upsized L_G. The three V_t options (HVT, SVT, LVT) were obtained later through CAD layer manipulations.

The clock tree topology mirrored that of [31], whereby a root-bufferized clock tree topology was chosen to minimize clock tree variability [70]. This is possible because the RC delays are comparatively much smaller than gate delays when at ULV [90]. In the CT buffers and clock gating cells, all gates within the timing path were implemented in the GP LVT variant, upsized to $L_G = 80$-nm.

In order to ensure achievable timings when accessing external peripherals, the clock generated by the AVS unit in the 1V power domain is routed through the ULV domain before reaching the peripherals themselves (Fig. 2.5). This ensures that the data and the clock signal have similar delays (rather than a

10ns latency and the requirement for a synthesized delay line of a more conventional approach), and facilitates timing closure during the placement phase. After emerging from the ULV domain, in active mode, it is also used to synchronise the 32kHz clock signal used in certain peripherals and avoid race conditions and redundant clock isolation precautions within the tools.

To further minimize redundant switching power, the prescalers used in the nominal-voltage domain employ a negative edge triggered flip-flops (FF) in a divide-by-two stepping topology proven previously in [31].

Finally, clock gating cells were automatically inserted at relevant positions throughout the circuit. Implemented with LVT MOSFETs within the clock timing path, and SVT variants outside in order to minimize power, the clock gating cells are specific to each power domain.

One challenge that arose was the inability for the tool to differentiate the six cell types (three power domains $\times$ positive and negative variants) and their use restriction to specific domains. This was solved through in-tool scripting, whereby the variant was manually selected in each case, with an additional layer of post-processing used to ensure correct V_t variants.

2.3.2 Level-shifter cell alignment challenge

One of the bizarre problems encountered was that the level-shifter cells, being not of standard size due to in-house design, could not be automatically placed into the required cell rows, even when dedicated rows of the correct height were set up.

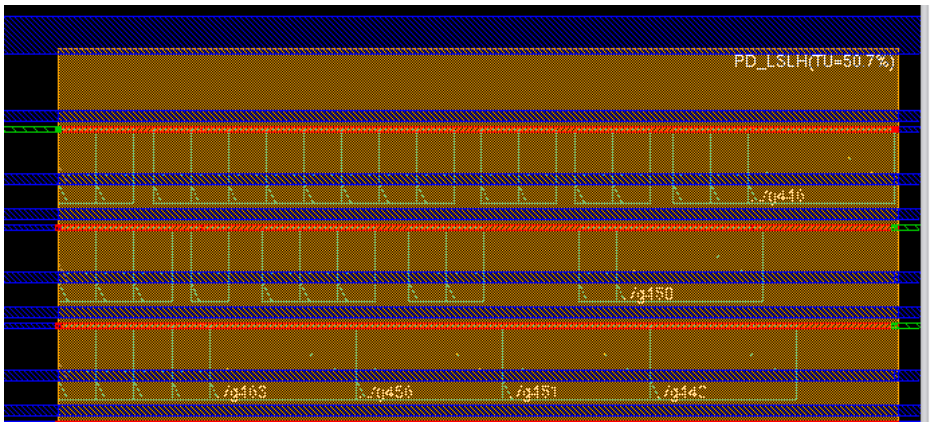


Fig. 2.6.: Misaligned level shifters in row of correct height and type, causing electrical short circuits as a result of automatic placement.

Attempts were made to limit these cells to dedicated power domains and regions featuring only rows of the correct type, but this similarly resulted in invalid cell positioning (Fig. 2.6).

The cells were therefore placed via a custom script, and prevented from moving during automatic placement. Attempts were made to minimize the timing distance through a first timing-driven inaccurate placement followed by a corrective shift of minimal distance. When the cells were then marked as badly placed and prevented the cell power rails from being inserted, these were drawn forcibly via a dedicated script and connected to the appropriate power distribution lines.

2.3.3 Light protection

One of the problems of having an image sensor on-board is that light has to be able to reach the sensor. Contrary, therefore, to a traditional chip where the packaging acts as a mask, the top of this chip's package is actually a quartz window.

Because of this, however, light will also expose other parts of the chip, causing an increase in consumption through light-induced excitation, and possible failures. It is therefore important to shield these parts of the chip from light exposure.

In order to achieve this, AP and M7 metal stripes (alternating in order to attain DRC density factors) are superimposed on the logic, as can be seen in the die micrograph (Fig. 2.4). However, this effectively reduces the number of routing layers, and while AP and M7, due to their high width and thus capacitance, are rarely used for signal routing, the high logic density of the CPU would have benefited from such layers, at least for power routing.

The routing levels were restricted to the lower layers, with few exceptions, and the shielding added manually later during DRC validation. Only the unused silicon areas were left open, as well as the MIM capacitors of the AVS and PMU, which are already protected by construction.

2.3.4 CIS routing congestion challenge

Another problem that emerged during the place and route flow was routing congestion in the digital CIS controller. Whereas the CIS controller row readout logic is synthesized [88], the CMOS imaging array itself is a hard macro. The interface between the two results in a dense logic arrangement for each sensor row, a problem that is further exacerbated by the stringent timing constraints required for low noise sampling.

Automatic, timing-driven routing resulted in a lumping of logic gates near the interface, inducing a logic misalignment with the rows and preventing the insertion of buffers, row decap filler, and nwell tie cells. The dense layout induced high congestion, a problem further accentuated by the absence of M7 and AP routing as these layers are used for light protection rather than power or signal routing.

To solve this, an attempt was made to use structured datapaths (SDPs), but at the time this did not have the required flexibility. Therefore the logic connectivity was extracted during synthesis through the use of a custom circuit

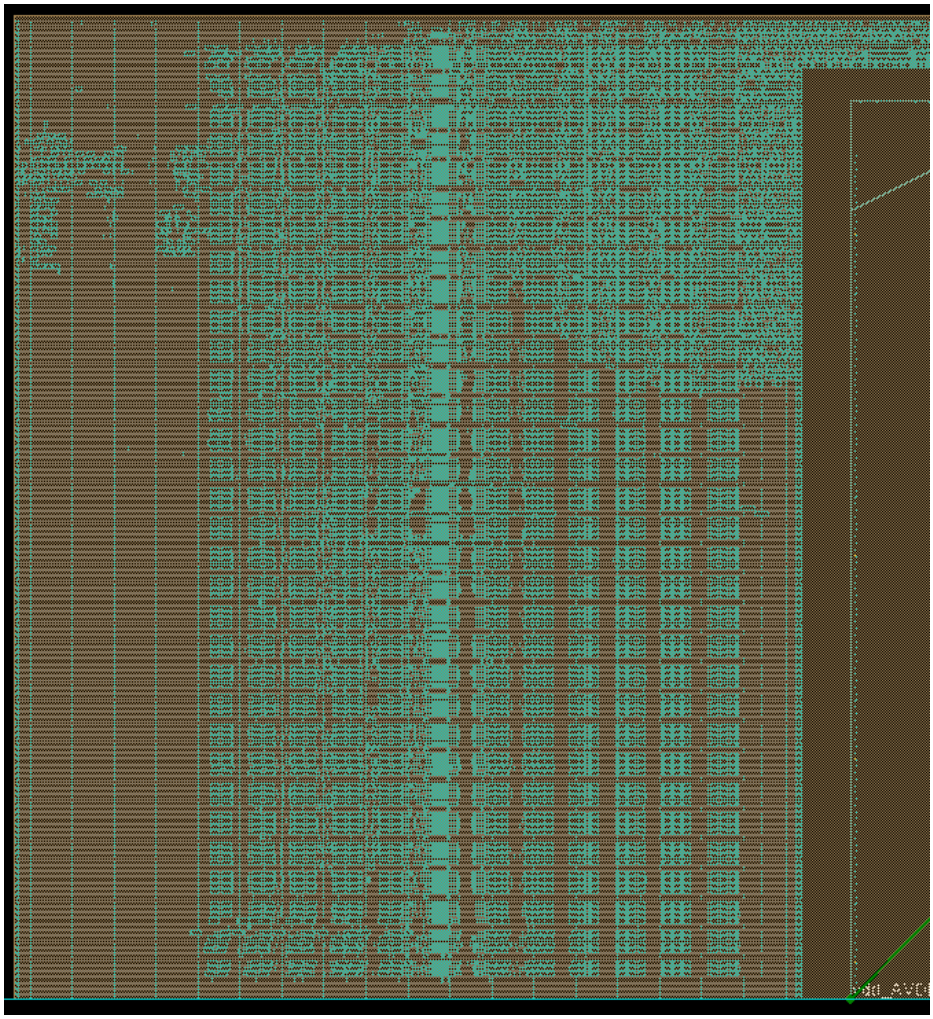


Fig. 2.7.: Structured arrangement of cells for the digital CIS controller, performed through a dedicated script. The high interface density with the CIS array hard macro rows (right of image) otherwise causes significant congestion. Cells involved in this structured placement fit a regular pattern leaving space for row capacitors, timing buffers, and other timing-driven placed cells (top right).

analysis program, and used for script-driven placement in `Encounter`. The resulting layout allows much more space for the insertion of buffers, and the row capacitors are inserted at regular intervals as necessary. Furthermore, the placement is connectivity-aware, enabling lower overall gate-to-gate delays within the sampling logic, resulting in better overall performance.

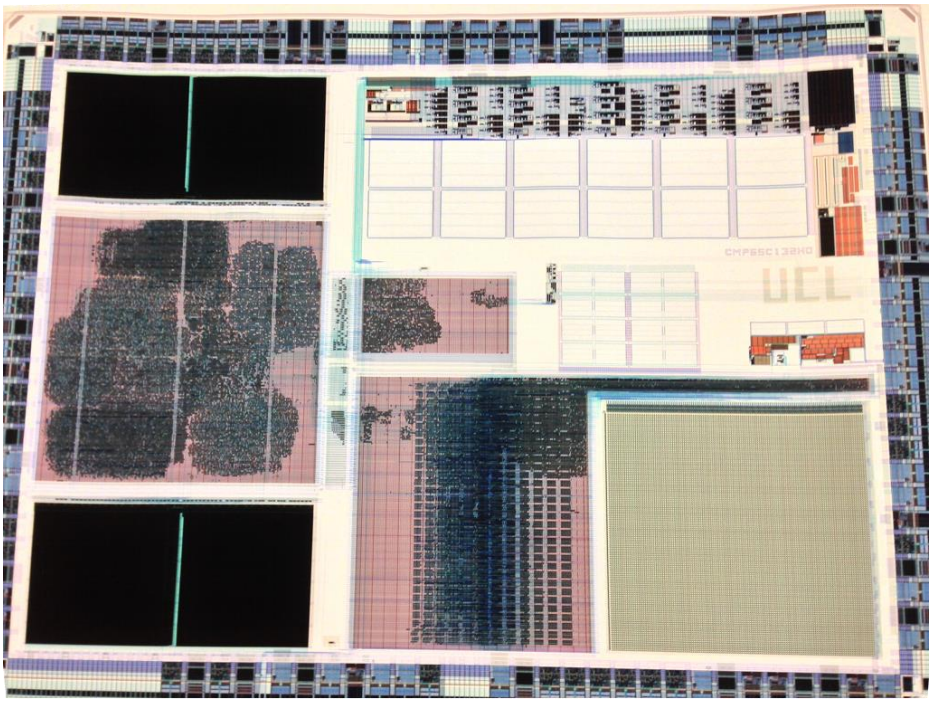


Fig. 2.8.: Cell implementation view of the chip, with power, routing, and shielding layers removed. The regular arrangement of cells in the digital CIS section results from a custom placement script developed to solve congestion issues. The darker regions near the memories are the ULV caches, and the CPU cells are visibly timing constrained, as their grouping shows. The placement of the three power lines over the CPU are also visible, and are aimed at providing a stable supply voltage in spite of any activity-related power surges.

Figures 2.7 and 2.8 show the resulting cell placement, with the regular row read-out logic visibly structured according to sampling requirements. The timing-driven placement is visible at the top right of the block, and uses both free rows and spaces voluntarily left in the regular arrangement.

Overall, this allowed the placement of dense logic without high resulting congestion.

2.3.5 Implementation summary

The design of this chip was a challenge due to the requirements of ULV design, but also because of the high number of hard macros, compact size, and dense logic arrangement, as well as limitations in the underlying tools.

During the place and route procedure, the timing requirements differed significantly from those that had been estimated for synthesis, requiring changes

to be made to the design. For instance, in order to achieve timing closure in the critical paths to and from the memory which are further from the CPU logic than anticipated, the operating voltage had to be raised slightly from 0.35 to 0.37V. Similarly, the interface with the CIS digital logic proved to be far slower than expected, requiring access to that peripheral to be multi-cycled (6×), while still achieving 266Mbps data transfer during image read-out.

Overall, however, these challenges were each faced one by one, and the chip was able to be taped-out.

2.4 RESULTS

Although the chip was taped out successfully, and although many of the analog blocks worked satisfactorily, the CPU itself could not be tested since it never requests a program to bootload and run. This embarrassing problem came as a surprise, since intense testing before tape out, with particular focus on the bootload process given its importance, had ruled out functional issues with the design. Both unit tests on the CPU itself, for each instruction and combinations with edge cases, and full-system functional tests with bootload carried out in simulation prior to tape-out failed to identify any problem.

The origin of the problem can be limited to either of two possibilities; either the simulation tools were faulty and did not detect a design error, or the problem lies in something not tested by the simulators. In the former case, the use of the Mentor Graphics Modelsim circuit simulator throughout the industry make the possibility of error very remote, leading to the conclusion that the problem lies in something not tested by the simulator. There are many such features in the design, starting with the different voltage domains. Modelsim uses libraries to describe the functionality and performance of cells at a given voltage under a range of connectivities, so cells like level shifters are not always accurately modelled. The same is true for a range of custom cells, such as the upsized L_G registers used in critical parts of the circuit, and also the manual V_t change through application of a mask. Problems can also be caused by unwitting short-circuits due to application of the light shielding, or by a voltage drop (IR) caused by the cell rows which, though it can be modelled, does not take into account the dynamic characteristics of the hard macros and the resulting contamination of the power supplies. There were also numerous problems during the initial foundry run, during which the dies could not be correctly processed, and while a respin was later made, the possibility exists that these dies share a common underlying manufacturing problem.

A number of different potential causes have been considered, and while a short-sighted early design choice opted not to include an on-chip debugger which would have allowed in-depth analysis, albeit at the cost of increased power consumption, a working hypothesis has been established. Measurements show significant noise on the ULV supply, accounting at times to over 150mV, a problem likely caused by the unusually long bonding wires required by the packaging. This noise is

sufficient to randomly reset the processor before being able to bootload, thus probably accounting for the failure. Attempts at limiting this noise are underway.

For this reason, the remainder of this chapter presents simulation results, which are believed to be very reliable. As a comparison, in [31] the power estimation at this stage yielded 194.9μW which gives 7.7μW/MHz compared to the actual 7μW/MHz actually obtained. The annotated switching activity was used in the place-and-route tool for power extraction, and the known DC/DC converter efficiency of 72% was used.

2.4.1 Image sharpening application

In order to demonstrate the functionality and performance of the system, a potential image sharpening application was chosen. This task fetches image data from the on-chip CIS, stores it in memory, then applies an image sharpening filter. With the appropriate filter, this can be used for edge highlighting and thus movement detection of whatever lies in the camera's field of view.

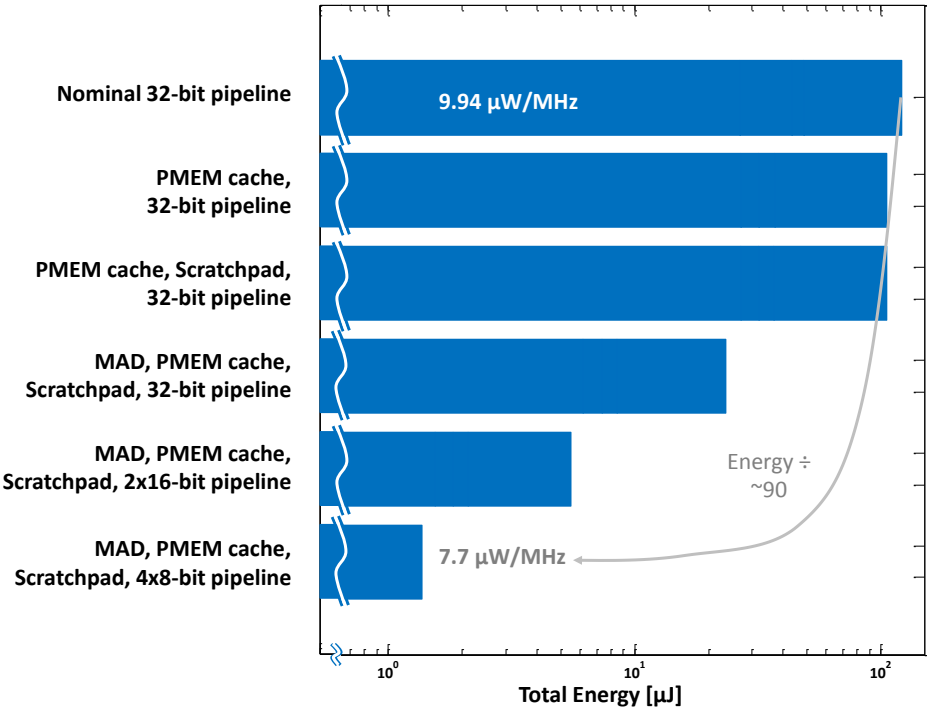


Fig. 2.9.: Total energy required for a sample 64×64-pixel image sharpening task, demonstrating significant energy reduction through MAD and SIMD operation (post-layout simulation data, TT die at 25°C).

The total system energy required for the image sharpening task and the contribution of the proposed techniques to the ultra-low system power are shown in Figure 2.9. Each of the proposed contributions reduces the overall energy required. The first two, the use of a program and data cache, reduce runtime power by limiting the access to the 1V memories, thereby reducing both level-shifter activity and memory line charging. The memories consume far less power, and while the consumption of the CPU and ULV caches increases, there is a significant power reduction overall. Due to the data-intensive nature of the tested application, the data scratchpad suffers from multiple misses, thereby reducing the potential energy gains apparent from this application.

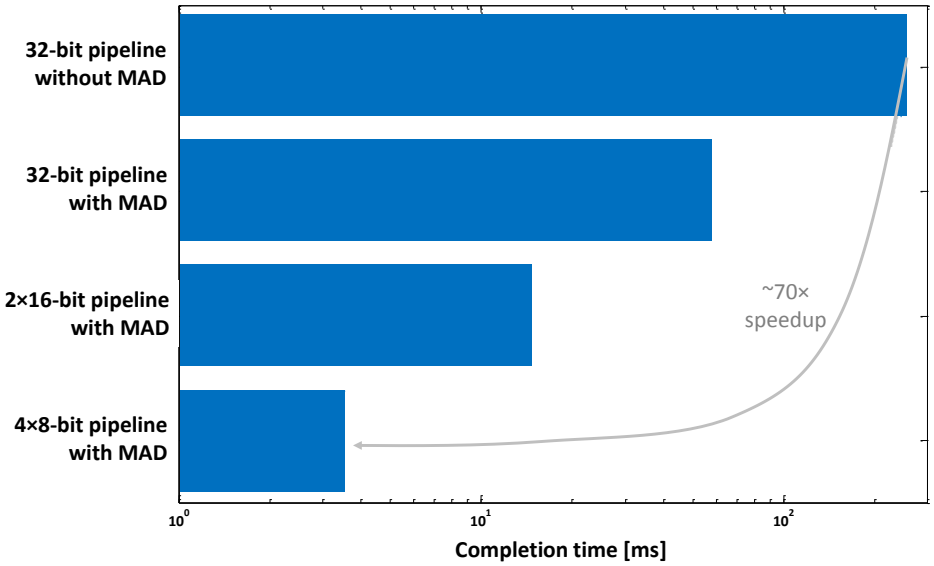


Fig. 2.10.: Time required for the system to execute a sample image sharpening task (64×64-pixel image resolution) in all operation modes, displaying the benefits of the MAD unit and SIMD mode (simulation results for 50MHz operation).

One of the most significant factors reducing the overall power consumption, though, is the acceleration enabled by the presence of a hardware MAD unit, and SIMD capability. Through the use of hardware acceleration, the CPU is able to perform multiplications and divisions with only a single instruction. While the actual calculation can take a number of cycles, the processor is free to either perform other tasks in parallel, or stall while awaiting the result, thereby no longer accessing the memories and thus lowering the overall power consumption. This acceleration amounts to ~70× (Fig. 2.10), as results from a lower resolution image show.

The active mode power breakdown (Figure 2.11) shows that the CPU power (including power converter inefficiency) dominates, as can be expected during processing-intensive tasks, consuming over 60% of the active power, with the

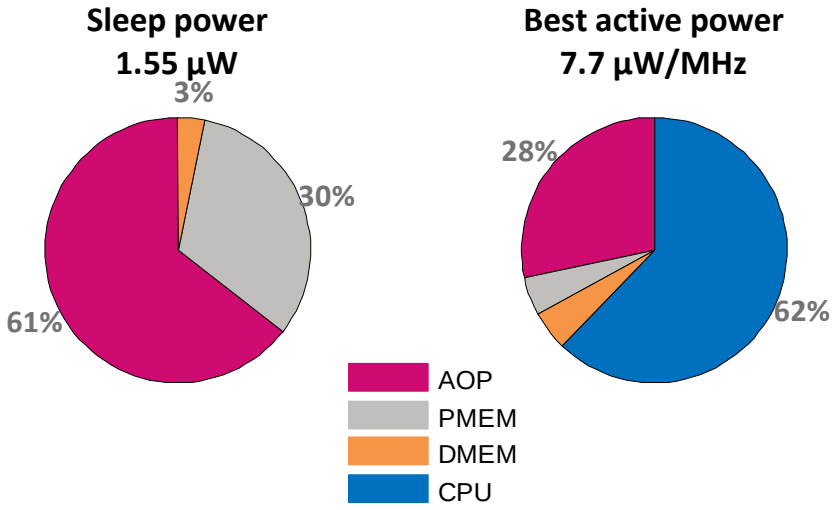


Fig. 2.11.: Power breakdown per component in sleep and active modes, showing the impact of power gating on the DMEM and CPU in sleep, and the large percentage of active power dedicated to processing (post-layout simulation data, TT die at 25°C).

AOPs consuming a large part of the remainder. Overall, an active power of 7.7μW/MHz can be achieved.

In sleep, however, the AOPs dominate the power since they continue to operate in that mode, allowing the timer to wake the processor after a configurable time interval or the SPIs to continue receiving data. This is followed by the program memory power consumption, which cannot be gated in order to maintain program retention. The data memory, however, can be power gated under CPU control during sleep, and the resulting power consumption of the latter is merely the result of the power gating transistor leakage. Lastly, there is an almost negligible CPU leakage when power gated by the AVS DC/DC converter. In this way, an overall sleep power of 1.55μW can be achieved.

2.4.2 Comparison to other systems

Table 2.1 compares Bellevue to other ULV micro-controllers of a similar class, implemented in the same technology and with similar features. Bellevue is the most aggressive in terms of voltage scaling, being able to function at just 0.37V, yet achieves the highest clock frequency at 50MHz, and has more than double the on-board SRAM memory capacity.

It is interesting to note that all the proposed architectures are RISC-based, and therefore have a limited instruction set that lends itself well to a stream-

lined physical implementation. They are also implemented in an advanced 65-nm technology, which validates that choice for ULP applications.

The choice of peripherals, however, varies significantly from one to another, highlighting the importance of task-related hardware accelerators. For instance [39] implements dedicated cryptographic accelerators that help it interact securely with a network, while [31] and [78] have a TDC and ADC respectively, for dedicated sensor interfacing. DMA is also a popular choice, since this allows processors with more complex bus structures to offload basic data copy or move operations, allowing the processor to either remain idle, or perform other tasks in parallel.

Overall, Bellevue, with its 50MHz operation and large memories capable of operating within the same power constraints as these processors, appears to be a good choice for processing-intensive low-power applications.

2.4.3 Comparison to sensor-only systems

Finally, the original question still needs to be answered; is on-board processing more power-effective from the point of view of the WSN than simply offloading all processing to a more capable node?

Consider the target application of image analysis, where an image is processed in such a way as to extract basic information from it, for example simple movement detection through edge detection. There are two main ways of doing this; either process the data directly on-board, or send all data to a remote node and perform the processing there, for effectively zero power when considered from the point of view of the WSN system. On the one hand, therefore, the local sampling and processing power must be considered, whereas in the second the sampling and data transmission costs must be considered. Sensing power is the same in both cases, and so is not taken into account in this calculation.

In the first case, the total energy required is shown in Fig. 2.9. Under the best conditions, the energy required for Bellevue to process the $128 \times 128 \times 8$-bit pixel array is 5.424μJ and is done in 14ms. The result of this operation can be transmitted for a much lower cost due to its reduced informational content, stored, or, depending on the application, even be used directly on-board thereby not requiring any transmission.

In the second case, the $128 \times 128 \times 8$ bits of data need to be transferred to a remote node via the wireless link. In order to provide an unbiased comparison, the energy overhead required to encode the network packets is not considered, the best possible case since networks on such open radio frequencies often have complex packet encoding procedures in order to resist collisions and interference. Currently the best low-power commercial (COTS) wireless radios are Bluetooth LE, ZigBee, and ANT, assuming the target node is within a 15m radius of the sensor.

Table 2.2 compares different COTS radio options. As can be observed, transmission of an image using a commercial radio costs 131kbits and requires 0.5 seconds at 250kbps, during which time the transmitter is active and consuming

Table 2.1.: Micro-controller characteristics and simulation performance compared to state-of-the-art micro-controllers in the same class.
† Simulated data at 25°CTT process corner.
* Excluding DC/DC conversion inefficiencies.

	Bellevue	**Bol**, JSSC, 2013 [31]	**Ickes**, ESSCIRC 2011 [78]	**Luetkemeier,** ISSCC, 2012 [39]	**Tang**, EDSSC, 2013 [83]
CPU	32bit MIPS-1 compat.	16bit MSP430 compat.	32bit ReISC	32-bit VLIW RISC	32-bit Spark RISC
Data format	SIMD 1×32bit, 2×16bit, 4×8bit	Scalar 1×16bit	Scalar 1×32bit	Scalar 1×16bit	Scalar 1×32bit
Technology	65nm LP/GP CMOS (dual core oxide)		65nm LP CMOS		65nm CMOS
Memory	32kB PM+ 32kB DM	16kB PM+ 2kB DM	8kB PM + 8kB DM	16kB PM+ 16kB DM+ 0.6kB SubVt	0.5kB PM+ 0.5kB DM
Cache	128B I$ + 128B D$	64B I$	128B I$ + 128B D$		
Clock [MHz]	50	25	1.65	4.2	10
Vdd [V]	0.37 CPU / 1.0 MEM	0.4 CPU / 1.0 MEM	0.6	0.5 CPU / 1.0 MEM	0.45
Peripherals	AVS, CIS, DC/DC, multiplier, divider	AVS, TDC, DC/DC, multiplier	DMA, ADC, JTAG	AES, CRC, multiplier, divider	NOR Flash SPI, DMA.
CPU+$ [μW/MHz]	7.1 - 8.2 (5.0-5.9 no DC/DC) †	3.5	6.5*	~18*	
System [μW/MHz]	7.7 - 12.0 †	7.0	10.8*		70
Standby [μW]	1.55 †	1.7			
Core area [mm^2]	1.05 (w/o CIS, PMU)	0.42	~1.0	~1.0	0.85

Table 2.2.: Selection of low-power COTS radio links for WSNs, indicating the required energy to transmit a sampled $128 \times 128 \times 8$-bit image.

Vendor & Model	**Speed [kbps]**	**Voltage [V]**	**TX [mA]**	**Energy [mJ]**
BlueGiga BLE112	100	3.3	27	117
Roving Networks RN41	300	3.3	65	94
Roving Networks nRF2AP2	20	3.0	15	295
Microchip MRF24J40MA	250	3.3	23	40
Medi et al., JSSC 2008 [91]	1,000,000	1.8	60	0.0141
IMEC, ISSC 2014 [92]	4,500	1	2.27	0.0661

power. This results in a very high power consumption for the operation, tens of milli-Joules, which is considerably more than the 5.424μJ required for on-board processing.

This is reduced somewhat by employing state-of-the-art research radio solutions [91, 92, 93], which are able to reduce the power consumption down to a few tens of micro-Joules, which is still higher than that required for on-board processing. The data processing speed is now, however, comparable to the 14ms ($\approx$ 10Mb/s) required by the processor, thus allowing realtime processing of the sensed image data (30fps maximum). Nevertheless, this does not take into account the limited range of transmission and the potentially lengthy radio synchronisation sequence, which induces a power and performance overhead.

If in addition to this the security of the data, the complexity of the wireless protocol, network management, and other such tasks are to be considered, the on-board processing option is attractive from a energy and throughput perspective. On-board processing also enables a myriad of possibilities, such as buffering or image processing for contrast improvement, automatic white balance, HDR, and possible intelligent analysis before transmission, applications which would otherwise be impossible due to the high power cost of transmission and limited data exchange rates.

All things considered, therefore, the choice of having a powerful on-board processor is justified in view of the energy requirements and functionalities expected from a modern WSN.

2.5 CONCLUSION

This chapter presented the Bellevue core, a MIPS-1 instruction set compatible SIMD 50MHz processor capable of operation at 0.37V, which together with a PMU, AVS and CIS make up the SunPixer SoC. Through a combination of

low-power techniques, both at the circuit design level and at the physical implementation level, simulation results show a low overall active power consumption of 7.7μW/MHz, and a low-power 1.55μW sleep mode. This result is achieved despite the large memories and increased data-processing capabilities made possible through a faster operation speed, SIMD mode, and a dedicated hardware acceleration for multiplication and division operations.

The power consumption of the system is also within the capabilities of typical energy harvesters (Section 1.4), assuming an appropriate duty-cycling is used when the source is limited. For instance, if taking an image and processing it takes less than 100ms, from a sleep state and back, then the effective duty-cycle for taking an image every second is less than 10%. In actual applications this duty-cycling can obviously be adjusted depending on energy reserves, harvested power levels, or based on detected events.

When designing this processor a choice was made to employ a 5-stage pipelined, dual-bus, SIMD architecture, and the question remains as to whether this was a good decision. As has been shown, processing speed is critical in such applications in order to minimize the overall energy. Employing a 5-stage pipeline largely allowed the increase of the system clock to 50MHz whilst operating at 0.37V, which is one of the major achievements of this design. A 3-stage pipeline would not have been able to achieve this operating frequency without raising the supply voltage, and so would have suffered from an energetic perspective. More than 5 stages would not have allowed the separation of tasks into dedicated stages, and the overall simplicity and elegance would have suffered as a result, without offering as significant an improvement in performance as from 3 to 5 stages. Furthermore the physical implementation must be considered; one of the reasons the clock frequency can be raised to 50MHz is the relatively short path distance, in particular paths to the memories. The 5-stage architecture has a logical flow with instructions being fetched from the PMEM in the first stage, to the DMEM being read and written in the 4th and 5th stages respectively. This matches well with the PMEM-CPU-DMEM physical layout employed in this design, and allows shorter memory access paths.

The choice of a dual-bus architecture for data and instructions follows a similar reasoning and justification. On the one hand, the presence of two buses allows wait-free memory accesses in the common cases where data is read or written at the same time as an instruction fetch, thereby increasing speed. And from a physical layout perspective, the two buses can follow different routes to the memories, again minimising path distance. Furthermore, the dual buses allows different caching mechanisms for both, and can therefore be optimised for each type of request: the instruction memory cache better handles sequential accesses and loops, whereas the data cache (scratchpad) is a random-access memory under software control. The choice of a Harvard architecture is therefore in line with the other design decisions of the architecture.

Lastly, in a similar vein, the SIMD mode allows faster processing of reduced-width data. This is a relevant choice for use in a WSN, where sensor data is often limited to 8 or 16bits, but the processor itself may require more for the higher-

level tasks such as managing a wireless network. This choice proved useful in the application targeted by this chapter, in that CIS data could be processed up to 4× faster. While the choice of having 4 8-bit ways was made in order to have no redundant hardware in any of the SIMD modes, more ways could conceivably be added should the parallel processing requirements increase in other applications.

While the choice of architecture did influence many other choices, and while there are doubtless many other solutions which might also have worked, the 5-stage pipeline choice allowed the objectives to be reached in an elegant manner, and was thus, all things considered, a valid choice.

Overall, this design showed the wisdom in placing a capable processor on-board for data-processing in order to reduce the overall energy consumption, and demonstrated the feasibility of operating a processing-intensive micro-controller within the constraints and limitations of an energy-harvesting system.

CHAPTER 3

PSEUDO-SYNCHRONICITY: AN ORTHOGONAL DATA-DEPENDANT OPERATION SPEEDUP TECHNIQUE

With the advent of mobile electronics requiring ever more computing power from a limited energy supply, there is a need for efficient systems capable of maximizing this ratio. Architectural enhancements must therefore be designed to enable high performance, all the while maintaining the power advantage. The technique proposed in this chapter allows the acceleration of combinatorial circuits beyond the performance generally achievable by conventional synthesis and timing closure, by exploiting the data-dependent delay variations inherent in such circuits. Through the automatic insertion of transition detectors within the target circuit, the progress of operations underway can be monitored and prematurely completed, thereby increasing the operation speed from the worst towards the average case. In addition, a synthesis flow is proposed to increase the proportion of fast paths, thereby increasing the technique's impact. The proposed technique was applied applied automatically to a series of benchmark circuits, and synthesis results show it to achieve an average performance increase of 29% over conventional synthesis, which must ensure deterministic timing closure. In spite of a power 21% overhead caused by the addition of the hardware required, this increased speed allows shorter wake periods and is thus able to lower the circuit's overall active power by 14%.

Contents

Associated author publications

JP2. F. Botman, D. Bol, J-D. Legat, and K. Roy, "Data-Dependent Operation Speed-Up Through Automatically Inserted Signal Transition Detectors for Ultralow Voltage Logic Circuits," in *IEEE Trans. Very Large Scale Integr. Syst.*, 2014.

CP5. F. Botman, D. Bol, and J-D. Legat, "Data-dependent operation speed-up through automatically-inserted signal transition detectors for ultra-low voltage logic circuits," in *Proc. IEEE Subthreshold Microelectronics Conference*, 2012.

While the preceding chapters presented ultra-low power micro-controllers capable of operating under the constraints of harvested energy operation, progress was mostly achieved through power-conscious circuit design and advances in the underlying technology. Despite these improvements, the technical challenge of developing a system capable of running on even more limited power sources and achieve ever-increasing expectations of functionality remains.

Having presented advanced low-power designs in Chapter 2, the question as to whether there might exist other forms of circuit optimization that could achieve further power reductions must be asked. This chapter proposes an orthogonal design technique aimed at lowering a circuit's power consumption, and was inspired by two observations made during the development of Bellevue; firstly, the overall power of a circuit is strongly linked to its operating speed T_{op}, and by accelerating a circuit it is able to sleep more and thereby save power, specifically through a reduction of the integrated leakage current $E_{leak} = V_{dd} \times I_{leak} \times T_{op}$. Secondly, the inherent variability of ULV circuits imposes larger timing margins based on the worst-case occurrence in order to guarantee functionality, with this trend set to become even more problematic at more advanced technological nodes.

Taken together, the technique proposed in this chapter aims to lower the power consumption of a circuit through the acceleration of operations, by allowing it to use timings based on encountered data rather than worst-case scenarios, while guaranteeing functionality in all cases. Set in the context of a slow submodule inserted into a larger system, the proposed technique allows the time per operation to be reduced from the worst-case timing to the average-case, depending on the data actually encountered during operation and the slow or fast paths required to perform the calculation on that data, by completing the operation as soon as the output result is stable and correct. The proposed technique is thus able to either lower the required energy per operation for a given performance, or yield better performance than that achievable by conventional synthesis.

The remainder of this chapter is organized as follows. Section 3.1 lays out the technique, Section 3.2 describes the dedicated cells required, and Section 3.3 presents simulation results of its application. Section 3.4 describes a change in the synthesis procedure that can increase the impact of the proposed technique, and conclusions and perspectives are drawn in Section 3.5.

3.1 PROPOSED TECHNIQUE

Every combinatorial circuit is composed of a number of paths from the inputs to the outputs that interact together in some way through logic operators. While some of these paths can be very short, for example a zero-test on an input leading to a fixed output value, others can be very long. Since the time required to process an operation depends on the length of the paths taken, the delay of a circuit can vary significantly as a function of its input data. This time differential can be exploited.

In order to accelerate the operation underway, the proposed technique attempts to separate the fast paths from the slow paths, and terminate the operation prematurely in the former case. In doing so, the difference in time between the slow and fast paths is no longer wasted, and thus ensures a higher average performance and a lower energy per operation. While asynchronous circuits do this intrinsically, they also have the large overhead of their synchronization signals. This technique does this synchronously, albeit on an arbitrary clock positive edge, thus introducing the notion of pseudo-synchronicity. This can be implemented either as a fast clock, or require that the target combinatorial circuit takes a number of clock cycles to complete, and is multi-cycled within the target system. In such a multi-cycled block, where N is the number of cycles required for its completion, this technique aims to complete in $1 \leq R \leq N$ cycles, thereby accelerating the system whenever $R < N$.

There are a number of ways to do this. In certain applications, such as image processing, data truncation is acceptable and incorrect results can be accepted for a few rare cases. In this case, R can be set to a value smaller than N at design-time in order to achieve satisfactory speedup and adequate error levels [94, 54]. However, in many cases incorrect results are not acceptable, in which case there are two main approaches; either the output value can be identified as being valid, or activity within the circuit can be detected [95]. It is assumed that for as long as there is activity, the output is not yet ready. Some approaches use pre-calculated tables to determine completion time based on the inputs [96], whereas others such as Razor [55] function by detecting timing errors after they have occurred, and correcting them using specially-designed latches.

In the case of the proposed technique, the target circuit block is considered as a modifiable netlist rather than an immutable block. Circuitry is inserted at regular time intervals along the combinatorial paths, in order to obtain feedback as to the progress of the operation. As shown in Fig. 3.1, the block is divided into $N-1$ 'timezones', at the end of which transition detectors (TDs) are placed. No TDs are inserted at the output of the block. Once a signal enters a timezone, it must activate the TD located at the end within the same clock cycle. This is ensured by construction, using timing closure constraints. In this way, at the end of a cycle and based on the state of the TDs, it is possible to determine whether the corresponding timezone was active or not. In contrast to conventional pipelining where temporal isolation is enforced, the detectors are non-intrusive and allow the signal to pass through unhindered and continue propagating. Thus a signal entering a timezone is guaranteed to activate the TD at the end of that timezone, but may also continue on to the following timezones within the same cycle, should only fast paths be used. As a result, for as long as TDs are activated, the operation is still in progress, and at least another cycle will be required to complete the operation. This is in contrast to Razor [97], where the latches prevent propagation beyond strict limits, and is essentially wasted.

Table 3.1 summarizes the potential speedup induced by this technique, in the case of a 3-timezone circuit. For very short paths, the signal is able to reach the output register within the first cycle, thereby activating all transition detectors

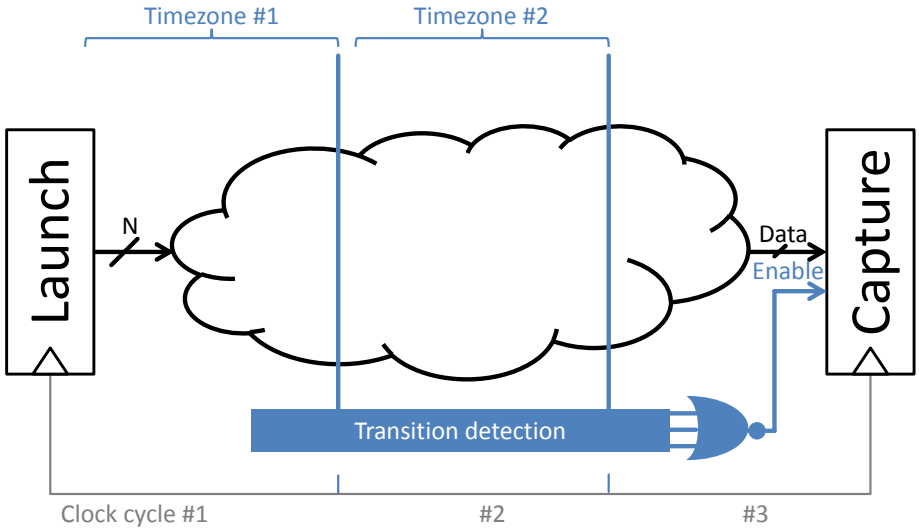


Fig. 3.1.: Application of the proposed method to a multicycle 3 combinatorial function bounded by two register blocks.

Table 3.1.: Effective number of cycles to complete operation, given a specific data-dependent path speed. For each cycle, the activated TDs are listed. On each cycle following no TD activation, the operation is terminated prematurely. Example for a 3-cycle block.

Path	**#1**	**#2**	**#3**	**#cycles to complete**
x3	1, 2	-	*completed*	2
x2	1, 2	2	-	3
x1	1	2	-	3
(no change)	-	*completed*	*completed*	1

in its path. Although the operation is complete, this technique will be unable to verify completion until the next cycle, so will take two cycles overall. For paths almost as short, the data just reaches the last detector at the end of the first cycle, but still has to propagate through the last timezone. By timing closure this will take at most one cycle, by which time no more detectors will have activated, triggering the end of the operation. In the longest case, timing closure ensures that the detector at the end of each timezone is activated, but will require all three cycles to complete the operation. In the special case where the combinatorial value does not change from that of the previous operation, no

detectors will be activated, and the operation will be complete on the next clock edge. Since we only use combinatorial circuitry, and only standard logic gates, the circuit is already in steady-state and has thus already completed.

A similar technique used to minimize energy consumption in ULV circuits is superpipelining [98]. Whereas in that instance a large number of intermediate registers are inserted into the circuit in order to minimize datapath length and thereby decrease inactivity, the proposed technique instead inserts TDs. This allows signals to propagate past synchronization barriers, thereby no longer being constrained by pre-established timing closure, and allows a shorter latency under favourable conditions.

Given the staggering number of signals at such intermediary stages in the combinatorial block, the area and switching power overhead of register insertion is significant, compared to the comparatively smaller and more efficient TDs that induce a smaller overhead. Contrary to other techniques that pre-analyze the different paths in order to isolate multicycle-inducing operands [96], this method functions at run-time based on the actual data, thereby not necessitating expensive condition tables or pre-computation. It also differs from systems such as Razor [55], that function on a pipeline rather than a combinatorial block and that detect errors after latching, rather than predicting the end of operation as is done in this technique. And while the completion of the operation could be sensed through other means, as outlined previously, this method lends itself reliably and with a manageable overhead to low-power, ULV systems.

Overall, this technique allows the acceleration of circuits by allowing data to propagate through the TDs, in contrast to superpipelining and conventional register-based solutions, and by terminating early if the encountered data uses paths faster than the worst case. In theory, considering an equi-probable distribution of path lengths, an effective speedup of around 25% can be expected by using this technique (see Table 3.1), though this will vary depending on the paths actually encountered during the operation.

3.2 DEDICATED CELLS

In order to apply the proposed method efficiently, two types of custom cells must be designed. At each timezone, a large number of signals must be monitored for activity. To do this, a dedicated TD cell design is presented that is both fast and small. And given that all the outputs of the TDs must ultimately be reduced to a single bit, more efficiently than using an OR-tree, a group of numerous-input OR cells must also be developed.

3.2.1 Dedicated TD cell

In view of the area and power requirements for the TD cell, a dynamic circuit was chosen (Fig. 3.2). The signal to monitor enters the cell at $D0$, and is passed through delay generators to produce inverted and delayed images of itself, $D1$ and $D2$ respectively. Similarly the reset signal, in this case the clock, enters the

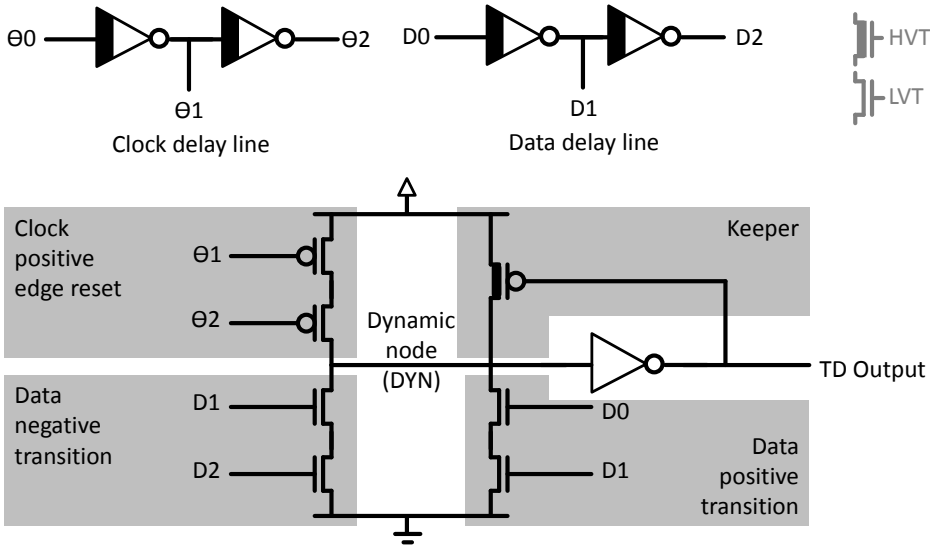


Fig. 3.2.: Dedicated transition detection cell.

cell via Θ0 and inverted and delayed images of itself are produced as Θ1 and Θ2. The latter could obviously be generated separately and shared for a group of TD cells.

The cell has an edge-triggered reset, and detects both positive and negative transitions. It consists of a dynamic net (DYN), maintained by a small high-V_T keeper transistor, that is either charged by the clock positive edge, or discharged by either edge of the data signal. An output inverter ensures sufficient drive to propagate down an OR-tree combining the different TDs to where the signal is needed. The cell is implemented on ST $65nm$ GP CMOS and characterized for $V_{dd} = 0.35V$. This technology offers many flavours of V_T which are useful for generating the delays. For this reason, most of the transistors are fast low-V_T in order to minimize the internal propagation time. The delay elements and keeper are, however, implemented using slow high-V_T transistors, since a large propagation time and low driving force are required there. The detector was validated through Monte-Carlo ELDO-SPICE simulations (Fig. 3.4).

Although no actual layout of this cell was performed, instead using an estimated cell area, in-circuit data yielded the following power consumption for these cells (Tab. 3.2). The cell was characterized using *Cadence Encounter Library Characterizer* for later insertion into the synthesis flow.

3.2.2 Large-input OR cells

Because of the high number of TD cells in each timezone, it is necessary to efficiently reduce their output down to a single 'timezone active' bit. One way

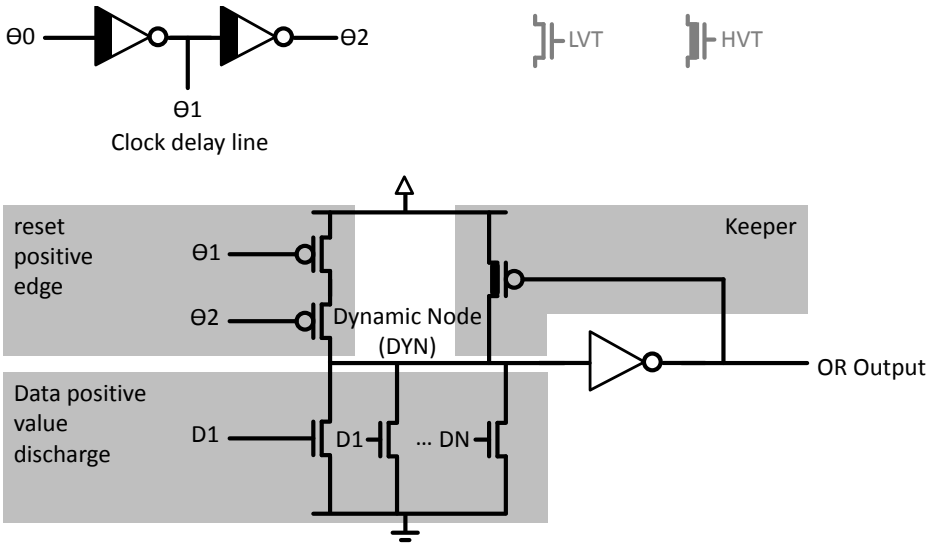


Fig. 3.3.: Dedicated multiple-input dynamic OR cell.

this can be done is by constructing an OR-tree down from the TD outputs. However, because of the number of signals to reduce and the induced switching and delays, the resulting circuit in synthesis would consume too much power and is too slow for this application. Therefore, a series of dedicated dynamic OR cells with 5, 10 and 20 inputs were designed (Fig. 3.3).

In essence, these are very similar to the TD cell described in the previous section. The data delay line is no longer necessary, since the cell works on level-activation rather than transitions, and the clock 'reset' signal can also be shared among cell groups. The dynamic net can be discharged by any of the input signals, and through its inversion at the output buffer stage it implements the OR function. The strength of the keeper transistor and discharge lines must be carefully sized in order to ensure proper operation. This cell was validated through Monte-Carlo SPICE simulations (see Fig. 3.4) and characterized automatically through *Cadence ELC*. The performance of each cell is reported in Tab. 3.2.

One point of concern is the stability of the internal dynamic node, since it is prone to destabilizations while in its active state due to subthreshold leakage currents and local process/dopant fluctuations (as examined in [99]). If the leakage current of the combined discharge transistors is too great, the dynamic node will be grounded parasitically in the absence of transition detection, causing an invalid output. Similarly, if the keeper cell is too strong, the dynamic node might not discharge in the event of an actual transition, again causing an invalid output, or delay the output excessively. A careful balance must therefore be struck between stability and switching speed, and the transistors must be sized accordingly. Fig. 3.4 shows a 1000-sample Monte-Carlo simulation of the dynamic

Table 3.2.: Performance of the design cells, as reported by SPICE simulations. Implemented on 65nm GP CMOS at 0.35V.

Cell	**Detection delay [ns]**	**Leakage power [nW]**
TD-cell	0.224	2.785
5-input DOR	0.270	2.521
10-input DOR	0.309	2.554
20-input DOR	0.346	2.743

node within the 20-input dynamic OR circuit, the most problematic case. No occurrences of voltage drop below the switching threshold were observed while in the 'active' state, nor of voltage rises while in the 'inactive' state. The chosen transistors share the same gate length $L_G = 80$nm, but the reset and keeper transistors have an 8× and 10× increased width, respectively, because of weak PMOS transistors in 65nm GP CMOS at 0.35V.

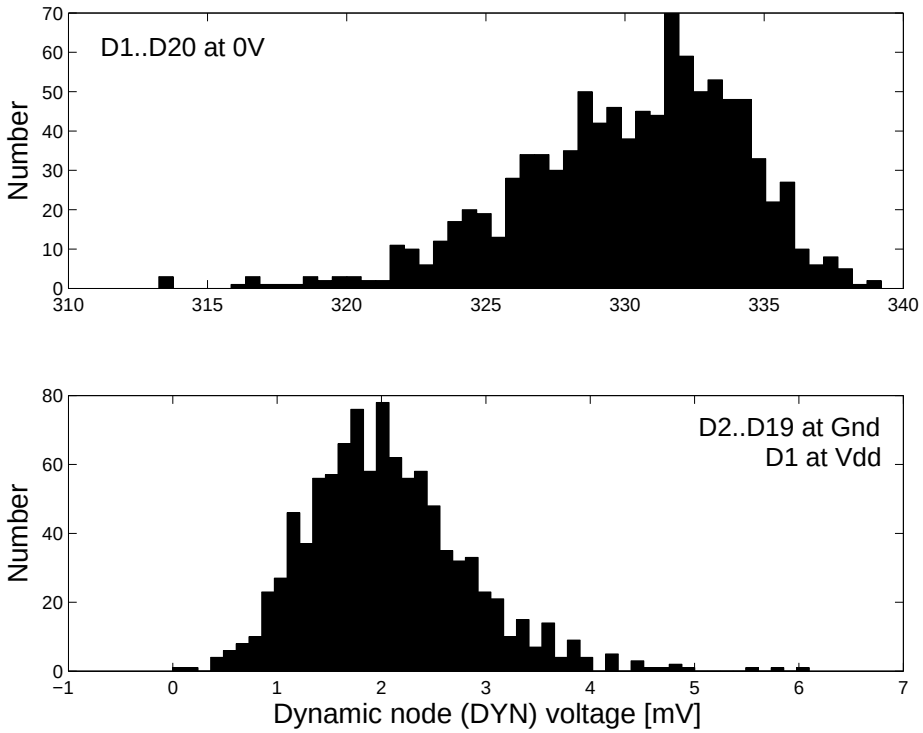


Fig. 3.4.: 1000-sample Monte-Carlo simulation of the voltage drop and rise of the internal dynamic node within the 20-input dynamic OR cell, showing no occurrences of shifting beyond the toggling threshold.

3.3 IMPLEMENTATION AND RESULTS

This section reports on the impact of the proposed technique on a number of test-cases. The implementation of the method, whose application is performed fully automatically, is described in Section 3.3.1, while the test-cases are presented in Section 3.3.2. Finally, the simulation results achieved are reported in Section 3.3.3.

3.3.1 Implementation

The technique proposed in this chapter can be applied automatically through a single program, that calls the appropriate sub-scripts for each part of the process, and is configured via a small number of user-provided inputs. All input HDL files, Verilog or VHDL, for the circuit under test should be grouped together in a single isolated directory, and the system period and number of timezones should be specified. An overview of the flow is given in Fig. 3.5.

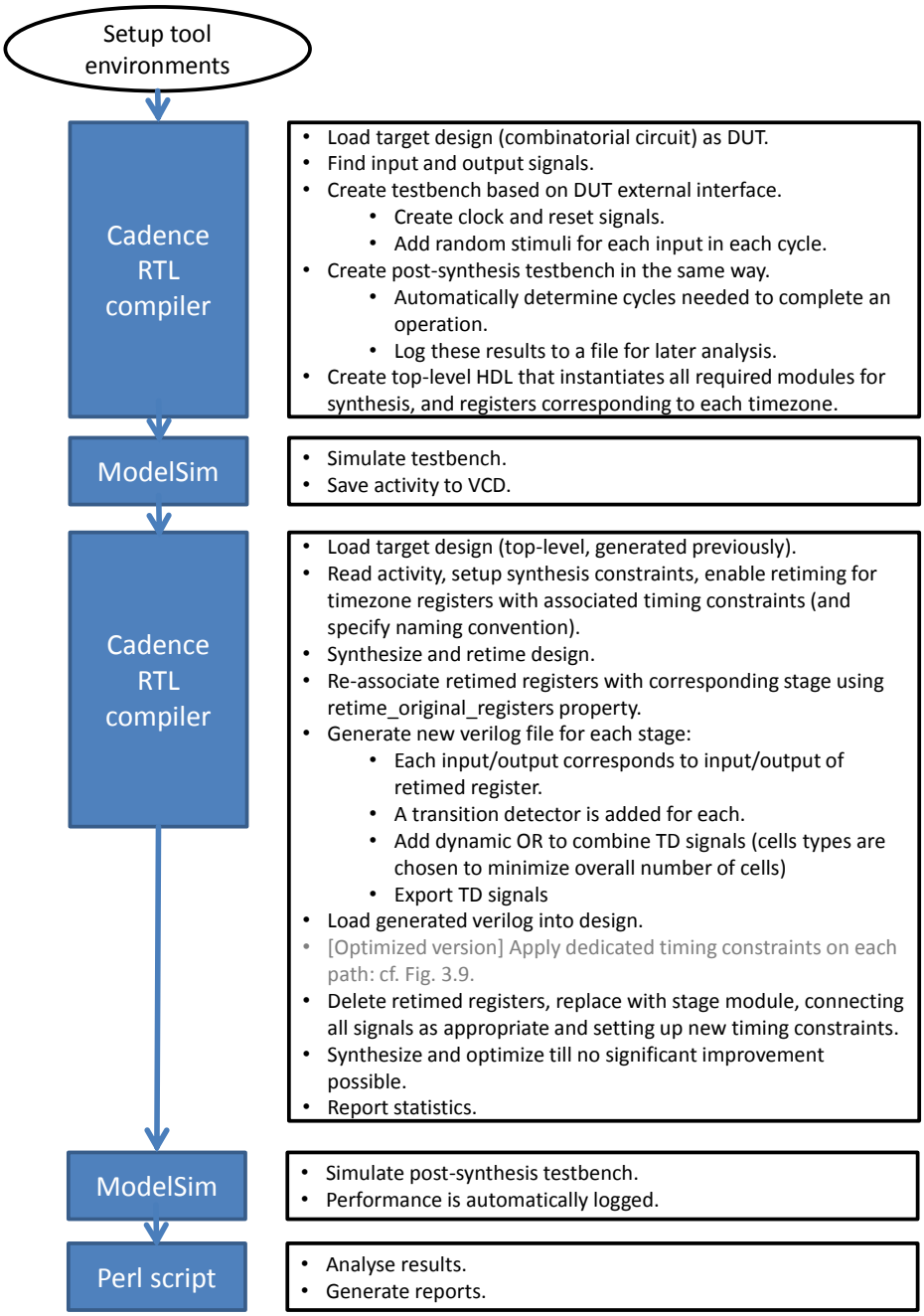


Fig. 3.5.: Synopsis of the operations required to apply the technique in the associated CAD tools.

The first task is to obtain the circuit activity, which is required in later steps in order to perform accurate power estimation. In order to achieve this, the program loads the required circuit into *Cadence RTL Compiler*, and uses its circuit-parsing functionality to write a pre-synthesis testbench, a post-synthesis testbench, and a top-level container netlist that binds the target circuit between register blocks at the inputs and outputs. These testbenches not only instantiate the circuit under test, they also generate the input, clock and reset signals, keep track of the activity, and, in the post-synthesis testbench, automatically track the efficiency of the applied technique. Next, the pre-synthesis testbench is loaded into *Modelsim*, is simulated using random inputs, and the circuit activity is extracted from that.

The next task is to apply the proposed technique. To achieve this, the circuit is loaded into *Cadence RTL Compiler* and the top-level netlist generated previously has a number of registers appended to each output that corresponds to the number of timezones requested. Constraints are set, and the registers are retimed to place them at regular intervals throughout the circuit, while guaranteeing timing closure between the different stages. Once these are correctly positioned, their corresponding timezone is identified, and the registers are replaced through netlist modification by a transition detector. All TDs of a given stage are then connected to an OR-tree composed of the custom cells described previously, in such a way as to minimize the depth of the tree by using the largest suitable OR cells available. Once all stages are thus modified, their outputs are brought together to form a single 'ready' signal, which is outputted at the top-level. All the different stages are moved to separate hierarchical instances, enabling trivial power extraction of the overhead caused by the technique. New constraints are then set (since there is no longer any inherent timing closure between stages), and the circuit is optimized.

Finally, statistics about the circuit such as power, area, and timing are extracted, and the resulting netlist with SDF delay annotations is simulated by *Modelsim* using the post-synthesis testbench generated previously. For each random input set, the system monitors the number of cycles required till completion, using the TDs as completion indicators, and over time builds a representation of the average performance of the modified circuit.

3.3.2 Testbench circuits

The following four circuits were adapted to be used in an entirely combinatorial fashion, and are intended to be representative of certain computational scenarios. In order to maintain simplicity, none of these are pipelined although the proposed technique could also conceivably be used in that context, possibly exploiting the activity indicators for better, finer-grained control or in the context of a wave pipeline. The circuits were synthesized at $0.35V$, which is close to the minimum energy operation point [31], in order to minimize the power consumption and also maximize the delay variation between paths in the circuits.

CRC The smallest and simplest circuit is a 16-bit Cyclic Redundancy Check generator used in PCI express. It is mostly composed of XORs, and is representative of low-resource calculative scenarios. Unoptimized, this circuit employs roughly 250 cells.

LUT A commonly-used circuit in combinatorial designs is a lookup-table, where as a function of a given input key, an arbitrary output value is returned. The selected circuit is a 1000-entry LUT, with a 10-bit selector, and 20-bit output value. The circuit was automatically generated, and the values stored are random with no discernible pattern. Uses ~3500 cells, unoptimized.

MAD A multiply-divide unit from a 32-bit microprocessor. It handles a number of different modes of varying complexity, multiplication or division of 8-bit, 16-bit or 32-bit numbers, which give rise to paths of widely differing lengths. Uses ~7000 cells, unoptimized.

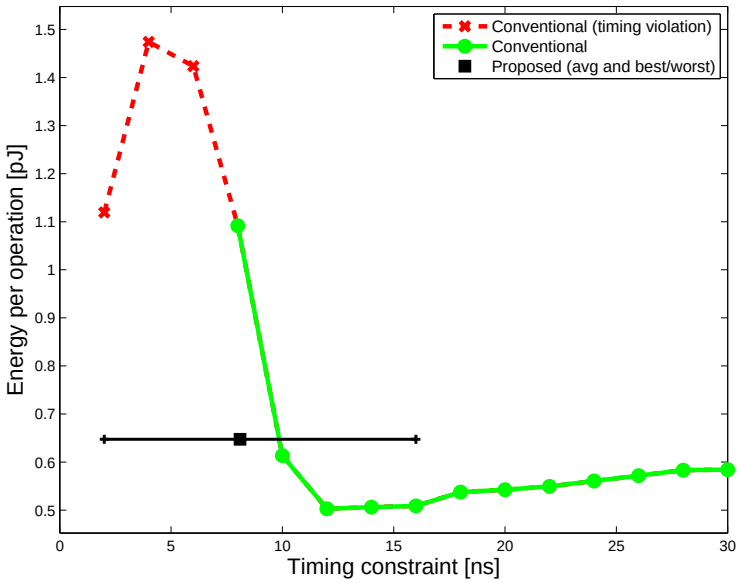
FP A 32-bit floating-point multiplication circuit. This is representative of larger circuits, which can be found in co-processors, which can greatly benefit from the proposed technique. Uses around ~2000 cells.

All of these circuits were first synthesized using conventional methods, and simulated under identical circumstances to those of the applications of the speedup technique, and serve as the baseline for the energy per operation under given timing constraints.

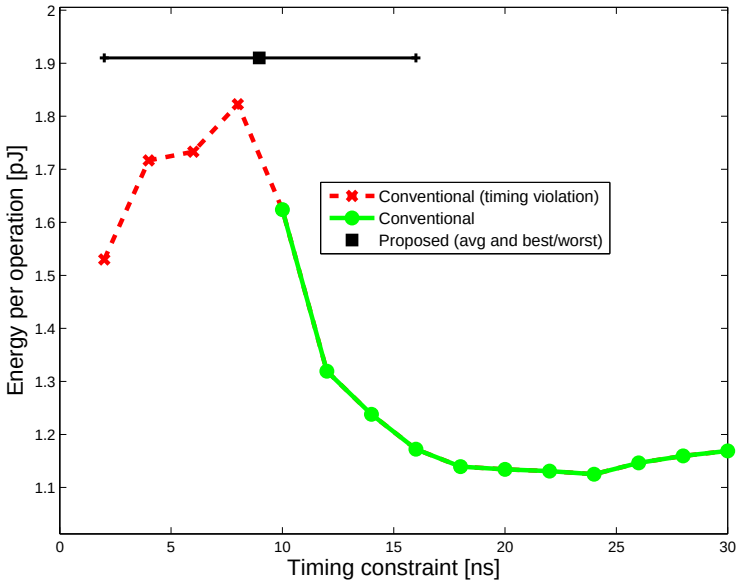
3.3.3 Simulation results

The simulation results for each of the test circuits are plotted in Fig. 3.6. For each circuit, a baseline is first established on the same circuit simulated under identical circumstances by synthesizing over a variety of system clock timing constraints, and the best achievable performance is noted. On each figure, the equivalent performance and power consumption of the circuit using the proposed technique is shown. The black square indicates average performance, whereas the associated bars indicate the range in performance observed.

It is well known that there is a power trade-off when optimizing a circuit during synthesis; if it is necessary to operate more rapidly, more or larger cells are required, thereby driving up the power used in the circuit. This is visible on the left of Figs. 3.7, where the circuit has performed full optimization of its cells due to the stringent timing requirements, yet still fails to meet those set. As these constraints are loosened (rightwards), the timing becomes achievable and the area necessary progressively decreases as less optimization is required, and with it the power. This trend can be clearly observed in the baseline results in Fig. 3.6. Achieving guaranteed high performance in a circuit is therefore very costly in terms of area and power, but the proposed method achieves similar performance for a far lesser cost by accepting that certain rare paths will not be

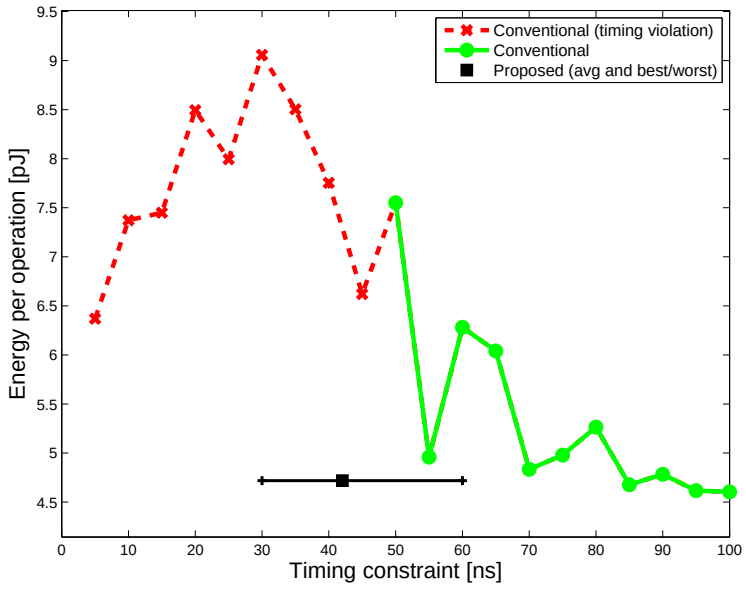


(a) CRC

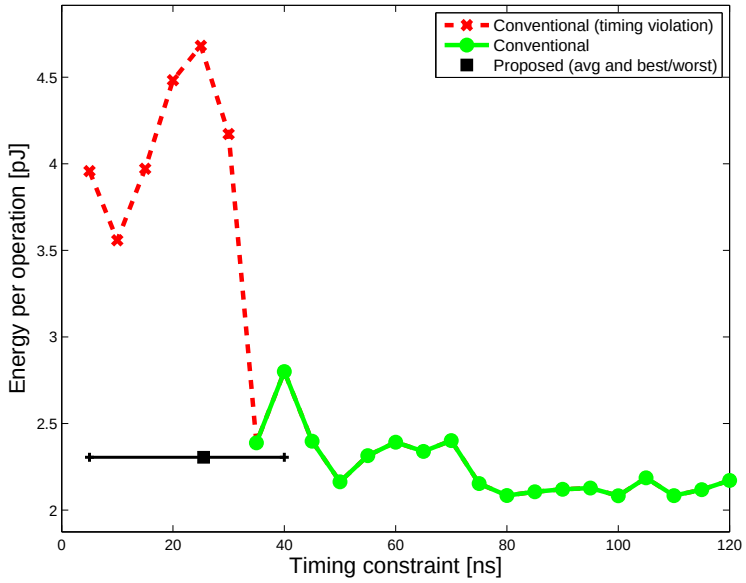


(b) LUT

Fig. 3.6.: Energy per operation and timing for each benchmark circuit, showing the best-performance application of the technique (average case shown by black square) as compared to the baseline synthesis result of each.



(c) MAD



(d) FP

Fig. 3.6.: Energy per operation and timing for each benchmark circuit, showing the best-performance application of the technique (average case shown by black square) as compared to the baseline synthesis result of each.

Table 3.3.: Synthesis results in $65nm$ GP CMOS at $V_{dd} = 0.35V$. For each circuit, the best result achievable in synthesis (before negative slack) is given, as well as applications of the technique at lowest energy and best performance, with a configuration specified by the number of timezones (T) and clock resolution (period).

Circuit	Configuration	Timing [ns] (min,	mean,	max)	Total Energy [pJ/op]	Overhead [pJ/op]	# TDs
CRC	Conventional (best speed)		8.0		1.092	-	-
	4T, 4ns period	4	9.98	16	0.357	0.009	305
	8T, 2ns period	2	8.10	16	0.648	0.013	534
LUT	Conventional (best speed)		10.0		1.624	-	-
	3T, 5ns period	5	12.47	15	0.963	0.025	689
	8T, 2ns period	2	8.96	16	1.910	0.066	2534
MAD	Conventional (best speed)		50.0		7.551	-	-
	5T, 20ns period	40	62.20	100	3.217	0.362	1990
	6T, 10ns period	30	42.03	60	4.719	0.430	3506
FP	Conventional (best speed)		35.0		2.389	-	-
	5T, 10ns period	40	48.09	50	1.584	0.067	476
	8T, 5ns period	5	25.50	40	2.305	0.072	963

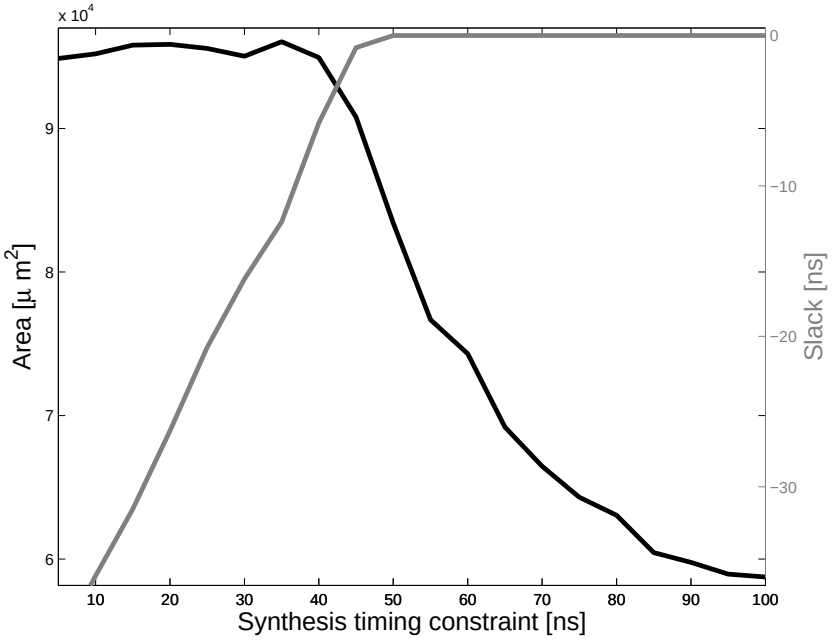


Fig. 3.7.: Area evolution of the MAD circuit as a function of synthesis constraints, and the resulting (negative) slack.

able to achieve the desired performance, although on average the circuit probably will. The proposed technique is therefore a good way to lessen the energy required to achieve high performance in a circuit.

Table 3.3 reports the raw values achieved. The energy overhead, as reported in the table, is the total energy consumed by the TDs and the corresponding OR-tree necessary for the application of the proposed technique, and generally does not account for more than a small percentage of the total energy. For each circuit, two configurations are given for the application of the technique; the first gives a lower energy-per-operation compared to the baseline circuit operating at the same average speed, and consistently achieves lower energy per operation for each testbench. This is largely due to the reason outlined previously, and the price for this decreased energy is the increased maximum time necessary to process the operation. By accepting this variation, better power results can be achieved. The second configuration, plotted in Figs. 3.6, yields an average performance that is, in most cases, better than that achievable under synthesis, showing an average of 17% increase with a best case of 37%. One notable exception is with the CRC circuit, where there is simply not enough internal circuitry to introduce a significant amount of fast paths: the logic depth of the critical path is only 19 cells, compared to 99 cells for the MAD circuit. Overall for this performance

increase, the average energy reduction is 16%, with only an 18% increase in the worst case.

Specifically, consider for example Fig. 3.6a for a CRC operation. Starting from a loose timing constraint, traditional synthesis methods have no trouble in obtaining a good solution, with no particular optimisation, but the clock speed is limited to 33MHz. As the constraints is tightened, the energy result lowers somewhat as the operation takes less time to complete and there is not yet any added hardware to meet the timing down to about 12ns. At a 10ns timing constraint, things start to change; a lot of extra hardware needs to be inserted or the existing hardware upsized to achieve better speed, which explains the increase in energy in spite of the increased operation speed. This explodes at the 8ns constraint as the tools struggle to find ways to achieve timing closure, and are not able to beyond that. The application of the proposed technique in this instance yields a certain hardware overhead, resulting in an energy overhead compared to the loosely-constrained version, but due to its data-dependant nature is able to achieve better speeds for a lower equivalent energy cost. The range indicated by the black bar is crucial; for data allowing very high speeds, the energy consumed is actually lower than what would have been required by traditional synthesis. On the contrary, for slower data beyond 10ns, the energy is higher. In this instance the average (black square) is equivalent in speed to that achievable by traditional synthesis, in part because the shallow logic depth of only 19 cells does not allow a significant number of fast paths to form. However, for an equivalent performance, the energy is almost 40% better (at the 8ns timing constraint).

From the raw data for the CRC circuit found in Table 3.3, it can be observed that in both configurations the effective synthesis timing constraint is 16ns. As a result, as can be observed on the figure, this is a hard boundary and there are no paths longer than this. The two configurations for this circuit differ in granularity: 4ns in the first case, and 2ns in the second as shown on the figure. The result is that the first will have fewer timezones, and thus a lower overhead, but will be limited to 4ns path segments which results in a higher average timing. There is thus a choice to be made when applying the technique, and a balance must be struck between timing and overhead.

In Fig. 3.6b, the energy per operation is higher in both configurations for a table lookup operation. While the energy graph itself follows a similar trend, the energy required by the technique is now higher than the equivalent synthesis outcome, even if the average speed is this time beyond that achievable by conventional synthesis. The reason for this is the nature of the HDL design itself which contains a high number of parallel tables. Each of these signals must be provided with TDs, and there is no realistic way of combining them without affecting functionality, and thus the power overhead.

For the MAD circuit in Fig. 3.6c, the synthesis trend increases rapidly with the timing constraint tightening, showing that the circuit is difficult to optimize. However the 99-cell logic depth gives this technique plenty of opportunity to encounter fast paths, so the average performance is this time significantly better than the synthesis equivalent, and for a lower energy.

Fig. 3.6d shows a similar result, although this time the energy reduction is not quite so marked. The synthesis effort shows a relatively flat trend as the timing constraint tightens, but suddenly explodes once it is no longer able to meet it. This has, of course, no physical reality and is just a representation of the effort the CAD tool took to attempt to meet the constraint.

On the whole, by accepting a non-deterministic timing range for the circuit, it is therefore possible to achieve better performance often at much lower power cost than with conventional synthesis methods. And in all cases, better power-per-operation could be achieved in a less timing-focused configuration.

3.4 OPTIMIZED SYNTHESIS FLOW

Although the results from the application of the proposed technique as laid out in the previous section are already quite promising, it is possible to improve the circuit speed, albeit at the cost of a slight increase in energy. A circuit synthesized without any constraints, or loose constraints, will have an endpoint timing distribution curve resembling Fig. 3.8a. When optimized under a more stringent timing constraint, the slow paths will be accelerated, through cell upsizing, replication, arithmetic optimization, and such techniques, yielding a distribution graph resembling Fig. 3.8b. In this way, timing is met.

When the proposed technique is applied on the resulting circuit, since timing constraints are then effectively quite loose, most endpoints will fall into the later timezones (Fig. 3.8c, black curve). On the whole this will increase the

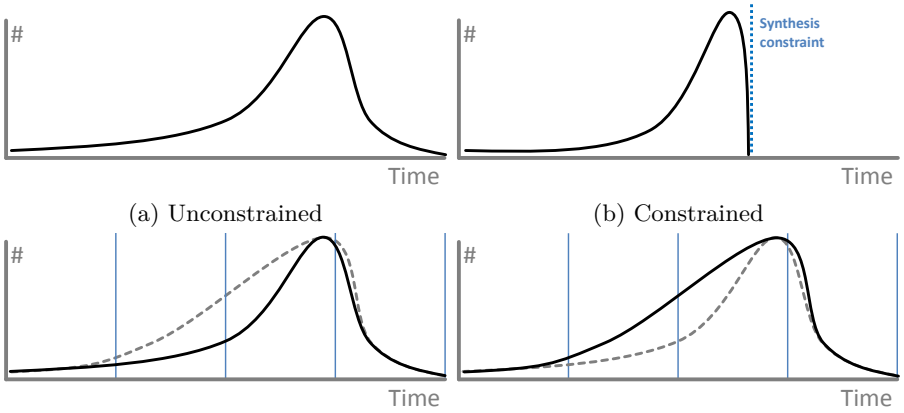


Fig. 3.8.: Representation of endpoint completion times, without any constraint, with a hard synthesis constraint, and with slow-path optimization. In (c) and (d) the dotted lines indicate the optimised and baseline counterparts of the distribution. In each case the distribution corresponds to a stylised representation of the endpoint histogram of the MAD circuit.

overall average speed, but there are still not many fast paths. If there could be a higher proportion of fast paths, the performance of the proposed technique could improve substantially.

Conventional synthesis optimizes the slow paths to achieve a certain performance, and may even downsize the cells composing the faster paths since there is in essence no advantage to their terminating early, and this may in fact save power. For this reason, synthesis tools strive to equalize the lengths of all paths to match the timing constraint set, and designers ensure their pipeline stages are balanced. Here, however, the opposite is required. Ideally, there should be many very fast paths, and a rare few slow ones, since this is where there is the greatest benefit in employing this method. CRISTA [96] achieved this separation by pre-calculating the employed path length and using pre-computed tables at runtime. This technique achieves a similar effect through the application of tailored synthesis constraints.

Since the entire process governing the application of the proposed method is automated, it is possible to insert an additional step (cf. Fig. 3.9). Before the retiming registers are replaced by TDs, the different endpoints can be analyzed, with results similar to Fig. 3.8a, and the fast paths can be identified. By adding a stringent timing constraint on each before optimization, these paths will be selectively upsized and optimized more than the slow paths, which in contrast to conventional synthesis will remain largely untouched. In this way, it is possible to shape the endpoint timing distribution, and achieve something resembling Fig. 3.8d. When comparing the two distributions, and their corresponding timezones, an effective increase in the number of slow paths falling into earlier timezones can be achieved. This, of course, unfortunately also comes with the power overhead increase of having to upsize a number of cells.

The results achieved by this method are presented in Fig. 3.10 and Table 3.5. Fig. 3.10c in particular clearly shows how the distribution of fast and slow paths have changed, since although the maximum and minimum timing points are the same as previously, the average has improved substantially. Overall, it can be observed that the performance of the technique combined with this optimization consistently match or improve on the results attained by the technique with no optimization, which in turn match or improve on the maximum performance achieved during conventional synthesis. The power overhead remains modest, although the energy now required to achieve the highest performance can be significant, and higher than that of the unoptimized technique. There remains, however, a configuration for each in which the energy per operation is still lower than for conventional synthesis at equal performance.

Table 3.5 reports the results achieved. In contrast to the results without optimization (Tab. 3.3), all circuits are able to perform faster than the best case in synthesis, even the CRC circuit. Whereas the baseline average timing improvement is 17%, with a 37% best-case improvement, the optimized flow is able to achieve a 29% average increase, with a 58% best-case. However the power cost is greater than that of the unoptimized technique, averaging 21% more than the conventional equivalent (although the worst case is a 48% increase) that must

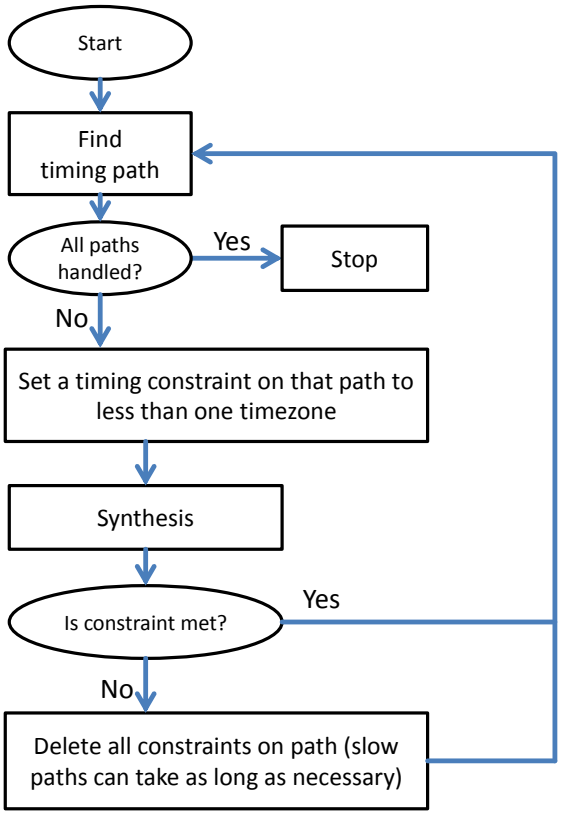
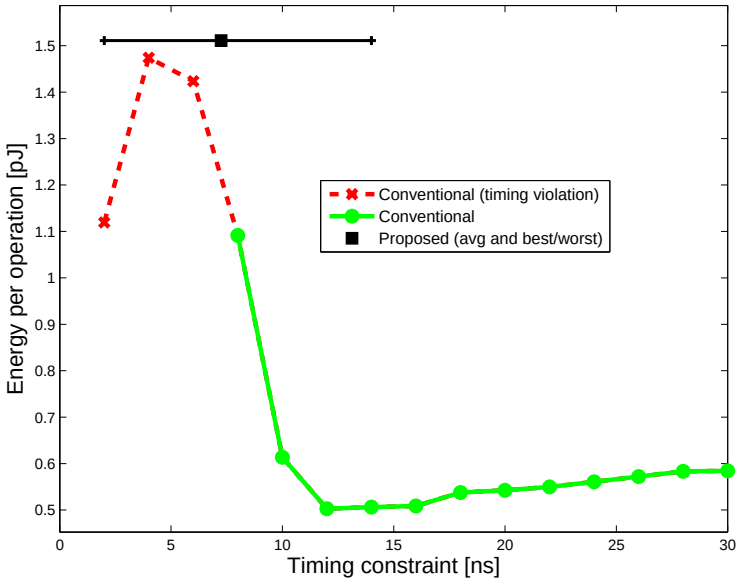


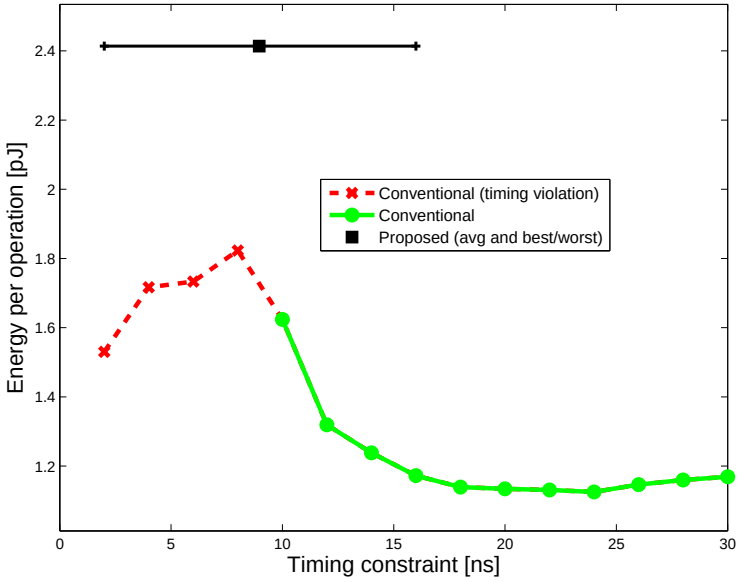
Fig. 3.9.: Overview of the timing optimization technique, inserted into the traditional flow (Fig. 3.5). The actual implementation can be streamlined and not perform as many synthesis iterations as illustrated here.

ensure deterministic timing closure. The overhead of the optimised technique remains roughly the same as the unoptimised version, with only an occasional small increase due to the increased number of TDs required to catch all signals at the early timezones, caused by the synthesizer's optimization of fast paths.

Through the application of this optimised technique, the CRC (Fig. 3.10a) and the LUT (Fig. 3.10b) now have an average performance better than the conventional alternative, albeit at the cost of an increased energy per operation. This is because, in these instances, the overhead of increasing the path speed and TD insertion outweighs the speed gains overall. This is particularly striking for the LUT, where as stated previously the nature of the circuit does not allow for significant speed variation, and the effort required to attempt doing so is significant. The performance increase is better marked for the MAD circuit (Fig. 3.10c), where the average performance can clearly be seen to have shifted towards faster paths, compared with the unoptimised version. Not only is the performance

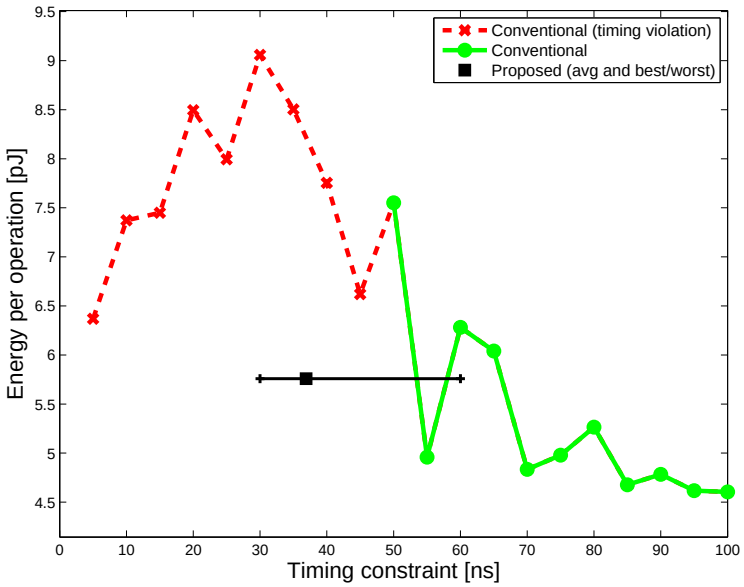


(a) CRC

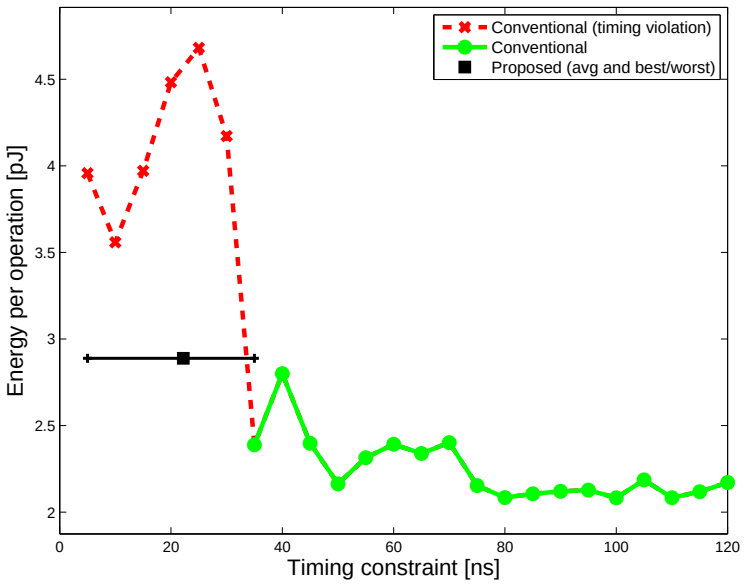


(b) LUT

Fig. 3.10.: Energy and timing achieved through the application of the technique (average case shown by black square) as compared to the baseline synthesis result of each test-case, using an optimized synthesis flow to increase the number of slow paths in view of increasing the overall performance of the proposed technique.



(c) MAD



(d) FP

Fig. 3.10.: Energy and timing achieved through the application of the technique (average case shown by black square) as compared to the baseline synthesis result of each test-case, using an optimized synthesis flow to increase the number of slow paths in view of increasing the overall performance of the proposed technique.

Table 3.4.: Comparison of the best performance achieved through application of this method, normalized to the best achievable result using conventional synthesis.

		Conventional	Proposed	
			Baseline	Optimized
CRC	Throughput	1	0.990	1.099
	E_{OP}	1	0.593	1.384
LUT	Throughput	1	1.116	1.116
	E_{OP}	1	1.176	1.486
MAD	Throughput	1	1.189	1.355
	E_{OP}	1	0.625	0.763
FP	Throughput	1	1.372	1.577
	E_{OP}	1	0.965	1.206

now significantly higher than the alternative, the associated energy per operation is also lower. The same trend can be observed for the floating-point circuit (Fig. 3.10d), where the average performance is also greater at lower energy cost.

These results show how the optimisation of the pseudo-synchronicity technique, and the associated redistribution of endpoints, allows an overall performance increase. It also shows how the results are more favourable in the larger circuits, the MAD and FP, which are sufficiently large to enable the formation of data-dependant speed variations. And more importantly, have a sufficient number of paths for the optimisation to effectively redistribute the endpoints to favour many fast paths with few much slower ones.

Overall, through optimization of the fast paths, the endpoint distribution can be changed, and better average performance can be obtained, with better power performance than can be achieved through conventional synthesis. Table 3.4 shows a comparison between the different synthesis methods employed, with a focus on high performance. Compared to the baseline result of the best performance achievable using conventional synthesis, imposed by a stringent timing constraint, the proposed technique achieves better performance in almost all cases, and with a significantly improved power consumption. The optimized technique is able to match or achieve better performance than either, and while the power consumption is higher than in the previous case, it remains below the power required for a conventional synthesis.

Table 3.5.: Synthesis results in $65nm$ GP CMOS at $V_{dd} = 0.35V$, using the optimized speedup technique.

Circuit	Configuration	Timing [ns] (min,	mean,	max)	Total energy [pJ/op]	Overhead [pJ/op]	# TDs
CRC	Conventional (best speed)		8.0		1.092	-	-
	4T, 4ns period	4	9.52	16	0.350	0.009	305
	7T, 2ns period	2	7.25	14	1.511	0.011	497
LUT	Conventional (best speed)		10.0		1.624	-	-
	3T, 5ns period	5	12.46	15	0.867	0.025	689
	8T, 2ns period	2	8.96	16	2.414	0.066	2534
MAD	Conventional (best speed)		50.0		7.551	-	-
	5T, 20ns period	40	62.20	100	3.364	0.362	1990
	5T, 10ns period	30	36.90	60	5.759	0.315	2930
FP	Conventional (best speed)		35.0		2.389	-	-
	8T, 5ns period	5	22.95	40	1.737	0.065	963
	7T, 5ns period	5	22.20	35	2.888	0.056	859

3.5 CONCLUSION

Conventionally, circuits are synthesized to attain certain fixed timing or power objectives. In such flows, all paths through the circuit are considered, and those that fail to meet the constraints are optimized. Similarly, in pipelined designs, a lot of effort is put into balancing the different stages, in order to ensure optimal performance overall. However, these approaches fail to account for the statistical frequency of the different paths.

This chapter presents a different approach; in order to achieve maximum performance on average, it is accepted that certain paths may take longer to complete than others. In doing so, fast paths are able to finish earlier than was conventionally possible, whereas slow paths are able to take additional cycles to complete. If the slow paths are rarely taken, then the average performance will increase, thereby increasing the computational potential of the system, which is one of the aims sought-after in modern systems. Or alternatively, when increased performance is not a design focus, the cells comprising the circuit no longer need to be as heavily optimized as for conventional flows under given timing constraints, thus resulting in significant power savings. This approach therefore engenders a shift in viewpoint when designing such systems; instead of ensuring that all paths are of similar lengths, this technique works best when there exists a large difference in path length, when there are many short paths and a few rare slow ones.

The technique described in this chapter implements this by using transition detectors to trace the activity within 'timezones' in the circuit, ending the operation prematurely when results are ready. No longer entirely synchronous, since there are no more fixed guarantees as to when an operation will complete, the notion of pseudo-synchronicity is introduced. The necessary detectors, and their associated circuitry, are inserted automatically into the circuit, and the overhead introduced in this way is largely offset by the power savings achieved at high performance. Additionally, an optimization technique is presented that shapes the endpoint distribution prior to applying the technique, which enables further performance increases.

The main advantage of this technique is a reduction in the overall latency of an operation. While standard pipelining techniques are able to increase the throughput, they do not affect (and sometimes indeed increase) the latency. This is useful in applications where the result of an operation is required before performing the next step. A simple and intuitive example application would be an online averaging algorithm, which requires the result from the previous data point before proceeding to the next. Whereas a traditional pipeline structure would have to stall while waiting for completion, this technique allows the loop to be performed faster and thus achieve better overall performance.

There are also many parallels between this technique and asynchronous operation. Asynchronous operation also allows the data to determine completion of a task, but uses synchronisation signals and isochronous forks in order to achieve this. Although the technique proposed in this chapter has a lower granularity

over the result, determined by the local timezone clock, it avoids the 40% power overhead [100] associated with asynchronous logic. Furthermore it allows the use of standard CAD tools, thereby reducing the design effort. Lastly, the optimised version of this technique allows the redistribution of endpoints in order to favour faster paths, something not so easily accomplished outside the proposed framework.

Results from the application of the proposed technique show that in most cases the average performance can meet or exceed that of conventional synthesis, by an average of 17% for the baseline and 29% for the optimized method. Because the technique is based on data actually encountered during operation, no pre-calculations are necessary nor are lookup tables, and the system can adapt to whatever is received. The results also show that through the endpoint shaping optimization, the best performance can be achieved, with an average of less than 21% energy increase. By imposing an average speed constraint rather than a deterministic timing closure constraint, the technique is shown to achieve far greater performance, leading to shorter system wake times and thus a 14% reduction in active power over a conventional flow.

As an example usage of this technique, it could be applied to the multiplication/division acceleration hardware in Bellevue (see Section 2.2.4). In the example application, sharpening a 128×128 image taken by the on-board CIS, the multiplication/division operations account for about 20% of all instructions processed. Applying the 14% active energy reduction to these operations would result in $5.424 \times 0.2 \times 0.14 = 0.152\mu$J of saved energy, about 3% of the total energy required to perform the task. More intense usage of the improved hardware would result in higher overall gains.

In the context of low-power applications, technology and power-aware circuit design lead to significant improvements in the overall power consumption of a WSN. However, in addition to that, this chapter presented an orthogonal circuit design optimization technique that allows a greater performance to be achieved for a smaller power usage than can typically be obtained by conventional methods. The proposed speedup technique lends itself well to ULV circuits that exhibit large variability, and is set to become even more relevant in advanced technologies.

CHAPTER 4

IANUS: A DUAL-PROCESSOR APPROACH TO THE WSN WORKLOAD BALANCING PROBLEM

For the Internet-of-Things, the underlying WSNs must meet stringent energy limitations while still achieving sufficient processing power to efficiently manage both the network and their allocated tasks. Whereas traditionally these nodes were optimized either for processing-intensive or monitoring applications, this chapter presents a low-power, dual-core processor system that can function efficiently in both modes. Software running on-board can transparently switch to a workload-optimized core during execution in order to best match the application and tasks being processed. In doing so it can either provide a lower-power performance for time-indifferent tasks, and provide flexible continuous monitoring capabilities during idle periods between processing bursts, or a DSP-capable core for data intensive tasks.

Contents

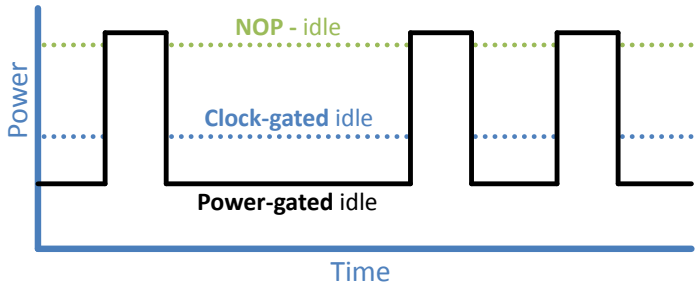
Associated author publications

JP3. F. Botman, J-D. Legat, and D. Bol, "Ianus: a workload-optimized ULV dual-processor system for monitoring and data-processing applications in the Internet-of-Things," in *IEEE Trans. Very Large Scale Integr. Syst.*, 2014.

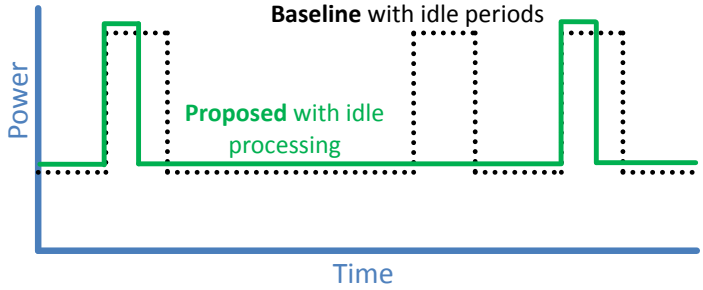
Chapters 1 and 2 described micro-controllers that employed aggressive voltage scaling in order to achieve a low-power system overall. By matching technology selection to circuit switching activity [101], combined with V_{DD} scaling down to ULV, overall system power can be reduced while maintaining performance [102, 28]. Additionally, these systems are often duty-cycled so that they remain inactive for long periods before waking up for a brief yet intense processing cycle. The energy per operation (E_{op}) as well as the active state processing speed are therefore the most important issue in such systems. Yet while this mode of operation is well suited to applications that can operate in bursts, it is not ideal for applications requiring continuous monitoring of sensor data. In such scenarios, the processing speed has little real-world importance and only the average power needs to be optimized.

Since a mixture of the two operational modes is often necessary in real-world applications, the micro-controller must be able to adapt to a varying workload. For instance, a sensor node may spend most of its time performing monitoring tasks, but will occasionally require higher performance for managing the wireless network or digital signal processing (DSP) tasks. While it is possible to use a high-performance processor combined with dedicated hardware for the monitoring [31, 34], the flexibility of an on-board processor is often required.

A typical approach to this problem is to employ dynamic voltage and frequency scaling (DVFS), which adapts the supply voltage V_{DD} and frequency dynamically at runtime to match the requirements of the application running on the system. However, such a system is not well suited to low-power WSNs for two reasons; firstly, the overhead of the DC/DC and clock generator required in order to produce a range of operating conditions is significant. And secondly, since the underlying transistors are not adapted to the switching activity, the operation at raised V_{DD} during processing bursts incurs a power overhead penalty. As a result, duty-cycled operation is usually employed instead (Fig. 4.1a), waking up regularly to ensure real-time response to events.



(a) Standard, duty-cycled approach with idle, clock gating, and power gating strategies.



(b) Proposed approach, with idle processing capabilities.

Fig. 4.1.: Timeline of power usage for a typical, duty-cycled high-performance mode and for the proposed system capable of idle processing for a limited overhead, thereby reducing the number of wakeup cycles required and their duration.

4.1 PROPOSED TECHNIQUE

This chapter presents an alternative technique that employs a dual-core processor system, where one of the cores is optimized for short-duration, intensive DSP applications and the other for continuous operation (Fig. 4.1b). By switching between these two modes, the system can use the most suitable core for the actual workload requirements of the application. The continuously-operating core ensures real-time response to stimuli and performs time-insensitive data processing, thereby reducing the number of activations of the more capable core, and the work to be performed during each wakeup.

In order to minimize any overhead induced by the continuously-operating core, this technique exploits a dual core oxide design and different clock speeds to achieve operation mode diversification, with appropriate optimizations for the expected workload.

In contrast to ARM's big.LITTLE architecture[1], and recent developments in that direction [103], the proposed technique is aimed at ultra-low power WSNs,

[1]White paper can be found at http://www.thinkbiglittle.com/.

operating at lower speeds and far lower power consumption which significantly alters the optimization strategies necessary. Rather than changing architecture between the different cores, a homogeneous assortment are used in this case, albeit with differing underlying physical implementation in order to match the anticipated switching activity (see Chapter 1.1). Based on the same design with the same instruction set, the implementation complexities are hidden from the software running on the system. Any features not supported on the low-power core will be automatically and transparently delegated to the high-performance core, making this transparent from the user's perspective.

The remainder of this chapter describes Ianus, an example implementation of the proposed technique. Through the use of two workload-dedicated cores, Ianus is able to balance the varying computational loads typical of WSN tasks, allowing low-priority, continuous tasks to operate on an appropriate core when the high-performance core is powered down. This allows monitoring and DSP tasks to operate on the same system with a reduced overall power consumption. The remainder of this chapter is organized as follows. Section 4.2 describes the architectural details of the proposed system, and Section 4.3 shows post-placement simulation results. Conclusions are laid out in Section 4.4.

4.2 IMPLEMENTATION

The objective of this work is to provide a low energy consumption in both low and high-workload scenarios, to build an overall system capable of operating in the varying workload environment of WSNs. In low-workload (monitoring) applications, the time required to complete a task is less relevant than the average power, whereas in high-workload (DSP-processing) applications a good processing speed is required which, in the context of WSNs, corresponds to a clock speed of ~50MHz (see Chapter 2) while still minimizing the energy per operation.

4.2.1 System architecture

In order to achieve these contrasting requirements, the system is built using a dual-core-oxide 65-nm CMOS technology with LP/GP transistor mix, which provides fast and high-leakage (GP) or slow and low-leakage (LP) transistor variants, and is structured as laid out in Fig. 4.2.

At the heart of the system are two processor cores (CPU-S and CPU-F) that are connected to the instruction and data buses to interface with two large 32kB memories and peripherals. The memories are foundry-provided SRAM macros which have a density advantage over ULV alternatives, but only work at nominal voltage (see Chapter 2.1). The peripherals are implemented using low-leakage transistors. Controlled by dedicated logic, the two cores alternate operation; CPU-S is implemented in a 0.35V ULV power domain using low-leakage LP transistors operating in the sub-threshold regime and, in order to minimize switching power, uses a slow 100kHz auxiliary clock. CPU-F, on the contrary, is implemented with faster GP transistors operating in the near-threshold regime in the

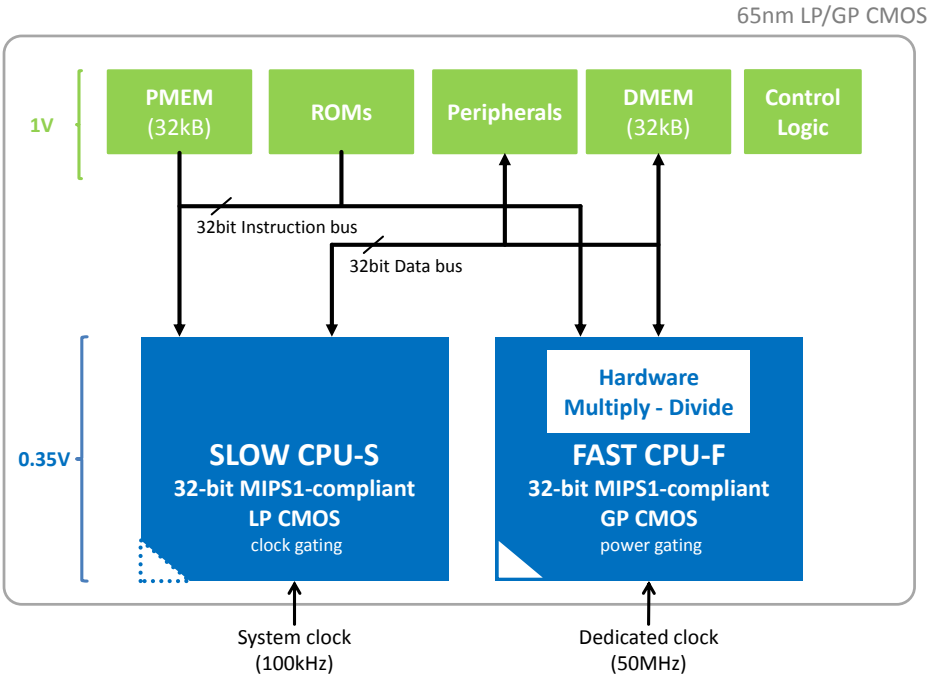


Fig. 4.2.: Overview of the Ianus processor system, implemented on 65-nm GP/LP CMOS with dual voltage and clock domains.

same 0.35V ULV domain in such a way as to ensure best performance with a 50MHz system clock. The choice of 0.35V for the ULV domain represents a favourable energy point [89], with the corresponding device partitioning [31].

In order to supply the 0.35V for the ULV domain and the associated 50MHz clock (CPU-F), an AVS with integrated DC/DC converter is used, like in Chapter 2. In order to maximize efficiency, CPU-F is power-gated when inactive and powered-up as appropriate when required (see Section 4.2.3) by enabling or disabling the power converter unit for that core. CPU-S easily meets its 100kHz clock constraint using LP transistors [28, 30] at ULV. However, this requires that the DC/DC converter remain active, though not generating a clock signal, and is able to adapt to the varying power requirement through dual-mode operation [104]. This was a design choice for the Ianus demonstration processor in order to obtain the best efficiency, but systems employing the proposed technique should consider the DC/DC converter operation overhead and size, perhaps placing CPU-S in the nominal power domain as part of the trade-off.

4.2.2 Processor cores

Each processor core used is a modified version of the Bellevue core (see Chapter 2). Bellevue is a 32-bit MIPS1-compatible 5-stage pipelined processor, designed for high efficiency at ULV. It has independent instruction and data buses, and able to operate at up to 50MHz at 0.35V.

Since only one core is active at any given time, as ensured by the control logic, both have a direct access to the buses. Each is able to receive external interrupts (IRQs), from off-chip or from an internal peripheral, which are handled by the active core. If an IRQ occurs during switch-over, the core being activated is responsible for the handling. The cores each have a custom reset vector that is configured to boot from a ROM mini-program responsible for either ensuring normal boot or state restoration after switch-over (see Section 4.2.3). By default, after reset, the slow core CPU-S is active.

Although both CPU-S and CPU-F are based on the same processor, they are still differentiated. CPU-S, for instance, lacks the hardware multiply-divide co-processor present in CPU-F. In order to maintain transparent operation of software across both, the `mult` and `div` instructions in CPU-S are implemented as a jump to a dedicated ROM mini-program that performs the requested calculation on CPU-F, and restores the result to the expected `HI` and `LO` registers. Given the fast switch-over between cores, this is actually faster than performing the calculation in software, and has the added benefit of making this entirely transparent from the software layer.

With regards to CPU-S, since the timing constraints are relatively loose, the architecture could have been modified to include fewer pipeline stages and as a result possibly further decrease its power consumption. It was decided, however, that this overhead was not significant and that the overruling priority was to make both cores compatible from a software and user perspective, something only possible by using an identical architecture. For this reason, CPU-S retains a 5-stage pipeline design.

As well as a number of minor differences, such as a custom register to expose CPU status and activation to the software, the two cores are differentiated during synthesis because of the different activity profiles configured for each one. Since a synthesis optimization attempts to minimize power, and the activity profile of each core is used as a basis for calculating that power, the highly targeted activity profile for each core induces different optimization priorities for each. This results in a dedicated physical implementation with associated underlying transistor variants which, combined with appropriate constraints, yields two cores that are highly optimized for certain types of workloads.

The result after place and route is that CPU-S, running in the ULV domain with a 100kHz clock, is better suited for monitoring applications, capable of very low energy per task, albeit operating at a much slower speed. CPU-F, in contrast, running at up to 50MHz at 0.35V, is better suited for data processing tasks.

4.2.3 Switch-over mechanism

Evidently, switching cores comes at a price, since the system is not actively working on the task for the duration of the switch-over. Therefore the break-even between when it is better to work on an optimized core in spite of the switch-over and when it is better to stay on an unoptimized core is not a trivial problem. For this reason, the switch-over is controlled by the software and not done automatically, since only the software has knowledge of the planned long-term activity profile and can therefore make better decisions than at the hardware level. However, in specific instances such as the multiplication and division instruction support in CPU-S, it is known that a switch-over to CPU-F is more power-effective and can thus be implemented in hardware. For example division on the slow core would require at least 200 cycles on the slow clock to complete, whereas by switching to the faster processor this would take at most 45 cycles to switch over, less than one to complete the operation (on the fast processor with fast clock), and optionally at most 40 to return to the slow processor immediately. This benefit is increased if the operations occur in groups, where the switching overhead can be further offset with regards to the speed increase.

The aim, therefore, is to perform the switch-over as fast as possible. Figure 4.3 shows the steps involved in the switch-over process. In order to switch cores, the application context of the active core, composed of the state of all internal registers and program counter (PC), needs to be transferred to the other core. Although this could have been implemented in hardware, it would have required too much of an overhead for something that is, in essence, rarely used.

The switch-over procedure therefore employs the processor itself to do the transfer, and the data memory as a transfer medium since it is common to both cores. All data that needs to be transferred is pushed onto the stack, and the stack pointer (SP) is saved to a dedicated control register (SPCTX) which the target core can use to recover the stack contents saved by the source core.

To initiate the transfer, the software calls a custom instruction which saves the PC onto the stack and jumps to a dedicated ROM mini-program, responsible for writing the SP value to SPCTX. This mini-program completes in 8 clock cycles. The act of writing to the SPCTX register also initiates a hardware FSM, which pauses execution on the source core, flushes the pipeline, and activates the target core. Since CPU-F is power gated when inactive, it must first be woken up when switching-over from CPU-S to CPU-F, requiring a further 3 cycles at 100kHz to complete. CPU-S is not power gated in this design, since it is designed to operate a majority of the time, so there is no core waking step when switching over from CPU-F to CPU-S. In both cases, the powering down is done in parallel with the execution on the other core so does not contribute to the switch-over delay.

Upon booting, the target core runs the ROM boot code and determines whether a normal boot or a context restore is necessary, based on the value of the SPCTX register into which the previous SP was saved. If a normal boot is required, the program stored in the instruction memory is run normally. Otherwise, the SP is loaded back into the core, the PC is restored from the stack,

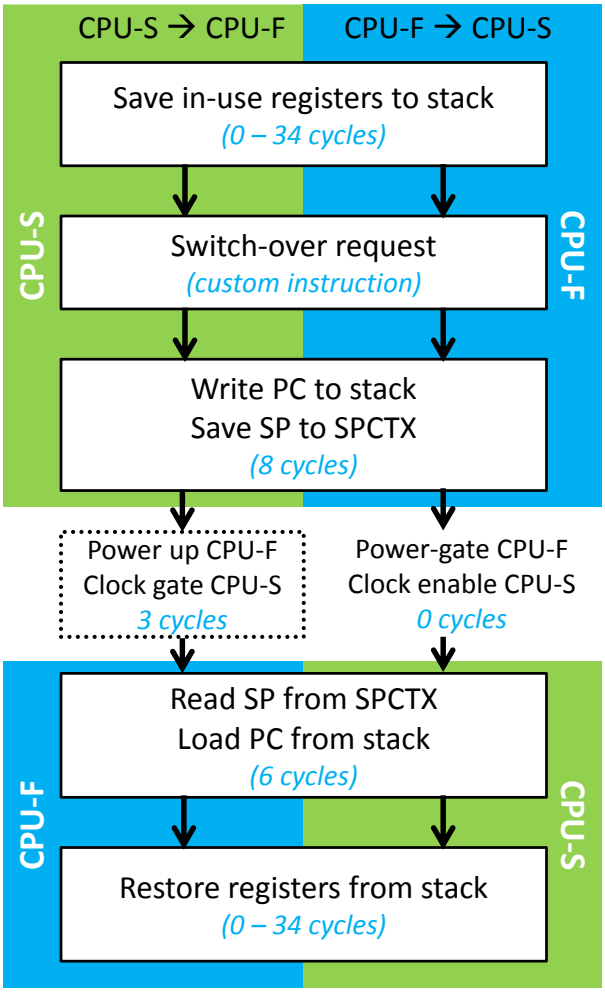


Fig. 4.3.: Switch-over procedure for Ianus, indicating number of cycles (relative to the source core) necessary for the transfer.

and execution resumes from where the other core left off. This process takes 6 clock cycles.

On top of the essential context consisting of the SP and PC, other internal registers may also need to be transferred, but the number of registers in use varies with the application and execution; transferring all 34 internal registers every time even if unused is wasteful. In fact, only the compiler is able to determine which registers are used at any point in time during the execution, and so the number of registers to transfer is determined by the compiler itself. To do so, the core-switch custom instruction described above is marked as potentially able

to alter all registers, and the compiler accordingly automatically adds code to save the relevant registers onto the stack before the instruction, and to restore them afterwards. Since through exchange of the PC the execution resumes just after the custom instruction, the complete application context can be transferred through the stack.

In this way, the complete application context can be transferred from one core to another in less than 100 cycles, often much less since only a subset of registers are active and must be exchanged depending on the application requirements itself. This is significantly less than the $20,000$[2] required in competing designs.

4.3 RESULTS

In order to evaluate the proposed system, its performance across a variety of benchmarks was compared to both a reference Bellevue core [89] and a commercial ARM CORTEXM0DS, a leading low-power processor for this class of applications. These were compared at both nominal voltage and ULV. The synthesis flow, testbench, and simulation framework is identical in each case, in order to fairly compare the different systems.

4.3.1 Benchmark programs

A series of four programs was developed in order to test the performance of the different designs when exposed to varying workloads. Two of these correspond to low-workload, control or monitoring scenarios, whereas the remaining two reflect high-workload tasks.

Accumulate This program performs data accumulation, sampling random 32bit data and storing the result in memory. The random data itself is obtained through a read request to an external register, accessed through the data bus, and fed random data by the testbench (the sampled data is therefore the same for all processors).

Average This program performs basic sensor noise removal by sampling random 32bit data, calculating the online average of that data, and checking for its convergence to within a pre-defined threshold.

AES-128 This program performs an AES encryption of a block of random data (plain text), representing a high-workload task.

Sharpening This program performs an image sharpening task by applying a 3×3 enhancement filter on a $64 \times 64 \times 8$-bit grayscale image stored in RAM, a high-workload task.

These benchmark programs are representative of tasks required in typical WSNs.

[2]Data from the ARM Whitepaper "Big.LITTLE Processing with ARM Cortex^{TM}-A15 & Cortex-A7", Sept. 2011.

4.3.2 Test conditions

In order to provide a fair comparison, the external interface of all designs was made identical in order to employ the same testbench. In the case of the ARM processor, which only has a single instruction/data bus, the two memories were mapped into a single, larger 64KB memory.

Similarly, it was decided that no instruction or data cache would be used, since the choice of cache architecture would create an inherent bias in favour of one of these processors, thereby affecting the results. This does, however, cause an increased power consumption of the memories and obviously, in an actual design employing the proposed technique, an appropriate cache should be chosen.

Since the compiler itself has a large impact on the output binary, the same `GNU GCC` compiler, albeit with a different backend port specific to each architecture, was used to generate all binaries. Moderate optimization was used, in order to prevent absurdly inefficient code or reflect compiler target support variations. Additionally, a custom implementation of standard-C routines was used in order to ensure no unfair optimization possibilities. Visual inspection of the resulting binaries revealed a code density advantage to the ARM processor, in no small part due to the THUMB mode it implements (see Table 4.1).

Table 4.1.: Program size comparison (instruction memory only) for each processor type, showing the impact of the ARM's THUMB mode, and the overhead of the switch-over code in Ianus.

Program	**ARM**	**Bellevue**	**Ianus**
Accumulate	320B	600B	728B
Average	360B	376B	536B
AES	6.8kB	13.0kB	13.0kB
Sharpening	3.4kB	4.6kB	4.7kB

For each core, the pre-synthesis activity of the system running each program one after another was obtained in simulation. The resulting activity file (consisting of the average of the four programs) was then used during synthesis for targeted power optimizations. After place and route onto a generic floorplan, the power performance was evaluated by using a dedicated activity profile for each program individually.

4.3.3 Power estimation procedure

In order to produce the results, the power for each processor must be obtained for all operating conditions that are to be tested, over a wide range of frequencies and voltage conditions.

Traditionally, power of the system would be obtained in the following manner, in order to gather the necessary test results. First, the pre-synthesis activity of the system running each program one is obtained through RTL simulation. The activity of each program running one after another is also required, so that the average activity of the processor under this combined workload is also known. It is the latter which is used as an input for synthesis, in order not to bias the resulting circuit towards a single benchmark. After place and routing onto a generic floorplan, the power performance is re-evaluated by using a dedicated post-place activity profile for each individual program, obtained through post-placement simulation.

However, the last part of this power estimation, the post-placement simulation, is very resource-intensive and requires a long time to simulate entire programs. Given the large number of simulations required to obtain the necessary data for a comprehensive comparison, this would have taken prohibitively long on the infrastructure available. Similarly, the pre-synthesis activity cannot, in general, be used as the gates change location, connectivity and name during the synthesis process, resulting in incomplete activity annotations and thus inaccurate power estimates.

For this reason, an alternate strategy was devised. Given the pre-synthesis activity and the pre and post-placement netlists, it is simply necessary to define a mapping between the original gates, which are referenced in the pre-synthesis activity file, and the resulting gates after placement. Once this mapping is identified, the circuit can be annotated and the power estimated, with gates inserted during the flow automatically using propagated activities.

As it happens, the Cadence Verification toolsuite (Conformal, `lec`) already does this as part of the verification process, used to ensure that the resulting netlist is functionally identical to the input netlist during the sign-off phase of ASIC design. Through dedicated scripts, it is possible to extract this mapping, and, after a small amount of post-processing, use it for power estimation in Cadence Encounter.

The resulting power estimation is just as accurate as one based on a post-placement simulation, provided the design meets all timing constraints, only takes significantly less time to obtain.

4.3.4 Core performance comparison

Core validation tests show that the Ianus cores are on par with the reference cores (Fig. 4.4) in terms of energy required to complete a given task. The design point of a 50MHz clock was used, with a 100kHz clock in the case of the Ianus slow core. This design point was chosen as it represents a typical processing capability for this class of processors (Chapter 2).

It can be noticed that although the slow core (CPU-S) takes far longer to complete, it consumes approximately the same energy as its faster counterpart, thereby providing a viable alternative for flexible computing. The contributions of the memories (not shown in the figure) is significant due to their presence

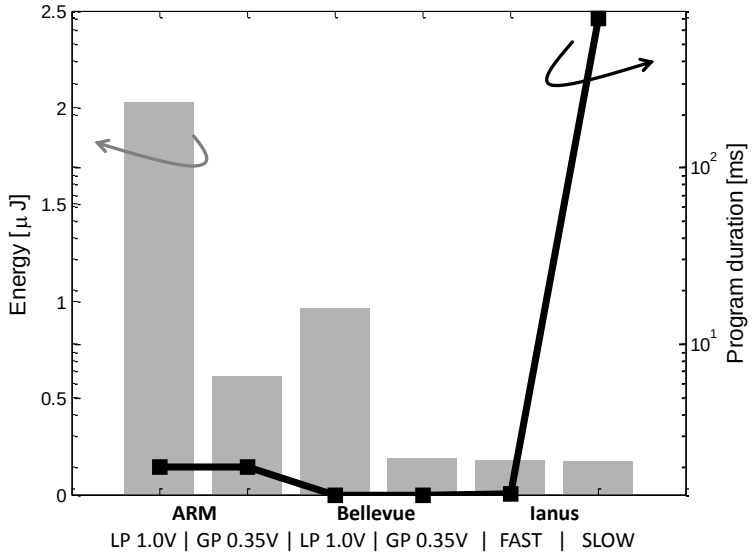


Fig. 4.4.: Energy performance of each core and time required to complete a data accumulation and manipulation task. System clock is 50MHz, while the slow core clock is 100kHz.

in the nominal-voltage domain. The impact of this could be mitigated through the use of a ULV cache [31], and should be done in actual designs. Doing so for this demonstration core, however, would unfairly bias towards a particular core through choice of architecture, and so was not used in this case. The power difference between nominal and ULV cores is significant, and underlines the importance of ULV designs in such low-power systems. The power for the Ianus fast core is equivalent to that of its Bellevue ULV cousin, for in spite of the overhead of the ROMs and control systems required for the proposed processor, the core itself is somewhat streamlined by removing features such as SIMD operation which are not needed in this specific case.

The time taken to complete the program is also shown (Fig. 4.4). This time remains constant for each core irrespective of supply voltage, since it is governed by the architecture rather than the operating conditions, for a given input clock frequency. Although the Ianus slow core (CPU-S) takes far longer to complete than the other cores, it also consumes far less because of its streamlined implementation and exclusion of complex accelerator, resulting in an overall equivalent energy consumption. Table 4.2 shows this effect in the case of a low-workload program, maintaining low-power operation, but significantly increasing the time required to complete the operation (by about 500× due to the difference in clock speeds).

In this way, Ianus can function efficiently in high-speed or low-speed modes, providing a large degree of flexibility for the target application.

Table 4.2.: Ianus processor performance comparison for a low-workload (data accumulation and very simple processing) task, achieving low power operation through use of the low-performance core.

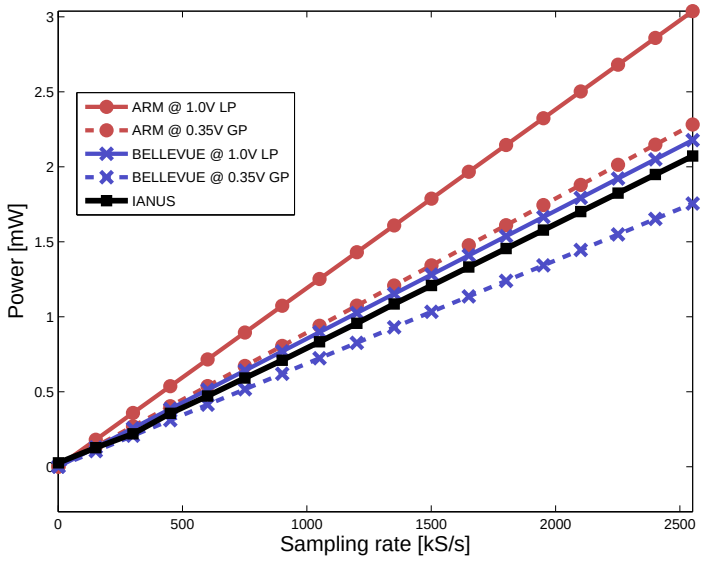
Processor	**Power [mW]**	**Time [ms]**	**Energy [μJ]**
ARM 1.00V LP	1.014	2.00	2.03
ARM 0.35V GP	0.305	2.00	0.61
BELLEVUE 1.00V LP	0.686	1.40	0.96
BELLEVUE 0.35V GP	0.131	1.40	0.18
IANUS CPU-F	0.129	1.40	0.18
IANUS CPU-S	0.00024	700.63	0.17

4.3.5 Workload-dependent performance achieved

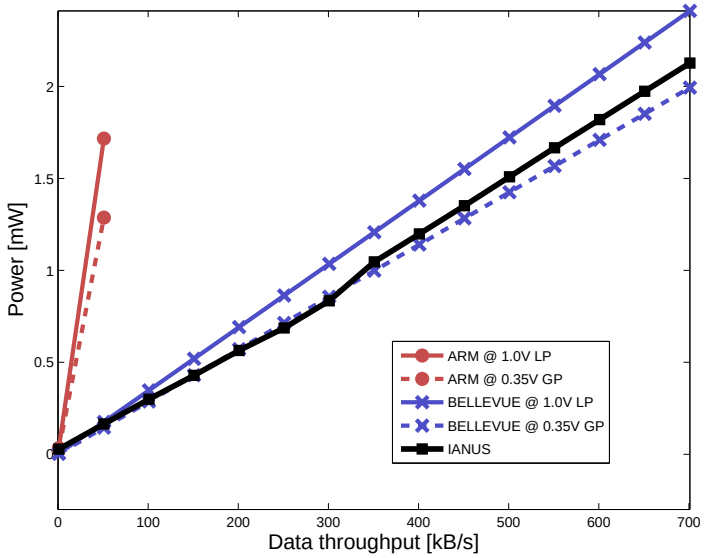
It is clear that a processor designed for 100% high-workload scenarios will perform better than the proposed system in an equivalent scenario, since Ianus suffers from the overhead of having two cores and control logic. However, real-world applications often face varying-workload tasks instead. Processors must not remain active when not working on a task and are power-gated, or at the very least clock-gated, in order to reduce idle power (Fig. 4.1a). Ianus, however, maintains an active slow core in order to handle monitoring and non-DSP tasks thereby both efficiently performing the required tasks and obtaining a power advantage through not having to power up the main processor to perform these limited, time-insensitive actions.

Figure 4.5 shows the performance of the proposed processor compared to competing designs over a variety of workload scenarios, taking into account the entire design composed of the cores, peripherals and memories. In each case, all processing is initially performed on the fast core until the throughput requirements are achieved, then the slow core is used to allow for low-power idle-computing. The Ianus performance is compared to the reference processors that operate under identical conditions. These are power-gated in an ideal manner once processing is complete, in that no shut-down or start-up, nor wake-up configuration via external peripheral is implemented. Whereas the Ianus power results are obtained from direct simulation of varying workloads, taking into account processor switch-over overhead, the reference power of the other processors is obtained through duty-cycling their respective active power based on the time needed to achieve the required throughput.

It can be observed that, as expected, the Bellevue processor at ULV consumes less power than at nominal power, and the same thing can be observed for the ARM. In all cases, Bellevue consumes less power than its ARM counterpart, the former being optimized for ULV operation. In particular, in the case of the Average benchmark (Fig. 4.5b), the use of a software division requires the

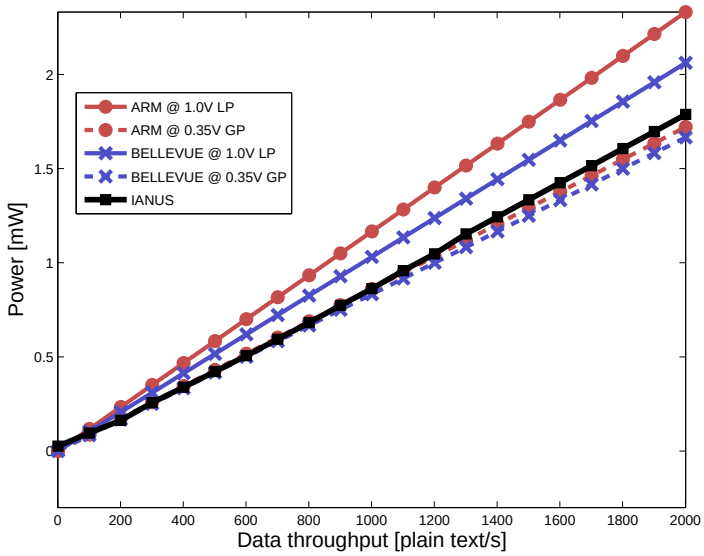


(a) Accumulate

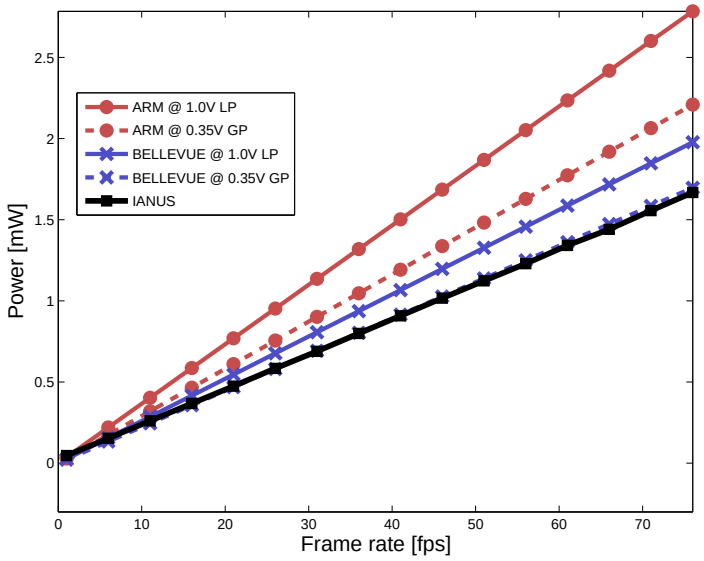


(b) Average

Fig. 4.5.: Average power required for each program on all designs, as a function of workload. Simulated post-place power performance of Ianus is compared to duty-cycled average power for the other processors, considering an ideal power gating technique with zero power in idle mode (which in reality would not be as favourable). As well as achieving the desired throughput, Ianus operates in low-power continuous monitoring mode, which is represented here via the overhead when compared to Bellevue ULV.

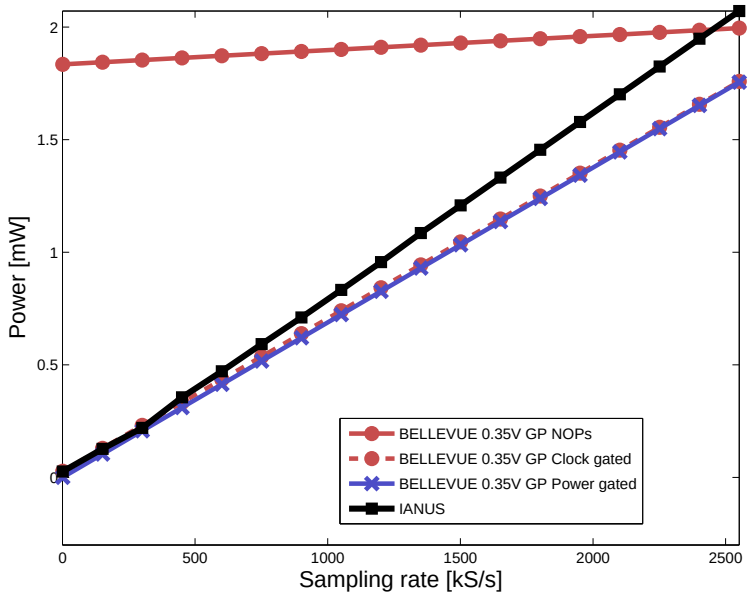


(c) AES

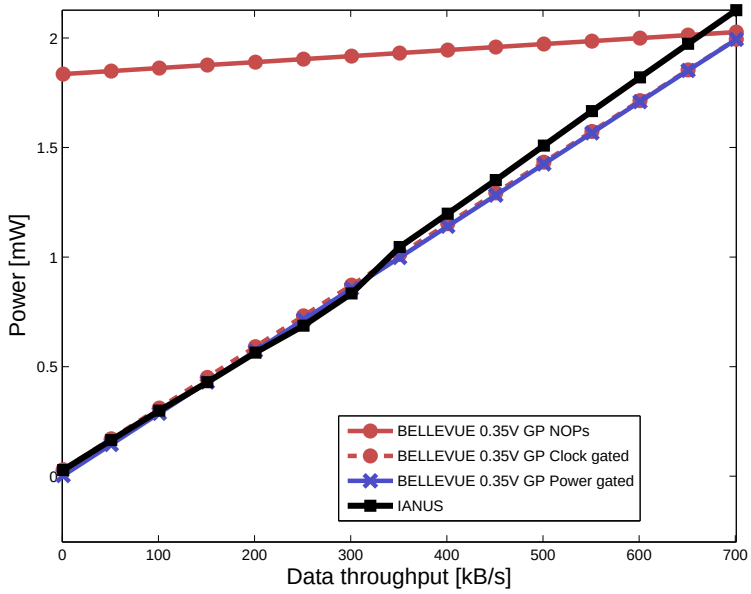


(d) Sharpening

Fig. 4.5.: Average power required for each program on all designs, as a function of workload. Simulated post-place power performance of Ianus is compared to duty-cycled average power for the other processors, considering an ideal power gating technique with zero power in idle mode (which in reality would not be as favourable). As well as achieving the desired throughput, Ianus operates in low-power continuous monitoring mode, which is represented here via the overhead when compared to Bellevue ULV.

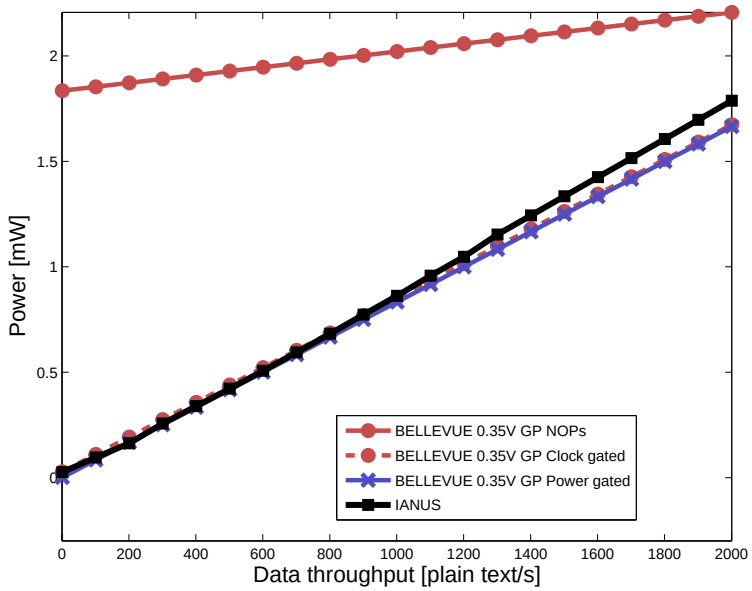


(a) Accumulate

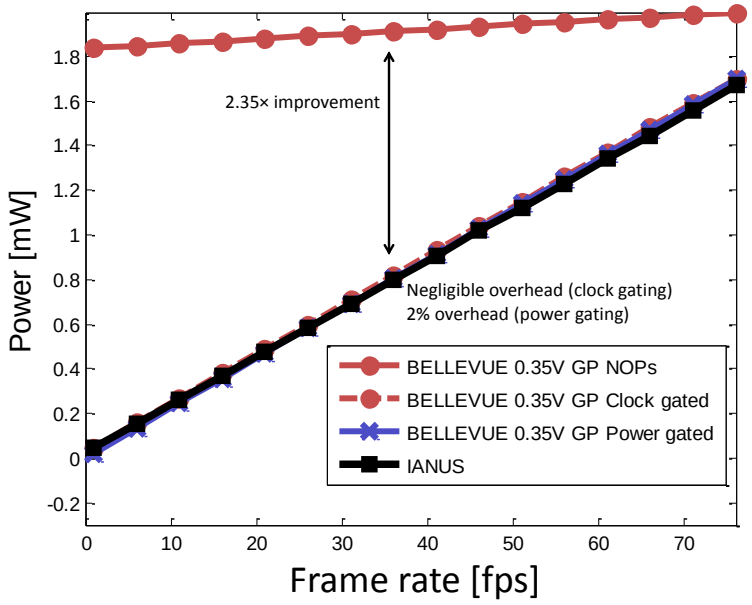


(b) Average

Fig. 4.6.: Average power of Ianus compared to the best alternative (Bellevue ULV) under ideal power-gating, clock-gating and idle-wait (NOP-loop) strategies. Data shown from 1 operation per second to the approximate maximum performance possible.

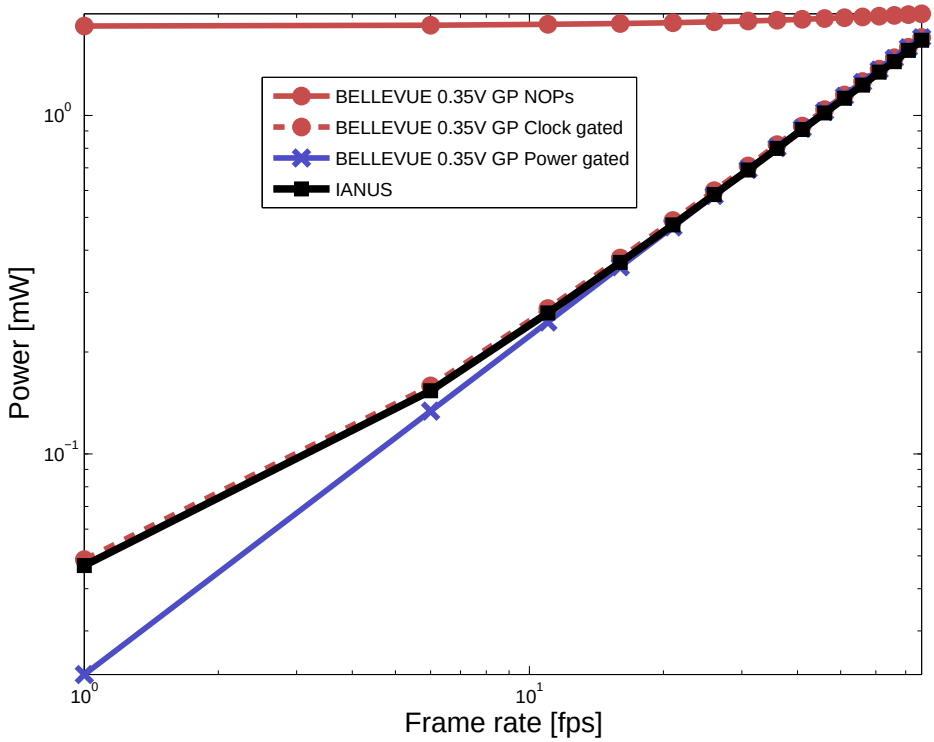


(c) AES



(d) Sharpening

Fig. 4.6.: Average power of Ianus compared to the best alternative (Bellevue ULV) under ideal power-gating, clock-gating and idle-wait (NOP-loop) strategies. Data shown from 1 operation per second to the approximate maximum performance possible.



(e) Sharpening (log-log view)

Fig. 4.6.: Average power of Ianus compared to the best alternative (Bellevue ULV) under ideal power-gating, clock-gating and idle-wait (NOP-loop) strategies. Data shown from 1 operation per second to the approximate maximum performance possible.

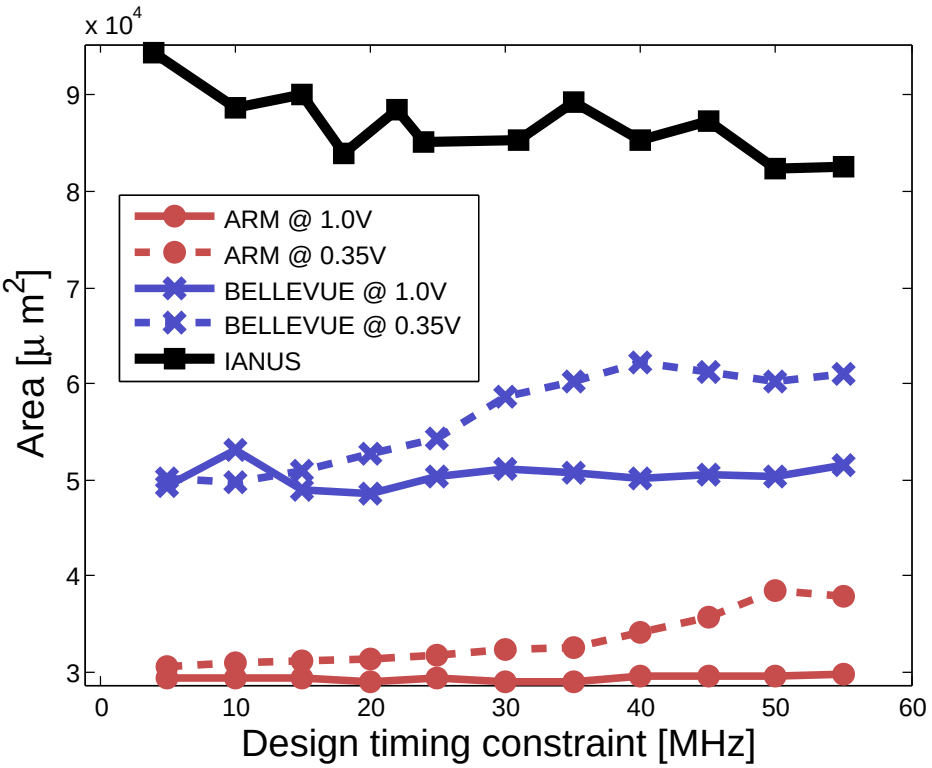


Fig. 4.7.: Area required for each design, including all cores, peripherals and embedded memories.

ARM processor to take longer per operation (resulting in a higher power usage), and makes it unable to meet the throughput achieved by Bellevue-based cores that feature hardware division. Similarly, for that benchmark, Ianus is strongly affected by having to switch cores.

Compared only to the best-performing alternative, in this case Bellevue implemented at ULV, Fig. 4.6 shows the power advantage of Ianus at ~50% workload. With a mostly negligible overhead (Fig. 4.6d) over a power-gated or clock-gated core, and 2.35× improvement over an idle core performing a `NOP`-loop, Ianus is able to efficiently combine high-performance operation with continuous low-performance processing during idle periods. Since in Ianus the fast core is power-gated when unused, the resulting static power (see Fig. 4.6e) is much lower than an active Bellevue ULV core and below that of its clock-gated alternative, but above that of the power-gated version due to the operation of the slow core, control units, peripherals, and memories.

Figure (Fig. 4.7) shows the area necessary to implement this, as a function of design constraint. The presence of two Bellevue cores within Ianus incurs an

area increase of 50% compared to Bellevue given the identical SRAM capacity. ARM uses somewhat less area than Bellevue, in large part because of the lack of dedicated data bus and fewer pipeline stages. For each, the area used in nominal and ULV domains is roughly the same at low-frequency, but whereas at nominal voltage the timing margins remain loose, at ULV buffers must be inserted at high frequency in order to maintain the desired performance, thereby accounting for the increased area.

In order to apply the proposed technique in actual designs, it is necessary to apply a few modifications to the architecture that were not included in Ianus in order to reduce the complexity of the overall circuit thereby allowing a better comparison between the alternatives. Firstly, an instruction and data cache should be added in the ULV domain, since this will strongly reduce the switching overhead to and from the memories in the nominal voltage block. Secondly, the efficiency of the on-board DC/DC should be evaluated and used to decide whether CPU-S should remain in the ULV power domain, or, if the inefficiencies are excessive, can be continuously powered in the nominal domain. Lastly, in the case that continuous operation is not required, power-down options or idle-power reduction mechanisms such as optimized `NOP` operations [38, 105] should be considered for CPU-S.

In the context of the Ianus processor, used mainly in order to evaluate the proposed technique, the choice was made to favour a fair comparison between cores. By applying the technique, Ianus is able to either perform time-insensitive tasks on a slow processor in order to achieve increased power efficiency (Table 4.2), or provide both the necessary throughput and idle processing capabilities for monitoring tasks with a negligible average overhead compared to conventional solutions. By having an always-active core, applications will not need to power-up CPU-F as often nor for as long, making this technique an interesting alternative. A WSN built using this processor will be able to perform high-processing bursts, but also maintain continuous operation in order to flexibly handle monitoring tasks that are time-insensitive, such as sensor acquisition and threshold monitoring.

4.4 CONCLUSION

In this chapter, a technique is proposed to address the shortcomings of burst-mode, high performance processors when applied to monitoring tasks. By combining two cores optimized for different workloads, the application itself can intelligently decide which is better suited to the task, and enable idle processing capabilities in order to reduce the frequency and duration of wakeups of the more powerful core. This technique is demonstrated on an example design, the Ianus system.

From a user perspective, operation on either core is transparent since the cores are architecturally identical and any unsupported features are automatically delegated to the high-performance core. Switching cores is also simple, and

can be initiated via a single instruction. Due to a fast core switch-over mechanism proposed, which transfers only the necessary application context and can complete in fewer than 100 cycles, the overhead of core switching is limited, allowing significant flexibility at the application level.

Ianus is implemented using a dual-voltage, dual-core-oxide setup that optimally exploits features of the technology, and shows how the combination of such flavours allows a greater design-time flexibility for low-power optimization of each core, which is most relevant in ULV systems. The proposed system performs low-workload tasks as efficiently as using the more capable core, albeit much slower. It allows idle-processing with negligible power overhead in spite of the extra hardware, and allows fewer, shorter high-power processing bursts. The performance of Ianus and its ability to support different workload scenarios, including idle-processing, make this technique an ideal for WSNs in the IoT, which are particularly sensitive to workload changes.

CONCLUSIONS AND PERSPECTIVES

The Internet-of-Things is an extraordinary vision for improving lives and well-being through sophisticated technology. In order to achieve this, the individual composing nodes, sensors, actuators and controllers must fade seamlessly into the framework of our environments. This requires each node to function as efficiently as possible, in order to be able to replace batteries and their associated maintenance with harvested energy operation. It also requires that each node be able to communicate wirelessly with a central control node, which poses an interesting power optimization problem.

While there are already existing ultra-low power systems, this thesis sought out further low-power advances through system-level optimizations and processor-level implementation techniques. For instance, in order to reduce the usage and power impact of the wireless link, a DSP-capable micro-controller was proposed to pre-process the sampled data and only send relevant information to the controller. However, this requires that the processor itself is of sufficient efficiency in order to make this trade-off worthwhile.

Specifically, this thesis sought to answer the following queries;

- Can a processor provide sufficient computing performance to carry out high-level tasks and signal processing, specifically the on-board analysis of sensed image data, whilst operating within the constraints of an energy-harvesting system?

Harvested energy operation results in there being very little available energy with which to operate, and it was by no means certain, given the state-of-the-art, that a complex system capable of performing the required tasks could be possible.

Building upon advanced low-power design techniques and the state-of-the-art (as laid out in chapter 1), this thesis proposes a micro-controller design that is efficient enough to function under the constraints of energy harvesting. Specifically, Bellevue (Chapter 2, Fig. 5.1) focuses on a high-performance and more capable, 32-bit core in order to target data-processing applications of sampled data. As well as using state-of-the-art implementation techniques, the custom-designed Bellevue core is capable of 32-bit, 50MHz operation at 0.37V. Archi-

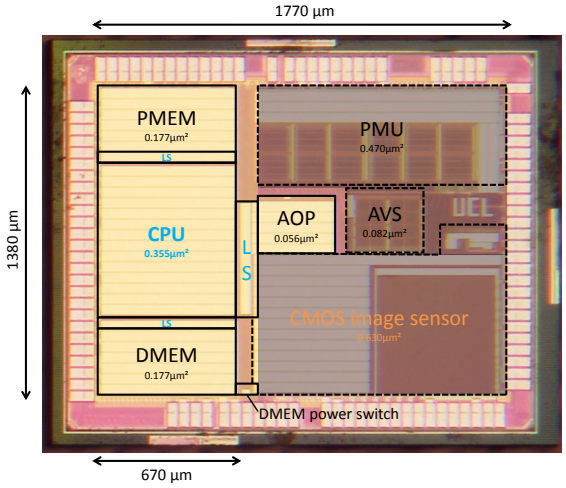


Fig. 5.1.: The SunPixer SoC chip containing Bellevue that was taped-out as part of this thesis.

tectural innovations brought to it include a variable-width SIMD pipeline, with SIMD-capable multiplication and division accelerators. This allows an increased processing capability, allowing 7.7μW/MHz operation overall, and a low-power 1.55μW sleep mode. It is also shown how the use of this processor to process images, obtained by the CIS, results in a significant power advantage over sending the data via a wireless link.

Through the development of the Bellevue chip, and the associated design techniques that enable low-power and high-performance, this thesis has shown that it is indeed possible to develop capable systems that can processed sensed data on-board better than by offloading to a remote node, and within the power constraints of an energy harvesting system.

Claim 1 A processor can be designed to provide sufficient computing performance to carry out high-level image-processing tasks of on-board sensed image data whilst operating within the constraints of an energy-harvesting system.

Claim 2 An on-board processor with data-processing capabilities can significantly improve system power consumption through a reduction in activity of the wireless link.

This thesis also sought to answer the following question;

- Beyond traditional low-power design techniques, are there alternative circuit design strategies that could yield increased processing speed in low-power systems?

Bellevue achieved high-performance and power efficiency in part through its advanced physical implementation, a trend which is set to continue as technology scaling continues. However, in addition to that, Chapter 3 considers the problem of a combinatorial block that does not fit into the overall system clock requirements. Through pseudo-synchronicity, by inserting TDs into the very fabric of the circuit, it is possible to accelerate the average performance of the block, in many cases up to 29% beyond that achievable by conventional synthesis methods that must consider the worst-case timing. By accelerating a circuit in this way, lower average power can be achieved through increased duty-cycling.

The pseudo-synchronicity technique proposed in this thesis shows that it is possible to move beyond the conventional limitations of traditional synthesis constraints, by considering average case events rather than designing for the worst-case.

Claim 3 Pseudo-synchronicity allows greater power efficiency by accelerating circuits up to 17% beyond that conventionally achievable if only the worst-case timing is considered.

Claim 4 By segregating fast and slow paths prior to the application of pseudo-synchronicity, overall average performance can be further increased to 29%, thereby reducing operation energy by 14%.

Lastly, this thesis considers the problem of varying workloads in WSNs;

- How can processors be made to operate efficiently in spite of the varying workloads typical of such devices?

Wireless sensor nodes are characterised by their need for high efficiency, and for diverse processing requirements. On the one hand, they may be required to perform simple data accumulation, and on the other, especially with regards to Claim 2, they may be required to perform data processing and complex management of the wireless link. DVFS lends itself badly to low-power systems due to the hardware overhead required, and the mismatch between the actual processing conditions and the design point. And simple duty-cycled operation results in the need to wake the processor sometimes redundantly in order to maintain real-time performance.

Chapter 4 proposes a solution to this, by using two cores each optimised for a given operation mode. Applications can switch to the appropriate core given the computational needs, and by using idle-processing the system can both react in real-time to sensor events and reduce the frequency and duration of wakeups required.

In Ianus, the micro-controller implementation of this technique, the use of the Bellevue cores at ULV, and optimized physical implementation at the transistor level, results in negligible overhead. Yet this offers the possibility of performing idle processing, thereby enabling the system to better match the workload, realtime requirements of the application, and number of wakeup cycles.

Claim 5 Through a dual-core, physically optimized approach enabling idle-processing, it is possible to efficiently match the workload requirements of such devices, and thereby save power.

In summary, this thesis contributes designs and techniques to improve the viability of nodes targeting the IoT, allowing lower power for greater performance than the current state-of-the-art. Specifically, these are orthogonal optimisations that can either be used together, or independently, giving IoT designers more options and greater flexibility when making design decisions. For instance, Bellevue is shown to be a capable and flexible processor that both achieves low-power and implements features such as SIMD that are well suited to IoT applications. It could therefore be employed independently as a power-saving processing platform for a node, achieving longer lifetimes as a result, or also in combination with the multi-processor approach demonstrated in Chapter 4. This would give the resulting system both the power advantages associated with the processor itself, as well as the idle-processing capabilities and low-latency event reaction associated with Ianus. In the same way, a designer could choose to employ the pseudo-synchronicity technique on a hardware accelerator to be used within the resulting system, in a bid to either increase performance, reduce power, or avoid multi-voltage design issues. When applied appropriately, the resulting system could feature increased performance and lower overall energy than alternative systems.

This independence is a direct consequence of the different levels of abstraction that each technique targets; with the multi-processor design, a system-level optimisation is employed, managing power from a fairly high, system-level viewpoint. The Bellevue processor core itself achieves its performance from optimisations at the architectural level, with the addition of power-saving features, as well as from a streamlined implementation. Lastly, the pseudo-synchronicity technique operates on the low-level circuit itself, enabled through extensions of the CAD tools and insertion of custom-designed cells. Together with the associated implementation work on both the physical layout and tape-out of the SoC, and on both the compiler and software, this thesis demonstrates a consideration of all the abstraction levels associated with processor design. And it is through the knowledge gained by such an analysis that these optimization techniques could be found on different levels, thus enabling them to be potentially used together as orthogonal optimization techniques.

But overall, this thesis is all about giving designers tools with which to make better choices, and alternatives to the previously inevitable solutions. Like cobblestones along the road of progress, it is hoped that these contributions will, in some way, pave the way to enabling the exciting applications of tomorrow.

A designer's manual

This thesis proposes a number of contributions to circuit design in the context of WSNs, in order to achieve increased power efficiency for that class of designs.

Figure 5.2 summaries a selection of these contributions by category of problem that the designer is attempting to tackle.

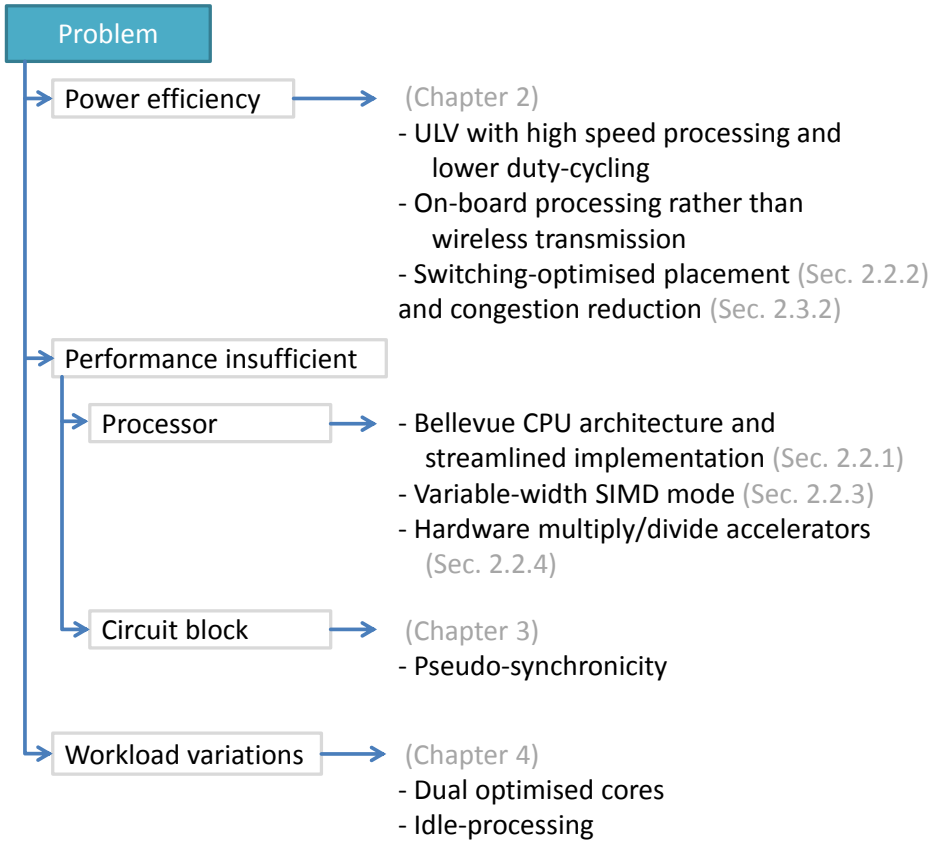


Fig. 5.2.: Summary of selected contributions of this thesis for solving the specified design problems.

Together, and combined with other low-power techniques from the state-of-the-art, these methods provide the tools towards the design of a pseudo-synchronous multi-processor system-on-chip for processing-intensive low-power applications.

Perspectives and future work

The past few years have seen exciting developments and changes in the field of electronics. On the one hand there has been a definite migration towards 'cloud' systems, where data is kept on remote but highly connected servers providing access to it from anywhere with an internet connection. On the other, there

has been significant developments in mobile technology, that now give users incredible computing resources that can fit inside their pockets.

This has the consequence that, as more data is centralised, it becomes easier to process and correlate with other data sources, increasing its value and relevance. And with mobile devices, not only has this data become available anywhere any time, these also add contextual information such as location. The well-known tagline of a popular smartphone maker, "there's an app for that", exemplifies from a software perspective what is coming on the hardware side. The IoT vision is already starting to become a reality, as 'connected devices' are now on the commercial market to monitor health aspects such as heart-rate, blood oxygen levels, and physical activity, paving the way for less localised IoT networks.

There remain, of course, challenges to be overcome, not least that of power consumption that forms the primary focus of this thesis. The available microcontrollers presented in the first chapters are set to improve as the underlying technology continues to scale, and alternative physical implementations such as fully-depleted SOI (FD-SOI) [106, 107] and FinFETs [108] become more widely available.

And most promising are the upcoming integration mechanisms such as through-silicon vias (TSV) [109] that will allow a broader spectrum of available technologies that can be accessed more power-efficiently and faster, allowing for example the combining of differing CMOS processes in the same die stack. In time, perhaps this will allow true 3D designs that will significantly improve device density and usher in a new wave of miniature applications.

Energy harvesting will become more prevalent, with more efficient control designs being developed. Micro solar cell efficiency will increase [110], and with the continued research into MEMS, energy harvesting sources are set to multiply [111]. Also, actuation will become possible on a smaller scale and cheaper to produce, possibly paving the way towards wall-sized displays, self-healing structures, or nano-actuation systems within the human body itself.

But one trend is certain, the number of devices is set to explode, and with it the number of wireless communications in any given area [112]. There will therefore be an evolution in the way wireless networks are set up, and point-to-point transmissions will eventually favour local cell-based networks with hops in order to minimize transmission distance and thus power. As a result, on a protocol level, communication channels will be required to survive interruptions, thereby pushing towards the adoption of strategies such as multi-path TCP in order to seamlessly handle multiple channels, making wireless links both more accessible and more reliable.

From the point of view of this thesis, and from the perspective gained through its elaboration, there remain a number of challenges to be tackled by future research;

- First and foremost, the benefits and challenges associated with more advanced technology nodes should be investigated. No doubt these will bring

power and speed benefits, but probably also reliability challenges that will call for architectural resilience and error tolerance or correction techniques.

- The pseudo-synchronicity technique described in Chapter 3 was only implemented on a combinatorial circuit block in the context of this thesis. However, it could also be transposed to either replace the pipeline structure of a processor, in the manner of a wave pipeline [113], or work within its structure to enable faster performance of the overall core by using average-case timing delays rather than the worst case, as per the operations actually requested of the processor.

- Similarly, the pseudo-synchronicity approach could be used to remove the requirement for a stabilised power source, and allow operation directly from a miniature solar panel, for example. Using the proposed technique, operations would be able to take as long as needed to complete, given the voltage and power levels available above the functional limits. This would eliminate the need for an AVS system, and hopefully increase the efficiency of the system.

- Lastly, it is clear that a lot of energy harvesting sources have their origins in RF-capture, not least active RFID systems. This captured energy translates to an alternating voltage at the system level, that typically has to be rectified before use, with the inevitable resulting losses. It would therefore be interesting to consider the possibility of supplying this alternating voltage directly to a processor, which would use it as its clock signal. Half of the pipeline stages would be reverse-polarised, enabling data to move from one stage to the next as the polarity of the supply changed. This would require the custom-design of 'register' cells that would be able to survive the zero-voltage transition and transfer their contents to a negatively-polarised structure. However, in theory this would allow the removal of both a power regulator and clock source from the target circuit, resulting in higher overall efficiency.

This thesis has contributed a number of techniques and designs to the field of processors for wireless sensor nodes, improving their power efficiency and increasing their processing capabilities. Through higher computational abilities for data-intensive applications, such devices will be able to function longer on more limited power sources, such as that stemming from harvested energy. The vision of having a multitude of sensors interacting together to form the Internet-of-Things is compelling, and though its introduction into our lives will be measured and subtle, it will serve the noblest purpose of improving the quality of life for all people; empowering them to live healthy and efficient lives, increasing their awareness and impact, and altogether making the world a better place.

REFERENCES

1. G. Moore, "Cramming More Components Onto Integrated Circuits," *Electron. Mag.*, vol. 86, no. 8, pp. 82–85, Jan. 1965.
2. S. Chalasani and J. M. Conrad, "A survey of energy harvesting sources for embedded systems," in *IEEE SoutheastCon 2008*. IEEE, Apr. 2008, pp. 442–447.
3. M. Balouchestani, "Low-power wireless sensor network with compressed sensing theory," in *2011 4th Annu. Caneus Fly by Wirel. Work.* IEEE, Jun. 2011, pp. 1–4.
4. F. Chen, A. P. Chandrakasan, and V. M. Stojanovic, "Design and Analysis of a Hardware-Efficient Compressed Sensing Architecture for Data Compression in Wireless Sensors," *IEEE J. Solid-State Circuits*, vol. 47, no. 3, pp. 744–756, Mar. 2012.
5. T.-C. Chen, T.-h. Lee, Y.-H. Chen, T.-c. Ma, T.-d. Chuang, C.-j. Chou, C.-h. Yang, T.-h. Lin, and L.-g. Chen, "1.4μW/channel 16-channel EEG/ECoG processor for smart brain sensor SoC," in *2010 Symp. VLSI Circuits*. IEEE, Jun. 2010, pp. 21–22.
6. M. Weiser, "The Computer for the 21st Century," *Sci. Am.*, pp. 94–104, Sep. 1991.
7. M. Seyedi, B. Kibret, D. T. H. Lai, and M. Faulkner, "A survey on intrabody communications for body area network applications." *IEEE Trans. Biomed. Eng.*, vol. 60, no. 8, pp. 2067–79, Aug. 2013.
8. W. W. Y. Ng and D. S. Yeung, "A brief survey on current RFID applications," in *2009 Int. Conf. Mach. Learn. Cybern.*, no. July. IEEE, Jul. 2009, pp. 2330–2335.
9. T. Chi, M. Chen, and Q. Gao, "Implementation and Study of a Greenhouse Environment Surveillance System Based on Wireless Sensor Network," *2008 Int. Conf. Embed. Softw. Syst. Symp.*, pp. 287–291, Jul. 2008.
10. G. Chen, S. Hanson, D. Blaauw, and D. Sylvester, "Circuit Design Advances for Wireless Sensing Applications," *Proc. IEEE*, vol. 98, no. 11, pp. 1808–1827, Nov. 2010.
11. L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Networks*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.
12. M. Hempstead, M. J. Lyons, D. Brooks, and G.-Y. Wei, "Survey of Hardware Systems for Wireless Sensor Networks," *J. Low Power Electron.*, vol. 4, no. 1, pp. 11–20, Apr. 2008.

13. M. Pedram, "Clock-gating and its application to low power design of sequential circuits," *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.*, vol. 47, no. 3, pp. 415–420, Mar. 2000.

14. M. Münch, B. Wurth, R. Mehra, J. Sproch, and N. Wehn, "Automating RT-level operand isolation to minimize power consumption in datapaths," in *Proc. Conf. Des. Autom. test Eur. - DATE '00*. New York, New York, USA: ACM Press, 2000, pp. 624–633.

15. N. S. Mazloum, J. N. Rodrigues, and O. Edfors, "Sub-V_T design of a wake-up receiver back-end in 65 nm CMOS," in *2012 IEEE Subthreshold Microelectron. Conf.* IEEE, Oct. 2012, pp. 1–3.

16. J. Meindl and J. Davis, "The fundamental limit on binary switching energy for terascale integration (TSI)," *IEEE J. Solid-State Circuits*, vol. 35, no. 10, pp. 1515–1516, Oct. 2000.

17. B. Zhai, D. Blaauw, D. Sylvester, and K. Flautner, "Theoretical and practical limits of dynamic voltage scaling," in *Proc. 41st Annu. Conf. Des. Autom. - DAC '04*. New York, New York, USA: ACM Press, 2004, p. 868.

18. T. Sakurai and A. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE J. Solid-State Circuits*, vol. 25, no. 2, pp. 584–594, Apr. 1990.

19. B. Calhoun, A. Wang, and A. Chandrakasan, "Device sizing for minimum energy operation in subthreshold circuits," in *IEEE Cust. Integr. Circuits Conf.* IEEE, 2004, pp. 95–98.

20. C. Piguet, "Logic design for low-power CMOS circuits," in *IEEE TENCON. IEEE Reg. 10 Int. Conf. Microelectron. VLSI.* IEEE, 1995, pp. 299–302.

21. D. Blaauw and B. Zhai, "Energy efficient design for subthreshold supply voltage operation," *Circuits Syst. 2006.*, pp. 3–6, 2006.

22. A. Wang and A. Chandrakasan, "A 180-mV subthreshold FFT processor using a minimum energy design methodology," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 310–319, Jan. 2005.

23. B. Calhoun, A. Wang, and A. Chandrakasan, "Modeling and sizing for minimum energy operation in subthreshold circuits," *IEEE J. Solid-State Circuits*, vol. 40, no. 9, pp. 1778–1786, Sep. 2005.

24. B. Davari, R. Dennard, and G. Shahidi, "CMOS scaling for high performance and low power-the next ten years," *Proc. IEEE*, vol. 83, no. 4, pp. 595–606, Apr. 1995.

25. S. Borkar, "Design challenges of technology scaling," *IEEE Micro*, vol. 19, no. 4, pp. 23–29, 1999.

26. N. K. Tiwari, J. Shrivas, S. Akashe, and R. Sharma, "Impact of technology scaling and supply voltage variation on half adder design in nanometer era," in *2012 World Congr. Inf. Commun. Technol.* IEEE, Oct. 2012, pp. 33–38.

27. ITRS Process Integration, "Internation Roadmap for Semiconductors," ITRS, Tech. Rep., 2011.

28. D. Bol, "Robust and Energy-Efficient Ultra-Low-Voltage Circuit Design under Timing Constraints in 65/45 nm CMOS," *J. Low Power Electron. Appl.*, vol. 1, no. 3, pp. 1–19, Jan. 2011.

29. B. Calhoun, J. Ryan, S. Khanna, M. Putic, and J. Lach, "Flexible Circuits and Architectures for Ultralow Power," *Proc. IEEE*, vol. 98, no. 2, pp. 267–282, Feb. 2010.

30. D. Bol, C. Hocquet, and F. Regazzoni, "A Fast ULV Logic Synthesis Flow in Many-Vt CMOS Processes for Minimum Energy Under Timing Constraints," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 59, no. 12, pp. 947–951, Dec. 2012.

31. D. Bol, J. D. Vos, C. Hocquet, F. Botman, F. Durvaux, S. Boyd, and D. Flandre, "SleepWalker: A 25-MHz 0.4-V Sub-mm^2 7-μW/MHz Microcontroller in 65-nm LP/GP CMOS for Low-Carbon Wireless Sensor Nodes," *IEEE J. Solid-State Circuits*, vol. 48, no. 1, pp. 20–32, 2013.

32. K. Roy, J. Kulkarni, and M. Hwang, "Low-voltage process-adaptive logic and memory arrays for ultralow-power sensor nodes," *2009 IEEE Sensors*, pp. 185–188, Oct. 2009.

33. L. Chang, R. K. Montoye, Y. Nakamura, K. A. Batson, R. J. Eickemeyer, R. H. Dennard, W. Haensch, and D. Jamsek, "An 8T-SRAM for Variability Tolerance and Low-Voltage Operation in High-Performance Caches," *IEEE J. Solid-State Circuits*, vol. 43, no. 4, pp. 956–963, Apr. 2008.

34. N. Ickes and D. Finchelstein, "A 10-pJ/instruction, 4-MIPS micropower DSP for sensor applications," in *ASSCC*, 2008, pp. 289–292.

35. C. Hocquet, F. Botman, D. Bol, and J.-d. Legat, "A near-threshold instruction cache with zero miss overhead time for dual- V dd microcontrollers," in *SubVt*, 2011.

36. N. J. Ickes, "A Micropower DSP for Sensor Applications," Ph.D. dissertation, Massachusetts Institute of Technology, 2008.

37. P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle : A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks," in *NSDI*, 2004.

38. F. Zhang, Y. Zhang, J. Silver, Y. Shakhsheer, M. Nagaraju, A. Klinefelter, J. Pandey, E. Carlson, A. Shrivastava, B. Otis, and B. Calhoun, "A Batteryless 19μW MICS/ISM-Band Energy Harvesting Body Area Sensor Node SoC," in *ISSCC*, 2012, pp. 3–4.

39. S. Luetkemeier, T. Jungeblut, and M. Porrmann, "A 200mV 32b Subthreshold Processor With Adaptive Supply Voltage Control," in *ISSCC*, vol. 57, 2012, pp. 4–5.

40. T. Feng, B. Jin, J. Wang, N. Park, Y. B. Kim, and F. Lombardi, "Fault tolerant clockless wave pipeline design," in *Proc. first Conf. Comput. Front. Comput. Front. - CF'04*. New York, New York, USA: ACM Press, 2004, p. 350.

41. S. Furber, J. Garside, S. Temple, J. Liu, P. Day, and N. Paver, "AMULET2e: an asynchronous embedded controller," in *Proc. Third Int. Symp. Adv. Res. Asynchronous Circuits Syst.* IEEE Comput. Soc. Press, Apr. 1997, pp. 290–299.

42. D. Tabak, "Move Architecture in Digital Controllers," *IEEE Trans. Comput.*, vol. C-29, no. 2, pp. 180–190, Feb. 1980.

43. A. W. Luczyk, "Superscalar Move Architecture for Power-Aware Computing," in *2007 14th Int. Conf. Mix. Des. Integr. Circuits Syst.* IEEE, Jun. 2007, pp. 349–354.

44. N. Clark and S. Mahlke, "Processor acceleration through automated instruction set customization," in *22nd Digit. Avion. Syst. Conf. Proc. (Cat. No.03CH37449)*. IEEE Comput. Soc, 2003, pp. 129–140.

45. N. Chabini and W. Wolf, "Unification of scheduling, binding, and retiming to reduce power consumption under timings and resources constraints," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 13, no. 10, pp. 1113–1126, Oct. 2005.

46. P. Stenström, M. Dubois, M. Katevenis, R. Gupta, and T. Ungerer, "Coffee: COmpiler Framework for Energy-Aware Exploration," in *High Perform. Embed. Archit. Compil.*, P. Raghavan, A. Lambrechts, J. Absar, M. Jayapala, F. Catthoor, and D. Verkest, Eds. Springer Berlin Heidelberg, 2008, pp. 193–208.

47. a. Asenov, "Random dopant induced threshold voltage lowering and fluctuations in sub-0.1 μm MOSFET's: A 3-D "atomistic" simulation study," *IEEE Trans. Electron Devices*, vol. 45, no. 12, pp. 2505–2513, 1998.

48. K. Bernstein, D. J. Frank, A. E. Gattiker, W. Haensch, B. L. Ji, S. R. Nassif, E. J. Nowak, D. J. Pearson, and N. J. Rohrer, "High-performance CMOS variability in the 65-nm regime and beyond," *IBM J. Res. Dev.*, vol. 50, no. 4.5, pp. 433–449, Jul. 2006.

49. J. Kwong and A. P. Chandrakasan, "Variation-Driven Device Sizing for Minimum Energy Sub-threshold Circuits," in *ISLPED'06 Proc. 2006 Int. Symp. Low Power Electron. Des.* IEEE, Oct. 2006, pp. 8–13.

50. D. Bull, S. Das, K. Shivashankar, G. S. Dasika, K. Flautner, and D. Blaauw, "A Power-Efficient 32 bit ARM Processor Using Timing-Error Detection and Correction for Transient-Error Tolerance and Adaptation to PVT Variation," *IEEE J. Solid-State Circuits*, vol. 46, no. 1, pp. 18–31, Jan. 2011.

51. J. De Vos, D. Flandre, and D. Bol, "Pushing adaptive voltage scaling fully on chip," *ASP J. Low-Power Electron.*, vol. 8, pp. 95–105, 2012.

52. T. Burd, T. Pering, A. Stratakos, and R. Brodersen, "A dynamic voltage scaled microprocessor system," in *IEEE Int. Solid-State Circuits Conf.*, vol. 35, no. 11. IEEE, 2000, pp. 294–295,.

53. J. Makipaa, E. Laulainen, M. J. Turnquist, and L. Koskinen, "Subthreshold timing error detection performance analysis," in *2010 12th Bienn. Balt. Electron. Conf.* IEEE, Oct. 2010, pp. 121–124.

54. D.-U. Lee, L.-W. Kim, and J. D. Villasenor, "Precision-aware self-quantizing hardware architectures for the discrete wavelet transform." *IEEE Trans. Image Process.*, vol. 21, no. 2, pp. 768–77, Feb. 2012.

55. D. Ernst, N. Kim, S. Das, and S. Pant, "Razor: A low-power pipeline based on circuit-level timing speculation," *MICRO*, no. December, 2003.

56. E. Maricau and G. Gielen, *Analog IC Reliability in Nanometer CMOS*. New York, NY: Springer New York, 2013.

57. T. Okada, H. Yoshimura, H. Aikawa, M. Sengoku, O. Fujii, and H. Oyamatsu, "A comparative 3D simulation approach with extensive experimental Vt/Avt data

and analysis of LER/RDF/reliability of CMOS SRAMs at 40-nm node and beyond," in *2010 Int. Conf. Simul. Semicond. Process. Devices*, no. 1. IEEE, Sep. 2010, pp. 121–124.

58. Y. Ban, S. Sundareswaran, R. Panda, and D. Z. Pan, "Electrical impact of line-edge roughness on sub-45nm node standard cell," in *Des. Manuf. through Des. Integr. III*, V. K. Singh and M. L. Rieger, Eds., Mar. 2009, pp. 727 518–727 518–10.
59. W. P. Griffin, A. Raghunathan, and K. Roy, "CLIP: Circuit Level IC Protection Through Direct Injection of Process Variations," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 20, no. 5, pp. 791–803, May 2012.
60. R. Rithe, S. Chou, A. Wang, S. Datla, G. Gammie, D. Buss, and A. Chandrakasan, "The Effect of Random Dopant Fluctuations on Logic Timing at Low Voltage," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 20, no. 5, pp. 911–924, May 2012.
61. M. Seok, D. Jeon, and C. Chakrabarti, "Pipeline strategy for improving optimal energy efficiency in ultra-low voltage design," in *DAT*. New York, New York, USA: ACM Press, 2011, p. 990.
62. B. Zhai, S. Hanson, D. Blaauw, and D. Sylvester, "Analysis and mitigation of variability in subthreshold design," *Proc. 2005 Int. Symp. Low power Electron. Des. - ISLPED '05*, p. 20, 2005.
63. J. Kwong, Y. Ramadass, N. Verma, M. Koesler, K. Huber, H. Moormann, and A. Chandrakasan, "A 65nm Sub-V_t Microcontroller with Integrated SRAM and Switched-Capacitor DC-DC Converter," in *2008 IEEE Int. Solid-State Circuits Conf. - Dig. Tech. Pap.* IEEE, Feb. 2008, pp. 318–616.
64. G. Gammie, N. Ickes, M. E. Sinangil, R. Rithe, J. Gu, A. Wang, H. Mair, S. Datla, B. Rong, S. Honnavara-Prasad, L. Ho, G. Baldwin, D. Buss, A. P. Chandrakasan, and U. Ko, "A 28nm 0.6V low-power DSP for mobile applications," in *ISSCC*, no. March 2010. IEEE, Feb. 2011, pp. 132–134.
65. S. Narendra, V. De, S. Borkar, D. Antoniadis, and A. Chandrakasan, "Full-Chip Subthreshold Leakage Power Prediction and Reduction Techniques for Sub-0.18μm CMOS," *IEEE J. Solid-State Circuits*, vol. 39, no. 3, pp. 501–510, Mar. 2004.
66. D. Bol, D. Flandre, and J.-D. Legat, "Nanometer MOSFET Effects on the Minimum-Energy Point of Sub-45nm Subthreshold Logic—Mitigation at Technology and Circuit Levels," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 16, no. 1, pp. 1–26, Nov. 2010.
67. J. Chen, L. Clark, and T.-H. Chen, "An Ultra-Low-Power Memory With a Subthreshold Power Supply Voltage," *IEEE J. Solid-State Circuits*, vol. 41, no. 10, pp. 2344–2353, Oct. 2006.
68. D. Bol, R. Ambroise, D. Flandre, and J.-D. Legat, "Analysis and minimization of practical energy in 45nm subthreshold logic circuits," in *2008 IEEE Int. Conf. Comput. Des.* IEEE, Oct. 2008, pp. 294–300.
69. S. Jocke, J. Bolus, S. Wooters, A. Jurik, A. Weaver, T. Blalock, and B. Calhoun, "A 2.6-μW sub-threshold mixed-signal ECG SoC," *VLSI Circuits, 2009 ...*, pp. 60–61, 2009.
70. M. Seok, D. Blaauw, and D. Sylvester, "Clock network design for ultra-low power applications," in *Proc. ACM Int. Symp. Low-Power Electron. Des.*, 2010, pp. 271–276.

71. D. Kamel, C. Hocquet, F.-X. Standaert, D. Flandre, and D. Bol, "Glitch-induced within-die variations of dynamic energy in voltage-scaled nano-CMOS circuits," in *2010 Proc. ESSCIRC*. IEEE, Sep. 2010, pp. 518–521.

72. A. Hande, T. Polk, W. Walker, and D. Bhatia, "Indoor solar energy harvesting for sensor network router nodes," *Microprocess. Microsyst.*, vol. 31, no. 6, pp. 420–432, Sep. 2007.

73. J. Randall and J. Jacot, "The Performance and Modelling of 8 Photovoltaic Materials under Variable Light Intensity and Spectra," in *World Renew. Energy Conf. VII*, 2002.

74. B. Warneke, B. Atwood, and K. Pister, "Smart dust mote forerunners," in *Tech. Dig. MEMS 2001. 14th IEEE Int. Conf. Micro Electro Mech. Syst.*, no. Mems. IEEE, 2001, pp. 357–360.

75. S. Hanson, M. Seok, Y.-s. Lin, Z. Foo, D. Kim, Y. Lee, N. Liu, D. Sylvester, and D. Blaauw, "A Low-Voltage Processor for Sensing Applications With Picowatt Standby Mode," *IEEE J. Solid-State Circuits*, vol. 44, no. 4, pp. 1145–1155, Apr. 2009.

76. B. Zhai, L. Nazhandali, J. Olson, A. Reeves, M. Minuth, R. Helfand, S. Pant, D. Blaauw, and T. Austin, "A 2.60pJ/Inst Subthreshold Sensor Processor for Optimal Energy Efficiency," in *VLSI*. IEEE, 2006, pp. 154–155.

77. M. Zwerg, A. Baumann, R. Kuhn, M. Arnold, R. Nerlich, M. Herzog, R. Ledwa, C. Sichert, V. Rzehak, P. Thanigai, and B. O. Eversmann, "An 82μA/MHz microcontroller with embedded FeRAM for energy-harvesting applications," in *2011 IEEE Int. Solid-State Circuits Conf.*, vol. 4, no. 3. IEEE, Feb. 2011, pp. 334–336.

78. N. Ickes, Y. Sinangil, F. Pappalardo, E. Guidetti, and A. P. Chandrakasan, "A 10 pJ / cycle ultra-low-voltage 32-bit microprocessor system-on-chip," *ESSCIRC*, pp. 159–162, 2011.

79. S. C. Bartling, S. Khanna, M. P. Clinton, S. R. Summerfelt, J. A. Rodriguez, and H. P. McAdams, "An 8MHz 75μA/MHz zero-leakage non-volatile logic-based Cortex-M0 MCU SoC exhibiting 100% digital state retention at V_{DD}=0V with <400ns wakeup and sleep transitions," in *ISSCC*. IEEE, Feb. 2013, pp. 432–433.

80. C. Arm, S. Gyger, J.-m. Masgonty, M. Morgan, J.-l. Nagel, C. Piguet, F. Rampogna, and P. Volet, "Low-Power 32-bit Dual-MAC 120 μW/MHz 1.0 V icyflex1 DSP/MCU Core," *IEEE J. Solid-State Circuits*, vol. 44, no. 7, pp. 2055–2064, Jul. 2009.

81. M. Fojtik, D. Kim, G. Chen, Y.-s. Lin, D. Fick, J. Park, M. Seok, M.-t. Chen, Z. Foo, D. Blaauw, and D. Sylvester, "A Millimeter-Scale Energy-Autonomous Sensor System With Stacked Battery and Solar Cells," *IEEE J. Solid-State Circuits*, vol. 48, no. 3, pp. 801–813, Mar. 2013.

82. D. Fick, R. G. Dreslinski, B. Giridhar, G. Kim, S. Seo, M. Fojtik, S. Satpathy, Y. Lee, D. Kim, N. Liu, M. Wieckowski, G. Chen, T. Mudge, D. Sylvester, and D. Blaauw, "Centip3De: A 3930DMIPS/W configurable near-threshold 3D stacked system with 64 ARM Cortex-M3 cores," in *2012 IEEE Int. Solid-State Circuits Conf.* IEEE, Feb. 2012, pp. 190–192.

83. W.-x. Tang, S.-w. Tung, and K.-y. Lin, "A 70μW/MHz ultra-low voltage microcontroller SPARK," in *2013 IEEE Int. Conf. Electron Devices Solid-state Circuits*. IEEE, Jun. 2013, pp. 1–2.

84. J. Choi, S. Park, J. Cho, and E. Yoon, "A 3.4-μW Object-Adaptive CMOS Image Sensor With Embedded Feature Extraction Algorithm for Motion-Triggered Object-of-Interest Imaging," *IEEE J. Solid-State Circuits*, vol. 49, no. 1, pp. 289–300, Jan. 2014.

85. J. De Vos, D. Flandre, and D. Bol, "A Sizing Methodology for On-Chip Switched-Capacitor DC/DC Converters," *IEEE Trans. Circuits Syst. I Regul. Pap.*, pp. 1–1, 2014.

86. J. De Vos, "Switched–Capacitor DC/DC Converters in Nanometer CMOS Technologies for Micro-Power Energy Management," Ph.D. dissertation, Université catholique de Louvain, 2013.

87. G. de Streel, J. De Vos, D. Flandre, and D. Bol, "A 65nm 1V to 0.5V linear regulator with ultra low quiescent current for mixed-signal ULV SoCs," in *FTFC*. IEEE, May 2014, pp. 1–4.

88. D. Bol, G. D. Streel, F. Botman, A. K. Lusala, and N. Couniot, "A 65-nm 0.5-V 17-pJ / frame.pixel DPS CMOS Image Sensor for Ultra-Low-Power SoCs achieving 40-dB Dynamic Range," in *Symp. VLSI*, 2014.

89. F. Botman, J. De Vos, S. Bernard, J.-d. Legat, and D. Bol, "Bellevue: a 50MHz Variable-Width SIMD 32bit Microcontroller at 0.37V for Processing-Intensive Wireless Sensor Nodes," in *ISCAS*, 2014.

90. S. Hanson, B. Zhai, M. Seok, B. Cline, K. Zhou, M. Singhal, M. Minuth, J. Olson, L. Nazhandali, T. Austin, D. Sylvester, and D. Blaauw, "Exploring Variability and Performance in a Sub-200-mV Processor," *IEEE J. Solid-State Circuits*, vol. 43, no. 4, pp. 881–891, Apr. 2008.

91. A. Medi and W. Namgoong, "A High Data-Rate Energy-Efficient Interference-Tolerant Fully Integrated CMOS Frequency Channelized UWB Transceiver for Impulse Radio," *IEEE J. Solid-State Circuits*, vol. 43, no. 4, pp. 974–980, Apr. 2008.

92. M. Vidojkovic, X. Huang, X. Wang, C. Zhou, A. Ba, M. Lont, Y.-h. Liu, P. Harpe, M. Ding, B. Busze, N. Kiyani, K. Kanda, S. Masui, K. Philips, and H. de Groot, "A 0.33nJ/b IEEE802.15.6/proprietary-MICS/ISM-band transceiver with scalable data-rate from 11kb/s to 4.5Mb/s for medical applications," in *2014 IEEE Int. Solid-State Circuits Conf. Dig. Tech. Pap.* IEEE, Feb. 2014, pp. 170–171.

93. R. K. Dokania, X. Y. Wang, S. G. Tallur, and A. B. Apsel, "A Low Power Impulse Radio Design for Body-Area-Networks," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 58, no. 7, pp. 1458–1469, Jul. 2011.

94. Z.-L. He, T. Chi-Ying, K.-K. Chan, and M. L. Liou, "Low-power VLSI design for motion estimation using adaptive pixel truncation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 5, pp. 669–678, 2000.

95. M. Dean, D. Dill, and M. Horowitz, "Self-timed logic using current-sensing completion detection (CSCD)," in *[1991 Proceedings] IEEE Int. Conf. Comput. Des. VLSI Comput. Process.* IEEE Comput. Soc. Press, Dec. 1991, pp. 187–191.

96. S. Ghosh, S. Bhunia, and K. Roy, "CRISTA: A new paradigm for low-power, variation-tolerant, and adaptive circuit synthesis using critical path isolation," *Comput. Des.*, vol. 26, no. 11, pp. 1947–1956, 2007.

97. S. Das, C. Tokunaga, S. Pant, W.-h. Ma, S. Kalaiselvan, K. Lai, D. M. Bull, and D. T. Blaauw, "RazorII: In Situ Error Detection and Correction for PVT and SER Tolerance," *IEEE J. Solid-State Circuits*, vol. 44, no. 1, pp. 32–48, Jan. 2009.

98. M. Seok, D. Jeon, C. Chakrabarti, D. Blaauw, and D. Sylvester, "A 0.27V 30MHz 17.7nJ/transform 1024-pt complex FFT core with super-pipelining," in *2011 IEEE Int. Solid-State Circuits Conf.* IEEE, Feb. 2011, pp. 342–344.

99. V. Kursun and E. Friedman, "Domino logic with variable threshold voltage keeper," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 11, no. 6, pp. 1080–1093, Dec. 2003.

100. A. Sridharan, C. Sechen, and R. Jafari, "Low-voltage low-overhead asynchronous logic," in *Int. Symp. Low Power Electron. Des.* IEEE, Sep. 2013, pp. 261–266.

101. F. Arnaud, B. Duriez, B. Tavel, L. Pain, J. Todeschini, M. Jurdit, Y. Laplanche, F. Boeuf, F. Salvetti, D. Lenoble, J. Reynard, F. Wacquant, P. Morin, N. Emonet, D. Barge, M. Bidaud, D. Ceccarelli, P. Vannier, Y. Loquet, H. Leninger, F. Judong, C. Perrot, I. Guilmeau, R. Palla, A. Beverina, V. DeJonghe, M. Broekaart, V. Vachellerie, R. Bianchi, B. Borot, T. Devoivre, N. Bicais, D. Roy, M. Denais, K. Rochereau, R. Difrenza, N. Planes, H. Brut, L. Vishnobulta, D. Reber, P. Stolk, and M. Woo, "Low cost 65nm CMOS platform for Low Power & General Purpose applications," in *Dig. Tech. Pap. 2004 Symp. VLSI Technol. 2004.* IEEE, 2004, pp. 10–11.

102. B. Calhoun, A. Wang, and A. Chandrakasan, "Modeling and sizing for minimum energy operation in subthreshold circuits," *IEEE J. Solid-State Circuits*, vol. 40, no. 9, pp. 1778–1786, Sep. 2005.

103. Y. Shin, K. Shin, P. Kenkare, R. Kashyap, H.-J. Lee, D. Seo, B. Millar, Y. Kwon, R. Iyengar, M.-S. Kim, A. Chowdhury, S.-I. Bae, I. Hong, W. Jeong, A. Lindner, U. Cho, K. Hawkins, J. C. Son, and S. H. Hwang, "28nm high- metal-gate heterogeneous quad-core CPUs for high-performance and energy-efficient mobile application processor," in *ISSCC.* IEEE, Feb. 2013, pp. 154–155.

104. J. De Vos, D. Flandre, and D. Bol, "A dual-mode DC/DC converter for ultra-low-voltage microcontrollers," in *2012 IEEE Subthreshold Microelectron. Conf.* IEEE, Oct. 2012, pp. 1–3.

105. F. Firouzi, S. Kiamehr, and M. B. Tahoori, "NBTI mitigation by optimized NOP assignment and insertion," in *2012 Des. Autom. Test Eur. Conf. Exhib.* IEEE, Mar. 2012, pp. 218–223.

106. J. Makipaa and O. Billoint, "FDSOI versus BULK CMOS at 28 nm node which technology for ultra-low power design?" in *2013 IEEE Int. Symp. Circuits Syst.* IEEE, May 2013, pp. 554–557.

107. G. de Streel and D. Bol, "Scaling perspectives of ULV microcontroller cores to 28nm UTBB FDSOI CMOS," in *2013 IEEE SOI-3D-Subthreshold Microelectron. Technol. Unified Conf.* IEEE, Oct. 2013, pp. 1–2.

108. X. Wu, F. Wang, and Y. Xie, "Analysis of Subthreshold Finfet Circuits for Ultra-Low Power Design," *2006 IEEE Int. SOC Conf.*, pp. 91–92, Sep. 2006.

109. M. Motoyoshi, "Through-Silicon Via (TSV)," *Proc. IEEE*, vol. 97, no. 1, pp. 43–48, Jan. 2009.

110. R. Mertens, "Trends in solar cell research," in *2008 15th Int. Symp. Phys. Fail. Anal. Integr. Circuits*. IEEE, Jul. 2008, pp. 1–5.

111. P. Mitcheson, E. Yeatman, G. Rao, A. Holmes, and T. Green, "Energy Harvesting From Human and Machine Motion for Wireless Electronic Devices," *Proc. IEEE*, vol. 96, no. 9, pp. 1457–1486, Sep. 2008.

112. D. Evans, "The Internet of Things: How the Next Evolution of the Internet Is Changing Everything," Cisco, Tech. Rep. April, 2011.

113. T. Feng, Z. Patitz, and N. Park, "A Design Method for Ultra-High Speed for A New Wave Pipeline-based Circuit," in *2008 IEEE Instrum. Meas. Technol. Conf.* IEEE, May 2008, pp. 1493–1498.