System on Chip (SoC) for Body Sensor Nodes: A system level approach

---

A Thesis

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

---

in partial fulfillment

of the requirements for the degree

Master of Science

by

Luisa Patricia Gonzalez Guerrero

December

2015

APPROVAL SHEET

The thesis

is submitted in partial fulfillment of the requirements

for the degree of

Master of Science

_____

AUTHOR

The thesis has been read and approved by the examining committee:

Benton H. Calhoun

_____

Advisor

Scott T. Acton

_____

Ronald Williams

_____

_____

_____

_____

Accepted for the School of Engineering and Applied Science:

Craig H. Benson, Dean, School of Engineering and Applied Science

December

2015

System on Chip (SoC) for Body Sensor Nodes: A system level approach

M.S. Thesis

Luisa Patricia Gonzalez Guerrero
lg4er@virginia.edu

June 30, 2015

Acknowledgement

I would like to thank my advisor Professor Benton H. Calhoun. The members of my comitte, Professors Ronald Williams and Scott Acton. My collegues during graduate school at University of Virginia, North Carolina State University and University of Michigan. Special words of apreciation to my family, the one that we build as we go, and the one we don't choose but what a crew!! Thanks for your non stopping support.

Abstract

The last 20 years have seen the development of several technologies that individually play an important role to enable Wireless Sensor Nodes to become a reality. First, the energy consumption has been reduced to the order of decades of μW thanks to digital and analog ultra-low power circuits that are able to operate reliably in the subthreshold regime. Second, the ability to harvest energy from the environment eliminates the dependency on bulky batteries and elongates the life of sensor nodes. These sensors must support sensing modalities and processing capabilities for a broad range of applications that can go from the most commonly showcased ECG (Electrocardiogram) to more demanding applications as wheezing detection for asthma management. All these components considerably increment the design complexity associated with these highly integrated SoCs. Developing a node for each application with the current RTL methodologies is not possible due to the prohibitive Non Recurring Costs (NRE) associated with the design and verification of such a complex systems. On the other hand, it has been proven that the optimum operation point for an ultra low power system highly depends on the application and the activity factor associated with it, thus a platform optimum for one application might be sub-optimal or not suitable for others. This scenario raises awareness to a question that has not been explored yet in the sensor nodes literature, "How do we efficiently integrate these complex Systems On Chip (SoCs) taking into account the application particular requirements".

This thesis explores these system level questions associated with SoC design and proposes two complementing approaches to address them. First, we design a re-use and verification aware methodology for wireless sensor nodes. This methodology uses interface based design and block abstraction. The combination of these two techniques, enable RTL configurability at the system level to meet particular application requirements, and allow hardware evolution/modification reducing the design NRE costs. We applied this methodology during the integration of a $6.45\mu W$ Self-Powered IoT SoC with Integrated Energy-Harvesting Power Management and ULP Asymmetric Radios. This SoC has the highest level of integration to its publication date. Second, inspired by the necessity of a system level approach, that covers architecture exploration and application validation in wireless sensor nodes, we setup an Application Oriented framework. This framework facilitates the communication between the algorithm developer, the hardware designer and the hardware verifier in order to explore the hardware trade-offs associated to a particular application. We called this framework AVEBoS and is intended for early exploration or late validation. Early exploration is

needed so the architecture is defined taking into account the requirements of one or a set of applications. Late validation is required to evaluate the performance of the node for a particular application under a specified condition. Using this framework and the re-use ideas explored for the SoC integration, we design, verified and evaluated the performance of a Short Time Fourier Transform accelerator for wheezing detection and a FFT-IFFT based 6 μW Pulse Oximetry Digital Signal Processing Data Path.

Following these system level strategies we expect to optimize the engineering effort invested in a single SoC, to start paving the path towards an architecture that gives birth to a family of SoCs able to adapt to different application requirements and operation conditions with a design cost that is spread across different solutions by the use of truly re-usable IP.

Contents

List of Figures

Glossary

SoC : System On Chip.

NRC : Non Recurrent Costs.

IoT : Internet of Things.

ExG : ECG, Electrocardiogram and EMG FIXME.

SN : Sensor Nodes.

IIR : Infinite Impulse Response.

PMU : Power Management Unit.

SEMATHEC : Semiconductor Manufacturing Technology.

ITRS: International Technology Roadmap for Semiconductors.

VE:Verification Environment.

HDL:Hardware Description Language.

RTL: Register Transfer Level.

DUT: Device Under Test.

Chapter 1

INTRODUCTION

The last 20 years have seen the development of several technologies that enabled Wireless Sensor Nodes to become a reality. Obstacles as bulky size due to batteries, and the energy constrains due to the wireless nature of the nodes, have been addressed; i.e. Novel Power Management Units (PMUs) with end to end high efficiency enable energy harvesting. This energy is collected in supercapacitors that support the whole node operation[1]. CMOS circuits in both the analog and digital domain optimized to operate in the subthreshold regime enable power consumption on the order of a couple of decades of μ W[2][3][4]. Ultralow power radios combined with heavy duty cycling[5] allow wireless communication during important programmable events[6][7]. All these circuits designed on isolation represent the best on their classes, and one of the remaining challenge is yet to address. How do we integrate all these components?

1.1 Outline

1.1.1 Integration Complexity

This thesis addresses some of the challenges faced during the Integration at the RTL behavioral level of a 6.45μW Self-Powered IoT SoC with on-chip Energy-Harvesting Power Management and Ultralow Power Asymmetric Radios[8]. The increased complexity of a node of this nature has a direct impact on the Non Recurring Costs (NRE) associated with development. Fig.1.1 shows the design complexity increment following Moore's law, in contrast with the productivity per engineer evolution over time. The productivity

gap created by the different evolution rates, dramatically increment the cost of design, placing this issue as one of the main threats for the semiconductor industry Moore's law[9].

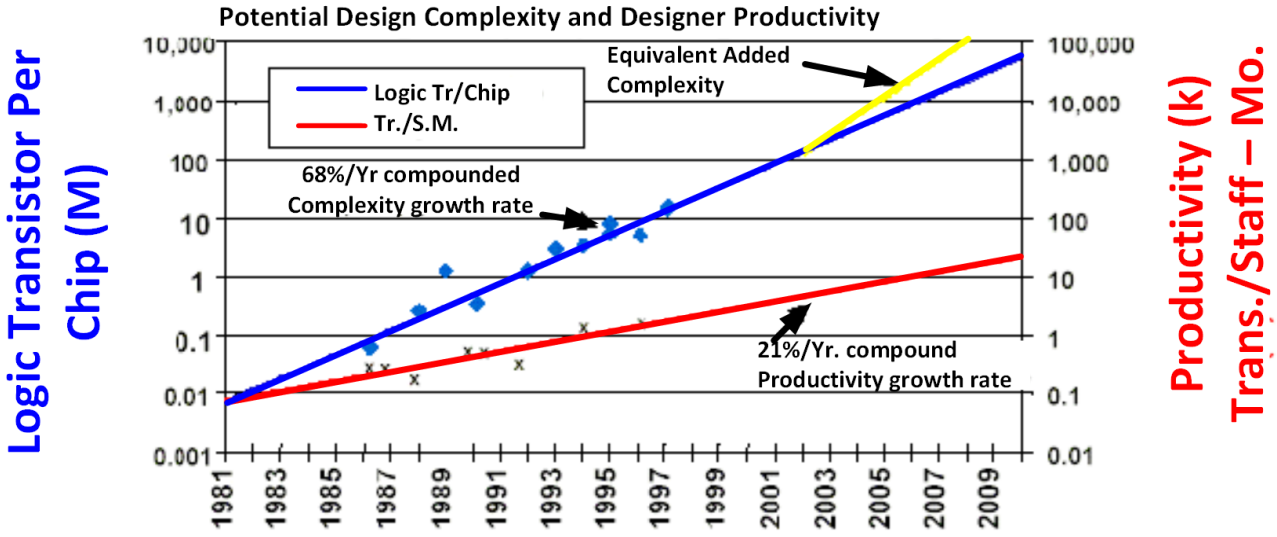

Figure 1.1: The productivity gap[10] predicted by SEMATHEC in 1997 and discussed by the ITRS in 2011[9] and 2013[11]

Chapter 2.1, proposes the use of interface-based design, block abstractions and a verification aware methodology for reducing the NRE costs associated with the design of highly integrated wireless sensor nodes. By using our methodology we argue digital design effort savings between $30-90\%$.

1.1.2 Application Oriented System Optimization and Verification Awareness

Wireless Sensor Nodes sense, process, and transmit physiological and environmental data. The conception of such a system starts with an application or a set of applications to be supported on node, i.e ECG or pulse oximetry. During the first stages of architecture conception, hardware architects, digital designers, verifiers and algorithm developers start modeling efforts that are usually disconnected. This broken link between SoC development collaborators leads to duplicated efforts in model generation.

Chapter 3.1 proposes An Architecture Agnostic Verification Environment (AVEBoS) that connects the Algorithm Developers and the Hardware Engineers. This chapter describes a tool that enables Hardware Architects, Digital Designers and Functional Pre-silicon Verifiriers to re-use the models generated by the Algorithm Developers during exploration/evaluation/development. At the same time, proposes an RTL de-

velopment methodology for Digital Designers to generate architecture agnostic IP that can be placed in a library for re-use across architectures.

The combination tool+methodology enables both ways communication. It enables the algorithm developer to explore bio-signal processing algorithms in a hardware aware way with the closest software description of a hardware IP: "Its behavioral RTL description". On the other hand, It allows the digital designer to integrate the modeling efforts started during algorithm development for platform validation and exploration.

Section 3.5, uses AVEBoS to generate a verification environment for a Short Time Fourier Transform (STFT) accelerator used for Wheezing Detection.

Beyond ECG

Chapter 4 presents a parameterizable pulse oximetry digital signal processing datath. This IP was desgined using the techniques described in Chapter 2.1 and Chapter 3.1, to create a truly re-usable IP block. In this way, we can easily generate an IP for a platform driven by energy constrains, with very low sampling rate and low precision demands or place the same IP in a platform that requires high levels of precision but have more energy available. AVEBoS enabled us to validate the IP for all the options, while generates a model that can be used at higher levels of abstraction.

And how does Ultralow Power fit in all this?

Energy consumption described by equation 1.1 is one of the most important optimization knobs for Wireless Body Sensors

$$E = \frac{1}{2}CV^2 f\alpha + I_{leak} \tag{1.1}$$

An Ultralow power SoC designed to fit the demands of a wireless sensor typically show cases one or several techniques for energy optimization. i.e. Subthreshold operation combined with a reduction in the operating frequency reduces the Voltage (V) and Frequency (f) [12]. Serializing the processing data path at the arithmetic unit level i.e. a serial adder[13], or at the block level i.e. IIR channels that share arithmetic units

[14] reduces the factor Capacitance (C) and leakage current (I_{leak}). And finally, Power gating and Clock gating reduces the consumption of idle power in under-utilized blocks (I_{leak}). However all this optimizations present tradeoffs, breaking points or optimum energy operation points, that will be dictated by application requirements as: activity factor α, sampling T_s and processing rates F, or expected accuracy. Fig.1.2 shows the Energy Vs VDD for different n_{eff} for an AND chain, where n is the length of the chain and α is the activity factor.

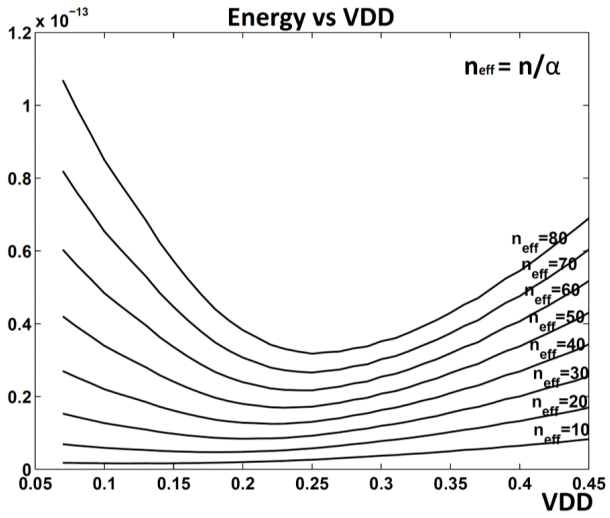

Figure 1.2: Energy Vs VDD for different n_{eff} for an AND chain, where n is the length of the chain and α is the activity factor [15]

Chapter4 describes power consumption considerations in a pulse oximetry datapath taking into account the particular constrains of the application in consideration.

1.2 Contributions of this thesis

This thesis makes the following contributions

- Proposes a methodology for Block abstraction and interface based design for body sensor nodes that reduces NRE costs in design and verification (Chapters 2.1 and 3.1)
- Uses this methodology to facilitate bus architecture evolution, modification, and or migration with no NRE cost associated (Chapter 2.1)

- Proposes an Architecture Agnostic Verification Environment for Body Sensor Nodes and demonstrates its use for the development of an accelerator for Wheezing Detection.(Chapter 3.1)
- Demonstrates a pulse oximeter FFT-IFFT based digital signal processing data path that operates under $6.4\mu W$. Architecture decisions are based on the application constrains. Application oriented architecture optimizations in the register files shows the tradeoffs in area and power relevant for the application.(Chapter 4)

Chapter 2

BSN

2.1 Introduction

The IoT term is used to describe a network in the near future where just about every device has embedded connectivity and some sort of intelligence. Some predictions talk about a value added to the global economy of $1.6-$4.6 trillions of dollars with a dramatic impact on city planning, first responders, military, health and dozens of other environments[16]. This recent explosion of the IoT will require a family of ultra low power SoCs targeted for a broad set of applications with very different sensing and processing requirements. However, the NRE cost associated to such a platform design can still be prohibitively high.

Since re-use and verification aware methodologies has the potential to increase productivity up to %340[9], thus reducing the development costs; we designed a re-use and verification aware methodology for Wireless Sensor Nodes. The methodology was adopted to develop our highly integrated, flexible SoC platform, which supports multiple sensing modalities, extracts information from data flexibly across applications, harvests and delivers power eficiently, and communicates wirelessly[8].

Note : Dear reader, Since several people (by several I mean 12 co-authors and a lot more that got drawn by the force, in one way or another) contributed to the development of this node, section 2.2 gives an overview of the platform while the rest of this chapter follows with the re-use and verification aware methodology proposed and tested for this development. For more information about the cutting-edge individual blocks and other integration details please refer to the following references: [1][2][3][4][6][7]

2.2 The Ultra Low Power IoT SoC

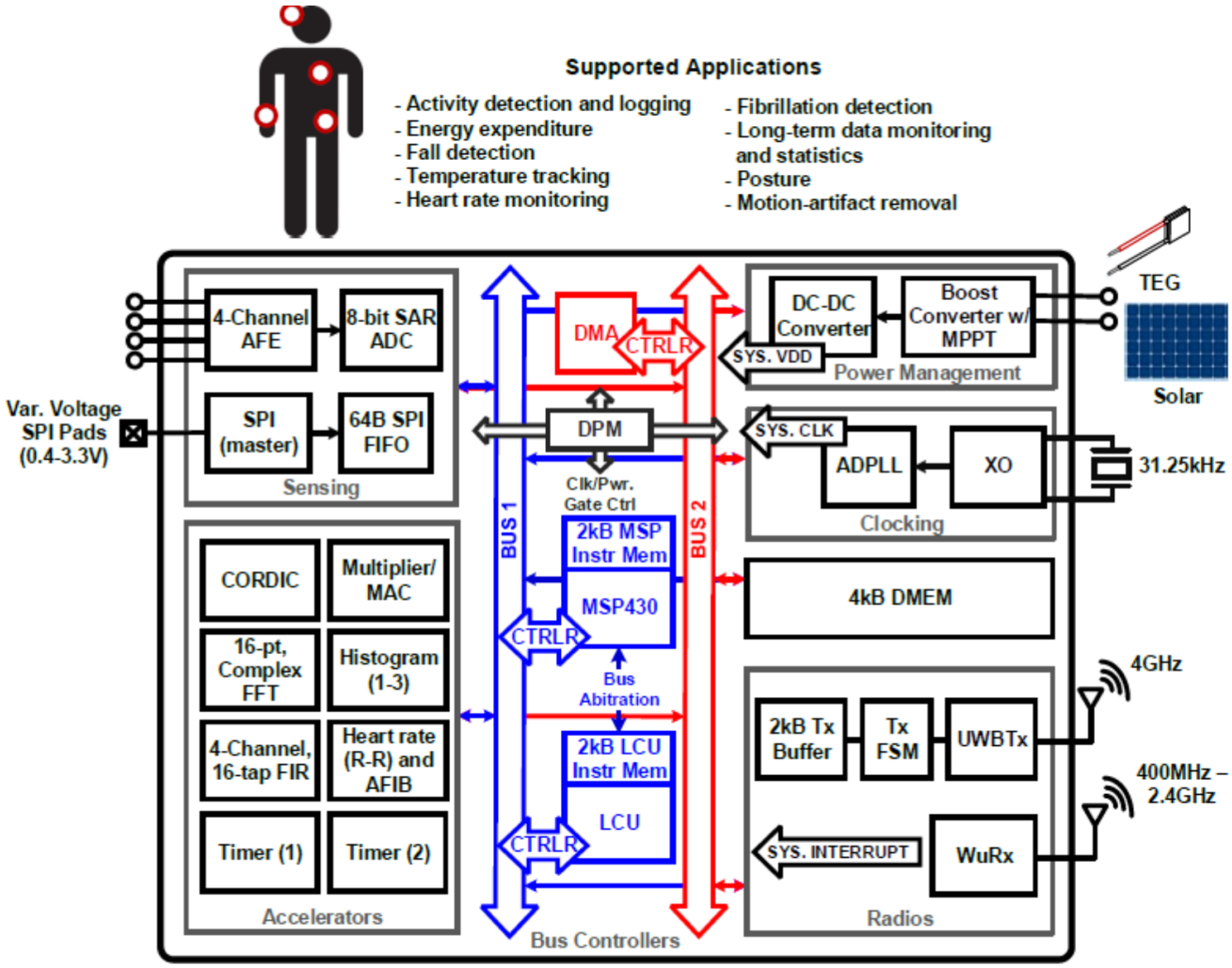

Figure 2.1: Self-powered IoT SoC Block Diagram[8].

We developed a 21.3A 6.45 μW self-powered IoT SoC with integrated energy-harvesting power management and ULP asymmetric radios, shown in Fig.2.1. This SoC integrates a power management unit (PMU) with a boost converter for solar and TEG energy harvesting and a single-inductor, multiple-output (SIMO) DC-DC converter for high end-to-end self-powered efficiency. An asymmetric radio leverages ULP ultra-wideband (UWB) transmission and an always-on ULP receiver to reduce RF power significantly relative to prior SoCs for communication at higher data rates in an energy harvesting platform. The sensing interface includes a 4-channel (2W/channel) AFE[17] and SPI with variable voltage output pads (0.4-3.3V) for commercial sensor compatibility. The OpenMSP430 (OMSP) processor[18] and a suite of accelerators

can execute numerous biomedical and environmental signal processing algorithms (e.g. filtering, peak detection, histograms) combining ASIC energy efficiency and flexibility. A lightweight control unit (LCU) can manage chip data and node control while the OMSP is on, and uses a custom ISA and interrupt-driven programs to reduce the program size. The chips flexible clocking unit, containing a programmable ADPLL and configurable system clock, can drive the system clock from a low-power crystal. The digital blocks run in sub-threshold on a 0.5V supply from the PMU, while the radios use both the 1.2V and 0.5V rails.

The chip uses two independent buses controlled by either the OMSP or LCU (for bus 1) or by the two-channel DMA (bus 2). The LCU can configure the OMSP as the main controller or as a bus peripheral that is used only for ALU or background operations. Since most data transfer occurs between the peripherals and the on-chip memories, the two-channel DMA (configured by OMSP or LCU) allows data movement on bus 2 in parallel with chip control on bus 1. Peripheral block wrappers decode and route bus data and manage independent block reset, clock gating, power-gating, and power mode settings. Most peripherals contain three power domains: always-on configuration registers, a bus decoder domain, and a block core logic domain, see Fig. 2.7. Programmable peripherals include a 4-channel, programmable FIR filter, a CORDIC, 16-point complex FFT/IFFT, two timer modules with capture/compare, a multiplier, and heart rate (R-R) AFIB detection. This suite of hardware accelerators supports flexible processing for a wide range of applications. The SoC has a 4 kB data memory and three, 2kB memories: Tx buffer, LCU instruction memory, and OMSP instruction memory. All SRAMs use an 8T sub-threshold bitcell and a read before write scheme. The memories are partitioned into 1 KB, independently power gate-able banks and operate down to 0.35V.

2.2.1 Measured Results

The chip was tested end-to-end for motion capture with an ADXL362Z accelerometer (over SPI). Fig.2.2 shows the SoC datapath tested. To date, this work has the highest level of integration, including energy harvesting and a full transceiver, for the lowest power. This work also has the highest energy harvesting / regulation efficiency, and achieves lower RF connectivity power by 38X. The carefully integrated ULP components on this SoC support numerous IoT applications on a self-powered platform. The chip micrograph is shown in Fig.2.3, indicating the high level of integration between system modules.

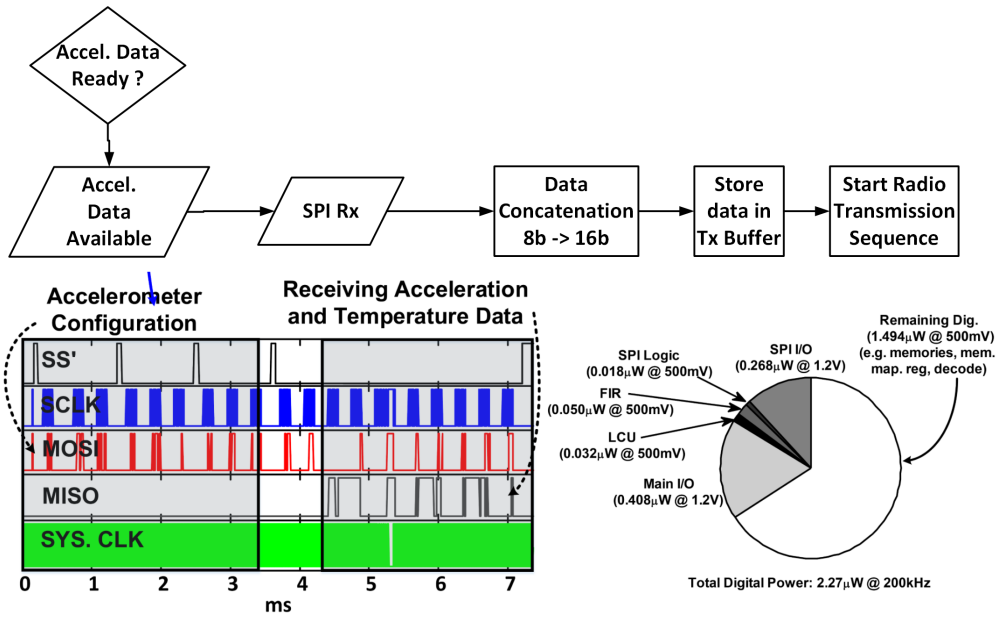

Figure 2.2: Measured Results for the SoC IoT using an accelerometer to sense Position and temperature, and start data transmission.

2.3 Integration Methodology

2.3.1 Literature review for Sensor Nodes architectures

Typically, the digital section of platforms alike, contains one or several micro controllers (drivers) as General Purpose Processor[17], openMSP430[8][19], ARM Cortex M0[20] or Custom Micro controllers[17][8], that allow for processing flexibility and control over the whole chip. This section also provides the glue logic that holds everything together. In order to achieve savings of up to 6800X in energy consumption, accelerators are connected through a bus as slave blocks to perform Digital Signal Processing (DSP), mixed signal blocks control and status monitoring, and memory interfaces. There can be one or several buses, driven by one or several drivers. Figs.2.4[a-d] show generalized block diagrams from the most recent Sensor Nodes found in literature.

Each new iteration of a Sensor Node of this nature with minor new features, or changes in the architecture, is costly in time and resources. Replacing one of the micro controllers would require the re-writing of at least 32% of the total HDL description files. Getting rid of one of the busses or changing the decodification scheme from local to global would have the same impact in time and resources. Even if the accelerators are re-used, every single one of them must be connected to a new bus, requiring changes at each block level that

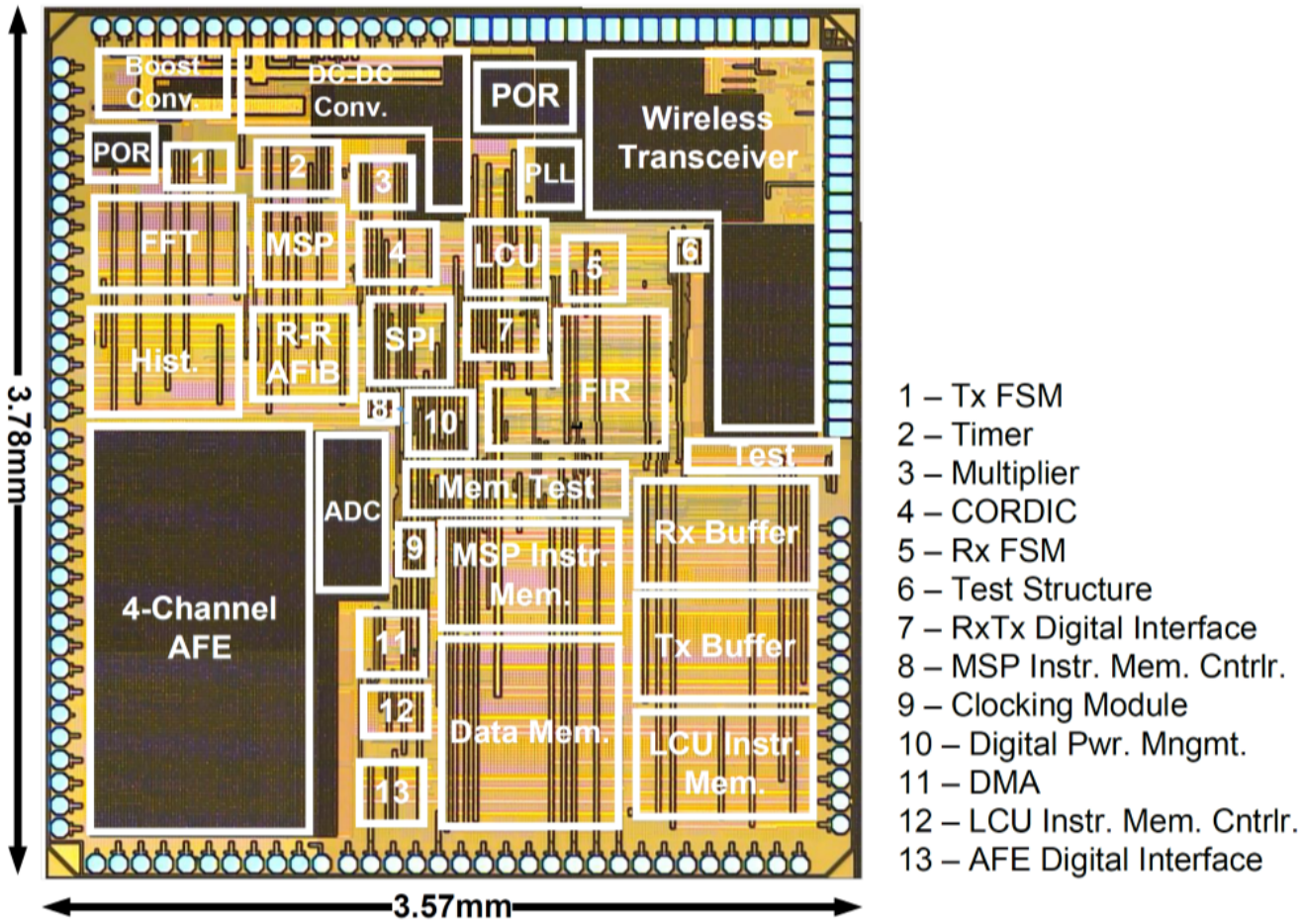

Figure 2.3: Chip Micropgraph for the Self-powered IoT SoC

must be re-verified, so a set of new testbenches and stimulus are required. Using a re-use methodology can increment productivity by up to 340%[9] in design and verification. However, regular strategies hardwire the RTL signals, see Fig.2.5a, in such a way that architecture re-use or evolution is almost impossible. This section addresses the limitations of the current approach and propose a methodology for re-use, re-configurabilty at the behavioral RTL level, and verification-awareness[21] for a sensor node. According with [21], a verification-aware design methodology is based in Formalization, Abstraction and Decomposition. Our methodology follows the three in the following way:

1. Formalization: We use SystemVerilog as a formal language to describe the system that is precise, unambiguous and require just a moderate extra-effort from designers due to its similarities with widely popular HDL description languages.

2. Abstraction: We use SystemVerilog interfaces and modports capabilities to remove the details of the

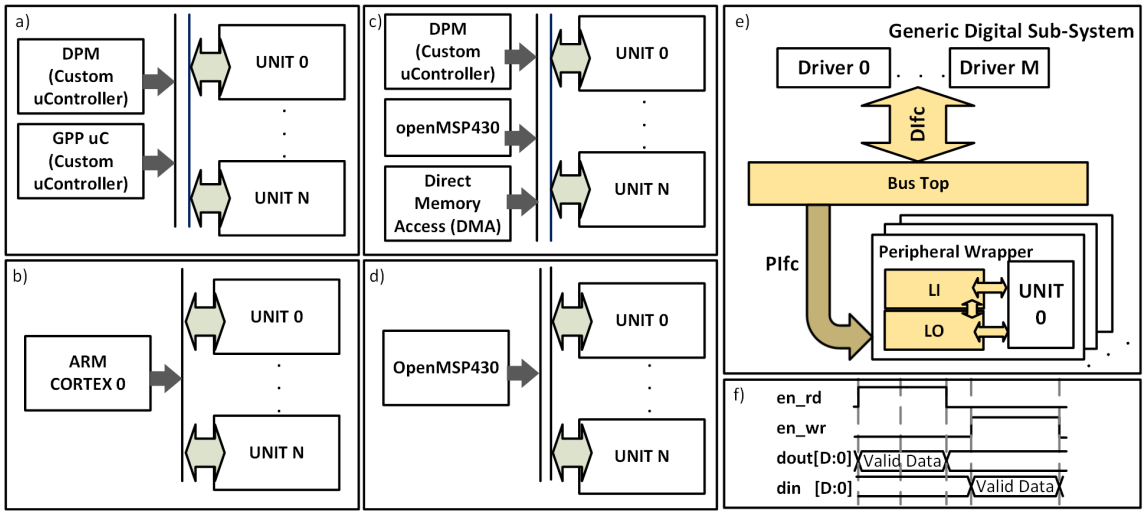

Figure 2.4: BSN Architectures. a) A Batteryless 19uW MICS/ISM-Band Energy Harvesting Body Sensor Node SoC for ExG Applications[17]. b) 21.3A 6.45uW self-powered IoT SoC with integrated energy-harvesting power management and ULP asymmetric radios[8]. c) 18.3A multi-parameter signal-acquisition SoC for connected personal health applications[20]. d) An energy-efficient biomedical signal processing platform[19].e) Block Diagram of the Generic Digital Subsystem using Abstraction and Decomposition proposed. f) Timing Diagram for Local interface assumed to connect the custom units.

interface at the system level. The interface abstraction allow us to integrate the major components of the system without hard wiring across the whole behavioral description detailed signals. An example of system level integration using interfaces is shown in Fig.2.5b.

3. Decomposition: We decompose the BSN architecture into basic components that can be designed and verified independently. More importantly it can be swapped and replaced according with the architecture needs without requiring major modifications on the integrated system.

The behavioral system level description is shown in Fig.2.4e. We conceptually divide the bus architecture in three major blocks: *Bus top* (BT), *Bus Local Generic input* (LI), *Bus Local Generic Output* (LO). To facilitate a homogeneous integration of the units we define a generic *Peripheral Wrapper* (PW) that contains the LI, LO and the unit itself. The basic blocks are connected to each other by two SystemVerilog interfaces: *Generic Driver Interface* (DIfc) and *Generic Peripheral Interface* (PIfc). We rely on the individual basic components re-definition to allocate for changes in the architecture. The less standard part of the generalization is the connection between the Peripheral Wrapper's and the custom UNITS, *Local Interface* (LIfc). For this we assume a write and read protocol as shown in Fig.2.4f which has been the case for our previous developments. Unfortunately, changes in this interface, might require re-definition of the *Peripheral Wrap-*

per. A regular non-reusable implementation of a system with an openMSP430 is shown in Fig.2.5a while our proposed implementation is shown in Fig.2.5b.

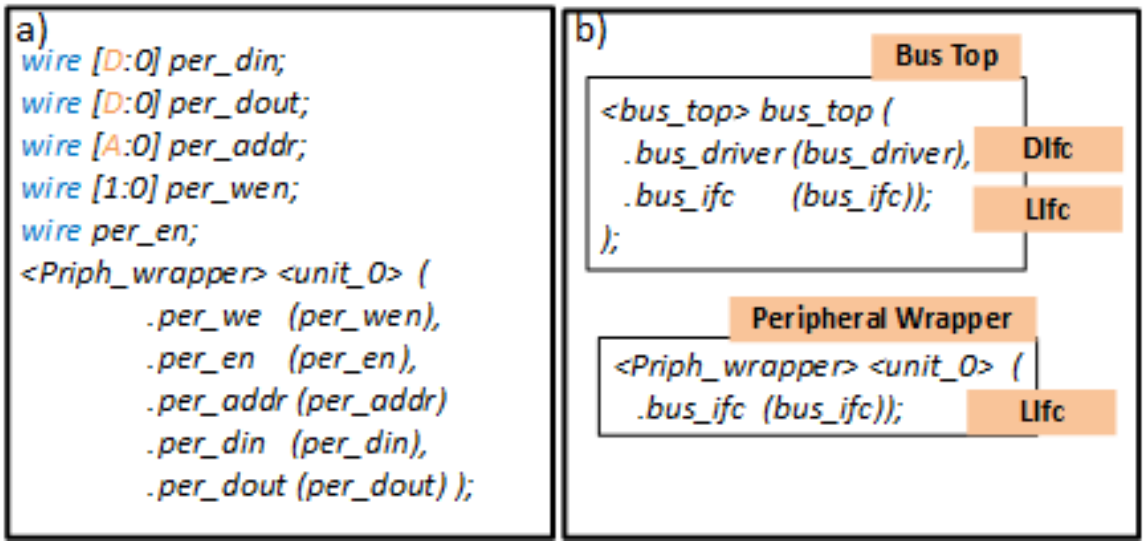

Figure 2.5: Sensor Node integration at the digital top level following: a) Typical integration. b) Re-use and verification aware methodology.

2.3.2 Bus architectures

Typically sensor node architectures use decoders to ensure that data is sent and received by the correct peripheral using a bus address. Global and local decoding are two of the main topologies. Global decoding relies on a single block, to decode the address and generate a unique enable signal used to connect the appropriate block to the bus. On the other hand, the local scheme, relies on each block for decoding the address and access the bus. Finding the right architecture, depends on the target application and the operational conditions [2]. Our methodology, allows us to move between different bus topologies with minimal design and verification extra cost. We re-define and swap the basic block abstractions BT and LI, and the interface PIfc. This is equivalent to only 2% of the total HDL description.

Fig. 2.6[a-b] compares the behavioral simulation results for the global vs local decodification schemes, while reading the I/Os connected to the bus. The DUT is a toy system with 2 units attached to the bus. Each unit has 5 I/Os. The systems has the following characteristics:

1. The DIfc is the same in both cases since we keep the same openMSP430 as a driver.

2. Global Decodification: The *Peripheral Interface* shows two 10 bits enable signals for read and write. One signal per I/O connected to the bus.

3. Global Decodifcation: At the *Peripheral Wrapper*, the unit has 5 I/O to be attached to the bus. In this case the read and write enable signals are 5 bits wide.

4. Local Decodification: The *Bus top* let the driver signals pass unmodified to the peripheral units

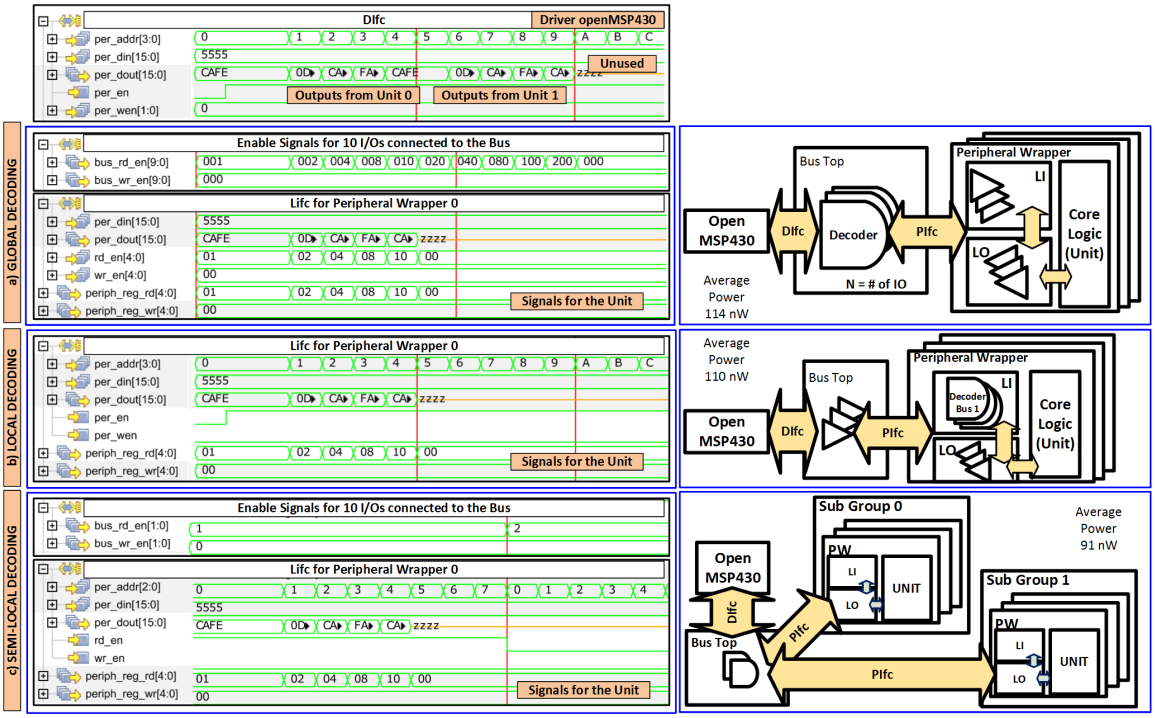

Figure 2.6: On the left simulation results for a 4 bits address bus system set for Global, Local and Semi-Local schemes. On the right, corresponding block diagrams for the implementations.

Fig. 2.6c, shows a more interesting decoding scheme. A semi-local bus architecture, can be implemented to benefit from the lower active energy of the global decodification scheme[2] while reducing its routing complexity. The semi-local approach divides the peripheral space in sub-groups. An enable signal for each sub-group is generated in the *Bus top* out of the most significant bits in the address bus, and routed only to the peripherals that belong to the sub-group. The remaining bits are routed to each *Peripheral Wrapper* to perform local decodification. Fig.2.6c, shows the signals for this approach. This system characteristics are:

1. The most significant bit of the address is used to generate the read and write enable routed to peripherals belonging only to the same sub-group ;

2. The remaining bits [2:0] are passed to the local decodification.;

At rigth, Fig.2.6 shows the block diagram for each implementation and the average power consumption for the three approaches. The example code shown in Fig. 2.5b is valid for any of the three bus architectures, local, global or semi-local.

2.3.3 BSN Example

We independently designed and verified the parameterizable *Generic Local Input* as a unit that can be re-used across all the peripheral units. The *bus top* was redefenided as a buffering stage for the signals coming from the drivers. All the connections between blocks at the system level and peripheral level uses the Difc, PIfc and the LIfc specified in Fig.2.6b which allow us to re-use the architecture for global, local or semi-local approaches by swapping the interface and basic block abstraction definition. Removing one of the busses would require changes limited to the *Bus Local Generic Output/Input* module which represents only 1% of all the verilog files. This version of the chip, follows a local decoding scheme as shown in Fig.2.7.

2.4 Limitations and Future Work

Re-use by abstraction and interface based design has a great potential in reducing costs of design and verification. However, this can be jeopardized by the extra time (up to 2.5X) it takes to make re-usable components. From[21], this extra time is attirbuted to:

1. The effort required to do an extensive analysis of the block and its potential application domains.
2. A more robust and complete set of test-benches has to be generated to verify the functionality of all the possible IP features.
3. The preparation of good documentation is vital to enable the re-usability of this components.

2.4.1 Specific Limitations for BSN

- For our BSN we included a Configuration Register block which is paramaterized and re-used across the whole architecture. Nevertheless it is hard to justify its use in other scenarios, and removing it

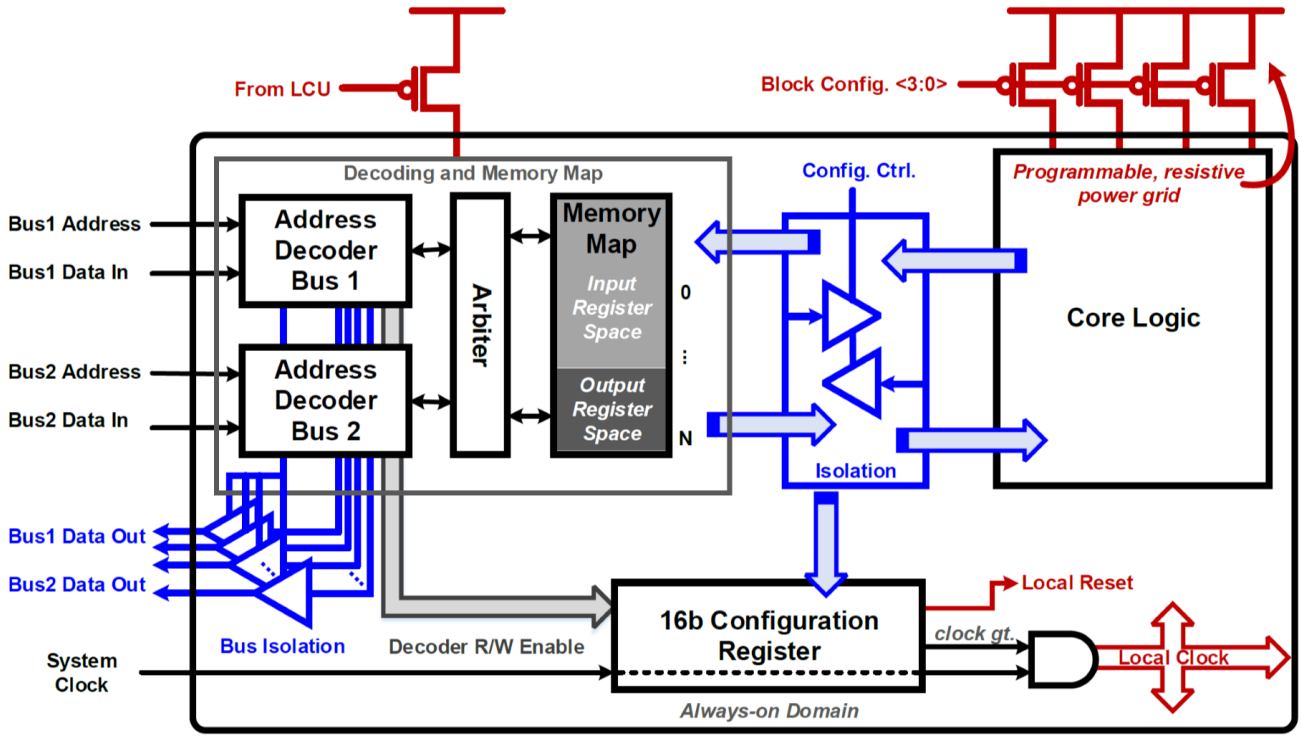

Figure 2.7: Block Diagram of the Peripheral Wrapper, Since the bus follows a local decodification scheme, the bus top is a module with buffering stages.

might cost a full re-definition of the Peripheral Wrappers, that beats the purpose of the abstractions. To fix this, we could have included the Register in the *Bus Local Generic Output*, so it could be easily discarded from a new architecture by simple re-definition of this block.

- The LIfc that connects with the custom UNIT is still an open question. Units that do not agree with the protocol specified require extra-work to adapt to the hard wire LIfc in the wrapper. This hinders in generality desired in the *Peripheral Wrapper*.

- Although, dropping in a new microprocessor can be easily accomplished by replacing the DIfc and basic block abstractions at the behavioral level; the stimulus generation associated with the binary code generation from Assembly or C code is more challenging to re-purpose. Exploration for a different abstraction at this level might be required to accomplish fully re-usability in this realm.

2.5 List of Contributions

Two revisions of this SoC were taped out. I partially participated on the first revision by collaborating in methodology, file directory structure organization and a CORDIC accelerator.

For the second revision I designed a unique re-usable methdology for Sensor Nodes system architectures that allows rapid prototyping and early integration. Below the list of specific contributions:

- I replaced hard-wired RTL interfaces for mixed signal blocks as memories, AFE, radios and power management by SystemVerilog interfaces. By abstracting the interfaces I started earlier system level integration without being affected by later modifications on these blocks. By defining the interfaces once and only once and let it be propagated through the hierarchy, we removed the potential for typing errors and disconnected nodes. Moreover, changes on this interfaces did not affect the integrated system since those changes were localized to just the SystemVerilog interface and the digital module that handles the communication with the mixed signal block.

- By using the methodology for bus abstraction and interfaces I generated 5 re-usable verilog components. A bus top, an input decoder, an output decoder and a configuration register. These modules are totally parameterized and were fully verified at the block level. The bus structure is connected using SystemVerilog interfaces. This changes helped to speed up the integration process. We moved to generate and verify a new decoder structure for each block attached to the bus to instantiate a fully verified object. These changes, gave an homogeneous structure to the entire bus, that helped the debugging process. As a verifier, I no longer needed to be familiar with the intricacies of each block to run system level tests. But make sure I can access the inputs and outputs of the blocks in a systematic way. Furthermore, this is the first architecture, designed for truly re-usability at the system level. The interfaces and block units can be replaced to remove a bus, add a bus, change the driver, move to global or semiglobal decodification, expand the accelerators memory space and expand or reduce the processing data-path with minimum impact in the RTL at the wrapper level, or system level.

- I created behavioral models for the mixed system blocks for verification purposes.

- Thanks to the changes described before, I created an early prototype of the the system level and a system level end to end test for the digital system. This allowed us to incrementally integrate new

blocks making sure that we always had a working copy in th repository. How many times was the test broken after submission!

- I designed a unified memory interface for the 2 instruction memories, data memory and Tx and Rx buffers. This helped to speed up design process since we passed from create and verify 5 different verilog modules to 1 single module. Changes in the memory interface, were immediatly propagated to the 5 memory controllers instead of having to modify one by one.
- Since the system does not have any hand-shaking meachanism, the drivers relay on the timer to check task completion. I re-designed the timer to allow for completelly independent operation.
- I designed an ultra low power CORDIC accelerator able to calculate up to 7 mathematical operations.
- I Integrated the OpenMSP430 with emphasis in the UART unit.
- I Co-Implemented the testport interface, which is the debug and program port for the whole chip.
- I created a verification environment for the SoC used during pre-silicon verification and post-silicon validation. The verification environment is able to generate stimulus for all the digital IO in the SoC pad ring or write LCU or openMSP430 code directly in the memory models used for verification. This environment, integrates an LCU compiler; which was helpful for system level verification due to the simplicity of the LCU ISA. The code is translated to the dual phase test port protocol in a transparent way which is usefull for post silicion validation. This environment is easy to use for pre-silicon, particularly helpful for new programmers. For post-silicon validation generates stimulus for both, a commerical Pattern Generator or is especially optimized for the use of an FPGA as a pattern generator, since the stimulus generation is based in a small state machine that simply reads from a memory
- I integrated at the RTL behavioral level all the blocks to the system.
- Planned, created and run system and block level tests

2.6 Conclusion

We presented the methodology we used to integrate our ultralow power highly flexible SoC in order to achieve re-usability and facilitate evolution of the architecture. A re-use and verification aware methodology can help to accelerate and reduce the costs of design and verification in the digital section of Sensor Nodes. For this we create a generic platform in two steps. First, we replace the regular HDL ports with system verilog interfaces (abstraction). Second, we identify typical building blocks (decomposition) that can be developed and verified independently to be instantiated across the whole architecture as generic blocks. The modularity of our approach enables early system level integration, even before individual blocks completion, allowing incremental architecture evolution. The re-usability of the abstractions and the blocks decomposition, enable us to migrate evolve or modify the system architecture by modifying only %2-3 of the files vs an optimistic %32 if following a regular methodology. We also eliminate the need of new test bench or stimulus generation when migrating between architectures. Following this methodology we can migrate from local, global and semi-local decodification or change the architecture basic parameters as bus address size or data size by swapping the basic building blocks definitions.

Chapter 3

Application Oriented, Architecture Agnostic, Verification Environment for Body Sensor Nodes, AVEBoS

3.1 Introduction

Fig. 3.1, depicts how Sensor Nodes architectures have evolved from sampling and transmitting raw data using General Purpose Processors (GPPs)[22] to highly integrated systems able to do relatively complex computation under the limited power budget of 19uW[17]. The future of the BSNs requires even more sophisticated energy efficient on node algorithms able to not only do signal based detection, but to predict events or even make decisions based on data analysis. The increasing complexity of these sensors is also reflected in an increment in Design and Verification Costs.

The Development of a typical SoC follows three basic stages. Idea generation, Definition and finally Implementation[23]. The Idea generation usually starts with an application or a set of applications to be supported. i.e. ECG, EMG, Pulseoximetry or Asthma monitoring. During the definition stage, a high level model of the algorithms is developed and evaluated. In current methodologies, algorithm modeling efforts, and design and verification tasks are disconnected. In fact, these models are often thrown away after exploration and experiments are completed. Furhtermore, they are often duplicated, once the models for the

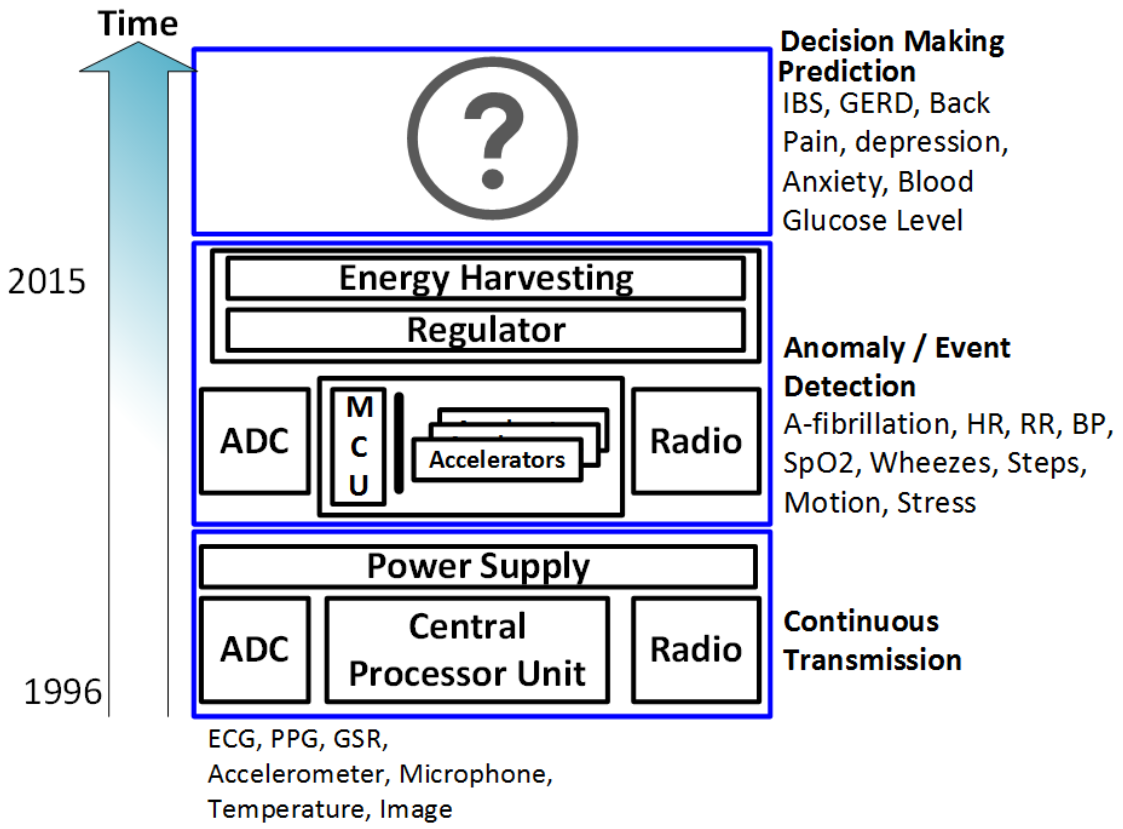

Figure 3.1: Evolution of Wireless Body Sensor Nodes driven by energy consumption optimization.

verification environment building starts.

Motivated by the effort duplication, and time and resources wasted, we designed AVEBoS, an Application aware Verification Environment for Body Sensors (AVEBoS) that enable us to re-use the codes built for algorithm development and exploration during the hardware development phase at different levels of the design hierarchy. Since AVEBoS is architecture agnostic, it enables the designer to port algorithms to different architectures and design without re-writting the models or the verification environment. AVEBoS provides the architect, designer and verifier with a communication path that links the Model, the Testbench, and, the Design Under Test (DUT). This new communication channel allows us to evaluate the hardware platform performance under the particular application characteristics. The main contributions of AVEBoS are:

1. Enables the Architect, Designer or Algorithm Expert to develop the model for initial exploration and experiments as a hardware and verification aware model. This allows to re-use the otherwise discarded

models across several stages of the hardware design. The process of building these models has to be transparent for the Algorithm depveloper so it does not represent an extra burden for algorithm exploration. On the other hand, the models have to be leveraged with minimum extra effort for verification purposes

2. Developing a Model that can be re-used as an application oriented verification tool at the block level, system level and across sensor nodes architectures.

3. Enables the Architect to evaulate the feasibility of using a hardware architecture or a block for a particular application.

3.2 AVEBoS as an Architecture Agnostic Verification Environment

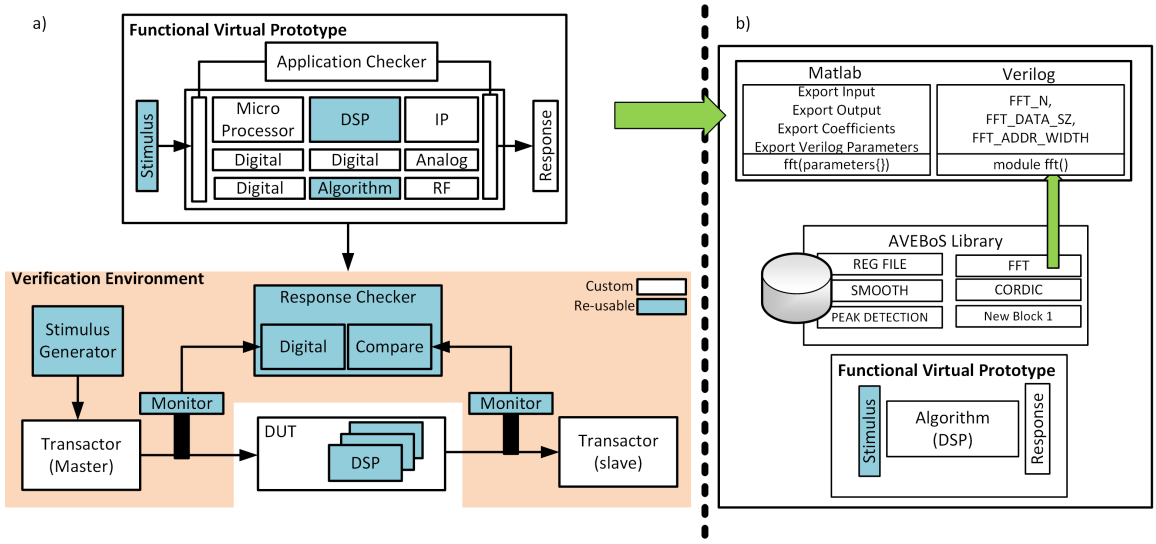

Figure 3.2: Verification Environment a) Typical Verification Environment block Diagram[23]. b) AVEBoS Functional Virtual Prototype.

A generic Verification Environment (VE) is shown in Fig.3.2a, and contains the following items:

- **Functional Virtual Prototype (FVP)**: A golden functional representation of the complete design and its testbench. A FVP unifies the use of system models for software, architectural analysis, and functional verification

- **Stimulus Generator**: Creates the data that the testbench uses to stimulate the design.

- **Transactors**: Change the level of abstraction in a testbench. The most common use is to translate from implementation-level signaling to a higher level transaction or the reverse.Transactors can behave as master, initiating activity with the design, as slaves responding to requests generated by the design, or as both a master and a slave.
- **Monitors**: Check the correct signaling and protocol of data transfer across design interfaces. In addition to passively monitoring the data transfers across interfaces, these monitors can encapsulate the data to be communicated to response checkers. The interface monitors should be application independent and written in a manner that allows their easy re-use.
- **Response Checkers**: The most application specific information in the testbench. The most efficient method for implementing the response checkers is to reuse the FVP as a reference model in the response checker.

To guarantee re-usability across Sensor Node architectures and portability we focus our approach in the components that are architecture agnostic, marked with color in Fig3.2. This components are basic building blocks since they don't change across architectures, or design hierarchy, therefore can be easily re-used.

3.3 AVEBoS Structure

AVEBoS is built using Perl, MATLAB and bash. MATLAB handles the signal representation, DSP, and, algorithm modeling. Perl creates a wrapper aroung MATLAB, the Testbench and the RTL behavioral simulation tool, to present an unified interface to the engineer.

The input for AVEBoS is a simple text file, that defines the verification environment. Fig. 3.3, shows extract examples to define: the input signal characteristics (a), the width for the processing data-path (b), parameters applied to only particular pieces of the algorithm (c)(d), The signals in the RTL to be monitored mapped to a reference model (e) and finally user defined post-processing scripts (f) if required.

Since the transactor is totally architecture dependant, it has to be created by the user. On the other hand, AVEBoS generates: monitors, stimulus and response checkers from the MATLAB model, using the parameters specified in the input file.

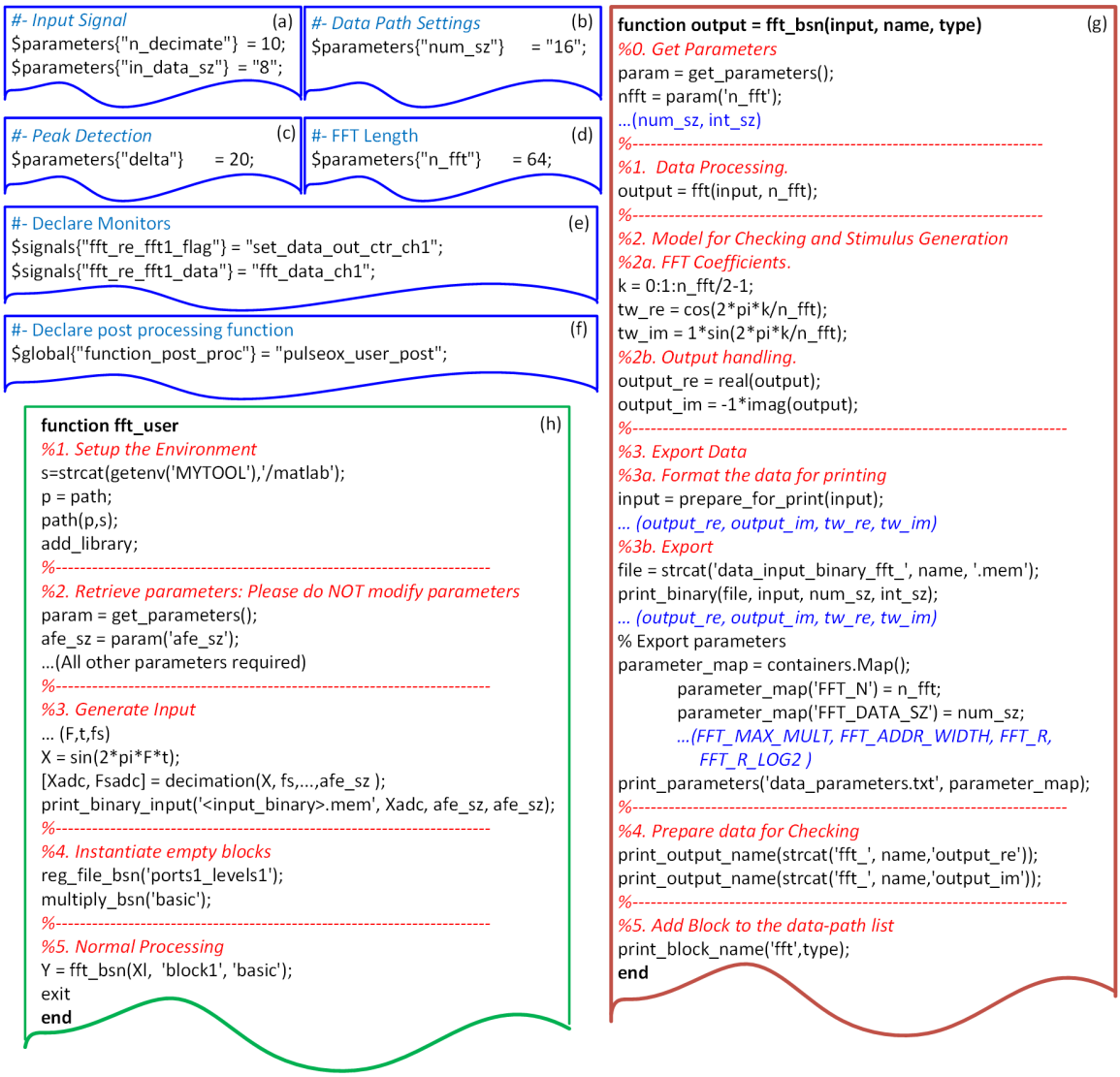

Figure 3.3: AVEBoS User Interface

The Functional Virtual Prototype is the heart of the system and is built in Matlab using the AVEBoS library. Since AVEBoS is application oriented and architecture agnostic, this FVP focuses on algorithms implementations such as ECG heart beat extraction or Afibrilation Detection. The expert develops the algorithm as usual, replacing the built in Matlab functions by functions in the AVEBoS library. This library, is conformed by wrappers for typical Matlab built in functions as the FFT shown in Fig. 3.3g, or custom algorithms developed by the user. Each wrapper can include functionality to export the Inputs, Outputs, Parameters and Coefficients required for RTL simulation.

Each Matlab function in the library is paired with an RTL description of the block. These RTL descriptions

should follow the methodology described in Section 3.4.1 to be fully parameterizable, which enables true re-usability of a block for evaluation or implementation under different constrains and architectures.

Due to the modularity of AVEBoS different RTL implementations of the same block can share the same model. This, gives freedom to the verilog designer for optimization without incurring in incremented verification costs. As a simple example, a synthesized register file can have different rows and word sizes which are easily set using verilog parameters; on the other hand, it can be optimized for low power consumption with per row clock gating, per bank clock gating and more advanced techniques as granular power gating. This optimization require a change in the verilog description, but no change in the high level modeling of the block or in the verification protocols. For this cases, all the three different optimizations would share the same model. Also, new models can be easily included to the library for later designs by following the generic stucture shown in 3.3g.

3.4 Verification at the Block Level

Reusing design blocks is prevalent today in large complex designs. It seems easier to reuse or modify a block from a previous design than to design a block from scratch. However, true re-usability of a block is hindered by:

1. The lack of abstraction of an IP that requires full modification of the behavioral description to adapt it to the new design.
2. The extra effort required for verification environment development and full verification of the adpated IP

3.4.1 Abstraction

To address the lack of abstraction we propose to write RTL block descriptions where basic processing, control logic, register banks, and basic arithmetic blocks are de-atached from each other. Fig3.4 shows the block diagram of a generic system that uses the FFT Core. Thanks to the interface abstractions, the basic building blocks of a full FFT solution are implemented in de-attached modules connected by adaptable interfaces. The re-usable FFT core in blue is part of the AVEBoS library. The Control logic is included in the

examples section but depends on the application and the architecture. In this way a fully verified FFT logic with parameters for Input Data Size, and FFT length N, can be instantiated without further modifications in new architectures as shown in Section 3.5 and Chapter **??**.

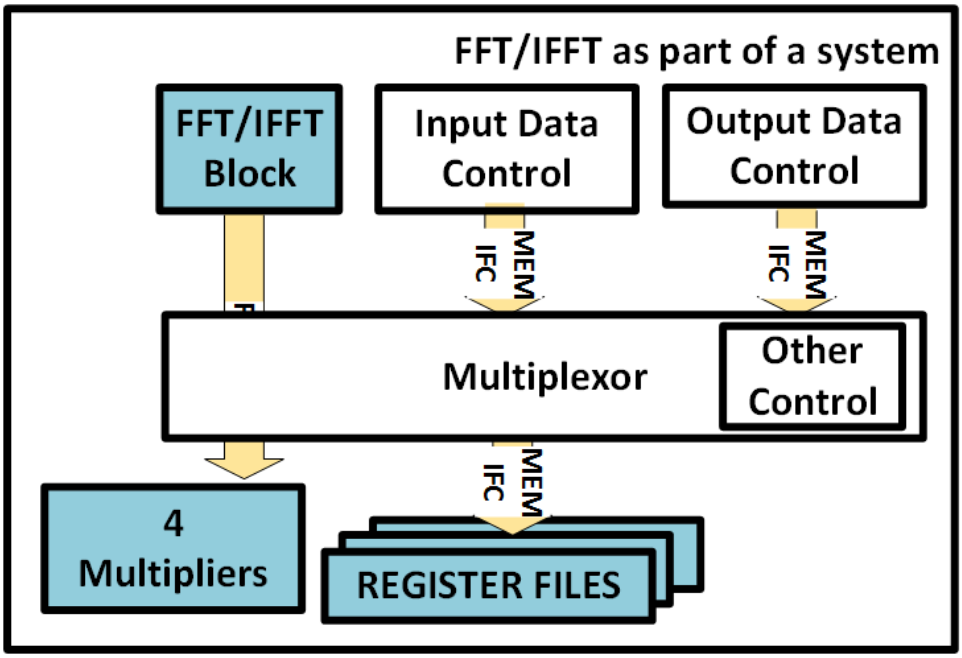

Figure 3.4: Block diagram of an example of the FFT core integrated as part of a system using the interfaces abstraction and the modularity of our approach.

3.4.2 Verification

One of the main requirements for a re-usable IP is a portable verification environment that enable coverage for all the valid combinations. Manual full validation is prone to errors and time consuming. AVEBoS allows the designer to loop through parameters, generating valid stimulus and checking the output. Fig. 3.5b shows the real part of the FFT for a 1 Hz sine wave input for different FFT N lengths. Fig. 3.5c shows the error vs Input data size. Fig. 3.5a shows a block diagram of the verification environment for the FFT parameterizable core.

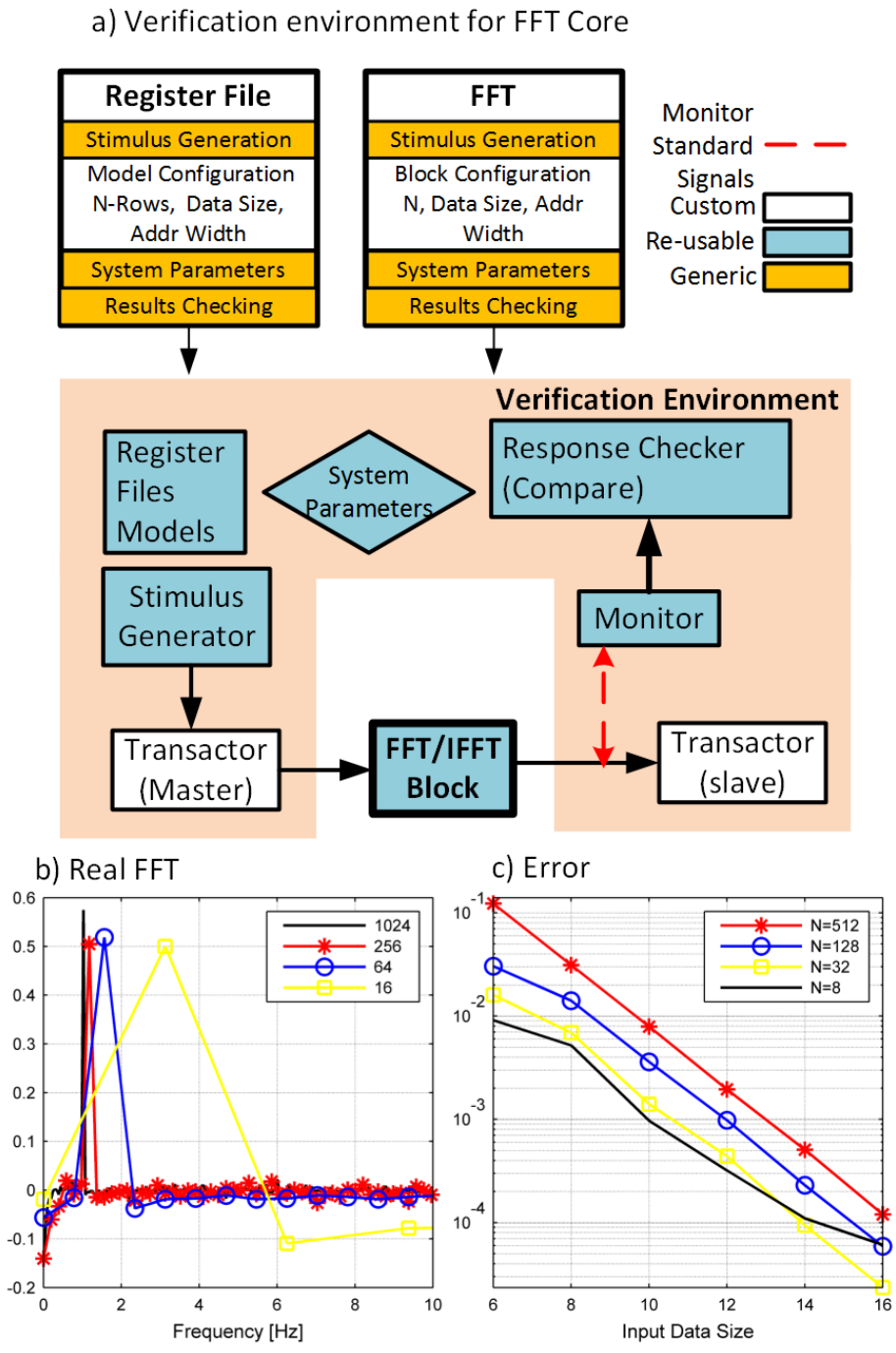

Figure 3.5: Block Level Verification

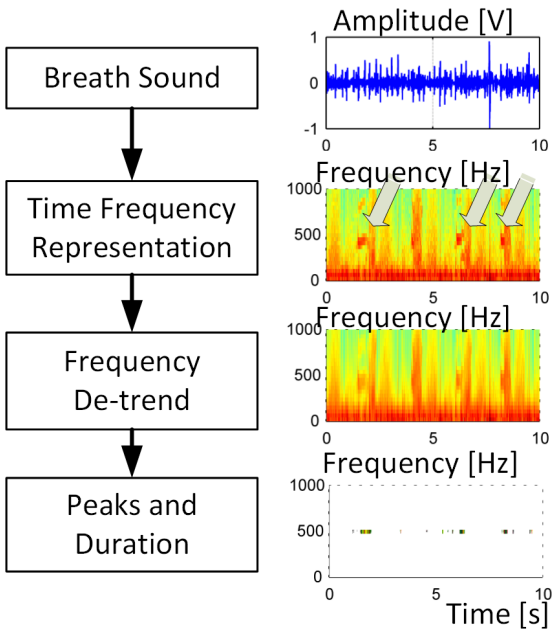

Figure 3.6: Wheezing Algorithm [24]

3.5 AVEBoS as an application oriented verification environment. System Level Example: Wheezing Detection

AVEBoS as a system level VE can leverage the block level models, or new models can also be created as part of a project, or to be added to the AVEBoS library. This Chapter described the Short Time Fourier Transform accelerator designed for Wheezing Detection following the re-use methodology described in Chapter 2.1 and Chapter 3.1.

3.5.1 Wheezing Detection Algorithm

Wheeze monitoring provides information about the extent of a wheezing episode. This information correlates well with conventional indices of asthma activity and is helpful in assessing efficacy of treatment[25]. Wheezes are musical adventitious lung sounds that last between 80ms to 250ms. Their frequency range extends from approximately 100 Hz - 1 kHz.[26]. Wheezes are detected using a microphone that records the breathing sound. The sound is then digitized and processed to detect a wheezing event.

Several algorithms has been proposed for wheezing detection[27][28], however, algorithms based on Short

Time Fourier Transfer (STFT) has shown better performance than others. Standard Fourier analysis allows the decomposition of a signal into individual frequency components, it does not tell us however, when those frequencies occurred[29]. On the other hand the STFT shifts a window across the data and calculates a Fourier transform for each new window position[30]. This enable the observer to identify a window interval for when the frequency occurred. An STFT based algorithm for wheezing detection is showed in Fig.3.6 and is based in [24]. First the Short Time Fourier Transform is calculated using a sliding 256 sample Haming window, with an overlap of 128 samples in the time domain. See Fig.3.6b. Second the trend of the frequency content is found and subtracted from the original to remove the underlying basic breath sound from the total breath sound. Doing so, large amplitude variations between samples are removed and a much smoother signal is produced. Fig3.6c . Lastly, peaks are identified and the following criteria is applied:

- The number of peaks coexisting at each time instance should be not greater than four, taking into account that wheezes usually have no more than three harmonics
- Peaks should have time duration greater than 150 ms

3.5.2 Wheezing detection hardware exploration and verification

The Block diagram for the spectrogram generator is shown in Fig.3.7a. This architecture shares 4 multipliers, (one per FFT butterfly) with three controllers: The FFT core, the Windowing Controller and the PSD controller. We setup the VE environment for the wheezing detection system as shown in Fig.3.7b. Blue components in the FVP are completely re-used at the system level from block level. The same applies for the RTL description. Fully validated RTL descriptions are re-used unaltered from the AVEBoS hardware library.

For the Spectrogram, we re-use the FFT hardware description as it is. However, based on the FFT original model, we create a new model that can be added to the AVEBoS library. This model generates the checkers for the windowing function, FFT and PSD required to generate an spectrogram. It enables three monitors that we use to check the interface at the windowing controller, the FFT and the PSD. Fig 3.8 shows the results for the spectrogram for different configurations of the hardware.

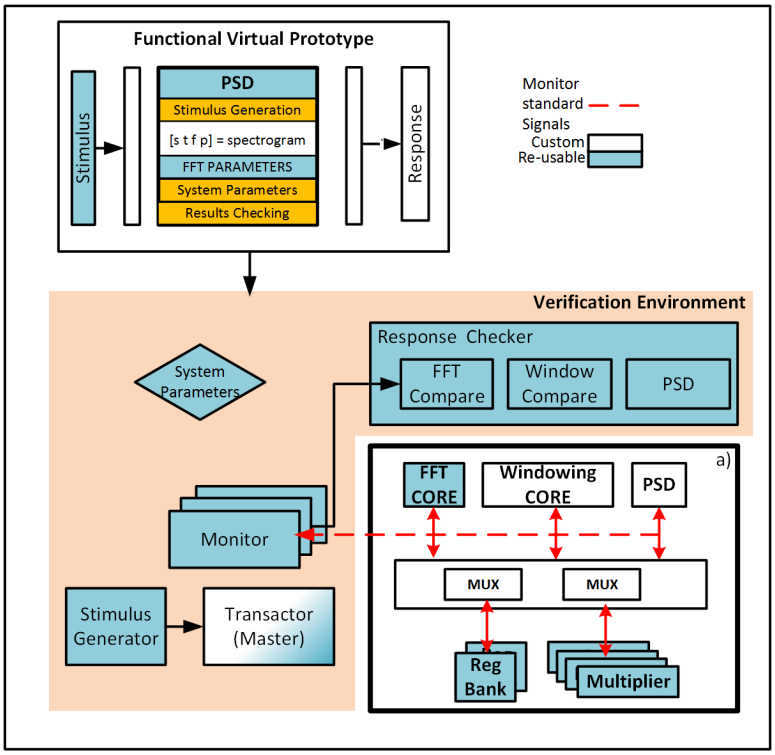

Figure 3.7: Spectrogram accelerator VE for wheezing detection.

3.5.3 AVEBoS vs commercial tools

Verification Languages

There are several verification languages as Specman (e), Open Vera and Proprietary Specification Language (PSL). These languages and the tool set that goes with it, are intended to facilitate writing test benches, stimulus, automatic checkers and check coverage. The development of a verification environment using these tools has three disadvantages when rapid prototyping is required:

- It implies the existence of a verification team that starts to develop the environment as soon as the architecture is drafted
- It is disconnected and discards the algorithm development, which models the expected results of the system for a given input.
- It will not be available until later during the development process. Usually when the verification environment is ready, the RTL is also well advanced.

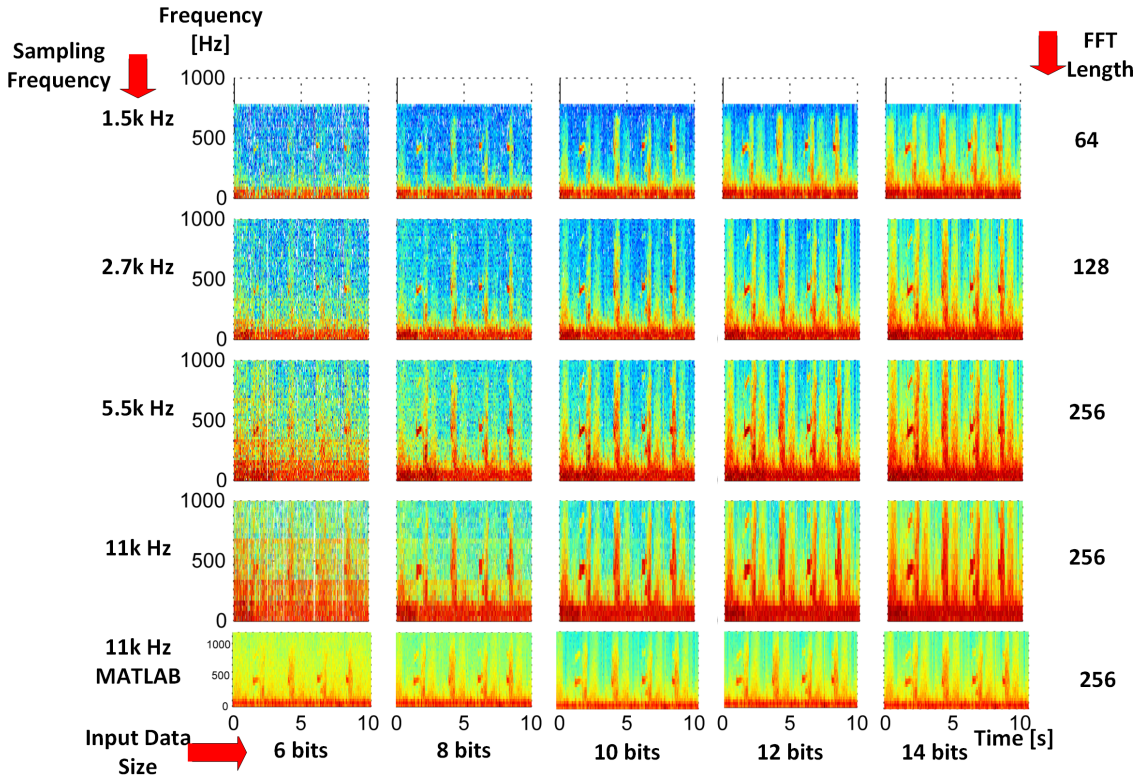

Figure 3.8: Spectrogram Results from AVEBoS varying the Sampling Frequency, the FFT length N and the input data size.

- It is specific for a particular architecture and limits the ability for re-use across design hierarchies and architectures

Although, AVEBoS does not intend to replace functional pre-silicon verification, this tool:

- Was designed to quickly evaluate whether a Sensor Node SoC supports a particular application requirements during pre-silicon verification at any stage of the development: Architecture definition, verification, validation. (Application Oriented).
- Generates Expected output, monitors and Functional Models that are Architecture agnostic. Meaning that the verification infrastructure can be re-used across block level, system level and different architectures with no effort for portability.
- The sample HDL library was designed following the re-use methodology: Abstraction and interface based design enabling the easy portability of the blocks.

3.5.4 Contributions

The MATLAB codes used as models for the wheezing detection accelerator, and the audio records used for testing were provided by Saba Semrani[31] (semrani@ncsu.edu) a graduate student at North Carolina State University. The Perl Wrapper that enables the communication between MATLAB, the HDL simulator and the MATLAB API was developed by Qing Qin (qq3za@virginia.edu) an under graduate student at UVA. I designed AVEBoS architecture, wrote the parameterized verilog, set up the directory structure, and collaborated in Perl, MATLAB and shell code development, testing and application validation.

Chapter 4

Pulse Oximetry

4.1 Introduction

We present a synthesizable IP block for digital signal processing of Photoplethysmogram (PPG) signals. This block combines Fast Fourier Transform (FFT) and Inverse FFT (IFFT) to create an ideal filter in the frequency domain which addresses the necessity of very sharp cut off frequencies for noise removal. We analyze the effect of varying architecture knobs such as Input Data Width (IDW), FFT length N and Sampling Rate T_s on the application performance, average power consumption and memory requirements. Applications that rely on the time of occurrence, i.e. the peak positions, are mostly affected by the sampling rate. On the other hand applications that depend on the feature value itself, i.e. the peak value, are affected by the number of bits used, IDW, to represent the PPG signal. To address different application needs we create a fuly parameterizable behavioral description of the system. To accelerate the design process, we use Standard Cell based Memory (SCM), which typically results on an increment in area and power consumption. To counteract this effect we combine architectural modifications in the memory banks and memory controllers to reduce the area to half while keeping the same average power performance. To showcase the IP block we use a Global Foundries 65 nm library optimized for near threshold operation for synthesys. Simulation results show an average power consumption of $6.2\mu W$ at 166KHz.

4.2 Pulse Oximetry

Pulse oximetry is one of the most popular techniques for physiological data acquisition. A typical PPG signal, Fig. 4.1a, contains a notable AC component caused by the heartbeat; it also shows a slower wandering movement due to the respiratory effect, a DC component originated by changes in the skin, and motion artifacts. Fig. 4.1b shows the Spectral Density (PSD) of the PPG signal, where two peaks can be easily identified. One peak located at the 0.1 Hz vicinity, corresponds to the Respiratory Rate (RR), and another one around 1-2 Hz vicinity, that corresponds to the Heart Rate (HR). A bandpass filter is typically used to extract the components under analysis. However, the main challenge is to obtain a very sharp cutoff filter able to separate the AC component from the DC component; and more challenging the HR from the RR while still being able to operate under an ultra low power regime. In this chapter we address the necessity for a very sharp filter by implementing a programmable FFT based filter, explained in section 4.3

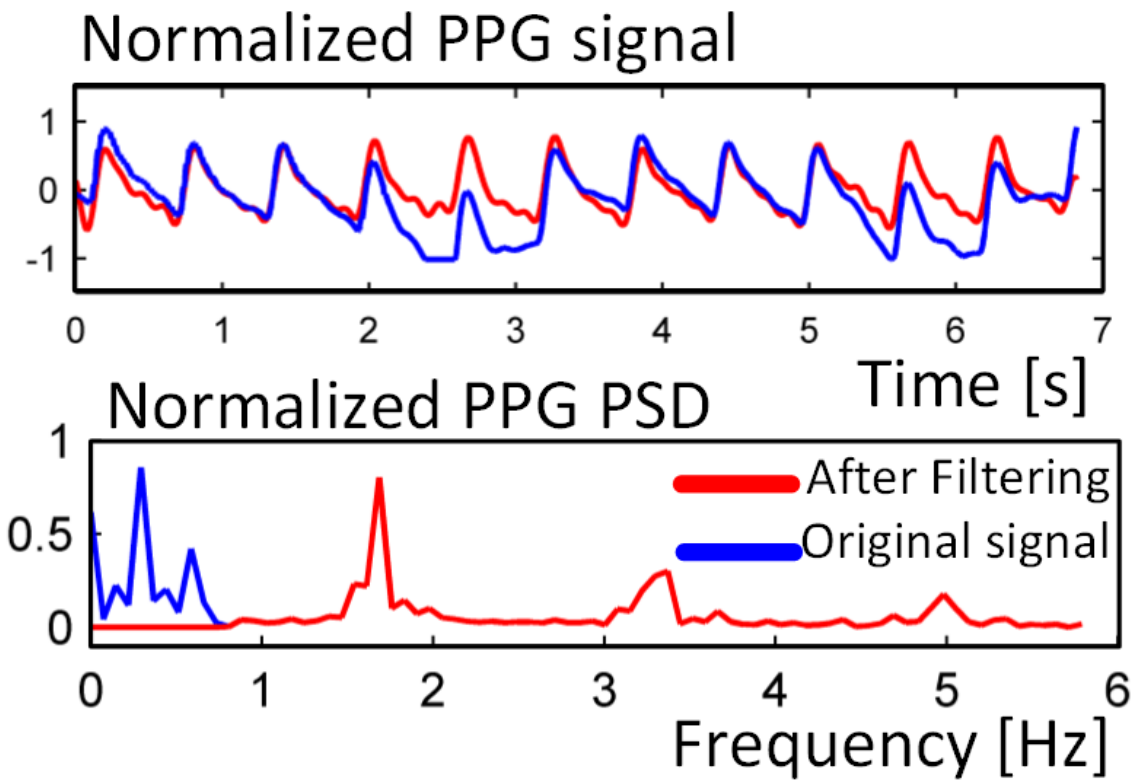

Figure 4.1: PPG Signal. a) Typical raw PPG signal before and after filtering. b) PPG signal spectrum before and after filtering

The signal processing requirements for a PPG signal changes according with the operation conditions. i.e. It is of the best interest of a fitness monitoring device to maximize the use of available energy. On the other hand, a new-born monitoring system relaxes the constrain in energy availability and emphasizes the need for improved HR resolution. A system able to meet the high HR resolution might not fit in the ultra low power regime and vice versa. This, added to the cost of design and verification of nowadays systems, motivates the re-use methodology we describe in Section 4.4.2. The rest of Chapter 4 is dedicated to explore the effect of architecture knobs on the application performance. One of the main drawbacks of FFT based algorithms is the high memory utilization. Section 4.5 describes the post-synthesis simulation results of the IP block and the optimizations proposed for the SCM to reduce the area occupied by the memory and the power consumed by idle portions of it.

4.3 Feature Extraction Algorithm

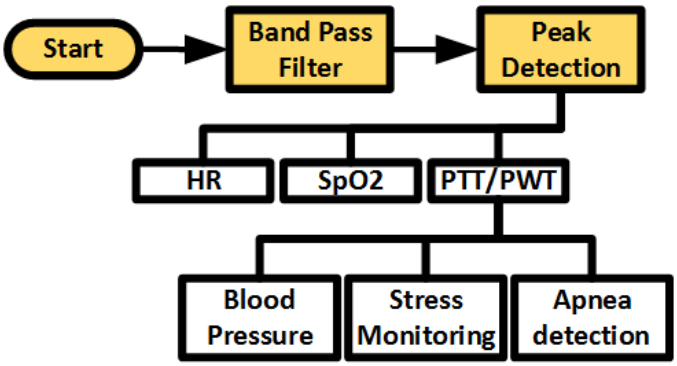

Figure 4.2: PPG Processing algorithm

Fig. 4.2 shows a typical PPG processing data flow. Bandpass filtering helps to remove the baseline wandering, noise and motion artifacts. This, is commonly implemented using Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) filters. Nakajima [32] used a 1024 taps FIR filter to separate the RR from the HR information. The FIR filter has low noise suppression performance while IIR filters might cause system instability. A third filtering method based on the Discrete Cosine Transform (DCT) removes unwanted frequency ranges of the signal by blocking specific DCT index in the DCT domain. The attractiveness of this method, Index Block Discrete Cosine Filter Method (IB-DCTFM) [33], resides on the filter ideal frequency response, which allows for precise selectivity in the frequency content. For this work, we

use an Index Block FFT Filter Method IB-FFTFM instead of an IB-DCTFM. A DCT based approach, has one main advantage over the FFT in this case. Due to the real response of the DCT vs. the complex response of the FFT, the memory required by a DCT is only half of the memory required by a FFT. However, direct mapping of the DCT to a hardware accelerator results in irregular implementations difficult to scale to higher order DCTs which compromisses the re-usability of the Block as IP. Besides, for filtering purposes we are not interested in the analysis of the PPG signal or feature extraction in the frequency domain; but in the ability to discriminate frequency content precisely which can be achieved with the FFT. Finally, wearable sensors often benefit from FFT accelerators[1][8].

Fig. 4.1 (below) shows the PPG spectrum after filtering out the DC and RR components. Fig. 4.1 (above) shows the PPG after filtering whithout wandering or DC component. After filtering, several PPG applications require feature extraction in the time domain. HR is usually calculated by detecting the Peak's positions. RR requires to filter out the HR and noise and valley's positions detection. The SpO2 requires the calculation of the $\frac{AC}{DC}$ ratio. To find the AC we calculate the difference between consecutive Peaks and Valleys. Also, Pulse Transient Time PTT and Pulse Width Time PWT analysis are used for blood pressure[34], stress monitoring[35] or apnea studies[36]. For all this, we include a Peak/Valley detection algorithm following the bandpass filter.

4.4 Architecture

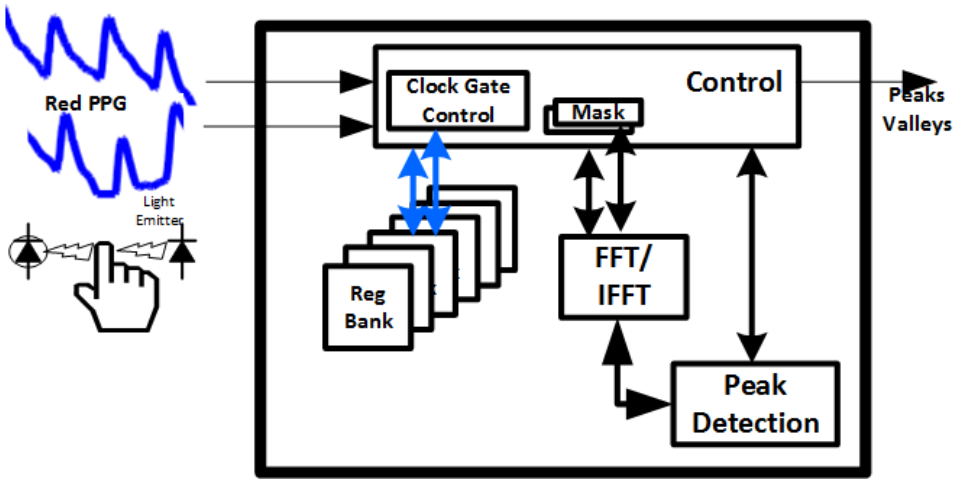

Figure 4.3: DSP accelerator Block diagram for pulse oximetry applications.

Fig. 4.3 shows the block diagram for the PPG digital signal processor. A FFT-IFFT block with length N and DPW as parameters, constitutes the heart of the system. We implement a hardware efficient FFT engine with ”in-place” reduced addressing logic based on the ideas described on [37]. For this application, we include six independent memory banks. Four banks each with $\frac{N}{2}$ rows are used for receiving new data until N samples for Infra-Red (IR) and Red (R) signals have been completed, and FFT calculation. Two extra banks allows easy data transfer for IFFT calculation. A Control block is used to multiplex the FFT-IFFT engine between the two channels. This control, also captures the DC component of the signal for SpO2 calculation and the filtered version of both PPG signals. To support filter programmability, we include a register based mask that identifies the blocking indexes described in Section II. The Filtered version of the signal is sent to a custom Peak/valley Detection accelerator that identifies the Peaks, Peak's Positions, Valleys, and Valleys Positions. The outputs of this IP Block can be the FFT of the PPG, the filtered version of the PPG, or the peak/valley positions.

Fig4.4, shows the data flow through the memory banks for the IP Block with an FFT of length 64. The events are enumerated at the top and described below:

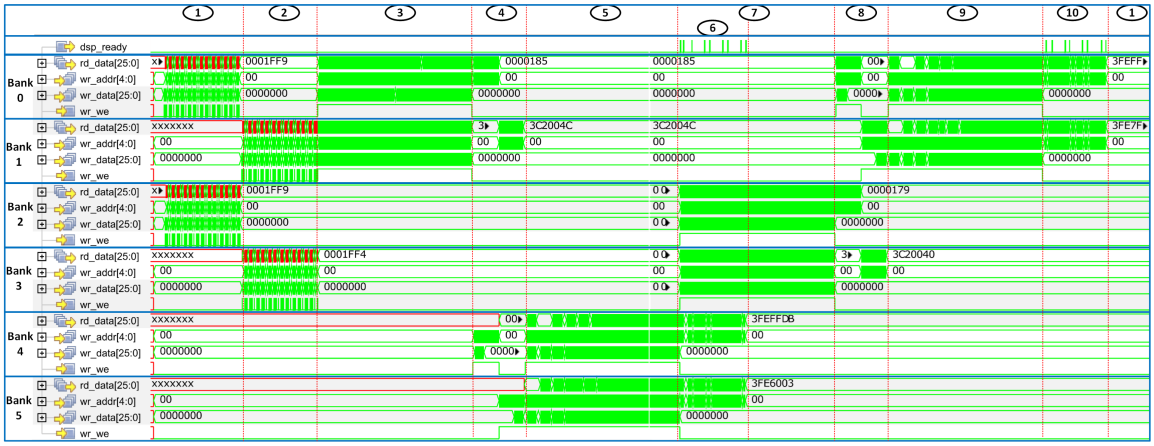

Figure 4.4: Verilog simulation results for the full datapath

1. Receive $\frac{N}{2}$ of R (Bank0) and IR (Bank2) samples and keep it in the bank while the window is completed.

2. Receive $\frac{N}{2}$ of R (Bank1) and IR(Bank3) samples and keep it in the bank while the window is completed.

3. FFT calculation for Channel 1 (Bank0-Bank1).

4. Read the FFT results for Channel 1, block the frequency index to be filtered out and write it in a new memory location. (Bank4-Bank5)
5. IFFT calculation for Channel 1 (Bank4-Bank5)
6. Read the recovered signal after filtering and pass it trhrough peak/valley detection. The *dsp_ready* signal identifies when a new peak or valley has been detected.
7. During the same time interval, start FFT calculation for Channel 2.(Bank2-Bank3)
8. Read the FFT results for Channel 2, block the frequency index to be filtered out and write it in a new memory location. (Bank0-Bank1)
9. IFFT calculation for Channel 2 (Bank0-Bank1)
10. Read the recovered signal after filtering and pass it trhrough peak/valley detection. The *dsp_ready* signal identifies when a new peak or valley has been detected.

4.4.1 Architecture knobs analyzed for application optimization

Using a Matlab model for HR, RR and $\frac{AC}{DC}$ calculation combined with the parameterizable behavioral HDL description of the system, we evaluate the following architecture knobs and its impact on the architecture performance:

- Sampling Rate T_s
- Input Data Width (IDP)
- FFT length N

Sampling Rate

Fig. 4.1b, shows that the PPG spectrum has all the relevant information in the [0-7]Hz frequency range, thus the minimum sampling rate dictated by Nyquist is 14Hz. However the accuracy of the HR, RR or any other application that depends of the peak/valley positions depend on the sampling rate T_s used to digitize the PPG. Each peak position can be detected with a maximum resolution of $\pm\Delta T_s/2$, as shown in Fig. 4.5a.

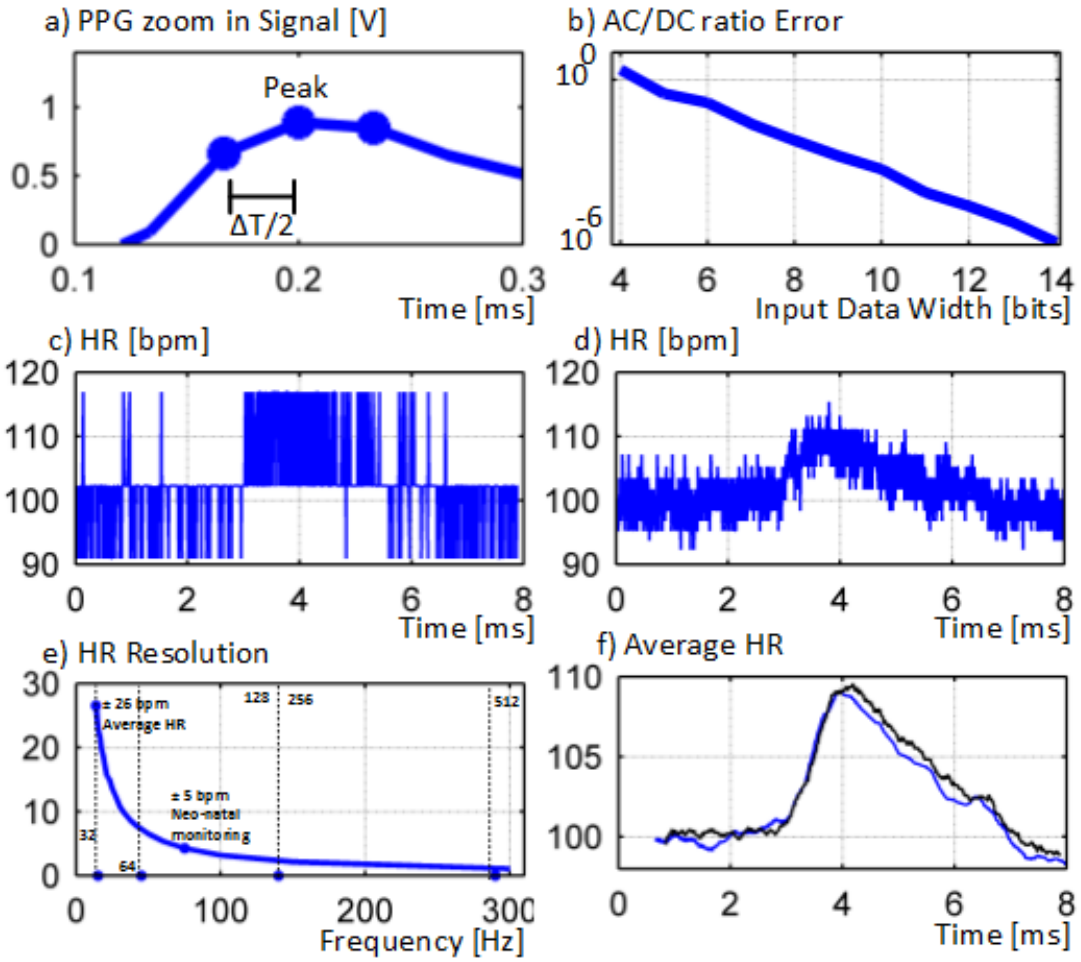

Figure 4.5: a) PPG signal zoom in around the peak showing the HR resolution dependency on the sampling rate. c) Ratio error varying the number of bits used to represent the PPG signal. b) HR calculated with a signal sampled at 14Hz. c) HR calculated with a signal sampled at 300 Hz. d) HR Resolution for different sampling rates. e) Averaged HR resolution for 14Hz and 300 Hz

Thus the HR Resolution is calculated using Eq. 4.1 and is plotted in Fig. 4.5e. The Nyquist sampling rate results in a poor HR resolution of ± 26 bpm while at 300 Hz the resolution is ± 1 bpm.

$$Resolution_{HR} = \frac{1}{(PeakPos2 - PeakPos1) \pm \Delta T_s} \tag{4.1}$$

Input Data Width

Although the number of bits used to represent the signal barely affects the HR or RR calculation, it affects the SpO2 calculation. The theoretical error of the $\frac{AC}{DC}$ ratio is calculated as shown in Eq. 4.2 where ΔNB

represent the minimum resolution achieved for a particular number of bits (IDW).

$$Error_{AC/DC} = \frac{AC}{DC}\left(\frac{AC}{\pm\Delta NB} + \frac{DC}{\pm\Delta NB}\right) \tag{4.2}$$

Fig. 4.5c shows the error for $\frac{AC}{DC}$ when varying the number of bits to represent the signal from 4 to 14. When using 4 bits the normalized error is rounded to 0.2. When using 14 bits the error is reduced to $1x10^{-}6$

FFT length (N) considerations

The FFT size depends on the sampling rate used to acquire the data. The minimum N should include enough data to detect a minimum of 3 peaks to obtain at least one valid HR and one $\frac{AC}{DC}$ calculation per window. Fig.4.5e shows the FFT N required for different sampling rates. The optimum sampling rate, depends on the application requirements. Fig. 4.5c shows the Matlab results of the HR calculated with a PPG sampled at 14 Hz, compared with a PPG signal digitized at 300 Hz, on Fig. 4.5d. The error of $\pm 26bpm$ is very noticeable at the lower sampling frequency. However, when post-processing the HR results with a moving average filter the average HR for both versions are nevertheless very similar, see Fig. 4.5f. In a fitness monitoring device, for example, there is no advantage on incrementing the sampling rate beyond the Nyquist frequency and a FFT length of 32Hz might suffice. On the other hand, a system for neo-natal monitoring requires an error of less than $\pm 5bpm$ [38] which translates into a minimum f_s of 75Hz and a FFT length N of 128.

4.4.2 Parameterizable behavioral description

In order to target different operation scenarios, and reduce the design and functional verification costs, we developed a parameterized behavioral description of the proposed architecture. Fig.4.6 contrasts the behavioral simulation results for an architecture set for N=128 compared with an architecture set for N=64. The more noticeable difference between these two simulations, is the time it takes to first complete the window, 128 samples vs 64 samples, and then complete the FFT-IFFT calculations for both channels. By changing a single parameter N, we can generate the behavioral description for a fitness monitor, being its main requirement ultra low power consumption; or a neo-natal monitoring system where the HR resolution is more important than the power limitation. Two more parameters can be set and propagated throughout the architecture: the IDW and Data Path Width according with the expected operation conditions for the IP

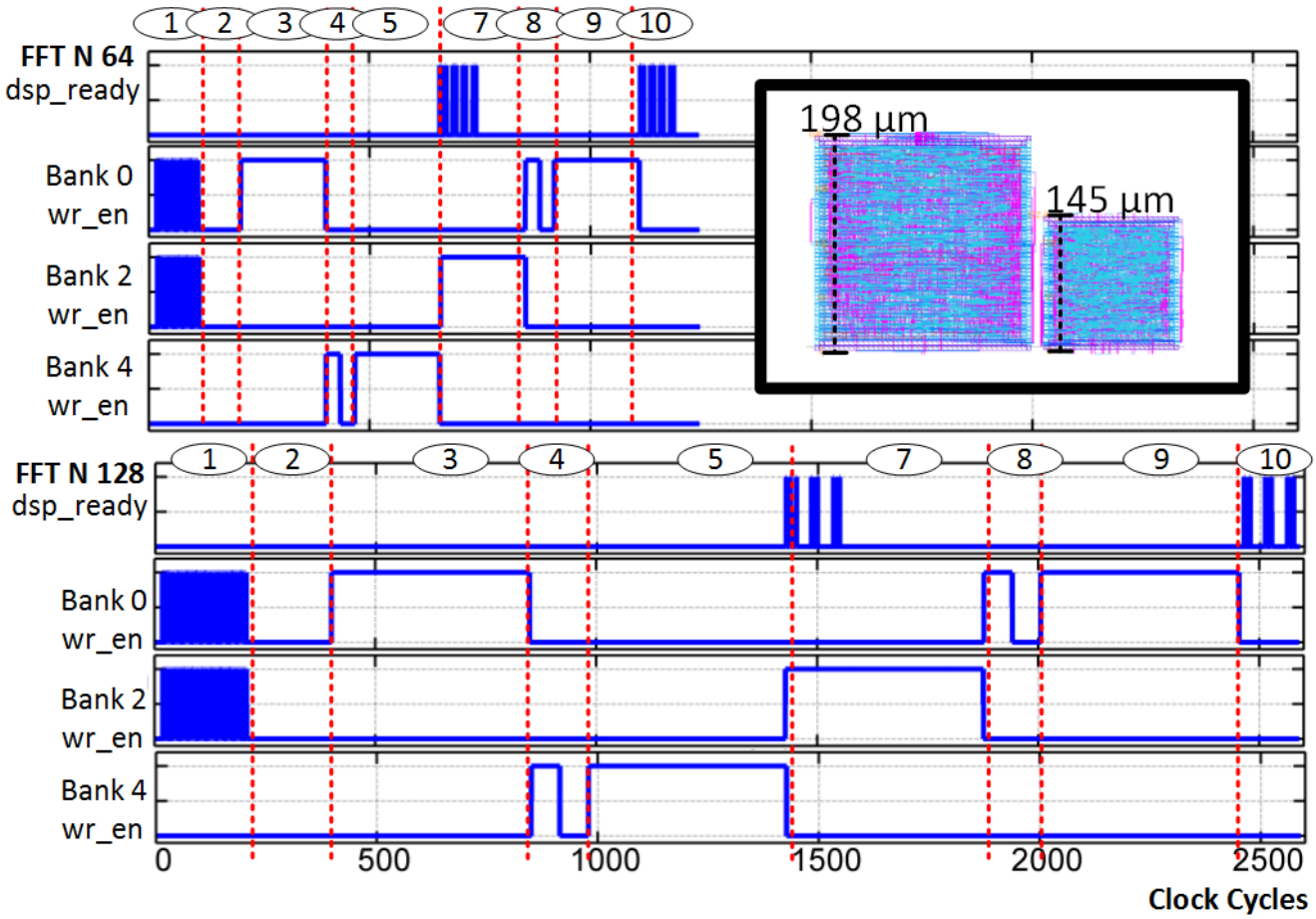

Figure 4.6: Verilog simulation results for a PPG processor with FFT length of 64 compared with a PPG processor with FFT length of 128. Also, see a physical implementation of a memory bank with clock gating per row (left) and without clock gating per row (right)

Block and the application requirements.

4.5 Results

As an example, we synthesized the behavioral description to obtain an architecture that supports a sampling rate between 15 and 40 Hz with a FFT of length $N = 64$ to obtain a HR error of less than $\pm 10 bpm$. The logic receives a 7 bits input to obtain an error in the $\frac{AC}{DC}$ ratio of less than ± 0.2. We synthesized this system using a Global Foundries 65 nm library optimized for nearthreshold operation at 500 mV. Optimizing the memory deserves special mention since a good portion of the area and the power consumption can be attributed to this portion of the design. We implement the memory as SCMs because of the reduced design effort,

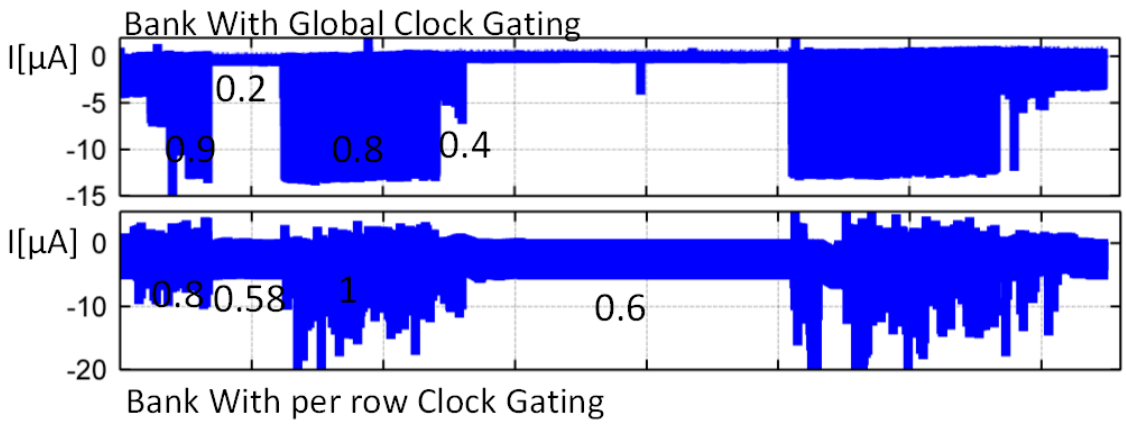

Figure 4.7: Current consumption for a memory bank implemented with per-row Clock gate (above) vs Global clock gate. The numbers on top of the simulated current corresponds to the average power consumption for that period of time in μW.

compared with custom based designs as SRAMS[39]. A typical approach to avoid the clock propagation to idle registers and reduce the power consumption in the memory is to add per row clock gate logic. This is effective in reducing the average power consumption at expenses of almost doubling up the area. Fig. 4.6, compares the size of both implementations after Place & Route. However, a close observation to the behavioral simulations waveform, Fig 4.4, shows that out of 6 banks only 2 are active simultaneously, for the majority of the window processing time. Therefore instead of a clock gate per row we implement a global clock gate strategy. Fig.4.7 compares the current consumed by a memory bank with global vs. per row clock gating. This approach reduces the the power consumption on the idle banks from $0.6\mu W$ to $0.2\mu W$ at the expense of increasing the power consumed on the active banks. To gain granularity in the clock gating approach, we divide each memory bank in sub-banks that can be clock gated independently. In the simulation results for this example, Fig. 4.8, we show the progressive increment in current when the clock is activated in a per sub-bank basis up to the point where all the bank is active. (Stage 1). Similarly, when the full memory bank is idle simulation results shows that the power consumption is reduced to $0.2\mu W - 0.1\mu W$.

Table4.1, compares the average power for the six banks in the system with the average power consumed by a memory bank following the per row clock gate approach. To complement the global clock gate approach, we implement a clock gate controller. This block piggy back in the input data counters and the state machines that multiplex the FFT-IFFT core between the channels to activate the clock enable signals for the memory

Component	Average Power Per Component.
Bank 0	0.75µ
Bank 1	0.7µ
Bank 2	0.65µ
Bank 3	0.65µ
Bank 4	0.65µ
Bank 5	0.69µ
Control	2.1µ
Bank with no Clock Gate	2.0µ
Bank with per row Clock gate	0.8 µ

Table 4.1: Power Consumption per block.

banks and sub-banks.

4.6 Conclusion

In this work we present a parameterizable block IP for PPG processing to be optimized according witht the operation conditions and metrics of interest of the application. An ideal configurable FFT-IFFT based bandpass filter allow us to effectively extract DC, HR and RR information. We synthesized the system with an optimized library for near-threshold operation at 500mV, 166KHz to allow ultra low power. We used SCMs for the FFT's memory banks. Optimizations in area and power can be achieved by careful analysis of the system activity. For sections of the memory with low utilization, a global clock gating scheme yields the same average power consumption than a per row scheme and reduces the area by roughly 50%.

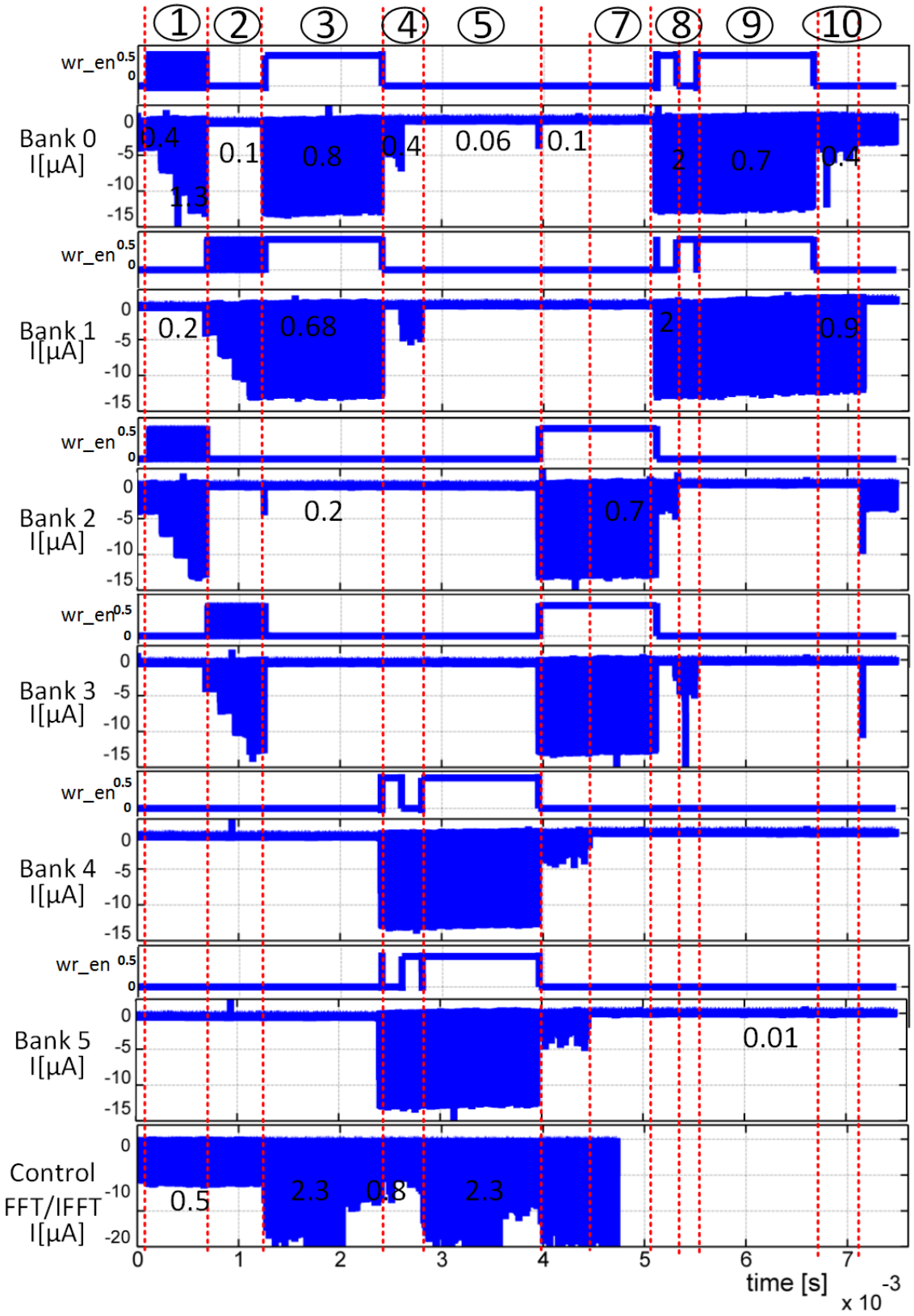

Figure 4.8: Current Consumption for each component in the data path. The numbers on top of the simulated current corresponds to the average power consumption for that period of time in μW. Note: There is a missing part in the current for the controller due to memory issues in the server.

Chapter 5

Conclusions and Future Work

1 trillion nodes on the IoT requires SoCs that are portable, consume power on the order of decades of μW, and wirelessly communicates events that are relevant to the user. Technology developments on the area of energy harvesting and analog and digital subthreshold design have enabled the development of these new kind of SoCs. However two main areas have not been really explored:

- The elevated NRE costs associated with the increased levels of integration required to develop these nodes.
- The lack of system level analysis prior to architecture definition taking into account specific application requirements.

5.1 NRE Costs

The BSN Team created a ULP wireless SoC with integrated power management, transceiver, sensing interfaces, DSP, and flexible clocking. This platform achieved the highest level of integration, including energy harvesting and a full transceiver, for the lowest power on both versions of the node. Furthermore, it included flexible sensing interfaces and on-chip processing for a diverse set of target applications.

This thesis in particular addressed the challenge of integration complexity by designing an implementing a re-use and verification aware methodology for Sensor Nodes:

- We removed the detailed descriptions of inter-block communication by using SystemVerilog interfaces.
- We defined lego block abstractions that can be easily replaced for architecture evolution, migration or modification.
- The combination of the two previous items enable us to :
  - Perform architecture evolution, modification or migration by changing only 1% or 2% of the total HDL description vs an optimistic %32.
  - Start early system level integration with abstract components that don't need to be fully developed or even defined. This methodology reduces the risk of blocks and interfaces that are still changing delaying or affecting the system level integration evolution.
  - Facilitates the verification process by creating an homogeneous, standardized architecture.

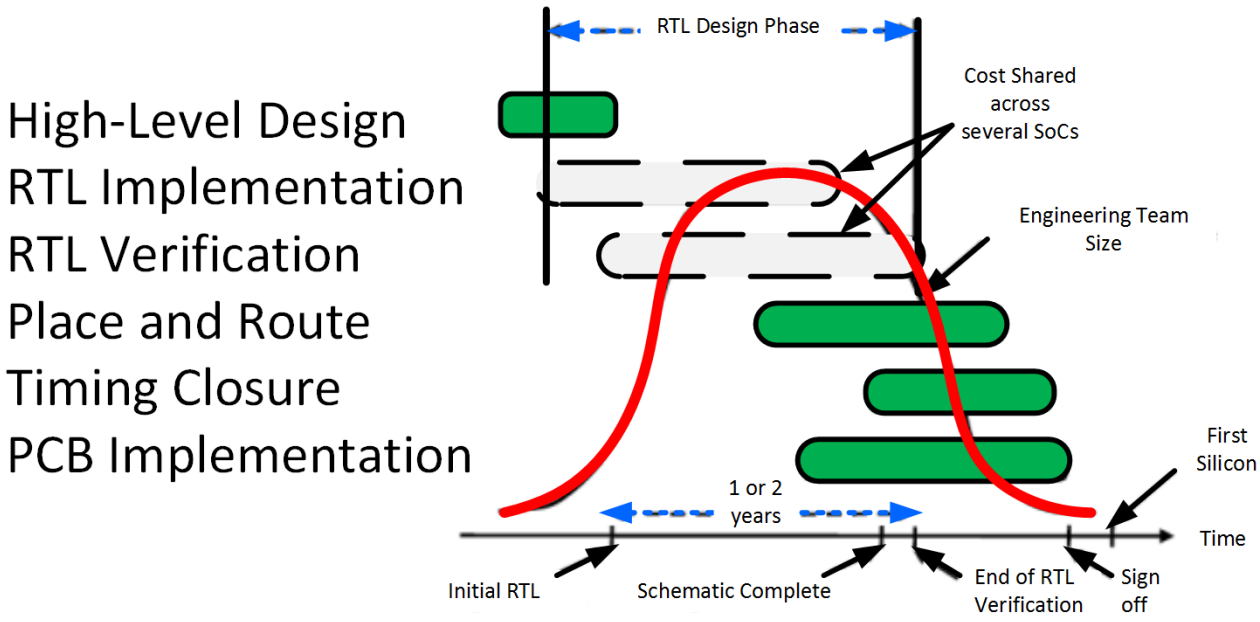

Figure 5.1: ASIC development timeline example. In gray the part of the development flow that this work addresses by designing re-use and abstraction methodologies. [40]

5.2 System Level Analysis from the Application perspective

Several techniques as Sub-threshold design, Clock gating, Power Gating, and DSP accelerators can be used for minimizing the power consumption of a wireless sensor node. However, the efficacy of all these techniques depend on the activity factor dictated by the application. Very little work has been done to address the evaluation of these SoCs from the applications perspective. We created a tool that enables the evaluation/exploration of a platform at the system level by re-using the algorithm codes for verification/hardware architecture exploration.

Beyond the verification of individual blocks, The major potential for AVEBoS is that it enables system level evaluation/exploration of applications with no extra engineering cost. We take advantage of the codes already developed during algorithm exploration. Furthermore, we use languages and tools that are already widely used by the designer and the algorithm developer, thus no extra difficulty is inserted to the process. Finally this application aware, architecture exploration is useful for an already existing platform (Algorithm validation, System Verification) or during the architecture definition stage.

5.2.1 Final Remarks

This work addresses for the first time the evaluation and exploration of Wireless Sensor Nodes from a system level perspective. We take the application requirements as the main driver to make informed decissions about the architecture in sub-blocks that can be incorporated in larger systems. With this methodology, we shift the view of a single SoC, to a family of SoCs able to adapt to different applications or operation conditions spreading the development and verification costs, see Fig.5.1, among different products.

5.2.2 Future Work

The application to be supported in a Wireless Sensor Node, should make a great impact during the architecture design stage. Although, in this work we start closing the gap between application and system level design, this area is just starting to be explored and there is still work to do in the following fronts:

- Integration Methodology:

  - There is still a need for rapid energy and power consumption estimations at the system level.
  - The pulseoximetry and the wheezing detection algorithm was not taped-out. Combining the results of each individual exploration as an application aware single accelerator that handles memory resources, power gating and clock-gating according the individual needs of the application is yet to be done.
  - The proposed concept of an architecture that is bus agnostic was show cased using toy models. The SoC described in Section 2.1 was designed with this idea in mind. However, this idea has yet to be explored at the full system level.

- AVEBoS:
  - Work in the AVEBoS tool flow to enable multivariable exploration.
  - AVEBoS architecture was designed to be modular. This modularity was thought to enable models from other sources as C models, and different HDL simulators. However this feature was never tested or explored.
  - AVEBoS handles a directory structure that enables true HDL re-usability. It generates a list of directories and files required to include a block as part of an architecture for the HDL simulator. This functionality can be expanded to automatically generate the list of files and directories required for the Synthesis flow. This reduce the potential for human errors when using a distributed RTL library as the proposed as part of AVEBoS.
  - Given the modularity and abstraction of the blocks and interfaces of the proposed approach, exploration towards automatic generation of sensor nodes seems feasible by using AVEBoS awareness of the re-usable IP properties.

Bibliography

[1] A. Shrivastava, "Mixed signal platform circuits for ultra low power systems," Ph.D. dissertation, University of Virginia, 2014.

[2] Y. Shakhsheer, "Lifetime improvement in body sensor networks," Ph.D. dissertation, University of Virginia, 2013. [Online]. Available: http://libra.virginia.edu/catalog/libra-oa:3658

[3] A. Klinefelter, "Vlsi architectures for digital signal processing on energy-constrained systems-on-chip," Ph.D. dissertation, University of Virginia, 2015.

[4] J. Boley, "Circuit and cad techniques for expanding the sram design space," Ph.D. dissertation, University of Virginia, 2014.

[5] G. Asada, A. Burstein, D. Chang, M. Dong, M. Fielding, E. Kruglick, J. Ho, F. Lin, T.-H. Lin, H. Marcy, R. Mukai, P. Nelson, F. Newberg, K. Pister, G. Pottie, H. Sanchez, O. Stafsudd, S. Valoff, G. Yung, and W. Kaiser, "Low power wireless communication and signal processing circuits for distributed microsensors," in *Circuits and Systems, 1997. ISCAS '97., Proceedings of 1997 IEEE International Symposium on*, vol. 4, Jun 1997, pp. 2817–2820 vol.4.

[6] N. Roberts and D. Wentzloff, "A 98nw wake-up radio for wireless body area networks," in *Radio Frequency Integrated Circuits Symposium (RFIC), 2012 IEEE*, June 2012, pp. 373–376.

[7] S. Oh, N. Roberts, and D. Wentzloff, "A 116nw multi-band wake-up receiver with 31-bit correlator and interference rejection," in *Custom Integrated Circuits Conference (CICC), 2013 IEEE*, Sept 2013, pp. 1–4.

[8] A. Klinefelter, N. Roberts, Y. Shakhsheer, P. Gonzalez, A. Shrivastava, A. Roy, K. Craig, M. Faisal, J. Boley, S. Oh, Y. Zhang, D. Akella, D. Wentzloff, and B. Calhoun, "21.3 a 6.45 uw self-powered iot soc with integrated energy-harvesting power management and ulp asymmetric radios," in *Solid- State Circuits Conference - (ISSCC), 2015 IEEE International*, Feb 2015, pp. 1–3.

[9] A. B. Kahng and G. Smith, "A new design cost model for the 2001 itrs," in *Quality Electronic Design, 2002. Proceedings. International Symposium on*. IEEE, 2002, pp. 190–193.

[10] A. Kahng, "Design technology productivity in the dsm era," in *Design Automation Conference, 2001. Proceedings of the ASP-DAC 2001. Asia and South Pacific*, 2001, pp. 443–448.

[11] G. Smith, "Updates of the itrs design cost and power models," in *Computer Design (ICCD), 2014 32nd IEEE International Conference on*, Oct 2014, pp. 161–165.

[12] A. Wang, B. H. Calhoun, A. P. Chandrakasan, A. Chandrakasan, and A. Chandrakasan, *Sub-threshold design for ultra low-power systems*. Springer, 2006, vol. 95.

[13] S. Khanna and B. H. Calhoun, "Serial sub-threshold circuits for ultra-low-power systems," in *Proceedings of the 2009 ACM/IEEE international symposium on Low power electronics and design*. ACM, 2009, pp. 27–32.

[14] M. Dong, K. Yung, and W. Kaiser, "Low power signal processing architectures for network microsensors," in *Low Power Electronics and Design, 1997. Proceedings., 1997 International Symposium on*, Aug 1997, pp. 173–177.

[15] B. Zhai, D. Blaauw, D. Sylvester, and K. Flautner, "Theoretical and practical limits of dynamic voltage scaling," in *Proceedings of the 41st annual Design Automation Conference*. ACM, 2004, pp. 868–873.

[16] J. Greenough. (2015, April) The 'internet of things' will be the world's most massive device market and save companies billions of dollars. [Online]. Available: http://www.businessinsider.com/how-the-internet-of-things-market-will-grow-2014-10

[17] Y. Zhang, F. Zhang, Y. Shakhsheer, J. Silver, A. Klinefelter, M. Nagaraju, J. Boley, J. Pandey, A. Shrivastava, E. Carlson, A. Wood, B. Calhoun, and B. Otis, "A batteryless 19 u w mics/ism-band energy harvesting body sensor node soc for exg applications," *Solid-State Circuits, IEEE Journal of*, vol. 48, no. 1, pp. 199–213, Jan 2013.

[18] O. Girard. (2010) Open msp430. [Online]. Available: http://opencores.org/ocsvn/openmsp430/openmsp430/trunk/openmsp430

[19] J. Kwong and A. Chandrakasan, "An energy-efficient biomedical signal processing platform," in *ESSCIRC, 2010 Proceedings of the*, Sept 2010, pp. 526–529.

[20] N. Van Helleputte, M. Konijnenburg, H. Kim, J. Pettine, D.-W. Jee, A. Breeschoten, A. Morgado, T. Torfs, H. de Groot, C. Van Hoof, and R. Yazicioglu, "18.3 a multi-parameter signal-acquisition soc for connected personal health applications," in *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International*, Feb 2014, pp. 314–315.

[21] A. L. Sangiovanni-Vincentelli, P. C. McGeer, and A. Saldanha, "Verification of electronic systems," in *Design Automation Conference Proceedings 1996, 33rd*. IEEE, 1996, pp. 106–111.

[22] K. Bult, A. Burstein, D. Chang, M. Dong, W. Kaiser *et al.*, "Wireless integrated microsensors," in *Proc. Tech. Dig. Solid-State Sensor and Actuator Workshop*, 1996, pp. 205–210.

[23] P. Wilcox, *Professional Verification*. Springer, 2004.

[24] S. A. Taplidou and L. J. Hadjileontiadis, "Wheeze detection based on time-frequency analysis of breath sounds," *Computers in biology and medicine*, vol. 37, no. 8, pp. 1073–1083, 2007.

[25] L. Bentur, R. Beck, M. Shinawi, T. Naveh, and N. Gavriely, "Wheeze monitoring in children for assessment of nocturnal asthma and response to therapy," *European Respiratory Journal*, vol. 21, no. 4, pp. 621–626, 2003.

[26] H. Pasterkamp, S. S. Kraman, and G. R. Wodicka, "Respiratory sounds: advances beyond the stethoscope," *American journal of respiratory and critical care medicine*, vol. 156, no. 3, pp. 974–987, 1997.

[27] M. Bahoura, "Pattern recognition methods applied to respiratory sounds classification into normal and wheeze classes," *Computers in biology and medicine*, vol. 39, no. 9, pp. 824–843, 2009.

[28] S. M. Shaharum, K. Sundaraj, and R. Palaniappan, "A survey on automated wheeze detection systems for asthmatic patients," *Bosnian Journal of Basic Medical Sciences*, vol. 12, no. 4, p. 249, 2012.

[29] L. Cohen, "Time-frequency distributions-a review," *Proceedings of the IEEE*, vol. 77, no. 7, pp. 941–981, Jul 1989.

[30] R. A. Altes, "Detection, estimation, and classification with spectrograms," *The Journal of the Acoustical Society of America*, vol. 67, no. 4, pp. 1232–1246, 1980.

[31] S. Emrani and H. Krim, "Wheeze detection and location using spectro-temporal analysis of lung sounds," in *Biomedical Engineering Conference (SBEC), 2013 29th Southern*, May 2013, pp. 37–38.

[32] K. Nakajima, T. Tamura, and H. Miike, "Monitoring of heart and respiratory rates by photoplethysmography using a digital filtering technique," *Medical engineering & physics*, vol. 18, no. 5, pp. 365–372, 1996.

[33] W. Shin, Y. D. Cha, and G. Yoon, "Ecg/ppg integer signal processing for a ubiquitous health monitoring system," *Journal of Medical systems*, vol. 34, no. 5, pp. 891–898, 2010.

[34] A. Arza, J. Lazaro, E. Gil, P. Laguna, J. Aguilo, and R. Bailon, "Pulse transit time and pulse width as potential measure for estimating beat-to-beat systolic and diastolic blood pressure," in *Computing in Cardiology Conference (CinC), 2013*, Sept 2013, pp. 887–890.

[35] S. Hey, A. Gharbi, B. von Haaren, K. Walter, N. Konig, and S. Loffler, "Continuous noninvasive pulse transit time measurement for psycho-physiological stress monitoring," in *eHealth, Telemedicine, and Social Medicine, 2009. eTELEMED '09. International Conference on*, Feb 2009, pp. 113–116.

[36] E. Gil, R. Bailon, J. M. Vergara, and P. Laguna, "Ptt variability for discrimination of sleep apnea related decreases in the amplitude fluctuations of ppg signal in children," *Biomedical Engineering, IEEE Transactions on*, vol. 57, no. 5, pp. 1079–1088, May 2010.

[37] X. Xiao, E. Oruklu, and J. Saniie, "An efficient fft engine with reduced addressing logic," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 55, no. 11, pp. 1149–1153, Nov 2008.

[38] C. Mann, C. Ward, M. Grubb, J. Teoh, J. Crowe, B. Hayes-Gill, N. Marlow, and D. Sharkey, "Can we improve delivery room monitoring for newborns? a novel photoplethysmographic heart rate monitor evaluated among stable nicu infants," *Archives of Disease in Childhood-Fetal and Neonatal Edition*, vol. 96, no. Suppl 1, pp. Fa2–Fa3, 2011.

[39] P. Meinerzhagen, S. Sherazi, A. Burg, and J. Rodrigues, "Benchmarking of standard-cell based memories in the sub-vt domain in 65-nm cmos technology," *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on*, vol. 1, no. 2, pp. 173–182, June 2011.

[40] R. Reis, V. Mooney, and P. Hasler, *VLSI-SoC: Advanced Topics on Systems on a Chip: A Selection of Extended Versions of the Best Papers of the Fourteenth International Conference on Very Large Scale Integration of System on Chip (VLSI-SoC2007), October 15-17, 2007, Atlanta, USA*. Springer, 2009, vol. 291.