Royal Institute
of Technology

Performance Analysis of Application-Specific Multicore Systems on Chip

Iyad Al Khatib

A thesis submitted to
The Royal Institute of Technology
in partial fulfillment of the requirements for
the degree of Doctor of Technology

June 13, 2008

Department of Electronic, Computer, and Software Systems (ECS)
School of Information and Communication Technology (ICT)
Royal Institute of Technology (KTH)

Stockholm, Sweden

Al Khatib, Iyad

Performance Analysis of Application-Specific Multicore Systems on Chip

Doctoral Thesis in Electronic and Computer Systems
Department of Electronic, Computer, and Software Systems (ECS)
School of Information and Communication Technology (ICT)
Royal Institute of Technology (KTH)

TRITA-ICT/ECS AVH 08:06
ISSN 1653-6363
ISRN KTH/ICT/ECS/ AVH-08/06--SE
ISBN 978-91-7178-960-0

KTH/ICT/ECS: Electronic, Computer, and Software Systems
Royal Institute of Technology
Isafjordsgatan 39
P.O. Box Forum 120
SE-164 40 Kista
Sweden

ABSTRACT

The last two decades have witnessed the birth of revolutionary technologies in data communications including wireless technologies, System on Chip (SoC), Multi Processor SoC (MPSoC), Network on Chip (NoC), and more. At the same time we have witnessed that performance does not always keep pace with expectations in many services like multimedia services and biomedical applications. Moreover, the IT market has suffered from some crashes. Hence, this triggered us to think of making use of available technologies and developing new ones so that the performance level is suitable for given applications and services. In the medical field, from a statistical viewpoint, the biggest diseases in number of deaths are heart diseases, namely Cardiovascular Disease (CVD) and Stroke. The application with the largest market for CVD is the electrocardiogram (ECG/EKG) analysis. According to the World Health Organization (WHO) report in 2003, 29.2% of global deaths are due to CVD and Stroke, half of which could be prevented if there was proper monitoring. We found in the new advance in microelectronics, NoC, SoC, and MPSoC, a chance of a solution for such a big problem. We look at the communication technologies, wireless networks, and MPSoC and realize that many projects can be founded, and they may affect people's lives positively, as for example, curing people more rapidly, as well as homecare of such large scale diseases. These projects have a medical impact as well as economic and social impacts. The intention is to use performance analysis of interconnected microelectronic systems and combine it with MPSoC and NoC technologies in order to evolve to new systems on chip that may make a difference. Technically, we aim at rendering more computations in less time, on a chip with smaller volume, and with less expense. The performance demand and the vision of having a market success, i.e. contributing to lower healthcare costs, pose many challenges on the hardware/software co-design to meet these goals. This calls upon the development of new integrated circuits featuring increased energy efficiency while providing higher computation capabilities, i.e. better performance. The biomedical application of ECG analysis is an ideal target for an application-specific SoC implementation. However, new 12-lead ECG analyses algorithms are needed to meet the aforementioned goals. In this thesis, we present two novel algorithms for ECG analysis, namely the Autocorrelation-Function (ACF) based algorithm and the Fast Fourier Transform (FFT) based algorithm. In this respect, we explore the design space by analyzing different hardware and software architectures. As a result, we realize a design with twelve processors that can compute 3.5 million arithmetic computations and respect the real time hard deadline for our biomedical application (3.5-4 seconds), and that can deploy the ACF-based and FFT-based algorithms. Then, we investigate the configuration space looking for the most effective solution, performance and energy-wise. Consequently, we present three interconnect architectures (Single Bus, Full Crossbar, and Partial Crossbar) and compare them with existing solutions. The sampling frequencies of 2.2 KHz and 4 KHz, with 12 DSPs, are found to be the critical points for our Shared-Bus design and Crossbar architecture, respectively. We also show how our performance analysis methods can be applied to such a field of SoC design and with a specific purpose application in order to converge to a solution that is acceptable from a performance viewpoint, meets the real-time demands, and can be implemented with the present technologies while at the same time paving the way for easier and faster development. In order to connect our MPSoC solution to communication networks to transmit the medical results to a healthcare center, we come up with new protocols that will allow the integration of multiple networks on chips in a communication network. Finally, we present a methodology for HW/SW Codesign for application-specific systems (with focus on biomedical applications) that require a large number of computations since this will foster the convergence to solutions that are acceptable from a performance point of view.

ACKNOWLEDGMENTS

"He who does not seek advice is a fool. His folly blinds him to Truth and makes him evil, stubborn and a danger to his fellow man."

Joubran Khalil Joubran- known also as Kahlil Gibran (1883 – 1931 A.D.)

Firstly, I would like to thank my advisor, Professor Axel Jantsch, whom I would simply describe as one of the greatest advisors I have ever seen for his knowledge, understanding, logic, and ease of cooperation. I thank Professor Axel from the bottom of my heart, for without him, surely and truly this thesis would not have existed.

I thank Dr. Robert Ronngren for his help and support all the time. Also many thanks are to Prof. Slimane Ben Slimane for the fruitful help and the nice talks. Special thanks are to Professor Luca Benini, whom I enjoy working with and learning from. In addition, I thank my friend professor Davide Bertozzi for creating a wonderful atmosphere at work. I have learned a lot from him, so I thank him for that! I thank Francesco Poletti for the great working days we spent in Italy. Infinite thanks are not enough for the great rose and wonderful person who never left me at any second of my hard and good times, Ms. Marilet De Andrade Jardin. Thank you very much for all the support, and I think that even if I give you everything I could, that won't be enough to really thank you for the amount of love, support and sweetness you showed. Special Thanks to my great friends Lennart Johansson, Victor Kordas, and Anders Nilsson, without whom I could not finish this thesis. Their understanding was very important! Many thanks go to Mr. Peter Ekman, Mr. Kaj Mikkos, and Ms. Paola Diaz whose support from ALMI for my patents was so great and unforgettable! Thank you very much! Thanks to Dr. Luc Onana Alima and his family for all the good times and the support. I also thank my friend Miguel Simon for the great work times and the wonderful laughs! Also I thank my friend Moustafa Chenine for the support and the good chats and advice. I also thank my friend Wim Michielsen for his advice and for being always supportive. I also thank all my friends, whose support was essential to finish this thesis. Special thanks to Katherine Stuart, whose love and support especially in my first few years in Sweden were basic for my success in many things including this thesis. I thank my brother Bassam Kayal for words can not describe how I trust him and count on him in most of my work and life issues especially in Sweden. I thank Bassam and his brothers and sisters, and mainly Mehdi and Adnan for the wonderful, fruitful and supportive chats and meetings! I also thank Zaidoon Jabbar, who was available to help me all the time! Moreover, I thank Mr. Khaled Khalil and his wonderful family. Khaled was very helpful and even just talking to him and getting his support saved me from wasting much of my time. Thank you very much Khaled to you and your understanding family! Many thanks are for Mohammad Saleh, who has helped me in many of my work issues in research and in many life matters. I will never forget the hard times that we went through together and also the nice good laughs. I thank my brother Nicolas Baron so much since, honestly, his presence in the hardest times of my life was the reason why I survived well. Many thanks fly to my brother Xavier Jombart and his beautiful lady Therese. I also thank my brother Giuseppe Russo for all his support and understanding. Special thanks to his family, his father Franco, his mother Franca, and his brother and all his uncles and aunts. With them, I always feel that I am with my family! Special thanks to the wonderful Calabria and Calabrians! Giuseppe remains in a very special place in my heart always! I thank also Dr. Carlo Spellucci, the traveling brother, and all his family. Many thanks for the great help many times. I thank Dr. Fabio Curti for all the help and for being an excellent brother and friend. Thanks to Prof. Maurizio Di Ruscio and Prof. Carlo Uliveiri, who are excellent friends and researchers! Discussions with them led to endless ideas for me! Thanks to Francesco as well! Thanks to my brother Diego for his emails in science and in social affairs. Many thanks to my brother Nunzio Giovinazzi for his great support and wonderful laughs and time spent together! I also thank my friend Oscar Alcamo for the wonderful time we spent together! I thank Christine Ardini for the wonderful chats we had while in the lab in Italy. Special thanks to Cristina Severi, Samanta, and Giulia for the great wonderful times and laughs we had together! Thanks to my brother Biagio Ancarola and all his family. His mother is like my mother, his sisters are like my sister, and his father is like my father! I love them and miss them always, and I send love to all Napoli and all Italy! I also thank my brother Paolo Ponnzio for the great help and the great support for, really, without Paolo many things in my life would not have existed. Thanks Paolo. I also thank my brother and roommate Paolo Bonfiglio, without whom I would not have finished this thesis! Special thanks to my Italian brother Alessandro Baldoni and his support when needed. Many thanks go to my brother Gianluigi Baldessi for his great support and help when needed. Italy has a special place in my heart, the same way as Lebanon has exactly. Many thanks to Tamer ElNadi, who has helped me many times, and his support was always great! Thanks to Professor Mohammad Ismail for the great advice and the great times talking and laughing! Special thanks also to my friend George in Greece and to my friends Yannis and Kharis in Crete! Special thanks to my great friends, Mohamed Bechara, Hasan Khalifeh, and Mazen Hajjar, who never hesitated to help at any time. I remain grateful to them all. I also thank my wonderful and great hearted friend Jad Nasreddine for all the wonderful conversations and advice. I also thank my friend Jad Atallah for his endless support and for always being there whenever I needed him, and I really mean it, whenever I needed him. Moreover, thanks to Sleiman Bou Sleiman for he was always there, and thanks for all the help and support Sleex! Thanks to my long-time roommate and friend Dr. Tarvo Raudvere for all the good chats, advice and fruitful discussions. Thanks for Abbe Issa and Jonny Youssef for the great times, the times of serious work, the support, and the great advice! Thanks to Abu Khaled, Samer for the nice laughs and the Kabab! Special Thanks of course are to my brother Adel El Hassan for the great support, endless advice, and the indescribable help he always gave me! Thank you so much Adel to you and your family, and special thanks to your lady for the Mouloukhiya! Thanks to my friend Ibrahim Himad for his help all the time and for the support and the Mouloukhiya. I thank Mohammad Ali for the help with the tickets.

Very Special thanks go to Mrs. Christian Gizrawi, the councilor of the Lebanese embassy in Sweden. for her endless support and help. Thank you very much Christian, you will always remain a friend! Many thanks to my friend Sharbil Nader for the endless support he showed!

Special thanks to Ahmad Doughan who was always there ready to laugh throughout our friendship. Special thanks to my brother Ossama Dimassi for his help, support, and advice. Your presence by the phone and when I needed you was always important for my success. Thanks also to my friend Rajab for all his wonderful laughs, jokes, and the nice advice! Many thanks to my great brother Hisham Ghawgi for his wonderful support in Sweden! Many thanks fly to my friend Irene Kordahi for the funny chats we have all the time! Thanks to my friend Jihad Wehbe for all the good times and support! Special thanks to Mr. Sarat Samanta in Sweden for the time spent chatting, the good advice, and the wisdom.

I remain grateful to my family, in the USA, who never hesitated to help me. In particular thanks to the Great Lady Sue! Thank you very much for being a great mother! Thanks to the genius Dave Peterson, who is not only family to be proud of, but also a very smart man, who solved many problems for me in life and scientifically! The endless love and support of Dave and Sue has been a great enrichment for my soul and life! I remain grateful to my mother and father in Lebanon, the Al Khatibs! Their endless love and support was the reason that made me what I am. Special thanks to my friend Richard Vine, who has been always supportive. Many thanks remain to my friend Dr. Loai Jalloul for his great help and support, as without him life would not have been the same! In the end, "What's Kebob" Loai? Many thanks to my friend Dr. Walid Ali Ahmad for all the great times we spent together laughing and joking. Thanks to Guy Davis and to Joseph. Guy's support was beyond description. His caring about my well being in many times is something I can not forget! I also thank Joseph for always being there! I would finally love to thank my "bestestest" friend Mr. Dudey Wudey III, Ian Marsh, soon Doctor Marsh for all the great times, laughs, and support! Deep thanks go to my uncle, Ph.D. and M.D. Ali Al Horr, who helped me understand all medical issues needed for this thesis. Many thanks and hugs to my great brother Wahid Al Kaisi, without whom I could not have done many things in this thesis! I miss you Wahid! "Lakaaan?" Many thanks to my friend Ahmad Abdallah at the AUB for his support on helping me understand ECG signals and their abnormalities. I also thank my great friend Anas Idris for all his support and the time spent together. I thank Amer Bzeih for all the wonderful times and support. Great thanks to Mr. Abu Antar, Hassan Zreik, who remains in a very special place in my heart for his support in logical thinking, in real friends' emotional support and advice, and in the discussions about the Dalai Lama.

Special thanks to the administrators in the ECS department, KTH, Hans, Julio, Richard, Enrico, Peter, Fredrik, and all others! Thanks to Carl Gustav Jansson for all the support and help! I thank Hans-Jörgen Rasmussen, Rita Fjällendahl, Angeta Herling, Lena Beronius, Rose-Marie Lövenstig, Gunnar Johansson, and Hans Larsson. I also thank Dr. Vlad Vlassov for all the nice chats and good advice.

Finally, and again I remain grateful to my families, in the USA and in the Lebanon, who never hesitated to help me. I also thank my brother Bassel Al Khatib for all his endless support. I thank my sister Raniem for the great conversations, emotional support, and love.

For all those who helped and I may have forgotten to mention, please forgive me and I thank you in this sentence.

"The true wealth of a nation lies not in its gold or silver but in its learning, wisdom and in the uprightness of its sons."

Joubran Khalil Joubran- known as Kahlil Gibran

(1883 – 1931 A.D.)

I dedicate this

To my families in the Lebanon and the USA

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Page

ABBREVIATIONS

3G	Third Generations for Mobile Communications
ACF	Autocorrelation Function
ADC	Analog to Digital Converter
ARM	Advanced RISC Machine, and prior to that Acorn RISC Machine, is a computer processor architecture developed by ARM Limited that is widely used in a number of embedded designs
ASIC	Application-Specific Integrated Circuit
aVF	Augmented Lead F label in the ECG technique
aVL	Augmented Lead L label in the ECG technique
aVR	Augmented Lead R label in the ECG technique
AWGN	Additive White Gaussian Noise
BBC	Backup Border Core
BC	Border Core in the NoC Autonomous System Proposed
BIONoC	Biomedical Network-on-Chip
BLIF	Berkeley Logic Interchange Format, to describe a logic-level hierarchical circuit in textual form
BPM	Beat Per Minute
CLRD	Chronic Lower Respiratory Disease
CFSM	Co-design Finite State Machine
CVD	Cardiovascular Disease
CPU	Central Processing Unit

DFT	Discrete Fourier Transform
DIP	Digital Image Processor
DMA	Dynamic Memory Allocation
DSP	Digital Signal Processing
DVB	Digital Video Broadcast, used in Satellite communications
DVB-RCS	DVB-Return Channel Satellite, a special technique for more reliability in DVB satellite communications
DVFS	Dynamic Voltage and Frequency Scaling is a technique that consists of varying the frequency and voltage of a microprocessor in real-time according to processing needs. This technique is used both for power-saving during off peak times, and as a protective measure to avoid over-temperature conditions
ECG	Electrocardiogram, a set of heart signals (also known as EKG)
EKG	Electrocardiogram, a set of heart signals (also known as ECG)
ENoP	External NoC Protocol
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FPGA	Field-Programmable Gate Array
FSM	Finite State Machine
GNTS	Global NAS Timing and Synchronization protocol
GPRS	General Packet Radio Service
HDL	Hardware Description Language
HW	Hardware
HRV	Heart Rate Variability
IC	Integrated Circuit

ICT	Information and Communication Technology
IFFT	Inverse Fast Fourier Transform (Inverse FFT)
IIR	Infinite Impulse Response; IIR Systems and filters have an impulse response function, which is non-zero over an infinite length of time. This is in contrast to finite impulse response filters (FIR) which have fixed-duration impulse responses
ILP	Instruction Level Parallelism
INoP	Internal NoC Protocol
IP	Intellectual Property
IT	Information Technology
LA	Left Arm sensor label in the ECG technique
LL	Left Leg sensor label in the ECG technique
MAC	Medium Access Control
MBC	Main Border Core in the NAS model
MF	Matched Filter
MFA	Message Format Agreement protocol in the NAS protocols
MPSoC	Multiprocessor System on Chip
NAS	NoC as an Autonomous System
NIC	Network Interface Card
NND	NAS Neighbor Discovery
NoA	Network-on-Chip Authentication
NoC	Network on Chip
PC	Personal Computer
PCMCIA	Personal Computer Memory Card International Association is an international standards body that defines and promotes the

	PC Card. PCMCIA now is used to mean a PC card
PDA	Personal Digital Assistant, is a handheld computer
QoS	Quality of Service
RA	Right Arm sensor label in the ECG technique
RAM	Random Access Memory
RF	Radio Frequency
RISC	Reduced Instruction Set Computer
RTEMS	Real-Time Executive Multiprocessor Systems
S/N	Signal-to-Noise Ratio
SCI	Serial Communication Interface
SDRAM	Synchronous Dynamic Random Access Memory
SIS	Silicon Integrated System
SoC	System on Chip
SQRS	Single-channel QRS detector, an ECG ambulatory algorithm
STFT	Short-Time Fourier Transform
SW	Software
UN	United Nations
USD	United States Dollars
VHSIC	Very High Speed Integrated Circuit
VLIW	Very Long Instruction Word, refers to a CPU architecture that uses instruction level parallelism (ILP)
WHO	World Health Organization of the United Nations
WNoC	Wireless NoC

WQRS	Single-channel QRS detector based on length transform, an ECG ambulatory algorithm
WSN	Wireless Sensor Network
XNF	Xilinx Netlist Format, a HDL for electronic circuit design developed by Xilinx, Inc.

Software and Communication Abbreviations

C/C++	C and C++ computer programming languages
MATLAB	MATLAB software and computer programming language
MPARM	Multiprocessor
Satcom	Satellite Communications
SystemC	System C computer language
Verilog	Verilog HDL is a hardware description language used to design and document electronic systems
VHDL	VHSIC Hardware Description Language

Unit abbreviations

µm	micrometer; equals 10^{-6} meter
µs	micro seconds, equals 10^{-6} seconds
bit	A binary digit that can be of value either 0 or 1
bps	bits per second
B	Byte: Information Unit equal to 8bits
G	Giga; equals 10^9, but for bits and bytes (binary data storage) it is equivalent to 1K*1K*1K

GHz	Giga Hertz
Hz	Hertz, unit of frequency
K	Kilo, equals 10^3, but for bits and bytes (binary data storage) it 1Kbytes is equivalent to 1024bits
KHz	Kilo Hertz
M	Mega, equals 10^6, but for bits and bytes (binary data storage) it is equivalent to 1K*1K = 1024*1024
MB	Mega Bytes
MHz	Mega Hertz
mm	millimeters, equals to 10^{-3} meters
ms	milliseconds, equals to 10^{-3} seconds
mV	mill-volts, equals to 10^{-3} volts
P4	Pentium Four
s, secs	second(s)

Definitions

Granularity	Granularity is the relative scale or size, and it can be the relative level of detail that characterizes an object or activity. It reveals how much a system contains separate components called granules.
Multinet-NAS	A NAS that is interconnected to more than one NAS at the same time.
Reliability	Bit error rates, bit loss, packet loss; Note that these terms are often mixed up, and the term QoS is sometimes used to refer to what we would rather call service guarantees or quality in general

NOMENCLATURES

$\frac{\partial y}{\partial t}(.)$	Partial derivative of *y* with respect to *t*
a	a is the vector that contains the filter coefficients for output y of the IIR filter
A(t), A[n], $ACF_y(.)$	Autocorrelation function of the signal *y*
authentic(.)	Function in the NND of the ENoP to check if the communicati9ng NoC can be authenticated
b	b is the vector that contains the filter coefficients for signal *x, the input signal for the IIR filter*
D(.)	Derivative
DFT(.)	Discrete Fourier Transform
f_s	Sampling frequency
g(*t*)	The window function used to slice the signal and apply the Fourier Transform in the Wavelets
h	Response function
H(z)	Transfer function of the filter
k	index in the discrete signal function
l	index in the discrete function
L	Number of Lags, i.e. needed number to have enough calculations to get the period of the signal
m	Index used in equations
Main_Backup(.)	Function in the NND of the ENoP that determines which link is the Main link and which is the Backup link between neighboring NASs (NoCs).

MFA(.)	Function in the NND of the ENoP that determines the message sizes and formats used for communication between two neighboring NoCs
min(x, y)	Minimum function that returns the minimum of *x* or *y*
n	Index used in equations
N	The maximum index of *n*, i.e. the total number of samples is *N*+1
N_MAX	The Maximum number used in the ENoP algorithm
N_Probes	Number of Probes in the NND algorithm
NHA_x	NoC Hardware Address proposed in the thesis, where *x* is an index reflecting the NAS number
NIF	Network Interface
n_{period}	The number of indices needed to have a period in the ACF-based algorithm
NPKI	The most recent running estimate of the noise peak in the Pan Tompkins algorithm
N_τ	The specific number of index points on the *x* axis that are equivalent to have τ (period) on the *x* axis.
PEAKI	PEAKI is the overall peak in the calculations of the Pan Tompkins algorithm
PROBE_PACKET_SIZE	The size of the packet used for probing in the NND algorithm in the ENoP
Probe_Reply	The parameter to detect if there was a reply from probing a NoC. It has a value of True or False.
PS	The value of the PROBE_PACKET_SIZE in the the NND algorithm in the ENoP
RANI	Random non-zero Digit used for the NND algorithm

RR_AVERAGE	The R-R measure in the Pan Tompkins algorithm
RR_HIGH_LIMIT	The high value of the RR measure based on RR_AVERAGE
RR_LOW_LIMIT	The low value of the RR measure based on RR_AVERAGE
RR_MISSED_LIMIT	The missed value of the RR measure based on RR_AVERAGE. After an RR MISSED LIMIT time has passed and no peaks were detected, we implement the search-back technique using T2 (the second threshold)
RR'_n	The most recent R-R interval that fell between the acceptable low and high RR-interval limits in RR_LOW_LIMIT, RR_HIGH_LIMIT, and RR_MISSED_LIMIT.
RR_n	The most recent R-R interval in the R-R calculations in the Pan Tompkins algorithm
R_y	Autocorrelation of the signal *y*
S(u, w)	The STFT calculated for the Wavelet Theory, where $f(t)$ is the signal under study, $g(t)$ is the window function used to slice the signal and apply the Fourier Transform, *u* is the time shift, and *w* is the frequency
SPKI	The most recent running estimate of the signal peak in the Pan Tompkins algorithm
synchronize(.)	Function in the NND of ENoP responsible to fix the synchronization for communication between two NoCs
T	Period of a periodic function in seconds
t	Time in seconds
t_0	The first sample time in the data chunk
t_1	The time variable sample time in the data chunk
T1, T2	Two adaptive thresholds (T1=2*T2) to select candidate QRS complexes in the Pan Tompkins algorithm

THRESHOLDI1	The first threshold applied in the Pan Tompkins algorithm
THRESHOLDI2	The second threshold applied in the Pan Tompkins algorithm
Time_Slot	The variable that is used for TS.
TS	The time of transmission, propagation, and processing of a probe message in the NND algorithm
u	The time shift of the window in the Wavelet
w	Frequency under study in the Wavelet
W(u, s)	Wavelet Transform , where *s* is the scale value
x	Index (variable) in the discrete function
X	Random Signal
y	Discrete signal used mainly for the ECG filtered signal in the equations
z	Variable of the filter in *H*(*z*)
Δt	Difference between t_1 and t_0
ρ	Index used in the equations
τ	Time period of a signal
ψ	The mother wavelet function

Chapter 1

Introduction

"There is nothing more difficult to take in hand, more perilous to conduct, or more uncertain in its success, than to take the lead in the introduction of a new order of things."

Niccolo Machiavelli (1469-1527 A.D.)

This chapter briefly introduces the problem we aim at solving and the motivation behind our goals. The main problem lies in analyzing one of the human heart signal-sets via the use of modern multicore technology instead of the nowadays used techniques that suffer from lack of accuracy and hinder the progress in heart diseases analyses. We also aim at founding a methodology that will foster and structure multicore HW/SW Codesign for Biomedical applications. The technology we adopt for the solution is the multiprocessor system-on-chip (MPSoC). MPSoC may also open new horizons to solving many problems (not only heart problems) that have been waiting for new technologies to evolve with new added values like quicker and finer analysis, parallelization, lower cost, flexibility, mobility, and lower power budgets. This chapter falls in three sections: section 1.1 describes my motivation that has been supported by my advisors, section 1.2 presents my contributions clearly, and section 1.3 introduces the organization of the remainder of this thesis.

1.1. Motivation

Concerned about the changes occurring in the beginning of the 21st century and their effects on human life, I looked at some statistics from the World Health Organization (WHO) and studies made by many healthcare centers [188], and I was appalled by some figures and threats presented. More precisely, I choose a statement by Anthony Rodgers [189], from the Clinical Trials Research Unit at the University of Auckland in New

Zealand, to describe the main drive behind my motivation since his statement clearly describes the problem and the possibility to find preventable solutions as follows:

> "Unless current trends are halted or reversed, over a billion people will die from cardiovascular disease in the first half of the 21st century. The large majority will be in developing countries and much of the life years will be lost in middle age. This would be an enormous tragedy, given that research in the last half of the 20th century showed that cardiovascular disease was largely preventable [189]. "
>
> Anthony Rodgers, 2004

Reading that the problem may be preventable, I was challenged to start searching for how I can use modern technology to aid in solving the problem. In this respect, we notice that the last decade has witnessed the birth of many electronic technologies including new wireless technologies, multicore systems, System on Chip (SoC), Network on Chip (NoC), and more. We observe a large scale success in Information and Communication Technologies (ICT) and services, however, at the same time we witness some performance problems in many services like real-time processing of multimedia services on embedded mobile systems and wireless networks [40]. In addition, in late 2000 the Information Technology (IT) market was staunchly affected by the falling down of the global market, which reached a level comparable to the 1940 crash and the nadir point in 1932 [140]. The market has not totally recovered from this crash since late 2000, and the ICT industry has suffered from a lack of investment and thus unsatisfactory performance or Quality of Service (QoS) in many related technologies and solutions. For instance, many companies had promised to deliver multimedia services over mobile networks, but they couldn't keep up with the time plans since many performance problems had faced such networks without any investment force to drive the development of solutions for these very problems. The lack of a satisfactory level of performance made many of these technologies not suitable for life critical services like biomedical applications. Hence, this also affects the advance in other fields (like biomedical engineering), which depend on the development of solutions in electronics and ICT in general. Consequently, this reality triggers me to think of attracting more investment by making use of available technologies and developing new ones so that the performance level becomes suitable for useful applications and services that have an impact on solving- at least- one of nowadays problems related to human life.

Some of these applications are in the medical field [72][73][74][75][79][82][83]. Hence, I choose the largest diseases from a statistical point of view: heart diseases, namely Cardio-Vascular Disease (CVD) and Stroke. According to the United Nations (UN) World Health Organization (WHO) report in 2005, cardiovascular disease is the number one cause of global deaths, and it is projected to remain the leading cause of death. The report estimates 17.5 million people that have died from CVD in 2005.

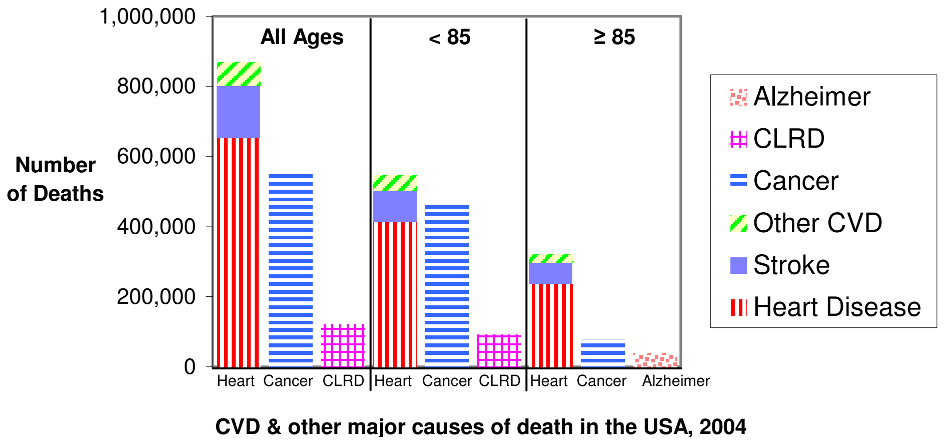


Figure 1. Number of deaths for the four largest causes of deaths: Heart diseases (CVD and Stroke), Cancer, Chronic Lower Respiratory Disease (CLRD), and Alzheimer; from the Heart Disease and stroke Statistics-2008 Update [69]. The numbers on top are in years of age: The first part on the left side includes patients from all ages. The middle part contains the statistics for patients under 85 years old. The part on the right includes the statistics for the patients above 85 years old.

Hence 30 % of all global deaths are due to CVD. The WHO 2005 report on CVD reads that "*If appropriate action is not taken, by 2015, an estimated 20 million people will die from cardiovascular disease every year, mainly from heart attacks and strokes* [187]." Moreover, this percentage of deaths is increasing every year [189]. Figure 1 shows the statistics in the USA *alone* for the major causes of deaths for different ages, for all ethnic groups, and for both women and men. The death statistics clearly show that Heart diseases exceed those for cancer, Chronic Lower Respiratory Disease (CLRD) like Asthmas, and Alzheimer. These statistics explain the high costs spent on heart diseases. Figure 2 shows some statistics from the American Heart Association, where billions of dollars are spent yearly on heart diseases (total of USD 326.1 billion for 2008) [69]. Hence, it would be of interest to find a solution that can help also in reducing these costs.

In this respect, I found in some new technologies like multicore SoC and NoC an opportunity to build solutions for such serious problems especially that multicore technologies are characterized by small sizes, high computation speeds (due to parallel processing), and lower power budgets [128]. On the other hand, wireless communication technologies and high speed networking have become essential in Telemedicine links [70][72][74][79]. This added to my inspiration to design and develop solutions, which may affect people's lives positively, e.g. curing people more rapidly [83]. Such solutions, if realized, may have medical, economic, social, and technical impacts. Moreover, many electronic chip-technologies are being used biomedical solutions [81]. Some companies and research organizations are also moving rapidly towards developing health-related projects [60]. Therefore, producing a commercial solution using MPSoC for medical purposes is not far from the electronic companies' abilities [81]. This reality has attracted my attention to use my knowledge in performance analysis of interconnected systems/microelectronics and combine it with the multicore/MPSoC technologies in order to try to work on a new system on chip that may make a difference [70].

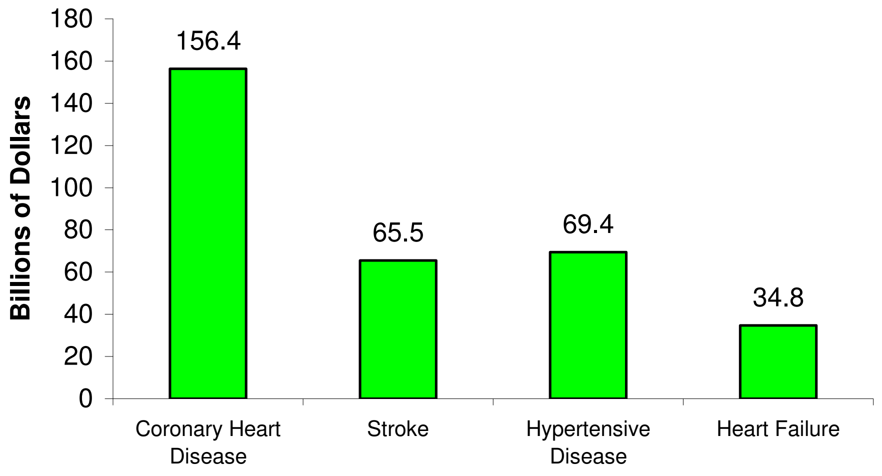


Figure 2. Estimated direct and indirect costs (total of USD 326.1 billion) of major CVD diseases and stroke in the USA in 2008; from the Heart Disease and Stroke Statistics-2008 Update [69].

Inspired by many talks with my advisor, Professor Axel Jantsch, and by many readings, I also found out that to be able to improve the MPSoC and NoC technologies, some interesting applications need to be implemented for this micro-networking technology in order that this field garners more attention and investment. This stems from my belief that in order for such solutions to be realized, then it is our responsibility as a scientific community to attract investment to such fields that have the potential to aid health related solutions. To accomplish this goal, we need to develop applications on SoC and MPSoC that reveal the real value and potential of such microelectronic technologies. Then I thought of SoC, MPSoC, and NoC as having one common denominator, which is being an 'on-chip network' [75][79][82], and many such on-chip networks may be part of larger networks [72][73][74][79][82]. In other words, networking is combining and connecting components to form a connected-system, whether on one chip, on one local area, or over a wide area. Therefore, I was motivated to render my research on an application that makes use of an 'on-chip network', i.e. making use of the small volume, low weight, multiprocessing computational capabilities, and lower power consumption. With this understanding, I believe that the best way to contribute to the progress in the field of MPSoC (which according to my belief includes NoC and SoC as subsets) is to utilize it in a new design for a special purpose that attracts attention (and investment), i.e. to be application-specific at least as a start.

The SoC applications I choose are medical applications, because of the need for a new technology to carry the complex analysis needed in medical applications. Many medical applications have remained without significant progress due to the lack of aiding technologies. For instance, in emergency situations like the use of ambulances (ground and plane ambulances), the physical size, power-consumption, and computational speeds of the computing device are crucial for success [73][79]. SoC could define a standard for many solutions in this respect. In addition, bio-informatics can gain some thrust when using MPSoC by being able to render more computations in less time, less volume, and less cost [77][80].

Another motivation for delivering services and applications for medicine and biomedical engineering is to have a good level of performance of biomedical devices, because these fields are very sensitive to errors. Biomedical, bio-informatics and space health applications (e.g., astronaut's health monitoring [74]) should be produced with a pedantic study on quality. To do so we need to analyze their performance during and after the design process, and we need to choose whether cycle or signal accuracy is needed for the performance analysis [80].

This thesis focuses, from an application point of view, on heart diseases since despite the ongoing advances in heart treatment, in many countries, the various forms of cardiovascular disease (CVD) and stroke remain by far the primary cause of death for both, men and women, regardless of ethnic backgrounds [188]. Many of these deaths are preventable by action on the major primary risk factors and with proper monitoring [78][80]. It is estimated that by 2010, CVD will be the leading cause of death not only in developed countries, but also in *developing countries* [188][189]. This causes a huge problem by year 2010, because developed countries do have- at least- some minimal standard for healthcare and emergency aid, but most developing countries do not have such a fast and efficient healthcare system, i.e. many CVD and Stroke cases that can be cured would not have the chance to reach the specialists/doctors. When the CVD problem reaches a critical point in developing countries (estimated to be 2010), there will be little time to act, and that motivated me more to try to work *now* to contribute to the solution and develop a cost-effective method that both developed and developing countries may use equally. Furthermore, since the rate of hospitalization increases with age for all cardiac diseases, a periodic cardiac examination is recommended [77]. Consequently, this fact increases the cost of hospitalization and health examinations in many *developed countries*, where the majority of the residents are elderly people.

Therefore, more efficient methods of cardiac diagnosis are desired to meet the great demand on heart examinations. However, state-of-the-art biomedical equipment for heartbeat sensing and monitoring lack the ability of providing large-scale analysis and remote, real-time computation at the patient's location (point of need) [77]. The intention of this work is to use MPSoC microelectronic technology to meet the growing demand for telemedicine services, especially in the mobile environment. The work attempts to address the existing problem of reducing the costs for hospitals/medical-centers through using MPSoC-based designs that may replace traditional biomedical machines and have higher quality, reduce the nurse's and doctor's work-load, and improve the quality of healthcare for patients suffering from heart diseases by exploring one potential solution. We also aim at founding a methodology for HW/SW Codesign for MPSoCs for biomedical applications. From the hospital side, deploying this solution will further reduce the costs of rehabilitating and following up on patients “primary care” since it allows better home-care [74]. Home-care ensures continuity of care, reduces hospitalization costs, and enables patients to have a quicker return to their normal life styles [77].

To analyze the heart, we specifically choose the Electrocardiogram (ECG/EKG), which is a set of electrical signals gathered from sensors connected to the patient’s body to monitor the heart’s activity [181]. In particular, we focus on the ‘12 Lead ECG’, which makes use of twelve signals gathered in parallel from the human body [22]. This ECG technique nowadays gives the best image of the heart but of course on the expense of having to deal with much more data in parallel. From a technical viewpoint, real-time processing of ECG data would allow a finer-granularity analysis with respect to the traditional eyeball monitoring of the paper ECG readout [77][78]. Eventually, warning or alarm signals could be generated by the monitoring device and transmitted to the healthcare center via telemedicine links, thus allowing for a prompter reaction of the medical staff [73][74][79]. In contrast, heartbeat monitoring and data processing are traditionally performed at the hospital, and for long monitoring periods a huge amount of collected data must be processed offline by networks of parallel computers [77][78]. New models of healthcare delivery are therefore required, improving productivity and access to care, controlling costs, and improving clinical outcomes. This poses new technical challenges to the design of biomedical ECG equipment, calling for the development of new integrated circuits featuring increased energy efficiency while providing higher computation capabilities. In addition, looking into these challenges in the application, the available technology of MPSoC, and the proposed solution, we find ourselves facing a multidisciplinary problem with different layers (medical, SW, HW, and design space exploration). Therefore, we also find a need to pursue some methodology that minimizes the time for reaching the desired solution and that can help in running many iterations of testing and HW/SW Codesign in a structured manner, where previous knowledge (success and mistakes) is saved and used. Thus, the evolution of the solution is done within the time limits of design and allows for easier implementation guaranteeing acceptable performance.

The fast evolution of biomedical sensors and the trend in embedded computing are progressively making this new scenario technically feasible. Sensors today exhibit smaller size, increased energy efficiency and therefore prolonged lifetimes (up to 24 hours), higher sampling frequencies (up to 10 kHz for ECG), and often provide for wireless connectivity [15]. Unfortunately, a mismatch exists between advances in sensor technology and the capabilities of state-of-the-art heart analyzers [24][25][93]. They cannot usually keep up with the data acquisition rate, and are usually wall-plugged, thus preventing for mobile monitoring. On the contrary, the deployment of wearable devices such as SoC devices has to cope with the tight power budgets of such devices, potentially cutting down on the maximum achievable monitoring period. In this thesis, we propose a wearable multi-processor biomedical-chip for electrocardiogram (MPSoC ECG biochip) paving the way for portable real-time electrocardiography applications targeting heart malfunctions. The biochip leverages the computation horsepower provided by many (up to twelve) concurrent DSPs and is able to operate in real-time while performing the finest granularity analysis as specified by the ECG application. In addition, in case of heart failure emergency aid should arrive in a period of few minutes from the time when the heart failed, otherwise brain damage may occur. Hence, real time analysis must be done in few seconds to allow the alarm signal to reach the emergency aid team, which should act immediately. The biochip system builds upon some of the most advanced industrial components for MPSoC design (multi-issue VLIW DSPs, high-throughput system interconnect and commercial off-the-shelf biomedical sensors), which have been composed in a scalable and flexible platform. Therefore, we have ensured its reusability for future generations of ECG analysis algorithms and its suitability for porting of other biomedical applications, in particular those collecting input data from wired/wireless sensor networks [71][83][134].

In addition, in order to fit this solution within the big picture for rescue at the point of need, we try to look at the general scenarios, where our biochip may fit. Hence, we look at the cases when more than one of the biochips is needed to be networked on a larger scale network in order to run applications in parallel. Therefore, there was a need to look not only at the performance and protocols of communications on the chip but also to look at inter-chip communication. We discuss this point further in the chapters that follow; however, we give a brief overview about a general picture of a scenario, where the biochip may be used for patient monitoring at the point of need, in Figure 3.

In the following chapters, we discuss our investigation that goes through all the steps of background knowledge, application design and implementation, design process from application functional specification to hardware modeling and optimization, and finally the induced methodology. System performance has been validated through functional, timing accurate simulation on a virtual platform. We point out the need for simulation abstractions matching the application domain. A 0.13µm technology-homogeneous power estimation framework leveraging industrial power models is used for power management considerations [3][123].

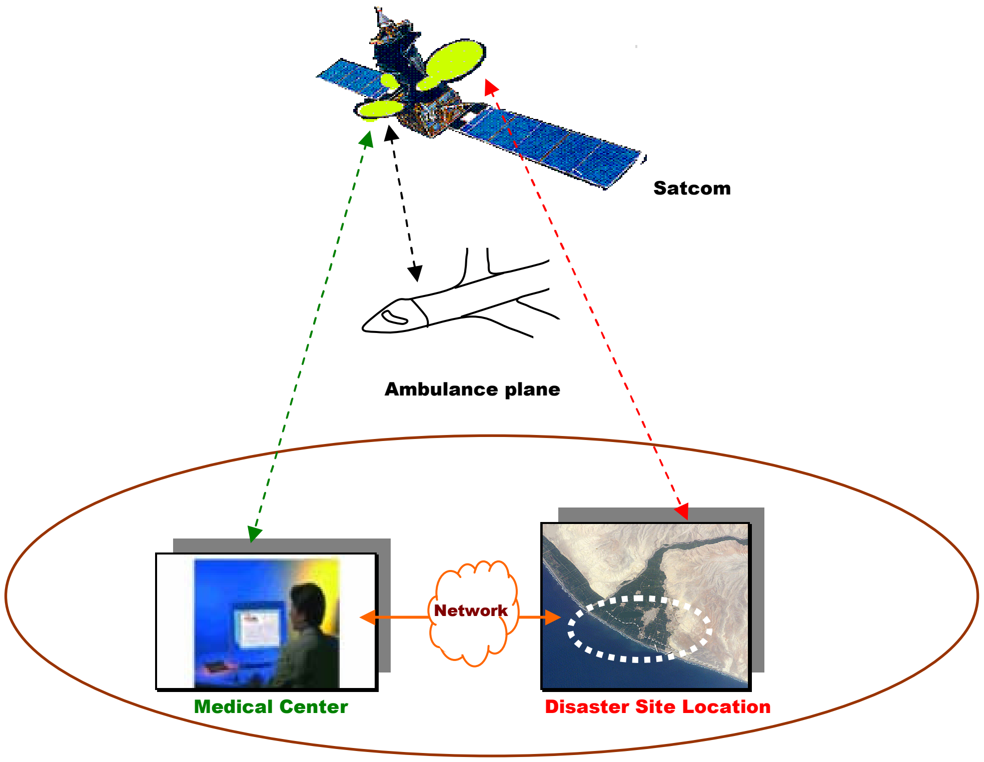


Figure 3. Healthcare solution scenario for aid at the disaster point (point of need). A scenario where the small size, small weight, and high computational ability of MPSoC provide a potential for applicable SW solutions for healthcare implemented on an MPSoC, e.g. heart analysis at the disaster site using MPSoCs that are connected via a communication link to a healthcare center. The connection can be via a satellite network or other communications network. Satellite aid remains intact and covers large areas when ground disasters occur.

The thesis, thus, presents the process of: (i) application understanding, (ii) implementation of the application with software functional specification, (iii) optimization and parallelization, and (iv) hardware design space exploration results. This leads to the final performance and energy-optimized solution. Last but not least, we come up with a general abstraction from the specific experiences we gathered over the design and testing processes [76]. This general case can serve as a methodology to follow for a faster hardware/software (HW/SW) Codesign for specific-purpose high-computational biomedical applications.

1.2. Contribution

The main contributions in this thesis stem from my interest in finding practical solutions for some problems that can be supported by the use of ICT solutions. We also developed some algorithms and performance analysis approaches (presented in Chapters 3, 4 and 5). A major achievement that leads all the contributions is that I searched for *fitting an application on MPSoC*, where MPSoC can be a grand aid and where the application has the possibility to make an impact. I can clearly state that the *choice of the 12 lead ECG medical application* and the *use of MPSoC and NoC for such an application* was my first idea to start the rest of the work. Moreover, throughout the course of work, I have managed the work of all the groups (internationally and locally) that were involved, and I have coordinated the work between the multidisciplinary groups. I made all efforts needed to get all groups in contact and update all on what was new, although some groups have been at far distances. Hence, *coordinating the investigation in an international dimension* and the persistence to solve many concurrent problems was part of the work I did. During the investigation, I was always discussing my new ideas and results with my advisor, who was judging the ideas, telling me on which ones to focus, and also giving me advice about where to look for more information. Moreover, I have always been contributing ideas to make the design flexible to be built from the start in a manner that suits future needs and can survive in a scalable manner, hence ensuring its reusability for the coming generations of MPSoC algorithms and other biomedical applications. Some of those ideas are published in DATE'06 [71][75][83][134].

In this thesis there are three major scientific/technical contributions, which are:

(1) The design and implementation of a new algorithms and techniques to analyze ECG signals in real time using state of the art multiprocessor SoC design, and also the transmission of the results of these signals within a larger

communications network and converge to a solution within the allowed time span. This first contribution is discussed in detail in **Chapter 3 and Chapter 4** of this thesis, and it is published in detail in the *ACM SIGMICRO Computing Frontiers Conference 2006 (CF'06)* [78], which was selected as one of four **CF'06 Best papers** to go to the *Lecture Notes on Computer Science (LNCS) Transactions on HiPEAC* [77]. Part of the ideas about a global solution and how to technically fit the MPSoC and NoC solutions for ECG analysis in a global network as well as performance analysis are also published at the *IEEE INFOCOM 2005* [74], the *IEEE INFOCOM 2006* [73][82], *DATE'06 Workshop* [75], *IEEE MWSCAS'05* [70], the *REAL Wireless Sensor Networks Conference* [72], and *the European Space Agency (ESA) Satlabs DVB-RCS Symposium 2005* [79]. Moreover, in order to fit the solution in some wireless networks, we needed some performance analysis for these networks, and that work was mainly done by M. Saleh from the KTH team, and my contribution was in helping him (as a second author) in performance analysis algorithms in [130][131]. In the aforementioned publications, the algorithms are created by me, and the choice to use NoC and MPSoC was mine, and accordingly I contacted multidisciplinary groups; The Lebanese group from the American University of Beirut (AUB) [78], helped me in implementing and coding the algorithm on PCs and fitting it to MPSoC. The simulation for NoC was programmed by me. The Bologna team and I worked thoroughly together on the design and testing of the MPSoC architectures as well as the design space exploration. There were some changes always needed to be made in software (SW) or hardware (HW) to fit the HW/SW Codesign to reach the best performance, and in this, when HW changes needed to be made, discussions were between me and the Bologna group, whose aid and support was very valuable. Moreover, the Bologna group supplied and supported us with the design and test emulators, where we tried the experiments and analyzed the results. When a change in SW was needed to be made in order to fit the HW/SW Codesign in some iteration, the decision and implementation was basically taken and done by me, and the Lebanese AUB team helped often in the implementation. The Karolinska team (biomedical team) helped us with the analysis of the biomedical layer results and providing me with some medical data and information, which I had to understand and analyze, and then I had to carry the results of the medical analysis to the other groups in a simplified manner. The CRPSM Space Research Center group from the Space Center in Italy provided me with information about specific interests and needs in performance and requirements for the biochip for astronauts' health requirements. Moreover, in the IEEE INFOCOM publications[73][74][82], DATE'06 Workshop publication [75], IEEE MWSCAS [70], ands ESA Satlabs Symposium'05 [79], another contribution is the design and performance analysis of the protocols/algorithms for the NoC autonomous system in order to be able to use multiple MPSoCs or NoCs in the same global solution.

(2) The development of a HW/SW Codesign platform for specific performance requirements that suits high computational applications on MPSoC; i.e. a contribution in the Design Space Exploration for a Specific-Purpose Multiprocessor System on Chip, which respects real-time performance requirements for biomedical applications. This platform, which includes the design of a suitable multi-processor architecture and the performance evaluation, is described and investigated in **Chapter 5** of this thesis, and its contributions are published in *DAC'06* [81], where the work was nominated for the **DAC'06 Best paper Award** as paper 9.1 (*http://web.imit.kth.se/~ikhatib/phd/publications/dac06-best-paper-candidate-page.jpg*), and also accepted at *the ACM Transaction on Design Automation of Electronic Systems 2008 (ACM TODAES'08)* [80]. In the aforementioned publications, the algorithm was designed by me, and the Lebanese team helped implementing it on the emulation platform, which was supplied by the Bologna group. The Bologna group and I made the HW/SW Codesign, testing and analysis. Decisions in HW were made by the Bologna group and me. Decisions in SW, were mostly made by me, after discussions. I served in this as a link between SW, HW, and biomedical (Karolinska) teams, where I was the only team member that could understand and link all information and transmit it in between all group members. The same linking between the teams from a HW/SW and global network fitting Codesign In the INFOCOM publications[73][74][82], DATE'06 Workshop publications [75], IEEE MWSCAS [70], ands ESA Satlabs Symposium'05 [79], my contribution is the

algorithm and the implementation of the algorithm, while the CRPSM group from the Space Research Center in Italy, provided information about space HW performance and requirements for the biochip. The Karolinska group provided some disease information and performance analysis for the biomedical applications. In the performance analysis part, I contributed in the patents [84][85], which I discuss the possibility to use them in the future in order to ensure better performance for on-chip networks and design the on-chip network in less time while knowing an exact quality metric for the on-chip network (these patents are briefly mentioned in the future work section). More precisely, the patents may be used in order to test the performance of each component/core on the chip before connecting all the cores on the chip, and then one can estimate the performance of the on-chip multicore interconnection before implementing the real chip. This would minimize costs and estimate the overall performance before spending time and money on the real chip. In addition, I am the first author on all papers for this contribution, i.e. I also did the coordination of the group work and writing.

(3) A Methodology for HW/SW Codesign for application-specific multicore system-on-chip. This methodology is discussed in **Chapter 6**, and it was published in the *CODES-ISSS 2007* [76]. In this work, the methodology is my immediate contribution. The Bologna group, has supplied the platform for simulation and we together made the MPSoC design and experiments for the results in the CODES-ISSS publication [76]. In addition, I am the first author on all papers for this contribution, i.e. I also did the coordination of the group work and writing.

In addition, in all the aforementioned publications in the contribution section [70][71][72][73][74][75][76][77][78][79][80][81][82][83] and the two patents [84][85], except in the two publications referred to as [130] and [131], I am the first author, where I was managing and coordinating the work of all groups, besides the technical and scientific contributions described above.

1.3. Organization of the Thesis

The rest of the thesis is organized as follows: *Chapter 2* reviews the background knowledge of related fields and research, *Chapter 3* presents the algorithms that we created for ECG analysis, *Chapter 4* presents the inter-multicore-chip communications within a larger network in order to fit the multicore chip in a general solution and also look into the scenario when more than one bio-multicore-chip is needed for the overall solution, *Chapter 5* discusses the HW/SW Codesign platform and the performance analysis results, *Chapter 6* explains the methodology that we induced from the analysis, and *Chapter 7* concludes the thesis. Last but not least, *Chapter 7* examines open issues, presents ongoing research, discusses already available work that may be used in the future, and lists ideas that are already for future work.

This thesis is a volume discussing research in a multidisciplinary environment (biomedical discipline, networking and on-chip multiprocessor networks, software discipline, and hardware discipline). Hence, I try to be as direct to the point and as concise as possible in order to be able to link the different disciplines for all readers (that may be specialists in one or more disciplines) in a way that is easy for the reader to grasp the ideas and most motivating to continue reading.

References are listed alphabetically and numbered accordingly using Arabic Numerals. Within the thesis text, numbers between *parentheses* correspond to *equations*.

Chapter 2

Background

"The knowledge of anything, since all things have causes, is not acquired or complete unless it is known by its causes."

Ibin Sina- also known as "Avicenna" (973-1037A.D.)

This chapter reviews the necessary knowledge for the chapters that follow and presents the causes of the problem we focus on solving; namely, human-heart electrocardiogram real-time analysis. Hence, this chapter presents the different subjects that form the background of this multidisciplinary work. Section 2.1 deliberates the medical and biomedical background of the electrocardiogram set of signals, which we use later (in Chapter 3) in order to design our algorithms and application. In Section 2.2 we present previous work by addressing two points in two subsections: (i) subsection 2.2.1 elaborates on the previously designed algorithms/software, and (ii) subsection 2.2.2 investigates the related hardware systems in the literature. Section 2.3 presents the need and positive effects of pre-filtering of sensor-recorded data before the analysis stage (the analysis stage is discussed in Chapter 3). In section 2.4, SoC, MPSoC, and NoC architectures are briefly introduced. Section 2.5 introduces some state of the art work on methodologies.

2.1. Biomedical Application Background

In order to fulfill our goal of monitoring and analyzing the human heart activity in real time, we find ourselves turning towards a huge set of random signals and large amount of information to comprehend before any decision on further steps is taken. To contribute to the solution, we have to look at the causes of the problem, i.e. we need background knowledge about the signals that are used nowadays to monitor the heart. In this respect, we briefly introduce the electrocardiogram (ECG) signal, which is an electrical recording of the heart activity that is used as a diagnosis tool by physicians and doctors since the last century to check the status of the heart [20]. For the purpose of designing an algorithm and code (SW), we have to differentiate between two biomedical signals: *sensor* signal and *lead* signal. Sensors on the body send *electrical signals*, and then every group of these *sensor-signals* are combined together to form what we call a *lead-signal* [22]. Basically, a lead is the voltage difference between two points on the human body [77]. In this respect, what we refer to in this thesis as an ECG signal is nothing but the signal that comes from a connection referred to- in the biomedical field- as a "lead" (i.e. a *group of sensor-recorded signals*). The ECG-lead signal is a voltage signal in *mV* [22], and it is of direct interest to our work. We can think of each lead like an electric camera that can take a view of the heart from a certain angle, hence, the more the cameras the better we view the heart shape and activity in its three dimensional space. Therefore, the more the leads the larger the information set (ECG data set) that we can get for the heart activity.

2.1.1. The Twelve Lead ECG

One of the latest techniques deployed for monitoring heart activity is the 12-lead ECG, which makes use of data coming from twelve ECG leads. This method relies on only nine sensors placed on the patient's body [78]. Figure 4 shows the nine sensors as: (i) three points on the limbs labeled as RA, LA, LL, and (ii) six points on the chest labeled as V1, V2, V3, V4, V5, and V6. From these nine sensors we get 12 lead signals [78]. The leads are divided into three groups: bipolar limb leads, chest leads, and augmented leads. The limb leads use the three main sensors (Figure 4) that are distributed together with a ground point as follows:

- RA: one sensor placed on the right arm;
- LA: a second sensor placed on the left arm;
- LL: a third sensor placed on the left leg;

and

- G: Ground on the right leg (RL) is connected by only a wire to be used as ground for the interconnected sensors.

As discussed earlier, physicians do not use sensor data directly, but rather use lead data. Hence, interconnecting RA, LA, and LL as known in Figure 4 gives the three leads (Lead I, Lead II, and Lead III) that are shown in Figure 4 on the red dotted lines and inscribed in a circle.

By using the three limb leads, physicians can use a method known as the 3-lead ECG, which suffers from the lack of information about some parts of the heart but is useful for some emergency cases to have quick analysis [77]. In this respect, medical doctors require more information and thus more ECG leads. Although we can get more interconnected leads from these three sensors (the augmented leads aVR, aVL, and aVF as shown in Fig.1), but these are not enough to give a view of most heart activity. Hence, six more sensors (V1-V6) are added on the chest (Figure 4). The corresponding chest lead voltages from V1 to V6 are measured with respect to Ground (G) on the right leg (RL).

In other words, using nine sensors and interconnecting them for the 12-lead ECG gives twelve signals known in biomedical terms as: Lead I, Lead II, Lead III, aVR, aVL, aVF, V1, V2, V3, V4, V5, and V6 (Figure 4).

The 12-lead ECG produces huge amounts of data especially when used for a long number of hours. Physicians use the 12-lead ECG method, because it allows them to view the heart in its three dimensional form; thus, enabling detection of many abnormalities that may not be apparent in the 3-lead technique. Figure 5 shows an explanatory example of a typical ECG signal. The most important points on the ECG signal are the peaks: *P, Q, R, S, T,* and *U*. Each peak is related to a heart action that is of importance to the medical analysis. Figure 6 shows real recorded signals from 12-leads, which are printed on the usual pink eyeballing paper. This printout is the classical medical technique used for looking at ECG signals, and it is still used in hospitals and healthcare centers.

However, the eyeballing paper printout makes the check of the different heart peaks and rhythms difficult and inaccurate due to its dependence on the physician’s eyes (see Figure 6 and Figure 7). On the other hand, when using digital recording (that allows digital filtering), we can determine the peaks more accurately [80]. Consequently, we can use digital computing to process the sensed data.

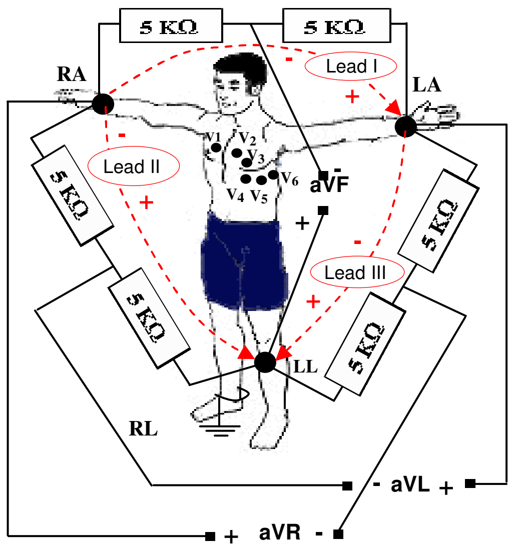


Figure 4. Interconnections of the 12-lead ECG technique; RA, LA, LL, & RL (G) are the right arm, left arm, left leg, and right leg sensors. RL is grounded (G). The augmented leads are aVR, aVF, and aVL. The chest leads are V1, V2, V3, V4, V5, and V6.

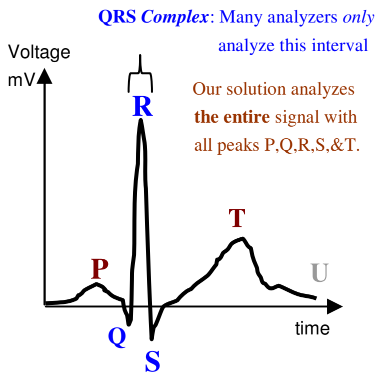


Figure 5. A typical ECG Signal; Lead I for a normal heart.

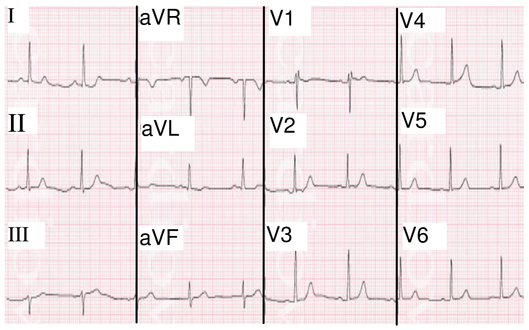


Figure 6. Complete paper readout of the 12-lead ECG, which is not accurate to see peaks nor easy to read for long recordings.

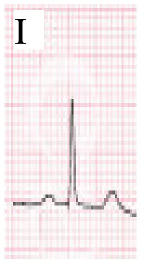


Figure 7. Zoom in Lead I from the paper printout of the 12-lead ECG signals of Figure 6. This zooming shows how reading the signal is dependent on the physician's eyes, hence makes analysis more difficult.

From a medical diagnosis viewpoint, there are normal medical ranges for the inter-peak time intervals, and every combination of different inter-peak intervals proves a type of heart illness or malfunction [22]. The most important of the peaks is the R peak, which refers to the largest heart blood pump. That is why many investigations and products try to detect only the R peak. However, that does not give a view of the whole heart activity. The whole heart activity can only be viewed if the whole Lead signal is analyzed with all the peaks.

Many ECG analysis methods use the three peaks Q, R and S, and the corresponding intervals between these three peaks. This interval from Q to R to S is known in biomedical terms as the QRS complex. Figure 5 shows the QRS complex. Many methods nowadays (e.g. in the research [116] and market [60]) use only this complex to analyze heart activity, while our proposed solution (discussed in Chapter 3) looks at all peaks and inter-peak intervals.

2.1.2. Current Steps in ECG Recording and Analysis

Nowadays, when the patient needs monitoring for a long number of hours, some devices like Holter(s) are used for recording while being carried on the patient's body. The recorder is connected to the ECG sensors on the patient body as shown in Figure 8.

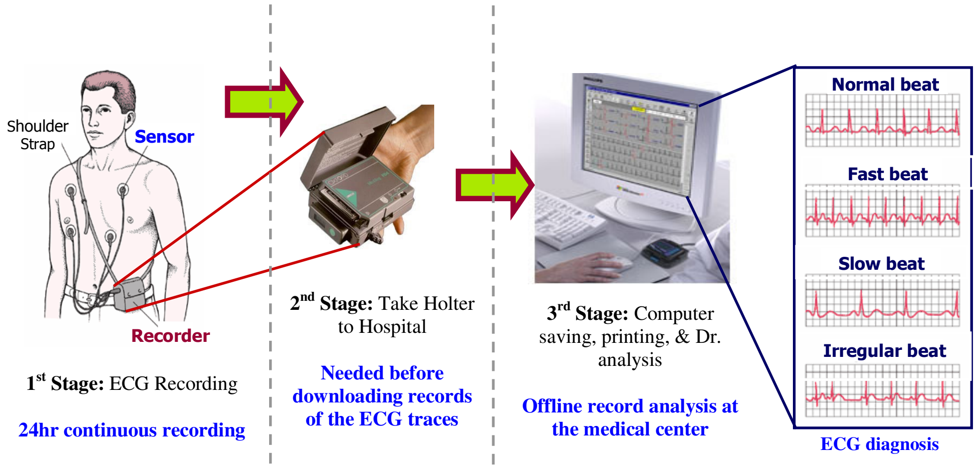

Shoulder Strap
Sensor
Recorder
1st Stage: ECG Recording
24hr continuous recording
2nd Stage: Take Holter to Hospital
Needed before downloading records of the ECG traces
3rd Stage: Computer saving, printing, & Dr. analysis
Offline record analysis at the medical center
Normal beat
Fast beat
Slow beat
Irregular beat
ECG diagnosis

Figure 8. Current methods to record ECG data for long number of hours and then carrying the recorded data to the healthcare center for printing and analysis.

We can divide the process of recording and analyzing ECG data into three stages:

- The *first stage* is during the time when the recording device is carried on the patient's body for many hours and running continuous recording.
- The *second stage* starts when the recording device capacity is full and/or its battery is no longer charged, then the recorder is taken by the patient to the hospital or healthcare center.
- The *third stage* includes connecting the recorder to the computers at the healthcare center/hospital, saving the data (which consumes some significant time), looking at

the data, choosing parts of the data to analyze, printing chosen data for the doctors, and finally analyzing the printed information.

Such a long process suffers from many delays. In addition, although it records the heart activity for a long period of time (up to 24 hours), if any heart malfunction occurs while the patient is wearing the Holter device (ECG recorder), there would be no real-time knowledge of such a malfunction at the hospital and a heart attack may occur, for instance. Hence, the remote and real-time monitoring through a communication link could constitute an interesting but challenging solution. In such a remote real-time solution (e.g. similar to the one shown in Figure 3), we can utilize cheaper devices, get information about the heart immediately, be proactive, and allow for operating on the patient more quickly. We discuss this solution and its requirements and challenges in the chapters that follow.

2.2. Previous Work

Since this thesis comprises multidisciplinary work, we survey previous work in these disciplines, and we divide the literature review to two parts: in subsection 2.2.1 we present previous work on the application layer (ECG analysis, algorithms and SW), and in subsection 2.2.2 we present previous work on the HW designs for ECG analysis. ECG monitoring and analysis garner significant attention in the research and in industry. Many companies have investigated and developed commercial solutions (e.g. [60]). Therefore, we study some of the most used (and applicable solutions), and we present them in what follows.

2.2.1. Software Algorithms

A relatively large number of algorithms (and corresponding SW implementations) have been designed for ECG analysis. We list some of the algorithms that have been well recognized and used. We, then, choose the main algorithm used nowadays (namely Pan Tompkins) in the ECG market and in medical centers/hospitals in order to compare it with our algorithms (in Chapter 3). Some of the popular ECG algorithms are the following nine algorithms: Padova, Nagoya-Fukuda, IBM Medis, HP (Agilent), Glasgow, GE (Marquette), Means, Hannover, and Louvaine (Louven) [47][65][91]. The evaluation of the nine aforementioned popular ECG algorithms compared to a standardized database of

ECG tracings [91] show that the original Louvaine algorithm had the best total accuracy of all the algorithms. These algorithms use QRS complex analysis (Figure 5).

From the viewpoint of our general solution and the big picture of a biochip fitting into a larger scale solution like the one shown in Figure 3, we look at some algorithms designed for ambulatory patient monitoring. However, these ambulatory algorithms all run the analysis on the QRS complex only, i.e. they do not provide for full ECG analysis. In what follows we briefly present these algorithms

2.2.1.1. Algorithms for Ambulatory Patient Monitoring

The main ambulatory algorithms analyze the QRS complex, and they are: Peak Detector, Single-channel QRS detector (SQRS), and the Length-Transform based Single-channel QRS detector (WQRS). We briefly present them below:

- The *Peak detector algorithm* is applied directly to the ECG data using a 167ms window that are divided into three 55.6ms segments. Hence, the maximum detectable beat frequency is 360 Beats per Minute (BPM). The algorithm scans the whole range searching for the maximum value. The algorithm checks whether the maximum value is on the central segment and higher than a certain threshold. If so is the case, then the position is marked as a beat, whose amplitude is saved as the new threshold. Hence, thresholds are updated.

- The *SQRS algorithm* is a *Single-channel QRS detector*. It is based on length transform, and it uses a Finite Impulse Response (FIR) filter as an approximation to the slope of an ECG signal [46][185]. Using a variable threshold, it detects and identifies QRS complexes from artifacts. The C code was extracted from PhysioNet [143] and adapted to handle data streams. If the filtered signal is greater than a threshold, the time is saved and the algorithm enters a decision phase. Two to four of these detections within 160ms of each other indicate that a normal beat was identified. More than 4 detections within 200ms of each other indicate an artifact. After a decision has been made, the algorithm is reset. Once every 2 seconds, if there is no detection, the threshold is reduced by 1/16th. If there are more than 4 detections, the threshold is increased by 1/16th. Every time a normal beat is detected, the threshold is recalculated, asymptotically converging to 1/4 of the maximum filter output obtained so far.

- The *WQRS algorithm* is a *Length-transform based single-channel QRS detector*, but it is based on Length Transform. It is originally obtained from PhysioNet [143] and adapted to handle data streams. It is based on the Length Transform of the ECG [46]. The algorithm output (after running the transformation) is compared also to a threshold. Then it is tested for a rising slope of the Length Transform.

The Peak Detector algorithm proves to be the fastest of the aforementioned three algorithm [46].

2.2.1.2. Pan Tompkins

The Pan-Tompkins algorithm is a QRS detection algorithm responsible for locating the R-peaks in the ECG signal and thus calculating the heart period. This algorithm gained popularity because it used the three known types of QRS detection: linear digital filtering, non-linear transformations, and decision rule algorithms.

This algorithm uses first an analog filter to band limit the ECG signal at 50Hz. Then, an A/D converter is used to sample the signal with a sampling frequency f_s=200Hz. After that a band-pass filter is used to remove high-frequency noise, P-waves, T-waves, and other artifacts; to realize this filter a cascade of a low-pass filter and a high-pass filter with a cut-off frequency of 15 and 5 Hz respectively is used. The transfer functions of the low pass filter and the high pass filter are in (1) and (2), respectively.

$$H(z) = \frac{\left(1 - z^{-6}\right)^2}{\left(1 - z^{-1}\right)^2} \tag{1}$$

$$H(z) = \frac{\left(-1 + 32z^{-16} + z^{-32}\right)}{\left(1 + z^{-1}\right)} \tag{2}$$

This is the result of subtracting a first order low pass filter from an all-pass filter. The resulting pass-band is a 3dB pass-band from about 5-12 Hz that maximizes the QRS energy, whose band is from 5 to 15 Hz. After this stage, we have removed noise, improved the SNR and thus lower thresholds can be used to enhance the detection process. We then pass the resulting signal through a first local peak detection algorithm, which identifies and marks all the peaks found in the signal. This algorithm uses a set of two adaptive thresholds T1 and T2 (T1=2.T2) to select candidate QRS complexes. The main reason behind the use of 2 thresholds is that when a certain limit time passes without locating any R-peak higher than T1, we implement a search-back routine with a new threshold T2 that is half the amplitude of T1. T1 and T2 are constantly updated based on the amplitude of the peaks of the ECG input. After this detection stage, we move further in processing the signal. First, we take the derivative of the signal, a non-linear transformation which contains information about the slope of the QRS, is calculated. The derivative used is a five-point derivative with the transfer function in (3).

$$H(z) = \frac{1}{8T}\left(-z^{-2} - 2z^{-1} + 2z^{1} + z^{2}\right) \tag{3}$$

Then we have the squaring stage which intensifies the slope of the frequency response of the differentiated signal and thus has its role to help detect false peaks like the T waves. The squaring function squares the signal point by point according to the operation in (4).

$$y(nT) = [x(nT)]^2 \tag{4}$$

After squaring we move the window integrator, whose main function is to deliver information about the width of the QRS complex and other waveform features. It is calculated using (5).

$$y(nT) = \frac{1}{N}\{x[nT-(N-1)T] + x[nT-(N-2)T] + ... + x[nT]\} \quad (5)$$

where N is the number of samples in the width of the integration window, and this value must be approximately equal to the widest QRS complex. For the given sampling frequency of 200Hz, N=30 is used.

After getting the result we pass this processed data through the detection algorithm already described above and compare these new results with those that we got in an earlier stage. Only the common QRS complexes between the 2 detections are taken as R-peaks. The output is a stream of pulses indicating the locations of the QRS complexes. The adaptive threshold that takes place in this algorithm is done based on (6), (7), (8), and (9).

$$SPKI = 0.125PEAKI + 0.875SPKI \quad (6)$$

$$NPKI = 0.125PEAKI + 0.875NPKI \quad (7)$$

$$THRESHOLD\_I1 = NPKI + 0.25(SPKI - NPKI) \quad (8)$$

$$THRESHOLD\_I2 = 0.5THRESHOLD\_I1 \quad (9)$$

where we have: PEAKI the overall peak, SPKI the most recent running estimate of the signal peak, NPKI the most recent running estimate of the noise peak, THRESHOLDI1 the first threshold applied, and THRESHOLDI2 the second threshold applied. In case the QRS complex is found using the second threshold, we use (10).

$$SPKI = 0.25PEAKI + 0.75SPKI \quad (10)$$

These equations are applied to the output of the moving window integrator. The same equations are applied to the output of the band-pass filter for the first detection but with the letter “I” in the variables name replaced by “F”, i.e. SPKF in stead of SPKI.

The time limit that we mentioned when talking about the search-back routine is called the RR interval (interval between 2 R-peaks). This quantity is also adaptive in the algorithm and is the average of 8 past RR intervals. We have 2 running RR measures as shown in (11) and (12).

$$RR\_AVERAGE = 0.125\left(RR_{n-7} + RR_{n-6} + ... + RR_{n}\right) \tag{11}$$

where RR_n is the most recent R-R interval

$$RR\_AVERAGE\_2 = 0.125\left(RR'_{n-7} + RR'_{n-6} + ... + RR'_{n}\right) \tag{12}$$

where RR'_n is the most recent R-R interval that fell between the acceptable low and high RR-interval limits in (13), (14), and (15).

$$RR\_LOW\_LIMIT = 0.92RR\_AVERAGE2 \tag{13}$$

$$RR\_HIGH\_LIMIT = 1.16RR\_AVERAGE2 \tag{14}$$

$$RR\_MISSED\_LIMIT = 1.66RR\_AVERAGE2 \tag{15}$$

After an RR MISSED LIMIT time has passed and no peaks were detected, we implement the search-back technique using T2 (the second threshold). When an RR interval is detected to be less than 360ms and the maximal slope occurring during the detected QRS is less than half of the QRS waveform that preceded it, the peak identified is considered as a T-wave and not a QRS complex.

We implemented the Pan-Tompkins algorithm for testing and validation. The Pan-Tompkins algorithm has a good performance, but we were able to find some cases where a high percentage of error was detected, we discuss these tests in Chapter 3.

2.2.2. Hardware Systems

We are not aware of any solution in the research or the commercial markets that is composed of a single-chip real-time analysis solution for full 12-lead ECG analysis that is able to estimate the heart period independent of the peak signals, and that can diagnose all the peaks: P, Q, R, S, and T and their inter-peak intervals to result in disease diagnosis. Most of the work done involves only recording huge amounts of data in large storage media and then analyzing the stored data, but not allowing the ease of patient mobility. Most of the time, the patient has to be confined to a bed for a number of hours (could be for a whole day) [77]. Some commercial solutions are only capable of concluding if the heart beat is normal or abnormal but can not specify the period nor could they diagnose the disease. Other real time solutions available in the market, in healthcare institutes, and in research organizations, are only capable of sensing and transmitting ECG data [18] to: either a local machine [17][113] or to a distant healthcare center [23][30] [31][34][35][97][99][139][156][172]. In both cases, the work that is executed involves checking if the heart beat is healthy or unhealthy without analyzing the disease and not in real-time. Moreover, the commercial solutions under study (e.g. [21]) do not look into the parallelization of the ECG analysis into multiple cores, so to speed up processing.

We divide the previous work that we investigated on SoC for ECG to three categories: (i) ECG analysis on SoC, (ii) ECG data transmission to remote centers, and (iii) simple implant circuits.

For the first type that runs ECG analysis on SoC we study mainly the work presented in [1][13][28][60][116][159][169][173]. In this category, we find one common point from an output point of view, which is that they do not run full 12-lead analysis. In addition, the work in [13], [116], and [159] can only detect if the heart beat is normal or not normal. From a HW viewpoint, the cited work in this category uses either one chip for each ECG lead or uses a PC. In Chapter 5, we discuss in detail the difference between these cited works and our solution. However we briefly discuss these works in what follows.

In [1] the authors propose, from an output point of view, a system that only distinguishes healthy from unhealthy patients. The work proposes Heart Rate Variability (HRV)[1] analysis with the use of wavelets (see Appendix 3). The HW system is based on four units: (i) a processing unit (prepares the input signal for analysis), (ii) a unit that manages control signals, (iii) a wavelet computation unit, and (iv) a wavelet coefficient evaluation unit. A very interesting HW issue for this work is that the system is able to calculate the HRV without the help of a PC. All computations are done on chip. The system takes the R-R

[1] Heart Rate Variability (HRV) is related to the beat-to-beat alterations in the heart rate. A decrease in HRV is a measure to predict future heart (coronary) abnormalities. Healthy patients have higher variability than unhealthy patients [168].

intervals of the cardiac signals as inputs and then finds the wavelet coefficients using the Discrete Wavelet Transform (DWT). DWT has four main advantages over the spectral analysis (FFT): (i) the analysis is done in time domain rather than in the frequency domain, (ii) simpler computations, (iii) cost effective, and (iv) simpler hardware.

In [13] the investigation tries to accurately measure HRV. The aim is to determine the QRS with high accuracy, low timing given a low power consumption and small sized device. The proposed algorithm is based on a host computer and a detector unit. The detector is connected to the host computer using the RS232 port, the serial port. The host computer is used to configure the detector device and to retrieve results. The detector unit is based on a DSP microcontroller; the Motorola 68HC16 DSP (contains an ADC inside it). The QRS detection algorithm uses optimized pre-filtering with a matched filter (MF)[2] and dual edge threshold detection. Pre-filtering attenuates various noise components. The decision device is a real time decision, without search back methods and without buffering of previous data.

The authors in [28] propose a physiological monitoring system mainly for space and terrestrial applications. The work uses only 2 ECG records continuously. Data can be either streamed to a base station through a Bluetooth link or stored on a flash memory (of 32 MB) and then downloaded to a PC using the serial port. The device uses 3 AAA batteries. Two important commercial systems are the Micropaq (from Welch Allyn) and the ApexPro (from GE medical systems). Other commercial devices, for example, are PCMCIA based. The device has small dimensions and weight, it uses PIC16LF877, processor speed 7.3MHz, 32MB of onboard flash memory. Device configuration and data download are done entirely from the PC.

The design in [60] is one of the commercial solutions nearest to our work, where eight (8) probes from sensors are directly input to one chip, which runs one lead analysis. The solution is DSP based with sequential operation. It interfaces to a SCI (serial communication interface), keyboard, external memory, display, and modem. It enjoys an 80MHz Clock frequency, 64KB program flash memory; 512KB program RAM, ADC with 8 channels. The maximum ECG sampling frequency they use 500 Hz.

[2] A Matched Filter (MF) is a linear filter that maximizes the output Signal-to-Noise (S/N) ratio.

The most interesting part in [116] is that the authors work on a SoC design for ECG processing. They present an algorithm for finding the R-Peak by using an adaptive threshold after the squaring and moving window integration operations. The heart rate is then dependent on the adaptive threshold and can be error-prone. The algorithm works under normal heart activity but does not guarantee abnormal cases. This work is one of the nearest to ours and we compare it in more detail with our work in Section 5.7.

Paper [159] presents the SW, which can be implemented on DSP, ASIC or FPGA, i.e. off the shelf IC using limited memory resources. C is the algorithm implementation language. The coder/decoder is capable to sample at 16 KHz, and they can do 16 bit quantization. The sampling rate is set to 2 KHz in order to easily have the same data that was captured and down-sampled to 1 KHz. The paper uses a 1 GHz PC.

The investigation in [169] proposes ECG/Holter portable equipment. The system is built around the Motorola MC68L11 microcontroller with clock frequency of 8 MHz, 8bit ADCs and 100Hz sampling rate. It uses the telephony networks as the underlying transmission infrastructure**.** The microcontroller is used for acquisition of the ECG signal and transmission through the telephony networks. The system uses 3 electrodes to measure the heart voltages (i.e. 3 lead ECG). The ECG gets amplified by a factor of 100 and then frequency modulated (at 1900Hz) before transmission through the Network. At the destination, they have a demodulator to recover the signal. The system can operate in two modes: normal and Holter. It also has an additional mode; Test mode. This mode is used when new services need to be downloaded from the PC. Due to this test mode enhancement, the device can be personalized for each patient; i.e. different heart parameters and threshold values. They system uses large FLASH[3] memory for the Holter mode. The minimum size required is 512KB.

The research in [173] uses 2-lead ECG with a sampling frequency of 120Hz only. The paper proposes the use of CodeBlue [31] as the underlying wireless link between the sensors of different patients and the base station. The system continuously monitors and records information from different patients and sends them to a PDA or a PC. The solution transmits the ECG data at a maximum data rate of 76,800 bits per second. The interesting

[3] Flash memory is a non-volatile computer memory, which can be electrically erased and re-programmed.

part is that the authors use TinyOS[4]; an event driven operating system. Applications are written using nesC language; an extension to C.

Regarding the second category of previous work (HW for ECG analysis) we study the work presented in [23][30][34][35][97][99][139][156][172]. In fact, we can summarize the main interesting point in this category as work on ECG measurement enhancement. They all utilize some techniques and HW to transmit the ECG measured signal to a remote center for analysis. Hence the first category is more interesting for our work.

The research work in [95] and [96] propose ECG R-wave based systems that are pacemakers. Hence they are interesting to look upon for implants that use the ECG R-wave.

An interesting previous work is the one presented in [113]. The paper proposes an ECG ARM-based Bluetooth Wireless Biosignals recording and monitoring system. The system consists of two main units: ECG analog preprocessing unit and the DSP unit. The digital processing is done by an ARM7 chip that interfaces to a Bluetooth link to transfer data to a computer or a PDA. In the ECG analog board high accuracy sensors were used in addition to different traditionally used filters. The ARM7 contain 64 KB of RAM. A three layer logical stack models the CPU: the lower layer models the control and management of the ARM and the Bluetooth radio. The second layer is the Bluetooth stack and the last one controls the Bluetooth connections and the acquisition. An application layer was added at the top to combine different bio-signals. The CPU clock is 13 MHz. An adaptive algorithm was implemented to get the R-Peaks and the peak-peak interval.

2.3. Sensing and Filtering

ECG analysis requires three main phases: (i) acquiring the signals from the leads, (ii) filtering the lead-signals (each alone), and (iii) analysis. Firstly, the sensing phase requires an ADC in order to be able to have digital data for our digital filter. We use 16 bit ADC, because our analysis algorithm and ECG biochip are designed based on having 16-bit filtered data as input. We briefly discuss the filtering method we use as an essential part of

[4] TinyOS is an operating system that originated to target embedded systems for wireless sensor networks, and it is written in nesC (www.tinyos.net).

our proposed solution in Chapter 3, and then we discuss the biochip design that depends on this filtering step in chapter 5.

The high investment in sensor technology and biomedical research in general gave birth to biomedical sensors that have more advanced features than the commercially available ones just a few years ago. For instance, the nowadays sensors are characterized by prolonged lifetimes (up to 24 hours), and higher sampling frequencies (up to 10 kHz for ECG) [15]. Some sensor companies have produced wireless biomedical sensors in order to aid patient mobility. This advance in biomedical sensors faces a mismatch with biomedical heartbeat analyzers that still lack behind to cope with the huge amounts of data, the high rates, and the wireless features that modern sensors can provide. Data provided by biomedical sensors suffers from several types of noise: physiological variability of QRS complexes (The QRS Complex as shown in Figure 5), baseline wander, muscle noise, artifacts due to electrode motion, and power-line interference [41]. The Filter type interesting for our research work is the Infinite Impulse Response (IIR) filter, which is discussed in section 3.2.

2.4. SoC and MPSoC Background

In the past 30 years, an exponential rate of progress occurred in semiconductor technology. Much of this is a direct consequence to Moore's law[5] [62][64]. Moreover, we also see a convergence of computing and communications as Gorden Moore explains in [63]. This convergence is also affected by the fact that performance of the gates on chip has been improving. For instance, the delay of a simple gate decreases by 13% per year, halving every 5 years. This trend has been maintained for nearly the past 30 years by the semiconductor industry (Figure 9). Therefore, these trends lead logically to progress in interconnecting microelectronic blocks on the same chip.

[5] Moore's Law states that the number of devices that could be fabricated on a single chip is doubled every 18 months, hence it quadruples every 3.5 years [62].

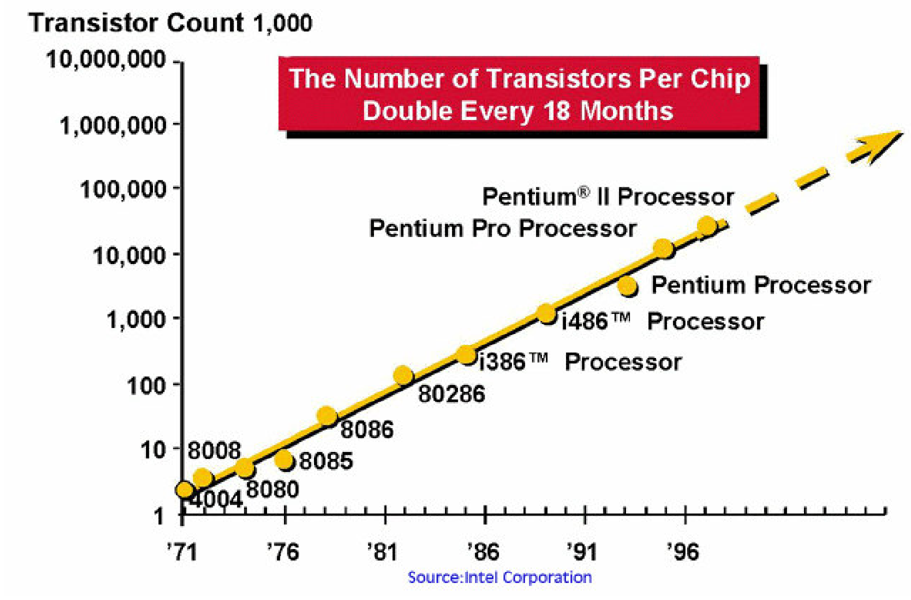


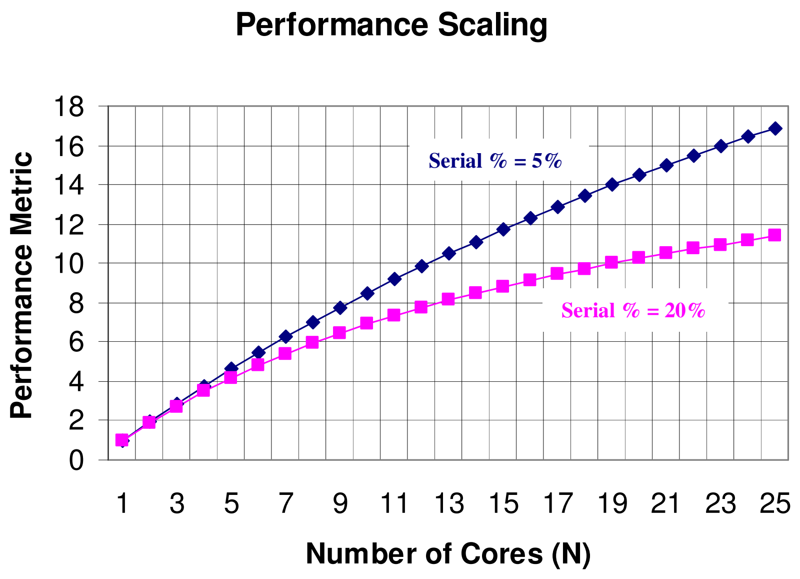
Figure 9. Moore’s Law: Number of transistors on chip double every 18 months.

In fact, the microelectronics technology allows us, nowadays, to have several connected cores on the same chip, which we refer to as System on Chip (SoC). SoC is defined as the integration of all components of a computer or another electronic system into a single Integrated Circuit (IC). Among the most advanced ICs are the microprocessors or "cores", which are available in computers as well as cellular phones, for instance. Digital memory chips and ASICs are other examples of integrated circuits that are used in ICT. The single chip may contain digital, analog, mixed-signal and Radio Frequency (RF) cores. Traffic of data on SoCs became an interesting topic as part of the communication vs. computation as we see some studies like in [56]. SoC has also paved the way to multicore trends under research and also in the market, where many processing cores can be connected in order to allow parallel processing [53]. For instance we see Intel Dual Core chips in nowadays PCs, and these chips have two processing units (cores) that speed up processing. This trend is also known as Multiprocessor System-on-Chip (MPSoC), where we can think of the MPSoC as a system of interconnected processors (computing cores) that can communicate [176], i.e. we go back to the point of convergence of computing and communications

[49][55][58][104][105][108][110][111][198]. Hence, bus partitioning and architectures become also very interesting [166]. That is what makes the MPSoC function like a microelectronics network, i.e. an on-chip network. Looking at NoCs and MPSoC in a similar network perspective was also touched in [33] and [55]. This leads to the hot topic in research and development now, known as Network on Chip (NoC) [2][4][5][6][7]. The NoC accommodates multiple asynchronous clocks. The NoC solution borrows the networking method to on-chip communications [8][9][10][11][12][26][27]. One of the NoCs we look at is the Nostrum NoCs [5][127] [137][165][198], which also has a designed simulator. Nostrum NoCs have dimensional mesh topologies [5]. More details on Nostrum are discussed in [42]. An application of JPEG encoding was studied on Nostrum in [191]. Performance analysis has been also a very important topic in the research since the success of SoC, MPSoC or NoC depends on having chips that can deliver a certain wanted level of performance.

The cited works in [32][42][43][44][45][199][200][201][202][203][204][205] discuss performance issues in such on-chip networks. The work in [154] also discusses how to migrate to tasks in MPSoC and runs a feasibility study for such migration on the chip technology trend. Moreover, the main reason for looking in multicore systems like MPSoC and NoC, is the fact that parallel processing can lead to better the performance. Figure 10 shows how the performance can increase in the system on chip when the number of cores increases. We also see many research papers on MPSoC that discuss MPSoC communication issues, i.e. gaining the dimension of a network to be analyzed from a QoS viewpoint, where QoS may include delay, communications protocols, losses, power consumption, and tradeoffs within these parameters; as we can see in [118][120][121][122].

An important set of research works are those running MPSoC simulation [106][107]. A very interesting topic is the design of on-chip networking protocols, types of data links and scalability in such networks, which have been addressed in many works like [129], [175], and [179]. Organizing Cache memory in SoCs is well studied in [194]. Synchronization in SoCs is a very interesting topic, and papers [145], [146], and [147] discuss these issues.



Amdahls law: Parallel Speedup = 1/(Serial% + (1-Serial%)/N)

Figure 10. Performance scaling vs. number of cores. The lower plot is for a serial percentage of 20%, and the higher plot is for a serial percentage of 5%. Hence, parallelizing SW is a key point for the success of multicore systems since performance is a key issue for success. N is the number of cores.

In the NoC trend we looked at many research works, and we can categorize the works there into: work on synchronous and asynchronous issues to get the on-chip network to achieve best performance and functionalities [36], performance analysis/evaluation [87][132][150][152][180] by looking at delays and packet loss [38][125][170][171], throughput and bandwidth [126], power consumption/management [37][61][66][67][88][112][135][174][177][190], NoC fault tolerance and error recovery [39][52][163][164], reliability [48][90], memory partitioning [50][51], layout [54][109], NoCs for real-time applications [100], core interfacing [138], protocols and routing techniques [178], traffic modeling [183], reconfiguration [186], and admission techniques [195][196]. NoC Designs, design flows, and methodologies are very interesting issues for our work, some of which are discussed in [98], [101], [102], [103], [114], [115], [149], [155], [167], [184], and [192]. A Method for designing NoCs with guaranteed QoS is discussed in [160]. NoC emulation is a crucial work for showing how NoCs can be useful for real applications [136][157]. The authors in [148] and [193] use FPGA for emulating a NoC. Interesting NoC simulators are published in [151] and [153]. Application-specific NoCs are considered to be the door to get NoC to see some light in the markets, and a study on application specific NoCs was investigated in [161][162]. The assembly and the instruction set for processors play a significant role in the performance and thus the success of NoCs. An interesting study is presented in [197]. The main reasons for the aforementioned progress discussed in this section are:

- *Scaling of the technology*, e.g. gate lengths have scaled to 0.18μm today from 50μm in the 1960's.
- *Cheap and simplified CMOS, where* the designers have adhered to rigid design methodologies and strategies that are enabling design automation, and that gives opportunities for a fast increase in the VLSI technologies.
- Fixed cost per wafer, which is more pronounced for small-volume products [92].

All in all, the current state of art in microelectronics allows us to deploy multi-cores on the same chip. However, this field still lacks some methodology in mapping real life applications onto such systems, and in many cases (like in NoC) still some standards are not defined. Hence, we do not find generally accepted protocols or algorithms that can serve this field in a way so that the mapping of applications (that are useful) can become a structured method. At the same time, the performance of multicore chips like MPSoC and NoC play a major role in the success of such technologies, especially when used to host real life applications. In addition, although there is a need to map application to multicore technology, we do not find a methodology presenting a structured set of steps to do so. We believe that the availability of such a methodology will aid this field in two ways: allowing easier implementation of HW/SW Codesigns for interesting applications, and boosting more investment in this field. In this thesis, we address these problems, and we map a life-critical application onto multicore systems on chip. Then we use performance analysis in order to show how such a HW/SW Codesign can serve other fields. Regarding the protocols/standards for communications, we propose a novel way for inter-multicore-chip communications and we present also some examples on inter-NoC protocols. This is because we believe that in the future, complex applications may require more than one multicore chip (MPSoC or NoC) to serve one aim. Hence, we believe that it is important to start designing the MPSoCs and NoCs in a way that will not hinder future inter-multicore-chip interactions. Moreover, we present a methodology for the MPSoC HW/SW Codesign, which addresses application mapping, Codesign, and testing/performance analysis.

2.5. Methodology

In this thesis we try also to build a set of steps to follow as a procedure in Multicore design for specific applications that require high computational capabilities. Therefore, we find it crucial to have an understanding and background in methodology in general. According to the Webster (See Appendix 4), the word "methodology" can be defined in several ways, but we choose the following definition for this work: "*a body of methods, rules, and postulates employed by a discipline, a particular procedure or set of procedures*." One of the hardest points in the design process is the starting point, which is many time a chaotic practice. For instance, in the software programming, this starting process is often referred to by *code and fix*. This way, may work if the system to build is relatively small, not much complicated, and is not life critical. However, when the system is large and is sensitive to errors, then running the design process as a chaotic "code/design and fix" method without any structure may lead to huge delays, complicated redesigns (that could mean repeating from start, i.e. more cost), and sometimes failure of the system. Therefore building a methodology is quite important, because any changes in the HW/SW partition may compel the designers to exhaustive redesign. Many designers try to fit most of the work in software and put only less arduous parts of the design to hardware so that they can respect deadlines. These methods suffer from the following main problems: (i) the lack of a unified HW/SW rendering, (ii) lack of compatibilities between different versions, (ii) difficulties in progress, (iv) priori decisions on design that do not aid in getting an optimized HW and optimized SW, and (v) unavailability of a structured design flow or methodology. These problems make specs testing, editing, and revision formidable. Hence, they result in direct and indirect negative impacts on the production and time-to-market. In what follows, we give some brief overview about Agile [117] methodology for SW and the POLIS [57] methodology for HW/SW Codesign. Agile methodologies try to have a tradeoff between *the lack of a process* and *too much of processes*, in order to build the satisfactory method that results in the required design. They are not as document-oriented as other SW methodologies. They are characterized by being code-oriented, hence also integrating documentation with the code so that progress is practically easier.

A more interesting methodology for the work in this thesis is the POLIS system, which revolves around a single representation like a Finite State Machine. However, it uses Co-design Finite State Machine (CFSM). CFSM shares with classical Finite State Machines (FSM) the functionality of giving outputs with only a finite amount of internal states, but it differs from FSM in that it uses a finite, non-zero, unbounded reaction time instead of the synchronous communication model. Every element of a network of CFSMs relates to a core of the system. The interesting point in CFSM is that it is unbiased towards neither HW nor SW. The steps or layers used in POLIS are:

1. *High Level Language Translation* for writing specs (e.g. VHDL)
2. *Formal Verification,* where it is possible to directly interface with existing verification methods based on FSM, making it able to verify designs that can be larger than was previously
3. *System Co-simulation,* where designers can get feedback on the design choices in both SW and HW
4. *Design Partitioning,* i.e. system-level design decisions on HW-SW partitions, architecture, and scheduling are taken. Here also design experience plays a large role. Thus, designers use feedback parameters on mechanisms from co-simulation and/or formal verification
5. *Hardware Synthesis, a* CFSM sub-network chosen for HW implementation is implemented and optimized using logic synthesis techniques from SIS. Each CFSM, interpreted as a Register-Transfer Level specification, can be mapped into BLIF, XNF, VHDL or Verilog.
6. *Software Synthesis, by* optimizing the desired behavior in a high-level, processor-independent flow, and translating the control/data flow graph into portable C code, then using any compiler to deploy/optimize with a specific instruction set.

A timing estimator quickly analyzes the program and reports code size and speed characteristics. In addition, real-time application-specific OS and I/O drivers are generated for every partitioned design.

7. *Interfacing Implementation Domains,* where interfaces between different hardware-software are automatically synthesized within POLIS and the communication can be through I/O ports.

In brief, most of the methodologies today do not look at life critical applications and how this imposes hard constraints on the Codesign, analysis, and performance testing. For instance, in fields like aviation and medicine, the HW/SW Codesign faces very serious and hard deadlines that must be considered in both processes: the design and the implementation. In addition, the current methodologies are confined to some specific languages for computing and for writing the SW (e.g. VHDL). In our work, we set a general methodology that is not confined to any specific computing language, and we show how it is useful to divide the design process to three layers: application, SW, and HW. The aim of the structure we propose is to help the designer in knowing, from the start, what to keep track of during the iterations of the HW/SW Codesign. This would decrease the time spent on the Codesign process. Moreover, it will aid in knowing and deciding where (i.e. at which steps) to run performance tests. We discuss this methodology in more detail in Chapter 6.

Chapter 3

Novel ECG Algorithms for HW/SW Codesign

"Science cannot solve the ultimate mystery of nature. And that is because, in the last analysis, we ourselves are a part of the mystery that we are trying to solve."

Max Planck (1858 – 1947)

This chapter discusses the application level and the novel algorithms that aid in advancing towards the solution. The application we focus on is the Electrocardiogram (ECG/EKG) biomedical practice, which requires an algorithm design and development. In this chapter we follow a top-bottom approach in describing the big picture for the solution, then we move down to the required system solution, and then proceed to the needed software. In section 3.1, we introduce the big picture scenarios, where a healthcare application like heart and ECG analysis can be applied to MPSoC as part of a general healthcare solution. Section 3.2 presents the filtering step, which is an essential step priori to analysis. Section 3.3 elaborates the first ECG analysis algorithms that we introduce in this thesis as the ACF-based algorithm. Section 3.4 presents a step further in trying to develop a less heavy algorithm from a computation viewpoint, the FFT-based algorithm. Section 3.5 compares our novel algorithms with each other and with the most widely used algorithm, namely the Pan-Tompkins algorithm (discussed in Chapter 2). The Pan-Tompkins algorithm has a good performance, but we were able to find some cases where a high percentage of error occurs.

3.1. Introduction

As discussed in the motivation of this thesis (section 1.1), one of the major aims is to find healthcare applications that leverage the use and interest in SoC and MPSoC platforms. In this respect, we follow a top-down approach in this section in describing the general scenarios and discussing where the MPSoC solutions fit. Then, we describe the proposed solution, and we focus in this chapter on the software part needed to be implemented on the hardware part (discussed in Chapter 5). We describe the big picture from a top view on the need for disaster-site and emergency healthcare, which has garnered a dramatic importance in the past years. There is a significant need for a general and global solution. There is no limit on the geographical location of a disaster, hence, the use of computing and communications becomes a direct need for a global emergency healthcare solution. In large scale disasters, the medical team may be fewer in number than the injured [73], hence an overall solution requires on-ground biomedical equipment of high performance to aid the medical team [79]. Figure 11 shows an example of a scenario for healthcare aid at the point of need (disaster point), where MPSoC can be attractive due to its small volume, light weight, and high computational capability. At such emergency cases checking and monitoring the patient's heart is essential. A healthcare application-specific MPSoC can be carried on the ambulance plane, or with the medical team at the disaster site to monitor hearts of the wounded and analyze them in real time, which is a basic and important check up to do in emergency cases for the heart. In addition to disaster scenarios, applications in home healthcare are becoming a predominant form of healthcare delivery. In addition, home-care ensures continuity of care, reduces hospitalization costs, and enables patients to have a quicker return to their normal lifestyles. Figure 12 shows an example of a home-healthcare.

Looking at the needs of the application as well as the market requirements for our solution, accordingly, we design our technical specifications. We refer to the emergency or disaster site as the *point of need*. Since the point of need can be anywhere on the plant and since we need high quality to carry medical data, a reliable high performance satellite protocol is vital for the telemedicine part of the solution. The communication links may suffer from disconnections, thus to overcome this problem, we put the largest and most important analysis segment on MPSoC as we discuss later in Chapter 4.

In our approach in thinking of the solution, we move from the big picture, to the required algorithms for heart analysis that we will fit on an MPSoC. Therefore, we move to the point of looking at how the wearable solution can be designed. As discussed in Chapter 2 the steps needed for the heart monitoring are: signal recording, filtering, and analysis. Due to these steps, we propose the design shown Figure 13. Then we move down from the top view of the solution to look at the SW needed for filtering and analysis.

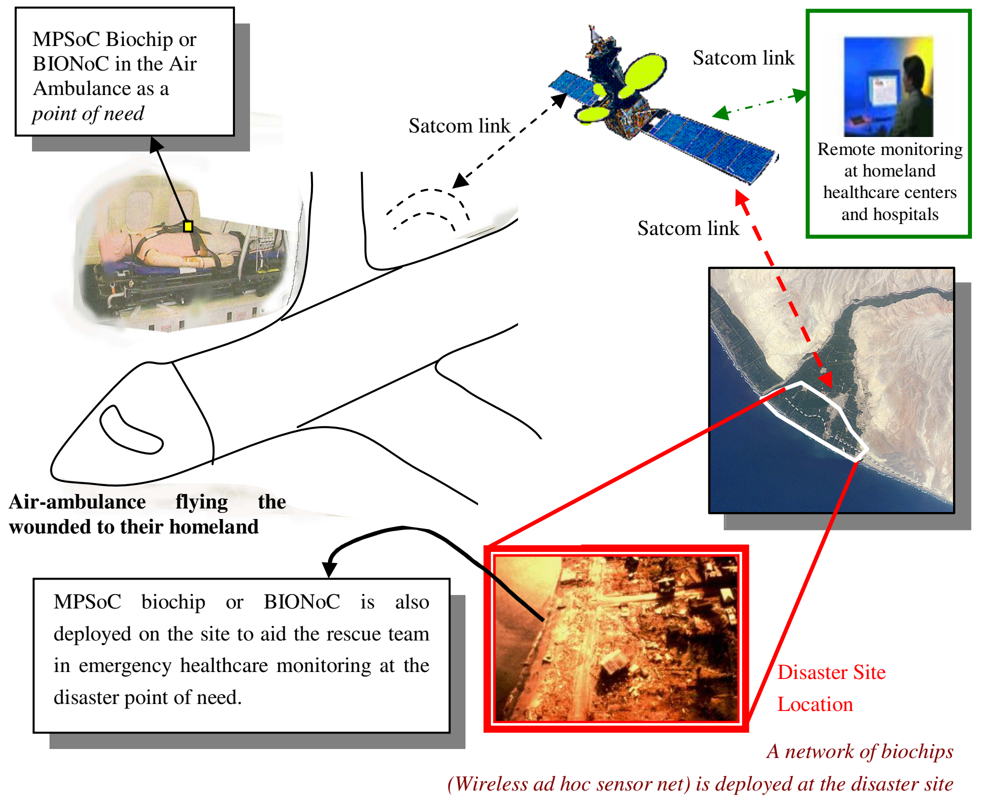

MPSoC Biochip or BIONoC in the Air Ambulance as a *point of need*
Satcom link
Satcom link
Remote monitoring at homeland healthcare centers and hospitals
Satcom link
Air-ambulance flying the wounded to their homeland
MPSoC biochip or BIONoC is also deployed on the site to aid the rescue team in emergency healthcare monitoring at the disaster point of need.
Disaster Site Location
A network of biochips (Wireless ad hoc sensor net) is deployed at the disaster site

Figure 11. Biochip deployment over a healthcare network for patient monitoring and real-time analysis at the point of need.

Most of the available solutions for detection of heart abnormalities (see sections 2.2.1 and 2.2.2) try to locate the QRS interval (see Figure 5) in order to estimate the heart period by calculating the distance of two consecutive R peaks. In spite of their lightweight complexity, such algorithms do not provide enough confidence in analyzing highly irregular heartbeats, associated with specific patients and/or arrhythmias [76]. Moreover, even when they work fine in defining the QRS interval, the other peaks (which represent other heart activities) will still be obscure for the doctors/nurses looking at the algorithm results.

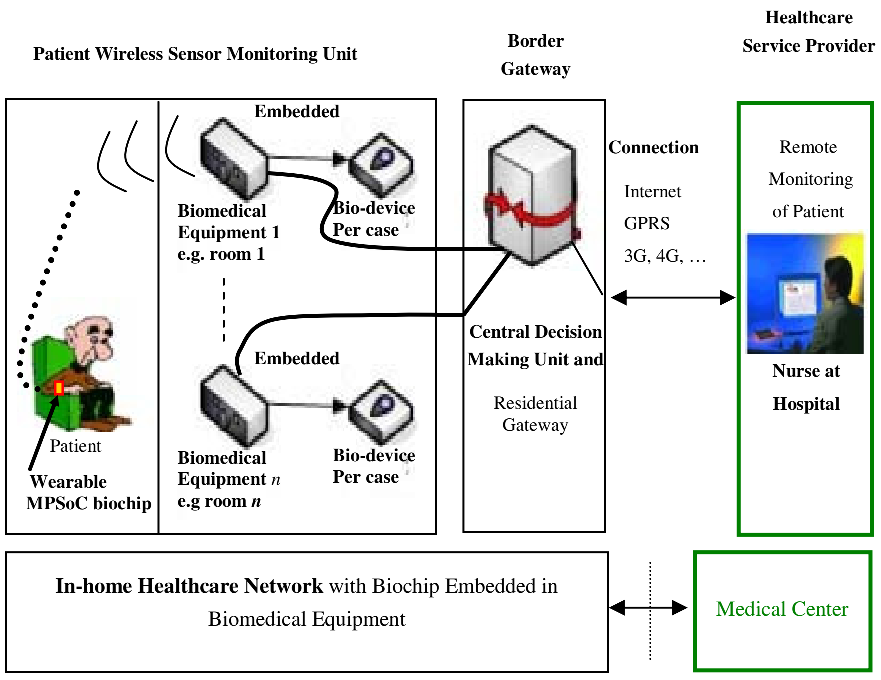

Patient Wireless Sensor Monitoring Unit
Border Gateway
Healthcare Service Provider
Embedded
Biomedical Equipment 1 e.g. room 1
Bio-device Per case
Connection
Internet
GPRS
3G, 4G, …
Remote Monitoring of Patient
Central Decision Making Unit and
Residential Gateway
Nurse at Hospital
Embedded
Patient
Wearable MPSoC biochip
Biomedical Equipment *n* e.g room *n*
Bio-device Per case
In-home Healthcare Network with Biochip Embedded in Biomedical Equipment
Medical Center

Figure 12. Home Healthcare scenario where Biomedical Applications on a wearable MPSoC can be applied. For instance, the patient can have his hearted monitored, via an application specific MPSoC.

At the biomedical level, a choice for the algorithm is a basic block for the success of the overall solution (the big picture we describe). Hence, to follow a path that may lead to the success of the algorithms (thus the solution) in the market in further steps, we start looking at the most widely used algorithm for heart beat detection in healthcare centers, namely the Pan Tompkins algorithm (described earlier in sub-section 2.2.1.2), in order to compare our novel algorithms with it. The Pan Tompkins solution is built to detect the QRS interval only, and its low complexity makes it suitable for porting on a large number of low-end portable devices. The disadvantages are of course many, e.g. lack of 100% success for all

ECG signals, confusing peaks and thus diseases, lack of full analysis of all heart activities, and- oftentimes- a lack of sufficient informative content provided to medical doctors. Some failures of the Pan-Tompkins algorithm are presented later in this chapter (section 3.5.1).

In a step to overcome the limitations of long-lived solutions, and keeping in mind that (for the overall solution) we rely on the performance accelerations guaranteed by MPSoC platforms, we came up with more computation demanding analysis (biomedical) algorithms than the traditional QRS detection techniques. We also point out their pros and cons from a comprehensive hardware-software implementation viewpoint. In what follows we refer to our two biomedical algorithms as: the ACF-based and the FFT-based algorithms. The presentation of these novel algorithms follows in sections 3.3 and 3.4, respectively. The algorithms, however, require that their input data be filtered to insure accuracy of the output (ECG analysis) results. Therefore, prior to discussing the new algorithms, we discuss the filtering step needed before running any of the computational algorithms for analysis.

In other words, our vision of the solution can be divided to two parts: the *on-chip analysis* and *communications* of the chip within larger networks in order to transfer the analysis results to the specialized medical centers. Hence, our vision of the solution in the first part (on-chip analysis) is shown in Figure 13, where the 12 ECG leads are input to a pre-analysis step. This pre-analysis step starts by running analogue to digital conversion. The ADCs output is required before the filtering process (with a filter for every lead) that we envision for this solution since we aim at allowing both hardware and software based filtering. Then we store the output of all filters on a data storage memory part. In this respect, we aim at using data storage in 16 bit binary format. We choose binary formats since they consume less space than other digital formats. We use the 16bit type since we aim at having finer granularity for the analysis while at the same time allowing for using currently available devices to realize the solution. In the second part (communications), we can not envision our real-time analysis chip without being able to send its results to healthcare professionals in medical centers (e.g. scenarios in Figure 11 and Figure 12). Consequently, we look at having protocols/methods that can aid with the convergence to a result while respecting the problem constraints (time, reliability, reactivity, and no loss) and while allowing for further improvements and simpler integration of other (and future) biomedical solutions on the same network, whether on-chip or between chips. With this vision of the solution, we start, in what follows, by discussing the pre-analysis filtering step (section 3.2), and then we describe our novel analysis algorithms in sections 3.3, 3.4, and 3.5. A note worth mentioning is that we present later in Chapter 4 some of the chip communications algorithms (intra and inter communications), which we propose for fitting the chip-solution to a real communications network carrying the biomedical-data results (analysis results).

3.2. Filtering As a Pre-Analysis Need

Data coming from sensors is subject to various types of noise discussed previously in section 2.3. Since lead-signals are calculated from sensor data, then the ECG biomedical leads suffer from many noise types, which impair the analysis of the 12 lead ECG. Running any computational algorithm on such noisy lead data will give erroneous results that may put the patient at a high risk. An Example is shown in the R-Peak detection marked by circled areas in Figure 14, where two peaks may be detected although there should be only one peak. Consequently, prior to the analysis algorithm, we send ECG lead-data through a filtering stage. We designed our own filter, which best suits our solution. In our work, many sensors may be chosen to serve our real-time aim, and at the same time have reasonable prices for the market success of the solution. For instance, one of the sensor types that may be used is the state of the art commercial sensor from Ambu Inc. silver/silver chloride ”Blue Sensor R” [15] shown in Figure 13. It is characterized by: 24 hour lifetime, superior adhesion, optimal signal measuring during stress tests, ease of carrying due to its small dimensions (57mm x 48mm), and ease of wearing.

In order to deal with noisy input signals, we designed an IIR filter with order 3 that outputs its results in 16-bit binary format (Figure 13). However, we need to be aware of the fact that we want to look in our solution at high sampling frequencies (250Hz, 1000Hz and above), because we want to: (a) make use of the available accuracy of the state of the art sensors, (b) have finer granularity of data, and (c) get more accurate analysis since in some cases more data samples are needed to discover a disease; like, for instance, the medical case known as the R on T phenomena [14], where the R and the T peaks are very near in time so we need a very high number of samples and an intelligent algorithm to discover them. Moreover, it is extremely important to choose a sampling frequency that minimizes the risk of aliasing.

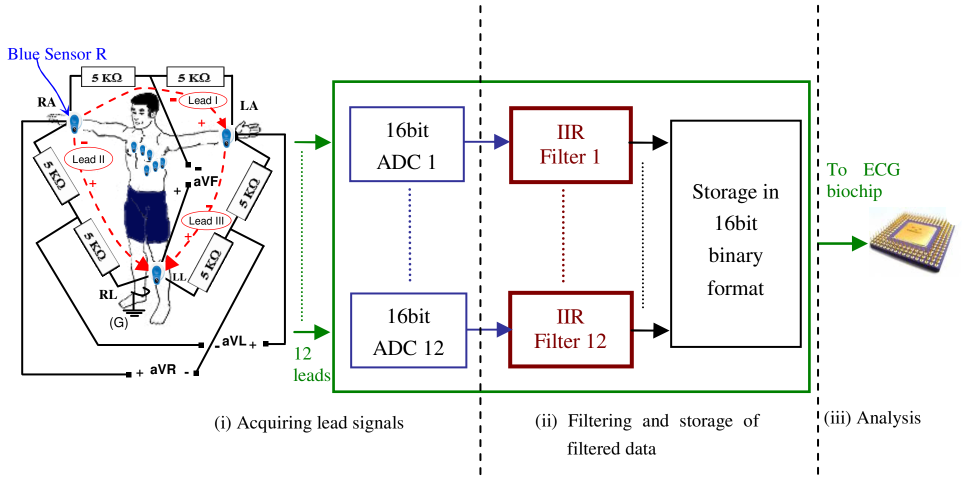


Figure 13. The system for sensing and filtering of ECG lead signals before sending data to the ECG biochip for analysis. Blue Sensor R is from Ambu Inc. [15]

The highest frequency needed for the ECG signal is 90Hz (due to the medical frequencies of the heart), which implies that the lowest sampling frequency that can be used is equal to the Nyquist rate (180Hz). However, in order to sample at such a frequency, the analogue signal has to be band limited to 90Hz, which can be achieved by the use of a complex analogue bandpass filter with a very sharp frequency response. This solution, although advantageous on limiting the amount of data to be stored, has a disadvantage on the analogue side, since the bandpass filter, being complex in order to meet the sharpness requirement, will probably have considerable power consumption. An alternative solution would be to sample at a frequency much higher than the Nyquist rate, such that the analogue bandpass filter can have a relaxed frequency response, while still effectively filtering out the frequencies that would cause aliasing during sampling. For instance, by choosing a sampling frequency of 5 KHz, all frequencies beyond 2.5 KHz would have to be filtered out before sampling, but that task is simpler than before, since all frequencies between 90Hz and 2.5 KHz can be attenuated without affecting the data needed for analysis. After sampling, band limitation to 90Hz can be implemented using a digital filter.

This approach has the advantage of using a lower-complexity bandpass filter, and reducing considerably the risk of aliasing and folding. Moreover, increasing the number of samples increases the accuracy of the sample, and makes the overall filtered signal smoother when used for analysis. Our IIR filter is built to deal with these problems. Another main advantage of using the IIR filter is to eliminate the noise that is directly proportional to the DC offset of the sensed ECG [41], which is around 0.1mV. The two plots in Figure 14 clearly show how the filtering algorithm remedies this problem. In our implementation, the filter is implemented in hardware on a dedicated chip feeding the external SDRAM memory of our biochip.

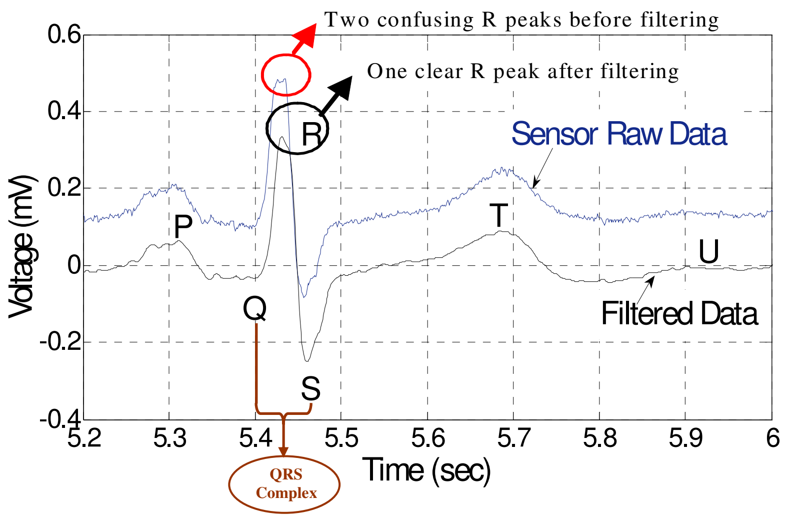


Figure 14. ECG raw and filtered data (lead I).

Our filter is the convolution of the noisy signal with the filter impulse response in (16) :

$$y[n] = \sum_k h[k] \times x[n-k] \tag{16}$$

where, $x[n]$ is the noisy signal, $h[n]$ is the filter impulse response, and n is the sample index. This filter in (16) is also an infinite impulse response (IIR, Chebyschev filter), so it can be written as (17):

$$y[n] = \sum_{k} x[n-l] \times b[l] - \sum_{m-1} y[n-m] \times a[m] \tag{17}$$

where, *y* is the output of the filter and *x* is the input, *b* is the vector that contains the filter coefficients for signal *x*, and *a* is the vector that contains the filter coefficients for output *y*.

The upper limits of the coefficients are dependent on the order of the filter being used. Our IIR filter is of order 3, because our ECG data does not require higher orders. We can improve our filter (when needed) by simply knowing the needed values of the coefficients in vectors *a*[.] and *b*[.]. Figure 15 shows the result when we apply our filter on the 12 lead signals which we read digitally. This filtering allows a finer analysis and hence more correct results.

Moreover, such digitally filtered leads make it easier for the physician and doctor to detect the malfunctioning parts of the heart by being able to see more clearly the signal and its peaks as is shown in Figure 14. After we achieve this filtering result, we are ready to run the analysis; hence we are ready to design the required algorithms for analysis.

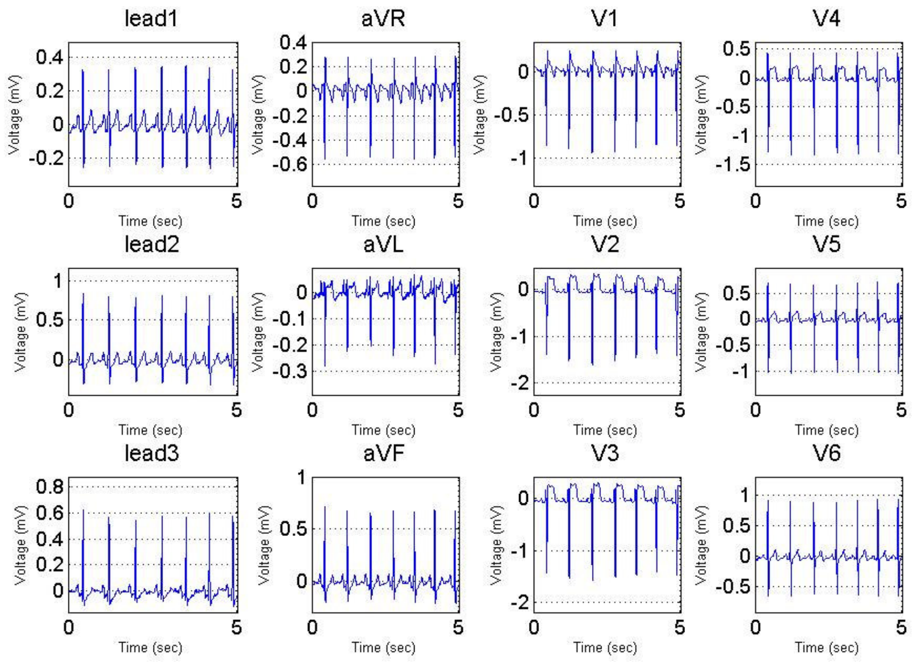


Figure 15. Twelve filtered ECG leads. Comparing with Figure 6 of the complete paper readout we can see the difference in the ability to digitally process data with higher quality.

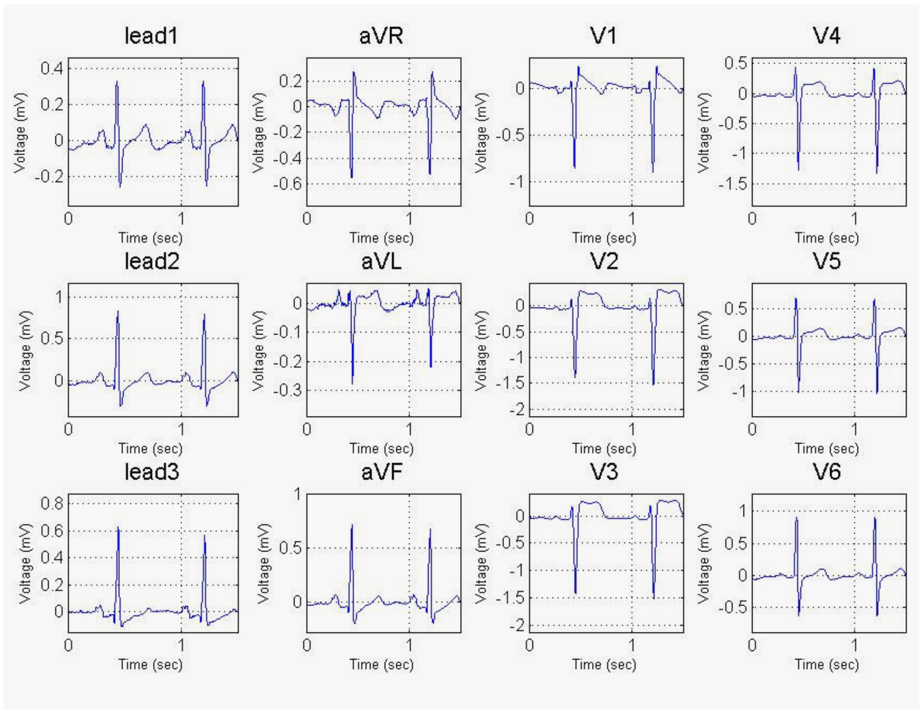


Figure 16. Twelve filtered ECG leads. Comparing with Figure 15, we can digitally zoom into the heart cycles we need. Comparing with Figure 6 of the complete paper readout we can see the difference in the ability to digitally process data with higher quality.

3.3. ECG Analysis Using the ACF-Based Algorithm

Most ECG systems make use of the Pan-Tompkins analysis algorithm [94], which targets QRS complexes (Figure 5; Figure 14) as discussed in Chapter 2. In principle, traditional ECG analysis starts from a reference point in the heart cycle (the R-peak is commonly used as the reference point). As a consequence, accurate detection of the R-peak of the QRS complex is a prerequisite for the reliable functionality of ECG analyzers [77]. However, as an effect of ECG signal high variability, R-peak detection might be inaccurate. For instance, in the R on T phenomena, a T peak may be wrongly taken for an R peak, and then the R-T interval will be considered as an R-R interval so that the period will be wrong. Other QRS parameters will be consequently inaccurate. As a result, traditional techniques may fail in detecting some serious heart disorders such as the R-on-T phenomenon (associated with premature ventricular complexes) [14].

The main algorithm contribution we have in our approach is to change the thinking, from looking at the peaks first and then using two peaks to get the distance between them to consider it as the period, to searching for the period directly. In other words, our treatment of the problem takes a different perspective: instead of looking for the R-peaks and then detecting the period, we detect the period first (via autocorrelation) and then look for the peaks. The reason why we thought of this aforementioned approach is the reality of high degree of randomness in the ECG signals. This randomness makes looking for peaks a cumbersome and erroneous process.

As a consequence, we had to create a mathematical solution for this ECG signal problem. One of the ideas that led to the solution we have now, was a question we had on why not try to look for the level of correlation between the ECG signal from a patient and a normal ECG signal. Then we found out that this may give some interesting results, however we may not get the period. What we would get is the level of correlativity these two signals have. Then we advanced a step ahead and thought of seeing at how much the patient ECG signal is correlated to itself. This was the start of a mathematical solution that can make a difference.

To understand this solution we start by describing it from the beginning. Since the ECG leads data is stored in digital format as shown in Figure 13, we then use the discrete signals for our mathematical solution. Hence, we consider $y[n]$ to be the ECG lead signal, where n is the index of the signal y, i.e. if $n = 0$, then $y[0]$ is the first value of the ECG signal. For example, the filtered lead signal shown in Figure 14, can be considered as y. However, in Figure 14 the *x-axis* does not show the index value n, but it shows the time value. In this respect, it is very important to note that there is a direct relationship between the time value and the index n. That is because this signal is a discrete signal. Hence, the signal y in terms of time (t) is the same as the signal y in terms of the index n. The relationship between the index n and the time value t is also dependent on the sampling frequency, f_s as shown in (18).

$$t = \frac{n}{f_s} \tag{18}$$

where t is considered with respect to the origin of the analysis-chunk, i.e. starting at a time equals to zero.

Hence, it would be easier to look at the time of the sample point as t_1, where in fact the difference between t_1 and the time at the origin (say t_0) is the value in (18). We express it in another way in (19).

$$\Delta t = t_1 - t_0 = \frac{n}{f_s} \tag{19}$$

The main idea is that if we run the autocorrelation function shown in (20) on the function y over the data sample recorded, and then we can get the coefficients of the ACF.

$$R_y[k] = \sum_{n=-\infty}^{n=\infty} y[n] \times y[n-k] \tag{20}$$

where R_y is the autocorrelation function, $y[n]$ is the ECG filtered signal, and k is the number of lags of the autocorrelation.

For our case of the ECG signal, $n \in \mathbb{N}$. Now that we want to find the period, we study the autocorrelation, and we see that in case a function is periodic, then its autocorrelation is periodic. Let us look together at a small example where a discrete function is correlated with itself as shown in Figure 17, Figure 18, and Figure 19.

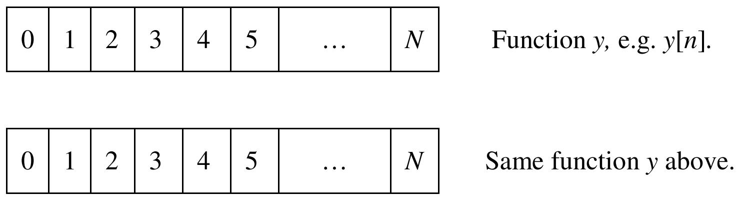


Figure 17. The first step in calculating the ACF. The numbers from 0 to *N* represent the value of the index *n*.

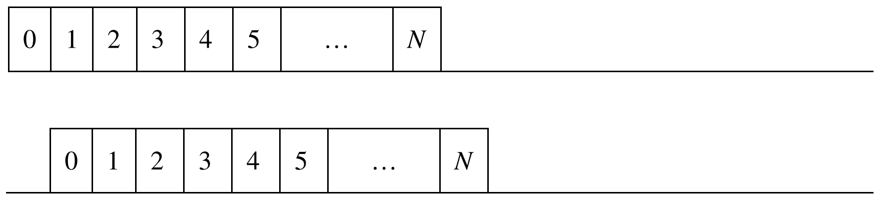


Figure 18. The second step in the calculation of the ACF. The numbers from 0 to *N* represent the value of the index *n*.

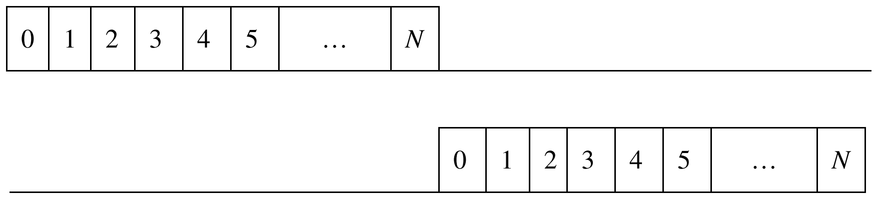


Figure 19. The Nth step in calculating the ACF. The numbers from 0 to *N* represent the value of the index *n*.

Every step of the ACF coefficients calculations (shown in Figure 17, Figure 18, and Figure 19) includes the multiplication of the number whose index is shown in the figures with the number whose index is just shown above it. We express it in the following:

$$R_y[0] = \sum_{n=0}^{N} y[n] \times y[n-0] = \sum_{n=0}^{N} y[n] \times y[n] = \sum_{n=0}^{N} y^2[n] \tag{21}$$

Then the calculation in Figure 17 is in (22) and the rest are similar.

$$R_y[0] = y[0]y[0] + y[1]y[1] + y[2]y[2] + ... + y[N]y[N]$$

$$= y^2[0] + y^2[1] + y^2[2] + ... + y^2[N] \tag{22}$$

$$R_y[1] = \sum_{n=0}^{N} y[n] \times y[n-1]$$

$$= y[0]y[0-1] + y[1]y[1-1] + y[2]y[2-1] + ... + y[N]y[N-1]$$

$$= y[0]y[-1] + y[1]y[0] + y[2]y[1] + ... + y[N]y[N-1]$$

$$= 0 + y[0]y[1] + y[1]y[2] + ... + y[N-1]y[N] \tag{23}$$

Similarly,

$$R_y[2] = \sum_{n=0}^{N} y[n] \times y[n-2]$$

$$= 0 + 0 + y[2]y[0] + y[3]y[1] + ... + y[N]y[N-2]$$

$$= 0 + 0 + y[0]y[2] + y[1]y[3] + ... + y[N-2]y[N] \tag{24}$$

Thus, after some iterations over *k*, we can reach a number *k* = *L* (*L* < *N*) that can be enough, in case of a periodic function *y*, to get the value of the period, and $R_y[L]$ will be

$$R_y[L] = \sum_{n=0}^{N} y[n] \times y[n-L]$$

$$= y[0]y[0-L] + y[1]y[1-L] + y[2]y[2-L] + ... + y[N]y[N-L]$$

$$= 0 + y[L]y[L-L] + y[L+1]y[1] + y[L+2]y[2] + ... + y[N]y[N-L]$$

$$= y[0]y[L] + y[1]y[L+1] + y[2]y[L+2] + ... + y[N-L]y[N] \qquad (25)$$

where *L* is a positive natural number related to the number of times needed for the calculations to get the period.

The main idea for *L* stems from the fact that if we minimize the calculations, then our convergence to a solution will be faster and also the power consumption on the HW system that we aim at developing, would be minimized. We will see later in Chapter 5 and chapter 6 how this number is important in the iterations to realize a HW/SW Codesign. As we can see from (25), the smaller *L* is, the less $R_y[.]$ calculations we would need, since *N-L* (which is the number of iterations that we can neglect will be larger). Now, in case *y*(t) is periodic with period *T* (in secs), then every *T* seconds (corresponding to some *n* steps) the multiplication of the signal *y*(*t*) by the signal *y*(*t-T*), would witness two identical signals being multiplied. For example, consider the same signal shown in Figure 17 but consider that every 3 samples a period is repeated. Then,

$$\begin{aligned} y[3] &= y[0] \\ y[4] &= y[1] \\ y[5] &= y[2] \\ &\cdot\ \cdot\cdot \\ y[i] &= y[i-3] \end{aligned} \qquad (26)$$

where *i* is an index value, i.e. $n = i \leq N$.

Therefore, we can express the function *y* with only three terms: *y*[0], *y*[1], and *y*[2]. Hence, we can express the autocorrelation (Figure 20, Figure 21, Figure 22, and Figure 23).

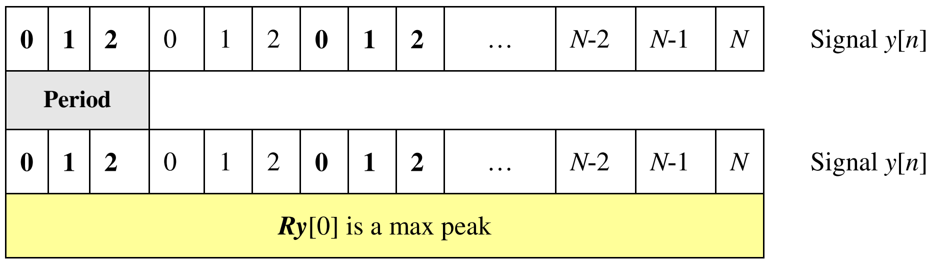


Figure 20. The first step in calculating the ACF for periodic $y(t)$. The numbers from 0 to N represent the value of the index n.. This is used for the calculation of $R_y[0]$.

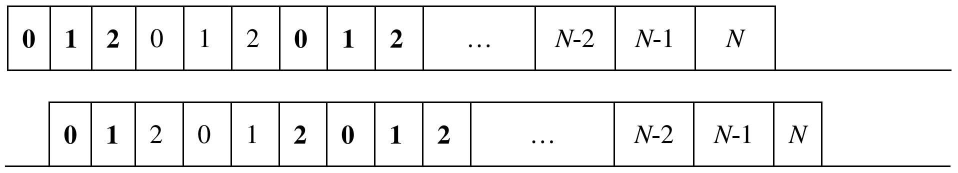


Figure 21. The second step in calculating the ACF for periodic $y(t)$. The numbers from 0 to N represent the value of the index n. This is used for the calculation of $R_y[1]$.

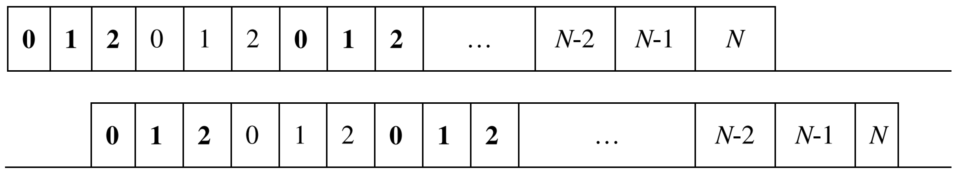


Figure 22. The third step in calculating the ACF for periodic $y(t)$. The numbers from 0 to N represent the value of the index n. This is used for the calculation of $R_y[2]$.

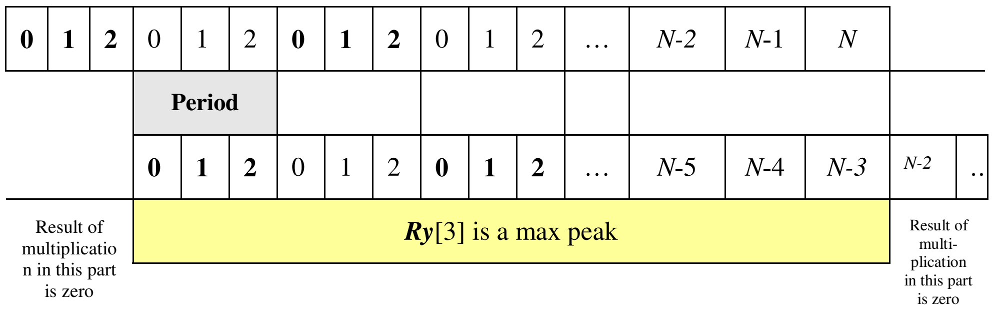

0 1 2 0 1 2 0 1 2 0 1 2 … N-2 N-1 N
Period
0 1 2 0 1 2 0 1 2 … N-5 N-4 N-3 N-2 ..
Result of multiplication in this part is zero
Ry[3] is a max peak
Result of multi-plication in this part is zero

Figure 23. The 4th step in calculating the ACF for periodic $y(t)$. Here we can see the clear overlap again of identical elements of the series. This is used for the calculation of $R_y[3]$.

Looking at the chronological steps from Figure 20 to Figure 21 then to Figure 22, and finally to Figure 23 we see that there is an identical matching in the elements of the two series (signals), and thus the maximum peak expressed in (22) will be reached, however it will be for a value smaller than the first peak since the multiplication of the first three elements of both series shown in Figure 23 results in a summation of zeros, i.e. the first period that corresponds to the first 3 elements of the upper series and the last three elements in the lower series, are all multiplied by zeros. Hence we will get a peak, but smaller than the first peak.

As we discussed before in (18), there is a direct relationship between the number *n* and the time value. Hence, the value of the period *T* can be simply calculated by substituting the values of *n* and f_s in (18) as shown in (20).

$$T = \frac{n_{period}}{f_s} \tag{27}$$

where n_{period} is the number of indices needed to have a period.

For example, in the case of our above example, $n_{period} = 3$. Hence the period in seconds is:

$$T = \frac{3}{f_s} \text{ seconds} \tag{28}$$

In the real case of ECG, a large number of sample points are needed, and the higher the frequency the higher the accuracy. However, now that we know that when we run the autocorrelation over a signal *y*, and if the signal is periodic, then every time there is an overlap with the periods (every *T*), we will have a peak in the ACF(*y*) = $R_y[.]$. Hence, every *T*, the ACF(*y*) is repeating the same behavior, which is having the peak. Simply, the distance between these ACF peaks is *T*, which is the period of *y*. In this example above, we can simply calculate the number of signal samples that are between two ACF peaks, and that is 3, because the 4th step in calculating the ACF reveals a second peak. In what follows we prove that the autocorrelation function of a periodic signal *y* (with period τ) is also periodic and its period is τ, the same as that of the signal *y*.

Let $A(t) = ACF_y(t)$ be a discrete sampled function (29)

and let $y(t) = y[n]$ be a periodic function with period τ.

τ corresponds to the specific number of index points on the *x* axis, N_τ.

Then $A(t) = A[n]$ (30)

We do the proof in the discrete sampled domain:

Now that $y(t)$ is periodic, then by definition of periodicity

$$y(t+\tau) = y(t) \tag{31}$$

and

$$y[n+N_\tau] = y[n] \tag{32}$$

and

$$\Rightarrow\ y[n] = y[n+N_\tau] = y[n-N_\tau] \tag{33}$$

$$A[k] = ACF_y[k] = \sum_{n=-\infty}^{+\infty} y[n]y[n-k] \quad \text{by definition.} \tag{34}$$

Then

$$A[k-N_\tau] = ACF_y[k-N_\tau]$$

$$= \sum_{n=-\infty}^{+\infty} y[n]y[n-(k-N_\tau)]$$

$$= \sum_{n=-\infty}^{+\infty} y[n]y[n-k+N_\tau]$$

$$= \sum_{n=-\infty}^{+\infty} y[n]y[n+N_\tau-k] \tag{35}$$

Substitute (33) in (35), then,

$$A[k-N_\tau] = \sum_{n=-\infty}^{+\infty} y[n+N_\tau]y[(n+N_\tau)-k] \tag{36}$$

Let $\rho = n + N_\tau$

So as $n \longrightarrow \pm\infty$; $\rho \longrightarrow \pm\infty$

Then let us substitute ρ in (36), we get (37).

$$\Rightarrow A[k-N_\tau] = \sum_{\rho=-\infty}^{+\infty} y[\rho]y[\rho-k], \tag{37}$$

which is exactly $ACF_y[k]$.

$$\Rightarrow A[k-N_\tau] = \sum_{n=-\infty}^{+\infty} y[n]y[n-k]$$

$$= ACF_y[k]$$

$$\Rightarrow A[k-N_\tau] = A[k], \tag{38}$$

which is the definition of periodicity of a function.

$\Rightarrow A$ is periodic with period N_τ corresponding to τ .

$\Rightarrow$ $A(t) = ACF_y(\tau)$ is periodic with period τ, which is the period of $y(t)$.

Hence, in order to calculate the period of the signal *y*, we calculate the distance between two consecutive maximum peaks in the ACF. However, for the ACF to reveal this inter-maximum-peak distance, we need a minimum set of sample points (n) out of the whole signal $y[n]$, which is made of N+1 samples. This minimum set of sample points is what we express in (25) as L. The choice of L is dependent on the application. In the example we give above, L=6 is enough since it requires only 6 sample points to get two maximum peaks and find the period. Hence in cases of a continuous signal (like signals coming from sensors for a long period of time) the value of L gives us a clear idea about the minimum time chunk of the signal that we need to send to the biochip for analysis, i.e. the size of the chunk of sample points that needs to be stored on the storage chip just before being read by the ECG biochip (as shown in Figure 13).

Indeed, if we run the autocorrelation over the whole sample (i.e. k runs from 0 to N), then this would reveal many periods. However, since the time constraint is a very important factor and since the power consumption is a very important factor, then minimizing the number L is of interest for the HW/SW Codesign process (discussed in Chapter 5), especially that the function we are processing (ACF) has many multiplications, i.e. consume a significant number of clock cycles in the core.

Although we know how to get the period from ACF inter-maximum-peaks, it is important to set a threshold to find the peaks. However, to set thresholds in a method, where we do not take a peak to be a maximum by mistake is dependent that the difference between the maximum peak and the other ACF peaks is significant. This is a very important step we have to solve. One solution is to transform the input signal *y* to a signal, whose peaks are exaggerated and whose small values are made nearer to zero as illustrated in Figure 24. At the same time, we need to run this transformation without adding a high number of multiplications or arithmetic operations in general since in the end we care about a SW algorithm that fits with the HW/SW Codesign.

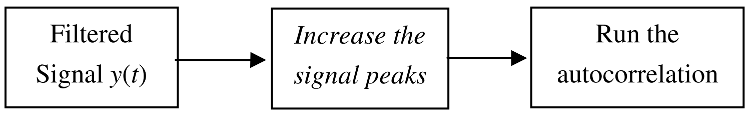


Figure 24. Illustration of the step between the filtered signal and its ACF.

In this respect, the derivative function is the best function we can run that can aid the purpose of increasing the signal peaks and at the same time not consuming a lot of arithmetic operations that may tire the system in time and power. Therefore, after reading the data chunk of the signal *y* with samples from the memory, a very helpful step is to calculate the derivative of the signal *y*(t). For our solution, we need to express the derivative in terms of *n*. Thus we approximate the derivative as follows:

$$\frac{\partial y}{\partial t}(t) \approx \frac{y[n+1]-y[n]}{(n+1)-n} = y[n+1]-y[n] \tag{39}$$

With this derivative, when there is a peak it will be increased with relative to the samples before it, and if the value of *y*[*n*] and *y*[*n*+1] are near to each other (i.e. no peaks) then the difference will look relatively smaller on the new derivative graph. To further illustrate the relative increase in the peak values, look at the signal in Figure 25-a and its derivative in Figure 25-b.

We also can simply prove that this process of derivative calculation would not affect the periodicity as we show in (40).

For a periodic signal *y*(*t*) with period *T*, then

Let $D(t) = \frac{\partial y}{\partial t}(t)$ (40)

Then,

$$D(t+T) = \frac{\partial y}{\partial t}(t+T) = \frac{\partial y}{\partial t}(t) + \frac{\partial y}{\partial t}(T) = \frac{\partial y}{\partial t}(t) + 0$$

$$\Leftrightarrow D(t+T) = \frac{\partial y}{\partial t}(t) \tag{41}$$

Substituting (40) in (41), then

$$\Leftrightarrow D(t+T) = D(t) \tag{42}$$

$\Rightarrow \quad D$ is periodic.

In (42) we can read that the derivative, *D*(*t*) of a periodic function with period *T*, is also periodic, and its period is also *T*.

Therefore, we adopt this solution of calculating the derivative of the filtered signal, and then we apply the ACF over the derivative. In this way, we guarantee that the signal is periodic. Then, the same period is carried in the derivative of the signal, and we guarantee that the calculation of the ACF of the derivative will give clearer peaks.

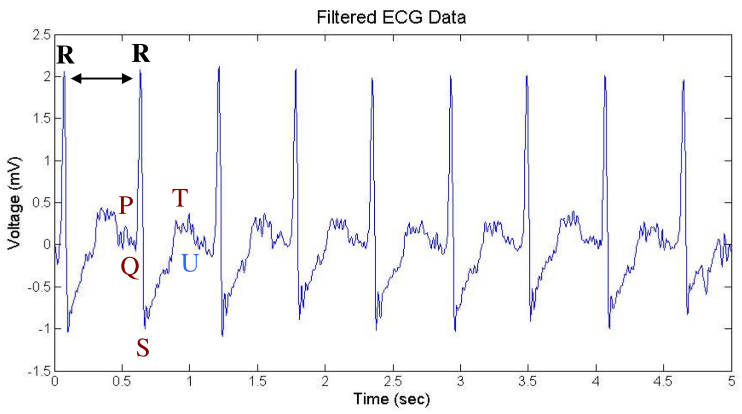
It is worth mentioning that when it comes to the HW/SW Codesign, we keep in mind using this mathematical solution within a range that is not beyond the capabilities of the HW architecture within the knowledge of the constraints, and allows for flexible testing during the Codesign process so that we can find, after a few iterations, the best suitable numbers for the SW, HW, and the application. In our ECG analysis case we use an autocorrelation function (ACF) to calculate the heartbeat period without looking for R peaks. Then, we can restrict our analysis to a time window equal to the period and detect all peaks. Designing an algorithm should be done with a look at the SW component. Although potentially more accurate, our algorithm incurs a higher computational complexity: 3.5 million multiplications, which have been reduced to 1.75 million through a number of code (SW) optimizations. In this respect, we also keep in mind that this algorithm SW implementation will run over a special HW. Thus, the single-chip multiprocessor architecture that will be selected for the practical implementation of the algorithm will provide the scalable computation horsepower needed for the highly accurate ECG analysis that we are targeting (which we discuss more in Chapter 5). The autocorrelation we use, as shown in (25), has a certain number of Lags (L) to minimize the computation for our specific application as discussed below. We validated our algorithm over several medical traces [133][142].

We run the experiments for n = 1250, 5000 and 50,000 relative to the sampling frequencies of 250, 1000, and 10,000Hz, respectively. In order to minimize errors and execution time we use the derivative of the ECG filtered signal since if a function is periodic then its derivative is periodic. Hence the autocorrelation function of the derivative can give the period as shown in Figure 25. In order to be able to analyze ECG data in real-time and to be reactive in transmitting alarm signals to healthcare centers (in less than a few minutes), a minimum amount of acquired data has to be processed at a time without losing the validity of the results.

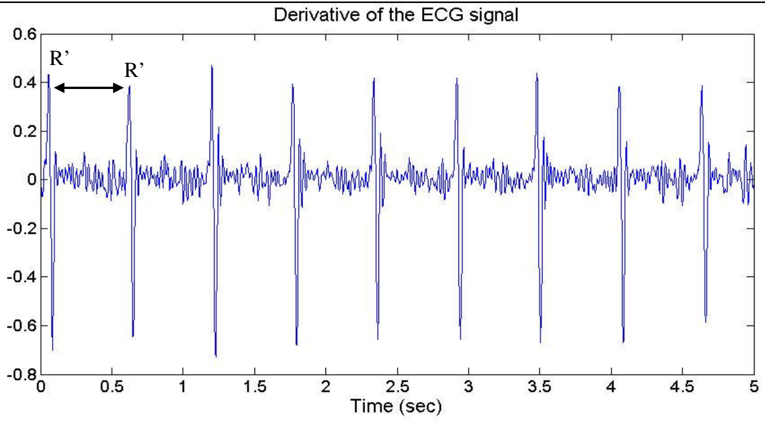
For the heart beat period, we need at least 3.5 seconds of ECG data in order for the ACF to give correct results. In some experiments, we used 4 second chunks to be more on the safe side of analysis; hence this chunk period is a SW parameter that will prove important in the HW/SW design later in the coming chapters.

The autocorrelation function is deployed within the algorithm shown in Figure 26, which computes the required medical parameters: heart period, peaks P, Q, R, S, T, and U, and inter-peak time spans. Peak heights and inter-peak time ranging outside normal values, which indicates different kinds of diseases, are detected with our algorithm. From a functional viewpoint, the algorithm consists of two separate execution flows: one that finds the period using the autocorrelation function (process 1 in Figure 26), and another one that finds the number, amplitude and time interval of the peaks in the given 4-second or 3.5 second ECG data (process 2 in Figure 26). In process 1, we firstly find the discrete derivative of the ECG signal. This will not affect the analysis since the derivative of a periodic signal is periodic with the same period. The advantage of taking the derivative, and thus adding some overhead to the code, is that the fluctuations taking place in the signal and especially those around the peaks would be reduced to a near-zero-value. Moreover, performance overhead associated with derivative calculation of the ECG signal is negligible compared to the rest of the algorithm, especially the autocorrelation part.

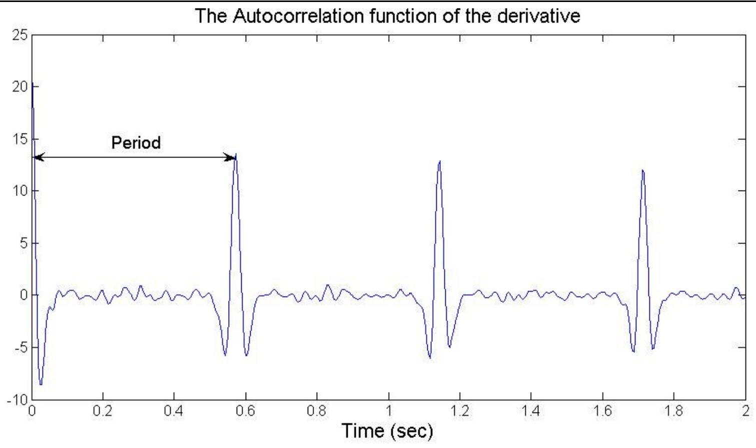
Finally, if the original signal is periodic, then the autocorrelation of the derivative of the signal is periodic as we prove above, with the same period as that of the original signal under test. In process 2, a threshold is used to find the peaks. This threshold was experimentally set to 60% of the highest peak in the given search interval.



(a)



(b)



(c)

Figure 25. Heart period analysis: (a) ECG signal peaks P, Q, R, S, T, and U; (b) derivative amplifying R peaks; (c) autocorrelation of the derivative characterized by significant periodic peaks having the same value as the period of the ECG signal in (b) and thus (a).

Our proposed ECG-analysis algorithm was conceived to be parallel and hence scalable from the ground up. Since each lead senses and analyzes data independently, each lead can then be assigned to a different processor. So, to extend ECG analysis to 15-lead ECG or more, then what is required is to change the number of processing elements in the system. Alternatively, more leads can be processed by the same processor core provided the real-time requirements are achieved. In the end, we design this algorithm to run on parallel processors, where every lead can be analyzed separately and the analysis results of each lead would be saved in one file. Then, a global 12 lead analysis can be run over all leads.

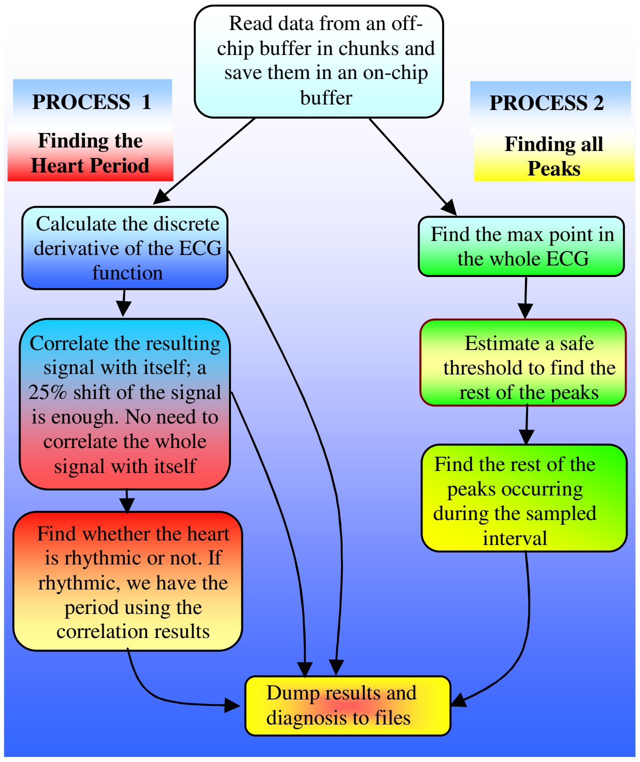

Read data from an off-chip buffer in chunks and save them in an on-chip buffer
PROCESS 1
Finding the Heart Period
PROCESS 2
Finding all Peaks
Calculate the discrete derivative of the ECG function
Correlate the resulting signal with itself; a 25% shift of the signal is enough. No need to correlate the whole signal with itself
Find whether the heart is rhythmic or not. If rhythmic, we have the period using the correlation results
Find the max point in the whole ECG
Estimate a safe threshold to find the rest of the peaks
Find the rest of the peaks occurring during the sampled interval
Dump results and diagnosis to files

Figure 26. The Autocorrelation function-based method for ECG analysis.

3.4. ECG Analysis Using the FFT-Based Algorithm

We investigate algorithm optimization before going to the HW design in Chapter 5. Looking for a less demanding analysis method from a computation viewpoint, while still retaining higher accuracy than the Pan Tompkins, we revert to the Fast Fourier Transform (FFT) algorithm to minimize the number of computations needed to estimate the autocorrelation coefficients. The FFT-based algorithm still uses the autocorrelation concept after it proved being efficient in terms of percentage of error, but we used a different approach to calculate it based on a mathematical property. This different approach relies on the calculation of the Fast Fourier Transform (FFT) of the signal or the FFT of the derivative of the signal. The autocorrelation is expressed by the following equation (25), and this expression can be seen as a convolution as in (43).

$$R_x[k] = x[k] * x[n-k] \tag{43}$$

Taking (43) to the Fourier domain we get (44).

$$DFT\{R_x[k]\} = DFT\{x[k] * x[N-k]\} = X[k]X[N-k] = X[k]^2 \tag{44}$$

Hence to get the autocorrelation function we take the FFT of the signal *X*[*k*] (we use FFT to calculate the DFT), then square it and take the inverse FFT (IFFT) to get the ACF function. This method will yield a big reduction in the running time and complexity. The logic of the code implementation is discussed in Appendix 2. In our implementation of the FFT algorithm, data are read from the ECG signal with a specified sampling frequency into a buffer with a limited buffer capacity of 3.5-4 seconds. Then, we differentiate the resultant signal so as to enhance its shape. Only the first quarter of the differentiated samples will be used as an input into the FFT block. This technique allows us to save time and unnecessary calculations while still getting to similar results. The differentiated data need to be reordered so that they are correctly used as an input to the FFT block. We perform decimal to binary conversion of the indices of the elements in this array, then reverse the bits of the binary representation of those indices, then perform binary to decimal conversion of the result to get the reordered indices. By correctly applying the butterfly method of the FFT and using the twiddle factors at every stage, FFT data are calculated. The algorithm uses two 'For' loops, the outer loop loops log (N/4) times (which represents the number of stages in the butterfly diagram), where *N* is the number of samples increased to the nearest power-of-2 value. The other loop loops *N*/4 samples, since at every stage; *N*/4 FFT values are calculated. We finally get the FFT data in the last stage. Although the FFT-based algorithm results are faster to compute, we observe a loss in the quality of the output autocorrelation plot in the end, i.e. the autocorrelation function coefficients (which is the final plot in both FFT-based and ACF-based algorithms) are not as accurate as the ones described in our ACF-based Algorithm, hence we win on computations and lose on accuracy (Figure 27).

Performing this same operation on different cases, we deduced that the peaks resulting from the FFT method yielded same results (same period) as with the direct method to calculate the ACF but we had a little difference between the data resulting from the 2 methods; the second peaks for the FFT method were lower in amplitude relative to the maximum at the origin than those for the direct method, which will require more accurate threshold for the FFT case as shown in Figure 28, and we discuss this in section 3.5.

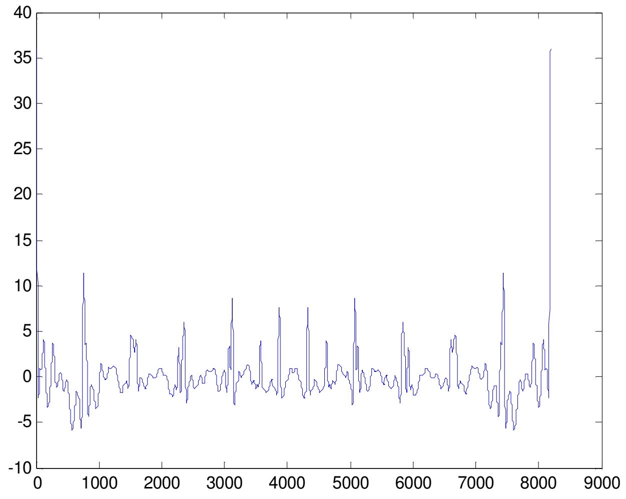


Figure 27. Autocorrelation data from the FFT-based algorithm. To get the heart period from this data, we take the peak at the origin and the next peak and then subtract the indices and divide by the sampling frequency.

We can see from the autocorrelation of the FFT function that the peaks are not as clear as in the ACF based algorithm.

3.5. ECG Analyses-Algorithms Comparison

The distinctive features of the aforementioned algorithms with respect to the traditional Pan Tompkins algorithm will be highlighted. We aim at proving user-perceived practical advantages of deploying more computation-demanding algorithms, and therefore the advantages that MPSoC technology can bring to the ECG domain in view of its scalable and energy efficient computation horsepower.

3.5.1. Comparison between Pan-Tompkins and the ACF Methods

We compare the Pan-Tompkins algorithm with both the ACF-based algorithm and the FFT-based algorithm since both of our novel algorithms use the idea of calculating the autocorrelations points to find the period. Although the Pan Tompkins algorithm (discussed in chapter 2) consumes many computations, it is quite fast since it detects slopes and amplitudes, however, to compare its efficiency and applicability we ran it over the standard MIT/BIH arrhythmia database [133].

In this respect our ACF based solution ran all the MIT/BIH arrhythmia data base cases without any failure, and it detected the heart period with an average percentage error of 3%. The error was calculated as discussed in the case studies below. However, when we ran the Pan-Tompkins implementation, we found out that the Pan Tompkins fails and may lead to erroneous results. We list below two cases of the MIT-BIH arrhythmia Database which failed (with a high percentage error) when using the Pan-Tompkins solution, while our solution showed a much better performance.

Moreover, looking at the Pan Tompkins algorithm discussion in Chapter 2, we realize that the human factor plays a role in the choice of thresholds and the relations therein.

3.5.1.1. First Case for the Pan-Tompkins Algorithm Failure

The first case when the Pan Tompkins algorithm fails is when run on RECORD 217 (Male, age 65) data taken from Lead II [133]. The trace characteristics are: paced beats, pacemaker fusion beats, and normal beats. The period using the eyeballing techniques, the Pan Tompkins algorithm and our ACF-based solution are 0.5361 seconds, 0.8722 seconds, and 0.5500 seconds, respectively. The eyeballing period was calculated from the first 2 R-peaks of the ECG data. The error when using the Pan Tompkins solution (which is the difference between the periods calculated in Pan Tompkins and Eyeballing divided by the Period using the eyeballing technique) is: 62.7%. Although it converges in a similar time as our ACF solution (<3.5s), but the results can be devastating for a patient. In the ACF case the error (which is the difference between the periods calculated in ACF method and Eyeballing method divided by the Period using the eyeballing technique) is only 2.6%, which is acceptable and still gives the physician a good margin to choose the right disease.

3.5.1.2. Second Case for Pan-Tompkins Algorithm Failure

The second case when the Pan-Tompkins algorithm fails is when run on RECORD 208 (Female, age 23) data taken from Lead II [133]. The trace characteristics are: ventricular couplets. The period using the eyeballing techniques, the Pan Tompkins algorithm and our ACF-based solution are 0.6111 seconds, 0.2583 seconds, and 0.6556 seconds, respectively. In this case, the error using the Pan Tompkins solution is: 57.8%, and the nurses and medical team can easily realize the advantages of our HW/SW solution that can give more accurate results than the nowadays solutions and can converge in real-time to the solution. The reason for the failure of the Pan Tompkins solution is that due to its afore-discussed algorithm, it tries to find correct thresholds that finally depend on the human choice. Moreover, it detects amplitude peaks, i.e. when the T and R waves are near, like the case of R-on-T phenomenon, the Pan Tompkins can not differentiate the R from the T and an R-T interval will be confused to be considered as an R-R interval, as revealed by the tests. However, using our ACF based SoC solution, we minimize the number of computations (although in millions) to suit the HW/SW co-design and find a method that takes the time axis as its point of reference and looks at the periodicity, accordingly, we detect the period first, without a need for a threshold for the amplitudes.

Comparing our solution with that of the widely used Pan Tompkins shows that we do not waste any power or time on looking for thresholds, but rather use a concrete form of ACF and autocorrelation-derivative to find if the heart is periodic, and we calculate the period immediately through the ACF. In this way, we leave no space for failure since we simply do not look at the amplitude axis. At the same time, we are able to design the hardware MPSoC that can cope with the required software and algorithm.

Therefore, the biomedical advantages that both our algorithms (ACF-based and FFT-based) have with respect to traditional solutions are mainly: (1) eliminating failure in calculating the heart period since we use a time based autocorrelation instead of using human-chosen thresholds, (2) ability to learn about all the heart activities since even if the Pan Tompkins gave a good detection, it already filters out the P and T waves, (3) ability to detect more diseases, (4) increased accuracy by increased sampling frequency, and (5) better scalability and parallelism.

3.5.2. Comparison between the ACF-Based and the FFT-Based Algorithms

Both the ACF-based algorithm and the FFT-based algorithm share the aforementioned advantages over the Pan Tompkins; however, we also can look at the medical implications when comparing the ACF-based algorithm with the FFT-based algorithm. In this respect, the FFT-based algorithm gives a faster result than the ACF-based algorithm, but it may not be as accurate, since the peak of the autocorrelation coefficients is surrounded by many similar peaks which are shown in Figure 28 and Figure 29.

Performing this same operation on different cases, we deduced that the peaks resulting from the FFT method yielded same results (same period) as with the direct method to calculate the ACF but we had a little difference between the data resulting from the two methods; the second peaks for the FFT-based method (Figure 28) were lower in amplitude relative to the maximum at the origin than those for the direct method (Figure 29). Hence they require more difficult thresholds choice for the FFT case. From a computation viewpoint, both the ACF direct method and the FFT method are clearly more demanding with respect to the Pan-Tompkins.

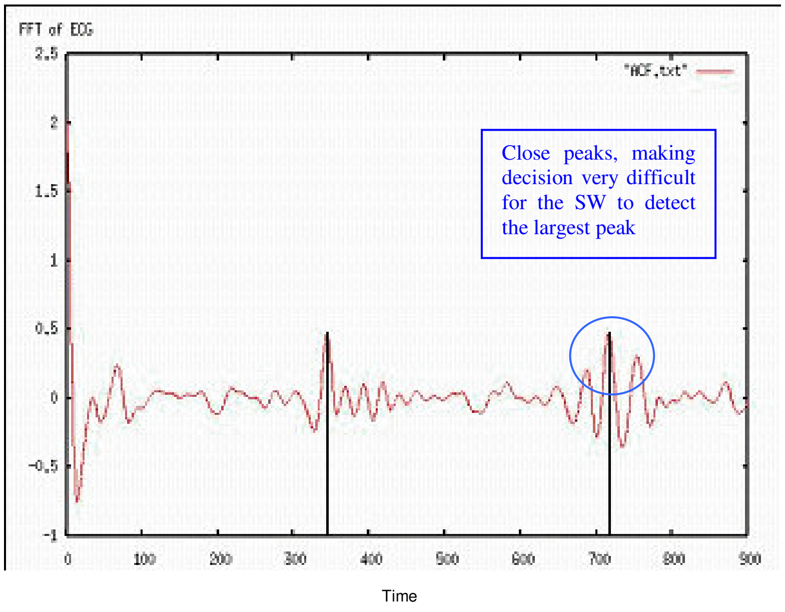


Figure 28. Period detection using FFT-based algorithm to calculate the autocorrelation coefficients.

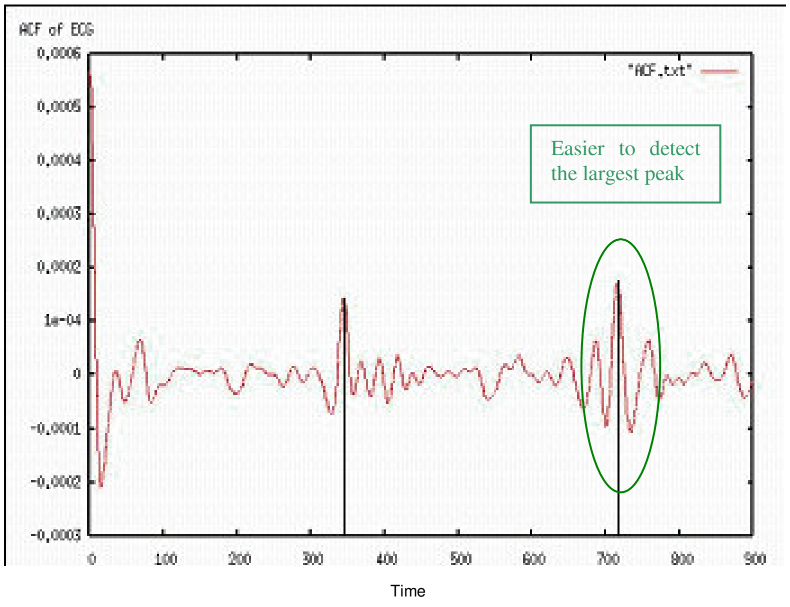


Figure 29. Period detection using the ACF-based direct method to calculate the autocorrelation coefficients for the same patient data presented in Figure 28.

The relative decrease of complexity that the FFT method is able to provide in the computation for our specific algorithm and SW design (in SW layer, i.e. dependencies and specs) of the autocorrelation function is well documented in Figure 30, where the effect of input data (*N*) on the number of algorithm multiplications is illustrated.

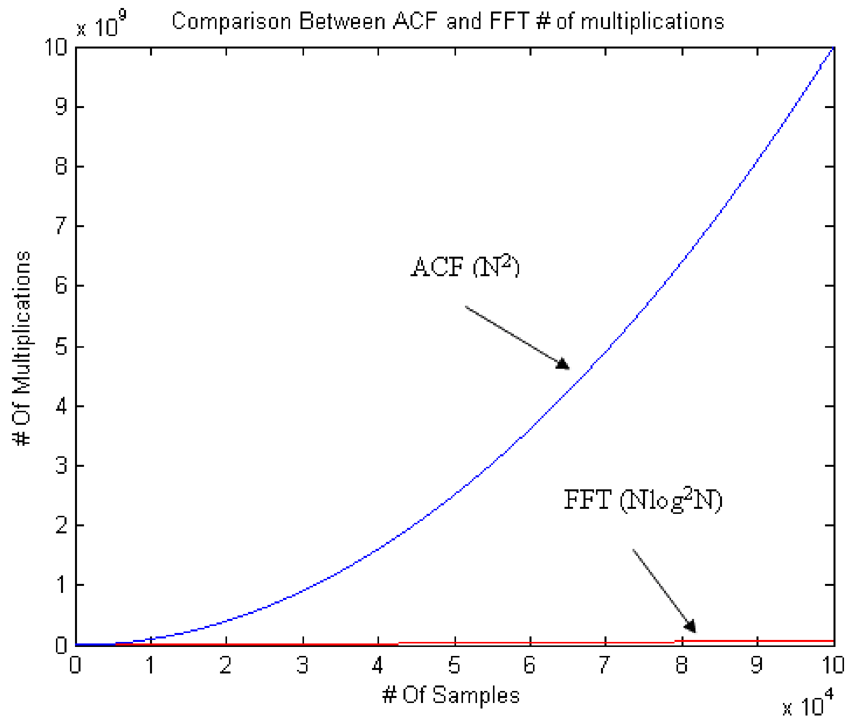


Figure 30. Number of multiplications for the direct ACF-based method, and the FFT-based method for our biomedical algorithm. The number of multiplications for the ACF-based method is O(N^2), and the number of multiplications for the FFT-based method is O($N\log^2 N$),

Chapter 4

Inter-Multicore-Chip Communications

"You can't cross the sea merely by standing and staring at the water."

Rabindranath Tagore (1861-1941 A.D.)

This chapter looks at the multicore chip as a small node in a larger network, where communications occur in order to transmit and receive the biomedical messages related to the biomedical analysis. In other words, in order to make use of our solution and algorithms, we can not just stand at the point of the HW/SW codesign, but also relate this biochip to a larger network able to carry the life-critical results. Hence, the chapter looks at the MPSoC as a network of multi-processors (or NoC) within a wider area network. Section 4.1 discusses fitting the MPSoC Solution in the big picture solution shown in Figure 11 and Figure 12. In Section 4.2, we present the work on external network-on-chip communications. In subsection 4.2.1, we present the principles for an external network-on-chip communications protocol, the ENoP. Subsection 4.2.2 elaborates on the algorithms for inter-NoC communications. In section 4.3, we present the simulation environment. Section 4.4 summarizes the chapter.

4.1. Protocols for Fitting the MPSoC Solution in the Big Picture Solution

In order to fit the biochip solution which will run the aforementioned biomedical algorithm we also stretched the thinking for when more than one biochip is needed. Hence, we looked at the biochip with multicore (MPSoC) as a network (on-chip network) within a network (communications network). Hence the idea of NoC best suits the solution when we look at the biochip as part of the global solution shown in Figure 12. However, NoC still faces some challenges [178]. In Telemedicine, care-links require a good quality link and communication path [78]. This may require more than one inter-connected NoC. One challenge will be to enable the interaction between many different NoCs. To succeed, we introduce the notion of NoC as an Autonomous System (NAS) [74]. The result is a reliable protocol for a biomedical chip, which we refer to as biomedical network-on-chip (BIONoC). This protocol aids BIONoC communications at the point of need (Figure 11 and Figure 12). Since a common protocol is required to establish multi-NoC interoperability, we define a mechanism to enable many NASs to interact together. We look at the NoC as a system of reference and as a basic building block in the multi-NoC interconnects environment. Each NoC is a self-governed system with its private internal policies that are designed and implemented by the manufacturer. We model this system as the NAS. Figure 31 shows a general NoC that can be connected to another NoC via two lines. These lines are connected to two Boundary Cores (BCs) inside the NAS. In order to allow for internal and external communications we create two protocols, namely the External NoC Protocol (ENoP) for inter-NoC communications and for communicating between a NoC and another end point (like sending the ECG analysis results to the medical center), and the Internal NoC Protocol (INoP) that aids the ENoP in transmitting messages coming from outside the NAS/NoC to reach the destination core inside the NoC.

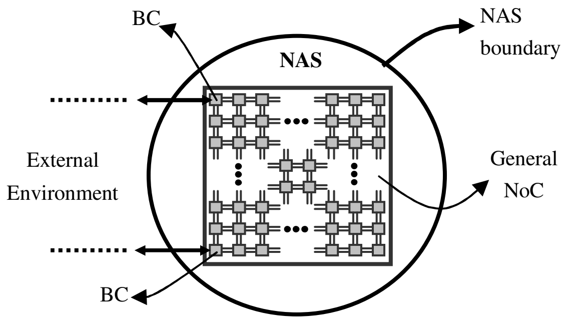


Figure 31. NoC as an autonomous system (NAS). A basic block for inter-NoC communications. BCs Run the ENoP on the external NoC link and INoP on the internal link.

A core is a set of internal NoC components physically connected to serve a set of services. The set of physically interconnected NoCs is also a logical set of connected NASs (Figure

32). The logical NAS entity enables us to define internal and external logical policies. This is a crucial step in managing networked systems since without the presence of well designed and defined logical policies; problems in scalability and interoperability arise. The NAS defines a logical boundary (see Figure 31) around the NoC separating internal NoC policies from external NoC policies. However, these policies are not totally separated as they share the BCs. Boundary Cores run the Inter-NoC protocols on the external connection and the Intra-NoC protocol on the internal connections. A NAS that is interconnected to more than one NAS at the same time is referred to as multinet-NAS, e.g. NAS1, NAS2, and NAS3 in Figure 32.

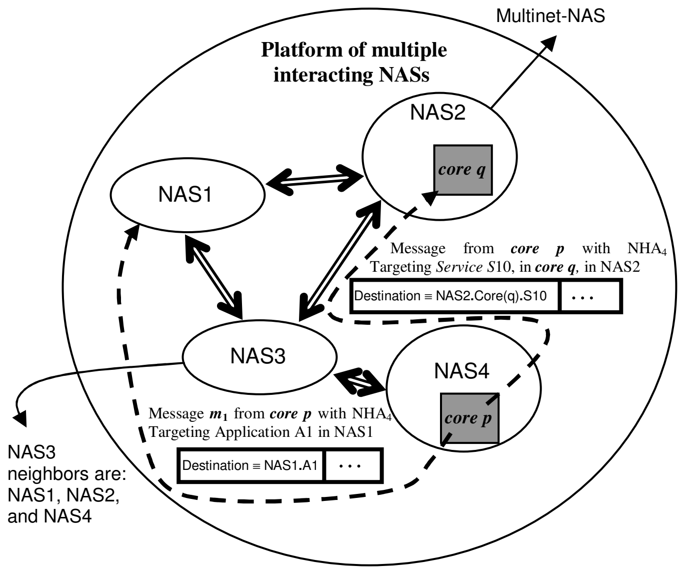

Multinet-NAS
Platform of multiple interacting NASs
NAS2
core q
NAS1
Message from core p with NHA4 Targeting Service S10, in core q, in NAS2
Destination ≡ NAS2.Core(q).S10
...
NAS3
NAS4
core p
Message m1 from core p with NHA4 Targeting Application A1 in NAS1
Destination ≡ NAS1.A1
...
NAS3 neighbors are: NAS1, NAS2, and NAS4

Figure 32. Logical topology of multiple NASs of physically connected NoCs (or BIONoCs in emergency situation for 4 patients) that are logically interacting to provide a larger platform with higher capacity and more capabilities in order to serve some common application or purpose [73].

Two NASs are said to be neighbors if they are connected and can interact, i.e. send and receive messages. These messages could be biomedical data transferred from one core to another in order to run further analysis via a SW that is available on the second core. For example, NAS1 has NAS2 and NAS3 as neighbors. In order to have reliable communications, we should be able to have a unique logical addressing scheme among NASs. Knowing that a NAS is associated with a physical NoC, this modeling inspires us to define a unique hardware address for each NoC, which may also reflect the vendor. In order to have reliable communications, we should be able to have a unique logical addressing scheme among NASs. Knowing that a NAS is associated with a physical NoC, this modeling inspires us to define a unique hardware address for each NoC, which may also reflect the vendor. Each NAS has its own unique address that is defined to be the NoC Hardware Address (NHA_x, where x is an index reflecting the NAS number). This scheme allows two general types of communications. The first type is when a specific core p, wants to interact with a specific core q. The other scheme is when a core p wants to interact with some application in some NAS. Figure 32 illustrates both examples. In the first example, core p of NAS4 wants to interact with some Service, $S10$, on core q of NAS2. The source address will then be the internal address of p associated with NHA_4. The destination address will be the 'pointed to' from the larger entity to the smaller entity as: NAS2.Core(q).S10. In this way, no mixing of addresses occurs even though two processors p and q may have the same internal addresses inside their respective NoCs. This is because the NHA_x is uniquely assigned to each NoC, i.e. no two NoCs have the same NHA_x. Industrially, this is realizable since we have many products in the market of such an addressing scheme. For instance, network interface cards (NICs), which have a unique hardware/MAC address per NIC. So, the NHA_x proposition is a solution not far from industrial practices. The second type of communications illustrated in Figure 32 also uses the proposed NHA_x scheme. The sending core needs to just know the address of the NAS, where the application resides. Core p as a sender will then just need to point to the destination as application A1 in NAS 1 (NAS1.A1). It is then the job of the ENoP Border Core of the NAS receiving the messages to deliver them to the INoP, which routes them to the core running the application. Although some networking fields have developed some solutions for problems that may seem similar like Wireless Sensor Networks (WSN), and the Internet, a thorough look at the inter-NoC communications gives insight to the differences between the problems and solutions we present and the already existing ones. One of the main differences between inter-NoC platforms and wireless sensor networks is that in inter-NoC communications the main unit of reference is a NAS containing a set of cores, while in WSN the main unit of reference consists of one core (usually containing one low power processor) [19]. Hence, in WSN, we do not care about internal routing in the sensor. In our solution the ENoP depends on information from the INoP to be able to send this information out of the NAS. What we share with WSN is the fact that the NASs (unit references) could be connected with a large variety of topologies; hence there is a need for topology discovery. This need is a common issue, but building protocols to address this need differs as we depend on some internal information from the NAS (NoC). On the other hand, when we compare our problem with that of the large scale networks like the Internet, then the issue of power consumption is a main difference. In the Internet we would not care when building and external autonomous system protocol if it would consume a lot of power or not. However, in building ENoP this is a critical issue. Moreover, ENoP is more sensitive to delays, as in NoCs we work with the nanosecond scale in comparison to the microsecond scale in large computer networks. In addition, since NoCs may be labeled by the application they run when made by some vendor, we realize a need to address an application in an inter-NoC platform. Hence, we designed the core-to-application interaction scheme, which we describe above. This mechanism does not exist in large-scale computer networks.

4.2. Inter-NoC Communications

Each of the Inter-NoC and intra-NoC mechanisms can have many protocols. We are not aware of any such solutions for NoC, and thus we developed our External NoC Protocol (ENoP) and Internal NoC Protocol (INoP). We anticipate that ENoP and INoP would be the trigger for the evolution of many other protocols for inter-NoC and intra-NoC communications. A major principle in inter-NoC communications when connecting two NoCs is to physically connect them via two links: a Main link, and a Backup link [75][82].

We do so in order to have redundancy so that we have a more fault tolerant system. In other words, if one of the links fails we have another backup link so that inter-NoC communications continue. The core connected to the Main link between two NASs is called the Main Border Core (MBC). The core connected to the Backup link is called the Backup Border Core (BBC). Figure 33 shows two connected NASs illustrating the concept of Main and Backup links. In this paper, we focus on the basic ENoP algorithms as an introduction to a novel approach for Inter-NoC communications.

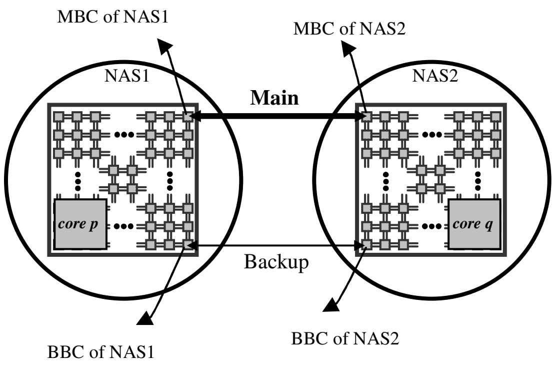


Figure 33. Two NASs interconnected using via a Main link and a Backup link. The choice of which connection will be the Main and which will be the Backup is made by the ENoP. Main and Backup connections may be wired or wireless. In Wireless Inter-NoC communications, each of the Main and Backup uses a different frequency channel to avoid collision.

4.2.1. ENoP Principles

The first ENoP principle is the initialization and neighborhood discovery. After a NoC turns on and stabilizes internally (which is taken care of by the INoP), the ENoP starts by checking whether this NoC is interconnected to one or more NoCs. Hence, ENoP initiates a NAS Neighbor Discovery (NND) algorithm, which is illustrated in Figure 34. The ENoP main algorithms are [79]:

1. NAS Neighbor discovery (NND)
2. NoC Authentication (NoA)
3. Message Format Agreement (MFA)

4. Global NAS Timing and Synchronization (GNTS)
5. External routing information exchange rules

In this thesis we focus on the NND algorithm as the neighbor NoC discovery constitutes the main event to be established before our biomedical applications starts. The other five algorithms can be simply simulated as messages transferring information in bits between the interconnected NoCs to agree on communications basics like message size.

4.2.2. Algorithms

Before NASs can interact, they must discover the NAS topology they belong to. ENoP is designed so that a NAS will automatically interconnect to another NAS or more. Hence some security issues arise. We are aware of the importance of security, but we do not investigate it in this thesis work. In this thesis we present when authentication shall happen, and that is after NoC neighbor discovery directly. The NND sends a probe message on its Main and Backup links to check if it is connected. To avoid collision with an incoming probe from a connected NoC, the NND waits for a random period of time before it sends the probe. This operation is repeated for a maximum number of times (N_MAX; line1, Figure 34), which is decided by the vendor. If, after N_MAX probes, there was no reply, then the NND considers that it is not connected to any NAS. By using random periods for the waiting time values we minimize the probability of collision in the situation when both interconnected NASs probe each other on the same line at the same time. NND considers the time of transmission, propagation, and processing of a probe message to be TS. Hence before it decides to probe again it waits a satisfactory period of 3TS (line 12, Figure 34). This period is more than the round trip time required for a probe to be sent and its reply to be received. This NND probing is done on both cores of the NAS, the MBC and the BBC. After probing and discovering the existence of a neighbor, connection is labeled as established. Then, the NND tries to authenticate this neighbor with a private *key*. Different NoCs may have different message formats and sizes. Hence, the message size and format should be agreed upon before ENoP starts sending inter-NoC traffic.

The most important parameter for message format agreement is the message size. We implemented this in our experiments using the minimum function *min*(*x,y*), which returns the minimum of *x* and *y*, as shown in (35).

$$\mathrm{M\_size}(\mathrm{NAS}_i;\mathrm{NAS}_{i+1}) = \boldsymbol{min}(\mathrm{M\_NAS}_i,\ \mathrm{M\_NAS}_{i+1}) \tag{35}$$

The ENoP goes through synchronization with the neighbors to be able to have snapshots of application states in case of failure so that these snapshots can be used.

```
Probes = N_MAX
Probe_Reply = 0
True = 1
False = 0
i= 0
Time_Slot = TS
PROBE_PACKET_Size = PS
While N_Probes < Probes do
   Get Random Non-zero-Digit  (RANI)
   Wait for RANI*TS
   Send (NIF=my_NIF, PROBE_PACKET(PS))
   Wait for 3*TS
   If Probe_Reply = True
      Send (NIF=my_NIF, NHA(my_hardware_address))
      Wait for 3*TS
       If Response = True
       i = i + 1
       NHAi = Read-NoC-Hardware-Address(i)
       Check_Authenticity (NHAi)
       If authentic(i) = True
          MFA(i)
          Synchronize(i)
          Main_Backup(i)
If Probe_Reply = False
      Then N_Probes = N_Probes + 1
```

Figure 34. NND algorithm discovering and authenticating a neighboring NAS then deciding the message format, synchronizing and choosing which link is Main and which is Backup [79].

We also ran a simulation for these protocols for a NoC that can be used or emergency cases. This sample is shown in Figure 35. This NoC may have different cores, some of which are connected to biomedical sensors and others that can run Speech recognition in order to check the patient's voice. One core can run a Digital Image Processing for medical images, and we have other memory cores. We also add an RF DSP part that can be left for wireless communications as proposed in [71][70].

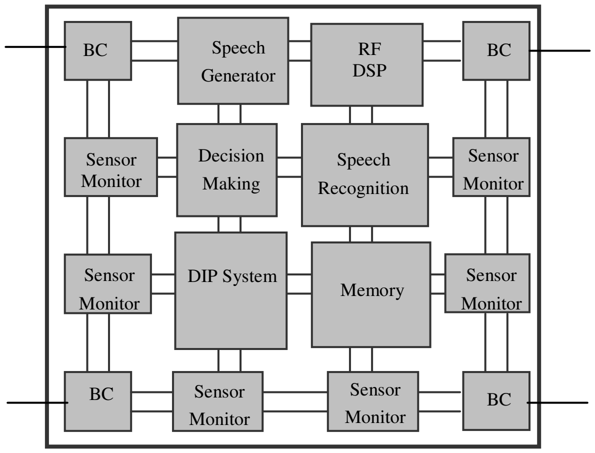


Figure 35. BIONoC design for the emergency case monitoring example.

4.3. The Simulation Environment

We ran a NAS topology simulation (using C++) for a medical emergency scenario, which we borrowed from biomedical scenarios using wireless Main and Backup links (with µs delays for IEEE802.11g transmissions at 54Mbps).

The simulator environment is shown in Figure 36, where the scenario runs communications between cores on 3 connected NoCs. Table 1 shows the results of simulation for only two interconnected NoCs.

These delays are used later to see how much margin is needed after the real time analysis of the ECG data in the multicore NoC.

Table 1. BIONoCSim++ results for real scenario of a two BIONoC sensor network

Scenario	**Time (ms)**	**Node of Alarm signal**
Emergency	31.71	On patient
Emergency	31.66	On patient
Emergency	31.30	On patient
Emergency	9.35	In equipment
Emergency	9.57	In equipment

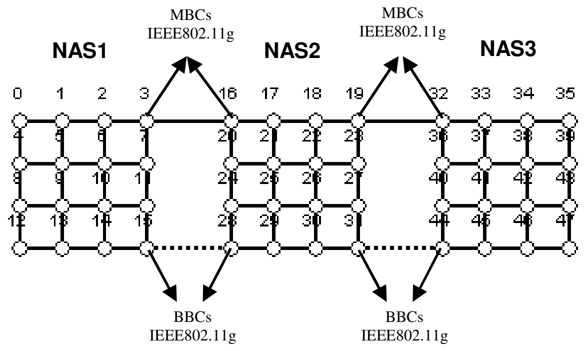


Figure 36. WNoCs simulation environment. Each NAS corresponds to a WNoC with 16 cores as described in Figure 35. The numbers over the nodes reveal the hardware address of each core of the 16 cores of the BIONoC shown in Figure 35. The MBCs and the BBCs run IEEE802.11g at 54Mbps.

The scenario for the emergency situation considers NAS1 to be wearable on the patient with its biomedical sensors connected to the patient. NAS2 (the middle WNoC in Figure 36) interconnects to the WNoC on the ambulance, and hence it is not self powered but rather uses the power input from the ambulance battery. The reason for this is that decision on calling the nearest available medical crew consume a lot of calculation and power. Therefore, we use the middle WNoC (NAS2) to make warning decisions to the medical team. However, the wearable WNoC is the decision maker if the patient requires direct care from the medical team. Consequently, NAS1 informs NAS2 that there is a need to route the warning signal to the nearest medical staff member via the suitable wireless channel chosen by NAS2. The third WNoC (NAS3) is given the task of deciding the best connection to the home healthcare center, i.e. it will run checks to find the best link quality from delay and packet loss points of view. Hence, on one end NAS3 connects to NAS2, and on the other end NAS3 connects to a long distance center (to a healthcare-center or hospital) via a commercially available communications link (Internet, Satellite, 3G, or GPRS, etc). NAS3 is also powered by the vehicular ambulance available. We simulate this scenario with three BIONoCs (Figure 36). Our simulations showed that the NoC discovery time is in milliseconds for the wireless link example of 54Mbps IEEE802.11g. As time here is a crucial parameter, we focus on looking at the time values for a "spread" of critical alarm from the moment that a calculated biomedical parameter (like a heart period failure) was sensed. The response time of the system was satisfactory from a medical point of view. Since warnings will be only generated from NAS1, we ran simulations that generate warnings from NAS1 only. The cores that are able to generate a warning on NAS 1 are cores 9, 10 and 13, which correspond to the decision making core, the speech recognition core, and the vision core.

The Decision making core decides based on the history of the monitored data. Nevertheless, in some cases the vision core or the speech core realize a critical case and do not waste time to send data to the decision making core, but rather send a global warning signal. An example of such a case where the speech signal recognizes a critical case is the snoring of a patient that may lead to suffocation while sleeping, especially if the patient was suffering already from a sleeping disorder or chronic disease.

Table 2. BIONoC simulation results oxygen level sensors of the environment shown in Figure 36

Convergence time (ms)	**Node of Alarm signal**
13.69	9; NAS1
13.69	10; NAS 1
13.53	13; NAS 1
7.46	25; NAS2
7.46	26;NAS2
7.30	29;NAS2

4.4. Summary

The biomedical algorithms and SW analysis carried out so far prove the improvements to abnormalities detection that heartbeat analyzers can enjoy. Our work is a first step in the direction of a full exploitation of MPSoC technology for this purpose, because a SW needs to be implemented before fitting later on the HW in the Codesign process. In order to better highlight the potential performance achievable with our SW, we coded and mapped the most computation-demanding ACF-based algorithm on a single PC first and on multiple PCs, and then we go a step further where we map it on an MPSoC platform in what follows in this thesis. With respect to the ECG algorithms discussed, we illustrate HW/SW design space exploration in Chapter 5 trying to move from algorithm-specific insights to general purpose indications for platform tuning. On the other hand, we presented algorithms to fit the biochip in a larger network and we, therefore, proposed the notion of a NoC as an autonomous system. We further developed this concept to design relative protocols for internal and external NoC communications.

Chapter 5

MPSoC Hardware-Software Codesign

"There's a way to do it better - find it."

Thomas A. Edison (1847 – 1931 A.D.)

In this chapter we go through the steps of the Design Space exploration and HW/SW Codesign for the MPSoC for ECG analysis. Section 5.1 introduces the need of MPSoC for such a solution. In section 5.2 we discuss the processing cores. Section 5.3 compares floating point and fixed point SW code for the ACF-based algorithm. Comparison between the computational resources of ARM TDMI and ST220 is elaborated with experiments in Section 5.4. Section 5.5 illustrates the experimental environment and MPSoC architecture for the solution. Hardware parallelism is discussed in section 5.6, where we elaborate the experiments, results, and scalability of the MPSoC solution. In section 5.7 we compare the solution with the state of the art in the market and in the research. Section 5.8 provides a sum up of the HW/SW Codesign work for the MPSoC.

5.1. Introduction

The main challenge in the 12 lead ECG application and algorithms (discussed in Chapter 3) arises from the high computational demand for processing huge amounts of ECG data in parallel, under stringent time constraints, relatively high sampling frequencies, and life-critical conditions [79][81]. The challenges become even more complex when the patient is mobile and remotely monitored (as in cases of homecare and emergency at the point-of-need as discussed in section 3.1) [68], because state-of-the-art biomedical equipment for heart monitoring lack the ability to provide large-scale analysis and remote, real-time computation at the patient's location. Unfortunately, limited processing power and tight power budgets of Holter devices have traditionally limited their functionality to data acquisition and recording of full ECG traces or abnormal events. Record analysis and diagnosis are performed offline at the medical center. Remote real-time ECG monitoring through dedicated telemedicine links necessitates the transmission of huge amounts of life-critical data over a communication link to a large set of computing devices on another location [31]. In cases of mobile patients ([81]), this requires a 100% functional always-ON wireless connection since losing a few heart-beat data may be life threatening.

The recent advances in silicon integration technology are paving the way for real-time ECG monitoring and analysis, thus allowing to promptly react to life-threatening heart malfunctions and to relax the requirements on telemedicine links. In particular, boosting clock frequencies of monolithic single-core microprocessors has clearly reached a point of diminishing returns and in the next few years performance gains will mainly come from an increase in the number of processor cores per chip. Although evidence of this trend is unmistakable in the domain of high-end microprocessors [182], the need to optimize performance per watt is likely to lead even portable devices to deploy multiple processor cores operating in parallel at lower clock speeds. Scalable chip multiprocessing is therefore emerging as a major design paradigm shift to provide scalable computation horsepower in a power efficient fashion. On one hand, this trend is accelerated by the stringent demands posed by increasingly complex software applications, but in turn it will force to radically revise programming models.

Heart activity monitoring and analysis provide a promising application domain for on-chip multiprocessor systems. The migration towards 12-lead ECG is facing computational challenges, especially in portable devices, where there is a need to meet tight computation and power budgets results in processing fewer leads. This is a limitation to the development of more sophisticated analysis algorithms, which are required to overcome the concerns posed by heartbeat analysis: physiological variability of QRS complexes, base-line wander, muscle noise, artifacts due to electrode motion, and patient mobility.

To overcome the aforementioned challenges and the problem of transmitting life-critical data on a wireless link (that is not reliable nor secure enough), the solution is to parallel-process the complex biomedical computations of the 12-lead ECG on a wearable multi-core system. Hence, the solution only transmits secure remote-alarm signals and only reports on the results of the analysis. These result-reports, are much smaller in size (a few bytes) than the ECG data (in Mega bytes), and if transmission fails, they can be re-transmitted until reception is acknowledged by the healthcare remote-monitoring center since they are saved on an off-chip memory for every analyzed ECG data chunk. Multi-

Processor System-on-Chip technology provides a good mean for the proposed solution. It consists of an energy-efficient and scalable hardware/software (HW/SW) platform that can meet computation demands at an affordable power cost. In this chapter, we introduce a novel MPSoC architecture for ECG analysis, which improves upon state-of-the-art mostly for its capability to perform 12-lead real-time analyses of input data with high sampling frequencies, leveraging the computation horsepower and the functional flexibility provided by many concurrent VLIW DSPs. This biomedical chip builds upon some of the most advanced industrial components for MPSoC design (multi-issue VLIW DSPs, system interconnect from STMicroelectronics, and commercial off-the-shelf biomedical sensors), which have been composed in a scalable and flexible platform. Therefore, we have ensured its reusability for future generations of ECG analysis algorithms and its suitability for porting of other biomedical applications, in particular those collecting input data from wired/wireless sensor networks. System performance has been validated through functional, timing accurate simulation on a virtual platform. A 0.13μm technology homogeneous power estimation framework leveraging industrial power models is used for power management considerations.

The first analysis was done to profile the execution of the code on the SW layer and to determine the best coding solution in terms of energy, execution time, and precision (in the HW layer). Furthermore, we have explored the design space searching for the best platform configuration for the 12-lead ECG data analysis. Alternative system configurations have been devised for different levels of residual battery lifetime, trading off power with accuracy.

5.2. Processor Cores

Our platform tuning effort was structured in three steps. Firstly, we determine the best coding solution in terms of energy, execution time and precision. Secondly, we look for the most efficient hardware platform configuration for the application at hand. Thirdly, we analyze the scalability of the platform to support future applications featuring a more aggressive parallelism. In doing so, we try to remove the communication bottleneck for the given number of processor cores; hence, we come up with a platform configuration which is again computation-limited.

5.3. SW Floating Point vs. Fixed Point Code

We ran two different code implementations: (a) one using *floating point* variables and (b) one using *fixed point integer* [16] with an exponent of 22. Figure 37 shows the results for the two different code implementations from *time* (*execution time*) and *energy* (*relative*) points of view. The ST220 processor core runs at 200MHz. We have performed the analysis for 3, 6 and 12 leads; furthermore we process each lead on a separate core.

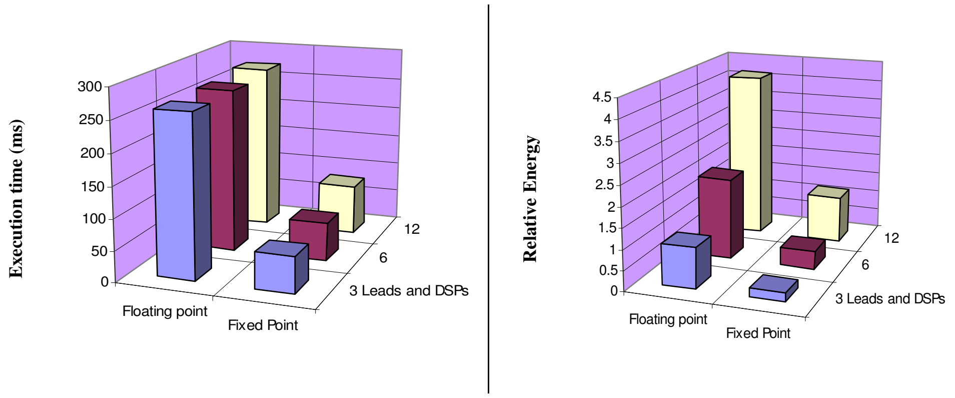


Figure 37. Comparison between different code implementations for the analysis of the 3-lead, 6-lead and 12-lead ECG. Data analysis for each lead is computed on a separate processor core. Sampling frequency of input data was 250Hz. System operating frequency was 200 MHz.

We found that the precision of the results obtained with fixed point code, by using 64 bit integer data types representation, almost matches the results obtained with floating point code for a large number of input data traces. On the contrary, the time needed to process data, and also the energy required, decreases up to 5 times. This is mainly due to the fact that, like many commercial DSPs, our processor cores do not have a dedicated floating point unit. Therefore, floating point computations are emulated by means of a C software library linked at compile time. Figure 37 also shows that even with 12 concurrent processors, the bus is not saturated, since we observe negligible effects on the stretching of task execution times. In contrast, adding more processors determines a linear increase in energy dissipation.

5.4. Comparison between Computational Cores

We then compared the performance of an ARM7TDMI with the ST220 DSP core, in order to assess the relative performance of the chosen VLIW DSP core with respect to a reference and popular architecture for general purpose computing, when put at work to process the computation kernel of our specific application. In order to have a safe comparison, we set similar dimensions of the cache memory (32KB) for the two solutions, and we run two simulations for the processing of one ECG-Lead at 250Hz sampling frequency. We count execution cycles to make up for the different clock frequencies.

We adopt this single-core solution, since our first aim is to investigate the computation efficiency of the two cores for our specific biomedical application, and de-emphasize system level interaction effects such as synchronization mismatches or contention latency for bus access. In Figure 38, we can observe that the ST220 DSP proves more effective both in execution time and energy consumption, as expected. In detail, the ARM core is 9 times slower than the ST220 in terms of execution time, and it consumes more than twice the energy incurred by the DSP. These results can be explained based on three considerations:

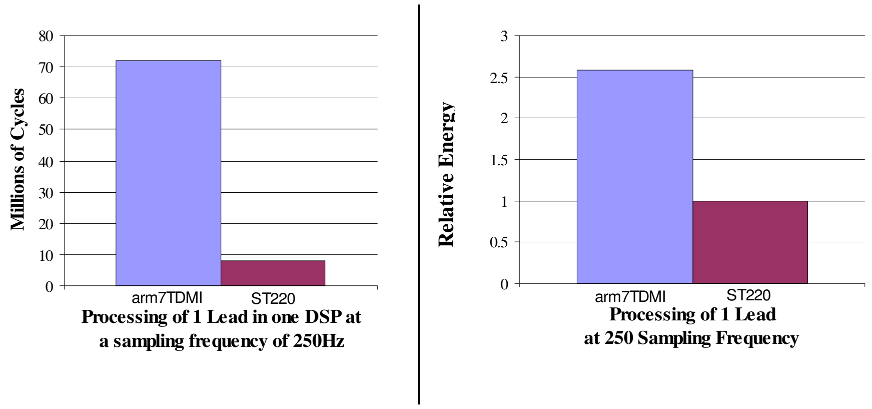


Figure 38. Comparing ARM7TDMI with ST200 DSP performances, when processing 1 Lead at 250Hz sampling frequency.

- The ST220 has better software development tools, which result in a smaller executable code. The size of the executable code for the ARM is 1.7 times larger than that of the ST220.

- The ST220 is a VLIW DSP core; therefore it is able to theoretically achieve the maximum performance of 4 instructions per cycle (i.e., 1 bundle).
- A metric which is related to both previous considerations is the static instructions per-cycle, which depends on the compiler efficiency and on the multi-pipeline execution path of the ST220. For our application, this metric turns out to be 2.9 instructions-per-bundle for ST220.

5.5. Experimental Environment and MPSoC Architecture

Based on previous findings in this chapter, we will adopt a HW/SW architecture consisting of the ST220 DSP core and a fixed point coding implementation of the algorithm for the experiments that follow. In order to process filtered ECG data in real-time, we choose to deploy a parallel MPSoC architecture. The key point of these systems is to break up functions into parallel operations, thus speeding up execution and allowing individual cores to run at a lower frequency with respect to traditional monolithic processor cores. Technology today allows the integration of tens of cores onto the same silicon die, and we therefore designed a parallel system with up to 13 masters and 16 slaves as shown in Figure 39. Since we are targeting a platform of practical interest, we choose advanced industrial components [119]. The processing elements are multi-issue VLIW DSP cores from STMicroelectronics, featuring 32kB instruction and data caches and with maximum clock speed of 400MHz. These cores have 4 execution units and rely on a highly optimized cross-compiler in order to exploit the parallelism. They try to combine the flexibility of programmable cores and the computation efficiency of DSP cores. This way, the platform can be used for applications other than the 12-lead ECG, so to make it cost-effective. Each processor core has its own private memory (up to 512KB), which is accessible through the bus, and can access an on-chip shared memory (8KB are enough for our target application) for storing computation results, prior to swapping to the off-chip memory. Other relevant slave components are a semaphore slave, implementing the test-and-set operation in hardware and used for synchronization purposes by the processors or for accessing critical sections, and an interrupt slave, which distributes interrupt signals to the processors. Interrupts to a certain processor are generated by writing to a specific location mapped to this slave core. The STBus interconnect from STMicroelectronics [3] was instantiated as the system communication backbone. STBus can be instantiated as both: shared-bus or crossbar (partial or full), thus allowing efficient interconnect design and providing flexible support for design space exploration and for platform scalability. In our first implementation, we target a shared bus to reduce system complexity (Figure 39) and assess whether application requirements can already be met or not with this configuration. We then explore also a crossbar-based system (Figure 40).

The inherent increased parallelism exposed by a crossbar topology allows decreasing contention on shared communication resources, thus reducing overall execution time. In our implementation, only the instantiation of a 3x6 crossbar was interesting for the experiments.

We put a private memory on each branch of the crossbar, which can be accessed by the associated processor core or by a DMA engine for off-chip to on-chip data transfers. Finally, we have a critical component for system performance which is the memory controller. It allows efficient access to the external 64MB SDRAM off-chip memory. A DMA engine is embedded in the memory controller tile, featuring multiple programming channels. The controller tile has two ports on the system interconnect, one slave port for control and one master port for data transfers. The overall controller is optimized to perform long DMA-driven data transfers and can reach the maximum speed of 600MB/s. Embedding the DMA engine in the controller has the additional benefit of minimizing overall bus traffic with respect to traditional standalone solutions. Our implementation is particularly suitable for I/O intensive applications such as the one we are targeting in this work.

In the above description, we have reported the worst case system configurations. In fact, fewer cores can be easily instantiated if needed. In contrast, this architectural template is very scalable and allows for further future increase in the number of processors. This will allow to run in real time even more accurate ECG analyses for the highest sampling frequency available in sensors (10,000Hz, and 15 leads, for instance). The entire system has been simulated by means of the MPSIM simulation environment [119], which provides for cycle-accurate functional simulation of complete MPSoCs at a simulation speed of 200Kcycles/second (on average), running on a P4@3.5GHz. The simulator provides also a power characterization framework leveraging 0.13µm technology-homogeneous industrial power models from STMicroelectronics [123]. The entire system has been simulated by means of the MPSIM simulation environment [119], which provides for cycle-accurate functional simulation of complete MPSoCs at a simulation speed of 200Kcycles/second (on average), running on a P4@3.5GHz. The simulator provides also a power characterization framework leveraging 0.13μm technology-homogeneous industrial power models from STMicroelectronics [123].

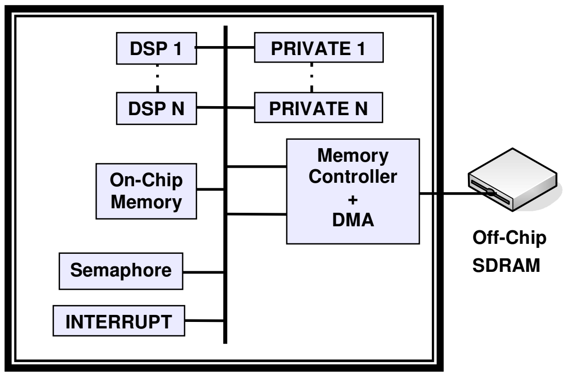


Figure 39. Single bus architecture with STBus interconnect.

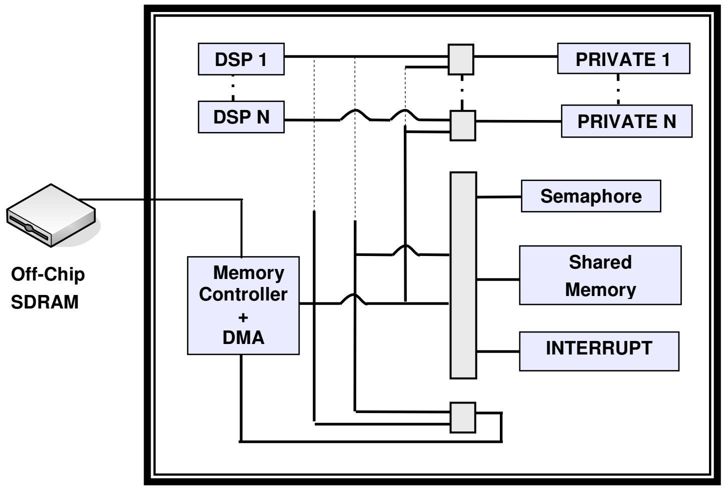


Figure 40. Crossbar architecture with STBus interconnect.

We believe that for life-critical applications, low-level accurate simulation is worth doing, although potentially slow, in order to perfectly understand system level behavior and have a predictable system with minimum degrees of uncertainty.

Each processor core programs the DMA engine to periodically transfer input data chunks onto their private on-chip memories. Moved data corresponds to 5 seconds of data acquisition at the sensors: 10kB at 1000Hz sampling frequency, transferred on average in 319279 clock cycles (DMA programming plus actual data transfer) on a shared bus with 12 processors. The consumed bus bandwidth is about 6Mbyte/sec, which is negligible for an STBus interconnect, whose maximum theoretical bandwidth with 1 wait state memories exceeds 400Mbyte/sec. Then each processor performs computation independently, and accesses its own private memory for cache line refills. Different solutions can be explored, such as processing more leads onto the same processor, thus impacting the final execution time. Output data, amounting to 64 byte, are written to the on-chip shared memory, but their contribution to the consumed bus bandwidth is negligible. In principle, when the shared memory is filled beyond a certain level, its content can be swapped by the DMA engine to the off-chip SDRAM, where the history of 8 hours of computation can be stored. Data can also be remotely transmitted via a telemedicine link.

In the above description, we have reported the worst case system configurations. In fact, fewer cores can be easily instantiated if needed but this is a first un-optimized HW design. This architectural template is very scalable and allows for further future increase in the number of processors. This will allow us to run in real time even more accurate ECG analyses for the highest sampling frequency available in sensors.

5.6. Required Level of HW Parallelism

Now, HW/SW optimization is investigated for a multi-DSP architecture processing a fixed point coding implementation of the ACF-based algorithm as an effective solution for ECG analysis. Now the problem of finding the minimum number of DSP cores to activate for the application at hand arises.

5.6.1. Experiments Description

We ran experiments in order to check the limits to respect the time figure of merit since our MPSoC is a real-time application based system. So we ran experiments to check the performance of each system design with increasing frequencies (up to 10 KHz). We also ran experiments to look for optimizing the algorithm together with the design by changing some algorithm parameters and looking into the overall performance of the specific biomedical application on each MPSoC design. We also ran experiments by distributing the application on different numbers of DSP cores for each design (shared bus, crossbar, and partial crossbar). The results of these experiments are presented below.

5.6.2. Experimental Results

We now want to optimally configure the system to satisfy the application requirements at the minimum hardware cost. We therefore measure the execution time and the energy dissipation for an increasing number of DSP cores in order to find the optimal configuration of the system. Since commercially available ECG solutions target sampling frequencies ranging from 250 to 1000 Hz, we performed the exploration for these two extreme cases for the 12-lead ECG signal. We analyze a chunk of 4secs of input data, which provides a reasonable margin for safe detection of heartbeat disorders. Note that the computation workload for the processor cores increases in a quadratic manner with increasing sampling-frequency (due to the specific application algorithm).

Figure 41 shows that if we increase the number of processors, the execution time scales linearly, which proves that second order effects typical of multi-processor systems (e.g., bus contention reducing the offered bandwidth to the processor cores with respect to the requested one) has only negligible effects on system performance, proving that the system is well configured and a single shared bus communication architecture is well suited for this application. However, this does not mean that the amount of data moved across the bus is negligible: around 100KB (at 1000Hz). This data is, however, read by the processor core throughout the entire execution time, thus absorbing only a small portion of the bus bandwidth. In this regime, the bus performance is still described as additive.

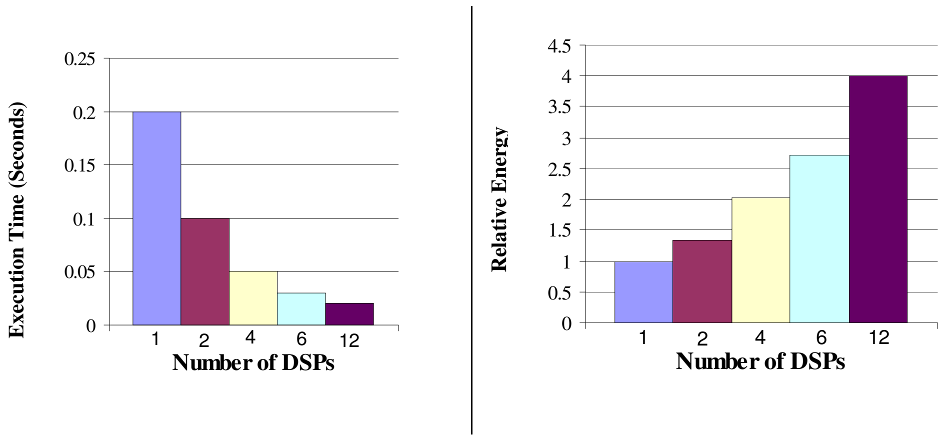


Figure 41. Execution Time and relative energy of shared bus at 250Hz sampling frequency.

Moreover, the perfect scalability of the application is also due to memory controller performance. In fact, at the beginning of the computation each processor loads processing data from the off-chip to the on-chip memory, hence, requiring peak memory controller bandwidth. The architecture of the memory controller proves capable of providing the required bandwidth in an additive fashion. By looking at the 1000Hz plot (Figure 42), we observe that 1 DSP is able to process an ECG-lead in slightly more than 3 seconds.

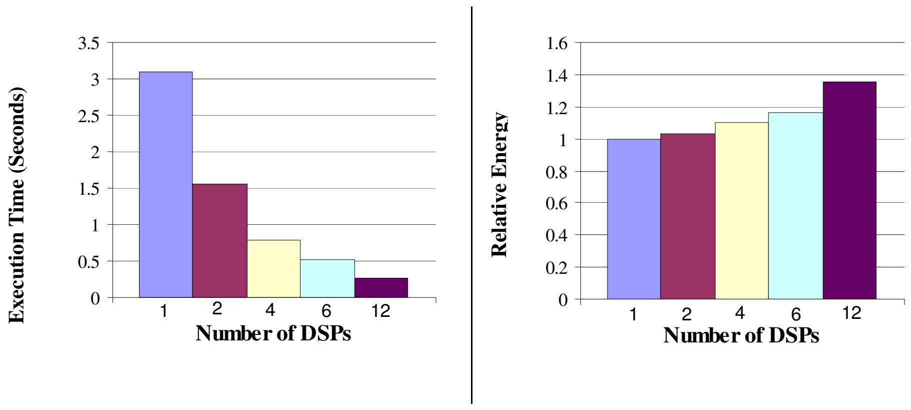


Figure 42. Execution Time and relative energy of shared bus at 1000Hz sampling frequency.

Therefore, we still have about 1 second left (before the 4secs deadline), which is enough to perform additional analysis of the results of the individual lead-computations and converge to a decision about the heart period and malfunctions.

5.6.3. Scalability

One main advantage of chip multiprocessing lies in its capability to meet the demand for higher sampling frequencies that are raised by the need to perform higher accuracy analysis and by the evolution of state-of-the-art sensor technology. In order to prove this, we conducted some experiments [80][81] and measured the maximum sampling frequency at which our MPSoC platform can be operated while still meeting real-time requirements of the ACF-based application. We consider a fully parallel 12-lead application spread over 12 processors. The resulting 2.2 KHz frequency indicates poor scalability. The reason for this is mainly the interconnect performance, which does not scale any more. In fact, bus busy (the number of bus busy cycles over the total execution time) at the critical frequency of 2200Hz is almost 100% (99.95%), i.e., the bus is fully saturated. This is due to the fact that the amount of data being transferred across the bus increases linearly with the sampling frequency. In order to make the platform performance more scalable, we revert to a full-crossbar solution for the communication architecture. The benefits are clearly observed in Figure 43, where the maximum analyzable frequency (with respect to real-time constraints) amounts now to 4000Hz, i.e. nearly twice as much as the performance with a shared bus. Moreover, we observe that average bus transaction latencies at the critical frequency are still very close to the minimum latencies, thus indicating that the crossbar is very lightly loaded. Another informative metric is the bus efficiency (number of cycles during which the bus transfers useful data over the bus busy cycles), which amounts to 71.83%.

We simulate the 12-processor system to get the upper bound on system performance and push the architecture to the limit. For the same reason, we restrict the analysis period to 3.5secs, which is the minimum value of the input data-chunk time-span derived from the biomedical algorithm. The results presented in Figure 43 show that with shared-bus architecture, the maximum sampling frequency that the MPSoC can handle without going beyond the real-time constraint is only around 2200Hz. This good performance is an effect of the lack of contention on the crossbar branches, which is in turn due to the high performance of the memory controller and to the matching of the application traffic patterns with the underlying parallel communication architecture. As a consequence, with a full crossbar the system performance is no more interconnect-limited but computation-limited. Since the computation workload of the system grows in a quadratic manner with the sampling frequency, it rapidly increases task execution times and reduces the available slack time with respect to the deadline.

We observe that the performance with a partial crossbar closely matches that of a full-crossbar (less than 2% average difference) but with almost 3 times less hardware resources. We found the optimal partial crossbar configuration (5x5 instead of a 13x13) by accurate characterization of shared bus performance. On a shared bus, we increased the number of processors and observed when the execution times started deviating as an effect of bus contention. With up to 4 cores connected to the same shared communication resource, this latter is able to work in an additive regime.

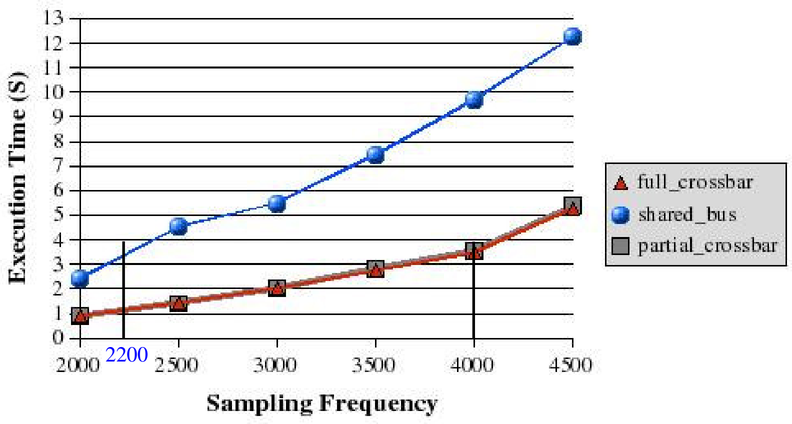


Figure 43. Critical sampling Frequencies for 3 architectures: (1) shared bus, (2) full crossbar, and (3) partial crossbar. The 3.5seconds real-time limit can be respected and achieved with 2200Hz sampling frequency when the Shared Bus architecture is used and with 4000Hz sampling frequency when either the Full crossbar architecture or the Partial Crossbar architecture is used.

Although the architecture cannot work in real-time at more than 4000Hz, we wanted to measure the execution time under non-real-time computation. In fact, from the execution time to process 3.5secs of input data, we can derive the amount of buffering that is required to store incoming data from the lead. By knowing the overall capacity of the off-chip memory, we then derive the maximum analysis time that we can afford at such high frequencies. Results are shown in Figure 13. Let us focus on the shared bus case, with 10 KHz. The execution time for analyzing 3.5secs is a bit less than 1 minute (around 57secs), which is 16 times more than the real-time constraint. As a consequence, we can still decide to perform this kind of analysis, but we need to buffer 16 input data chunks while we are processing one chunk. Since an off-chip SDRAM memory can be 512MB, we can perform 3.5 minutes analysis before saturating the memory. With a full crossbar, this time amounts to around 14 minutes.

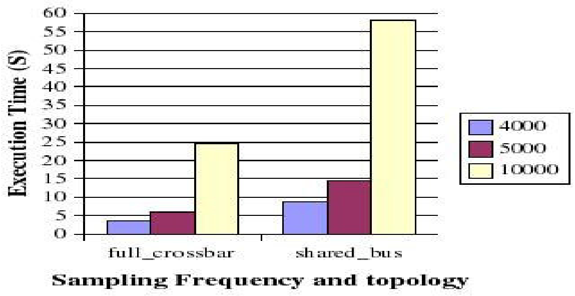


Figure 44. Analysis-time investigation with increasing frequencies (up to 10 KHz) for the ECG application.

An interesting remark is that using multiple cores to decrease execution time and have more margins for online diagnosis incurs a higher energy cost for the 250 Hz case than the 1 KHz case. This is due to the fact that the increased workload in the 1 KHz case justifies an increased hardware parallelism. The energy-ratios plot in Figure 45 confirms that the overhead for introducing more processors is worth in the 1 KHz case, while it is not fully justified for the 250Hz case due to the different computation complexities to be tackled.

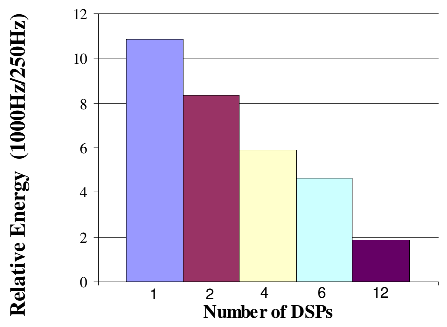


Figure 45. Relative Energy Ratios between the 1000Hz and the 250Hz sampling frequency experiments.

5.7. Solutions Comparison

We compare our platform to the existing solutions from an application point of view and SoC point of view. From the SoC view point, the advantages of our design are mainly in the performance for real time analysis and power consumption, where our experiments show that we can converge to a full analysis in a time less than the time needed to read the new coming data from the heart. More precisely we compare the cited background work in Section 2.2.2 on HW systems and we focus on the category of previous work that is nearest to our work.

In [1] the solution can only aid the application to distinguish between healthy or unhealthy patients, while in our work, the HW solution supports full analysis. Although they run their calculations on chip, the calculations done are simpler and do not aid in full ECG analysis. Moreover, there is no study on the best filtering method to use. The chip can take only one signal at a time. However, our solution can run the 12 leads at the same time. There is no time study given about the delays and how their HW may fit for real-time analysis.

The work in [13] tests the system using artificial ECG signals that are corrupted with various noise types, while in our tests we used real ECG data. Our solution and the one in [13] both have timing as a figure of merit. In addition the authors use two real testing cases only from the MIT-BIH records, but the two tested cases have low time resolution

(f_s=360Hz), while in our work, we used very high frequencies and we tested the whole MIT-BIH records. Their QRS error-detection rate is up to 2.2 % in the tested schemes. They use only one microcontroller in the detection device, and the bulk of processing is done on the PC, while our HW solution is all on chip. Thus this is not a portable device. They rather need to transmit the whole ECG signal to the PC through the RS232. The detector software code was written in HC16 assembly code, i.e. it is cumbersome, time consuming, and error prone if not tested rigorously. Their software update is in assembly, i.e. compared to our solution it harder since our application is totally based on standard C language. Moreover, the work in [13] uses a pre-filtering procedure like we did, to increase the reliability of the detection algorithm. However, their filtering procedure is followed by a Matched Filter. Thus the MF convolves the received signal with some predetermined response (the matched response of the ECG). In other words, the MF assumes a unique ECG for all persons, most probably a normal one, and then add Additive White Gaussian Noise (AWGN)[6] to it to take the effect of noise before the convolution process takes place. Thus, they use a probabilistic approach. This is very similar to the auto-template procedure presented in [159]. However, the auto-template procedure is more reliable, since it tries to match the response of the incoming signal to one of several signals available in the database, and not a unique one as presented in [13]. From a frequency viewpoint, the work in [13] uses a sampling frequency f_s=500Hz. However, to provide a better timing resolution, the MF output is linearly interpolated at 4f_s=2000Hz. The MF is followed by a decision device (Dual edge more accurate than the traditional decision device). The decision device uses an adaptive threshold, 40% of the highest amplitude component. Comparing to our work, we go beyond the 500Hz and 2 KHz frequencies and we even study up to 10 KHz. We also use an adaptive threshold to find the R-Peaks after finding the period. The authors also use an artificial ECG signal, based on approximations, while we use real life ECG leads data. The work in [13] firstly finds the R-R timing metric and uses it to find other figures of merit on the PC. Comparing to our work, we notice the major difference, where we find the period first, based on the proposed algorithm, and thus the R peaks can be determined more accurately. The authors of [13] detect the signal with timing errors, where more than 30 % of the pulse period is neglected from the analysis.

[6] Additive White Gaussian Noise is a random noise characterized by a wide frequency range relative to the signal.

The PC also sends configuration parameters for the detector like the MF impulse response. The PC contains an algorithm that can detect and disregard false measurements. In our work there is no PC involved, as our HW/SW Codesign fits the whole analysis of the data on the chip. From a power point of view, we see that their detector device needs an equivalent to the 1000mAh Ni-Cd battery. However we cannot compare ours to them, because we implement different functionalities on our chip.

In [28] none of the available prototypes or commercial solutions provides a combination of allowing being wearable, small size, and functionality of having body motion and physiological parameters to be investigated simultaneously. The initial prototype was built around a PDA (Compaq/HP iPaq Pocket PC), with Microsoft Windows CE. The system provided 2-lead ECG and respiration signals, which is not enough for full analysis. It is linked to the PDA via a Bluetooth link. The PDA then uses IEEE 802.11b[7] to transmit data in real time over the internet to a server. It is true that they run real time data transmission, but they do not run real time analysis at the point of need (on the patient location). The drawbacks also are that the system does not have robust connectors and it has short battery life. The paper proposes a multi-parameter monitoring and transmission device, one parameter of which is the 2-lead ECG, but the paper does not propose any analysis, while we use 12-lead ECG and real-time analysis. They could use the PDA for more powerful analysis. They focus on the underlying hardware (the different used components) and describe different testing environment that they have done (lab testing, field testing, in water), while we focus on the underlying analysis algorithm rather than the sampling and transmission

The authors in [159] run detection of abnormal QRS intervals, which can be fit on many HW types of DSP, ASIC or FPGA. They run the analysis in three steps: (i) filtering, however, they do not mention the type of the filter used, and they only mention the use of MATLAB, (ii) auto-template, which they store as a normal ECG signal to be able to compare with, and (iii) correlation of the incoming buffered signal with the template. This technique assumes a unique normal ECG for a given triggering threshold (or guarantees a similar ECG). Hence, unlike our work, it does not give account for some variability from person to person and does not present proof on that. In the auto template process, the normal ECG is formed based on the input signal. Therefore, the question remains on what the analysis would be when the input ECG is abnormal. The algorithm proposes real time correlation but the correlation process has to wait for the auto-template process to finish, which analyzes the whole sample before forming a normal ECG. Thus it adds more time overhead. So the whole application ends up not being a real time application, unlike the solution we propose. Moreover, the solution in [159] can only tell normal form abnormal heart beats, while we do full analysis. The authors use correlation with a normal signal, while we do autocorrelation to find the period and rhythm. They can not tell the type of arrhythmia, and they use it for 1 lead only, while we run 12 lead analysis. They use a 1GHz PC together with the chip, while we only use the MPSoC.

The work presented in [169] is only focused on the measurement and transmission, but is interesting for our work from the point of view of fitting the biochip within a general telephony or communications network.

The work in [173] is interesting from the point of view that it addresses one of the current needs for transmitting emergency medical signals to healthcare centers. However, they analyze only 2 leads instead of 12 leads. Moreover, we use only a single chip, i.e. less power consumption for the 12 leads. They digitize the signal at 10bit while we use 16 bits.

The work in [113] is interesting since they use ARM7TDMI microprocessor unit with 64KB RAM (optimized for cost and power-sensitive consumer applications), while we have used this ARM7 processor in a phase to look for the best processing core for our specific application. They acquire a single lead signal, while we run 12 leads. They run on a resolution of 12bits, while our design uses 16bits. The sampling rate used is 250Hz while our sampling rates range from 250Hz to 10 KHz. They use a Butterworth filter with 3 poles and a tunable cut frequency, while we use IIR filters. The system is composed with 2 blocks: an ECG pre-processing board and an 8-channel DSP. The solution uses an external ADC, because it affects less in terms of resolution and noise. The system does not transmit continuously.

We also choose the work in [60] and [116] to compare in more details with our work since they are the nearest solutions to ours from commercial and research viewpoints,

[7] IEEE 802.11b is one of the Wireless Local Area Network standards allowing for 11Mbps transmission speeds.

respectively. The results of comparison are shown in Table 3. From the application side, we look at the comparison with nowadays applications used on SoCs after we have compared it with the nowadays most used algorithm, the Pan Tompkins, in Chapter 3.

Table 3. Comparison between ECG analysis research SoC solution [116], available SoC commercial solution [60], and our three MPSoC designs: ST-PCB is partial crossbar, ST-CB is Full Crossbar, and ST-SB is the Shared Bus solution. f_S is the sampling frequency

Solution	**Data Bits**	**Memory**	f_S **(Hz)**	***Analysis* Time**	**SoC Input**	**Filter**	**Application Results**
[116]	10	8KB Cache	250	No study	1	Notch	Analysis on only QRS Complex, & only decides if the heart is healthy or unhealthy
[60]	12	No available information	800	No study	8	IIR	Analysis on only QRS Complex, & only decides if the heart is healthy or unhealthy
ST-PCB	16	Off chip = 512MB 32kB I-cache/DSP 32kB D-cache/DSP	4,000	< 3.5s	12	IIR	Full 12-LEAD Analysis: heart-period; P, Q, R, S, & T peaks; & **detect disease**
ST-CB	16	Off chip = 512MB 32kB I-cache/DSP 32kB D-cache/DSP	4,000	< 3.5s	12	IIR	Full 12-LEAD Analysis: heart-period; P, Q, R, S, & T peaks; & **detect disease**
ST-SB	16	Off chip = 512MB 32kB I-cache/DSP 32kB D-cache/DSP	2,200	< 4s	12	IIR	Full 12-LEAD Analysis: heart-period; P, Q, R, S, & T peaks; & **detect disease**

Comparing our application-based MPSoC designs, we can choose the best architecture relative to the biomedical purpose. Hence, for a solution that competes with and performs better than the existing commercial solutions [60] (input sampling frequency from 250 to 1 KHz), we adopt our shared bus system architecture (Figure 39) since its advantages over existing solutions are:

- ECG SoC available solution nearest to ours is [116], but our design performs better on analysis time-consumption. Our solution is easier to deploy since the 12 leads are input to one SoC instead of having 12 SoCs (i.e. cheaper to scale with increasing leads)
- In our designs, we can do full 12-lead ECG analysis at relatively high frequencies. Our design is optimal especially that we can offer the choice of the SoC architecture: shared bus, full crossbar, or partial crossbar. Our solution allows that this decision be based on the biomedical need of a frequency range. Table 3 shows the advantage of our three designs to the best available designs we are aware of in the research and in the market.

5.8. Summary

We listed the different steps in design space exploration for an application-specific MPSoC architecture for real-time ECG analysis as a solution for the largest medical problem from the point of view of leading to the highest number of deaths in the globe (CVD and Stroke). Our solution leverages the computation horsepower of many (up to 12) concurrent DSP cores to process ECG data. This solution paves the way for novel healthcare delivery scenarios (e.g., mobility) and for accurate diagnosis of heart-related diseases. We describe the design methodology for the MPSoC and explore the configuration space looking for the most effective solution, performance and energy-wise. We present three interconnect architectures (Single Bus, Full Crossbar, and Partial Crossbar) and compare them with existing solutions. The sampling frequencies of 2.2 KHz and 4 KHz, with 12 DSPs are found to be the critical points for our Shared-Bus design and Crossbar architectures, respectively. We compare our solution with the nowadays existing solutions in research, in the market, and in many hospitals. Our solutions offer significant advantages in SoC HW/SW Co-design, and it proves no single failure case when run and tested on the standard MIT/BIH arrhythmia database, while the widely used Pan Tompkins solution failed in some cases, of which we reveal two cases.

We also illustrate the potential advantages that MPSoC technology can bring to real-time ECG analysis. They can be summarized as follows: larger time margin to run diagnosis algorithms, energy efficiency (provided DVFS support is available, and which is left for future work), improved scalability to challenging higher sampling frequencies and to more accurate ECG analysis algorithms.

Chapter 6

The Resulting Methodology

"There are no rules here - we're trying to accomplish something."

Thomas A. Edison (1847 – 1931 A.D.)

This chapter builds upon the previous results and the experience through the design and implementation work and the high number of trials and errors made. We then start to put the structured steps of a methodology to follow when having a high computational application (especially in biomedical engineering) to integrate as a Software on a specific Hardware. Section 6.1 introduces the work on methodology. In section 6.2 we present the view on the virtual platform for co-simulation. Section 6.3 stresses on the pedantic methods needed for life critical applications within a HW/SW Codesign. High task level parallelism is also briefly discussed in section 6.4. A point will always be reached when working on multicore applications with high computation, and that is the communications bandwidth requirements (section 6.5) and the computation vs. communication parallelism (section 6.6). Section 6.7 briefly summarizes the resulting SW and HW. Finally, section 6.9 summarizes the chapter.

6.1. Introduction

In the embedded systems field, many design methods require to specify hardware and software separately. However, in our case, HW, SW, and the interaction in-between are what lead to a successful performance of the system. In other words, in the end we care about the performance of the whole solution, i.e. the interaction between the hardware and software to give the final output. Moreover, the final output in our system is a life critical one.

The specific application domain addressed in this work calls for specific design methodologies for which full system modeling accuracy is critical, since the resulting architecture needs to be highly predictable. Uncontrolled run-time fluctuations of the system might lead to incorrect detection of life-threatening events in the heartbeat traces.

For this purpose, the most daunting challenge for system designers perhaps consists of accurately predicting the impact of the communication and I/O architectures on system performance. Even more critical is the ability to capture the interaction between these two sub-systems. Our architecture allows to compare existing analysis algorithms and to develop new ones, while keeping low-level implications of software decisions under control through the support of a virtual platform.

6.2. Virtual Platform

When we view the target applications of healthcare domain in relation to the candidate hardware platforms for it, the following requirements become evident: (i) life-critical application, (ii) high task level parallelism, (iii) communication bandwidth requirements, and last but not least (iv) computation and communication parallelism. We discuss each of the aforementioned requirements separately and integrate them within a methodology that can be used to guide HW/SW Codesign for application specific multicore systems (MPSoC).

6.3. Life-Critical Application

Contrary to other domains, such as multimedia and entertainment, we need to come up with *accurate* system component models already in the early design stages, so to limit the degrees of execution unpredictability to the minimum. Therefore, transaction-level modeling did not seem to us as the best option, but we rather focused on *cycle-accurate modeling and simulation*.

6.4. High Task Level Parallelism

Each individual lead signal can be potentially processed in parallel with minimum inter-task interaction. Only in the end of the data processing phase, such interaction is eventually needed by diagnosis tasks to reveal heartbeat abnormalities and to analyze them. Therefore, a *parallel hardware platform* is well suited for this kind of algorithms.

6.5. Communication Bandwidth Requirements

Regardless of the specific ECG analysis algorithm, large amounts of data have to be stored in the system memory to keep the records of heartbeat activity over a long observation window. This involves a continuous flow of data towards the off-chip memory. Moreover, incoming new sampled data to be processed must be considered as well in order to assess the communication bandwidth requirements. From a hardware viewpoint, there are two important implications. On one hand, processor core-to-off-chip memory communication is becoming a major bottleneck in modern MPSoC platforms. Optimization of memory access of multiple on-chip computing cores to the finite bandwidth of off-chip memory is a key point to the success of parallel MPSoC architectures. This calls for an incremental effort for *accurate modeling off-chip memory controllers*, including their *optimization engines* and interfacing mechanisms. On the other hand, even admitting unlimited I/O bandwidth, the communication architecture turns out to be the most critical component for system performance. In the context of parallel MPSoCs, such performance is in fact not computation but rather *communication limited*. It is, therefore, necessary to accurately model this component (we chose *cycle- and signal-accuracy*) in order to assess the scalability of the hardware platform at hand. Finally, having independently optimized communication and I/O architectures might be misleading, since the *interaction* between specific features of the communication protocol and a many-to-one traffic pattern might degrade ideal performance of both components [158]. Finally, optimizations at the memory controller might make this interaction unpredictable unless an accurate virtual platform is set up.

6.6. Computation and Communication Parallelism

Storage of processing results and the next computation round should be handled in parallel for efficient execution. Therefore, the deployment of DMA engines is highly attractive in this case. However, two conflicting requirements should be taken into account. On one hand, storage data size needs to be kept to the minimum to extend the observation window (given the finite memory size), while on the other hand sufficiently long data transfers are required to amortize the cost of DMA transfer programming.

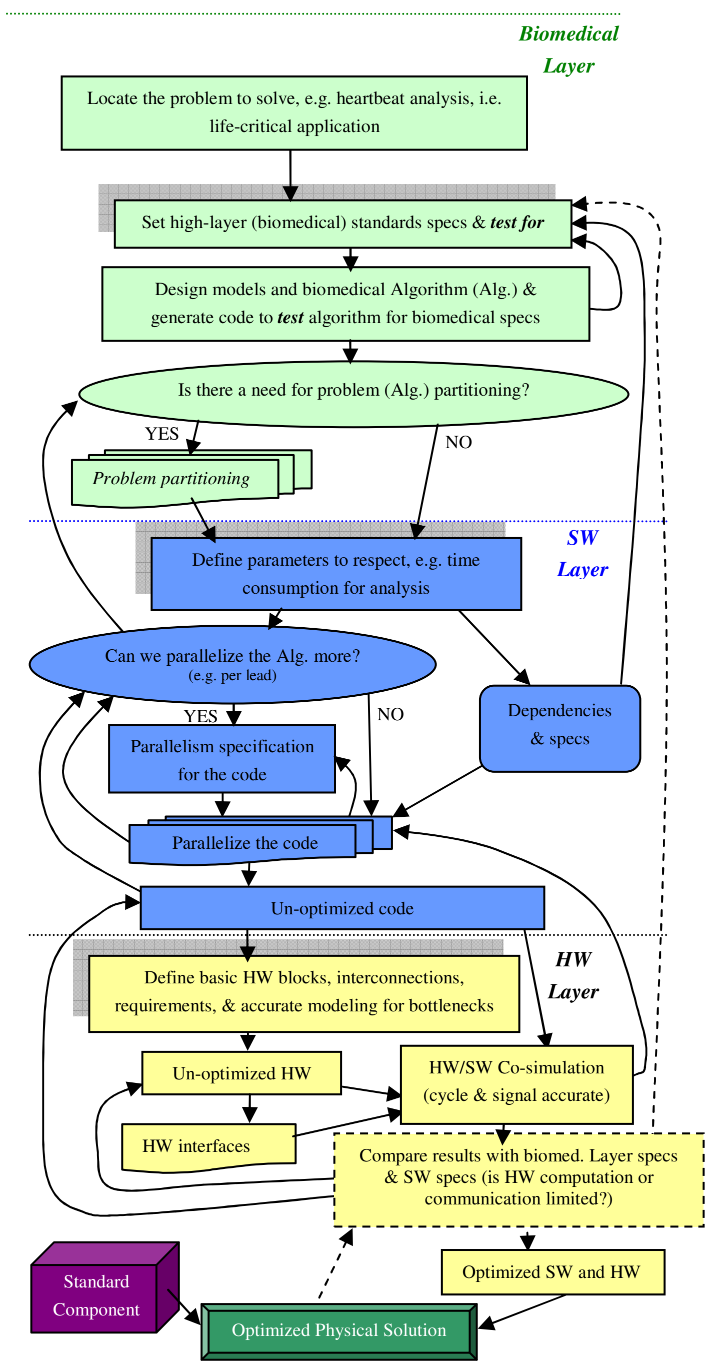

Biomedical Layer
Locate the problem to solve, e.g. heartbeat analysis, i.e. life-critical application
Set high-layer (biomedical) standards specs & test for
Design models and biomedical Algorithm (Alg.) & generate code to test algorithm for biomedical specs
Is there a need for problem (Alg.) partitioning?
YES
NO
Problem partitioning
SW Layer
Define parameters to respect, e.g. time consumption for analysis
Can we parallelize the Alg. more? (e.g. per lead)
YES
NO
Dependencies & specs
Parallelism specification for the code
Parallelize the code
Un-optimized code
HW Layer
Define basic HW blocks, interconnections, requirements, & accurate modeling for bottlenecks
Un-optimized HW
HW/SW Co-simulation (cycle & signal accurate)
HW interfaces
Compare results with biomed. Layer specs & SW specs (is HW computation or communication limited?)
Optimized SW and HW
Standard Component
Optimized Physical Solution

Figure 46. Methodology used for high-end biomedical application HW/SW co-design (applied for ECG analysis).

In a step further to implement the HW/SW co-design based on the above analysis, we are pushed to found a guiding (flow of logical steps) methodology that we refer to in order to keep on optimizing the life-critical *biomedical algorithm*, *SW*, and *HW*. This type of methodology was induced throughout the work of the design and implementation, which we present in a flow of steps and their inter-relations for HW/SW co-design for high-end biomedical applications. This methodology is summarized in the flowchart presented in Figure 46.

After defining the requirements, we divide the multidisciplinary problem into three layers: (i) application layer (biomedical problem), (ii) software layer (algorithms and coding), and hardware layer (HW Design, co-simulation, testing, etc). These divisions are demonstrated in Figure 46. In brief, the starting point is always in finding the right application (problem) that has some level of significance like a life-critical application of heart ECG analysis in real time with high sampling frequencies (more than 1KHz). At this stage and at every stage, we need to keep in mind and thus in the design flow the relationship between different parameters in different layers and the effects on other layers and stages.

A note worth mentioning is that at this stage some high layer specs are defined. For instance, the value of the normal period and the maximum period length that could be available are set. This will be very essential in defining the value of the lags *L*, discussed in Chapter 3. To know what *L* can be for our SW part, we need to know the minimum length of the signal that we need. Moreover, we find a major need to look at whether the biomedical problem can be partitioned (e.g. period analysis and peak analysis for ECG signals). This defines one sort of parallelism that is reflected in the SW code in a later stage. Then we design the required algorithm, and we generate the relative code to test it. Once the algorithm is done, a crucial step is to see whether the algorithm (application) can be partitioned (12 different parts, each responsible for a different lead).

In the SW layer, we define the parameters which we have to respect in the analysis, for instance the real-time limits in seconds especially in relationship to sampling frequency. In addition, the value *L*, and the size of the data chunks (which is dependent on the time specs in the biomedical layer) are decided in this layer. Further changes in these parameters may occur based on the Codesign step. After these SW specs are defined, we look further into partitioning the code itself. This SW partitioning is different from the application partitioning since it looks at the code itself. Both, the application partitioning and the SW partitioning affect the performance of the whole HW/SW system from a time and power consumption viewpoints. Once the first parallelized code is generated and tested (on one processor), we need to define the code dependencies like the size of data-chunks, their specifications about the number of samples per data-chunk and the length of recording, and where and how these chunks shall be stored (e.g. binary format). At this stage, we succeed in having a working code, but that does not necessarily mean this may be the optimized one for the specific HW computation and communication requirements. Hence, we can start a first test on different cores to see the effect on the code, and in our case, we ran codes based on floating point and on fixed point integers with exponent 22 on ST220 at 200MHz, and we found out that with fixed point integers (64bit integer data types), we have the same precision as floating point code, but the processing time and the energy consumption are decreased 5 times as shown in Figure 37, This calls for the use of fixed point integer for instance.

The parallel code itself, its dependencies, and the application limitations, help inspire the HW designer to choose and correlate HW blocks for accurate modeling. The result is an un-optimized HW with interface mechanisms that is ready to load the SW to go further to the HW/SW co-simulation stage. This first HW design means that many components may change, and many on-chip communication network designs may also change based on the co-simulations. For instance, we ran the first non-optimized code on ARM7TDMI and ST220 with similar dimensions of cache memory (32KB) to see for the best HW component that can aid our goals. The ARM core is 9 times slower than the ST220, and consumes more than twice the energy (Figure 38). Since this is a life-critical application, it requires cycle and signal accurate co-simulation and can run much iteration before reaching the desired results. The results of the co-simulation are the major outcome to test and compare with specs in above stages in higher layers. A note worth mentioning is that in the HW architecture, we aim at removing HW bottlenecks and SW bottlenecks. Moreover, we aim at design space exploration while looking for input sampling frequencies that range from 250Hz to 1 KHz and higher. Since we deal with a life critical application, we must go back to the SW and application layers to pedantically test for the higher layer specs and limitations. These tests may affect the optimization of both: SW and/or HW. Once the test results are acceptable, we can take the HW/SW Codesign that is considered optimized and allows further physical development.

Based on all the above analysis, we came up with a virtual platform with mixed modeling abstractions so to trade-off accuracy with simulation time. The virtual platform we use for the co-simulations is the MPSIM environment [119] as a starting point for intensive extension and customization for the application. SystemC is used as a descriptive language for the simulation engine due to its ability to perform efficient HW/SW co-design. The processor cores model was at the level of their instruction set. Chip architecture was modeled with clock-cycle accuracy extended to the chip boundaries. While retaining this level of accuracy, processor cores were modeled at the level of their instruction set, while only the on-chip bus was modeled with signal accuracy due to its criticality. As an effect, we were able to achieve simulation speeds up to 200 KCycles/sec on a P4@3.5GHz. Industry-provided and technology-homogenous 0.13µm power models of all system components were deployed for power analysis [3] [123].

6.7. Example of a Resulting HW and SW Architecture

When using this methodology we ended up with a software architecture that is fully distributed and an independent RTEMS [18] operating system runs on top of each processor core. Kernel images reside in the private memory of each core. A middleware support for inter-processor communication was ported to the system, and a special purpose library was used for DMA programming [59]. Based on the findings in going through the methodology, HW/SW optimization can be investigated.

We found that when sampling ECG data from biomedical sensors at 250 Hz, 1 DSP core is enough to meet the real-time requirements of the ACF-based SW-method (12 leads analysis): largely meeting the required 4 seconds for real-time computation of input data frames. Interestingly, even increasing the 1 KHz sampling frequency, and therefore increasing the computation workload of processor cores, 1 DSP is still able to complete the 12-lead analysis in slightly more than 3 seconds. This leaves about 1 second left to run diagnosis algorithms online (as a partition of the main algorithm). The increased sampling frequency led to a 15x increment of execution time and to a 90% increase in overall system energy. By increasing the number of processor cores in both cases, we got good scalability results (relative to execution time) up to 6 cores, with diminishing returns with increased parallelism. Unfortunately, the energy increases, since the power overhead of having more cores running is not offset by execution time savings. This was expected, for the deployed processor cores are not general purpose processors and are therefore more power-consuming.

However, more processor cores can be used if the 1 second margin is not satisfactory, or if the platform is augmented with dynamic voltage and frequency scaling (DVFS) support (this is work in progress). This latter solution would make the multiprocessor solution efficient not only from a performance perspective, but even from an energy viewpoint. The large slacks that are available in the current implementation make application of DVFS very promising and a conclusion we get from using such methodologies.

Interestingly, using multiple cores to decrease execution time and have more margins for online diagnosis incurs a higher energy cost for the 250 Hz case than the 1 KHz case discussed in Chapter 5. This is due to the fact that the increased workload in the 1 KHz case justifies an increased hardware parallelism.

6.8. Methodology in Brief and Tradeoffs

Below we summarize the basic points to keep in mind in the process of HW/SW Codesign while working with our proposed methodology:

- Design of an algorithm is made by understanding the application and the existence of a new technology of MPSoC
- Understanding of the Number of Processors and Parallelism
- Communication Interconnect structure
    - Shared bus: System interconnect saturation limits performance scalability (100% busy at 2.2 KHz)

- Full crossbar: Parallel topology removes scalability limitations, doubled system performance, average and minimum bus transaction latencies are close to each other, and high bus efficiency: 72% of the bus busy time
- Partial crossbar: almost equals full crossbar performance with almost 3 times less hardware resources: 5x5 instead of 13x13

6.9. Summary

This methodology work goes through an application-specific design for a multicore system. The application we focus on is a high computational one with a stress on the life critical point since it is related to human healthcare. The ECG application domain gives this high computational demand and the high level of accuracy at work. The methodology is able to help envision, design, develop, and test full system modeling accuracy on three layers (application, SW, and HW). It also enables us to obtain high HW/SW parallelism exploitation, computation parallelism, and communication parallelism. Finally, we illustrate, for practical case studies, the advantages of deploying more computation-demanding analysis algorithms for the quality of ECG analysis.

Chapter 7

Conclusions

"I think and think for months and years. Ninety-nine times, the conclusion is false. The hundredth time I am right."
Albert Einstein (1879 – 1955 A.D.)

Exploiting multicore technology to serve other disciplines (like medicine) in saving people's lives is a major goal we have from the beginning of this thesis work. However, there is a lack of structured methodology to follow for such multidisciplinary-design processes, and every case remains a solitary experience. Hence, we start by locating one large problem in healthcare (namely heart diseases) and its corresponding challenge in biomedical engineering (namely 12-lead ECG analysis), and then we take advantage of the field of MPSoC since it inspires us to produce HW/SW Codesigns that assist in finding a solution. The healthcare problem arises from the fact that medicine is still-unfortunately- dependent in cases like ECG on methods invented decades ago (when processing capabilities were not as advanced as today) and they suffer from the lack of: full analysis at high sampling frequencies, real-time reactivity, flexibility, low error, mobility, and ability to lower healthcare costs. Thus, the SW part of the solution faces a sub-problem, which is the need for innovative algorithms capable of running real-time full-ECG analysis and allowing for parallelization. Consequently, we created two new algorithms (ACF and FFT based) that have the required added-values and allow for easy manipulation in the HW/SW Codesign process. Many Codesign/Co-simulation iterations result in three new MPSoC architectures (shared bus, full and partial crossbar) that improve upon state of the art in SoC design for ECG analysis and add the ability to choose the design based on the biomedical need. Ultimately, to attend to our major goal, we use the experience we garner in the aforementioned ECG-problem-solving to found a methodology that we can employ in application-specific-MPSoC HW/SW Codesign. Such a methodology fosters the convergence to a solution that is acceptable from a performance analysis viewpoint, i.e. it meets the real-time demands, abides by the life-critical constraints, uses current technologies, and paves the way for easier and faster development. Section 7.1 articulates our concluding remarks. Section 7.2 is the summary.

7.1. Concluding Remarks

In this thesis we present a ***methodology*** **for HW/SW Codesign** for systems that require a large and challenging number of computations. This methodology summarizes our experience in a systematic procedure, technique, and mode of inquiry proper to a particular discipline; that of biomedical applications with high computational demand. The methodology also paves the way for an easier process of design space exploration and HW/SW Codesign for biomedical chips. In particular, we focus on a biochip for the human heart problems, and the application to map on this chip is electrocardiogram-analysis. We choose the biomedical application of ECG, because of its importance in the medical field and its market value, which we discuss in chapter 1. From the experience we have gathered over the line of research, we notice some general points that can be put together as a structure. This structure became the aforementioned methodology.

In this respect, we start our investigation by looking at three issues: the application (ECG), the available technology (MPSoC), and the proposed solution. Thus, we find ourselves facing a multidisciplinary problem with different layers as follows: life-critical, SW, HW, co-design, co-simulation, and implementation layer. We start by deciding on the requirements. The first requirement is related to the life-critical application. Hence, cycle accurate modeling and simulation seems as the best option. The second requirement is in relation to the high task level parallelism, where the 12 lead signals are parallel-processed. Therefore, a parallel HW platform is best suitable for our algorithms. The third requirement is related to the communication bandwidth, where a large amount of data on-chip and off-chip direct our attention to the communication limitation rather than computation limitation in MPSoC. Consequently, it is necessary to choose cycle and signal accuracy in the model. Last but not least is the requirement related to computation vs. communication parallelism, where storage and processing should be handled in parallel (for efficiency). Thus, DMA engines become attractive.

After defining the requirements, we then collect the different levels of the multidisciplinary problem into three main layers: (i) application layer (biomedical problem), (ii) software layer (algorithms and coding), and hardware layer (HW Design, co-simulation, testing, etc). These divisions are demonstrated in Figure 46 in Chapter 6. In brief, the starting point is always in finding the right application (problem) that has some level of significance, e.g. the life-critical application of real-time ECG analysis under high sampling frequencies (more than 1 KHz). Then we design the required algorithm, and we generate the relative code. Once the algorithm is done, a crucial step is to see whether the algorithm (application) can be partitioned, e.g. 12 different parts, each part is responsible for a different lead. In the SW layer, we define the parameters that we have to respect in the analysis. For instance, the real-time limits (in seconds) are a crucial constraint to respect, especially in relationship to the sampling frequency. The higher the frequency the more the data to analyze, and the more the time needed to analyze it. After these SW specs are defined, we look further into parallelizing the code itself and fixing code dependencies like the size of data-chunks. At this stage, we succeed in having a working code, but that does not necessarily mean this is the optimized code for the specific HW, and for the computation and communication requirements. The parallel code, its dependencies, and the application limitations help inspire the HW designer to choose and decide on the HW blocks, interconnections and architecture for accurate modeling. At this stage we have a non-optimized HW with interface mechanisms that are ready to load the SW and run cycle-and-signal accurate co-simulation. Hence, we can run much iteration before reaching the desired results. A note worth mentioning is that we aim at removing HW bottlenecks and SW bottlenecks. Moreover, we aim at design space exploration while looking for input sampling frequencies that range from 250Hz to 1 KHz and higher. Since we deal with a life critical application, we must go back to the SW and application layers to pedantically test for the higher-layer-specs and limitations. These tests may affect the optimization of both: SW and HW. Once the test results are acceptable, we consider the HW/SW Codesign optimized and can go for further physical development. The virtual platform we use for the co-simulations is the MPSIM environment [119] as a starting point for intensive extension and customization for the application. SystemC is used as a descriptive language for the simulation engine. The processor core model was at the level of their instruction set. Chip architecture was modeled with cycle accuracy, but the bus was modeled with signal accuracy since it is very critical in the communication. The simulation speeds achieved were up to 200 KCycles/sec on a P4@3.5GHz. Industry provided and technology-homogenous 0.13μm power models of all system components were used for power analysis.

In addition to the aforementioned methodology, and as a result of the HW/SW co-simulation, we present in this thesis ***three HW/SW Codesign architectures for ECG real-time analysis***. The HW designs are three *MPSoCs with*: (i) *single shared bus*, (ii) *full cross bar,* and (iii) *partial crossbar*. For all the three designs, we exploit industrial IP cores of a 200 MHz system: ST220 4-issue VLIW DSPs with 32 KB instruction and data caches, STBus interconnect from STMicroelectronics, in-house optimized memory controller with DMA capability, and a whole system modeled with the MPSIM virtual platform. Hence, we can retain a multi-DSP architecture processing a fixed-point-coding implementation of the ACF-based algorithm. We aim at finding the minimum number of DSPs for the application and the maximum achievable sampling frequency while meeting real-time requirements. We also aim at comparing with state of the art chips in the market and research. The typical state-of-the-art frequency range used in ECG analysis is 250Hz - 1 KHz. In this respect, for our first MPSoC architecture with shared bus, we are able to achieve a working system with acceptable performance for a maximum of 2.2 KHz input sampling frequency. Moreover, we reached a maximum of about 4 KHz with the crossbar architectures (Figure 43), i.e. with the full and the optimized partial crossbar. The number of cycles during which the bus transfers useful data over the bus busy cycles is 71.83%. The different architectures form a kind of different levels to choose from in order to suit the market solution. In other words, our work in that direction gives us the possibility to choose a different design for the different need (cost, performance, real-time limit, sampling frequency, etc). See also Table 3.

Besides the methodology and the three MPSoC architectures, we also present **two novel algorithms for ECG analysis**, namely the ACF-based algorithm and the FFT-based algorithm. We discuss these two algorithms in detail in Chapter 3. Both algorithms are based on the autocorrelation concept, but the FFT uses a faster method to detect the autocorrelation function coefficients. These algorithms are the immediate result of the need to have a SW (code) that can fully analyze the 12-lead ECG and at the same time allow for parallelization to run over an MPSoC. Hence, to solve the problems of already in-use (but unsatisfactory) algorithms, we had to compare our 2 novel algorithms with the most widely used algorithm in the market, namely the Pan Tompkins algorithm. The Pan Tompkins only detects the R peak and QRS complex of the ECG signal. Hence the Pan Tompkins doesn't allow full analysis of the ECG-lead. In addition, the Pan Tompkins was designed originally to be simple in order to fit the computing at the early 1980's (when it was developed). Hence, it was made to run on the Z80 (Zilog) or the NSC800 (National Semiconductor) microprocessor. However, the Pan Tompkins algorithm may fail to give correct results for the application layer since the human factor plays a role in the choice of the initial values wherefrom the algorithm starts.

On the other hand, in our algorithms, instead of looking for the R-peaks and then detecting the period, we thought of detecting the period first (via autocorrelation) and then looking for the peaks. For the calculation of the period, we use the autocorrelation function in (20).

By autocorrelation, we derive the period without looking for peaks. Then, we accurately find peaks in a time window equal to the period. However, this comes at a higher computational complexity (1.75 Million Multiplications in our implementation). For the heartbeat period, we need at least 3.5-4 seconds of ECG data chunk in order for the autocorrelation function to give accurate results. We tested the algorithms against the MIT-BIH database and the success was 100%. Both, the FFT and ACF based algorithms share the advantage of full signal analysis (all peaks) over the Pan Tompkins, i.e. discovery of more diseases. In addition, the ACF has the advantage of having clear autocorrelation peaks, i.e. eliminating confusing peaks and thus period-detection-errors. In other words, after comparing these two new algorithms with the most widely used algorithm (Pan Tompkins), we compare our two algorithms with each other. The algorithm that proved to best serve the biomedical and SW layers was the ACF-based ECG-analysis algorithm. Therefore, all the SW that was optimized for our HW was for the ACF-based algorithm, which consumes a huge number of computations.

We also present ***general interoperability protocols*** for inter-NoC communications. These interoperability protocols address the problem of the need of more than one multicore chip for some applications. In particular we discuss the NoC discovery algorithms and the External NoC Protocol for inter-NoC communications. These protocols allow for fitting the multicore chip in a larger network as discussed in chapter 3. Hence, this is a step forward on the roadmap to reach a global solution.

7.2. Summary

To sum up the above discussion, we focus on the five following points:

1. Methodology

In comparison to the available solutions, when our proposed methodology is used, it will result in a solution that uses a single chip with higher computational capabilities, low power consumption, smaller volume (compared to using 12 chips connected on the human body), lower cost, higher analysis speed, finer granularity of results, more flexibility in integrating higher order solutions (future 15 lead) by installing a new SW on a similar HW, mobility, and last but not least on the application (medical) layer the solution allows for more accurate readings by the doctor/nurse. Hence our solution allows for more accurate disease diagnosis in less time. A note worth mentioning is that in such applications, time is a crucial parameter. With our proposed methodology, the resulting solutions will enable the medical team to prescribe the right treatment at the right time. Moreover, the methodology we present aids in: application analysis, algorithm partitioning design, SW designs, HW architecture design, performance analysis, tradeoffs, and tuning.

2. Real-time Nomadic ECG Analysis Challenges

We look at the challenging biomedical application of the 12-lead ECG, with different frequencies in the range of KHz, and we create robust algorithms that can fit the SW parallelization, HW bottlenecks (computation and communication), and real-time diagnosis.

3. Autocorrelation-Based Algorithm as a Promising Alternative to Traditional Techniques

We also show how the ACF can serve as a better solution than the traditional technique, especially for full analysis. In order to achieve a full solution and system that can serve our main goal, an MPSoC is required to handle the increased computational requirements.

4. HW-SW Platform Exploration

In the design space exploration, we see how VLIW DSPs are more energy efficient than RISC cores, how the Bus-based interconnect limits rate to 2.2 KHz sampling frequency in the shared bus case, and how the VLIW core becomes the bottleneck at the 4 KHz sampling frequency.

All in all, we demonstrate a practical case, which has an impact on a medical application with a large potential market. We show how MPSoC supports this solution. This work proposes an application-specific design methodology for the ECG domain, which envisions full system modeling accuracy, high HW/SW parallelism exploitation, and computation vs. communication parallelism. In addition, this thesis illustrates the potential advantages that MPSoC technology can bring to real-time ECG analysis, which can be summarized in the following three points: (i) larger time margin to run diagnosis algorithms, (ii) energy efficiency, (iii) and improved scalability to challenging higher sampling frequencies and to more accurate ECG analysis algorithms. Finally, regarding practical case studies, the thesis shows the advantages of deploying more computation-demanding analysis algorithms for higher quality of ECG analysis.

Chapter 8

Open Issues & Future Work

"Progress lies not in enhancing what is, but in advancing toward what will be."

Francis Bacon (1561 – 1626 A.D.)

This chapter lists open issues in the presented research and thus suggests future work. We discuss the research points that have been left without investigation, and then look at the open issues to suggest specific future tasks of research.

In this thesis, some issues remain open regarding the cost of the final system and the area. Finding the ultimate exact thresholds require more testing on real patient data. Although we left this issue open, we have made this part easier by automating the analysis, validation algorithm, and implementation code.

Another open issue is the set of tests to be made for the use of other threshold techniques like using wavelet theory, which is briefly introduced in Appendix 3. These tests can show the effectiveness of the wavelet theory in choosing thresholds for peak detection.

From the open issues and more reflection, we suggest some future work for investigation in what follows:

- HW implementation of the Filter in two stages: analogue and digital
- Exploring DVFS & power management
- Designing of a module for the Co-simulation phase that aides the methodology in giving results about the chip area
- Developing a module that can aid in knowing the cost based on the business and industrial-economic models and work available, and this can be a combination of multidisciplinary group. It may be a good application for a larger project that helps the methodology gain more dimensions
- Fabricating and running the code on a real MPSoC
- Running accurate power consumption tests

- Designing and fabricating a t-shirt with the sensors, leads, and prototype chip to try the real system as this may show new real implementation problems to solve
- Testing with a wireless technology like WLAN together with 3G, DVB, and others to transmit the medical results from the biochip in a real environment to check for the final solution
- Applying a general method based on the referred patent [84][85] in order to design future multicore chips up to the required specifications in performance in the minimum time
- Using a common set of metrics for benchmarking the hardware design based on the patents [84][85]
- Investigating the future issue of 3D on-chip networks [86][89] since 3D microelectronics are becoming more of a reality.

REFERENCES

[1] A. Alexandridi and G. Manis, Hardware Design for the Computation of Heart rate Variability, Journal of Medical Engineering & Technology, Vol. 26, No. 2, pp. 49-62, March-April 2002.

[2] A. Banerjee, R. Francis, J. Lee, J. May, S.W. Moore, and R.D. Mullins, Towards a Communication-Centric Design Methodology, Design, Automation and Test in Europe (DATE'06), Special Workshop on Future Interconnects and Networks on Chip, March 2006.

[3] A. Bona, V. Zaccaria, and R. Zafalon, System Level Power Modeling and Simulation of High-End Industrial Network-on-Chip, In Proceedings of Design and Test in Europe Conference (DATE), Paris, France, pp. 318-323, February 2004.

[4] A. Hemani, A. Jantsch, S. Kumar, A. Postula, J. Öberg, M. Millberg, and D. Lindqvist, Network on chip: An Architecture for Billion Transistor Era, In Proceeding of the IEEE NorChip Conference, November 2000.

[5] A. Jantsch and H. Tenhunen, editors, Networks on Chip, Kluwer Academic Publishers, February 2003.

[6] A. Jantsch and H. Tenhunen, Will Networks on Chip Close the Productivity Gap?, Networks on Chip, Chapter 1, pp. 3-18. Kluwer Academic Publishers, February 2003.

[7] A. Jantsch, Models of Computation for Networks on Chip, in the Proceedings of the Sixth International Conference on Application of Concurrency to System Design, June 2006.

[8] A. Jantsch, NoCs: A New Contract between Hardware and Software, in the Proceedings of the Euromicro Symposium on Digital System Design, September 2003.

[9] A. Jantsch, R. Lauter, and A Vitkowski, Power Analysis of Link Level and End-to-End Data Protection on Networks on Chip, in Proceedings of the IEEE International Symposium on Circuits and Systems, 2005.

[10] A. Leroy, P. Marchal, F. Robert and F. Catthoor , Spatial Division Multiplexing: A Novel Approach for Guaranteed Throughput on NoCs , IMEC Belgium, VUB Belgium, KUL Belgium, 2005.

[11] A. Pullini, F. Angiolini, D. Bertozzi, and L. Benini, Fault Tolerance Overhead in Network-on-Chip Flow Control Schemes, in the Proceedings of 18th Annual Symposium on Integrated Circuits and System Design (SBCCI'05), Florianópolis, Brazil, pp. 224-229, September 4-7, 2005.

[12] A. Pullini, F. Angiolini, P. Meloni, D. Atienza, S. Murali, L. Raffo, G. De Micheli, and L. Benini, NoC Design and Implementation in 65nm Technology, Proceedings of the 1st ACM/IEEE International Symposium on Networks-on-Chip, Princeton (NJ), USA, pp. 273-282, May 7-9, 2007.

[13] A. Ruha, S. Sallinen, and S. Nissila, A Real-Time Microprocessor QRS Detector System with a 1-ms Timing Accuracy for the Measurement of Ambulatory HRV, in the IEEE Transactions on Biomedical Engineering, Vol. 44, No. 3, pp. 159-167, Marhc1997.

[14] A. Segal, EKG tutorial, EMT-P 1997, http://www.drsegal.com/medstud/ecg/

[15] Ambu Inc., Biomedical devices company, http://www.ambuusa.com/

[16] ARM DAI 0033A Note 33: Fixed Point Arithmetic on the ARM, September 1996.

[17] Association of Cardiac Technology in Victoria- ACTIV, http://www.activinc.org.au/

[18] B. Bouyssounouse and J. Sifakis, Embedded Systems Design: The Artist Roadmap for Research and Development, Springer, 2006.

[19] B. Deb, S. Bhatnagar, and B. Nath, ReInforM: Reliable Information Forwarding Using Multiple Paths in Sensor Networks, the 28th Annual IEEE Conference on Local Computer Networks (LCN), Octber 2003.

[20] B. Lo, S. Thiemjarus, R. King, and G. Yang, Body Sensor Network– A Wireless Sensor Platform for Pervasive Healthcare Monitoring, In Adjunct Proceedings of the 3rd International Conference on Pervasive Computing (PERVASIVE'05), Munich, Germany, pp. 77-80, May 2005.

[21] BIOPAC Systems Inc., http://biopac.com/

[22] C. Chan, J. Han, and D. Ramjeet, LabVIEW^{TM} Design of a Vectorcardiograph and 12-Lead ECG Monitor: Final Year Project for the Bachelor of Science Degree in the University of Manitoba, March 2003.

[23] C. Coelho Jr., A. O. Femandes, J. C. Conway, F. Coaea Jr., H. S. Carvalho, and J. M. Mata, A Biomedical Wearable Device for Remote Monitoring of Physiological

Signals, in Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation (ETFA'03), Vol. 2, pp. 708-713, 16-19 September, 2003.

[24] C. Harland, T. Clark, and R. Prance, Electric Potential Probes– New Directions in the remote sensing of the human body, Measurement Science and Technology, Vol. 13, pp. 163-169, 2002.

[25] C. Harland, T. Clark, and R. Prance, High Resolution Ambulatory Electrocardiographic Monitoring Using Wrist-Mounted Electric Potential Sensors, Measurement Science and Technology, Vol. 14, pp. 923-928, 2003.

[26] C. Nicopoulos, D. Park, J. Kim, N. Vijaykrishnan, M. Yousif, and C.R. Das, ViChaR: A Dynamic Virtual Channel Regulator for Network-on-Chip Routers**,** in Proceedings of the 39th Annual International Symposium on Microarchitecture (MICRO), pp. 333-344, 2006.

[27] C. Sauer, M. Gries, S. Dirk, J.C. Niemann, M. Porrmann, and U. Rückert, A Lightweight NoC for the NOVA Packet Processing Platform , Design, Automation and Test in Europe (DATE'06), Special Workshop on Future Interconnects and Networks on Chip, 2006.

[28] C. W. Mundt, K. N. Montgomery, U. E. Udoh, V. N. Barker, G. C. Thonier, A. M. Tellier, R. D. Ricks, R. B. Darling, Y. D. Cagle, N. A. Cabrol, S. J. Ruoss, J. L. Swain, J. W. Hines, and G. T. Kovacs, A Multiparameter Wearable Physiologic Monitoring System for Space and Terrestrial Applications, in the IEEE Transactions in Information Technology in Biomedicine, Vol. 9, No. 3, September 2005.

[29] C. Yen, W. Chung, M. Chi, and S. Lee, A 0.75-mW Analog Processor IC for Wireless Biosignal Monitor, ACM ZSLPED '03, August 25-27, Seoul, Korea, 2003.

[30] C. Yi Lee and C. Toumazou, Ultra-Low Power UWB for Real Time Biomedical Wireless Sensing, in the Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS) 2005, Vol. 1, pp. 57-60, May 23-26, 2005.

[31] Code Blue, Medical Wireless Sensor Networks,

http://www.eecs.harvard.edu/~mdw/proj/codeblue

[32] D. Andreasson and S. Kumar ,STAR : An Efficient Routing Strategy for NoC with Mixed QoS Requirements, Jönköping University, Sweden, 2006.

[33] D. Bertozzi, A. Jalabert, S. Murali, R. Tamhankar, S. Stergiou, L. Benini, and G. De Micheli, NoC Synthesis Flow for Customized Domain Specific Multiprocessor

Systems-on-Chip, IEEE Transactions on Parallel and Distributed Systems, Vol. 16, Issue 2, pp. 113 – 129, Feb. 2005.

[34] D. Jun and Z. Hong-Hai, Mobile ECG detector through GPRS/Internet, the 17th IEEE Symposium on Computer-Based Medical Systems (CBMS'04), Bethesda MD, USA, June 2004.

[35] D. Jun, X. Miao, Z. Hong-hai, and L. Wei-feng, Wearable ECG Recognition and Monitor, Computer-Based Medical Systems, in the Proceedings of the 18th IEEE Symposium on Computer-Based Medical Systems (CBMS'05), pp. 413–418, 23-24 June, 2005.

[36] D. Lattard, E. Beigne, C. Bernard, C. Bour, F. Clermidy, Y. Durand, J. Durupt, D. Varreau, P. Vivet, P. Penard, A. Bouttier, and F. Berens, A Telecom Baseband Circuit based on an Asynchronous Network-on-Chip, IEEE International Conference on Solid-State Circuits (ISSCC 2007), San Francisco, CA, USA, February 11-15, 2007.

[37] D. Pamunuwa, J. Öberg, L R Zheng, M Millberg, A Jantsch, and H Tenhunen, Layout, Performance and Power Trade-offs in Mesh-Based Network-on-Chip Architectures, in the IFIP International Conference on Very Large Scale Integration (VLSI-SOC), Darmstadt, Germany, December 2003.

[38] D. Park, C.A. Nicopoulos, J. Kim, N. Vijaykrishnan, and C.R. Das, A Distributed Multi-Point Network Interface for Low-Latency, Deadlock-Free On-Chip Interconnects, in Proceedings of the 1st International Conference on Nano-Networks (Nano-Net), pp. 1-6, 2006.

[39] D. Park, C.A. Nicopoulos, J. Kim, N. Vijaykrishnan and C.R. Das, Exploring Fault-Tolerant Network-on-Chip Architectures, in Proceedings of the International Conference on Dependable Systems and Networks (DSN'06), pp. 93-102, 2006.

[40] D. Wu, T. Y. Hou, W. Zhu, Z. He, and Y. Zhang, Adaptive QoS control for real-time video communication over wireless channels, International Journal of Ad Hoc and Ubiquitous Computing, Volume 1, Numbers 1-2, pp. 27-37, November 10, 2005.

[41] E. Company-Bosch and E. Hartmann, ECG Front-End Design is Simplified with MicroConverter, Journal of Analog Dialogue, Vol. 37, pp. 10-15, 2003.

[42] E. Nilsson and J. Öberg, PANACEA – A Case Study on the PANACEA NoC- a Nostrum Network on Chip Prototype, Technical Report, TRITA-ICT/ECS R 06:01, School of Information and Communication Technology, Royal Institute of Technology, Electrum 229, SE-164 40 Kista, Sweden, April 2006.

[43] E. Nilsson and J. Öberg, Reducing Peak Power and Latency in 2-D Mesh NoCs Using Globally Pseudochronous Locally Synchronous Clocking, in the Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis, September 2004.

[44] E. Nilsson and J. Öberg, Trading off Power Versus Latency Using GPLS Clocking in 2D-Mesh NoCs, in the Proceedings of the International Symposium on Signals, Circuits and Systems (ISSCS), July 2005.

[45] E. Nilsson, M. Millberg, J Öberg, and A Jantsch, Load Distribution with the Proximity Congestion Awareness in a Network on Chip, in the Proceedings of the Design Automation and Test Europe (DATE), pp. 1126-1127, March 2003.

[46] E. Pino, L. Ohno–Machado, E. Wiechmann, and D. Curtis, Real–Time ECG Algorithms for Ambulatory Patient Monitoring, in the Proceedings of the AMIA Annual Symposium 2005, pp. 604–608, 2005.

[47] ECG-Numerique Report, Cardionics/Louvaine ECG Algorithm, Two Published Comparisons With Industry-Leading Competitive Interpretive Algorithms, 2003. http://www.ecg-numerique.org/

[48] F. Angiolini, D. Atienza, S. Murali, L. Benini, and G. De Micheli, Reliability Support for On-Chip Memories Using Networks-on-Chip, Proceedings of the International Conference on Computer Design (ICCD) 2006, San José (CA), USA, Oct 1-4, 2006.

[49] F. Angiolini, J. Ceng, R. Leupers, F. Ferrari, C. Ferri, and L. Benini, An Integrated Open Framework for Heterogeneous MPSoC Design Space Exploration, Proceedings of the Design, Automation and Test in Europe Conference and Exhibition 2006, pp. 1145-1150, Munich, Germany, Mar 6-10, 2006.

[50] F. Angiolini, L. Benini, and A. Caprara, An Efficient Profile-Based Algorithm for Scratchpad Memory Partitioning, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 24 Issue 11, pp. 1660-1676, November 2005.

[51] F. Angiolini, L. Benini, and A. Caprara, Polynomial-Time Algorithm for On-Chip Scratchpad Memory Partitioning, Proceedings of the 2003 ACM International Conference on Compilers, Architecture, and Synthesis for Embedded Systems (CASES), San José (CA), USA, pp. 318-326, Oct 30-November 1, 2003.

[52] F. Angiolini, M. H. Ben Jamaa, D. Atienza, L. Benini, and G. De Micheli, Improving the Fault Tolerance of Nanometric PLA Designs, Proceedings of the Design, Automation and Test in Europe Conference and Exhibition 2007, Nice, France, pp. 570-575, April 16-20, 2007.

[53] F. Angiolini, P. Meloni, D. Bertozzi, L. Benini, S. Carta, and L. Raffo, Networks on Chips: A Synthesis Perspective, Parallel Computing: Current & Future Issues of High-End Computing, NIC Series, Vol. 33, pp. 745-752, 2006.

[54] F. Angiolini, P. Meloni, S. Carta, L. Benini, and L. Raffo, Contrasting a NoC and a Traditional Interconnect Fabric with Layout Awareness, Proceedings of the Design, Automation and Test in Europe Conference and Exhibition 2006 (DATE'06), Munich, Germany, pp. 124-129, Mar 6-10, 2006.

[55] F. Angiolini, P. Meloni, S. Carta, L. Raffo, and L. Benini, A Layout-Aware Analysis of Networks-on-Chip and Traditional Interconnects for MPSoCs, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 26, Issue 3, pp. 421-434, March 2007.

[56] F. Angiolini, S. Mahadevan, J. Madsen, L. Benini, and J. Sparsø, Realistically Rendering SoC Traffic Patterns with Interrupt Awareness, Proceedings of the IFIP VLSI-SOC Conference 2005, Perth (WA), Australia, pp. 211-216, October 17-19, 2005.

[57] F. Balarin, P. D. Giusto, A. Jurecska, C. Passerone, E. Sentovich, B. Tabbara, M. Chiodo, H. Hsieh, L. Lavagno, A. Sangiovanni-Vincentelli, and K. Suzuki,

Hardware-Software Co-Design of Embedded Systems- The POLIS approach, The Springer International Series in Engineering and Computer Science, Vol. 404, 1997.

[58] F. Menichelli, L. Benini, M. Olivieri, M. Donno, and L. Bisdounis, A simulation-based power-aware architecture exploration of a multiprocessor System-on-Chip design, IEEE/ACM Design Automation and Test in Europe, February 2004.

[59] F. Poletti, A. Poggiali, and P. Marchal, Flexible Hardware/Software Support for Message Passing on a Distributed Shared Memory Architecture, The Design and Test in Europe Conference (DATE'05), Nice, France, April 16-20, 2005, 736-741

[60] Freescale^{TM} semiconductor, Personal Electrocardiogram (ECG) Monitor, http://www.freescale.com/

[61] G. Liang, Design of Frequency Controller for Minimizing Power Consumption in Network-on-Chip, Master's thesis, School for Information and Communication Technology, Royal Institute of Technology, Stockholm, Sweden, October 2005.

[62] G. Moore, Cramming More Components onto Integrated Circuits, Electronics Magazine, Volume 38, Number 8, April 19, 1965.

[63] G. Moore, Innovation More Important than Ever in Platform Era, Intel Developer Forum, San Francisco March 1, 2005.

[64] G. Moore, Progress in digital Integrated Electronics, Technical Digest, IEEE International electron Devices Meeting, pp. 11-13, 1975.

[65] G. P. Li, J. Waldura, and P. Delvaux, The New Cardionics ECG Program and Its Comparison with Other Programs, Japanese Heart Journal; 35 (Supplement), pp. 257-258, 1994.

[66] H. Wang, L. S. Peh, and S. Malik, Power-driven Design of Router Microarchitectures in On-chip Networks, In Proceedings of MICRO 36, San Diego, November 2003.

[67] H. Wang, X. Zhu, L.S. Peh and S. Malik, Orion: A Power-Performance Simulator for Interconnection Networks, In Proceedings of MICRO 35, Istanbul, Turkey, November 2002.

[68] Heart and Stroke Foundation of Canada, The Changing Face of Heart Disease and Stroke in Canada 2000 Annual report.

[69] Heart Diseases and Stroke Statistics, 2008 Update Report, American Heart Association and American Stroke Association, Learn and Live, 2008. http://www.americanheart.org/presenter.jhtml?identifier=3018163

[70] **I. Al Khatib** and M. Ismail., WNOC- A Microelectronic System Architecture Suitable for Biomedical Wireless Sensor Network, in the Proceedings of the 48th IEEE International Midwest Symposium on Circuits and Systems (MWSCAS'05), Cincinnati, Ohio, USA, August 7-10, 2005.

[71] **I. Al Khatib**, Emerging Networks on Chip: Lessons Learned to Win the Future, The 2006 Design Automation and Test in Europe (DATE'06) Special Workshop on Future Interconnects and Networks on Chip, Munich, Germany, March 6-10, 2006.

[72] **I. Al Khatib**, A. Jantsch, and M. Saleh, Simulation of Real Home Healthcare Sensor Networks Utilizing IEEE802.11g Biomedical Network-on-Chip, in the Proceedings of the Workshop on Real-World Wireless Sensor Networks (REALWSN'05), Stockholm, Sweden, June 20-21, 2005.

[73] **I. Al Khatib**, A. Jantsch, and R. Nabiev, ECG-BIONET: A Global Biomedical Network for Human Heart Monitoring and Analysis, IEEE INFOCOM 2006 Student Workshop, Barcelona, Spain, April 24, 2006.

[74] **I. Al Khatib**, A. Jantsch, B. Kayal, R. Nabiev, and S. Jonsson, "Wireless Network-on-Chip as an Autonomous System: A Novel Solution for Biomedical Healthcare and Space Exploration Sensor-Networks," INFOCOM 05 Workshop, Miami, Florida, USA, March 2005.

[75] **I. Al Khatib**, A. Jantsch, R. Nabiev, and L. Onana-Alima, Interoperability Protocol and Algorithms for Network-on-Chip Autonomous-System Communications for the Next Generation Biomedical Sensor-Networks, The 2006 Design Automation and Test in Europe (DATE'06) Special Workshop on Future Interconnects and Networks on Chip, Munich, Germany, March 6-10, 2006.

[76] **I. Al Khatib**, D. Bertozzi, A. Jantsch, and L. Benini, Performance Analysis and Design Space Exploration for High End Biomedical Applications: Challenges and

Solutions, International Conference on Hardware-Software Codesign and System Synthesis (CODES-ISSS 2007), Salzburg, Austria, September 30-Oct 5, 2007.

[77] **I. Al Khatib**, D. Bertozzi, F. Poletti, L. Benini, A. Jantsch, M. Bechara, H. Khalifeh, M. Hajjar, R. Nabiev, and S. Jonsson, Hardware/Software Architecture for Real-Time ECG Monitoring and Analysis Leveraging MPSoC Technology, In the Lecture Notes on Computer Science (LNCS) Transactions on High-Performance Embedded Architectures and Compilers (HiPEAC), pp. 239-258, 2007.

[78] **I. Al Khatib**, D. Bertozzi, F. Poletti, L. Benini, A. Jantsch, M. Bechara, H. Khalifeh, M. Hajjar, R. Nabiev, and S. Jonsson, MPSoC ECG Biochip: A Multiprocessor System-on-Chip for Real-Time Human Heart Monitoring and Analysis, ACM SIGMICRO International Conference on Computing Frontiers, Ischia, Italy, May 2-5, 2006.

[79] **I. Al Khatib**, F. Curti, G. Russo, and R. Nabiev, Sat-BIONET: A Satellite Telemedicine Application and Platform Utilizing DVB-RCS for Medical Aid at the Disaster Site, 1st SatLabs DVB-RCS Symposium, European Space Agency (ESA), the European Space Research and Technology Centre (ESTEC), Noordwijk, Netherlands, September 8–9, 2005.

[80] **I. Al Khatib**, F. Poletti, D. Bertozzi, L. Benini, M. Bechara, H. Khalifeh, A. Jantsch, and R. Nabiev, A Multiprocessor System-on-Chip for Real-Time Biomedical Monitoring and Analysis: ECG Prototype Architectural Design Space Exploration, accepted at the ACM Transactions on Design Automation of Electronic Systems (TODAES), 2008.

[81] **I. Al Khatib**, F. Poletti, D. Bertozzi, L. Benini, M. Bechara, H. Khalifeh, A. Jantsch, and R. Nabiev, A Multiprocessor System-on-Chip for Real-Time Biomedical Monitoring and Analysis: Architectural Design Space Exploration, Design Automation Conference (DAC'06) 2006, San Francisco, California, USA July 24-28, 2006.

[82] **I. Al Khatib**, G. Russo, and R. Nabiev, Performance Analysis of Interoperability Protocols and Algorithms in Networks-on-Chip for the Next Generation Biomedical Sensor-Networks, IEEE INFOCOM 2006, Barcelona, Spain, April 23-29, 2006.

[83] **I. Al Khatib**, M. Saleh, and A. Jantsch, EEG BIONoC: Electroencephalogram Biomedical Network-on-Chip for Brain Analysis of Epileptic Patients, The 2006 Design Automation and Test in Europe Conference (DATE'06), PhD Forum presentation, Munich, Germany, March 6-10, 2006.

[84] **I. Al Khatib**, PATENT, Performance of Component, Patent Registration number PCT/SE2004/001397, intellectual property protection under the Patent Cooperation Treaty, 2008.

[85] **I. Al Khatib**, PATENT, Quantifying Quality of Communication Component, Patent Registration number: 0302609-3 Sweden, 2005.

[86] I. Loi, F. Angiolini, and L. Benini, Supporting vertical links for 3D networks on chip: toward an automated design and analysis flow, Proceedings of the Nano-Net Conference 2007, Catania, Italy, September 24-26, 2007.

[87] I. Walter, I. Cidon, R. Ginosar, and A. Kolodny, Access Regulation to Hot-Modules in Wormhole NoCs, the First International Symposium on Networks-on-Chip 2007 (NOCS'07), May 7-9 May, 2007.

[88] J. Kim, C.A. Nicopoulos, D. Park, N. Vijaykrishnan, M.S. Yousif, and C.R. Das, A Gracefully Degrading and Energy-Efficient Modular Router Architecture for On-Chip Networks, in Proceedings of the 33rd Annual International Symposium on Computer Architecture (ISCA), pp. 4-15, 2006.

[89] J. Kim, C.A. Nicopoulos, D. Park, R. Das, Y. Xie, N. Vijaykrishnan, and C.R. Das, A Novel Dimensionally-Decomposed Router for On-Chip Communication in 3D Architectures, in Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA), pp. 138-149, 2007.

[90] J. Kim, D. Park, C.A. Nicopoulos, N. Vijaykrishnan and C.R. Das, Design and Analysis of an NoC Architecture from Performance, Reliability and Energy Perspective, in Proceedings of the Symposium on Architecture for Networking and Communications Systems (ANCS), pp. 173-182, 2005.

[91] J. L. Willems, C. Abreu-Lima, P. Arnaud, J. H. van Bemmel, C. Brohet, R. Degani, B. Denis, J. Gehring, I. Graham, and G. van Herpen, The Diagnostic Performance of Computer Programs for the Interpretation of Electrocardiograms, New England, Journal of Medicine; 325, pp. 1767-1773, 1991.

[92] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, Digital Integrated Circuits: A Design perspective, Second edition, Pearson Education Inc., 2003

[93] J. Malmivuo and R. Plonsey, Bioelectromagnetism: Principles and Applications of Bioelectric and Biomagnetic Fields, Oxford University Press, 1995.

[94] J. Pan and W. Tompkins, A Real-Time QRS Detection Algorithm, IEEE Transactions on Biomedical Engineering, Vol. BME-32, No. 3, pp. 230-236, March 1985.

[95] J. Rodrigues, T. Olsson, L. Sornmo, and V. Owall, A Dual-Mode Wavelet Based R-Wave Detector using Single-V_t for Leakage Reduction, in the IEEE International Symposium on Circuits and Systems 2005 (ISCAS'05). Vol. 2, pp. 1330 – 1333, May 23-26, 2005.

[96] J. Rodrigues, V. Owall, and L. Sornmo, A Wavelet Based R-Wave Detector for Cardiac Pacemakers in 0.35 CMOS Technology, in the Proceedings of the 2004 International Symposium on Circuits and Systems 2004 (ISCAS'04), Vol. 4, May 23-26, pp. IV-13-16, 2004.

[97] J. Yao, R. Schmitz, and S. Warren, A Wearable Point-of-Care System for Home Use That Incorporates Plug-and-Play and Wireless Standards, IEEE Transaction on Information Technology in Biomedicine, Vol. 9, No. 3, pp. 363-371, September 2005.

[98] K. Goossens, Multiple Use-Cases for A Network-on-Chip Design Flow, In Future Interconnects and Network on Chip workshop at Design, Automation and Test in Europe Conference and Exhibition (DATE), March 2006.

[99] K. Hung, Y. T. Zhang, and B. Tai, Wearable Medical Devices for Tele-Home Healthcare, Engineering in Medicine and Biology Society, IEMBS 26th Annual International Conference of the IEEE EMBS, Issue 2 Vol. 7, San Francisco, CA, USA, pp. 5384-5387, September 2004.

[100] K-H. Nielsen, Evaluation of Real-Time Performance Models in Wormhole-Routed on-Chip Networks, Master thesis, Institute of Microelectronics and Information Technology, Royal Institute of Technology (KTH), Stockholm, Sweden, April 2005.

[101] L. Benini and D. Bertozzi, Network-on-Chip Architectures and Design Methods, in the IEE Proceedings on Computers and Digital Techniques, Vol. 152, Issue 2, pp. 261–272, March 2005.

[102] L. Benini and G. De Michelli, Networks on Chip: A New Paradigm for Systems on Chip Design, in the Proceedings of the 39th ACM IEEE Design Automation Conference (DAC'02), New Orleans, Louisiana, USA, pp. 418–419, 2002.

[103] L. Benini and G. De Micheli, Networks on Chips: A New SoC Paradigm, Computer, Vol. 35, Issue 1, pp. 70–78, January 2002.

[104] L. Benini, D. Bertozzi, A. Guerri, and M. Milano, Allocation and Scheduling for MPSoCs via decomposition and no-good generation, the 11th International Conference on Principles and Practice of Constraint Programming (CP2005), 2005.

[105] L. Benini, D. Bertozzi, A. Guerri, M. Milano, and F. Poletti, Measuring Efficiency and Executability of Allocation and Scheduling in Multi-Processor Systems-on-Chip, Atti della Giornata di Lavoro: Analisi sperimentale e benchmark di algoritmi per l'Intelligenza Artificiale (AIIA-RCRA05), 2005.

[106] L. Benini, D. Bertozzi, D. Bruni, N. Drago, F. Fummi, and M. Poncino, SystemC Cosimulation and Emulation of Multiprocessor SoC Designs, Computer, Vol. 36, Issue 4, pp. 53–59, April 2003.

[107] L. Benini, D. Bertozzi, D. Bruni, N. Drago, F. Fummi, and M. Poncino, Legacy SystemC Co-Simulation of Multi-Processor Systems-on-Chip, in the Proceedings of the 2002 IEEE International Conference on Computer Design: VLSI in Computers and Processors, 2002, pp. 494–499, 2002.

[108] L. Benini, D. Bertozzi, A. Bogliolo, F. Menichelli, and M. Olivieri, MPARM: Exploring the Multi-Processor SoC Design Space with SystemC, The Journal of VLSI Signal Processing, Volume 41, Number 2, pp. 169–182, 2005.

[109] L. Benini, L. Macchiarulo, A. Macii, and M. Poncino, Layout-Driven Memory Synthesis for Embedded Systems-on-Chip, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 10, Issue 2, pp. 96–105, April 2002.

[110] L. Benini, L. Macchiarulo, A. Macii, E. Macii, and M. Poncino, From architecture to layout: partitioned memory synthesis for embedded systems-on-chip, in the

Proceedings of IEEE/ACM Design Automation Conference (Las Vegas NV, USA, Jun.2001), IEEE Press, pp. 784-789, Piscataway NJ, USA, 2001.

[111] L. Bisdounis, C. Dre, S. Blionas, D. Metafas, A. Tatsaki, F. Ieromnimon, E. Macii, P. Rouzet, R. Zafalon, and L. Benini, Low-Power System-on-Chip Architecture for Wireless LANs, IEE Proceedings of Computers and Digital Techniques, 2004.

[112] L. Guang and A. Jantsch, Adaptive Power Management for the on-Chip Communication Network, the 9th Euromicro Conference on Digital System Design (DSD 2006), August 2006.

[113] L. Piccini, L. Arnone, F. Beverina, A. Cucchi, L. Petrelli, and G. Andreoni, Wireless DSP Architecture for Biosignals Recording, in the Proceedings of the Fourth IEEE International Symposium on Signal Processing and Information Technology 2004, pp. 487-490, December 18-21, 2004.

[114] L. Xia, Simulated Annealing Techniques for Mapping Cores onto 2-D Mesh Network on Chip, Master's thesis, ICT/ECS-2007-56, School for Information and Communication Technology, Royal Institute of Technology, Stockholm, Sweden, May 2007.

[115] M. Behar, A. Mendelson, and A. Kolodny, Trace Cache Sampling Filter, PACT, the ACM Transactions on Computer Systems (TOCS), Vol. 25, Issue 1, No. 3, February 2007.

[116] M. Chang, Z. Lin, C. Chang, H. Chan, and W. Feng, Design of a System-on-Chip for ECG Signal Processing, The 2004 IEEE Asia-Pacific Conference on Circuits and Systems, Tainan, Taiwan, December 2004.

[117] M. Fritzsche and P. Keil, Agile Methods and CMMI: Compatibility or Conflict, Journal of e-Informatic Sofwtare Engineering, Vol. 1, Issue 1, pp. 9-26, 2007.

[118] M. J. Irwin, L. Benini, N. Vijaykrishnan, and M. Kandemir, Techniques for designing energy-aware MPSoCs, in Multiprocessor Systems-on-Chip, Morgan Kaufman, 2004.

[119] M. Loghi, F. Angiolini, D. Bertozzi, L. Benini, and R. Zafalon, Analyzing On-Chip Communication in a {MPSoC} Environment, In Proceedings of Design and Test in Europe Conference (DATE0’04), Paris, France, pp. 752-757, February 2004.

[120] M. Loghi, M. Letis, M. Poncino, and L. Benini, Exploring the Energy Efficiency of Cache Coherence Protocols in Single-Chip Multi-Processors, in Proceedings of the ACM Great Lakes Symposium on VLSI (GLSVLSI), pp. 276 – 281, 2005.

[121] M. Loghi, M. Poncino, and L. Benini, Analyzing Power Consumption of Message Passing Primitives in a Single-chip Multiprocessor, International Conference on Computer Design (ICCD), pp. 393-396, 2004.

[122] M. Loghi, M. Poncino, and L. Benini, Cache Coherence Tradeoffs in Shared Memory MPSoCs, ACM Transactions on Embedded Computing Systems, Vol. 4, No. 3, 2005.

[123] M. Loghi, M. Poncino, and L. Benini, Cycle-Accurate Power Analysis for Multiprocessor Systems-on-a-Chip, GLSVLSI04: Great Lake Symposium on VLSI, Boston, MA, USA, pp. 401-406, April 2004.

[124] M. Meyer, Energy-Aware Task Allocation for Network-on-Chip Architectures, Master's thesis, School for Information and Communication Technology, Royal Institute of Technology, Stockholm, Sweden, March 2006.

[125] M. Millberg and A. Jantsch, Increasing NoC Performance and Utilisation Using a Dualpacket Exit Strategy, 10th Euromicro Conference on Digital System Design, Lubeck, Germany, August 2007.

[126] M. Millberg, E. Nilsson, R. Thid, and A. Jantsch, Guaranteed Bandwidth Using Looped Containers in Temporally Disjoint Networks within the Nostrum Network on Chip, in the Proceedings of the Design Automation and Test Europe Conference (DATE), February 2004.

[127] M. Millberg, E. Nilsson, R. Thid, S. Kumar, and A. Jantsch, The Nostrum Backbone- A Communication Protocol Stack for Networks on Chip, in the Proceedings of the VLSI Design Conference, Mumbai, India, January 2004.

[128] M. Ruggiero, A. Acquaviva, D. Bertozzi, and L. Benini, Application-Specific Power-Aware Workload Allocation for Voltage Scalable MPSoC Platforms, In Proceedings of the International Conference on Computer Design (ICCD 2005), San Jose, CA, USA, October 2005.

[129] M. Ruggiero, F. Angiolini, F. Poletti, D. Bertozzi, L. Benini, and R. Zafalon, Scalability Analysis of Evolving SoC Interconnect Protocols, in the Proceedings of the 2004 International Symposium on System-on-Chip, pp. 169-172, 2004.

[130] M. Saleh and **I. Al Khatib**, Performance of Secure Ad hoc Sensor Networks Utilizing IEEE802.11b WEP, in the Advanced Industrial Conference on Wireless Technologies (ICW 2005), Montreal, Canada, August 14-18, 2005.

[131] M. Saleh and **I. Al Khatib**, Throughput Analysis of WEP Security in Ad Hoc Sensor Networks, The Second International Conference on Innovations in Information Technology (IIT'05), Emirates Towers Hotel, Dubai, UAE, September 26-28, 2005.

[132] M. Zhong, Evaluation of Deflection-Routed on-Chip Networks, Master's thesis, School for Information and Communication Technology, Royal Institute of Technology, Stockholm, Sweden, August 2005.

[133] MIT-BIH Arrhythmia Database, Tape Directory and Format Specification, Document BMEC TR00, Mass. Inst. Tech., Cambridge, 1980.

[134] N. Chevrollier and N. Golmie, On the Use of Wireless Network Technologies in Healthcare Environments, Proceedings of the fifth IEEE workshop on Applications and Services in Wireless Networks, ASWN2005, pp. 147-152, June 2005.

[135] N. Eisley and L.S. Peh, High-Level Power Analysis for on-Chip Networks, In Proceedings of the 7th International Conference on Compilers, Architectures and Synthesis for Embedded Systems (CASES), Washington D.C., September 2004.

[136] N. Genko, D. Atienza, G. De Micheli, L. Benini, J. M. Mendias, R. Hermida, and F. Catthoor, A Novel Approach for Network on Chip Emulation, in the Proceedings of the IEEE International Symposium on Circuits and Systems, 2005 (ISCAS 2005), Vol. 3, pp. 2365 – 2368, May 23-26, 2005.

[137] N. El-Guindi and P. Elsener, Network on chip: PANACEA - a Nostrum Integration, Technical report, Swiss Federal Institute of Technology Zurich, February 2005.

[138] P. Bhojwani and R. Mahapatra, Interfacing Cores with on-Chip Packet-Switched Networks, in the Proceedings of the 16th International Conference on VLSI Design 2003, pp. 382 – 387, January 4-8, 2003.

[139] P. Bobbie, C. Arif, H. Chaudhari, and S. Pujari, Electrocardiogram (EKG) Data Acquisition and Wireless Transmission, WSEAS Transactions on Computers, vol. 3, issue 8, pp. 2665–2672, October 2004.

[140] P. Coggan, History Lessons for the Markets, in the Sepcial Issue on The World in 2008, The Economist, pp. 139-140, December 2007.

[141] P. Meloni, S. Carta, R. Argiolas, L. Raffo, and F. Angiolini, Area and Power Modeling Methodologies for Networks-on-Chip, Proceedings of the Nano-Net Conference 2006, Lausanne, Switzerland, September 14-16, 2006.

[142] PhysioBank, Physiologic signal archives for biomedical research,

http://www.physionet.org/physiobank/database/ptbdb/

[143] Physionet, The Research Resource for Complex Physiologic Signals, http://www.physionet.org/

[144] R. Dobkin, V. Vishnyakov, E. Friedman, and R. Ginosar, An asynchronous router for multiple service levels networks on chip, in the Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC'05), pp. 44 – 53, 2005

[145] R. Dobkin, R. Ginosar, and C. Sotiriou, Data Synchronization Issues in GALS SoCs, in the Proceedings of ASYNC, pp. 170-179, 2004.

[146] R. Dobkin, V. Vishnyakov, E. Friedman, and R. Ginosar, An Asynchronous Router for Multiple Service Levels Networks on Chip, in the Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC'05), pp. 44-53, 2005.

[147] R. Dobkin, Y. Perelman, T. Liran, R. Ginosar, and A. Kolodny, High Rate Wave-Pipelined Asynchronous on-Chip Bit-Serial Data Link, in the Proceedings of the 13th IEEE International Symposium on Asynchronous Circuits and Systems, (ASYNC'07), pp. 3–14, March 2007.

[148] R. Gindin, I. Cidon, and I. Keidar, NoC-Based FPGA: Architecture and Routing, the First International Symposium on Networks-on-Chip 2007 (NOCS'07), May 7-9 May, 2007.

[149] R. Holsmark and S. Kumar, On Options for Accessing Region in NoC, Design, Automation and Test in Europe (DATE'06), Special Workshop on Future Interconnects and Network on Chip, Munich, Germany, March 10, 2006.

[150] R. Pop and S. Kumar, On Performance Improvement of Concurrent Applications Using Simultaneous Multithreaded Processors as NoC Resources, Norchip Conference, 2006. 24th, Linkoping, Sweden, pp. 191-196, November 2006.

[151] R. Thid, A Network on Chip Simulator, Master's thesis, Department of Microelectronics and Information Technology, Royal Institute of Technology, IMIT/LECS 2002-17, August 2002.

[152] R. Thid, I. Sander, and A. Jantsch, Flexible Bus and NoC Performance Analysis with Configurable Synthetic Workloads, the 9th Euromicro Conference on Digital System Design (DSD 2006), August 2006.

[153] R. Thid, M. Millberg, and A Jantsch, Evaluating NoC Communication Backbones with Simulation, in Proceedings of the IEEE NorChip Conference, November 2003.

[154] S. Bertozzi, A. Acquaviva, A. Poggiali, and D. Bertozzi, Supporting Task Migration in MPSoCs: A Feasibility Study, Design, Automation and Test in Europe, 2006.

[155] S. Kumar, A. Jantsch, J-P. Soininen, M. Forsell, M. Millberg, J. Öberg, K. Tiensyrjä, and A. Hemani, A Network on Chip Architecture and Design Methodology, in the Proceedings of IEEE Computer Society Annual Symposium on VLSI, April 2002.

[156] S. Led, J. Fernández, and L. Serrano, Design of a Wearable Device for ECG Continuous Monitoring Using Wireless Technology, Proceedings of the 26th Annual International Conference of the IEEE EMBS, San Francisco, CA, USA, September 1-5, 2004.

[157] S. Mahadevan, F. Angiolini, M. Storoaard, R. G. Olsen, J. Sparso, and J. Madsen, Network Traffic Generator Model for Fast Network-on-Chip Simulation, in the Proceedings of the Design, Automation and Test in Europe, Vol. 2, pp. 780–785, March 7-11, 2005.

[158] S. Medardoni, M. Ruggiero, D. Bertozzi, L. Benini, G. Strano, and C. Pistritto, Capturing the Interaction of the Communication, Memory and I/O Subsystems in

Memory-Centric Industrial MPSoC Platforms, The Design, Automation & Test in Europe Conference & Exhibition, 2007 (DATE '07), Nice, France, April 16-20, 2007.

[159] S. Mohammad, Z. Lim, Y. Lin, A. Sani, K. Zanjani, M. Paracha, and J. Jenkins, Real-Time ECG Analysis Using a TI TMSC54x Digital Signal Processing Chip, Journal on Computers in Cardiology, pp. 541-544, 2003.

[160] S. Murali, L. Benini and G. De Micheli, Mapping and Physical Planning of Networks-on-Chip Architectures with Quality-of Service Guarantees, in the Proceedings of the 2005 Conference on Asia South Pacific on Design Automation, Shanghai, China, pp. 27-32, 2005.

[161] S. Murali, P. Meloni, F. Angiolini, D. Atienza, S. Carta, L. Benini, G. De Micheli, and L. Raffo, Designing Message-Dependent Deadlock Free Networks on Chips for Application-Specific Systems on Chips, Proceedings of the IFIP VLSI-SOC Conference 2006, Nice, France, pp. 158-163, October 16-18, 2006.

[162] S. Murali, P. Meloni, F. Angiolini, D. Atienza, S. Carta, L. Benini, G. De Micheli, and L. Raffo, Designing Application-Specific Networks on Chips with Floorplan Information, Proceedings of the International Conference on Computer Aided Design (ICCAD) 2006, San José (CA), USA, pp. 355-362, November 5-9, 2006.

[163] S. Murali, R. Tamhankar, F. Angiolini, A. Pullini, D. Atienza, L. Benini, and G. De Micheli, Comparison of a Timing-Error Tolerant Scheme with a Traditional Re-Transmission Mechanism for Networks on Chips, Proceedings of the 2006 International Symposium on System-on-Chip, Tampere, Finland, pp. 27-30, November 13-16, 2006

[164] S. Murali, T. Theocharides, N. Vijaykrishnan, M.J. Irwin, L. Benini, and G. De Micheli, Analysis of Error Recovery Schemes for Networks on Chips, IEEE Design & Test of Computers, Vol. 22, Issue 5, pp. 434–442, September-October 2005.

[165] S. Penolazzi and A. Jantsch, A High Level Power Model for the Nostrum NoC, the 9th Euromicro Conference on Digital System Design (DSD 2006), August 2006.

[166] S. Srinivasan, F. Angiolini, M. Ruggiero, N. Vijaykrishnan, and L. Benini, Simultaneous Memory and Bus Partitioning for SoC Architectures, Proceedings of

the IEEE International SOC Conference (SOCC) 2005, Washington (DC), USA, pp. 125-128, September 25-28, 2005.

[167] S. Stergiou, F. Angiolini, S. Carta, L. Raffo, D. Bertozzi, and G. De Micheli, Xpipes Lite: A Synthesis Oriented Design Flow For Networks on Chips, Proceedings of the Design, Automation and Test in Europe Conference and Exhibition 2005, Munich, Germany, pp. 1188-1193, March 7-11, 2005.

[168] S. Thurner, M. C. Feursten, S. B. Lowen, and M. C. Teich, Receiver-Operating-Characteristic Analysis Reveals Superiority of Scale-Dependent Wavelet and Spectral Measures for Assessing Cardiac Dysfunction, in the Physical Review Letters, 81, pp. 5688-5691, 1998.

[169] S. Toral, J. Quero, M. Perez, and L. Franquelo, A Microprocessor based System for ECG Telemedicine and Telecare, The 2001 IEEE International Symposium on Circuits and Systems, ISCAS 2001, Vol. 4, pp. 526–529, issue of 6-9 May 2001.

[170] T. Bengtsson, A. Jutman, S. Kumar, and R. Ubar, Delay Testing of Asynchronous NoC Interconnects, 12th International Conference on Mixed Design of Integrated Circuits and Systems, Kraków, June 22-25, 2005.

[171] T. Bjerregaard and J. Sparsø, A Router Architecture for Connection-Oriented Service Guarantees in the MANGO Clockless Network-on-Chip ,in the Proceedings of Design, Automation and Test in Europe (DATE'06), Vol. 2, pp. 1226- 1231, March 7-11, 2005.

[172] T. Desel, T. Reichel, S. Rudischhauser, and H. Hauer, A CMOS Nine Channel ECG Measurement IC, in the Proceedings of the 2nd International Conference on ASIC 1996, pp. 115-118, Shanghai, China, Oct. 21-24, 1996.

[173] T. Fulford-Jones, G. Wei, and M. Welsh, A Portable, Low-Power, Wireless Two-Lead EKG System, in the Proceedings of the 26th Annual International Conference of the IEEE EMBS, San Francisco, CA, USA, September 1-5, 2004.

[174] T. Li, Estimation of Power Consumption in Wormhole Routed Networks on Chip, Master's thesis, Institute of Microelectronics and Information Technology, Royal Institute of Technology (KTH), April 2005.

[175] T. Morad, U. Weiser, A. Kolodny, M. Valero, and E. Ayguadé, Performance, Power Efficiency and Scalability of Asymmetric Cluster Chip Multiprocessors, Computer Architecture Letters, Vol. 4, July 2005.

[176] T. Richardson, C. Nicopoulos, D. Park, N. Vijaykrishnan, Y. Xie, C. Das and V. Degalahal, A Hybrid SoC Interconnect with Dynamic TDMA-Based Transaction-Less Buses and On-Chip Networks, in Proceedings of the 19th International Conference on VLSI Design, pp. 657-664, 2006.

[177] T. Tao Ye, G. De Micheli, and L. Benini, Analysis of Power Consumption on Switch Fabrics in Network Routers, in the Proceedings of the 39th ACM IEEE Design Automation Conference (DAC'02), New Orleans, Louisiana, USA, pp. 524-529, 2002.

[178] T. Tao Ye, L. Benini, and G. De Micheli, Packetization and Routing Analysis of On-Chip Multiprocessor Networks, Journal of Systems Architecture: The EUROMICRO Journal, Vol. 50, Issue 2-3, Special Issue: Networks on chip, pp. 81-104, February 2004.

[179] T. Ye, L. Benini, and G. De Micheli, Packetized On-Chip Interconnect Communication Analysis, IEEE CS Proceedings of Design, Automation and Test in Europe (DATE) Conference, Munich (Germany), pp. 344-349, March 4-7, 2003.

[180] U. Ogras and R. Marculescu, 'It's a Small World After All': NoC Performance Optimization via Long-range Link Insertion, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 14, Issue 7, pp. 693-706, July 2006.

[181] V. Fuster, Epidemic of Cardiovascular Disease and Stroke: The Three Main Challenges, Circulation, Vol. 99, Issue 9, 1132-1137, 1999.

[182] V. Oklobdzija and R. Krishnamurthy, High-Performance Energy-Efficient Microprocessor Design, Series on Integrated Circuits and Systems, springer, 1st Edition, August 9, 2006.

[183] V. Soteriou, H. Wang, and L. Peh, A Statistical Traffic Model for On-Chip Interconnection Networks, In Proceedings n Proceedings of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Monterey, CA, September 2006.

[184] V. Soteriou, N. Eisley, H. Wang, B. Li, and L.S. Peh, Polaris: A System-Level Roadmap for On-Chip Interconnection Networks , In Proceedings of the 24th International Conference on Computer Design (ICCD), San Jose, CA, October 2006.

[185] W. Englese and C. Zeelenberg, A Single Scan Algorithm for QRS Detection and Feature Extraction, IEEE Computers in Cardiology, pp. 37-42, 1979.

[186] W. Heirman, J. Dambre, I. O'Connor, and J. Van-Campenhout, Reconfigurable Optical Networks for On-Chip Multiprocessors, Design, Automation and Test in Europe (DATE'06) Special Workshop on Future Interconnects and Networks on Chip, Munich, Germany, March 10, 2006.

[187] World Health Organization (WHO), Cardiovascular Disease Report, 2005, http://www.who.int/cardiovascular_diseases/en/

[188] World Health Organization (WHO), The Atlas of Heart Disease and Stroke, Part Five: The Future and the Past,
http://www.who.int/cardiovascular_diseases/resources/atlas/en/

[189] World Health Organization (WHO), Report on: The Future of CVD, from The Atlas of Heart Disease and Stroke, Part Five: The Future and the Past:
http://www.who.int/cardiovascular_diseases/en/cvd_atlas_25_future.pdf

[190] X. Chen and L. Peh, Leakage Power Modeling and Optimization in Interconnection Networks., In Proceedings of the International Symposium on Low Power and Electronics Design (ISLPED), Seoul, Korea, August 2003.

[191] X. Qian, Implementation of a JPEG Encoder on the Nostrum Network-on-Chip, Master thesis, School for Information and Communication Technology, Royal Institute of Technology, Stockholm, Sweden, July 2005.

[192] X. Yang, Mapping Applications onto Networks-on-Chip with a Genetic Algorithm, Master's thesis, School for Information and Communication Technology, Royal Institute of Technology, Stockholm, Sweden, June 2007.

[193] X. Xun, Schematic Design of an FPGA-Based Network on board, Master thesis, School for Information and Communication Technology, Royal Institute of Technology, Stockholm, Sweden, July 2007.

[194] Z. Guz, I. Keidar, A. Kolodny, and U. Weiser, Nahalal: Cache Organization for Chip Multiprocessors, IEEE Computer Architecture Letters, Vol. 6, No. 1, May 2007.

[195] Z. Lu and A. Jantsch, Flit Admission in on-Chip Wormhole-Switched Networks with Virtual Channels, in the Proceedings of the International Symposium on System-on-Chip 2003, November 2004.

[196] Z. Lu and A. Jantsch, Flit Ejection in on-Chip Wormhole-Switched Networks with Virtual Channels, in the Proceedings of the IEEE NorChip Conference, November 2004.

[197] Z. Lu and A. Jantsch, Network-on-Chip Assembler Language, Technical Report TRITA-IMIT-LECS R 03:02, Version 1.0, Institute of Microelectronics and Information Technology, Royal Institute of Technology (KTH), Stockholm, Sweden, June 2003.

[198] Z. Lu and A. Jantsch, Traffic Configuration for Evaluating Networks on Chips, in the Proceedings of the 5th International Workshop on Systems on Chip (IWSOC), July 2005.

[199] Z. Lu, A User Introduction to NNSE: Nostrum Network-on-Chip Simulation Environment, Technical Report, Royal Institute of Technology, Stockholm, November 2005.

[200] Z. Lu, B. Yin, and A. Jantsch, Connection-Oriented Multicasting in Wormhole-Switched Networks on Chip, in the Proceedings of the IEEE Computer Society Annual Symposium on VLSI, March 2006.

[201] Z. Lu, I. Sander, and A. Jantsch, Refining Synchronous Communication onto Network-on-Chip Best-Effort Services, Advances in Design and Specification Languages for SoCs - Selected Contributions from FDL 2005, Springer Verlag, 2006.

[202] Z. Lu, I. Sander, and A. Jantsch, Towards Performance-Oriented Pattern-Based Refinement of Synchronous Models onto NoC Communication, 9th Euromicro Conference on Digital System Design (DSD 2006), August 2006.

[203] Z. Lu, L. Tong, B. Yin, and A. Jantsch, A Power Efficient Flit-Admission Scheme for Wormhole-Switched Networks on Chip, In Proceedings of the 9th World Multi-Conference on Systemics, Cybernetics and Informatics, July 2005.

[204] Z. Lu, M. Liu, and A Jantsch, Layered Switching for Networks on Chip, in the Proceedings of The 44th ACM/IEEE Design Automation Conference (DAC'07), San Diego, CA, USA, pp. 122-127, June 4-8, 2007.

[205] Z. Lu, M. Zhong, and A. Jantsch, Evaluation of on-Chip Networks Using Deflection Routing, in the Proceedings of the 16th ACM Great Lakes Symposium on VLSI (GLSVLSI), Philadelphia, PA, USA, pp. 296-301, 2006.

Appendices

"The value of an idea lies in the using of it."

Thomas A. Edison (1847 – 1931 A.D.)

Appendix 1

Autocorrelation Function and Its Properties

To understand autocorrelation, one can think of convolution. Convolution is an integral that expresses the amount of overlap of one function *g* as it is shifted over another function *f*. It therefore "blends" one function with another. The convolution is sometimes also known by folding.

The convolution of *f* and *g* is written as: *f*g*. It is defined as the integration of the product of two functions after one of them is reversed and shifted as shown in (A1-1).

$$f * g(t) = \int_a^b f(\tau) g(t-\tau) d\tau \qquad \text{(A1-1)}$$

The integration range depends on the domain on which the functions are defined. The values, *a* and *b*, can tend to infinity. In case of a finite range, *f* and *g* are often considered to extend periodically in both directions, so that the term $g(t\text{-}\tau)$ does not imply a range violation. This use of periodic domains is sometimes called a cyclic, circular or periodic convolution. Extensions with zeros are also possible. Using zero-extended or infinite domains is sometimes called linear convolution, especially in the discrete case below, which is of interest to this thesis.

Discrete convolution

For discrete functions, one uses the discrete version of the convolution, as shown in (A1-2).

$$f * g(m) = \sum_n f(n) g(m-n) \tag{A1-2}$$

When multiplying two polynomials, the coefficients of the product are given by the convolution of the original coefficient sequences.

By generalizing the above cases, the convolution can be defined for any two integral functions defined on a locally compact topological group. Evaluating discrete convolutions using the above formula applied directly takes O(N^2) arithmetic operations for N points, but this can be reduced to O[N.log(N)] using a variety of fast algorithms.

Equation (A1-3) shows the Autocorrelation function is a convolution of two functions.

$$R_y[k] = \sum_{n=-\infty}^{n=+\infty} y[n] y[n-k] \tag{A1-3}$$

Properties of Autocorrelation function

1. A fundamental property of the autocorrelation is symmetry, $R(i) = R(-i)$, which is easy to prove from the definition. In the continuous case, the autocorrelation is an even function as shown in (A1-4) and (A1-5)

$$R_f(-\tau) = R_f(\tau) \tag{A1-4}$$

when f is a real function, and an Hermitian function

$$R_f(-\tau) = R_f^*(\tau) \tag{A1-5}$$

when *f* is a complex function.

2. The continuous autocorrelation function reaches its peak at the origin, where it takes a real value, i.e. for any delay τ, as shown in (A1-6).

$$\left|R_f(\tau)\right| \leq R_f(0) \tag{A1-6}$$

   This is a consequence of the Cauchy-Schwarz inequality. The same result holds in the discrete case.

3. The autocorrelation of a periodic function is, itself, periodic with the very same period.

4. The autocorrelation of the sum of two completely uncorrelated functions (the cross-correlation is zero for all τ) is the sum of the autocorrelations of eachfunction separately.

5. Since autocorrelation is a specific type of cross-correlation, it maintains all the properties of cross-correlation.

6. The autocorrelation of a white noise signal will have a strong peak at τ = 0 and will be close to zero for all other τ values. This shows that a sampled instance of a white noise signal is not statistically correlated to a sample instance of the same white noise signal at another time.

7. The Wiener-Khinchin theorem relates the autocorrelation function to the power spectral density via the Fourier transform as shown in (A1-7) and (A1-8):

$$R(\tau) = \int_{-\infty}^{+\infty} S(f)e^{j2\Pi f\tau}df \tag{A1-7}$$

$$S(f) = \int_{-\infty}^{+\infty} R(\tau)e^{-j2\Pi f\tau}d\tau \tag{A1-8}$$

Appendix 2

Detailed Description of the FFT-Based Algorithm

The FFT-based algorithm still uses the autocorrelation concept after it proved being efficient in terms of percentage of error, but we used a different approach to calculate it based on a mathematical property.

This different approach relies on the calculation of the Fast Fourier Transform of the signal (FFT), or the FFT of the derivative of the signal. The ACF is expressed by (A2-1).

$$R_x[k] = \sum_{n=-\infty}^{+\infty} x[n]x[n+k] \tag{A2-1}$$

and this expression can be seen as a convolution as shown in (A2-2).

$$R_x[k] = x[k] * x[n-k] \tag{A2-2}$$

Taking the above equation into the Fourier domain we (A2-3).

$$DFT\{R_x[k]\} = DFT\{x[k] * x[N-k]\} = X[k]X[N-k] = X[k]^2 \tag{A2-3}$$

Hence to get the autocorrelation function we take the FFT of the signal *X*[*k*] (we use the FFT to calculate the DFT), then square it and take the inverse FFT (IFFT) to get the ACF function. This method will yield a big reduction in the running time and complexity. The logic of the code proceeds as follows:

First we note that we will be using the radix-2 FFT, and hence we need to zero-pad the input data samples to the nearest power of 2 to be able to use the butterfly model for the FFT.

Going to the details of the code, we perform first the bit reversal function responsible to give us the right mapping from the original indices to the FFT indices and we store the reordered result in a new array. Then we manipulate in a ‘For’ loop the operations of the butterfly, i.e. multiplications and additions along with the necessary twiddle factors to get the FFT data, where we will need 2 arrays to store the real and imaginary parts of the resulting FFT.

After getting the complex FFT value we need to take the square of it and then perform the IFFT to get the ACF data. The square of a complex number of the form $a+b\underline{i}$ is a^2+b^2, so we square the data in both arrays containing the FFT real and imaginary parts. Then we use the inverse path in the butterfly, adjusting the operations order and the multiplication by the right twiddle factors to get the IFFT. We perform bit-reversal on this array to get the right indexing for the resulting IFFT which will be the ACF data in the same time.

One feature of this algorithm is that calculations are performed on place, i.e. we use the same array for the input as well as the output, and this is a property of the butterfly map that will have also a big influence on the use of memory for the algorithm (reduced memory usage).

Looking at the results we get, we can see that the real part is equal to the ACF data if we calculate using its direct formula and that the imaginary part is 0 for all the entries.

After that, we take the array containing the real numbers and perform detection on it as mentioned in the description of the ACF-based algorithm.

The code also contains logic to determine all the R, P, and T peaks and this is done by going over the original signal and using appropriate thresholds to detect candidate peaks.

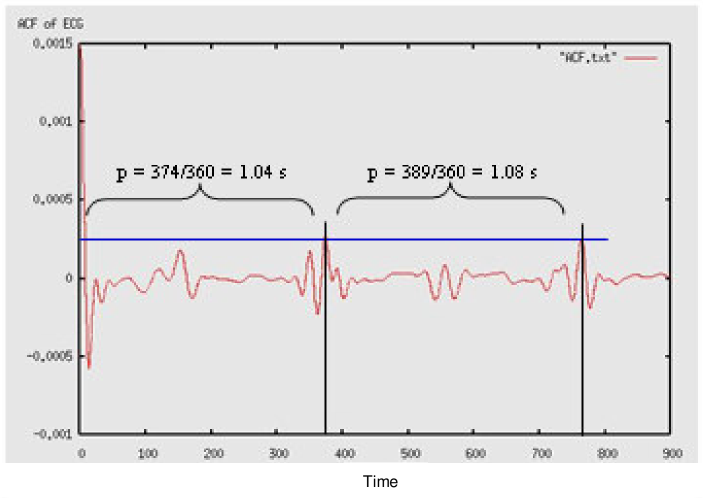
After implementing the algorithm in C, we plot the autocorrelation results as shown in Figure 27. To verify the validity of our results we plotted and compared the autocorrelation data from both: the direct calculation method and the FFT method for different patients with different diseases from the MIT-BIH database. A sample from the comparisons results is shown in Figure 47 below.

In this particular case, we can see from the 2 plots in Figure 47 that we have 2 peaks we can detect using an adequate thresholds, and in the case of FFT, the thresholds are more difficult due to the proximity of a first noise peak to the threshold. For each plot we can see that we have approximately equal periods between the detected peaks (1.04 ≈ 1.08 for both) and we have the same periods for the same intervals for the plots (1.04 for the first interval and 1.08 for the second).

Performing this same operation on different cases, we deduced that the peaks resulting from the FFT method yielded same results (same period) as with the direct method to calculate the ACF, but the second peaks for the FFT method were lower in amplitude relative to the maximum at the origin than those for the ACF-based method. Therefore we will require more accurate and harder thresholds for the FFT case.

After verifying the correctness of our results, we went to a complexity analysis study to evaluate the performance of this new algorithm and its advantages over the old algorithm. We studied the effect of the input data on the number of multiplications and additions as well as the effect of the augmentation of the input data on the number of these arithmetic operations, and the results were very satisfactory in terms of computation complexity.

We show the comparison between the direct ACF-based method and the FFT-based method in terms of the number of multiplications in Figure 30.



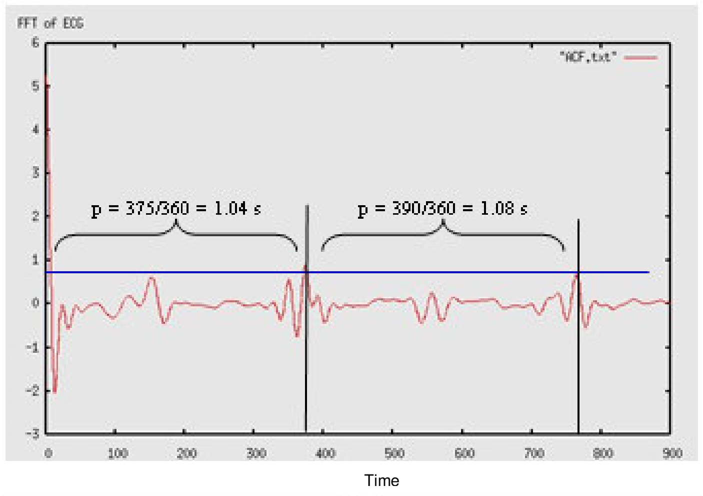


Figure 47. Comparison between the ACF-based algorithm and the FFT-based algorithm.

Appendix 3

Wavelet Overview

The wavelets are mathematical functions that are used to cut up a certain signal or function into different frequency components, and then study each component with the required resolution. There are some other mathematical methods that are widely used to this functionality, one of which is the famous Fourier Transform. However, the wavelet analysis has proven its advantages over the Fourier Transform, when it comes to analyzing signals that contain discontinuities and sharp spikes, and also in situations, where multi-resolution signal analysis is required both: in time and frequency domains.

The fundamental ideas behind wavelet analysis are to analyze the signal according to scale, and use a new set of basis functions called the analyzing wavelets, as will be discussed below.

Comparison with Fourier Analysis

The Fourier Transform is a famous and widely used technique used to analyze the frequency content of a whole signal. In other words, it breaks down the signal into a set of sinusoids of different frequencies and of different amplitudes.

The Fourier Transform comes in handy when the analysis of the frequency content of the whole signal is required; however, this method has a major drawback: when transforming into frequency domain, all time information about the signal is lost. In other words, one can not tell when a particular event (of a specific frequency) happened at a specified time.

To mitigate this deficiency, the Short-Time Fourier Transform (STFT) was developed. The main idea is that several windows that span the whole signal are applied, and then Fourier transform is applied to each of these windowed signal slices. The end result is a two-dimensional function of time and frequency. So, the STFT provides a compromise between time-view and frequency-view of the signal. However, this still has a drawback of having the window size fixed, in addition to the disadvantage carried out from the Fourier Transform that appears at sharp edges and discontinuities of the signal.

Here, the Wavelet analysis can form a next logical step, where a variable window size is used according the level of resolution required.

The similarities and differences between the Fourier analysis and Wavelet analysis are summarized below:

Similarities:

- Both techniques are linear mathematical operations that, when applied to a signal, and they result in a set of coefficients that can be used to recover the original signal if applied properly.
- Both of these techniques map the signal using a set of basis functions; however there is a major difference in the type of basis functions used.
- In both techniques, the basis functions are localized in frequency, meaning that they can be used to detect the signal content at a specified frequency.

Differences:

- The most interesting difference is the difference in the basis functions used to analyze the signal and their localization is space or *compactness*. Fourier analysis uses sine and cosine functions, which are not limited or localized in time. On the other hand, wavelet analysis uses an infinite set of possible wavelet functions that can be chosen depending on the desired application. The beauty is that these wavelet functions are localized in space, or *compact*.
- Another difference is that the window size used in the analysis is fixed in the Fourier analysis or the STFT, while it is variable in wavelet analysis.
- Wavelet analysis proves its efficiency when the signal has discontinuities and sharp edges, while Fourier analysis fails.

The variation of the window size depends on the level of detail of the frequency analysis. For example, in order to determine high-resolution or detailed frequency analysis, a small window with a compressed wavelet (i.e. low scale) is used to detect the rapidly changing details. The opposite is true: a high scale stretched wavelet (bigger window) is used to detect slow changing features and low frequency localized contents.

What is a Wavelet?

A wavelet is a function that has to satisfy certain mathematical properties and conditions. Wavelets are used as the basis functions of the wavelet transform. They compromise an infinite set and different wavelet families which differ in how compact, localized in space and frequency, and smooth they are. Each wavelet family has its advantages and disadvantages which specify how they can be applied into applications.

Each wavelet family has a prototype wavelet function that is called the analyzing wavelet or the mother wavelet. *Daughter* wavelets are derived from the mother wavelet by scaling and shifting operations.

Sample wavelet functions with their corresponding names are shown in Figure 48.

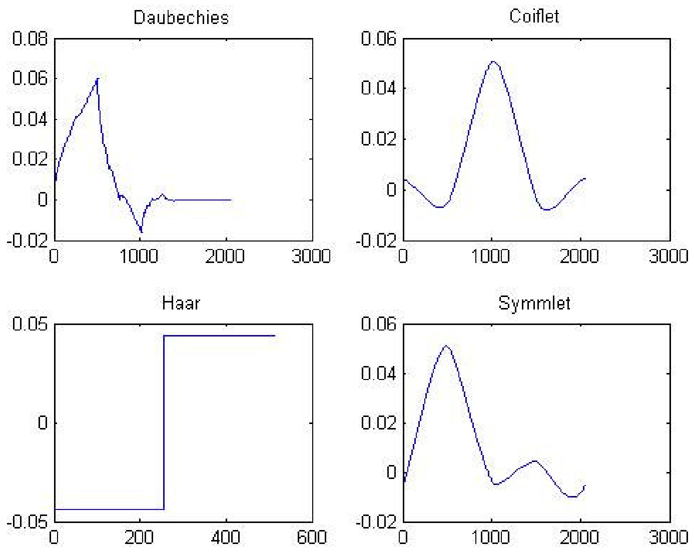


Figure 48. Example of wavelet functions

The scaling compresses or expands the wavelet in time, and shifting moves the wavelet in time. Compression of the wavelet defines the window size and thus the frequency resolution, and the shifting provides the analysis of the signal at different intervals.

The Discrete Wavelet transform (DWT) is the tool that is used to apply wavelet analysis to the signal under study. In the DWT, the scales and shifts are done based on powers of two, or what is called *dyadic* scales and positions. This proved to be very much efficient in processing.

The DWT is computed in five simple steps:

1. Choose a wavelet according to the desired application, and start comparing it to the first section of the original signal.
2. Compare results in a number that represents the level of correlation of the wavelet with the chosen section of the signal. It can be considered as the correlation level.
3. Shift the wavelet to the next slice of the signal and find the correlation coefficient, and repeat these steps until the whole signal is covered by the wavelet with the specified scale.
4. Scale the wavelet, and repeat the aforementioned steps: find the correlation of this scaled wavelets with different positions of the signal until the entire signal is covered.
5. Repeat the steps from 1 to 4 for all the scaled wavelets.

These steps result in a matrix or a function of scale and position that can be plotted in three dimensions, where the first and second dimension are the scale and position, and the third dimension is the level of correlation with the considered wavelet. The resulting 3D plot is called the *scalogram*.

Very Brief Mathematical Summary

Equation (A3-1) describes how the STFT is computed.

$$S(u,w) = \int_{-\infty}^{+\infty} f(t)g(t-u)e^{-jwt}dt \qquad \text{(A3-1)}$$

where *f(t)* is the signal under study, *g(t)* is the window function used to slice the signal and apply the Fourier Transform, *u* is the time shift of the window, and *w* is the frequency under study.

The resulting matrix *S* is the STFT, which is a function of time shift and frequency. Note here that the choice of the window function *g* affects the values of the STFT, and that the window size stays constant for all intervals and time shifts.

The wavelet transform is computed using the (A3-2).

$$W(u,s) = \int_{-\infty}^{+\infty} f(t) \frac{1}{\sqrt{s}} \psi^{*}\left(\frac{t-u}{s}\right) dt \qquad \text{(A3-2)}$$

where ψ is the mother wavelet function and *s* is the scale value that is used to scale the wavelet function. The resulting matrix *W* is a function of time shifts and scales. Note here that the window size specified by the wavelet function is changing for each scale value.

Appendix 4

The Definition of "Methodology"

By the Merriam Webster online dictionary

www.webster.com

Methodology:

1: a body of **methods**, rules, and postulates employed by a discipline**:** *a particular procedure or set of procedures*

2: the analysis of the principles or procedures of inquiry in a particular field

A **Method** is:

1: a procedure or process for attaining an object: as **a** (1)**:** a systematic procedure, technique, or mode of inquiry employed by or proper to a particular discipline or art (2)**:** a systematic plan followed in presenting material for instruction **b** (1)**:** a way, technique, or process of or for doing something (2)**:** a body of skills or techniques

2: a discipline that deals with the principles and techniques of scientific inquiry

3-a: orderly arrangement, development, or classification**:** plan **b:** the habitual practice of orderliness and regularity

4: capitalized**:** a dramatic technique by which an actor seeks to gain complete identification with the inner personality of the character being portrayed.