Linköping Studies in Science and Technology
Thesis No. 1173

Design of multi-standard baseband processors

Anders Nilsson

Linköpings universitet
INSTITUTE OF TECHNOLOGY

LiU-TEK-LIC-2005:28
Department of Electrical Engineering, Linköping University
SE-581 83 Linköping, Sweden
Linköping 2005

ISBN: 91-85299-69-3
ISSN: 0280-7971

Abstract

Efficient programmable baseband processors are important in order to enable true multi-standard radio platforms and software defined radio systems. The ever changing wireless network industry also requires flexible and versatile baseband processors to be able to adapt quickly to new and updated standards. The convergence of mobile communication devices and systems require multi-standard capabilities in the processing devices. The processors do not only need the capability to handle differences in a single standard, often there is a great need to cover several completely different modulation methods such as OFDM, CDMA and single carrier modulation with the same processing device. All this requires a programmable baseband processor because a pure fixed-function ASIC solution is not flexible enough. Furthermore, ASIC solutions for multi-standard baseband processing are less area efficient than their programmable counterparts since processing resources cannot efficiently be shared between different operations and standards. This project was initiated for the above mentioned reason as a continuation of a previous baseband processor project at the research group. Accordingly, this thesis is devoted to the design of area efficient, low clock rate, fully programmable baseband processors. A reduction of the clock rate will simplify the design of the processor as well as save power in the application. Since most multi-standard processing devices will be used in a mobile environment, low power is essential. Normally, extra computing resources must be added to a system designed for low clock rate operation compared to a regular solution, resulting in a higher area and complexity of the chip.

In this project effort has been made to create efficient base architectures maintaining a low area and clock rate while also maintaining flexibility and processing capability. At the same time design methods for the required DSP execution units within the processor have been developed.

Usually general baseband processing includes many tasks such as error control coding/decoding, interleaving, scrambling etc, however in this thesis because of time and resource limitations, the focus is on the symbol related processing, although the bit manipulation and forward error correction tasks are also studied regarding acceleration.

Preface

This thesis presents my research from October 2003 to April 2005. The following three papers are included in the thesis:

- Anders Nilsson, Eric Tell and Dake Liu; "An accelerator architecture for programmable multi-standard baseband processors"; in *Proceedings of the Intl. conference on Wireless networks and Emerging technologies, WNET2004*; Banff, Canada 2004, pp 644-649
- Anders Nilsson, Eric Tell and Dake Liu; "A fully programmable Rake-receiver architecture for multi-standard baseband processors"; in *Proceedings of the Intl. conference on Networks and Communication systems, NCS2005*; Krabi, Thailand 2005, pp 292-297
- Anders Nilsson, Eric Tell, Daniel Wiklund and Dake Liu; "Design methodology for memory-efficient multi-standard baseband processors"; submitted for review to Asia-Pacific Communications Conference, APCC'2005.

The following five papers also related to my research are not included in the thesis:

- Anders Nilsson, Eric Tell and Dake Liu; "A Programmable SIMD-based Multi-standard Rake-receiver Architecture"; in *Proceedings of EUSIPCO'2005*; Antalya, Turkey September 2005
- Anders Nilsson, Eric Tell and Dake Liu; "Processor friendly peak-to-average reduction in multi-carrier systems"; in *Proceedings of Swedish System-on-a-chip conference*; Båstad, Sweden 2004.

- Anders Nilsson, Eric Tell and Dake Liu; "Acceleration in programmable multi-standard baseband processors"; in *Proceedings of Radiovetenskap och Kommunikation 2005*; Linköping, Sweden 2005.
- Eric Tell, Anders Nilsson and Dake Liu; "A Programmable DSP core for Baseband Processing" in *Proceedings of NEWCAS*, Quebec, Canada, June 2005
- Eric Tell, Anders Nilsson and Dake Liu; "A Low Area and Low Power Programmable Baseband Processor Architecture" in *Proceedings of IWSOC*, Banff, Canada, July 2005

Acknowledgments

First of all I would like to thank Professor Dr. Dake Liu for accepting me as a PhD-student in his group. I would also like to thank Eric Tell and Dr. Olle Seger who also are involved in the baseband processor project, for their support and co-operation in the project. Further, I would also like to thank Dr. Daniel Wiklund for his support in understanding the WCDMA standard documents. All my fellow PhD students, Andreas Ehliar, Johan Eilert, Per Karlström, Di Wu and all my other co-workers at the Computing Engineering group: thanks a lot for your ideas and comments about my work.

Thanks to the research engineers Anders Nilsson Sr and Niklas Carlén for their help with the computers and PCB cad system. Many thanks goes to Greger Karlströms for his practical support and habit of luring me off in the middle of the night at the office and making me take some time off from the hard work, which I needed most.

Finally I would like to thank my parents, Hatice and Bo Nilsson for their invaluable support, encouragement and love.

This work was supported by the Swedish Foundation for strategic Research (SSF) through the Strategic Integrated Electronic Systems Research at Linköpings Universitet (STRINGENT) center.

Anders Nilsson
Linköping, May 2005

Contents

Abbreviations

- AGC: Automatic gain control.
- ASIC: Application specific integrated circuit.
- ASIP: Application specific instruction set processor.
- ADC: Analog to Digital Converter.
- BBP: Baseband processor.
- CDMA: Code division multiple access.
- DAC: Digital to Analog Converter.
- DSP: Digital signal processing or processor.
- FDD: Frequency division duplex.
- FPGA: Field programmable gate array.
- ISA: Instruction-set atchitecture.
- MAC: Medium Access Control or Multiply-and-accumulate unit.
- OFDM: Orthodonal frequency division multiplex.
- RF: Radio frequency or register file.
- TDMA: Time division multiple access.
- SDR: Software defined radio.

- SIMD: Single instruction multiple data.
- TD-SCDMA: Time division synchronous CDMA.
- TDD: Time division duplex.
- TTM: Time to market.
- VLIW: Very long instruction word.
- WCDMA: Wideband CDMA.

Part I

Introduction

Chapter 1
Introduction

Baseband processing and baseband processors will become increasingly important in the future because more and more devices will be connected together by means of wireless links. Since the number of radio standards grows increasingly fast and the diversity among the standards increases, there is a need for a processing solution capable of handling as many standards as possible at the same time not consuming more chip area and power than a single-standard product. This trend is driven by the convergence of mobile communication devices. In order to achieve the required flexibility and to reach optimal solutions, programmable processors are necessary in contrast to other solutions such as accelerated standard processors and similar devices. A programmable solution will also help companies to reduce the Time To Market (TTM) since they can reuse the same hardware platform over several product generations and only do software changes in between.

Programmable baseband processors are also required in order to fulfill the old dream of fully software defined radio (SDR) systems. In the future, software defined radio systems will most certainly be used to both enable truly worldwide usable products and to efficiently utilize the scarce radio frequency spectrum available, for a wide consumer population. Furthermore existing solutions based on ordinary digital signal processors, do not have the computing power required to perform the computations needed to handle most modern radio standards, and the

power consumption of such circuits is high due to their inherent flexibility. To enable programmable baseband processors, we need new processor structures which are optimized for this computing domain but still very flexible within the frame of the same domain.

The goal of this research project is to create new such power- and area efficient architectures, suitable for future multi-standard radio networks. In this thesis the results of the research are presented. The results include both a new processor architecture and design methods for this architecture.

1.1 Scope of the thesis

The scope of this thesis is programmable baseband processors and how they can be designed. Special attention has been paid to three distinct areas of baseband processor design:

- Selection and design of execution units and accelerators.
- Multi-standard issues.
- Scheduling and instruction issue.

The over-all goal is to find low clock rate and low power solutions. General baseband processing includes many tasks such as error control coding/decoding, interleaving, scrambling etc, however in this thesis because of time and resource limitations, the focus is on the symbol related processing, although the other tasks are also studied regarding acceleration. Symbol related processing is defined as the operations performed between the map/de-map operation and the radio interface. The thesis presents both my current and recent research regarding these three areas and also gives an introduction of programmable baseband processing.

1.2 Organization

The thesis is divided into three parts; in Part I background about baseband processing and related work is given. In part II, the unique properties of baseband processing are discussed in chapter 4. Chapter 5 describes the design methodology and Chapter 6 describes the processor architecture, whereas chapters 7 - 8 highlight selected areas of the research project. The research methodology and my achievements are presented in chapters 9 - 10. Finally, a selection of my published papers are presented in part III.

The purpose of Part I and II is to give an introduction to baseband processing and to introduce the concepts discussed in the papers. The research results are presented in the conference papers included in Part III.

1.3 Included papers

The papers presented in this thesis are the result of research conducted during the background investigations in the design of a new multi-standard baseband processor architecture. The papers cover a wide range of topics related to baseband processing and baseband processor design. The papers cover:

- Overall processor architecture.
- Selection of accelerators.
- CDMA based channel equalization.
- An efficient OFDM processor architecture and scheduling.

The first paper presents an accelerator architecture for multi-standard baseband processors. The paper includes accelerator selection methodology as well as a proposal for accelerators and a base architecture to use in a merged WLAN and 3G system. The second paper discusses the design

and implementation of multi-standard rake receiver functions in a programmable baseband processor. It also presents a novel complex short MAC (CSMAC) execution unit which delivers processing power enough to run all rake functions in software with the same efficiency as in a fixed function device. The third paper is devoted to a design methodology for memory efficient baseband processors targeted at OFDM applications such as WiMAX or WLAN systems. The paper presents both a design methodology and a base architecture which yields both area and power efficient processors for OFDM applications.

Chapter 2
Background

2.1 System perspective on baseband processing

A typical wireless communication system usually contains two to three different processors as shown in Figure 2.1. The processors are:

- A baseband processor.
- A medium access control processor.
- An application processor.

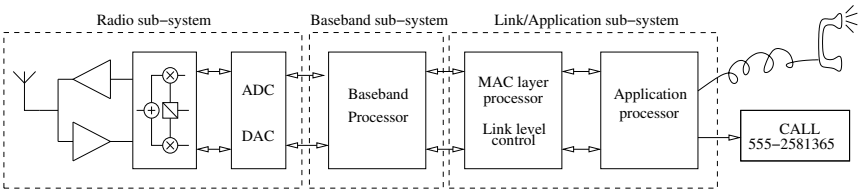

Figure 2.1: Example of communication system.

The baseband processor is the processor closest to the radio-interface in the processing hierarchy. The baseband processor is responsible for modulating the bits received from a higher layer into a discrete waveform, sent to the DAC and then transmitted over the air. The baseband

processor is also responsible for detecting a received waveform, to synchronize to it and extract information bits from the received waveform. These bits are then delivered to the higher layers which assemble the data into user services such as wireless LAN packets or voice packets in a cellular telephone.

If the application only requires a smaller amount of control functions, the MAC layer functionality could be merged with application into a single processor. However due to the nature of baseband processing, baseband processing tasks are usually separated from the application processor.

2.1.1 Baseband processing tasks

The processing tasks within the baseband processor can be divided into six distinct classes, three in the transmit path and three in the receive path [1] [2]. In the transmit path the classes are:

- Channel coding.
- Modulation.
- Symbol shaping.

And in the receive path the classes are:

- Filtering, synchronization, gain-control.
- Demodulation, channel estimation and compensation.
- Forward error correction.

These processing tasks are illustrated in figure 2.2. In general, this schematic view of baseband processing tasks is true for most radio systems. In the transmit path from the MAC layer to the radio, air interface data are first sent to a channel coding step which adds redundancy to the transmitted data, interleaves data both in time and in frequency and scrambles the data to remove regularities in the data stream. The binary data are fed to the modulator stage which converts the binary data into

one or many *symbols*. A symbol can be represented as one or a series of complex numbers representing a waveform or a single value. This stream of complex numbers is then sent to the symbol shaping stage which filters and smooths the signal in order to remove unwanted spectral components.

At the receiver side, all the operations are performed in reverse. The stream of complex data values from the ADC is first fed to a digital front-end stage which contains a digital filter, synchronization functions and gain control functionality. The filtered and synchronized symbol stream from the digital front-end is then fed into the demodulator which performs demodulation and channel compensation on the received symbols. Binary data are then extracted from the received symbols and fed to the forward error correction unit, which utilizes the redundancy added by the channel coder stage in the transmitter to correct for any transmission errors encountered.

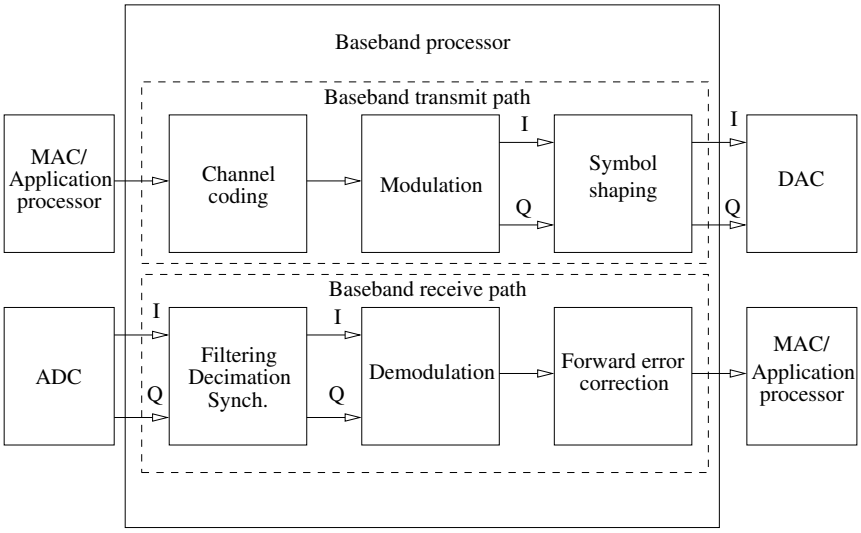

Figure 2.2: Tasks of a baseband processor.

2.2 Baseband processing devices

To implement a baseband processing algorithm, it is necessary to have a suitable processing device. In general there are three different classes of processing devices suitable for baseband processing. The devices range from rigid but powerful devices to very flexible, less efficient devices and everything in between the two extremes. In this research project the following classes of processing devices are identified:

1. General purpose DSP processors.
2. Fixed function ASICs.
3. Application specific DSP processors.

The general purpose DSP processors are most versatile and flexible while the fixed function ASICs have the possibility to be completely optimized for one single task. Application specific instruction set processors are a trade-off between the flexibility of a general purpose DSP processor and the processing power of fixed function hardware.

2.3 General purpose DSP processors

General purpose signal processors deliver unparalleled flexibility to a signal processing system. However, this flexibility consumes both chip area and power. General purpose processors are also more complex to design and implement as no limitations or other simplifications can be done on the architecture, compared with other architectures. Furthermore, general purpose DSPs are not optimized for complex computing. A single scalar general purpose DSP with double accumulator registers requires at least four cycles in order to complete a single complex-MAC operation. This limits the usefulness of general purpose DSPs in baseband processing, since most baseband processing is carried out in the complex domain.

Most common in baseband processing are general purpose fixed/floating -point DSP processors from Texas Instruments [3] and Analog Devices [4].

2.4 Fixed function ASICs

Fixed function ASICs are the opposite to general purpose DSPs, since they lack flexibility. However the lack of flexibility enables the designers to fully optimize the solution for the chosen task. Fixed function ASIC baseband processing solutions can achieve very high performance and in the same time have low power consumption. A drawback with this technology is the excessive hardware overhead for multi-standard devices; since most computing resources cannot easily be shared between different standards. Fixed function ASICs are often combined with a general purpose RISC processor for control. Such WLAN solutions are for example available from Atmel, AT76C509 [5].

2.5 ASIP DSPs

Application specific instruction set processors can achieve maximum performance and flexibility under limited area and power constraints since they can be optimized for a certain application domain. In this research project, only ASIP DSPs are considered for implementation since they can provide an optimal trade-off between flexibility, power consumption and performance.

2.6 The need for programmability

There are three main reasons for using programmable baseband processors. The most important reason is the processors inherent flexibility. This flexibility allows the designer to implement diverse standards within the same device. The device will also be more area efficient since computing resources are reused between different standards.

The second important reason is the ability to allow standard updates in existing wireless devices. This will not only extend the useful lifespan of a product, it will also reduce the non-recurring engineering costs for manufacturers of wireless equipment since they can reuse the same de-

vice with only a different software for a longer time, thus extending the hardware's life span and at the same time reducing development costs.

The last but not the least reason is to provide dynamic allocation of computing resources during operation of a wireless device. This could be used to trade off mobility handling capability versus data rate. For example, in severe channel conditions the mobile device could spend the available processing power on channel compensation but when the channel condition are good it could use the resources on providing higher data rate to the end user. The need for programmable baseband processors is further discussed in Chapter 4.

References

[1] John B. Andersson, *Digital Transmission Engineering* IEEE Press 1999, ISBN 0-13-082961-7

[2] H Heiskala, J T Terry, *OFDM Wireless LANs: A Theoretical and practical guide*, Sams Publishing, 2002

[3] Texas Instruments, http://www.ti.com/

[4] Analog Devices Inc.,http://www.analog.com/

[5] Atmel Corp., http://www.atmel.com/

Chapter 3
Related work

3.1 Introduction

The programmable baseband processor research has only been around for a few years even though the concept of SDR has been around for much longer time than that. The dream of creating a true SDR system has led to the creation of a number of different processing architectures, everything from standard processors to specialized configurable execution units based on FPGA technology.

Unfortunately there are few publications in the area of programmable baseband processing. Also, most academic work tend to focus on a small part of a programmable baseband processing system or focus more on design-tools rather then on new architectures.

Classical industrial solutions are often conservative and the instruction set of the processors is often not optimized for baseband processing. Instead classical, well known processing solutions are modified to manage baseband processing tasks, resulting in inefficient solutions.

The most common baseband processing approach today is to combine standard RISC processors, such as ARM [1] or MIPS [2] or a single scalar DSP with ASIC baseband processing blocks. The drawback of this solution is the lack of flexibility of the ASIC parts and the extra overhead imposed by the superfluous flexibility of the general purpose processor

core. Usually the lack of flexibility in the ASIC block cannot easily be compensated for by extra processing capability in the controller processor.

3.2 Other architectures

In this section a short introduction of the different commercial solutions is given. However, since only Atmel provides documentation of their processors without a non-disclosure agreement, not many details of the processor architectures are known.

3.2.1 Sandbridge Technologies Inc

Sandbridge Technologies [3] provide a processor architecture named "Sandblaster" [4]. The Sandblaster DSP contains a RISC based integer execution unit and SIMD execution units.

3.2.2 SystemOnIC, now Philips research

Before SystemOnIC was acquired by Philips, they developed a programmable baseband processor named Hipersonic I [5].
The Hipersonic processor contains three different parts:

- Hard-wired logic for computationally intensive processing.
- A flexible DSP (OnDSP).
- An optional protocol processor for scalar computations.

The OnDSP is a DSP based on SIMD execution units.

3.2.3 Morpho technologies

Morpho Technologies [6] provides a baseband processor architecture named "M-rDSP architecture" which consists of a 32 bit RISC processor and a reconfigurable cell array of 8 to 64 cells. Each reconfigurable cell has

an ALU, MAC, and highly optimized specialized functional units that can be used for different wireless applications.

3.2.4 Atmel mAgic DSP

With the mAgic DSP architecture, Atmel [7] claims to have the world's first complex domain, extended precision VLIW DSP core for SoC implementation. The mAgic core provides single-cycle execution of complex arithmetic operations, such as FFT butterflies and arithmetic operations. However, since the processor is based on VLIW technology, the program memory cost is high and the control overhead is large.

3.3 Comparison

A comparison of the architecture presented in this thesis with the previously presented architectures yields several important differences. None of the commercial solutions uses native complex data-paths nor has support for vector instructions according to our definition of vector instructions. Another main difference is the single-issue CSIMD technology used in the presented architecture. The CSIMD architecture is presented in Chapter 6, in Paper 2 and in [8]. The other solutions are based on conventional pure VLIW, SIMD or array elements with its associated control overhead. The closest architecture in terms of complex oriented computing is the mAgic VLIW architecture from Atmel. The mAgic architecture can concatenate several real-valued data-paths to form a complex operation.

The presented architecture is similar to the architecture of the BBP1 chip [8] created earlier in this research project. Compared with publicly available on-line resources from respective processor vendor, the BBP1 chip is both more powerful and area efficient than the other architectures. However, the other presented architectures are generally more flexible outside the baseband processing domain, then the architecture presented in this thesis.

References

[1] ARM Limited, http://www.arm.com

[2] MIPS Technologies, http://www.mips.com/

[3] Sandbridge Technology, http://www.sandbridgetech.com/

[4] John Glossner et al. A Software-Defined Communications Baseband Design. pp 120-128, IEEE Communications Magazine, January 2003

[5] J. Kneip et al. Single chip programmable baseband ASSP for 5 GHz wireless LAN applications. IEICE TRANS. ELECTRON, Feb 2002.

[6] Morpho Technology, http://www.morphotech.com/

[7] Atmel Corp., http://www.atmel.com/

[8] Eric Tell, Anders Nilsson and Dake Liu; "A Low Area and Low Power Programmable Baseband Processor Architecture" in *Proceedings of IW-SOC*, Banff, Canada, July 2005

Part II

Baseband processing

Chapter 4
Baseband processing

4.1 Introduction

In this chapter some of the unique properties of baseband processing and some of the challenges faced in a baseband processing system are described. The system environment of the baseband processor in the research papers is also described and clarified in this chapter. All radio and channel impairments considered are also presented here.

4.2 Challenges

In this research project the following four demanding challenges for the baseband processor to manage regardless of modulation methods and standards are identified:

- Multi-path propagation and fading. (Inter-symbol interference)
- High mobility.
- Frequency and timing offsets.
- Radio impairments.

These four challenges impose a heavy computational load for the processor. Besides the above mentioned challenges, baseband processing in general also faces the following two challenges:

- High dynamic range.
- Limited computing time.

4.2.1 Multi-path propagation and fading

In a wireless system data are transported between the transmitter and receiver through the air and are affected by the surrounding environment. One of the greatest challenges in wide-band radio links is the problem of multi-path propagation and inter-symbol interference. Multi-path propagation occurs when there are more than one propagation path from the transmitter to the receiver. Since all the delayed multi-path signal components will add in the receiver, inter-symbol interference will be created. Since the phases of the received signals depend on the environment, some frequencies will add constructively and some destructively, thus destroying the original signal. Unless the transmitter and receiver sit within an echo-free room or any other artificial environment the transmission will usually be subjected to multi-path propagation. There is only one common communication channel which is usually considered echo-free viz. satellite links. Multi-path propagation is illustrated in Figure 4.1.

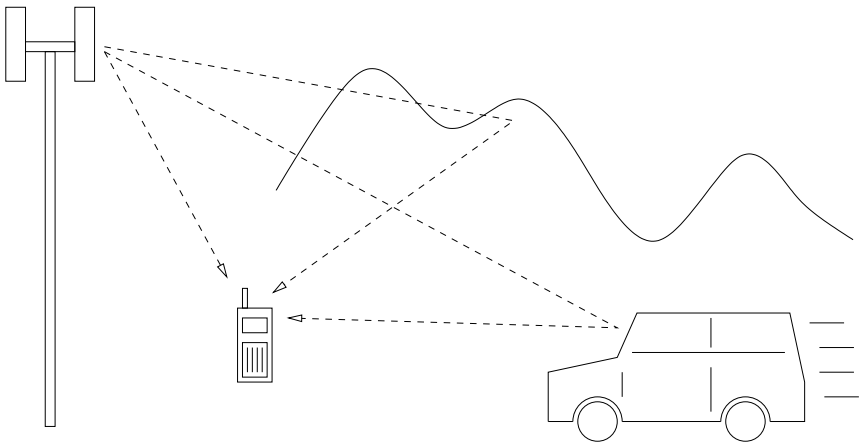

Figure 4.1: Multi-path propagation.

Multi-path propagation can be characterized by the channel impulse response. From the channel impulse response several important parameters can be derived. The most important parameter is the RMS *delay-spread*, σ_τ, which describes the RMS distance in time between multi-path components. The channel impulse response, also known as the Power Delay Profile (PDF) of a channel is illustrated in Figure 4.2.

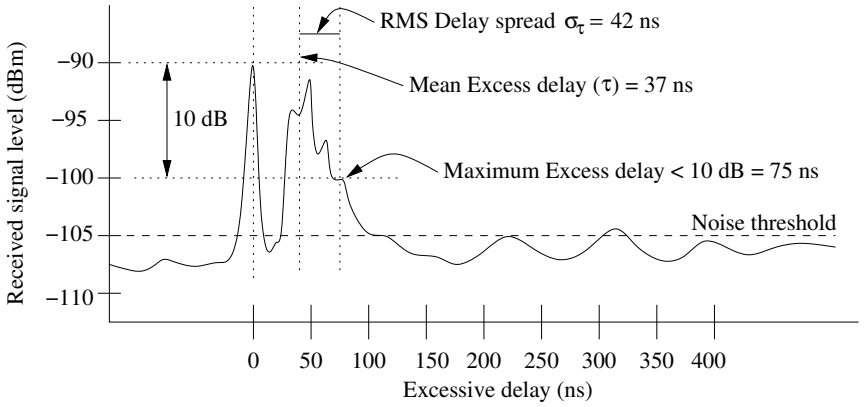

Figure 4.2: Power Delay Profile

The delay-spread imposes a limit of the shortest symbol period usable. This will in turn restrict the data-rate of a transmission system. A rule of thumb is to use symbol durations, which are at least 10 times the delay spread ($10 \cdot \sigma_\tau$) if the system operates without advanced channel equalizers.

In the example in Figure 4.2 the shortest symbol duration would be limited to 420 ns, which give a symbol rate of 2381 symbols/sec. In this example, a delay-spread of 42 ns was used, which corresponds to a maximum path-distance of about 12 meters. In outdoor systems the delay-spread is often in the range of several micro-seconds, which further limits the symbol rate. Common reference "channels" are specified by various institutes and standardization organs to be used in benchmarking of channel equalizers. In Table 4.1, two common channels are presented, the ITU Pedestrian A and ITU Vehicular Channel Model B [1]. The channel

models represent a user walking 3 km/h in an urban environment and traveling in a car at 60 km/h respectively. The channel models specify a number of multi-path components (taps) with their delay and average power. The phase and amplitude of the multi-path component are assumed to be Raleigh distributed.

Table 4.1: Power delay Profiles for common channels

	ITU Pedestrian A		**ITU Vehicle B**	
Tap	**Delay (ns)**	**Average power (dB)**	**Delay (ns)**	**Average power (dB)**
1	0	0	0	-2.5
2	110	-9.7	300	0
3	190	-19.2	8900	-12.8
4	410	-22.8	12900	-10.0
5	-	-	17100	-25.2
6	-	-	20000	-16.0

The effects of multi-path-propagation and resulting inter-symbol interference are referred to as *fading*. For narrow-band systems (with long symbols), the effect of inter-symbol interference can be assumed to be constant over the entire frequency of the channel used (flat fading). However, in wide-band systems the effects of inter-symbol interference will cause frequency dependent fading, causing parts of the transmitted signal spectrum to be destroyed. To combat frequency selective fading in wide-band systems, advanced equalizers must be used to compensate for multi-path channels. Another solution to avoid the problem of wide-band channels in multi-path environments is to divide the wide-band channel onto many narrow-band channels and treat them as flat faded channels. This is the basic principle of OFDM transmission systems [2]. However, since it is not possible to use OFDM technology in all situations, advanced equalizers must be employed for example in CDMA networks.

4.2.2 Dynamic range

Another problem faced in practical systems is the large dynamic range of received signals. Both fading and other equipment in the surrounding will increase the dynamic range of the signals arriving in the radio front-end. It is common with a requirement of 60-100 dB dynamic range handling capability in the radio front-end [3]. Since it is not practical to design systems with such large dynamic range, automatic gain control (AGC) circuits are used. This implies that the processor measures the received signal energy and adjusts the gain of the analog front-end components to normalize the energy received in the ADC. Since signals falling outside the useful range of the ADC cannot be used by the baseband processor, it is essential for the processor to continuously monitor the signal level and adjust the gain accordingly. Power consumption and cost of the system can be further decreased by reducing the dynamic range of the ADC and DAC as well as the internal dynamic range of the number representation in the DSP processor. By using smart algorithms for gain-control, range margins in the processing chain can be decreased.

4.2.3 Mobility

Normally, the channel is assumed to be time invariant. However, if the transmitter or receiver moves, the channel and its fading will be time varying. Mobility in a wireless transmission causes several effects, the most demanding effect to manage is the rate at which the channel changes. If the mobility is low, e.g. when the channel can be assumed to be stationary for the duration of a complete symbol or data packet, the channel can be estimated by means of a preamble or similar known sequence. However, if mobility is so high that the channel changes are significant during a symbol period, this phenomenon is called *fast fading*. Fast fading requires the processor to track and recalculate the channel estimation during reception of user payload data. Hence, it is not enough to rely on an initial channel estimation performed on a packet or frame start.

Mobility can be described by the *channel coherence time*, T_c, which is

inversely proportional to the maximum Doppler shift of the channel. For example, a WCDMA telephone operating at 2140 MHz will encounter a Doppler shift of 118 Hz when the telephone travels at 60 km/h towards the base-station. For a correlation of 0.5 of the current channel parameters and the channel parameters after the time T_c, the following formula applies:

$$T_c = \frac{9}{16\pi f_m}$$

Where f_m is the maximum Doppler shift of the channel. This yields the channel coherence time of the previous example to be 1.5 ms. For WCDMA, there are modes specified for up to 250 km/h, corresponding to a Doppler shift of 492 Hz and a channel coherence time of $T_c = 363\mu s$.

At 250 km/h the coherence time of the channel is less than half of the slot-time, which implies that the processor must track channel changes during the reception of a data slot. The Doppler shift will also create the same effects as frequency offsets. However, the effects of frequency offsets can easily be compensated by de-rotating the received data [2].

4.2.4 Radio impairments

Along with distortion and other effects added by the channel, the transmission system can in itself also have impairments. Such impairments could be:

- Carrier frequency offset.
- Sample frequency offset.
- DC-offset.
- I/Q non-orthogonality.
- I/Q gain mismatch.
- Non-linearities.

The items listed above are all common impairments in radio-front ends and they affect the performance of the whole system. Since the correction of the impairments require extra computing resources, it is essential to include the correction of the impairments as early as possible in the project.

One common problem in direct up-conversion transmitters is the problem of non-orthogonality of the I and Q baseband branches. Often, the quadrature-phase carrier is created by delaying the in-phase carrier to create a $90°$ phase-shift. However, this phase-shift will be frequency dependent and only provide $90°$ phase-shift at one frequency. Non-orthogonality will create severe problems for QAM modulations of high order, and create unwanted AM modulation of constant-envelope modulation schemes. The effects of non-orthogonality on 64-QAM are illustrated in Figure 4.3.

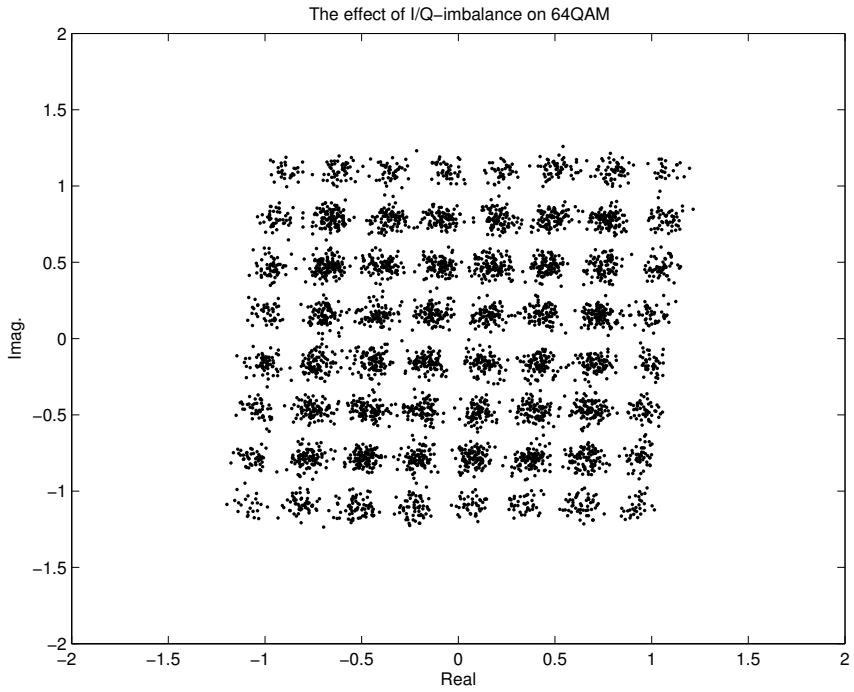

Figure 4.3: Non-orthogonality

Radio impairments are discussed in the books [3] and [4] among others.

4.2.5 Processing capacity challenges

Since baseband processing is a strict hard real-time procedure, all processing tasks must be completed on time. This imposes a heavy workload for the processor during computationally demanding tasks such as Viterbi-decoding, channel estimation and gain control calculations. In a packet based system, the channel estimation, frequency error correction and gain control functions must be performed before any data can be received.

This may result in an over-dimensioned processor, since the processor must be able to handle the peak work load, even though it may only occur less than one percent of the time. Here, in this case programmable DSPs have an advantage over fixed function hardware since the programmable DSP can rearrange its computing resources to make use of the high available computing capacity all time.

4.3 System simulation environment

During this research project, all the above mentioned challenges and impairments have been taken into consideration when benchmarking the system and selecting algorithms. This is essential for guaranteeing the quality of research results provided by this project.

4.4 Task chains

Another interesting property of baseband processing is its data flow structure. Analysis of baseband algorithms shows that the baseband algorithms are very stable and have little data dependencies. It can also be shown that baseband processing algorithms can be decomposed into task chains with little backward dependencies between computing tasks.

By utilizing this fact, very efficient computing hardware with flexibility within the domain of baseband processing can be developed. By optimizing an ASIP processor architecture for computations based on long vectors of complex-based data, the processing power over other solutions can be improved while reducing both chip area and power consumption of such solution. Task chains can also be used to create task-level pipelines in order to increase the processing parallelism. This is enabled by the few backward dependencies between processing stages.

The base architecture used in this project relies on the decomposition of most baseband tasks into task-chains.

References

[1] ITU ITU-R M.1225, Guidelines for evaluations of radio transmission technologies for IMT-2000, 1997.

[2] H Heiskala, J T Terry, *OFDM Wireless LANs: A Theoretical and practical guide*, Sams Publishing, 2002

[3] Bosco Leung, Behzad Razavi, "RF Microelectronics", Prentice Hall PTR, ISBN 0-13-887571-5, 1998

[4] "VLSI for Wireless Communication", Prentice-Hall, Pearson Education, Inc. ISBN 0-13-861998-0, 2004

Chapter 5
Design methodology

5.1 Introduction

In this chapter, the processor design methodology and some of the tools and evaluation methods applied are described. The overall design method can be described by the following steps:

1. Analysis of covered standards.
2. Algorithm selection.
3. Analysis of processing tasks.
4. Selection of execution units.
5. Specification of the instruction set.
6. Benchmarking and profiling.
7. Scheduling and mapping onto selected architecture.
8. Implementation.

Between each of these stages in the methodology, the models used are refined and the result of the previous stage is evaluated by means of benchmarking and profiling. It is well known that good methods of evaluation are essential for the method to be successful.

5.1.1 Analysis of the covered standards

In this first stage of the method, the covered standards are analyzed in terms of estimated memory usage, roughly which algorithms to use and which precision of the executions units is needed. Then, this information is used to determine a good starting point for the base architecture to use in the later stages of the method. In this stage, high-level (system) models are created in either Matlab or C++. The models are used to get further knowledge and information about system and hardware requirements.

5.1.2 Algorithm selection

When the covered standards have been analyzed and high-level models have been created, it is time to do the detailed algorithm selection. In a high level model, it does not make any difference how a correlation is performed, but if the model is to be used to select and dimension hardware resources it is important to know exactly how it is going to be implemented. Algorithm selection is one of the most important tasks during the design of a baseband processor since the result will have a great influence on the rest of the project. In general, telecommunication standards are only specified on the transmitter side. Often no hints about suitable receiving algorithms are given. This further complicates the algorithm selection part and illustrates the importance of good algorithm choices.

5.1.3 Analysis of processing tasks and selection of execution units

Now, that the algorithms have been selected, the kernel functions are analyzed and execution units are selected and dimensioned. Also special execution units and accelerators are designed in this stage.

5.1.4 Scheduling and mapping

This is the final stage of the design method before the implementation stage and release of the Instruction Set Architecture (ISA). Now the algo-

rithms and hardware are scheduled in detail to ensure full architectural support for the desired standards and operations.

5.1.5 Instruction set specification

When the algorithms and execution units have been selected, the instruction set of the processor is fixed. The instruction set is based on benchmarking done in the earlier stages of the methodology. The instruction set architecture acts as a interface between software and hardware subprojects in the research project. When instructions are selected careful investigation of benchmark results is necessary since excessive instructions will increase the necessary width of the instruction word. The number of instructions is a trade-off between ease of software implementation and hardware complexity.

5.2 Evaluation

No method can be implemented and executed as a straight list shown above; in reality there are a lot of dependencies between the stages of the method. Also both benchmarking results and knowledge must be passed between the stages. Often several iterations must be performed in order for the project to converge towards a good solution. To help the project converge, it is of greatest importance to have good evaluation methods to help and guide the design process towards "a better solution". Good evaluation methods are also important in order to produce *comparable* results between different baseband processor vendors and architectures. In the area of programmable baseband processor it is extremely hard to compare two different solutions with each other. In general there are only three practical metrics, which could be used in a comparison:

1. Required clock frequency.
2. Chip area.
3. Power consumption.

One problem which was faced during the design process, was to accurately estimate the above mentioned properties from a high level design. The earlier in the design stage, the less accurate results can be tolerated, however many of the design choices made early in the design phase relies on these estimations. Since the effects of early architectural decisions are large on the rest of the project, it is crucial to know the effects of an early design decision. The three metrics mentioned above however can be accurately estimated using results from simulations. When the system has been scheduled, the required clock frequency of the design can be estimated with good accuracy. This frequency can then be used to calculate memory access rates, access power and power of execution units. By using all available data from the simulation environment and commercially available memory generators and similar tools, power and area figures can be estimated with good accuracy. These figures can then be checked with known data from previously manufactured chips with roughly the same topology. Lastly, the area of the chip can be estimated fairly accurately by modern design tools without having to run the complete place-and-route stage. Actually this approach yields fairly accurate estimations which could be used both as comparisons with other solutions and as guidance when improving the design.

5.3 Modeling

In order to achieve correct benchmark results, it is also necessary that the input stimuli used in simulations are correct. When selecting algorithms for implementation of a specific standard, it is not just enough to consider precision requirements, it is also important to consider what other effects the surrounding components of the baseband processor have on both performance and signal quality. One of the main tasks in the specification stage of the requirement specification is to decide what impairments of the actual radio front-end to include compensation for, and what channel conditions to include in the models. In section 4.2.4 of this thesis, a comprehensive list of both radio and channel impairments is discussed.

Chapter 6
Processor architecture

6.1 Introduction

Traditional methods of addressing the baseband processing requirements for advanced wireless terminals use an architecture based on the combination of DSPs and complex custom ASICs. This combination of general-purpose DSPs and custom ASICs has become an expensive and cumbersome approach because of its lack of cost-effective scalability and flexibility to meet the next generation wireless device requirements. There is clearly a need for architectures providing enough flexibility while maintaining low cost and complexity. As a response to this challenge, a new architecture is presented which also serves as a foundation for the research in this thesis. Over the course of the research project, the architecture has been changed, transformed and continuously been improved to facilitate even lower area and higher power efficiency as well as standard support.

6.2 Functional specification

Before the design of a DSP processor is started, it is important to formalize the functionality of the hardware. It is also important to extract the kernel functions. However, kernel functions alone are not enough to start the design of a processor. Care must also be taken to plan for roughly how

many clock cycles there will be available per data symbol. This figure is directly related to the symbol period and the desired clock frequency of the processor. Based on the available number of clock cycles during a symbol and the "cycle cost" of the kernel-functions a good start for the architecture can be selected [1].

6.3 Top-level architecture

Since baseband processing consists of both symbol related computations in the complex domain and bit-manipulation, it is preferable to use a processor architecture which is divided into two separated parts. One part is based on complex valued computing elements and memories and one part contains scalar computations and bit manipulation. This will increase the over-all efficiency of the baseband processor.

These two parts are both controlled from the same controller and data are transmitted between the two parts by a network bridge or a map/de-map-unit. In this research project, the architecture is noted a *Clustered SIMD, CSIMD* architecture, since the architecture consists of several SIMD computing clusters connected to memory banks through a memory cross-bar and an on-chip interconnect network.

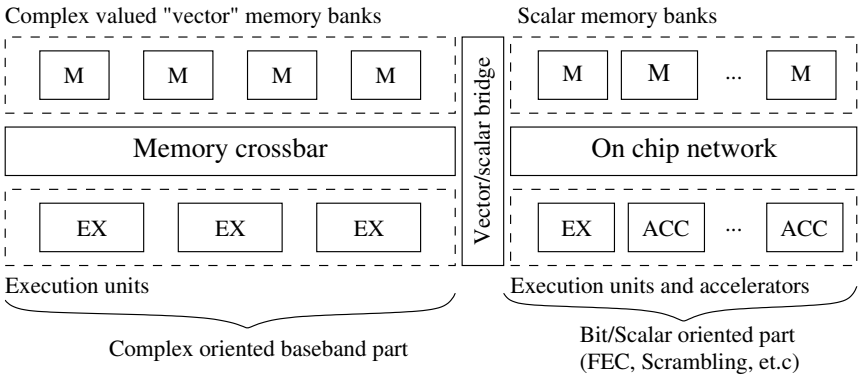

Figure 6.1: Top level architecture of a baseband processor system.

The CSIMD architecture is a completely new processor architecture which drastically reduce the control overhead in baseband processors. A CSIMD based processor can achieve the same processing performance as a VLIW-machine with the control overhead and program memory width of a simple RISC controller [3]. Another feature of the CSIMD architecture is the "single issue" concept which is described in section 6.6.

By connecting memory blocks and execution units to a crossbar switch, the architecture can perform all baseband processing tasks without moving any data between memories. Instead memory blocks are reconnected between execution units. Compared to traditional solutions, this scheme saves many memory operations and allows the clock frequency of the processor to be lowered significantly.

6.4 Execution units

The processor contains four classes of execution units:

1. One scalar DSP processor core.
2. Complex MAC SIMD units.
3. Short complex MAC SIMD units. (Complex ALUs)
4. Accelerators

The scalar DSP core is a small controller core, containing only a real valued MAC and ALU together with other controller functionality [3] [4]. All other execution units within the baseband processor utilizes fixed point complex numbers as their native representation. The DSP core's main task is to coordinate the SIMD execution clusters and to maintain quality control of the processed data (such as gain control and similar functions). The SIMD clusters can execute different tasks simultaneously while every data-path within the cluster performs the same instruction on different data.

6.5 Vector instructions

Analysis and benchmarking of baseband-processing tasks shows that most operations are performed on long *vectors* of complex numbers. To utilize this property of baseband processing and to use task-chains, the new processor architecture has been designed to perform vector operations efficiently. The processor has a special class of instructions, namely vector instructions that operate on a vector of data using memories as direct operand buffers. In this architecture vectors can be of any length between 2 and 128 elements long. By using vector instructions both processing efficiency and code-density can be significantly increased [3] [2]. Address generation for data vectors will be discussed in Section 6.7.

6.6 Instruction issue

The difference between this processor architecture and a VLIW processor is the way instructions are issued to the execution units. By further utilizing the fact that baseband algorithms can be decomposed into task chains, the control-path can be further simplified by only allowing a single instruction to be issued every clock cycle. In a VLIW processor an instruction word is issued to each execution unit each clock cycle. However, since the execution units employed in this architecture are designed to execute long vector operations without any intervention from the controller core, it is not necessary to be able to issue a new instruction to all execution units each clock cycle. To reduce the complexity and improve the efficiency of the control path, the architecture only allows one instruction to be issued every clock cycle. However since most vector instructions operate on long vectors, many RISC instructions can be executed during the vector operation [2]. This is illustrated in Figure 6.2.

The single issue concept and vector operations are discussed in Paper 2 [2] and in [4]. Unlike VLIW-machines, this architecture will allow concurrent vector operations and control flow instructions without the large control overhead and memory usage of a pure VLIW machine.

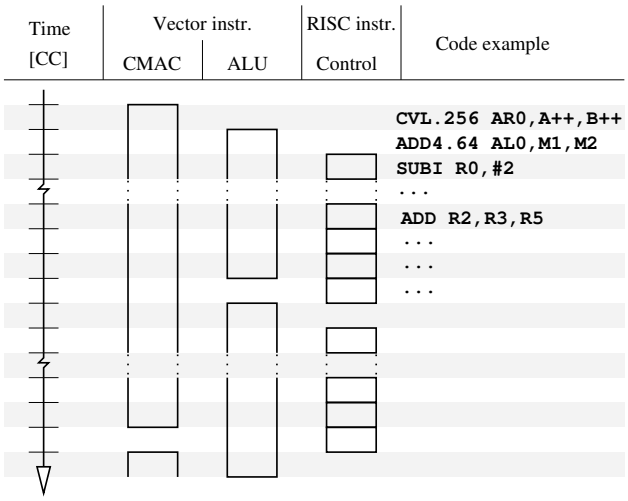

Figure 6.2: Instruction issue in a CSIMD processor

6.7 Memory system

To be able to use vector instructions it is important to have an efficient data movement mechanism to transport data between computing clusters. In the presented architecture an on-chip interconnect network is utilized which connects the execution units to the memory banks. The network consists of a crossbar switch which connects a port of a memory bank to a corresponding port on an execution unit [3]. No addresses are transfered over the network since the architecture relies on de-centralized addressing. This implies that every memory bank contains its own address generator units (AGU). The AGU supports a number of different addressing modes, such as modulo addressing, bit-reversed addressing and special addressing modes used in channel equalization (Rake finger addressing). To support the high memory bandwidth required by execution units such as a radix-4 operation running on a 4-way CMAC unit, each memory bank is divided into several parallel memories. To ensure that data are available according to the access pattern of transforms and similar operations, a special reordering crossbar is used to permutate the data

order between the network interface and the internal memory banks. The design of memory efficient multi-standard baseband processors is further discussed in Paper 3 [5].

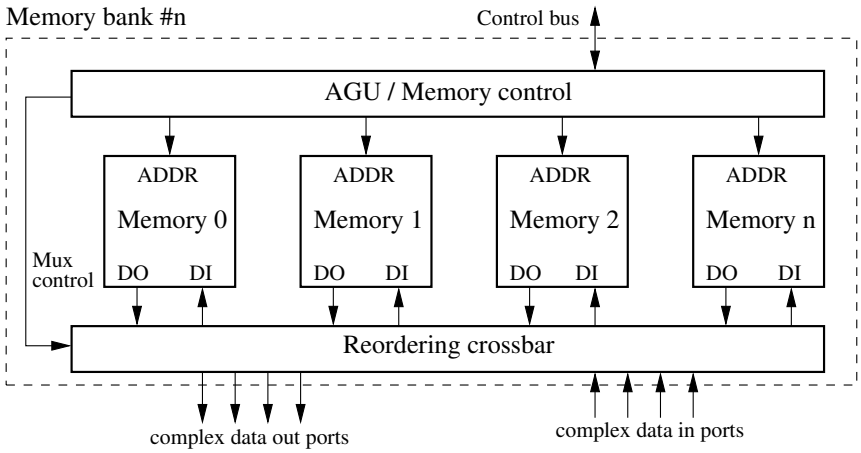

Figure 6.3: Overview of memory banks.

An overview of a memory bank is shown in Figure 6.3.

6.8 Scheduling

By statically schedule the complete processor, further simplifications can be made. This implies that the processor core statically schedules both the on-chip network and the memory crossbar. As a result of this, perfect predictability is achieved which allows the processor hardware to be optimized and simplified. Also, since the program and data flow is known in advance, memory resources can be allocated and memory bandwidths can be guaranteed thus making program and data cache memories unnecessary. Scheduling of OFDM reception tasks is examplified in Paper 3 [5].

References

[1] D. Liu; "Design of embedded DSP processors"; Department of Electrical Engineering, Linkoping university, To be published 2005.

[2] Anders Nilsson, Eric Tell and Dake Liu; "A fully programmable Rake-receiver architecture for multi-standard baseband processors", in *Proceedings of the intl. conference on Network and Communication Systems*, Krabi, Thailand 2005

[3] Eric Tell, Anders Nilsson and Dake Liu; "A Programmable DSP core for Baseband Processing" in *Proceedings of NEWCAS*, Quebec, Canada, June 2005

[4] Eric Tell, Anders Nilsson and Dake Liu; "A Low Area and Low Power Programmable Baseband Processor Architecture" in *Proceedings of IW-SOC*, Banff, Canada, July 2005

[5] Anders Nilsson, Eric Tell, Daniel Wiklund and Dake Liu; "Design methodology for memory-efficient multi-standard baseband processors"; submitted for review to Asia-Pacific Communications Conference, APCC'2005.

Chapter 7
Acceleration

7.1 Introduction

As described in earlier chapters, modern wide-band modulation schemes require tremendous computing capacity, ruling out regular implementations in general purpose processors. The key to further increased processing capacity and still maintained flexibility is to introduce acceleration in the processor. An accelerator is extra hardware added to a programmable processor, which in itself performs a certain class of pre-configured tasks while the processor could perform other operations. Acceleration could also be used to extend the functionality of single instructions of the processor. However, every extra accelerated function will increase the hardware cost, so selecting the right accelerators to cover most processing needs over the multiple standards is essential. Acceleration can be performed on many different levels and on different types of functions:

- Function level acceleration.
- Instruction level acceleration.
  - Complex instructions.
  - Addressing support, etc

7.1.1 Function level acceleration

This class of acceleration accelerates a complete function such as a Viterbi-decoder or a complete FFT transform. This kind of acceleration is usually implemented as a stand-alone *accelerator*. In this case, an accelerator is defined as an execution unit connected to the processor's internal communication network, which can perform processing tasks independently of the processor core.

7.1.2 Instruction level acceleration

Instruction level acceleration is the other form of acceleration employed by the baseband processor architecture. Instruction level acceleration implies that the processor is equipped with powerful special instructions, which accelerate a special task. This could for example be an FFT instruction, which performs one FFT butterfly operation in a single clock cycle. Other instruction level accelerators could be addressing support for different cycle consuming addressing modes such as modulo, bit-reversed and rake finger addressing.

7.2 Accelerator selection method

Since accelerators add extra hardware and complexity to the design, it is also important that the choice of what to accelerate is made right.

In order to make good design choices and achieve an optimal design we need a method of benchmarking the selected algorithms and a method of deciding what to accelerate. In order to compare the computational load of a certain algorithm, a "MIPS cost" is defined. The MIPS cost corresponds to how many million instructions-per-second a regular processor would require in order to perform the specified function. The MIPS cost is calculated as follows:

$$MIPS_i = \frac{OP_i \cdot N_i}{t_i}$$

where $MIPS_i$ is the associated MIPS cost, OP_i the number of clock cycles required by a standard DSP processor to perform the operation, N_i the number of samples or bits to process and t_i is the maximum time allowed for the operation to complete. For symbol related operations the time to perform the operation is considered to be the symbol duration. Here it is very important that the scheduling of the tasks is done correctly since the latency requirements are important to consider.

By analyzing all standards in term of processing operations and kernel functions, similar operations in different standards can be found. These operations are then used as acceleration candidates. When the MIPS cost has been calculated for all kernel functions; silicon usage must be estimated for all functions considered for acceleration.

Finally, when all kernel functions have been identified and their associated silicon area has been estimated the following three points must be considered:

1. **MIPS-cost.** A function with a very high MIPS cost must be accelerated since the operation cannot be performed by a regular processor.
2. **Reuse.** A function that is performed regularly and is used by several radio standards is a good candidate for acceleration.
3. **Circuit area.** Acceleration of special functions is only justified if there can be considerable reduction of clock frequency or power compared to the extra area added by the accelerator.

Any kernel function or operation fulfilling one or more of the previous points is a good candidate for hardware acceleration. Acceleration is further discussed in the Paper 1 [1].

7.3 Configurability and flexibility

Since it is desirable that the chosen accelerators can be used by as many standards as possible, flexibility and especially configurability of the accelerators is important. However, as an increased flexibility can increase

area and power consumption it is important to analyze different degrees of configurability of the unit. Examples of flexibility and configurability of accelerators are:

- Selectable decimation ratio in a digital front-end.
- Configurable filter coefficients in the front-end.
- Changeable polynominial in bit manipulation units based on LFSR structures.
- Interleavers with flexible interleaving depth.
- Map/de-map accelerators with up-dateable gain and constellations.

The same method to choose which functions to accelerate, based on function reuse and circuit area, can be used within the actual design to find the optimal balance between circuit performance and flexibility.

References

[1] A. Nilsson, E. Tell; *An accelerator structure for programmable multi-standard baseband processors.* Proc. of WNET2004, Banff, AB, Canada, July 2004.

Chapter 8
Channel equalization for CDMA systems

8.1 Introduction

In this chapter channel equalization for CDMA systems is described. Since many new systems such as the 3G networks are based on CDMA technology, it was natural to include CDMA processing functionality in a multi-standard processor design project. Usually channel equalization for CDMA systems is performed by using a rake receiver [1] structure. The rake receiver has been used for a long time in classical CDMA modems, however it has normally been implemented as fixed function hardware since it has been considered to be too computationally demanding for implementation in any other way.

However, as the convergence of mobile communication devices increases there is a huge need for more and more standard coverage in communication devices. This has been a problem for implementors of fixed function rake units since the implementation of certain key parts of the rake unit differs significantly from standard to standard even though the basic principle is the same. Currently there are four more or less connected 3G standards:

- WCDMA-FDD, Frequency division duplex WCDMA.

- WCDMA-TDD, Time division duplex WCDMA.
- WCDMA-HSDPA, High speed data packet access.
- TD-SCDMA, Chinese 3G

In Western Europe WCDMA-FDD is currently deployed and in operation. Nowadays the networks are being upgraded to support HSDPA in order to provide the users with data rates in the 2-10 Mbit/s range [2]. The design of a multi-standard rake receiver architecture is described in Paper 2 [3].

8.2 Rake receivers

A rake receiver is a time-domain equalizer which in principle identifies the strongest multi-path components and align them constructively. Internally, the rake receiver contains two different parts:

- A channel estimator.
- Channel compensation units.

The channel estimator identifies both the delay, phase and the amplitude of a limited number of multi-path components. The delay, phase and amplitude of the multi-paths forms the *channel estimate*. The channel estimate is used in the channel compensation stage. Each of the multi-path components is assigned to a channel compensation unit, noted a *rake finger*. If the rake receiver consists of four rake fingers, only the four strongest multi-paths are used in the process. A schematic view of a rake receiver is presented in Figure 8.1,

In most papers regarding flexible rake receivers, such as the classical FlexRake [1], only how to implement the channel compensation part is discussed. However, benchmarking and resource allocations [3] show that in reality, channel estimation is more computationally demanding than channel compensation.

The use of so called *multi-code transmission* [2] and soft hand-over complicates the channel estimation and compensation.

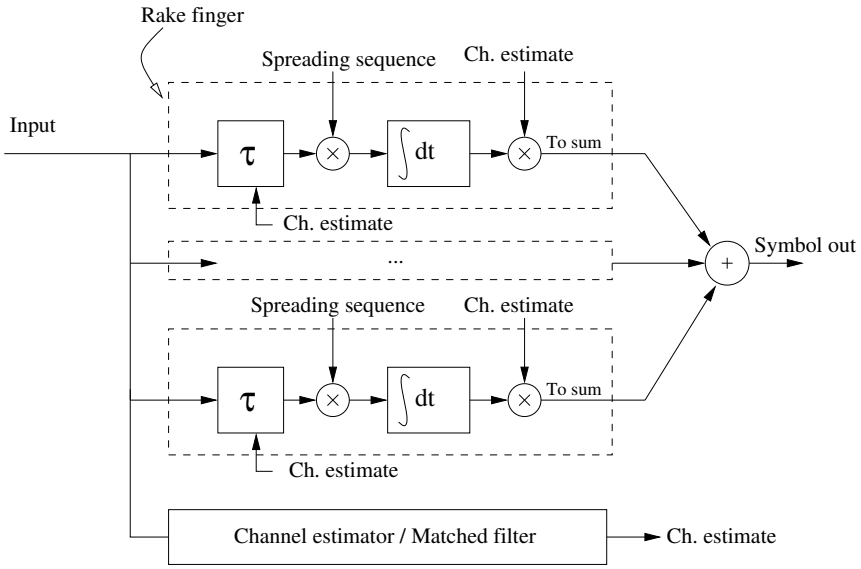

Figure 8.1: Rake receiver.

8.3 Multi-code transmission

To further improve the data-rate to a user in a CDMA network, the user can be assigned several spreading codes. This will allow the user to receive each of the codes in parallel. Since the de-spread operation is included in the classical rake finger, one rake finger is needed for each multi-code used. If the number of rake fingers is limited, the system must trade-off channel compensation capability and data-rate. Multi-code transmission is necessary to achieve the higher data-rates used in the WCDMA and HSDPA systems.

8.4 Soft handover

Since CDMA networks can be built as single-frequency networks, handover between base-stations can be done by *soft handover*. Soft handover can be explained as: when a user is on the border between two base-

stations, the handover is performed gradually by transmitting data from both base-stations instead of switching base-stations instantly. This is possible due to multi-code transmission; a further development is *softer handover* where the mobile terminal is handed over between sectors in the same base-station. Then the same code-set is used so that the user cannot distinguish the new transmitter from a strong echo. The 3GPP WCDMA standards require mobile stations to handle up to 6 simultaneous base-stations during soft handover. A soft handover scenario with 3 base-stations and three multi-codes is illustrated in Figure 8.2. Here, the same code-set is not reused during the handover phase. In this example, at least nine rake fingers are necessary in the mobile terminal in order to de-spread the data.

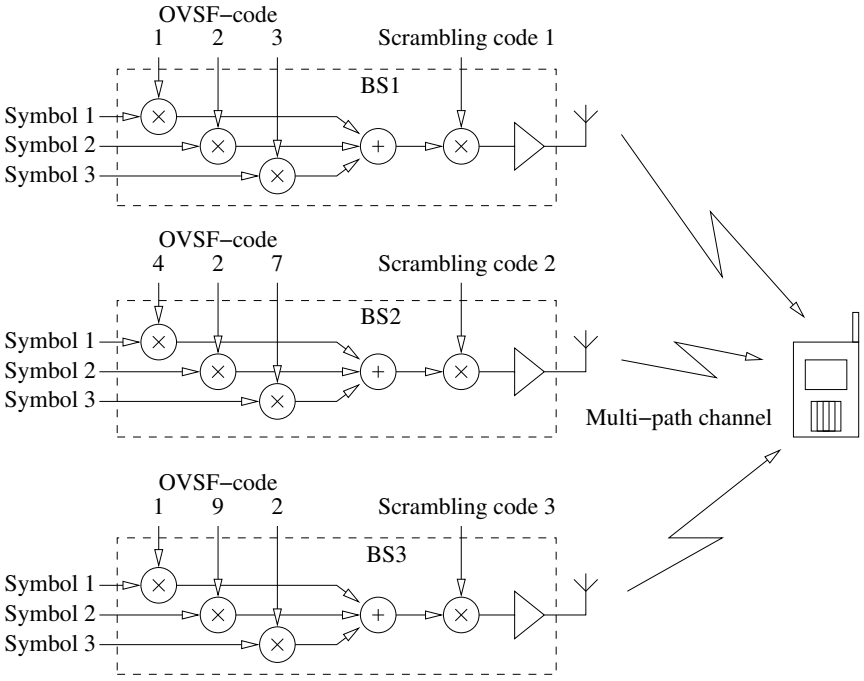

Figure 8.2: Soft handover with three base-stations and three multi-codes

8.5 Programmability

By providing a programmability to a rake receiver, computation resources can easily be redistributed dynamically among receiver tasks and allow the processor to trade multi-code capability with soft handover capability in real-time. Paper 2 [3] is devoted to the design of such processor architectures.

8.6 WLAN

The same principle of channel compensation for 3G networks can also be used on CDMA based WLAN systems such as IEEE802.11b [4]. The main difference between 3G systems and CDMA based WLANs is the higher chip-rate of WLAN systems (11 Mcps versus 3.84 Mcps for 3G). However, since the mobility requirement of WLAN systems is much less than in 3G systems, the computational load is roughly the same. To fully understand all aspects of rake receivers in practice, a test system was built and real IEEE802.11b data were recorded in the baseband domain. The data were used as input stimuli to the rake receiver models to further verify the accuracy of the used models. The results of a measured channel estimate for IEEE 802.11b is presented in Figure 8.3. Each sample corresponds to $45\mu s$ or 13.6 meters.

References

[1] Lasse Harju, Mika Kuulusa and Jari Nurmi, "Flexible implementaion of a WCDMA Rake reciver", in *Proceedings of the IEEE Workshop on Signal Processing Systems (SIPS)*, Oct. 2002

[2] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Physical channels and mapping of transport channels onto physical channels (FDD) 3GPP TS 25.211 V6.0.0

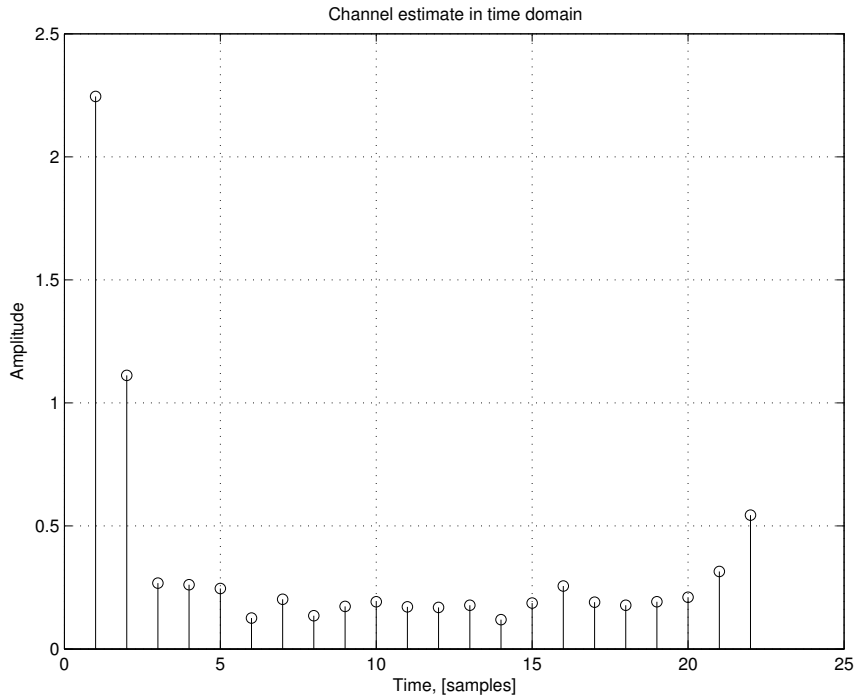

Figure 8.3: Measured channel impulse response in an office environment.

[3] "A fully programmable Rake-receiver architecture for multi-standard baseband processors", in *Proceedings of the intl. conference on Network and Communication Systems*, Krabi, Thailand 2005

[4] IEEE 802.11b, Wireless LAN MAC and PHY specifications: Higher-Speed Physical Layer Extension in the 2.4 GHz Band, 1999.

[5] Holma, Harri, "Wcdma For Umts: Radio Access for Third Generation Mobile Communications", John Wiley And Sons Ltd, UK, 2004 ISBN 0470870966

Chapter 9
Research methodology

This chapter briefly describes the research methodology used in this project. The research methodology can be summarized by the following steps:

1. Study and analysis of general baseband processing and the associated technology.
2. Survey of other research projects and related work within the same field.
3. Formalization of project goals and scope.
4. Formulation of a design and evaluation methodology for programmable baseband processors.
5. Developement and refinement of a processor architecture according to the previously mentioned methodology.
6. Implementation of the created processor.
7. Evaluation.

All details of how to model, design and evaluate baseband processors are presented in Chapter 5. As with any research project, many iterations are performed during the research work. However, it is important to remember the research methodology in order to ensure a high quality

of the research results and findings. During the initial phases of a research project, both the scope and goal of the project will most certainly change. However, at some time early in the project a decision must be made, clearly defining what to include and what to leave out. This is essential to keep the project going, since there is no limit on how much time a researcher can spend on an interesting problem.

Chapter 10
Achievements and future work

10.1 Introduction

The main achievements in this research project can be divided into five different topics all aiming at giving a solid background for designing the next generation of baseband processors:

- Specification and design of accelerators.
- Design of execution units and memory systems for rake processing.
- Algorithm selection.
- Specification and implementation of simulation environment.
- Instruction issue and memory architectures.

10.2 Achievements

As stated earlier, the goal of this research project was to create design methods and hardware for power and area efficient, low clock rate multi-standard baseband processors. All achievements presented below are the result of research with this high level goal.

10.2.1 Accelerators

A large part of the research effort has been spent on accelerator selection and design. In this area I have formulated guidelines for algorithm and accelerator selection and have specified, designed and implemented multi-standard accelerators for IEEE802.11a/b/g and WCDMA systems. Special care has been taken to ensure multi-standard support and flexibility in the designed and implemented accelerators. A number of accelerators have been designed and implemented in the BBP1 processor [1].

10.2.2 Digital front-end

Effort has also been put on specification, design and implementation of multi-standard digital front-ends used in the baseband processor for filtering / symbol shaping and decimation/interpolation purposes.

Multi-standard digital front-ends are necessary to off-load the previous mentioned tasks from the processor, however it is most important that the front-end fulfills all requirements imposed by all modes in all conceivable standards. Since the filter requirements vary from system to system, careful design of the front-end is necessary to ensure support for a wide range of filter specifications with very low area.

A versatile multi-standard front-end has been designed and implemented as part of the BBP1 processor. The front-end includes digital filters, a decimator and a frequency error compensator unit. During the design of the digital front-end research and modeling of filter requirements, required precision, scheduling and energy efficiency have been performed. In addition to the digital filter and the decimator, a frequency error compensator has been designed and implemented in the same unit.

10.2.3 Rake channel equalization

Another main part of the project was to understand and create a processing architecture capable of performing channel equalization of WCDMA and similar CDMA based systems in software. During this part of the

project all steps from high level simulation and system specification to RTL coding of several execution units have been taken. Examples of designs are a small but versatile short-complex-MAC unit and a memory scheme, which will allow a programmable baseband processor to run the Rake receiver algorithms smoothly. Moreover, both system simulation models and bit-true models have been created for WCDMA transmission, reception and channel compensation.

10.2.4 Algorithm selection and development

Along the other activities in the project, much research effort has been spent on gaining an understanding of how to select algorithms suitable both for a certain type of problem, but also suitable for a certain architecture. Here the result of this algorithm research has been incorporated in the design and implementation of all hardware and all architectures presented in this thesis. Algorithms have been studied, selected, designed and implemented for both OFDM and CDMA transmission, reception and channel compensation.

10.2.5 Models

To be able to create correct and realistic models of baseband processing systems and transmission channels, much effort has been spent on understanding both the properties of the radio front-end and the channel, but also the requirements imposed on the system by the MAC layer. Here I have created high-level models as well as bit- and cycle-true models, which embody all relevant impairments and constraints imposed by the system environment. These models have been used as a foundation for the design of the processor architecture. Furthermore, the results of the algorithm related research have been incorporated in all models.

10.2.6 Instruction issue

A fundamental property of the baseband processor architecture is the single-issue concept that has been refined since the first BBP generation. The concept has been extended and transformed into a new architecture with many clustered SIMD execution units. Together with this research scheduling methods have also been created.

10.3 Future work

All the research presented in this thesis constitutes a solid background for designing a new generation of programmable baseband processors. The next step is to start the implementation phase of the BBP2 baseband processor chip based on the knowledge accumulated during the first part of the research project.

From the implementation and tape-out of the BBP2 chip, valuable information regarding implementation aspects can be learned and used to improve both the architecture itself and the design methodology. The ultimate goal of the baseband processor research is to create a flexible, area efficient and low-power baseband processor solution capable of handling all standards from single-carrier systems to OFDM, CDMA and MIMO systems.

Another important part of the research project is to do further quantitative comparisons and analyses of performance and silicon efficiency of other design-choices and possible architectures to back design decisions made during the research project.

References

[1] Eric Tell, Anders Nilsson and Dake Liu; "A Low Area and Low Power Programmable Baseband Processor Architecture" in *Proceedings of IW-SOC*, Banff, Canada, July 2005

Part III

Papers

Chapter 11
Introduction

In this part of the thesis, a selection of my research results and papers are presented. The three papers presented are the result of background investigations performed in the design of a new baseband processor architecture designed within the research project.

Each of the papers covers a part of the research used as foundation for this new design. The papers cover:

- Overall processor architecture.
- Selection of accelerators.
- CDMA based channel equalization.
- An efficient OFDM processor architecture and scheduling.

The new processor will be used as a demonstrator for low power, true multi-standard baseband processing. In Paper 1, an accelerator architecture for multi-standard baseband processors is presented, later, in Paper 2, a multi-standard Rake-receiver architecture is presented and lastly in Paper 3, a design methodology for memory-efficient multi-standard baseband processors is presented.

Chapter 12
Paper 1

An accelerator architecture for programmable multi-standard baseband processors

Anders Nilsson, Eric Tell and Dake Liu
Department of Electrical Engineering
Linkoping University
Linkoping, Sweden

ABSTRACT
Programmability will be increasingly important in future multi-standard radio systems. We are proposing an architecture for fully programmable baseband processing, based on a programmable DSP processor and a number of configurable accelerators which communicate via a configurable network. Acceleration of common cycle-consuming DSP jobs is necessary in order to manage wide-band modulation schemes. In this paper we investigate which jobs are suitable for acceleration in a programmable baseband procsessor supporting a number of common Wireless LAN and 3G standards. Simulations show that with the proposed set of accelerators, our architecture can support the discussed standards, including IEEE 802.11a 54 Mbit/s wireless LAN reception, at a clock frequency not exceeding 120 MHz.

KEY WORDS
CDMA, OFDM, DSP, SDR

12.1 Introduction

Programmable baseband processors are necessary to support multiple radio standards, since a pure ASIC solution will not be flexible enough. ASIC solutions for multi-standard baseband processors are less area efficient than their programmable counterparts since processing resources cannot be shared between different operations.

In this paper we present an approach to combining baseband processing of multiple radio standards into an area efficient and versatile baseband processor. The key to increase processing capacity and still maintain flexibility is to introduce *accelerators* in the processor. An accelerator is extra hardware added to a programmable processor which performs a certain pre-configured task while the processor is free to perform other operations. However, every extra accelerated function will increase the hardware cost, so selecting the right accelerators to cover most processing needs over multiple standards is essential. In this paper we identify

key accelerators and present an architecture for a programmable processor which is aimed at the following radio standards:

Table 1.

Standard	Modulation	Type
WCDMA	CDMA	3G
TD-SCDMA	CDMA	3G
IEEE 802.11b	DSSS/CCK	Wireless LAN
IEEE 802.11a	OFDM	Wireless LAN

We prove our accelerator concept by using the most demanding algorithm in every accelerator. By using 54 Mbps IEEE 802.11a [1] and 3.84 Mchip/s WCDMA [1] in our calculations we ensure architectural support for less demanding communication standards.

The paper is organized as follows. In section 13.2, we survey four different communication standards and provide an overview of the operations associated with each standard. In section 12.3, we discuss the proposed accelerators. In section 12.4 an accelerator interconnect network proposal is presented. In section 14.7 our results are presented. Finally conclusions are drawn in section 14.8.

12.2 Survey of communication standards

To be able to decide which functions to accelerate we survey the different communication standards listed in Table 1. The communication standards have been analyzed focusing on sample rates, required functionality of a baseband processor, and cycle cost/latency requirements. We have chosen to restrict ourselves to only focus on reception in this paper, since the receiver is more computation demanding than the transmitter where all data are known in advance. The maximum data rate of each standard has been used in our calculations. The analysis is divided into several processing tasks:

- Radio front-end processing.

- Symbol processing.
- Data recovery.
- Forward error correction and channel coding.

In order to compare the computational load of a certain algorithm, we define "MIPS cost" which corresponds to how many million instructions-per-second a regular processor would require in order to perform the specified function. The MIPS "cost" is calculated as follows:

$$cost_i = \frac{OP_i \cdot N_i}{t_i}$$

where $cost_i$ is the associated MIPS cost, OP_i the number of clock cycles required by a standard DSP processor to perform the operation, N_i the number of samples or bits to process and t_i is the maximum time allowed for the operation to complete. For symbol related operations the time to perform the operation is considered to be one symbol time. Latency requirements imposed by the standards [1],[7],[1],[7] has also been taken into account when calculating the required MIPS cost.

12.2.1 Radio front-end processing

The first task for a baseband processor is to filter and decimate the input signal, in order to reduce interference from adjacent channels and to relax requirements on the analog-to-digital converter.

The occupied bandwidth, the over-sampling rate (OSR) and the required sampling rate are presented in the following table:

Standard	Bandwidth	OSR	Sample rate
IEEE 802.11a	20 MHz	2	40 MHz
IEEE 802.11b	11 MHz	2	22 MHz
W-CDMA	3.84 (5) MHz	4	15.36 MHz
TD-SCDMA	3.84 (5) MHz	4	15.36 MHz

As stated in the table above, the maximum sample rate the processor needs to process is 40 MHz.

As decimation filter, we use a raised-cosine FIR filter, which maintains the phase of the received signal. Raised-cosine filters have a symmetrical impulse response with N/2-1 zero taps for a filter length of N.[13] This property will reduce the computation load significantly since the processor does not need to calculate those taps.

The number of taps in the FIR filter is determined by the smallest roll-off factor. The roll-off factor determines the transition band of the filter. A lower roll-off factor yields better filter performance since the transition band is small, however the length of the filter increases accordingly. IEEE 802.11a [1] requires a roll-off factor of $r = 3/64 = 0.0468$. Well known FIR filter design formulae [13] give the required number of taps from the roll-off factor. An $r = 3/64$ yields a filter of length 21 taps.

For comparison, we have asserted a cycle cost of 30 operations per processed input sample. This cycle cost also includes control flow instructions. Normally all 21 complex filter taps are processed. However since at most every second output sample is used, savings can be made. 30 operations per sample is a moderate estimation for such a large filter. The resulting required MIPS cost is:

Standard	Required MIPS
IEEE 802.11a	1200
IEEE 802.11b	440
W-CDMA	600
TD-SCDMA	600

12.2.2 Symbol processing

Since synchronization and channel estimation schemes in all four communication standards are diverse, algorithms and functions cannot easily be shared between the different standards. Initial channel estimation and synchronization cost has been estimated to:

Standard	Required MIPS
IEEE 802.11a	108
IEEE 802.11b	12
W-CDMA	25
TD-SCDMA	25

In Direct Sequence Spread Spectrum (DSSS) and CDMA systems, channel estimation is performed by using a matched filter (correlator) to estimate the channel impulse response. The channel estimate is only calculated for as many points as RAKE (see below) fingers used.

Since synchronization and channel estimation is only performed at the beginning of IEEE 802.11a packets and the processing power requirement is acceptable, the synchronization tasks are run completely in software. However for CDMA and DSSS modulation schemes, the matched filter correlator is run continuously. The associated MIPS cost of continuous channel estimation is included in the MIPS cost for the RAKE unit.

For DSSS and CDMA systems, channel compensation and de-spread is performed by a RAKE receiver [3]. The RAKE unit has a certain number of taps (often referred to as "fingers"), which correspond to taps in the channel impulse response. By using four "fingers", up to 90% of the received signal energy can be used to recreate the transmitted signal in an office environment.[4] The maximum delay in the delay element τ is 127 chips to accommodate a delay of 33 μs at 3.84 Mchip/s. A Rake finger is shown in Figure 13.2.

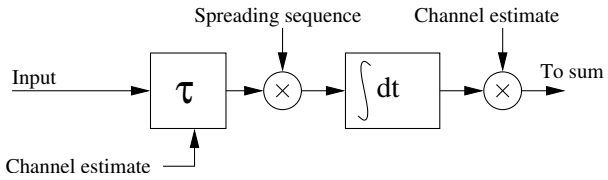

Figure 12.1: Rake finger.

However in OFDM systems, the symbols are recreated in the frequency

domain. The conversion from time domain to frequency domain is performed by an FFT. The cost of a 64 point Radix-4 FFT is approximately 430 clock cycles in a regular processor. The associated processing cost for reconstructing the transmitted symbols is presented in the following table:

Standard	Type	Req. MIPS
IEEE 802.11a	FFT	108
IEEE 802.11b	RAKE	550
W-CDMA	RAKE	384
TD-SCDMA	RAKE	384

12.2.3 Data recovery

When the data-symbols have been recreated the de-mapper extracts binary information from the received symbol. De-mapping is tedious work. For each 64 QAM symbol (6 bits worth of data in IEEE 802.11a) 6 comparisons and 6 loads must be made. This yields a total of two operations per received bit. In higher data rate modes of IEEE 802.11b the data are partly processed by using a Modified Walsh Transform. (MWT) [7] The computation of a MWT is similar to a FFT. The cost of de-mapping and data recovery is summarized in the following table:

Standard	Required MIPS
IEEE 802.11a	108 MHz
IEEE 802.11b	160 MHz
W-CDMA	8 MHz
TD-SCDMA	8 MHz

12.2.4 Forward error correction

Forward error correction and channel coding are the most computation demanding functions performed in a baseband processor and are necessary in order to improve data rate over noisy channels.

Several different encoding schemes are often combined to further reduce the bit error rate (BER).

- **Interleaving.** Data are reordered to spread neighboring data bits in time and in the OFDM case among different frequencies.

- **Convolutional codes.** This class of error correcting codes includes regular convolutional codes as well as turbo codes. Turbo codes and regular convolutional codes is very similar. Decoding of convolutional codes is performed by the Viterbi algorithm [11], whereas Turbo codes are decoded by utilizing the Soft output Viterbi algorithm. [12].

- **Scrambling.** In several standards data are scrambled with pseudo-random data, to ensure an even distribution of ones and zeros in the transmitted data-stream.

Since all these schemes operate on bits, and since bit operations and irregular execution is very inefficient in a signal processor, the required MIPS cost is very high. Interleaving for IEEE 802.11a costs 5 op/bit whereas interleaving for WCDMA and TD-SCDMA costs 32 op/bit. The costs of interleaving for the different standards are:

Standard	Required MIPS
IEEE 802.11a	270
IEEE 802.11b	–
W-CDMA	122
TD-SCDMA	122

Viterbi and Turbo codes are considered large research areas by themselves. We acknowledge the MIPS cost for Viterbi and Turbo decoding according to the following table.[11][12]

Standard	Required MIPS
IEEE 802.11a	4000
IEEE 802.11b	–
W-CDMA	1964
TD-SCDMA	1964

Scrambling is also a very cycle consuming task for a programmable DSP since it requires many bit-operations. We have asserted 3 op/bit for scrambling in IEEE 802.11a/b. This yields the following MIPS costs:

Standard	Required MIPS
IEEE 802.11a	162
IEEE 802.11b	33
W-CDMA	–
TD-SCDMA	–

TD-SCDMA uses Reed-Solomon codes as extra data protection in packet transfer mode. An estimated MIPS cost of 20 MIPS is required for Reed-Solomon (RS) decoding. However, since RS is only used in TD-SCDMA and the cost is low, it will not be considered for acceleration.

12.3 Proposed accelerators

In Figure 12.2 a summary of all MIPS costs are presented. The method of selecting functionality to accelerate must consider:

1. **MIPS cost.** A function with a very high MIPS cost must be accelerated since the operation cannot be performed by a regular processor.
2. **Reuse.** A function that is performed regularly and is used by several radio standards is a good candidate for acceleration.
3. **Circuit area.** Acceleration of special functions is only justified if there can be considerable reduction of clock frequency or power compared to the extra area added by the accelerator.

	IEEE 802.11a	IEEE 802.11b	WCDMA	TD–SCDMA
Filter and decimation	1200	440	600	600
Channel estimation	108	12	25	25
RAKE	—	550	384	384
FFT/MWT	108	160	—	—
Tracking	12	—	—	—
Demap	108	22	8	8
Interleaver	270	—	122	122
Viterbi/ Turbo	4000	—	1964	1964
Scrambler	162	33	—	—
CRC	—	0.2	0.2	0.2
Reed–solomon	—	—	20	—

Figure 12.2: Breakdown of operations and MIPS cost for various standards.

We propose acceleration of operations circled together in Figure 12.2. As shown in the figure operations common to most standards are grouped together. By accelerating the selected functions we ensure support for the communication standards listed in the figure. The accelerators we suggest are:

- A configurable decimator and filter.
- A four "finger" RAKE accelerator for use in CDMA and DSSS modulation schemes.
- A Radix-4 FFT/Modified Walsh transform accelerator. This accelerator is used in OFDM modulation schemes and in IEEE 802.11b

which uses MWT.

- A Turbo/Viterbi decoder.
- A configurable block interleaver.
- A configurable scrambler.

A brief description of the properties required by the proposed accelerator follows:

12.3.1 Decimator

As described earlier IEEE 802.11a requires a FIR filter with a roll-off of ~ 0.05. This implies a FIR filter length of 21 taps. By allowing the filter to be reconfigured, decimation and filtering can be accommodated for all other communication standards covered by this processor. Since the required MIPS for filtering and decimation is very high and the operations are performed on every input sample this function is accelerated.

12.3.2 RAKE unit

By using four rake fingers up to 90% [3] of the received signal energy can be used to recreate the signal in an pedestrian/office environment. We propose a four finger rake accelerator utilizing an accumulator unit and a simple complex multiplier capable of multiplying samples with $\pm 1 \pm i$. The accelerator also contains 512 words of complex memory for delay path storage, de-spread code generators and a matched filter which performs multipath search and channel estimation. Since the input signal is oversampled 4 times, a fractional delay stage is not necessary to provide sub-chip resolution. An overview of the RAKE unit is presented in figure 12.3.
A more detailed description of a flexible RAKE unit is presented in [10].

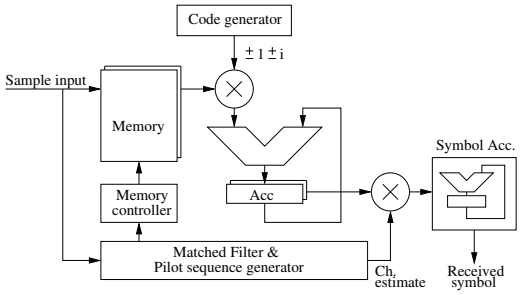

Figure 12.3: RAKE Accelerator

12.3.3 Radix-4 FFT/MWT

By using a Radix-4 butterfly and flexible address generators a configurable FFT/Walsh transform accelerator can be built. The structure of such an accelerator is described in [7]. The structure can perform a 64 point FFT in 54 clock cycles and a modified walsh transform for IEEE 802.11b in 18 clock cycles.

12.3.4 Viterbi/Turbo decoder

The Viterbi/Turbo decoder is the most demanding block to implement without acceleration. Since Viterbi and Turbo decoders are frequently used, several implementations exist in the research community.

In [12] a reconfigurable 54 Mbps Viterbi decoder and a 2 Mbps Turbo decoder are presented. The required clock frequency is 60.5 MHz for 2 Mbps Turbo decoding.

12.3.5 Interleaver

In [3] a multi-mode block interleaver accelerator for IEEE 802.11a has been implemented. Following results have been achieved: Interleaving of 288 bits of data consume 34 clock cycles. The estimated area for the complete block interleaver including memories is 0.0270 mm^2 in a 0.18μm process. Since block interleaving in TD-SCDMA and WCDMA is similar, the same

structure with a memory block that is written column-wise and read row-wise can be used. Convolutional interleaving for WCDMA/TD-SCDMA is performed by using flexible address generators and regular memory.

12.4 Network

The network is a kind of bus connecting accelerators to the baseband core and to each other. The accelerator network is essential in the baseband processor. We are using a passive network since the network configuration is static during operation. The actual network consists of two sub-networks, one used for sample based transfers and a serial network for bit based transfers. The division of the two networks is essential to improve the throughput of the networks since bit based transfers require tedious framing and de-framing of data chunks not equal to the data width of the network.

Each sub-network consists of a crossbar switch which is configured from the processor. The switch allows related accelerators to be connected with each other and system memories. This crossbar approach enables the data to flow seamlessly between accelerator units without the intervention of the DSP core. The processor is only involved during creation and destruction of network connections.

By allowing several accelerators to communicate with each other, several accelerated functions can run in parallel and thus lower the computation time further. The functionality of the network is illustrated in Figure 12.5.

12.5 Results

The main focus of this paper has been to identify accelerators needed for a programmable baseband processor capable of receiving WCDMA, TD-SCDMA and IEEE 802.11a/b. The different communication standards have been analyzed in terms of required functionality of the baseband

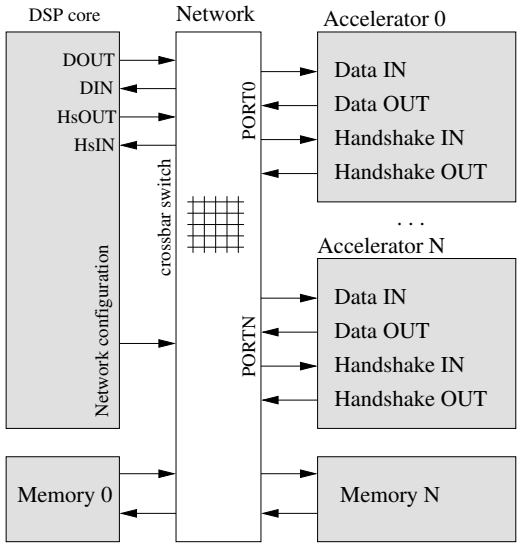

Figure 12.4: Accelerator network.

processor, cycle cost and the added area of acceleration. We have identified the following accelerators:

- Filer and decimator.
- RAKE accelerator.
- FFT/MWT accelerator.
- Demapper.
- Viterbi/Turbo decoder.
- Interleaver.
- Scrambler.

The proposed accelerators provide functional coverage for the most demanding job in a wide range of radio standards. Combined with the flexibility of a programmable DSP core, this architecture is well equipped for future standard updates.

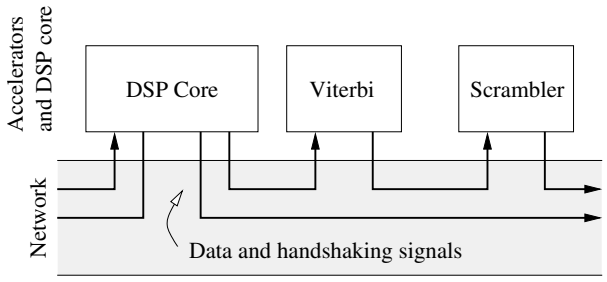

Figure 12.5: Example of logical network connections.

A complete programmable baseband processor architecture is presented in Figure 12.6. Simulation of a processor with an accelerator network as shown in Figure 12.4 has shown that the processor can be run at a clock frequency not exceeding 120 MHz while receiving 54 Mbit/s IEEE 802.11a data. The clock frequency constraint is derived from latency requirements found in IEEE 802.11a. Since the memories are connected to the configurable network, memories can easily be "moved" between different accelerators and the core, without the need of costly memory move operations by the processor.

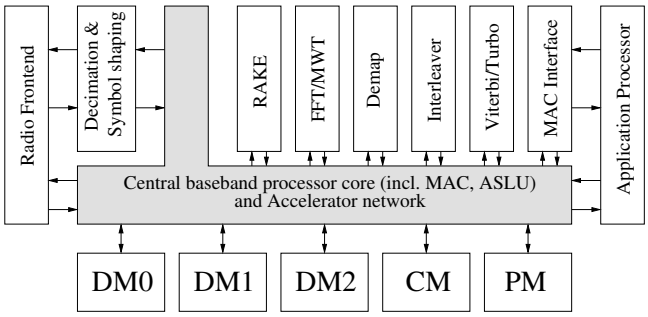

Figure 12.6: Top architecture of the programmable baseband processor.

To estimate the area used by accelerators, we have synthesized all but the Viterbi/Turbo decoder using Synopsys Design Compiler for a UMC

0.18 μm library. The synthesis result is presented in the following table:

Operation	Without acceleration MIPS	Req. acc. frequency MHz	Area mm^2
Decimator	1200	40	0.17
RAKE	384	16	0.190
FFT/MWT	160	13.5	0.28
Demapper	108	12	0.016
Viterbi	4000	60	1.1
Interleaver	270	8.5	0.047
Scrambler	162	54	0.011
Network	-	-	0.126

All MIPS requirements except the RAKE and FFT unit correspond to IEEE 802.11a. The maximum MIPS requirement on the RAKE unit originate from WCDMA, and the requirement on the FFT/MWT unit originate from IEEE 802.11b. The area of the Viterbi/Turbo decoder is estimated from gate count given in [12].

A complete accelerator architecture except Viterbi/Turbo decoder but including the accelerator network occupies 0.84 mm^2 in a 0.18 μm process. The total area including core, accelerators, network and memories occupies approximately 2.5 mm^2.

12.6 Conclusion

Programmability is essential for multi-standard baseband processors. In order to be able to process high bandwidth communication standards in a programmable processor, acceleration is necessary. As a response to this, we have presented an accelerator architecture for programmable baseband processors targeted at software defined radio applications. We have also presented a selection of functions to accelerate for a CDMA/OFDM

baseband processor. Our architecture is versatile and area efficient since we ensure maximum utilization of each accelerator.

[1] IEEE 802.11a, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications High-speed Physical Layer in the 5 GHz Band, 1999.

[2] IEEE 802.11b, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Higher-Speed Physical Layer Extension in the 2.4 GHz Band, 1999.

[3] Eric Tell, Dake Liu; A Hardware Architecture for a Multi Mode Block Interleaver; Submitted to *International Conference on Circuits and Systems for Communications (ICCSC)*, Moscow, Russia, June 2004

[4] Eric Tell, Olle Seger och Dake Liu; A Converged Hardware Solution for FFT, DCT and Walsh Transform; Proc. of the *International Symposium on Signal Processing and its Applications (ISSPA)*, Paris, France, Vol. I, pp. 609 - 612, July 2003

[5] H Heiskala, J T Terry, *OFDM Wireless LANs: A Theoretical and practical guide*, Sams Publishing, 2002

[6] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Physical channels and mapping of transport channels onto physical channels (FDD) 3GPP TS 25.211 V3.5.0

[7] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Spreading and modulation (TDD) 3GPP TS 25.223 V5.3.0

[8] 3rd Generation Partnership Project; User equipment radio reception and transmission; 3GPP TS 25.102 V6.0.0

[9] H. Holma and A. Toskala; *WCDMA for UMTS Radio Access For Third Generation Mobile Communications*, John Wiley and Sons, Inc., 2001.

[10] L. Harju, M. Kuulusa, J. Nurmi; Flexible implementation of WCDMA RAKE receiver; *Signal Processing Systems, 2002. (SIPS '02)*. IEEE Workshop on , 16-18 Oct. 2002 Pages:177 - 182

[11] S. Lin, D. J. Costello Jr.; *Error control coding, Fundamentals and Applications*; Prentice Hall.,1983

[12] Cavallaro, J.R.; Vaya, M.;Viturbo: a reconfigurable architecture for Viterbi and turbo decoding; Proceedings of *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003*. (ICASSP '03). Volume: 2, 6-10 April 2003 Pages:II - 497-500 vol.2

[13] L. Wanhammar, H. Johansson; *Digital filters*; Dept. of EE, Linkoping University, Linkoping, Sweden, 2001

Chapter 13
Paper 2

A fully programmable Rake-receiver architecture for multi-standard baseband processors

Anders Nilsson, Eric Tell and Dake Liu
Department of Electrical Engineering
Linkoping University
Linkoping, Sweden

ABSTRACT

Programmability will be increasingly important in future multi-standard radio systems. We are presenting a fully programmable and flexible DSP platform capable of efficiently performing channel estimation and Maximum Ratio Combining (MRC) based channel equalization for a large number of wireless transmission systems in software. Our processor is based on a programmable DSP processor with SIMD-computing clusters. We also map Rake receiver kernel functions supporting a large number of common Wireless LAN and 3G standards to this micro-architecture. The use of the inherit flexibility for future standards is also discussed. Benchmarking show that with the proposed instruction set architecture, our architecture can support channel estimation, equalization and decoding of: WCDMA (FDD/TDD-modes), TD-SCDMA and the higher data rates of IEEE 802.11b (CCK) at clock frequency not exceeding 76 MHz.

KEY WORDS

CDMA, Rake, MRC, DSP, SDR

13.1 Introduction

The ever changing wireless network industry requires flexible and versatile baseband processors to be able to quickly adapt to new and updated standards. This requires programmable baseband processors in order to support multiple radio standards, since a pure ASIC solution will not be flexible enough. ASIC solutions for multi-standard baseband processors are also less area efficient than their programmable counterparts since processing resources cannot be efficiently shared between different operations.

In this paper we present a micro-architecture which combines several SIMD computing *clusters* into a processor core. (Clustered SIMD) Unlike other flexible Rake architectures [2], this architecture will run both the multi-path search, Rake fingers and Maximum Ratio Combining (MRC) in software. A clustered SIMD machine is defined as a processor core

with several heterogeneous SIMD-data path clusters (i.e. a 4-way ALU and a separate 4-way CMAC). In a clustered SIMD micro-architecture, only one instruction is issued every clock cycle. Unlike VLIW-machines, this micro-architecture will allow concurrent vector operations and control flow instructions without the drawbacks and memory usage of a pure VLIW machine. The combination of SIMD units and an efficient memory architecture allows us to drastically improve the processing parallelism without adding extra complexity of VLIW machines. We have chosen to map channel estimation and correction of the following communication standards to our architecture:

Standard	**Modulation**	**Type**
WCDMA	CDMA/FDD	3G
WCDMA-TDD	CDMA/TDD	3G
TD-SCDMA	CDMA/TDD	3G
IEEE 802.11b	DSSS/CCK	Wireless LAN

Our processor architecture is proven by mapping the most demanding algorithms to the architecture. We both include Frequency Division Duplex (FDD) and Time Division Duplex (TDD) CDMA-modes[1][4] in order to fully prove the architecture. Time Division Duplex mode is the most demanding mode since it requires very short computing latency on the channel estimation compared to the FDD mode where a pilot channel is transmitted simultaneously with the data channel. By proving the TDD mode, we can ensure architectural support for the less demanding modes.

The paper is organized as follows. In section 13.2, we survey four different communication standards and provide an overview Rake based channel equalization. In section 13.3, we discuss the proposed architecture. Then in section 13.4 SIMD clusters are discussed. In section 13.7 scheduling is discussed. In section 14.7 our results are presented. Finally conclusions are drawn in section 14.8.

13.2 Background

13.2.1 Survey of communication standards

The chip rate, the over-sampling rate (OSR) and the required sampling rate for the investigated standards are presented in the following table:

Standard	Chip rate	OSR	Sample rate
IEEE 802.11b-DSSS	11 Mcps	2	22 MHz
IEEE 802.11b-CCK	11 Mcps	2	22 MHz
WCDMA-FDD/TDD	3.84 Mcps	4	15.36 MHz
TD-SCDMA	1.28 Mcps	4	5.12 MHz

By using an OSR of 4 for WCDMA/TD-SCDMA, the multi-path searcher will provide sub-chip resolution without fractional delay filters.

13.2.2 Rake based channel equalization

Rake receivers are often used in CDMA systems to cancel inter-symbol interference resulting from multi-path propagation and to utilize the multi-path diversity.

The idea of a Rake receiver is to identify a number of different multi-path components and align them constructively, both in time and phase, thus utilizing the created multi-path diversity. The combination of the different components is performed by Maximum-Ratio Combining (MRC).

The function of delaying a certain transmission path and correct its phase is often referred to as a "Rake finger".

A Rake finger is illustrated in Figure 13.1, and the principle of a Rake receiver is presented in Figure 13.2.

The number of Rake fingers needed is determined by the multi-path environment. In a Rayleigh fading outdoor environment (ITU Pedestrian B) [3] up to 93% of the scattered energy could be utilized by a four finger Rake.

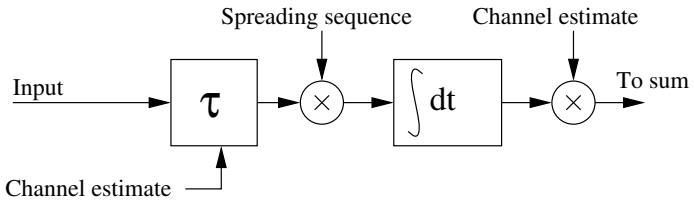

Figure 13.1: Rake finger.

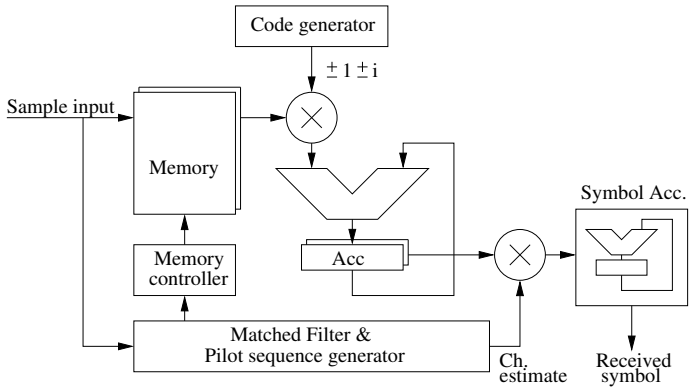

Figure 13.2: Principle of a Rake receiver.

13.2.3 Flexibility requirement

In 3G standards (WCDMA/TD-SCDMA) de-spreading is accomplished by first de-scrambling the data by a scrambling sequence (Gold code), then by separating each data channel by multiplication with an Orthogonal Variable Spreading Factor (OVSF) code.

User data rates can be varied by either reducing the spreading factor or by assigning multiple OVSF codes to a user. Traditional Rake receivers are usually implemented as pure ASIC solutions or as rigid accelerators to processor cores. However, this diversity among CDMA based standards including multi-code transmission requires large flexibility in the receiver. In contrast, conventional Rake-receivers de-spread each of the multi-path components by a single compound code. This obstructs the use of multi-

code transmission. However, by reordering the operations performed in a Rake receiver and running them in a programmable environment, it is possible to facilitate multi-code de-spread without any significant hardware overhead. In the CDMA based Wireless LAN systems such as IEEE 802.11b[7], data symbols are either spread by a sequence having good auto correlation function, or modulated by Complementary Code Keying (CCK). In this case the Rake receiver only performs MRC on chip level.

13.2.4 Multi-path search

Channel estimation can be divided into several tasks: Packet/Frame detection, multi-path search and symbol synchronization. Multi-path search is performed by using matched filters. However the diversity of the different standards render fixed matched filters inefficient. Since transmission delays for the different multi-path components can be large (greater than a number of symbol times), the channel estimator must not only search for synchronization on chip level, it must also search for symbol synch on all multi-paths.

In our processor the multi-path search is completely performed by software running on a 4-way complex ALU and a 2-way complex multiplication and accumulation (CMAC) unit. The hardware micro-architecture is discussed in section 13.4.

13.3 Architecture overview

The processor is constructed as a DSP processor with multiple SIMD-execution units. The data paths are grouped together into SIMD clusters. Each cluster has it's own load/store unit and vector controller. The clusters can execute different tasks while every data path within a cluster performs a single instruction on multiple data. The processor architecture is illustrated in Figure 14.4.

The processor core consists of five main blocks:

- A 4-way complex ALU.

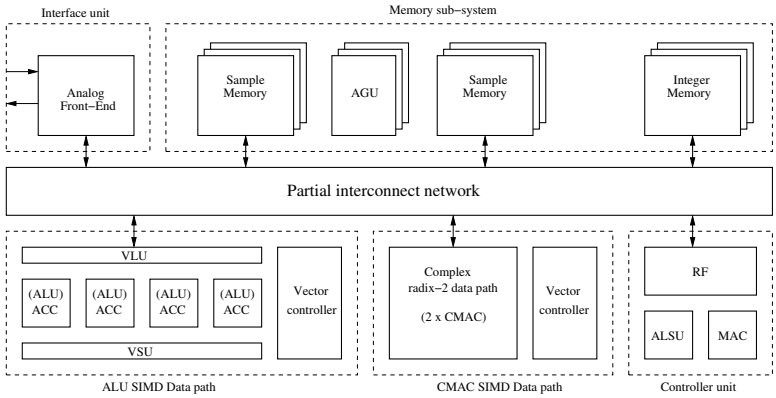

Figure 13.3: Processor micro-architecture

- A 2-way CMAC.
- A memory sub-system with address generators.
- A RISC-type controller.
- A partial interconnect network.

The SIMD units are the 4-way complex ALU, and the 2-way complex MAC. The main difference between this processor and a VLIW processor is the instruction issue process. By only allowing one instruction issue per clock cycle, excessive hardware could be saved. The SIMD clusters can simultaneously run independent tasks. When a task is finished, data are transfered between the SIMD units and the processor core by a memory swap[8]. Hence costly memory moves are avoided. Memory management is further discussed in section 13.5.

13.3.1 Instruction set

The instruction set architecture of the processor consists of three classes of compound instructions.

1. **RISC instructions**, operating on 16 bit integers.

2. **DSP instructions**, operating on complex numbers.

3. **Vector instructions**, running vector operations on a particular SIMD-cluster.

The RISC-instruction class contains most control oriented instructions and this instruction class is executed on the controller unit of the processor. (Control ALSU and MAC-unit). The DSP-instructions operate on complex-valued data and are executed on one of the SIMD-clusters. Vector instructions are extensions of the DSP instructions since they operate on large data-sets and utilize advanced addressing modes and vector loop support.

13.3.2 Instruction issue

Analysis of baseband algorithms shows that the receiving algorithm can be decomposed into task-chains with little backward dependencies between tasks. This allows different tasks to be performed in parallel on SIMD execution units. However, these tasks normally operate on large data sets, such as de-scramble or de-spread of a whole memory block. Instead of using a VLIW-scheme, this property of baseband processing is used in the design of the instruction set architecture. To reduce the complexity and improve the efficency of the control path, only one instruction can be issued every clock cycle. However since vector (SIMD) instructions run on long vectors, many RISC instructions can be executed during the vector operation. This is illustrated in Figure 13.4.

Unlike VLIW-machines, our architecture will allow concurrent vector operations and control flow instructions without the drawbacks and memory usage of a pure VLIW machine.

13.3.3 Task synchronization

In a micro-architecture containing several vector execution units, data synchronization is important. In our architecture, we provide special instructions for control flow synchronization. As shown in Figure 13.4,

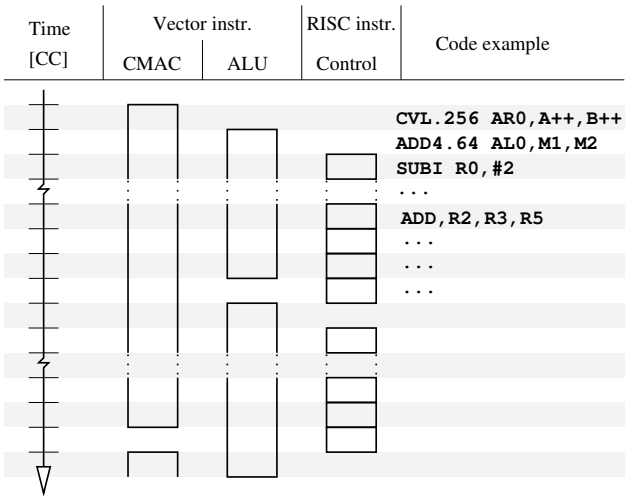

Figure 13.4: Instruction issue.

several vector operations can be executed in parallel. By using `idle` instructions, the control flow can be halted until a certain vector operation is completed.

13.4 SIMD processing clusters

This processor contains three execution units: the controller unit and two SIMD clusters. Common to the two SIMD computing clusters are the Vector controller and Vector load/store unit. (VLU/VSU). The VLU is the interface towards the memory blocks and the network interface. The VLU can load data in two different ways. In one mode, multiple data items are loaded each clock cyckle from a bank of memories. In the other mode, data are loaded one item at a time and then distributed to the SIMD-data paths in the cluster. This later mode is used to reduce the number of memory accesses when consecutive data are processed by the SIMD cluster.

13.4.1 Vector-ALU unit

The complex vector ALU along with address and code generators are the main components used for Rake finger processing.

By implementing a 4-way complex ALU unit with an accumulator, we can perform either four parallel correlations or de-spread four different codes at the same time. These operations are enabled by adding a "short" complex multiplier capable of only multiply by $\{0, \pm 1; 0, \pm i\}$. The short complex multiplier can be controlled from either the instruction word, a de-scrambling code generator or from a OVSF code generator. All sub-units are controlled from a vector controller which manages load and store order, code generation and hardware loop counting.

To relax the memory interface, a special vector load/store unit is employed. The VLU/VSU contains registers to reduce the number of memory data fetches over the network. If a consecutive data items are processed the load unit can reduce the number of memory fetches by $3/4$. A part of the micro-architecture of the Vector ALU is shown in Figure 13.5.

13.4.2 Vector-CMAC unit

The control and load/store structures for the vector CMAC unit are identical to the control structures of the Vector ALU. The vector CMAC contains two full complex data paths which can be run separately or together as a Radix-2 FFT butterfly. [6]

13.5 Memory sub-system

Memory management and allocation are critical in order to efficiently use SIMD architectures. The data memory consists of several small memory blocks with its associated Address Generator Unit (AGU). These memories are in turn connected to a partial interconnect network[8] where they can be connected to several different computing engine ports. The memory sub-system design can be divided into two different areas, address generation for Rake finger processing and data movement architecture.

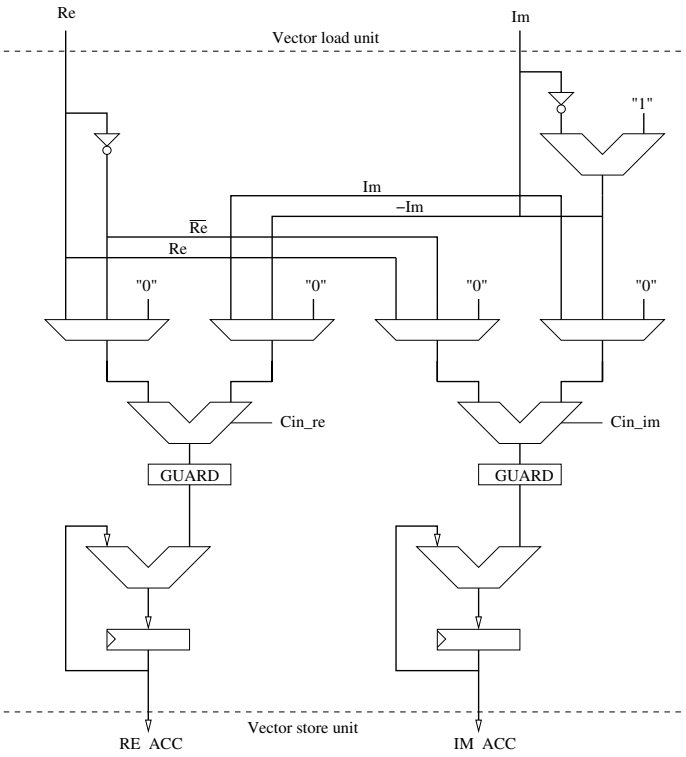

Figure 13.5: Part of vector ALU data path micro-architecture

13.5.1 Rake finger addressing

The length of memory buffers used for delay equalization is determined by the maximum delay-spread (Δt), the chip-rate ($1/T_c$) and the OSR. The minimum memory length is:

$$N = \Delta t \cdot \frac{OSR}{T_c}$$

$N = 184$ when $\Delta t = 12\mu s$[3], OSR=4 and $1/T_c = 3.84$ Mcps. Delay equalization in the Rake fingers is performed by using a circular memory buffer and a number of configurable address generators. Each physical memory contains it's own address generator capable of performing modulo addressing and FFT addressing.

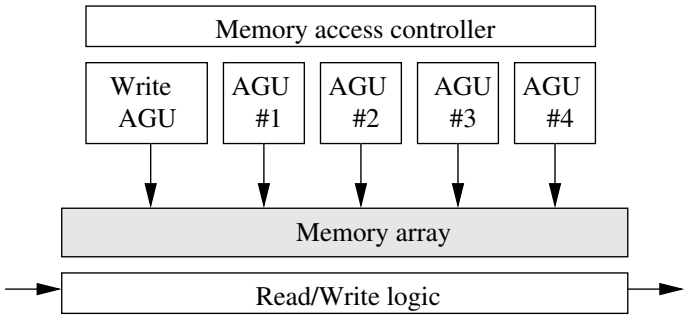

Figure 13.6: Memory addressing system for Rake fingers

Only one memory is used by all Rake fingers. This memory serves as a circular buffer where one sample is written and four samples are read for each input sample. To reduce the complexity of the memory architecture, the memory accesses are time interleaved. For WCDMA Rake finger processing the memory access rate is:

$$\text{Access rate} = \frac{OSR \cdot (1+N_f)}{T_c} = 76.8 \text{ [Mop/s]}$$

Where $N_f = 4$ is the number of Rake fingers.

The use of a circular memory buffer with configurable address generators allows us to handle transmission delays many times greater than one symbol time. The maximum transmission delay which is acceptable is only limited by memory length.

13.5.2 Data movement architecture

In a clustered SIMD processor it is important to have an efficient data movement mechanism to transport data between computing clusters. In our micro-architecture we use memories connected to a partial network. These memories can be connected to different ports on the different computing clusters. Since it is not necessary to connect all memories to all computing elements, the network is optimized to only allow certain memory configurations, hence "partial network". In order to transfer data

between these partial networks, several memory blocks are assigned to both sub-networks. These memory blocks are used as ping-pong buffers between tasks. They are illustrated in Figure 13.7.

Costly memory moves are avoided by "swapping" memory blocks between computing elements. This strategy provides an efficient and predictable data flow without costly memory move operations.

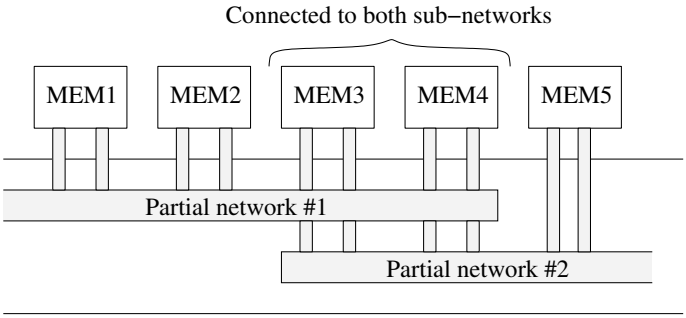

Figure 13.7: Ping-Pong memory blocks.

13.6 Functional mapping

The four main kernel functions performed by a rake receiver are:

1. Delay equalization.
2. Multi-Path search.
3. De-scramble and de-spread.
4. Maximum Ratio Combining.

By separating the de-scrambling (using Gold codes) operation from the de-spread operation (using OVSF codes), the same hardware can be reused between operations in WCDMA and TD-SCDMA systems. The mapping of the Rake finger functionality to hardware blocks is shown in Figure 13.8. First the 4-way complex ALU unit are used to de-scramble the

received data. By later feeding each accumulator with the corresponding OVSF-code the architecture can run de-spread on four simultaneous OVSF codes.

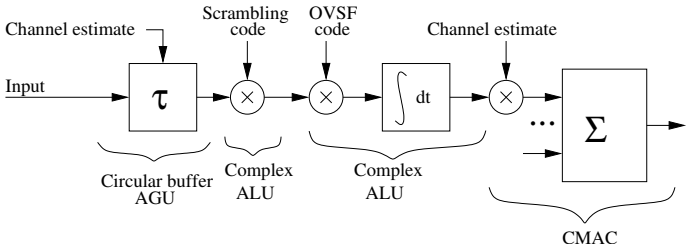

Figure 13.8: WCDMA Rake mapping.

The channel estimation task is also mapped to the same SIMD units as the Rake finger processing. Correlation with the spreading/pilot sequence is performed by the Complex ALU, whereas peak detection and absolute maximum search is performed by the CMAC unit.

13.7 Scheduling

In packet based systems such as IEEE802.11b, the frame is so large that it cannot be stored entirely in memory. This requires the channel estimation process to be completed before the reception of the payload data. As a result of this, the required processing performance is huge during the preamble stage. This is illustrated in Figure 13.9.

The processor architecture must have a peak performance matching the worst case computational load encountered.

In this case all hardware resources are allocated to the channel estimation process during the preamble. However, in WCDMA systems, channel estimation can be performed at the same time as the reception of data since a pilot channel is always transmitted.

Furthermore, the communication network and memory resources are statically scheduled in order to reduce the complexity of the memory ar-

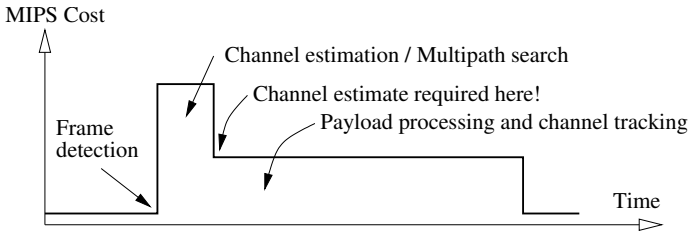

Figure 13.9: Worst case scheduling, IEEE802.11b

chitecture while maintaining a predictable worst case timing. To achieve optimal resource utilization, the number of computing elements is balanced between SIMD clusters so that each subtask consumes approximately the same number of clock cycles.

13.7.1 Power considerations

The processor architecture is designed with low power techniques in mind. The control paths are minimized due to the inherit control locality of the CSIMD architecture. Furthermore, single port memories are used to reduce the power consumption due to memory accesses. Also, when an execution unit is not used, it's inputs are masked to reduce unnecessary activity on internal logic.

13.8 Results

The main focus of this paper has been on a flexible and efficient micro-architecture capable of performing MRC-based channel equalization for common and future CDMA based communication standards in software.

We have presented a system architecture, a memory sub-system and we have mapped Rake kernel functions to this architecture. The kernel functions used by Rake receiving algorithms are benchmarked on the hardware and the result is listed below. Benchmarks illustrate the following cycle cost for different Rake kernel functions:

Kernel function	Length	Cycle cost	
`vabsqr`	64	66	$\|x_i\|^2$
`vmul`	16	18	$c_i \cdot x_i$
`vmac`	256	132	$\sum c_i \cdot x_i$
`vmac2`	256·2	260	$2\|\|\sum c_i \cdot x_i$
`vsmac`	64	18	$\sum (\pm 1 \pm i) \cdot x_i$
`vsmac4`	64·4	70	$4\|\|\sum (\pm 1 \pm i) \cdot x_i$

Limitations on the interconnect network restrict the data alignment for data used in parallel vector instructions such as `vmac2` and `vsmac4`. Complete receiver algorithms, including multi-path search and Rake finger processing are also benchmarked. The following results were achieved:

Standard	Required operating frequency
WCDMA	76 MHz
WCDMA-TDD	76 MHz
TD-SCDMA	65 MHz
IEEE 802.11b	72 MHz

As shown in the table above, this architecture is capable of performing all kernel functions associated with a rake receiver for the standards discussed at a clock rate not exceeding 76 MHz. This limit is given by the memory access rate for WCDMA. The cycle cost of control code execution is masked by vector operations running in parallel.

13.9 Conclusion

Programmability is essential for multi-standard baseband processors. In order to be able to process high bandwidth communication standards in a programmable processor new architectures are necessary. As a response to this, we have presented a micro-architecture for programmable baseband processors targeted at software defined radio applications. We have

also presented a mapping of Rake kernel functions to our programmable baseband processor. Our architecture is versatile and area efficient since we ensure maximum utilization of each execution unit.

[1] 3rd Generation Partnership Project; Physical channels and mapping of transport channels onto physical channels (FDD) 3GPP TS 25.211 V6.0.0

[2] L. Harju, M. Kuulusa, J. Nurmi; Flexible implementation of WCDMA Rake receiver; *Signal Processing Systems, 2002. (SIPS '02)*. IEEE Workshop on , 16-18 Oct. 2002 Pages:177 - 182

[3] 3rd Generation Partnership Project; User equipment radio reception and transmission; 3GPP TS 25.102 V6.0.0

[4] H. Holma and A. Toskala; *WCDMA for UMTS Radio Access For Third Generation Mobile Communications*, John Wiley and Sons, Inc., 2001.

[5] A. Nilsson, E. Tell; *An accelerator structure for programmable multi-standard baseband processors.* Proc. of WNET2004, Banff, AB, Canada, July 2004.

[6] Dake Liu, Eric Tell; *Chapter 23: Low power baseband processor for communications*, Low Power Electronics Design edited by prof. Christian Piguet, CRC press Nov. 2004 ISBN: 0849319412

[7] IEEE 802.11b, Wireless LAN MAC and PHY specifications: Higher-Speed Physical Layer Extension in the 2.4 GHz Band, 1999.

Chapter 14
Paper 3

Design methodology for memory-efficient multi-standard baseband processors

Anders Nilsson, Eric Tell, Daniel Wiklund and Dake Liu
Department of Electrical Engineering
Linkoping University
Linkoping, Sweden

ABSTRACT

Efficient programmable baseband processors are important in order to enable true multi-standard radio platforms and software defined radio systems. In programmable processors, the memory sub-system accounts for a large part of both the area and power consumption. This paper presents a methodology for designing memory efficient multi-standard baseband processors. The methodology yields baseband processor micro-architectures which eliminates excessive data moves between memories while still allowing true flexibility by utilizing SIMD clusters connected to memory banks via an internal network.

The methodology has successfully been used to create a multi-standard baseband processor for OFDM-based wireless standards. This paper discusses the IEEE 802.16e (WiMAX), DVB-H (Digital Video Broadcast - Handheld) and DAB (Digital Audio Broadcast) standards. The architecture is truly scalable to accommodate future OFDM systems. Scheduling and resource allocation shows that with the proposed memory structure and architecture, the processor can manage the baseband functions of the described standards operating at 80 MHz and using only 28k words of memory.

14.1 Introduction

In future radio systems, multi-standard signal-processing capability will be increasingly important. To support a large amount of different standards, programmable baseband processors are necessary since ASIC solutions are not flexible enough. Furthermore, ASIC solutions for multi-standard baseband processing are less area efficient than their programmable counterparts since processing resources cannot easily be shared between different operations.

Generally the memory sub-system accounts for a large portion of both the area and power consumption in digital signal processing systems [1]. It is important that the memory architecture balances bandwidth against the access rate to reduce the overall power and area cost for the memory

sub-system. In order to reduce power consumption in the processor it is also important to ensure that the data is accessible in the right memory. This will eliminate unnecessary data moves between memories.

This paper presents a design methodology for multi-standard baseband processors and a base architecture which the methodology uses. The base architecture consists of a DSP controller core, flexible SIMD execution units, and a memory system interconnected by a memory crossbar switch.

The methodology and the base architecture are demonstrated by mapping several OFDM-based radio standards to this processor architecture. This paper considers three different standards: IEEE 802.16e [2], DVB-H [3] and DAB [4].

All baseband functions such as synchronization, channel compensation, and frequency/timing offset cancellation have also been taken into account when firmware tasks have been analyzed and mapped to the processor.

The base architecture are optimized for vector oriented baseband processing rather than single operations unlike other programmable architectures [1], [5], [6]. This solution is based on memory tiles which are connected to the execution units through a memory crossbar switch. Instead of moving data between memories, data memories are reconnected to the appropriate input port of the corresponding execution unit.

The paper is organized as follows: In section 14.2 the design methodology are presented. The methodology utilizes a base processor architecture which is presented in section 14.5. In section 14.6 the design methodology are applied to the discussed standards, and results are presented in section 14.7. Conclusions are drawn in section 14.8.

14.2 Design methodology

In order to create an efficient baseband processor architecture, both in terms of memory usage, area, and power efficiency, a methodology has been developed that consists of the following steps:

1. Analysis and high-level modeling of the application and algorithm selection.
2. Mapping of the selected algorithms to a base architecture.
3. Benchmarking and analysis of the algorithms on the architecture to find limiting factors on memory usage, memory access rate, latency requirements and computing capacity.
4. Selection of memory sizes, memory organization and width of the execution units.
5. Selection of accelerators (if needed).

All of the steps above must be performed for all considered radio standards that should be supported by the architecture. By creating an architecture that supports the worst load in all corners of the design space, this will ensure architectural support for new and diverse standards which fall inside the limitations imposed by the worst case. The goal of the design methodology is to find parameters to the base architecture such as:

- Required clock frequency.
- Memory size and number of memory banks.
- Required execution units. (e.g. ALU and CMAC)
- Number of lanes in the execution units. The memory organization will follow the number of lanes in the execution units in order to fulfill memory access requirements.
- Which accelerators to use.

In the first stage of the methodology, a high level model is used to find all kernel functions used in the standards. These kernel functions will then be mapped to components in the base architecture framework. Information about processing deadlines and maximum allowed latency is also extracted from either the simulation models or the standard specification in this stage.

14.3 Analysis phase

To aid the analysis of the mapping of the selected baseband processing algorithms onto the base architecture, two important factors must be considered: MIPS cost and maximum allowed computing latency. These two factors will be considered for all kernel operations encountered during the first stage of the methodology.

14.3.1 MIPS cost

In order to compare the computational load of a certain algorithm, we define MIPS cost which corresponds to how many million instructions per second a regular DSP processor would require in order to perform the specified function. The MIPS cost is calculated as follows:

$$MIPS_i = \frac{OP_i \cdot N_i}{t_i}$$

where $MIPS_i$ is the associated MIPS cost, OP_i the number of clock cycles required by a standard DSP processor to perform the operation, N_i the number of samples or bits to process and t_i is the maximum time allowed for the operation to complete. The time to perform the operation is usually considered to be one symbol time.

This metric indicates the approximate computing complexity of an algorithm or operation. Since the time allowed for an operation to complete is dependent on scheduling, the MIPS cost analysis and scheduling must be iterated until a satisfying solution is found.

14.3.2 Latency

In most OFDM based systems there are three factors limiting the computing latency:

- **Channel equalization.** In a packet based system, the channel estimation performed on the preamble must be complete before the processing of the user payload symbols is started.

- **High level acknowledge packets.** In time division duplex systems and TDMA systems such as WLAN and WiMAX the standard stipulates the maximum allowed time from the end of a received packet to the beginning of the transmitted response packet.

- **User delay.** User protocols such as voice services limit the overall latency in a communication system.

14.4 Component selection

When the baseband tasks have been analyzed both in terms of latency requirements and MIPS cost, the results can be used to select vector execution units, memories, and accelerators.

14.4.1 Vector execution units

To ensure efficient execution of all kernel functions, we utilize vector execution units. Analysis of general OFDM signal processing reveals that all associated kernel functions are based on convolution, vector multiplication and FFT. Both convolution, vector multiplication and FFT can be performed on a multi-lane MAC unit. As a response to this we need flexible and efficient vector execution units mainly focused on MAC functionality. To increase the processing parallelism and maintain a high computing efficiency, we propose the use of multi-lane MAC units. A multi-lane MAC unit can easily be extended to perform a FFT butterfly each clock cycle [7].

The MIPS cost and memory access rate yielded from the analysis of the worst case load will be used to select the number of lanes in the execution units. The width of the execution units will be a trade-off between circuit area (more memories are needed in higher radix circuits) and desired clock rate. The total number of memory accesses must also be taken into account when selecting execution units since this has a large impact on power efficiency.

14.4.2 Memory organization

The number of memory banks is mainly determined on the chosen task level parallelism whereas the width of each bank is determined by the memory bandwidth required by the execution units.

A radix-2 compatible MAC unit needs to read three complex data items and write two complex data items each butterfly operation during FFT calculations. A radix-4 unit will on the other hand require a total of 11 memory accesses per butterfly operation.

14.4.3 Accelerators

When deciding which functions to accelerate the following must be must considered:

1. **MIPS cost.** A function with a high MIPS cost may have to be accelerated if the operation cannot be performed by a regular processor.
2. **Reuse.** A function that is performed regularly and is used by several radio standards is a good candidate for acceleration.
3. **Circuit area.** Acceleration of special functions is only justified if there can be considerable reduction of clock frequency or power compared to the extra area added by the accelerator.

An operation which fulfills one or more of the previous points is a good candidate for hardware acceleration.

14.5 Base architecture

To aid the design methodology, a base architecture has been used that serves as a framework to which we add components such as memories and execution units. A complete baseband processor consists of two main parts, one baseband processing part which mainly operates on vectors of complex numbers and one scalar part which operates on integers and single bits. The latter part is mainly used for forward error correction (FEC)

and bit manipulation whereas the former part is used to extract soft data symbols that can be de-mapped into bits. However, since bit manipulation and FEC tasks are so diverse from baseband processing we limit ourselves to the baseband functionality in this paper. An efficient multi-standard architecture for acceleration of bit manipulation and FEC tasks is presented in [8]. The base architecture consists of memory banks connected to execution units via a memory crossbar switch. The architecture is shown in Figure 14.1.

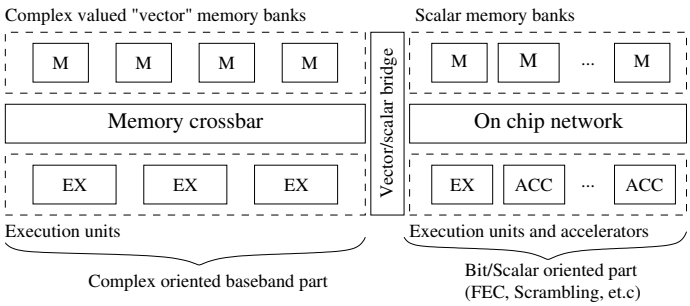

Figure 14.1: Base architecture.

The crossbar allows any memory to be connected to any execution unit. Execution units span the range from a DSP controller core, to multi-lane complex MAC and ALU SIMD data-paths. Accelerators such as an analog front-end and mapper/de-mapper together with a network bridge to the scalar part of the processing system are also attached to the memory crossbar.

The base architecture relies on the observation that most baseband processing tasks operate on a large set of complex-valued vectors (such as auto-correlation, dot-product, FFT and convolution). This allows us to use *vector instructions*, i.e. a single instruction that triggers a complete vector operation such as a complex 128 sample dot-product.

To support this kind of vector instructions, the execution units must be able to process large data chunks without any intervention from the processor core. This in turn requires the execution unit and memory

sub-system to have automatic address generation and efficient load/store subsystems. As a response to this, the base architecture utilizes de-centralized memories and memory addressing together with vector execution units.

14.5.1 Memory banks

Since the architecture relies on de-centralized memory addressing, each memory bank has an associated address generator unit (AGU). This unit is configured by the controller core before a vector instruction is issued and then operates autonomously during the vector operation. The AGU of each memory bank can either operate in normal, bit-reversed or modulo mode. The memory bandwidth can be increased by dividing each memory bank into several parallel memories. To enable the possibility to access all individual memories from a single port of the memory bank, we introduce a *reordering* crossbar switch. This crossbar switch allows any single memory to be accessed from any of the ports. This feature is especially useful when the memories are written by a single port device such as the mapper or the analog interface. Later, the memory content can be accessed in parallel in order to facilitate higher radix based algorithms and transforms.

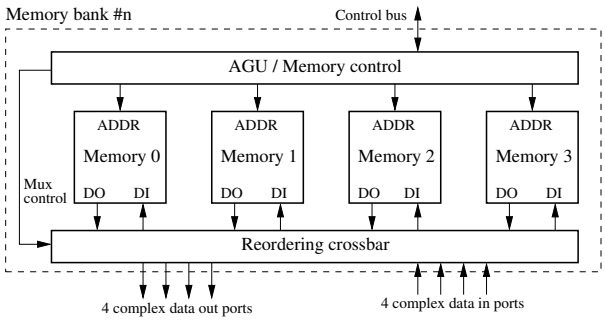

Figure 14.2: Example of a 4-way memory bank.

14.5.2 Task level pipelines

By using several memory banks connected to the memory crossbar switch, all execution units can operate in parallel. This allows the architecture to support "task level pipelines". Task level pipelining is a method of increasing the processing parallelism by running several independent jobs simultaneously and passing data between the jobs at specific times.

This technique is also used to store incoming samples in one memory while other execution units operate concurrently. When a processing task is finished the memory banks are reconnected so that the output memory buffer from an execution unit is connected to the input port of the next execution unit in the processing chain. This memory arrangement allows the base architecture to perform all OFDM reception and transmission tasks without any data moves between memories.

14.6 Application of the methodology

Following this methodology, high level models of the discussed standards have been developed and kernel functions have been derived. The kernel functions have been analyzed both in terms of resource usage and latency requirements according to the methodology. To be able to correctly account for all memory accesses, operations and data dependencies, the reception algorithms of OFDM systems have been analyzed with respect to memory usage, memory access rates, latency requirements, and memory access patterns according to the methodology. Since reception is more computationally demanding than transmission, we only consider reception tasks in this paper. Baseband reception tasks of OFDM data consist of the following steps[9]:

- Packet/frame detection.
- Synchronization.
- Frequency error estimation/tracking.

- Frequency error correction and decimation/filtering.
- Channel estimation/tracking.

- Guard period removal.
- FFT.
- Phase tracking and correction.
- Channel correction.
- De-map.

Tasks such as packet/frame detection and synchronization are only performed when the receiver initially searches for a valid OFDM packet or frame. The second task group is preformed on every received sample whereas the last group is performed on every received data symbol. Analysis and implementation of the reception tasks in the table above reveal three kernel operations which are frequently used:

- **Complex FFT** is used to transform time domain data into frequency domain data. The FFT is also used for cyclic correlation when performing synchronization.
- **Complex MAC** is used for dot-product calculations.
- **Complex vector multiplication** is used for channel compensation and cyclic correlation.

14.6.1 Vector execution units

To ensure the efficiency of a baseband processor, all kernel functions need to run as efficiently as possible. Since the base architecture assumes execution units which can sustain one operation per clock cycle, the memory access rate can be used as a measure of the MIPS cost. In Table 14.1 the

number of memory accesses are listed together with each operation. Since the discussed standards have many different modes, the most demanding has been used for this analysis.

Table 14.1: Overview of memory access rates and parameters

Operation	**IEEE 802.16e**	**DVB-H**	**DAB**
FFT	6144 op	147456 op	67584 op
Channel comp.	768 op	12288 op	6144 op
De-map	256 op	4096 op	2048 op
Symbol length: (incl. max GI):	320 pts	5120 pts	2560 pts
Symbol duration: (incl. GI):	$23.15\mu s$	$462\mu s$	$1246\mu s$
Avg. access rate:	309 MHz	354 MHz	60 MHz

In the previous table, a radix-2 based FFT calculated in a single multiplication unit is assumed. In practical signal processing circuits there are a few different efficient methods of calculating a FFT (or its inverse). Most common are techniques utilizing hardware which could perform a radix-2 or radix-4 butterfly in one clock cycle. As described in section 14.4.1 the radix of the execution units is a trade off between area and access rate. By using a radix-4 FFT butterfly, the memory access rate for FFT operations can be significantly reduced. Since the dynamic power consumption of CMOS circuits and memories are proportional to the clock frequency and access rate, it is beneficial to increase the processing parallelism to reduce the required clock frequency and lower the memory access rate. One way to increase the processing parallelism is to utilize Radix-4 butterflies in the FFT. A Radix-2 FFT of length N uses $\frac{N}{2}log_2N$ butterflies compared to $\frac{N}{4}log_4N$ butterflies for Radix-4. When single cycle Radix-4 butterflies are used to increase the processing parallelism, the memory sub-system needs to manage eleven memory transactions each clock cycle for the FFT in addition to the memory transactions needed for other concurrent oper-

ations.

14.6.2 Memory banks

By analyzing the kernel operations required in OFDM reception and all data dependencies, it can be shown that four memory banks of the same size as the received data symbol including guard period are sufficient in a flexible baseband processor. Latency requirements for all the discussed standards impose a limit which requires all symbol processing tasks to be executed within a symbol period. A mapping of the IEEE 802.16e OFDM standard onto the architecture described in Figure 14.4 is presented in Figure 14.3.

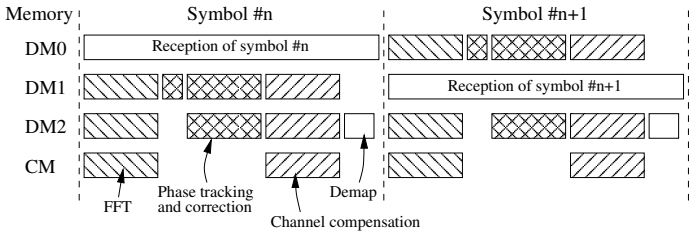

Figure 14.3: Memory scheduling for OFDM reception.

The memories are scheduled as follows:

1. DM0: Used to store incoming symbols.
2. DM1: Used as source for calculations.
3. DM2: Used as destination of calculation results.
4. CM: Used as coefficient memory for calculations and for storage of constant vectors.

This scheme and the memory crossbar switch eliminate the need for memory moves. A possible design alternative is to use a dual-port memory instead of the two data memories that are reserved for computations. However, since the power consumption and the area for a double-port

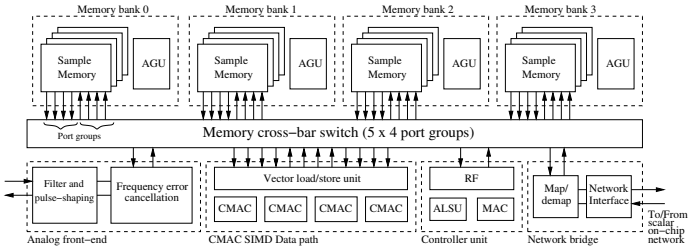

Figure 14.4: Resulting processor architecture

memory is roughly twice of a single port memory, we are only considering single port memories. The overhead due to many small memories must also be considered against both the required memory access rate and data access patterns.

14.6.3 Accelerators

Of the symbol processing tasks, four tasks are selected to be implemented as accelerators: The packet detector, frequency error compensation, filtering/decimation, and mapping/de-mapping. These four tasks fulfill all requirements for implementation as an accelerator. They are also used among all discussed standards and can be efficiently implemented in hardware.

14.7 Results

Application of the presented methodology on the three discussed standards yields the following architecture parameters:

1. Four memory banks of totally 28k words of memory.
2. A radix-4 compatible CMAC unit.
3. Acceleration of front-end tasks and mapping/de-mapping.
4. A controller core with scalar data-paths.

The resulting multi-standard OFDM processor architecture is presented in Figure 14.4 and the memory requirements are presented in Table 14.2. The maximum performance figures are related to the IEEE 802.16e standard whereas the maximum memory requirement are derived from the most memory demanding standard, DVB-H, which uses up to 5120 samples per symbol that are stored alternatingly in DM0 and DM1.

Table 14.2: Organization of memory banks

Bank	Amount	Organization	Total area
DM0:	4	2048x32	0.92 mm^2
DM1:	4	2048x32	0.92 mm^2
DM2:	4	1024x32	0.51 mm^2
CM:	4	2048x32	0.92 mm^2

In Table 14.3 peak memory power consumption and access statistics are presented for main receiving tasks in IEEE 802.16e implemented on this architecture.

Table 14.3: Cycle and memory access costs

Operation	Clock cycles	Accesses per cycle	Memory power	Energy
256 pts FFT	256	11	48 mW	154 nJ
Ph. tracking and comp.	264	1	4 mW	14 nJ
Ch. comp.	256	3	12 mW	42 nJ
De-map	64	4	16 mW	13 nJ

Area and power figures are collected from commercially available memory generators for a standard $0.13\mu m$ digital CMOS process. Mapping and simulations yield a maximum required clock rate of only 80 MHz when the presented architecture runs reception tasks of the discussed standards. This maximum operation frequency is a result of latency re-

quirements of the IEEE 802.16e standard which specifies the longest turnaround time between two packets.

14.8 Conclusions

The main focus of this paper has been to present a design methodology together with an efficient base architecture which yields memory efficient programmable multi-standard OFDM baseband processors. The proposed base architecture increases the memory efficiency and reduces the memory power consumption by using de-centralized addressing and vector execution units. By eliminating unnecessary moves between memories and reducing the required clock frequency of the processor, power can be saved while maintaining full flexibility. This architecture is also scalable to easily accommodate future standards such as IEEE 802.20 Mobile Broadband Wireless Access (MBWA) systems.

[1] T. Fujitsawa et al., *A Single-Chip 802.11a MAC/PHY with a 32b RISC Processor*, ISSCC Dig. Tech. Papers, pp. 144-145, Feb. 2003.

[2] IEEE Standard for local and metropolitan area networks, Part 16: Air Interface for Fixed Broadband Wireless Access Systems. WirelessMAN-OFDM

[3] ETSI EN 300 744, Digital Video Broadcasting. DVB-T/DVB-H

[4] ETSI EN 300 401, Radio broadcasting systems: Digital Audio Broadcasting to mobile, portable and fixed receivers.

[5] J. Kneip et.al. *Single Chip Programmable Baseband ASSP for 5 GHz Wireless LAN Applications*, IECICE Trans. Electron., vol.E85-C, N0.2 February 2002.

[6] J. Glossner et al, *A Software-Defined Communications Baseband Chip*, IEEE Communications Magazine, January 2003.

[7] Eric Tell, Olle Seger and Dake Liu; A Converged Hardware Solution for FFT, DCT and Walsh Transform; Proc. of the *International Symposium on Signal Processing and its Applications (ISSPA)*, Paris, France, Vol. I, pp. 609 - 612, July 2003

[8] Anders Nilsson, Eric Tell and Dake Liu, *An Accelerator Structure for Multi-Standard Programmable Baseband Processors*, Proc. of IASTED Intl. Multi-Conf. on Wireless and Optical Com., pp 644-649, July 2004

[9] H Heiskala and J T Terry, *OFDM Wireless LANs: A Theoretical and practical guide*, Sams Publishing, 2002