Architecture and Application of a Dynamically Reconfigurable Hardware Array for Future Mobile Communication Systems

Ahmad Alsolaim**, Jürgen Becker*

*Darmstadt University of Technology
Institute of Microelectronic Systems
Karlstr. 15, D-64283 Darmstadt, Germany
e-mail: {becker, glesner}@mes.tu-darmstadt.de

Manfred Glesner*, Janusz Starzyk**

**Ohio University
Electrical and Computer Engineering
Athens, OH 45701 USA
e-mail:{alsolaim,starzyk}@bobcat.ent.ohiou.edu

Abstract

The evolving of current and future broadband access techniques into the wireless domain introduces new and flexible network architectures with difficult and interesting challenges. The system designers are faced with a challenging set of problems that stem from access mechanisms, energy conservation, error rate, transmission speed characteristics of the wireless links and mobility aspects. This paper presents first the major challenges in realizing flexible microelectronic system solutions for future mobile communication applications. Based thereupon, the architecture design of flexible system-on-a-chip solutions in the digital baseband processing for future mobile radio devices is discussed. The focus of the paper is the introduction of a new parallel and dynamically reconfigurable hardware architecture tailored to this application area. Its performance issues and potential are discussed by the implementation of a flexible and computation-intensive component of future mobile terminals.

1 Introduction and Motivation

The various future demands for flexible mobile communication systems presents a set of challenging problems to system designers in the wireless industry. The requirement of future generations mobile terminals can be summed in dynamic flexibility, higher performance and less power consumption compared to current terminals. The combination of advances in integrated circuit technology and novel system-level solutions can contribute efficiently to the widespread commercialization of mobile high-speed communication systems. In the last years, the fast technological development in *very large scale integration* (VLSI) possibilities has brought the notion to single *system-on-a-chip* (SoC) solutions. Thus, the implementation of various functions required by different abstraction layers of a wireless mobile network should result in a highly integrated single-chip in the future, according to the dramatic improvement in the size and speed of electronic devices in recent years. Trends in microelectronic systems design point to higher integration levels, smaller form factor, lower power consumption and cost-effective implementations. The achievement of this goal has to be efficiently supported by the concurrent development of new design methods including in addition such aspects as flexibility, mixed-signal system-level exploration, re-usability and top-down SoC design. The design of mobile baseband systems involves several heterogeneous areas, covering various aspects in communication system application, in efficient CAD tool support, as well as in microelectronic architectures and technology questions. A good understanding of all relevant points related to those inter-disciplinary areas is essential to the success of the final product.

Future mobile communication systems, e.g. third generation (3G) systems, will not only offer the same old services (voice transmission and low data rates) with improved quality, but in addition these devices will have to offer many new exciting services, which will range from internet browsing to real-time multimedia communication applications. Moreover, next generation mobile terminals should also support new services that will soon emerge when the system is deployed. The upcoming future standards should also allow the introduction of such new services as easy as possible. Thus, the design of a corresponding mobile system has to reflect all these forecasted services and flexibility. At the same time the mobile devices should realize all services within the physical and operational requirements of the given mobile system infrastructure. In addition, the mobile terminal has to provide an acceptable power consumption in order to be feasible for multimedia terminal operation. Finally, the *time-to-market* and low price requirements have to be fulfilled in order to be competitive. This results in the following two major requirements:

- a new efficient system model in 3G systems, e.g. CDMA-based transmission schemes being highly flexible and adaptable to new services [3], and
- new innovative flexible microelectronic design solutions.

Currently, most of the microelectronic system solutions in mobile communication are a combination of ASICs, microcontrollers, and Digital Signal Processors (DSP) devices. Reconfigurable hardware architectures have been proven in different application areas [1] [2] to produce at least one order of magnitude in power reduction and increase in performance. Reconfigurable architectures are suitable for implementing filters, correlators, multipliers etc. In [2] a universal data-driven reconfigurable accelerator machine has been developed, which is targeted to execute computation-intensive parts with regular data dependencies out of all kind of scientific applications. In our approach we evaluate hardware architecture trade-offs regarding universality versus application-specific solutions. Therefore, the potential to integrate application-tailored coarse-grained dynamically reconfigurable architectures into SoC-solutions for future generations mobile terminals is described in this work and demonstrated by the performance of analyzed computation-intensive application parts within mobile communication systems. The proposed reconfigurable SoC-architecture parts provide for the here selected application area and components more flexibility than ASICs and better performance values than DSPs or even today´s fine-grained commercial reconfigurable devices [31] The major general goal is to evaluate flexibility versus power/ performance trade-offs by releasing the DSP for other tasks, or by migrating functionality from ASICs to our coarse-grained reconfigurable hardware supporting the implementation of highly efficient SoCs for hand-held devices in mobile communication systems.

The paper is structured as follows: in section 2 the future challenges and considerable aspects for cost-effective, low power and high-performance SoC implementations in next generation´s mobile communication systems are discussed. Here, potential parallel hardware/software architectures of flexible reconfigurable SoC-solutions for the digital baseband processing are sketched. Section 3 provides a detailed description and introduction of the proposed dynamically reconfigurable parallel hardware part of such SoCs, including its evaluated performance issues and implementation status. The usefulness of this new coarse-grained parallel array architecure is illustrated in section 4 by the implementation analysis of a computation-intensive application part in future mobile terminals, e. g. a flexible CDMA-based RAKE-receiver.

2 Flexible Hw/Sw System-on-a-Chip Solutions for Mobile Communication

2nd generation (2G) mobile communication systems, i.e. GSM and IS-95 standards, had been rigorously defined and optimized to provide operation for a certain application. On the other hand, 3G systems, i.e. based on the UMTS standard, will be defined to provide a transmission scheme

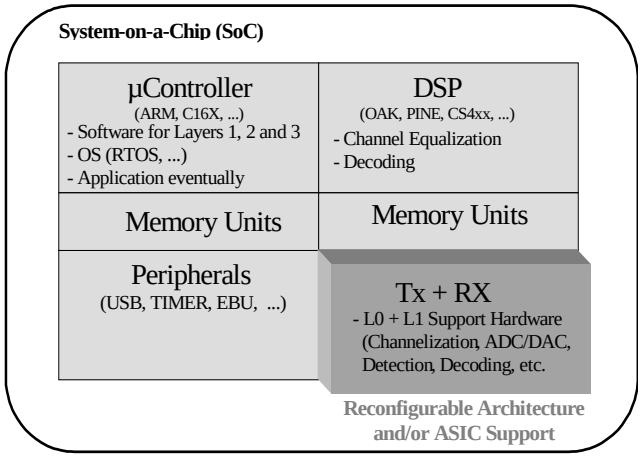


Figure 1 SoC-Architecture Components of a *Baseband Single Chip Mobile Receiver*

which is highly flexible and adaptable to new services [35]. This vision add a new dimension to the challenges within the digital baseband design, since the final microelectronic systems must be able to support this flexibility and adaptability. In addition, the cost of the final system is closely related to the number of components assembled on board. The reduction in cost is not only gained by reducing the cost of components, but also by reducing the cost of assembly and testing of the final system. The *time-to-market* introduction and the reduction of cost are the most important factors in determining the success of final products, especially in this application area. In the literature are many proposed solutions listed to handle these flexibility requirements [5]. Concepts such as *Software Radios* are discussed in detail [4] [7]. Since within such concepts the necessary overall system performance is missing [27] [28], alternative solutions have to be developed. For computation-intensive functions with flexibility requirements found in mobile communication applications, reconfigurable hardware offers an alternative solution to the software programmable DSPs. The DSP has been the basis of the hardware implementation of digital communication applications for the last fifteen years or more. But in the last five years, reconfigurable computing has emerged as a new hardware implementation methodology, with a very promising performance, also for arithmetic data-paths applications that dominate the digital communication [1] [2]. This relatively new hardware architecture concept can provide increased system performance at lower cost and risk of system implementation. Thus, this so-called structural programmable logic combines the flexibility of a general-purpose DSP as well as the speed, density, and low cost of ASIC solutions. Therefore, the enhancement of reconfigurable hardware to increase system performances within the here viewed application area can be two fold:

- in some applications programmable logic substitutes the DSP processor entirely, or
- programmable logic works concurrently with the DSP processor, migrating computation-intensive functions, e. g. to save power and to release the DSP processor for other functions.

DSP processors are more suitable to handle algorithms that show a large amount of irregularity in execution. Such irregularities are reflected in the throughput rates, control flow or frequent branching. On the other hand, reconfigurable hardware architectures are in general well suited for applications such as matched filters, FIR filters, and correlators [6]. The major reason that these architectures perform better than DSPs in such applications is the fact that the corresponding algorithms can be decomposed into many parallelized tasks, whereas each task often involves operations such as Multiply-accumulate (MAC). Since most DSPs offers only single MAC-operation units, the parallelizable computations have to be executed serially, instead of being executed in parallel in FPGAs.

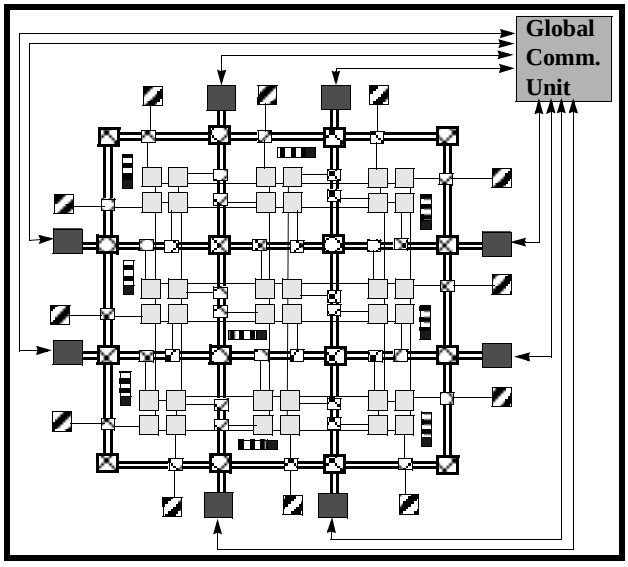
In addition, there are many operational challenges, such as battery life, easy and flexible terminals to exploit dynamically different and new services, e g. also by downloading upgrades and new services or protocols from the internet and configure the hand-held devices according to these downloaded codes. Flexibility can be defined as the ability of the mobile terminal to support many modes of operation, e g. voice, audio, video, navigation, data transmission etc.. This means for example also, that the mobile device has to have the ability to operate within different standards, such as GSM and IS-95. Adaptability is the ability of the mobile terminal to easily and quickly accommodate a new service. In addition to the mentioned requirements, with the more and more important *time-to-market* and *risk minimization* aspects apply here by reducing the ASIC development risks and times enormously, or by avoiding ASIC fabrication completely in some cases.

The target SoC architecture may be composed of different cores such as DSPs, microcontrollers and memories, as well as of reconfigurable hardware and/or various ASIC support parts. An overview of a possible SoC architecture related to a *Baseband Single Chip Mobile Receiver* is shown in Figure 1. As stated above, the choice of the final target architecture will result from a detailed application and performance analysis while considering VLSI oriented implementation issues. The required flexibility will be supported by the inclusion of a new coarse-grained dynamically reconfigurable architecture realizing layer 0 (L0) and layer 1 (L1) hardware operations, e.g. channelization, detection, decoding etc.. Since in next generation wireless communication systems a high degree of flexibility is required during operation, dynamically reconfigurable architectures have a great potential to support the corresponding applications efficiently and to substitute several ASIC or DSP computation parts within today´s microelectronic solutions.

3 A Dynamically Reconfigurable and Parallel Array Architecture

As explained in the previous sections, next generation´s mobile communication systems will provide the customer with a large variety of different services, whereas some of them are not known yet. In addition to that, known services have large spectrum of requirements, e. g. different data rates, different quality of services (QoS), and real-time services etc.. For preparing future mobile terminals and its microelectronic components to cope with all these challenges, we developed a new coarse-grained and dynamically reconfigurable architecture. The integration of this application-tailored but flexible hardware architecture within flexible SoCs solutions for the digital baseband processing will support the efficient realization of the above mentioned features.

The proposed *Dynamically Reconfigurable Architecture for Mobile Systems (DReAM)* consists of an array of parallel operating coarse-grained *Reconfigurable Processing Units*



Legend:

- Reconf. Proc. Unit (RPU).
- Dedicated IO (DIO).
- Comm. Switching Unit (CSU).
- Configuration Memory Unit (CMU) and its Controller.
- Switching Box (SWB).
- RPU to Bus Connection point.
- 16-Bit Global Interconnect Line.
- 16-Bit Local Interconnect line.

Figure 2: Hardware Structure of the Dynamically Reconfigurable DReAM Architecture

(RPUs). Each RPU is designed for executing all required arithmetic data manipulations for the data-flow oriented mobile application parts, as well as to support necessary control-flow oriented operations. The complete DReAM array architecture connects all RPUs with reconfigurable local and global communication structures (see Figure 2). In addition, the architecture will provide efficient and fast dynamic reconfiguration possibilities for the RPUs as well as for the interconnection structures, e.g. only partly and during run-time while other parts of the reconfigurable architecture are active. In the following, the design, structure and performance issues of the major hardware components in the DReAM architecture are explained.

The decisions during the design of the architecture were mainly based on the careful reviewing of the tailored application area requirements and on our personal design experiences. We started the design of DReAM performing a bottom-up approach by focussing on a list of the most complex and flexibility demanding application parts in future mobile receivers, e. g. RAKE-receiving parts, interpolation filtering, searcher algorithms, coding and modulation techniques etc.. Based on the set of used arithmetic and control-flow operations the performance/power optimized structure development of the RPUs, and of so-called *Communication Switching Units (CSUs)* was done. As shown in Figure 2, the DReAM architecture consists of a scalable array of RPUs that have 16-bit fast direct local connections between neighboring RPUs, whereas each four RPU sub-array shares one common *Configuration Memory Unit* (CMU). The CMU holds configuration data for performing fast dynamic reconfiguration for each of these four RPUs and is controlled by one responsible CSU. Each CSU controls two CMUs and four global interconnect *Switching Boxes* (SWB). The detailed hardware structure of the SWBs and the related global as well as local inter-RPU communication mechanisms are described in section 3.3.

In addition, *Dedicated I/O Units* (DIOs) for fast and parallel transfers of the input/output data of DReAM are placed around the array architecture. Each DIO can be connected either to one RPU at the border of the DReAM architecture, and/or to RPUs inside the array through the global interconnect lines. Every DIO is able to perform the internal local and global DReAM communication protocols (see section 3.3), as well as the interfacing functionality to the other components of the flexible SoC, e.g. the DSP, the microcontroller, and the on-chip memories. Thus, the final functionality and hardware structure of the DIOs is not completely fixed yet, because the internal SoC communication interfaces have to be decided later.

All CSUs communicate to one *Global Communication Unit* (GCU), which coordinate centralized all dynamic reconfiguration steps, e. g. of the RPUs as well as of the global interconnection structure. Moreover, the GCU controls the external communication with other hardware components of the flexible SoC. During dynamic reconfiguration of the RPUs, first, the GCU triggers the responsible CSUs to initiate the controllers of the corresponding CMUs for loading a new configuration to the selected RPUs. This fast transfer process of reconfiguration code is then completed by an efficient burst-mode protocol between the CMU- and the RPU-controllers. The dynamic reconfiguration algorithm for various scenarios (see section 2) in future mobile communication is performed by the GCU, e. g. deciding which parts of the array have to be reconfigured with a particular configuration data set, stored either in the CMUs or in an extrnal memory.

3.1 Hardware Structure and Performance of the Reconfigurable Processing Unit (RPU)

The dynamically *Reconfigurable Processing Units* (RPUs) are the major hardware components of the DReAM architecture for executing the arithmetic data manipulations within application parts of mobile communication systems requiring flexibility, e. g. dynamic hardware reconfiguration. Thus, these application-tailored RPUs perform efficiently the required coarse-grained (8-bit) integer operations needed here. In contrast, the CLBs (*Configurable Logic Blocks*) of today´s commercially available fine-grained and universal FPGA-chips are operating on the 1-bit level [31]. As shown in Figure 3 each RPU consists of:

- two dynamically reconfigurable 8-bit data paths, called *Reconfigurable Arithmetic Processing Units* (RAPs),
- one *Spreading Data Path* (SDP),
- one *RPU-controller*,
- two *dual port RAMs*, and
- one *Communication Protocol Controller*

Each RAP can perform all necessary arithmetic operations (8-bit) identified in the above mentioned examined application parts of mobile communication systems. The performance values of these operations for n operation repetitions on a stream of data are provided in Table 1, and are based on a 0.35 µm CMOS standard cell synthesis for the RPU by using an Mietec/Alcatel process. For the repeated operation execution only one configuration set is necessary. The available set of two-input 8-bit operations support either operations with one fixed operand (constant), as well as operations with two variables as inputs. The detailed hardware description of these fast integer operation implementations will be provided in section 3.2.

One *Spreading Data Path* (SDP) for fast and efficient execution of CDMA-based spreading tasks is designed and implemented in each RPU. This SDP unit can be used together with the adding operations of 2 RAPs for implementing efficiently fast complex PN-code correlation operations

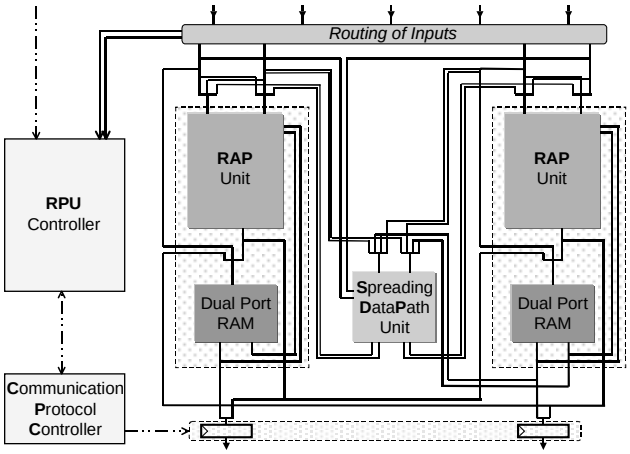


Figure 3 Hardware Structure of the *Reconfigurable Processing Unit* (RPU)

(see section 3.2). Such spreading operations are required often in QPSK-modulation (*Quadrature Phase Shift Keying*).

The *RPU-controller* is responsible for guiding all data manipulations and transfers inside the RPU, as well as to determine from which local neighbour RPU or global interconnect line input data is consumed. Moreover, the *RPU-controller* performs together with the CMU and its controller the fast dynamic reconfiguration of the RPU. .

Operation)*)* n repetitions of single operations (0.35 µm CMOS)	**Speed Best Case**		**Speed Worst Case**	
	cycles	**freq. [MHz]**	**cycles**	**freq. [MHz]**
Multipl. with changing Y	n+10	60	10n+1	11.8
Multipl. with fixed Y	n+2	100	n+10	60
Division	n	120	43n	2.8
MAC with changing Y	3n	40	11n	10.9
MAC with fixed Y	n+2	100	n+11	57.1
Addition	n	120	n	120
Subtraction	n	120	n	120

Table 1: DReAM Operation Performance Values

The two 16-by-8 bit *dual port RAMs* within each RPU are used as look-up-table (LUT) when performing the fast 8-bit multiplication operations (see section 3.2). In addition, both can be used as normal memory, or one of these two RAMs can be used as a normal data memory, if necessary with FIFO-behavior, e. g. for buffering intermediate results within the data-driven and asynchronously operating DReAM array architecture.

3.2 Efficient 8-bit Constant/Variable Arithmetic Operators for Mobile Computing

Reconfigurable Arithmetic Processing Units (RAPs): The RAP unit is built around fast 8-bit integer multiplier operators, as shown in Figure 4. The multiplier is designed to provide a very high speed constant multiplication (i.e. one of the operands is constant for some time interval) and small compact design by using modified Look-Up Table (LUT) multiplication procedure. According to [33] most of the multiplication within the mobile system are fixed operand operation. The main idea of the LUT multiplication is to generate all possible multiplication of the fixed operand Y (8-bit) and store them in the LUT, then use the variable operand X as the address to the LUT, so providing the result R=X*Y in one cycle. The problem of such implementation is that the large number of cycles needed to generate all the 256 multiplicands of 8-bit Y operand, and the large area consumed by the LUT.

In typical multipliers, the LUT required for 8-bit multiplication is 16-words (12-bit) deep. This is done by breaking the the address of variable operand X into two parts, upper and lower parts. This way the number of cycles to generate all multiplications of operand Y are reduced to 16 cycles. But the LUT need to be accessed two times to generate the two intermediate results and then add them to get the final result. In the RAP multiplier dual port RAM is used for the LUT so that the intermediate result can be read at the same time. The LUT-size is reduced to 8-words (12-

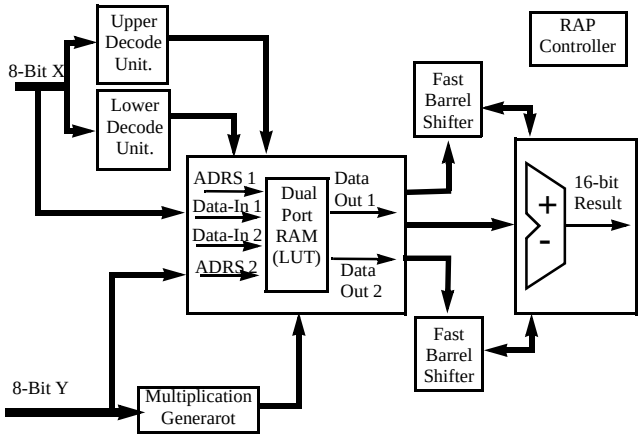


Figure 4: Hardware Structure of the *Reconfigurable Arithmetic Processing Unit* (RAP)

bit) deep by storing only the odd multiples of the fixed operand Y, the even multiples can be generated from an odd multiple by a simple shift operation. For example 2*Y and 4*Y multiples can be generated by a 1 or 2 shifts of the Y operand. On the other hand 10*Y can be generated by reading 5*Y from the LUT and then shifting once. A barrel shifter is provided for fast shift operations. Simple and small decoding units in front of the LUT are used to generate the number of shifts and the corresponding LUT-addresses. As shown in Table 1 the speed of the multiplier is 60 Mhz when Y is fix, but it reduces to 11Mhz if Y is variable. In page 10 is reported that using a similar method with two ROM-based Look-up tables implemented on Xilinx XC4000-5 FPGA has a speed of less than 42 Mhz. The multiply-accumulate (MAC) operation is as fast as the multiplication operation. Additional registers and multiplexers are added (not shown in Figure 4) to implement the MAC operation. Division is a simple shift and compare algorithm, which has been chosen to use the available hardware. The addition and subtraction are straightforward implementations on the RAP unit.

The RAP unit can adapt it's speed according to the operand nature, i.e. whenever the Y-operand is constant for some period of time, the RAP unit will increase its speed drastically. As shown in Table 1, the best and worst cases of the RAP's speed during the execution of one operation corresponds to a constant or changing Y-operand, respectively.

A configurable *Spreading Data Path* (SDP) is implemented in every RPU. The SDP-unit can be configured to do either a complex or normal fast and efficient execution of CDMA-based spreading tasks. As shown in Figure 10, the unit can be used with the two adders in the RPU to perform a complex correlation function found in (*Quadrature Phase Shift Keying*) QPSK-modulation, or it can perform one- or two-channel normal correlations, as found in (*Binary Phase Shift Keying*) BPSK-modulation. It is mainly designed to perform a complex correlation operation for QPSK-scheme on 8-bit data word with serial code (PN-code, for example) for any number of iterations N (code length, for example), according to the following equations:

$$Out1 = \sum_{n=0}^{N} -PN1 \cdot IN2 + PN2 \cdot IN1$$

$$Out2 = \sum_{n=0}^{N} PN1 \cdot IN1 + PN2 \cdot IN2$$

For one- or two-channel BPSK:

$$Out1 = \sum_{n=0}^{N} PN1 \cdot IN1$$

The SDP can also be utilized in many other functions, e.g. despreading, synchronization, etc. The inputs to the unit can come from outside the RPU or from the local memory.

The *RPU-controller* is a FSM-based control unit that is responsible for guiding all operations for data manipulations and transfers within the RPU. In addition to the typical controlling tasks, the controller supports also conditional operations implementations. Moreover, the *RPU-controller* performs, together with the CMU and its controller, the fast dynamic reconfiguration of the RPUs.

Two *dual port RAMs* are available within each RPU. These RAMs are 8-word deep and 16-bit wide. The RAMs are used as look-up-tables when performing the fast 8-bit multiplication operations by the RAP units. In addition, both RAMs can be used as a normal data memory with a simple addressing mechanism. One of the RAMs (RAM 2 in page 5) is equipped with a FIFO-controller to implement an 16X8 asynchronous FIFO. The controller is also able to connect with FIFOs controllers in other RPUs that fall in the same vertical line to form a larger FIFO-based RAM of the size 16X16, 16X32, etc.

3.3 Fast Inter-RPU local and global Communication Mechanisms

Each RPU is locally connected to it's four neighbors (North, East, South, and West) through 16-bit fast direct connection lines. In addition, it can be connected to the global lines through a SRAM-based switching box (SWB), as shown in Figure 2. The data-driven communication mechanism inside the DReAM array architecture is realized by an asynchronous communication protocol, performed on the 16-bit local and global interconnect lines. The protocol is an efficient hand-shaking protocol. For local communication between neighboring RPUs a half-interleaved handshake is implemented (1-cycle delay), and for global inter-RPU communication a fully-interleaved handshake is used (2-cycle delays). This has to be done due to the difference in

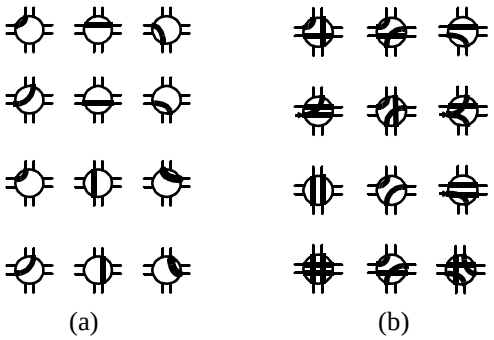


Figure 5: Switching Box routing. (a) Possible switching connections. (b) Multiple connection combination examples.

length between the local and the global interconnect wires, resulting in different communication signal delays.

Each RPU has a *Transmitting Unit* (TXU) and *Receiving Unit* (RXU). When an RPU is ready to receive data, a *ready-signal* is sent to the transmitting RPU, followed by an *acknowledgment-signal* after the receiving RPU have consumed this data. Whenever one RPU cannot consume arriving data, a *halt-signal* has to be sent to the transmitting RPU to delay the transfer operation until a *resume-signal* is received. For this reason a 2-bit control line run in addition to every data line so that the TXU in the transmitting RPU and the RXU in the receiving RPU can be synchronized by the halt/resume and ready/acknowledge signals.

The global interconnect lines are implemented by two 16-bit lines, running to neighbouring SWBs in the way, that each RPU has access to global interconnection (see Figure 2). As shown in Figure 6, each global line coming from one direction can be routed to any of the other three directions. Each SWB consists of 12 switching-points being implemented by 16 SRAM-controlled pass transistors. The lines are named based on their location and direction with respect to the SWB, e. g. a line going to the upper west direction is called *west upper line* (WUL). The upper horizontal lines can be connected with the left vertical lines, and the lower horizontal lines can be connected to the right vertical lines (see Figure 6). Based thereupon, Figure 5 (a) shows all possible one-line connections, which can be combined in all possible ways, if there is no resource conflict by two one-line connections driving the same line. Some examples of multiple-line connections are given in Figure 5 (b).

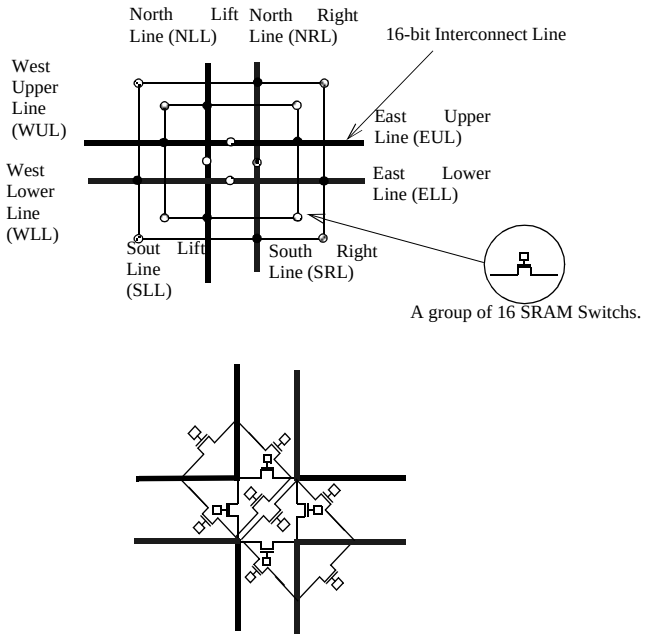


Figure 6: SRAM-based Switching Box (SWB)

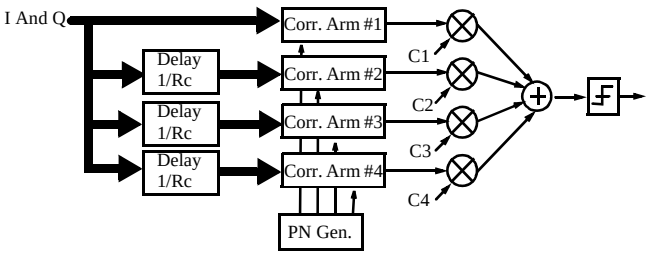


Figure 7: Front-end of RAKE-Receiver

4 Application Mapping: Flexible CDMA-based RAKE-Receiver

The following example of a computation-intensive RAKE-receiver component from future mobile communication systems will be mapped manually onto DReAM. The goal of our approach is to map selected application parts onto the reconfigurable SoC-part, which are too complex for the DSPs, especially for low power DSPs with reduced clock rates, but which still require operation flexibility, so that ASIC implementations would be not sufficient. For future CDMA-based mobile communication systems RAKE-receivers are essential while consuming a huge computation performance [33]. ASIC implementations could provide such a performance [30], but are not flexible enough to adapt to the various situations and services in the next generation of wireless communication systems. Another important point is the risk minimization by the implementation flexibility of reconfigurable hardware, which is very important, especially in the case of late specification changes, or different and changing standards.

In DS-CDMA systems the data signal is multiplied by a PN sequence with much higher bandwidth than the one of the data signal. Thus, the Spread Spectrum (SS) signal is well matched for multipath channel. i.e. more than one copy of the transmitted signal will arrive at the receiver with different time delays. If the time delay between the received copies is more than one chip duration, then a RAKE-receiver (with n fingers, where n is the number of potentially path signals) can resolve and then combine the signals according to their SNR as shown in Figure 7. The received data signal for each finger can be found as a complex correlation (QPSK-modulation) of the binary delayed signal y(t) and the conjugate of the PN-code (*pn**) [30]

$$r = \sum_{n=0}^{L-1} \int_0^T y\left(t - \frac{n}{W}\right) c_n^*(t) pn^*(t) dt$$

where *cn** is the complex conjugate of the weighting factor for each finger.

Parameter	Value
Chip Rate Rc	4 to 32 Mcps
Spreading Gains N	16 to 1024
Data Rate Rd	4k to 2Mbps
Carrier Freq. Fo	2GHz
Receiver Speed v	0 and 20 Kmph
Channel Model	AWGN and Doppler.

Table 2MATLAB simulation parameters of RAKE-Receiver

A RAKE-Finger despread the received signal with a correlator. The despreaded signal is then multiplied by a complex amplitude to correct phase-error and to weight each finger according to *Maximal-Ratio-Combining* (MRC) strategy. The received data is first converted from the analog to the digital domain (ADC) and then 4 times oversampled. As shown in the Figure 10, every four data samples in the two branches In-phase (I) and Quadrature (Q) are loaded to the RAM. Then, the synchronization unit block select the appropriate sample from the I- and Q-branches in order to reduce the phase-difference between the I- and Q-branches. The data coming out of every finger is weighted according to it's SNR level. The data of all fingers are then combined before dumping the data to the demodulator (not shown in Figure 10). More signal processing can take place afterwards (e.g. de-interleaving and decoding). The number of Rake-fingers depend on the channel-profile and the chip-rate, e. g. the higher the chip rate, the more resolvable paths [32]. IS-95 uses RAKE-receivers with four fingers [35]. Table 2 shows the simulation parameters that has been used for the simulation of the implemented receiver. We used MATLAB 5.2 for simulating a RAKE-Receiver with four fingers for the data rate required in the third generation standard. Assuming, there is ADC that can support that high sampling rates, Figure 8 shows the effect of increasing the number of fingers on the performance of the receiver, the lower bit-error rates (BER). As shown in Figure 8, for lower values of SNR the effect of increasing the number of fingers will increase the performance of the system. By increasing the number of fingers from three to five fingers, the performance of the receiver for 10dB SNR is equal to that of 15dB SNR. In addition, the simulation results show that 8-bit quantization produces more than three times improvement over 4-bit quantization. Although the demodulation is not implemented here, it is interesting to say that 8X8-bit multiplication for the demodulator causes very small degradation in the performance of about 0.25 dB, compared to the unquantized case [30].

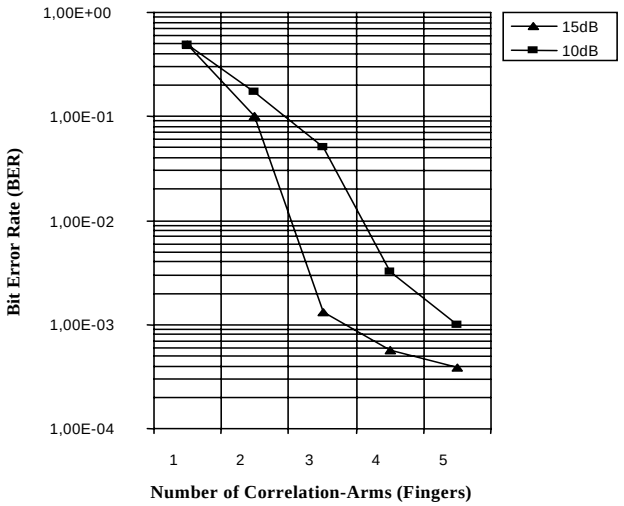
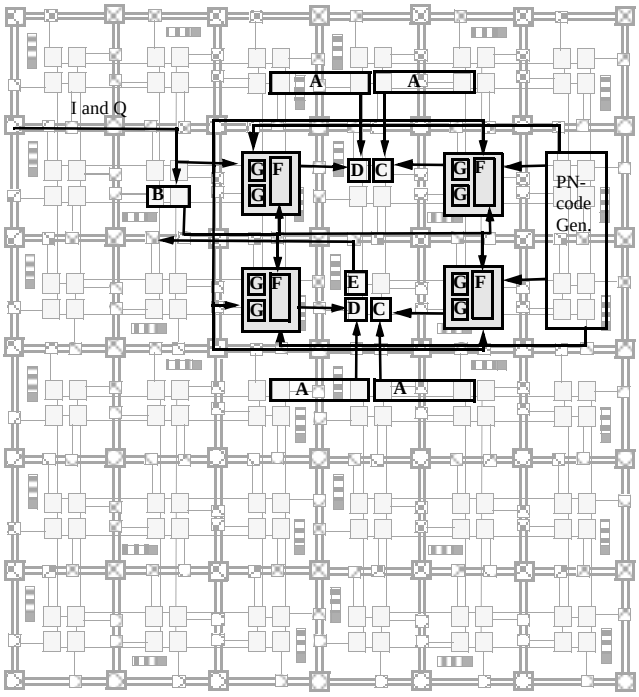


Figure 8: Bit-error-rate (BER) as function of number of fingers in the RAKE-Receiver



A: Channel tracker, which provide the gains of each finger (Ci)
B: Channel sounding allocating 2 correlators to scan the impuls response
C: Multiplier.
D: Multiplier and adder.
E: Adder.
F: RAM
G: Correlator.

1 RAKE-finger

Figure 9: Rake-Receiver example mapped to the DReAM Array Architecture.

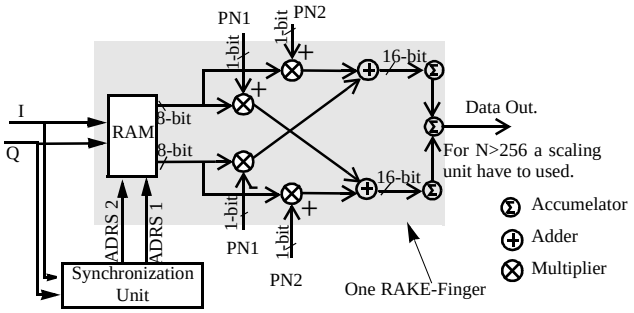


Figure 10: One Rake-Finger Implementation: ADRS1 and ADRS2 are provided by the Synchronization Unit.

Table 3 lists the performance requirements of the

Hardware Function.	**Required performance** [Mhz]	**DReAM Performance available**
Dual Port RAM	96	120
Spreader	24	120
Adder	24	120
Accumulator	72	100

Table 3: RAKE-Receiver Performance Issues

RAKE-Receiver implementation and the determined performance values of the partly synthesized DReAM architecture, For a data symbol rate of 32 Msymbols/s, i.e. one symbol (8-bit) is arriving every 0.125 microseconds. It can be seen that DReAM can provide an acceptable performance for such high future data rates. One possible mapping of the front-end of the CDMA-based RAKE-Receiver onto the dynamically reconfigurable DReAM hardware array is provided in Figure 9.

5 Conclusions

The paper presented an overview of the challenges in realizing flexible microelectronic system solutions for future mobile communication applications. We emphasize on the architecture design of flexible *system-on-a-chip* (SoC) solutions in the digital baseband processing for mobile radio devices. This paper introduced a new coarse-grained dynamically reconfigurable architecture (DReAM), including its potential for SoC-solutions in adaptive air interface candidate systems for future generations of wireless communication systems. The hardware architecture structure of DReAM was discussed in detail. DReAM is tailored to future mobile signal processing, providing an acceptable trade-off between flexibility and application performance requirements. The goal is to map those application parts onto the reconfigurable architecture, which are too complex for the DSPs, especially for low power DSPs with reduced clock rates, but which still require different types of flexibility, so that ASIC implementations would be also not sufficient. Future CDMA-based mobile communication systems have a huge potential for such application parts, whereas the advantage of their risk minimized and flexible implementation is very important, especially in the case of late specification changes (standards!). In addition, new promising design strategies like IP-based design show the potential for efficient CAD development tools, supporting low cost product development with short *time-to-market*.

The usefulness of the introduced approach was demonstrated by analyzing and implementing a computation-intensive mobile signal processing algorithm e. g. a CDMA-based RAKE-receiver algorithm, onto the DReAM architecture, resulting in providing the required performance and flexibility trade-offs. Power analysis / optimization of DReAM and automatized application mapping will be the next steps of this project.

References

[1] P. Athanas, A. Abbot: Real-Time Image Processing on a Custom Computing Platform, IEEE Computer, vol. 28, no. 2, Feb. 1995.

[2] R. W. Hartenstein, J. Becker et al.: A Novel Machine Paradigm to Accelerate Scientific Computing; Special issue on Scientific Computing of Computer Science and Informatics Journal, Computer Society of India, 1996.

[3] H. Erben, K. Sabatakakis: Advanced software radio architecture for 3rd generation mobile systems., Vehicular Technology Conference, 1998. VTC 98. 48th IEEE Published: 1998 Volume: 2 , Page(s): 825 -829 vol.2

[4] D. Efstathio, et al.: Recent Developments in Enabling Technologies for Software Radio, IEEE Comm. Mag., Aug. 1999. pp. 112-117.

[5] S. K. Knapp,: Using Programmable Logic to Accelerate DSP Functions, Xilinx, Inc. 1995.

[6] G. R. Goslin, : Using Xilinx FPGAs to Design Custom Digital Signal Processing Devices, Proc. of 1995 DSPx Technical Program, pp. 595-604.

[7] Mitola.: The software Radio Architecture., IEEE Communication Mag. , May 1995, pp. 26-38.

[8] J. Becker, A. Kirschbaum, F.-M. Renner, M. Glesner: Perspectives of Reconfigurable Computing in Research, Industry and Education; 8th International Workshop On Field Programmable Logic And Applications, FPL´98, Tallinn, Estonia, August 31-Spetmber 3, 1998, Lecture Notes in Computer Science, Springer Press, 1998

[9] A. Kirschbaum, J. Becker, M. Glesner: Run-Time Monitoring of Communication Activities in a Rapid Prototyping En-

vironment; Proc. of 9th IEEE Int´l Workshop on Rapid System Prototyping (RSP´98), Leuven, Belgium, 1998.

[10] A. Kirschbaum: Ein Rapid-Prototyping-Verfahren zum Entwurf von Kommunikationsarchitekturen in eingebetteten Systemen, Dissertation, Technische Universität Darmstadt , Germany, 1998

[11] T. Hollstein, J. Becker, A. Kirschbaum, M. Glesner, M.: HiPART: A New Hierarchical Semi-Interactive HW-/SW Partitioning Approach with Fast Debugging for Real-Time Embedded Systems; Proc. of the 6th Int. Workshop on Hardware/Software Codesign, Seattle, USA, 1998.

[12] S. Kumar, S. Nanda, "High Data-Rate Packet Communications for Cellular Networks Using CDMA: Algorithms and Performance", IEEE J. Select. Areas Commun., vol. 17, pp. 472-485, March 1999

[13] A. Chockalingam, M. Zorzi. Energy Efficiency of Media Access Protocols for Mobile Data Networks, IEEE Trans. Comm., vol. 46, pp. 1418-21, Nov. 1998.

[14] M. Zorzi: Energy Management in Personal Communications and Mobile Computing, IEEE Personal Communications, vol. 5, June 1998.

[15] A. Chockalingam and L. B. Milstein. Open-Loop Power Control Performance in DS-CDMA Networks with Frequency Selective Fading and Non-Stationary Base Stations, Wireless Networks, vol. 4, no. 3, pp. 249-261, 1998.

[16] A. Chockalingam, M. Zorzi, L. B. Milstein, and P. Venkataram, Performance of a Wireless Access Protocol on Correlated Rayleigh-Fading Channels with Capture, IEEE Trans. Communications, vol. 46, no. 5, pp. 644-655, May 1998.

[17] P. Venkataram, A. Roy, and A. Chockalingam, ‘‘Performance of a Link Control Protocol for Local Wireless Multimedia Communications,’’ IEEE GLOBECOM’98, Sydney, November 1998.

[18] J. Mikkonen, J. Aldis, and G. Awater, "The Magic WAND - Functional Over-view, IEEE J. Select. Areas Commun., vol. 16, pp. 953-971, Aug. 1998.

[19] J.-C. Chen, K. M. Sivalingam, P. Agrawal, and R. Acharya, On Scheduling of Multimedia Services in a Low-Power MAC for Wireless ATM Networks," in Proc.IEEE PIMRC ’98, (Boston, MA), Sept. 1998.

[20] J-C. Chen, K. M. Sivalingam , P. Agrawal, and S. Kishore, "A Comparison of MAC Protocols for Wireless Local Networks Based on Battery Power Consumption", IEEE INFOCOM , (San Francisco, CA), Apr. 1998.

[21] S. Kishore, P. Agrawal, K. M. Sivalingam and J-C. Chen, "MAC Layer Scheduling Strategies during handoff for wireless multimedia information networks", In IEEE International Conference on Personal Wireless Communications (ICPWC), (Mumbai, India), Dec. 1997.

[22] T. Hollstein, J. Becker, A. Kirschbaum, M. Glesner: DICE - An Interactive Approach to Hardware/Software Co-Design of Heterogeneous Real-Time Systems; Proc. of the Baltic Electronic Conference, Tallinn, Estonia, 1998.

[23] M. Glesner, J. Becker, T. Hollstein, A. Kirschbaum, S. Ortmann: Hardware/Software Co-Design für Eingebettete Systeme in der Informationstechnik; thema FORSCHUNG 1/99, TU Darmstadt, Verlag fuer Marketing und Kommunikation, Januar 1999

[24] C. Ajluni,:Redefining EDA In The New Age of Intellectual Property, it Electronic Design, pp. 64-75, Jan. 1998.

[25] Y. Zorian, R. K. Gupta,: Design and Test of Core-Based Systems on Chips, it IEEE Design & Test of Computers, pp. 14-25, Oct. - Dec. 1997.

[26] B. Tuck, Integrating IP blocks to create a system-on-a-chip, it Computer Design, pp. 49-62, Nov. 1997.

[27] David Nicklin :Utilising FPGAs in Re-configurable Basestations And Software Radios, Xilinx Inc. Electronic Eng. Mag.

[28] Gregory Ray Goslin: A Guide to Using Field Programmable Gate Arrays (FPGAs) for Appliccation-Specific Digital Signal Processing Performance, Xilinx Inc. 1995.

[29] Neil W. Bergmann et al.: Comparing the Performance of FPGA-Based Custom Computers with General-Purpose Computers for DSP Application, proc. of IEEE workshop on FPGAs for Custom Computing Machines, 1994, pp.164-171.

[30] Stephen D. Lingwood, et al. ”ASIC Implementation of a Direct-Sequence Spread-Spectrum RAKE-Receiver.”, IEEE 44th Vehicular Technology Conference, 1994.

[31] Xilinx Corp.: http://www.xilinx.com/products/virtex.htm.

[32] Jhong Sam Lee, et al.,.: CDMA Systems Engineering Handbook.., Artech House, Boston. 1998.

[33] Peter Jung, Joerg Plechinger., “M-GOLD: a multimode basband platform for future mobile terminals”,CTMC'99, IEEE International Conference on Communications, Vancouver, June 1999.

[34] Ken Chapman, “Fast integer multipliers fit in FPGAs.” EDN magazinee’s Desighn Ideas, www.ednmag.com, March 1993.

[35] Tero Ojanpera, et. al.,: Wideband CDMA for Third Generation Mobile Communicatios., Artech House Pub., 1998.